Computing and Informatics, Vol. 40, 2021, 1–28, doi: 10.31577/cai_2021_1_1

UNSUPERVISED ADAPTATION FOR HIGH-DIMENSIONAL WITH LIMITED-SAMPLE DATA CLASSIFICATION USING VARIATIONAL AUTOENCODER

Mohammad Sultan Mahmud, Joshua Zhexue Huang*
Xianghua Fu, Rukhsana Ruby, Kaishun Wu

College of Computer Science and Software Engineering
Shenzhen University, Shenzhen 518060, China
&
National Engineering Laboratory for Big Data System Computing Technology
Shenzhen University, Shenzhen 518060, China
e-mail: {sultan, zx.huang, fuxh, ruby, Wu}@szu.edu.cn

Abstract. High-dimensional with limited-sample size (HDLSS) datasets exhibit two critical problems: (1) Due to the insufficiently small-sample size, there is a lack of enough samples to build classification models. Classification models with a limited-sample may lead to overfitting and produce erroneous or meaningless results. (2) The 'curse of dimensionality' phenomena is often an obstacle to the use of many methods for solving the high-dimensional with limited-sample size problem and reduces classification accuracy. This study proposes an unsupervised framework for high-dimensional limited-sample size data classification using dimension reduction based on variational autoencoder (VAE). First, the deep learning method variational autoencoder is applied to project high-dimensional data onto lower-dimensional space. Then, clustering is applied to the obtained latent-space of VAE to find the data groups and classify input data. The method is validated by comparing the clustering results with actual labels using purity, rand index, and normalized mutual information. Moreover, to evaluate the proposed model strength, we analyzed 14 datasets from the Arizona State University Digital Repository. Also, an empirical comparison of dimensionality reduction techniques shown to conclude their applicability in the high-dimensional with limited-sample size data settings. Experimental results demonstrate that variational autoencoder can achieve more accuracy than traditional dimensionality reduction techniques in high-dimensional with limited-sample-size data analysis.

* Corresponding author

Keywords: HDLSS problem, dimensionality reduction, unsupervised framework, variational autoencoder, deep learning

Mathematics Subject Classification 2010: 68-T99

1 INTRODUCTION

By essence, in many domains, including computational biology, bioinformatics, ecology, geology, neuroscience datasets are characterized by a small number of samples N (records), but a large number of features p (dimensions). These datasets are called the high-dimensional limited-sample size (HDLSS) dataset (*aka* 'fat' dataset), often written as $p \gg N$. HDLSS data classification and clustering both are crucial and challenging tasks in data mining and machine learning. High variance and bias are the main concern for HDLSS data analysis. As a result, simple and highly-regularized classification and regression techniques often become the method of choice [1].

The caution of insufficiently small-sample size has been flagged, especially dangerous to draw conclusions from the limited-sample dataset [2, 3, 4]. In HDLSS datasets, typically sample size is too small to allow for the split into train-test testing or k-fold cross-validation. However, data miners train a classifier model and estimate the classification accuracy. It can be challenging to build a stable and reliable classifier and draw a conclusion from such limited-samples.

The difficulty occurs when dealing with high-dimensional data, where the accuracy of classifiers or clustering algorithms tends to deteriorate are often referred to as the curse of dimensionality [5, 6]. Consequently, dimensionality reduction (DR) is an innovative and important tool in the fields of data analysis, data mining, and machine learning. Several techniques have been proposed for DR such as principal components analysis (PCA) [7, 8, 9], independent components analysis (ICA) [10], factor analysis (FA) [11], multidimensional scaling (MDS) [12], and non-negative matrix factorization (NMF) [13]. Traditional methods like PCA, ICA, FA, and classical MDS suffer from being based on linear models.

However, recently, some nonlinear dimensionality reduction (NLDR) (*aka* 'manifold learning') methods have been developed and have become a popular topic. Traditional DR methods PCA, ICA, FA, and TSVD usually require sufficient data, otherwise, they might be less effective. In the context of $N > p$, there is a relatively large application of PCA, ICA, FA, and TSVD. In the case of $p \gg N$, transformed lower-dimension (d) is lower than or equal to sample size ($d \leq N$), there is difficulty to preserved information about the original data in such too lower-dimensional space. Therefore, for the HDLSS problem ($p \gg N$), it is clear that the basic formulation of PCA, ICA, FA, and TSVD does not work.

Over the decades, deep learning (DL) has succeeded in a variety of fields to extract information from high-dimensional data such as image, speech, text, and

vision [14, 15]. The limitation of DL is getting a large number of training data to ensure learning accuracy. Different types of deep learning architecture have been proposed to solve the problem of insufficient samples [16, 17]. Recently, unsupervised deep learning models such as generative adversarial net (GAN) and variational autoencoder (VAE) have shown the modeling power without the labels. VAEs harness to generate 'blurry' data compared with other generative models, also more stable to train [18]. Moreover, unlike many existing techniques (e.g., PCA, ICA, FA), VAE also capable of reducing the dimension as necessary from the high-dimensional space.

Recently, we examined that VAE based dimensionality reduction outperforms PCA, fastICA, FA, NMF, and LDA in HDLSS data classification in a supervised model [48]. It is also addressed that classifiers and obtained reduced dimensions show inconsistent behavior w.r.t classification accuracy and vary considerably. This discrepancy raises the supervised framework applicability to HDLSS data analysis, yet critical. Although there are varieties of classification algorithms, the challenge is an appropriate selection in the application of the limited-sample domain. Hence, it is more advantageous to use an unsupervised framework. We favored an idea of the unsupervised model, in [19]. It is noteworthy to mention that this paper is an extension of our work reported at the 4th International Conference on Advanced Robotics and Mechatronics [19].

This study manifests an extensive empirical analysis of traditional DR techniques and the effectiveness of the approach we proposed in [19]. The proposed DR approach can maintain a reasonable size of dimensions even after the reduction, unlike many existing methods that often reduce the dimensions too heavily. The problem with a huge number of dimensions is known as the curse of dimensionality, but we argue that there is a blessing of dimensions as well in the sense that we often need a reasonable size of dimensions for useful data analysis. The proposed DR approach can maintain a reasonable-size of dimensions and utilize the blessing of dimensions in the HDLSS setting.

The contribution of this study is to present an unsupervised framework for HDLSS data classification. In particular, we employed the deep learning technique variational autoencoder for dimensionality reduction, and the clustering is applied on the obtained latent variables (low-dimensional space) to group data, and then validated clustering results with the original class labels. We tested the effectiveness of the proposed framework in varieties types of fourteen HDLSS datasets, such as biology, image, mass, and spectrometry, and comparisons with various reduced dimensions in classification are also shown. Moreover, we provided an empirical comparison of different dimensionality reduction methods, compare their performances on a wide range of challenging HDLSS datasets, and conclude their applicability to HDLSS application.

The paper is organized as follows. Section 2 surveyed related works on dimensionality reduction of HDLSS data analysis. In Section 3, the idea of the proposed method is described. Empirical comparisons and concluding remarks are in Sections 4 and 5, respectively.

2 RELATED WORK

2.1 State-of-the-Art Data Dimensionality Reduction Techniques

HDLSS data analysis is vital for scientific discoveries in many areas. When dealing with HDLSS data, the overfitting and high-variance gradients are the main challenges in majority models. In the past, significant work has been done on HDLSS asymptotic theory, where the sample size N is fixed or $N/p \to 0$ as the data dimension $p \to \infty$ [20, 6]. In the HDLSS context, Jung and Marron explored several types of geometric representations and showed inconsistent properties of the sample eigenvalues and eigenvectors [21].

In past decades, numerous dimensionality reduction (DR) techniques, including PCA [7, 8, 9], ICA [10], FA [11], MDS [12], NMF [13] proposed. PCA is perhaps one of the oldest and best-known DR methods in high-dimensional data processing and mining. Traditional methods like PCA, ICA, FA, and classical MDS suffer from being (based on) linear models. Recently, to discover the intrinsic manifold structure of the data, nonlinear DR algorithms are developed, such as locally linear embedding (LLE) [22], kernel PCA (KPCA) [23], sparse PCA (SPCA) [24], and spectral embedding (SE) [25]. DR methods can be roughly categorized into supervised and unsupervised. Semi-supervised DR is recognized as a new issue in semi-supervised learning, which learns from a combination of both labeled and unlabeled data. Table 1 presents a summary of canonical DR methods to clarify their characteristic in HDLSS ($p \gg N$) settings.

Supervised or classification methods are often used for HDLSS data analysis. Most achievements in the supervised model show that more samples and lower-dimension can improve the performance of classifiers. However, sufficient large-samples are essential to building a classification model with good generalization ability, expected that perform equally well on the training and independent testing dataset. Consequently, the classification technique does not suit with small-sample size dataset, to avoid overfitting (training and validation data), the unsupervised (that is, clustering) methods also applied for HDLSS analysis.

Many researchers considered PCA in the classification and clustering of biological data in the context of HDLSS, among them are [26, 27, 28, 29]. In fact, PCA reduces the dimensionality of the data linearly, and it may not extract some nonlinear relationships of the data. In the same vein, [30, 31] pointed that though many researchers considered PCA as a DR method, it is even more useful for data visualization in high-dimensional contexts. NMF is another widely used tool for high-dimensional data analysis. NMF has also been applied for gene clustering, microarray and protein sequence data analysis, and recognition [32, 33]. PCA is deterministic while NMF is stochastic, so NMF appears to be more suitable for HDLSS data analysis than PCA. In [48], explored that PCA, ICA, FA, LDA, MBDL, and NMF are not efficient for dimensionality reduction in HDLSS data classification.

For decades, deep learning (DL) techniques have achieved state-of-the-art performance with large-sample sizes in many domains. Nevertheless, recently, few efforts

have been devoted to applying DL to the HDLSS settings by [34, 53, 15]. DLs also suffer overfitting on HDLSS problems. The 'Dropout' method was proposed to prevent overfitting by reducing the parameters of the full-connection layer, for detail see [35, 16]. Also, a transfer learning-based deep convolutional neural network (CNN) has been developed to solve the problem of the small-sample dataset [17]. In the last few years, a variety of supervised and semi-supervised deep learning models has blossomed in the context of natural language processing (NLP). Recently, there have few efforts to develop unsupervised learning techniques by building upon variational autoencoders [36, 37, 38].

Algorithm	Method	Degrees of Freedom
Principal components analysis (PCA) [7, 8, 9]	LDR	$d \leq N$
Independent components analysis (fastICA) [10]	LDR	$d \leq N$
Factor analysis (FA) [11]	LDR	$d \leq N$
Truncated SVD (*aks* LSA) [39]	LDR	$d \leq N$
Latent Dirichlet allocation (LDA) [40, 41]	NLDR	$d < p \star$
Mini-batch dictionary learning (MBDL) [42]	NLDR	$d < p \star$
Non-negative matrix factorization (NMF) [13]	NLDR	$d < p \star$
Kernel PCA (KPCA) [23]	NLDR	$d \leq N$
Sparse PCA (SPCA) [24]	NLDR	$d < p \star$
Locally linear embedding (LLE) [22]	NLDR	$d < N$
Spectral embedding (SE) [25]	NLDR	$d < N$
Multidimensional scaling (MDS) [12]	NLDR/LDR	$d < p \star$
Autoencoder (AE) [43, 45]	NLDR/LDR	$d < p \star$

Degrees of freedom is possible computed number of latent variables

$\star$ indicates succeed at most are desired to keep dimension

Table 1. A summary of most known and used dimensionality reduction techniques in the HDLSS setting. N: number of the samples, d: dimensionality of the latent space, p: dimensionality of the data space, LDR: linear dimensionality reduction; NLDR: nonlinear dimensionality reduction.

2.2 Variational Autoencoder (VAE) Model

Kingma and Welling [43] introduced the VAE, which is based on the autoencoding framework as a latent variable generative model (see Figure 1). VAE can discover nonlinear explanatory features through data compression and nonlinear activation functions. A traditional autoencoder (AE) consists of an encoding and a decoding phase where input data is projected into lower-dimensions and then reconstructed. AE is deterministic and trained by minimizing reconstruction error. In contrast, VAE is stochastic and learns the distribution of explanatory features over samples. VAE achieves these properties by learning two distinct latent representations: a mean (μ) and standard deviation (σ) vector encoding. The model adds a Kullback-Leibler (KL) divergence term to the reconstruction error, which also regularizes

weights by constraining the latent vectors to match a Gaussian distribution [44]. In a VAE, these two representations are learned concurrently through the use of a reparameterization trick that permits a backpropagated gradient. Importantly, projected data onto an existing VAE feature space enabling new data to be assessed. In this, we aim to build a VAE that compresses high-dimensional features and reveals a relevant latent space.

A VAE performs density estimation on $p(x, z)$ where z are latent variables, to maximize the likelihood of the observed data x, where $x_i \in X \subset \mathbb{R}^m$ is the i^{th} observation: $\log p(X) = \sum_{i=1}^{N} \log p(x_i)$.

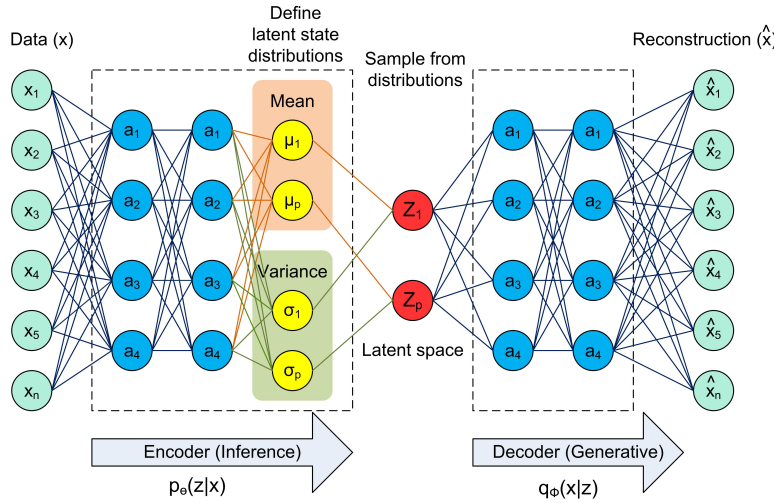


Figure 1. Variational autoencoder (VAE) framework [43, 45]

A VAE consists of an encoder, a decoder, and a loss function unit.

The encoder is a neural network, compresses data x into a latent space z. Encoder's transformed representation is d-dimensional, which is much small than the original p-dimensions. The lower-dimensional space is stochastic; encoder output parameter is $p_\theta(z|x)$, which is a Gaussian probability density. Encoder weight and bias parameter is θ.

The decoder is another neural network, gets input as latent representation z and output the parameters of a probability distribution of the data. Its weight and bias parameter is ϕ. Decoder reconstructs the data is denoted by $q_\phi(x|z)$. It goes from a smaller to a larger dimension. Information loss computed using the reconstruction log-likelihood $\log q_\phi(x|z)$. This measure states how effectively decodes the z into N real-valued numbers. VAE uses a decoder-based generative model as

$$p(x, z) = p(x|z)p(z),$$
$$p(z) = \mathcal{N}(z; 0, 1).$$

The loss function of the VAE is the negative log-likelihood with a regularizer. Since the marginal likelihood is difficult to work with directly for non-trivial models, instead a parametric inference model $p(x|z)$ is used to optimize the variational lower-bound on the marginal log-likelihood

$$\mathcal{L}(x, \theta, \phi) = -\mathbb{E}_{q(z|x_i)}[\log q_\theta(x_i|z)] + \mathbf{KL}(p_\theta(z|x_i)\|p((z)). \tag{1}$$

In Equation (1), the first term of $\mathcal{L}$ is reconstruction error or expected negative log-likelihood of the i^{th} data point of the decoder. The second term $\mathbf{KL}(.\|.)$ is a regularizer, the KL-divergence between the encoder and decoder distribution, to minimize the KL-divergence from a chosen prior distribution.

3 METHOD

This study proposes an unsupervised adaptation for HDLSS data classification, which aims to exclusively apply a generative model variational autoencoder (VAE) to investigate dimensionality reduction ability on the HDLSS dataset. In this framework, we divided an unlabeled HDLSS dataset into groups based on the hidden properties of the data. However, conventional classification techniques cannot cope with this HDLSS dataset due to insufficient sample size to build and test a classifier or cross-validation. The proposed unsupervised scheme for HDLSS data classification is illustrated in Figure 2.

Figure 2. Proposed framework

Consider $\mathbb{D} = [X, Y] = [(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k), \ldots, (x_N, y_N)]$ be a $N \times p$ data matrix, where $p \gg N$; where p and N are the number of features and samples, respectively. $x_i \in X$ is the i^{th} observation and the class label is $y_i \in Y$ belonging to C classes. X is mapped a choice of p-dimensional onto a d-dimensional representations $\mathbf{Z}$, $z_i = (z_1, z_2, \ldots, z_d)$, where $d < p$, such that the transformed lower-dimensional representations $z_i = \mathbf{Z}^T x_i$ can preserve the information of the original data. The key aspects of the framework are as follows:

Dimensionality reduction: The first step of the proposed framework is the dimensionality reduction, which receives the HDLSS dataset as input, and then class labels are removed from the dataset. Consequently, the deep learning model VAE is applied to the unlabeled data to project desirable high-dimensional data onto lower-dimensional space. The deep learning technique VAE empowers the method to avoid overfitting.

Clustering: Then, the clustering technique is applied to the obtained transformed low-dimensional space to find the data groups. The exploratory and unsupervised learning nature of the clustering demands efficient to use that would benefit from the combination in the strength of the framework. A clustering technique groups similar data in a cluster, whereas dissimilar data in different clusters. K-means is widely used and one of the prominent data mining techniques for its simplicity. In this study, simple K-means clustering is used. Determining the number of clusters in a dataset is fundamental in K-means clustering, which requires the user to specify the number of clusters K to be generated. There are different methods for identifying the optimal number of clusters in a dataset, including DBSCAN, Xmeans, I-Nice, Elbow, Silhouette, Gap statistic.

Decision making: The last step of the proposed framework is decision making, which validated the clustering results with the original class labels. Assume that sample points from one class form clustered in the same group.

4 EXPERIMENT AND DISCUSSIONS

4.1 Datasets for Experiments

The experiments were conducted on 14 high-dimensional limited-sample size $p \gg N$ datasets obtained from the Arizona State University repository[1]. Table 2 presents the detail of the datasets.

4.2 Experiment Settings

Experiments were designed for the empirical study of DR techniques on the HDLSS dataset. We applied two types of experiments:

1. without dimensionality reduction (WDR), which ensures that all the original features were used for classification, and
2. with dimensionality reduction (DR), where original data space mapped into a new space with a much smaller number of dimensions were used for classification.

In this study, different choices of latent-space were investigated (i.e., 2, 10, 20, 50, 100, 150, 200, 250, ..., 500) to see how the dimensionality of the projected space affects the performance. To evaluate the effectiveness of dimensionality reduction various DR methods were applied, such as VAE, AE, PCA, Kernel PCA, LLE, MDS, Sparse PCA, NMF, Truncated SVD, SE.

Computations were performed using machines with x64-based processor Intel(R) core i7-7700, CPU 3.60 Hz, and 8.0 GB memory. VAE code implementation using the CPU based on Tensorflow and Keras libraries.

[1] `http://featureselection.asu.edu/`

ID	Dataset	Abbrev.	N	p	c	Type
1	ALLAML	ALL	72	7 129	2	continuous, binary
2	CARCINOM	CAR	174	9 182	11	continuous, multi-class
3	CLL_SUB_111	CLL	111	11 340	3	continuous, multi-class
4	GLI_85	GLI	85	22 283	2	continuous, binary
5	GLIOMA	GMA	50	4 434	4	continuous, multi-class
6	NCI9	NCI	60	9 712	9	discrete, multi-class
7	PROSTATE_GE	PROS	102	5 966	2	continuous, binary
8	SMK_CAN_187	SMK	187	19 993	2	continuous, binary
9	TOX_171	TOX	171	5 748	4	continuous, binary
10	ORLRAWS10P	ORL	100	10 304	10	continuous, multi-class
11	PIXRAW10P	PIX	100	10 000	10	continuous, multi-class
12	WARPAR10P	WPAR	130	2 400	10	continuous, multi-class
13	WARPPIE10P	WPIE	210	2 420	10	continuous, multi-class
14	ARCENE	ARC	200	10 000	2	continuous, binary

Table 2. Characteristics of the datasets. ID 1–9 are biological, 10–13 are face image, and 14 is mass-spectrometry dataset (N: number of samples, p: number of features, and c: number of classes).

4.3 VAE Design

For the structure of the VAE, we exhaustedly investigated the best setting, such as the number of intermediate layers, the size of each intermediate layer, batch size, and learning rates. It is found that the network structure of VAE also affects the performance of the feature extraction. In the experiment, VAE is performed on the single intermediate layer (encode) with the following architecture: input encoded onto d-dimensional latent space ($d = z = 2, 10, 20, 50, 100, 200, \ldots, 500$) and reconstructed back to the original dimension. We kept the intermediate dimension as 10 % of the original data space. The network parameter optimized with an 'adam' optimizer, included 'rectified linear units' and batch normalization in the encoding stage, and 'sigmoid' activation in the decoding stage. A parameter scope is performed on batch size 50, 100, 150, and 200; epochs 100, 200, and 300; learning rates 0.005, 0.001, 0.0015, and 0.0025; and warmups (k) 0.01, 0.05, 0.001, and 0.0005. k controls how much the KL-divergence loss contributes to learning. In general, training was relatively stable for many parameter combinations. Ultimately, the best parameter combination based on validation was batch size 100, learning rate 0.0005, and epochs 200. Training stabilized after about 120 epochs.

4.4 Determining Number of Clusters in Dataset

A large variety of clustering methods has been proposed to discover the inherent cluster structure in data. DBSCAN [46], Xmeans [47], and I-nice [49] are popular methods for determining the number of clusters, K. We use these three methods to

determine the K-value for the clustering of this study. Table 3 presents the obtained number of clusters of the algorithms. Results showed that the DBSCAN is inefficient when applied to large-dimensional data. To determine the K value for K-means, we assumed that the number of classes is equal to the number of clusters.

4.5 Evaluation Criteria

The attained results were analyzed in terms of three external cluster evaluation measures: purity [50], rand index (RI) [51], and normalized mutual information (NMI)[52]. Purity is the percent of the total number of objects classified correctly, it is calculated as follows:

$$\text{Purity} = \frac{1}{N}\sum_{1}^{K} \mathbf{max}_j |C_i \cap Y_j| \tag{2}$$

where N is number of objects in the dataset, K is number of clusters, C_i is a cluster in C, and Y_j is the classification which has the **max** count for cluster C_i.

Rand index (RI) is another popular cluster validation index, measures the percentage of correct decisions, it can be defined as Equation (3).

$$\text{RI} = \frac{TP + TN}{TP + FP + FN + TN} \tag{3}$$

where TP and FP are the numbers of true positive and false positive, whereas TN and FN are the numbers of true and false negative, respectively.

Normalized mutual information (NMI) is the mutual information between the clustering and the classification on the shared object membership, with a scaling factor corresponding to the number of object in the respective clusters, can define by Equation (4).

$$\text{NMI} = \frac{I(C_i, Y_i)}{[H(Y_j) + H(C_i)]/2} \tag{4}$$

where $I(C_i, Y_i)$ denotes the mutual information between true assigned class and obtained cluster label, and $H(C_i)$ is the entropy of cluster C_i while information about Y_j classes is available. The range of NMI is between $[0, 1]$. A higher value indicates a better quality of clustering.

4.6 Analysis and Discussions

The attained results were analyzed in terms of the average on different used dimensions. The results of each selected dimension were obtained by the best of 5 runs. Tables 4, 5, and 6 show the summarized achieved averaged values of different dimensionality reduction (DR) techniques for each experimental dataset's purity, RI, and NMI, respectively.

	DBSCAN										I-nice										Xmeans									
Dataset	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE
ALL	1/2	1/1	1/1	1/1	1/3	1/1	1/1	1/2	1/2	1/2	2/3	2/3	2/2	2/2	2/3	2/4	2/3	2/2	2/3	2/3	2/2	2/2	2/3	2/3	2/3	2/3	2/2	2/3	2/2	2/2
CAR	2/8	1/7	1/4	1/2	2/5	1/8	1/7	3/10	2/8	2/8	7/10	7/10	4/8	2/5	5/11	8/12	7/9	10/12	8/11	8/10	6/10	3/5	8/12	2/2	4/12	3/4	2/3	4/11	2/3	6/9
CLL	1/2	1/2	1/4	1/1	1/2	1/2	1/2	1/2	1/2	1/2	2/4	3/5	2/4	2/4	2/5	2/5	2/5	2/4	2/3	2/3	2/4	2/3	2/5	2/2	3/4	2/3	2/2	2/5	2/3	2/3
GLI	1/3	1/3	1/4	1/1	2/6	1/2	1/1	2/3	1/3	2/3	2/4	2/3	2/3	2/4	2/4	2/3	2/4	2/3	2/3	2/3	2/3	2/2	2/2	2/2	2/4	2/2	2/2	2/5	2/2	2/4
GMA	1/2	1/1	1/3	1/2	1/4	1/2	1/1	1/2	1/1	1/2	3/4	2/3	2/4	3/5	3/5	3/5	2/4	2/4	3/4	3/5	2/5	2/2	2/3	2/5	2/7	2/2	2/2	2/3	2/3	2/5
NCI	1/3	1/2	1/2	1/2	1/3	1/2	1/2	1/2	1/2	1/3	6/9	5/8	6/8	7/9	8/10	6/8	7/9	6/9	6/8	6/10	2/5	2/3	2/4	2/2	2/4	2/2	2/2	2/4	2/4	2/5
PROS	1/2	1/1	2/6	1/2	1/3	1/3	1/2	1/2	1/1	1/2	2/4	2/4	2/4	2/5	2/5	2/4	2/5	2/3	3/3	3/4	2/4	2/4	2/6	2/2	2/9	2/4	2/3	2/6	2/3	3/5
SMK	1/2	1/1	1/2	1/5	1/5	1/5	1/1	1/2	1/1	1/2	2/4	2/4	2/4	2/4	2/3	2/4	3/5	2/3	2/4	2/4	2/4	2/3	2/4	2/3	2/5	3/4	2/3	2/5	2/3	3/6
TOX	1/2	1/6	1/1	1/1	1/7	1/4	1/7	1/8	1/6	1/2	3/5	3/5	4/6	2/4	4/6	3/7	3/5	3/5	3/5	3/5	2/5	2/3	3/5	2/2	2/3	3/4	2/2	2/5	2/2	2/3
ORL	2/9	1/7	1/12	1/7	1/8	2/11	3/7	8/9	1/7	3/10	7/11	7/9	7/12	6/11	7/11	8/12	6/11	8/11	7/11	8/11	2/6	2/3	2/10	2/2	6/16	5/7	2/3	2/5	2/3	5/8
PIX	3/10	3/10	2/6	1/8	3/10	7/11	3/10	6/10	4/10	4/8	6/11	6/11	5/8	6/9	6/11	6/12	6/11	7/11	7/11	7/10	4/12	2/9	2/13	3/4	3/4	7/14	3/10	2/14	4/7	7/13
WPAR	2/8	1/9	1/12	1/6	2/9	7/8	1/9	7/10	2/10	4/7	6/9	5/8	5/8	5/9	5/9	5/9	5/9	6/9	5/8	6/9	2/5	2/3	2/3	2/2	2/5	2/3	2/2	2/2	2/2	2/4
WPIE	2/8	1/17	1/17	1/5	1/7	1/6	7/16	1/3	1/8	3/9	7/9	6/9	7/11	4/6	6/8	6/9	6/11	8/9	6/9	8/9	2/8	2/3	2/2	3/5	6/17	3/5	2/3	3/17	2/5	2/7
ARC	1/2	1/5	3/6	1/3	3/4	1/5	1/4	1/7	1/5	1/2	2/4	3/5	3/5	3/5	3/4	3/5	3/5	3/3	3/5	2/3	2/7	3/7	3/16	3/3	3/13	3/8	2/9	7/12	4/7	2/8

Table 3. Summarized results of three cluster number determination methods on each dataset. The values shown in the table are minimum and maximum number of clusters obtained over the applied different number of dimensions, i.e., 2, 10, 20, 50, 100, 200, 300, 400, 500 (min/max).

Table 4 shows the achieved average purity of different DR techniques for each experimental dataset; it is observable that VAE outperformed others. Considering all the experiments, VAE (1.4), AE (3.2), and SPCA (3.6) were ranked from first to third, respectively. KPCA (5.1) and TSVD (5.3) were ranked as fourth and fifth ranks subsequently. It reveals the strength of VAE, preserve more information in lower-dimensional space in the context of HDLSS.

Besides, the average RI of each algorithm over all experiments is listed in Table 5. Among the techniques, VAE (1.3), AE (2.6), and SPCA (3.9) were ranked from first to third in terms of the correct decision. Nonetheless, the superior RI of VAE shows that not only it copes with the HDLSS but also outperforms traditional DR techniques such as LLE, MDS, PCA, KPCA, NMF, SE, TSVD. Based on the reported results in Table 6, VAE (2.3) is ranked as the most normalized mutual information measure. AE (3.1) is ranked second, and the third rank is assigned to SPCA (3.9).

Based on the observations from Tables 4, 5, and 6, it can be seen VAE is robust against the HDLSS dataset. AE, SPCA, and KPCA also perform well, while traditional DR techniques PCA, NMF, LLF, MDS, TSVD, and SE provide quite poor performance.

To assess the importance of the projected space size in the HDLSS problem, we can examine in a little more detail the performances of the 14 datasets with different numbers of dimensions, as shown in Appendix A (Tables A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14). From observation, it is therefore of interest to note that the performance gained with the raise of dimension size. It is impressive that VAE almost always achieved the highest accuracy in used different reduced dimensions. Moreover, it seems that SPCA and MDS are affected by the size of dimensionality and stable for a wide range of dimensions, while other methods (i.e., PCA, KPCA, LLE, SE) typically require relatively more dimensions to obtain good accuracy. It is worthwhile to mention that the use of VAE, AE, SPCA, and KPCA is advantageous compared to other techniques, they can preserve more information in possible higher-dimensional space. Though, analyzing in lower-dimensional space is much easier than in a higher-dimensional space. Noted that 7 out of 14 datasets (i.e., 1, 2, 4, 5, 9, 10, and 11) were provided the best results where respective dimensions $d > N$. So, it is reasonable to try to more latent space concerning the preservation of information that increases the chances of obtaining useful results. Thus, it can be concluded that VAE is providing the best DR for the unsupervised HDLSS data classification in this study; also, AE, SPCA, KPCA can be a competitive choice.

One major limitation of this framework is that when the data has a complex distribution (each class has different distribution). For instance, we assumed the number of clusters is equal to the number of classes, the assumption is not valid when distinctive mini-clusters exist.

Dataset	WDR	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE
ALL	70.8	73.4 (3)	72.6 (4)	64.2 (9)	67.4 (8)	56.3 (10)	73.6 (2)	70.8 (5)	68.1 (7)	69.8 (6)	86.1 (1)
CAR	66.7	56.7 (3)	55.1 (5)	47.8 (8)	50.9 (7)	43.6 (9)	57.2 (2)	54.6 (6)	37.2 (10)	56.6 (4)	71.4 (1)
CLL	53.2	55.2 (4)	55.1 (5)	54.1 (8)	53.0 (9)	55.3 (3)	55.7 (2)	54.4 (7)	50.5 (10)	54.6 (6)	61.0 (1)
GLI	64.7	69.8 (5)	69.7 (6)	67.1 (7)	70.5 (4)	65.9 (9)	71.0 (2)	70.9 (3)	61.6 (10)	66.8 (8)	71.4 (1)
GMA	60.0	62.6 (2)	57.0 (6)	46.0 (10)	55.6 (7)	48.0 (9)	60.7 (4)	62.0 (3)	52.7 (8)	58.5 (5)	65.6 (1)
NCI	43.3	50.1 (2)	42.1 (4.5)	42.1 (4.5)	40.0 (8)	37.1 (9)	44.6 (3)	41.7 (6)	35.2 (10)	40.8 (7)	50.9 (1)
PROS	57.8	58.5 (4)	58.4 (6.5)	57.8 (8)	58.5 (4)	57.1 (9)	58.5 (4)	58.4 (6.5)	55.8 (10)	58.6 (2)	59.7 (1)
SMK	51.9	55.1 (8)	55.7 (3)	58.2 (1)	56.4 (2)	52.6 (10)	55.6 (4)	55.4 (5.5)	54.3 (9)	55.4 (5.5)	55.3 (7)
TOX	44.4	51.3 (2)	45.1 (5.5)	39.8 (9)	43.8 (7)	40.5 (8)	46.3 (4)	48.2 (3)	36.4 (10)	45.1 (5.5)	56.9 (1)
ORL	76.0	76.1 (2)	72.4 (4.5)	60.5 (8)	72.4 (4.5)	59.5 (9)	73.7 (3)	72.0 (6)	47.0 (10)	68.2 (7)	78.7 (1)
PIX	81.0	86.4 (3)	77.4 (7)	57.3 (9)	81.4 (6)	59.8 (8)	86.8 (2)	81.8 (4.5)	54.3 (10)	81.8 (4.5)	88.0 (1)
WPAR	32.3	37.4 (2)	27.5 (6)	31.2 (4)	27.2 (7)	27.1 (8.5)	27.8 (5)	23.8 (10)	31.6 (3)	27.1 (8.5)	39.1 (1)
WPIE	31.0	58.8 (2)	30.1 (7)	36.7 (4)	29.8 (9)	27.4 (10)	29.9 (8)	30.6 (5)	38.9 (3)	30.2 (6)	62.2 (1)
ARC	34.0	64.8 (3)	64.9 (2)	55.3 (9)	62.9 (7)	55.0 (10)	63.2 (6)	63.8 (4)	59.6 (8)	63.6 (5)	66.3 (1)

Table 4. Average purity (in %) of different dimensions of different techniques on datasets. Higher value is better and values in parentheses indicate the rank of algorithm.

Dataset	WDR	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE
ALL	58.1	63.9 (2)	59.7 (4)	56.3 (7)	55.9 (9)	50.5 (10)	60.6 (3)	58.4 (5)	56.1 (8)	57.7 (6)	76.8 (1)
CAR	91.2	89.3 (3)	87.7 (6)	80.3 (8)	87.0 (7)	76.5 (10)	89.4 (2)	88.0 (5)	78.4 (9)	89.0 (4)	93.3 (1)
CLL	55.3	57.6 (3.5)	57.2 (5.5)	52.2 (9)	56.6 (7)	52.4 (8)	57.6 (3.5)	57.2 (5.5)	47.5 (10)	57.7 (2)	58.7 (1)
GLI	53.8	68.2 (1)	57.3 (7)	56.1 (8)	58.7 (3)	54.6 (10)	58.5 (4)	58.3 (5)	57.4 (6)	55.7 (9)	58.8 (2)
GMA	73.1	74.6 (3.5)	73.7 (6)	57.8 (10)	70.1 (7)	59.5 (9)	74.6 (3.5)	73.8 (5)	64.8 (8)	74.7 (2)	75.4 (1)
NCI	80.7	82.9 (2)	80.9 (6)	79.8 (8)	81.0 (5)	76.6 (9)	82.7 (3)	82.6 (4)	56.4 (10)	80.0 (7)	85.1 (1)
PROS	50.7	51.0 (5)	51.0 (5)	51.2 (2)	51.0 (5)	50.9 (8.5)	51.0 (5)	50.9 (8.5)	50.4 (10)	51.0 (5)	51.6 (1)
SMK	49.8	50.7 (2)	50.4 (6.5)	51.5 (1)	50.6 (3)	50.0 (10)	50.4 (6.5)	50.3 (8.5)	50.5 (4.5)	50.3 (8.5)	50.5 (4.5)
TOX	67.9	69.1 (2)	68.2 (3)	57.8 (9)	66.0 (7)	59.1 (8)	68.1 (4)	67.8 (5)	46.2 (10)	67.7 (6)	72.2 (1)
ORL	93.6	93.4 (3.5)	93.4 (3.3)	85.9 (9)	92.7 (6)	86.5 (8)	93.5 (2)	92.8 (5)	80.7 (10)	91.9 (7)	94.3 (1)
PIX	95.1	96.4 (3)	94.2 (7)	83.0 (10)	95.2 (6)	85.6 (8)	96.5 (2)	95.4 (5)	84.7 (9)	95.5 (4)	96.8 (1)
WPAR	83.9	83.3 (2)	82.6 (4)	72.0 (10)	82.7 (3)	78.0 (8)	82.1 (6.5)	82.1 (6.5)	74.3 (9)	82.3 (5)	85.2 (1)
WPIE	82.3	87.3 (2)	83.2 (3.5)	78.2 (8)	83.2 (3.5)	76.6 (9)	82.8 (5)	82.4 (7)	74.6 (10)	82.6 (6)	90.1 (1)
ARC	54.9	53.8 (3)	54.2 (2)	50.5 (9)	53.3 (7)	50.3 (10)	53.4 (6)	53.7 (4)	51.9 (8)	53.6 (5)	55.1 (1)

Table 5. Average RI (in %) of different dimensions of different techniques on datasets. Higher value is better and values in parentheses indicate the rank of algorithm.

4.7 Run-Time

The average run-time of different applied dimensions of each dimensionality reduction method on each dataset is provided in Table 7. It clearly illustrates that AE and VAE are slower and computationally expensive than the corresponding methods for training the network, which lasted more than 2× to 3×. It can be seen that VAE is faster than AE to achieve selected (suitable) dimensionality reduction. Besides, KPCA, LLE, MDS, SE, SPCA, TSVD, NMF, and PCA were not much different in running time. Furthermore, VAE and AE consumed more run-time compared to other methods but was provided the best performance.

Dataset	WDR	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE
ALL	.090	.148 (3)	.139 (4)	.118 (6)	.078 (9)	.052 (10)	.149 (2)	.114 (7)	.081 (8)	.119 (5)	.446 (1)
CAR	.322	.590 (3.5)	.574 (6)	.504 (8)	.523 (7)	.457 (9)	.593 (2)	.576 (5)	.341 (10)	.590 (3.5)	.723 (1)
CLL	.187	.258 (3)	.253 (5)	.145 (9)	.231 (7)	.181 (8)	.260 (2)	.245 (6)	.132 (10)	.272 (1)	.256 (4)
GLI	.197	.147 (5)	.205 (2)	.090 (7)	.129 (6)	.033 (10)	.174 (3)	.217 (1)	.074 (8)	.049 (9)	.159 (4)
GMA	.491	.525 (2)	.540 (1)	.287 (10)	.456 (7)	.332 (9)	.517 (4)	.506 (5)	.378 (8)	.520 (3)	.487 (6)
NCI	.435	.492 (2)	.419 (6)	.424 (5)	.396 (9)	.404 (8)	.459 (3)	.457 (4)	.359 (10)	.417 (7)	.520 (1)
PROS	.019	.040 (5)	.023 (8.5)	.041 (4)	.023 (8.5)	.060 (1)	.023 (8.5)	.023 (8.5)	.043 (3)	.024 (6)	.049 (2)
SMK	.001	.010 (5.5)	.009 (7)	.033 (1)	.012 (3)	.010 (5.5)	.008 (9)	.008 (9)	.019 (2)	.008 (9)	.011 (4)
TOX	.164	.318 (2)	.239 (6)	.195 (8)	.242 (5)	.152 (9)	.259 (4)	.272 (3)	.133 (10)	.235 (7)	.355 (1)
ORL	.849	.820 (4)	.822 (2.5)	.708 (8)	.803 (5)	.645 (9)	.829 (1)	.798 (6)	.544 (10)	.781 (7)	.822 (2)
PIX	.902	.904 (2)	.863 (7)	.696 (9)	.871 (5)	.709 (8)	.910 (1)	.870 (6)	.637 (10)	.877 (4)	.901 (3)
WPAR	.288	.372 (2)	.230 (9)	.302 (4)	.241 (6)	.231 (8)	.246 (5)	.206 (10)	.306 (3)	.236 (7)	.415 (1)
WPIE	.328	.514 (2)	.316 (5.5)	.405 (3)	.310 (8)	.245 (10)	.316 (5.5)	.308 (9)	.386 (4)	.314 (7)	.659 (1)
ARC	.091	.073 (3)	.080 (2)	.022 (9)	.059 (7)	.011 (10)	.063 (5.5)	.069 (4)	0.038 (8)	.063 (5.5)	.090 (1)

WDR: without dimensionality reduction; AE: autoencoder; KPCA: kernel PCA; LLE: locally linear embedding; MDS: multi-dimensional scaling; SE: spectral embedding; SPCA: sparse PCA; TSVD: truncated singular value decomposition; NMF: non-negative matrix factorization; PCA: principal component analysis; VAE: variational autoencoder

Table 6. Average NMI of different dimensions of different techniques on datasets. Higher value is better and values in parentheses indicate the rank of algorithm.

4.8 Statistical Analysis

In this section, we examined two statistical significance tests deemed most appropriate for the multiple-methods evaluation. We carried the nonparametric sign test and Friedman test for hypothesis testing.

Dataset	AE	KPCA	LLE	MDS	SE	SPCA	TSVD	NMF	PCA	VAE
ALL	66.1	11.4	11.0	10.5	11.5	13.0	11.7	11.8	12.6	32.0
CAR	71.4	14.6	12.4	12.3	13.6	15.1	12.4	12.3	13.6	38.4
CLL	75.0	18.3	16.1	22.8	22.4	22.5	18.5	18.1	18.4	42.0
GLI	77.9	25.4	25.1	24.6	24.7	24.4	26.8	25.7	24.7	44.9
GMA	56.7	8.4	9.6	11.3	10.9	10.6	9.2	8.5	10.4	27.7
NCI	62.4	13.4	13.2	13.4	13.5	13.9	14.0	14.5	13.3	33.4
PROS	58.5	11.7	10.3	10.8	11.6	12.6	11.6	11.5	12.6	27.5
SMK	76.4	24.5	23.2	24.4	24.3	24.3	25.1	23.3	23.3	45.4
TOX	57.4	11.1	10.1	10.2	11.9	12.3	11.2	11.6	12.4	29.4
ORL	64.2	19.9	17.7	21.8	21.4	20.7	18.5	17.5	17.7	40.6
PIX	61.1	18.9	14.4	17.8	16.5	19.6	14.6	14.7	14.9	40.1
WPAR	45.3	9.0	10.6	10.4	12.7	12.1	12.9	8.5	9.1	28.3
WPIE	49.4	9.3	11.3	11.3	13.6	13.3	12.3	9.5	9.4	30.4
ARC	71.4	22.6	23.0	23.5	21.9	22.8	21.3	21.8	21.6	47.4

Table 7. Average run-time of different reduced dimensions of different methods for each dataset (in seconds). The values shown in the table are the average of applied different dimensions, i.e., 2, 10, 20, 50, 100, 200, 300, 400, and 500.

4.8.1 Sign Test

Figure 3 illustrates a statistical comparison of VAE over state-of-the-art techniques. The nonparametric test, right-tailed sign test is carried out in the significance level $\alpha = 0.05$ (i.e., 95 % significance level). In the figure, for each metric, the first ten bars exhibit the z-value (test statistic value) for VAE against other techniques, whereas the eleventh bar presents the **z-ref** value. If the calculated z-value is greater than the **z-ref** value, then it indicates that the observed performance of VAE against the corresponding technique is statistically significant. From Figure 3, it is clear that the results obtained by VAE are significantly better than without DR and traditional DR techniques, although NMI seems quite poor for WDR and SPCA.

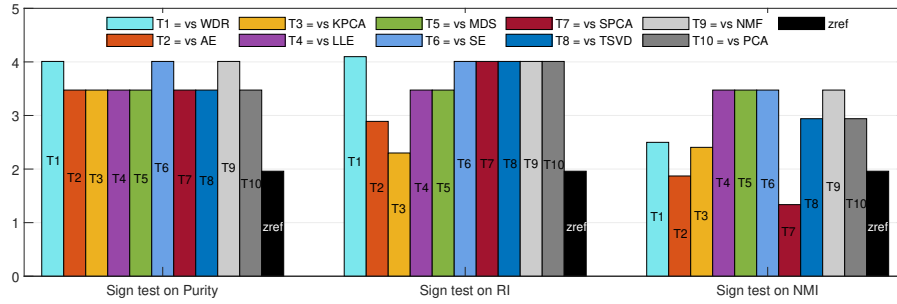


Figure 3. Sign test of VAE on the used 14 experimental datasets

4.8.2 Friedman Test

The Friedman test is used to assess there are any statistically significant differences between the distributions of methods. The p-values (**.000** $<$ .05) for this test are very small. Therefore, it is plausible that the eleven methods are significantly different. From Table 8, we have sufficient evidence to conclude a statistically significant difference between VAE and methods. For pairwise comparisons, we observed there were no significant differences between VAE and AE, SPCA.

5 CONCLUSION

This paper motivates the necessity of adopting moderate-dimensionality reduction and an unsupervised framework for high-dimension limited-sample size (HDLSS) data analysis. It proposes an unsupervised framework to deal with the classification of HDLSS data. The proposed method attempts to project the high-dimensional

Hypothesis Test Summary						Pairwise comparisons				
Friedman test on	**Null Hypothesis**	**Test Statistic**	**Kendall's W**	**Sig.**	**Decision**	**Sample Pair**	**Test Statistic**	**Std. Test Statistic**	**Sig.**	**Adj. Sig.**
Purity	The distributions of WDR, AE, KPCA, LLE, MDS, SE, SPCA, TSVD, NMF, PCA, and VAE are the same.	74.995	0.536	**.000**	Reject the Null hypothesis	T1	5.071	4.046	.000	**.003**
						T2	1.857	1.994	.046	1.000
						T3	4.214	3.362	.001	**.043**
						T4	6.214	4.957	.000	**.000**
						T5	5.607	4.473	.000	**.000**
						T6	7.964	6.353	.000	**.000**
						T7	2.500	1.994	.046	1.000
						T8	4.286	3.149	.001	**.035**
						T9	7.786	6.211	.000	**.000**
						T10	4.786	3.818	.000	**.007**
RI	The distributions of WDR, AE, KPCA, LLE, MDS, SE, SPCA, TSVD, NMF, PCA, and VAE are the same.	86.706	0.619	**.000**	Reject the Null hypothesis	T1	4.964	3.960	.000	**.004**
						T2	1.643	1.311	.190	1.000
						T3	3.964	3.162	.002	.086
						T4	7.179	5.727	.000	**.000**
						T5	4.786	3.818	.000	**.007**
						T6	8.429	6.724	.000	**.000**
						T7	2.964	2.365	.018	.993
						T8	4.679	0.541	.000	**.012**
						T9	8.214	6.553	.000	**.000**
						T10	4.643	3.704	.000	**.012**
NMI	The distributions of WDR, AE, KPCA, LLE, MDS, SE, SPCA, TSVD, NMF, PCA, and VAE are the same.	47.282	0.338	**.000**	Reject the Null hypothesis	T1	3.536	2.821	.005	**.264**
						T2	0.679	0.541	.588	1.000
						T3	2.857	2.279	.023	1.000
						T4	4.321	3.447	.001	**.031**
						T5	4.571	3.647	.000	**.015**
						T6	6.286	5.014	.000	**.000**
						T7	1.643	1.311	.190	1.000
						T8	3.643	2.906	.004	.201
						T9	5.393	4.302	.000	**.001**
						T10	3.607	2.878	.004	.220

Asymptotic significances (2-sided tests) are displayed. The significance level is 0.05.

Table 8. Friedman test results for different methods. T1 = VAE vs. WDR; T2 = VAE vs. AE; T3 = VAE vs. KPCA; T4 = VAE vs. LLE; T5 = VAE vs. MDS; T6 = VAE vs. SE; T7 = VAE vs. SPCA; T8 = VAE vs. TSVD; T9 = VAE vs. NMF; T10 = VAE vs. PCA.

data onto lower-dimensional space using variational autoencoder (VAE), then clustering is applied to the obtained lower-dimensional latent-space to find the groups and classify input data. The deep learning approach VAE enables the framework to avoid overfitting. To evaluate the method fourteen HDLSS datasets and three evaluation criteria were applied. Also, an empirical comparison is shown between VAE and state-of-the-art DR techniques. The results of the experiment demonstrated the effectiveness of the approach. In particular, experimentally investigated that dimension reduction of VAE is better than traditional techniques in the context of HDLSS data classification.

An effective and efficient DR method is essential for HDLSS data analysis. HDLSS data classification severe overfitting and high-variance gradients, whereas an unsupervised framework proved to be a good alternative. In contrast to the traditional DR method while use VAE can reduce the dimension as suitable from the HDLSS data that enhances the performance. This study combines the advantages of both unsupervised DR and unsupervised classification. A future line of this research is to study what kind of encoders and decoders are best suited for the

HDLSS problem. Another interesting future line of research will be finding an efficient dimension selection method (determining moderate d from p). Also, we are interested in designing a general framework that works on both unsupervised and semi-supervised settings. Finally, the reliability of the HDLSS data classification can increase in the meta or ensemble model.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Grant Nos. 61972261, 61473194).

APPENDIX

Detail performances of the 14 datasets with different numbers of dimensions in Tables A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14.

REFERENCES

[1] Hastie, T.—Tibshirani, R.—Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition. Springer Series in Statistics, 2009, doi: 10.1007/978-0-387-84858-7.

[2] Li, Y.—Li, T.—Liu, H.: Recent Advances in Feature Selection and Its Applications. Knowledge and Information Systems, Vol. 53, 2017, No. 3, pp. 551–577, doi: 10.1007/s10115-017-1059-8.

[3] Kuncheva, L. I.—Rodríguez, J. J.: On Feature Selection Protocols for Very Low-Sample-Size Data. Pattern Recognition, Vol. 81, 2018, pp. 660–673, doi: 10.1016/j.patcog.2018.03.012.

[4] Kuncheva, L. I.—Matthews, C. E.—Arnaiz-González, Á.—Rodríguez, J. J.: Feature Selection from High-Dimensional Data with Very Low Sample Size: A Cautionary Tale. 2020, arXiv: 2008.12025v1.

[5] Köppen, M.: The Curse of Dimensionality. 5$^{\text{th}}$ Online World Conference on Soft Computing in Industrial Applications (WSC5), 2000.

[6] Lv, J.: Impacts of High Dimensionality in Finite Samples. The Annals of Statistics, Vol. 41, 2013, No. 4, pp. 2236–2262, doi: 10.1214/13-aos1149.

[7] Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, Vol. 2, 1901, No. 11, pp. 559–572, doi: 10.1080/14786440109462720.

[8] Hotelling, H.: Analysis of a Complex of Statistical Variables into Principal Components. Journal of Educational Psychology, Vol. 24, 1933, No. 6, pp. 417–441, doi: 10.1037/h0070888.

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	70.8	72.2	72.2	72.2	73.6	74.8	76.4	75.0	73.6	54.0	60.6	57.0	70.0	63.4	62.0	73.7	73.7	60.6	.135	.124	.130	.142	.135	.131	.212	.204	.121
KPCA	**73.6**	73.6	72.2	70.8	x	x	x	x	x	**60.6**	60.6	59.3	58.1	x	x	x	x	x	**.155**	.137	.139	.125	x	x	x	x	x
LLE	55.6	**84.7**	59.7	56.9	x	x	x	x	x	49.9	**73.7**	51.2	50.3	x	x	x	x	x	.020	**.348**	.018	.087	x	x	x	x	x
MDS	58.3	70.8	70.8	70.8	63.9	65.3	75.0	66.7	65.3	50.7	58.1	58.1	58.1	53.2	54.0	62.0	54.9	54.0	.017	.125	.125	.125	.027	.039	.162	.001	.079
SE	54.2	61.1	59.7	50.0	x	x	x	x	x	49.6	51.8	51.2	49.3	x	x	x	x	x	.016	.055	.002	.138	x	x	x	x	x
SPCA	72.2	73.6	73.6	73.6	75.0	73.6	73.6	75.0	72.2	59.3	60.6	60.6	60.6	62.0	60.6	60.6	62.0	59.3	.130	.145	.155	.155	.171	.155	.145	.171	.115
TSVD	72.2	76.4	68.1	66.7	x	x	x	x	x	59.3	63.4	55.9	54.9	x	x	x	x	x	.139	.190	.059	.068	x	x	x	x	x
NMF	72.2	66.7	75.0	66.7	66.7	69.4	66.7	65.3	63.9	59.3	54.9	62.0	54.9	54.9	57.0	54.9	54.0	53.2	.139	.068	.163	.068	.068	.067	.068	.039	.047
PCA	**73.6**	65.3	75.0	65.3	x	x	x	x	x	**60.6**	54.0	62.0	54.0	x	x	x	x	x	**.155**	.079	.162	.079	x	x	x	x	x
VAE	72.2	81.9	**76.4**	**91.7**	**91.7**	**88.9**	**93.1**	**93.1**	**86.1**	59.3	70.0	**63.4**	**84.5**	**84.5**	**80.0**	**86.9**	**86.9**	**75.7**	.151	.299	**.190**	**.564**	**.573**	**.554**	**.675**	**.615**	**.388**

Table A1. Results of different dimensions of different techniques for the ALLAML dataset (72 observations, 7 129 features, 2 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	43.1	50.6	51.7	53.4	64.4	62.1	62.1	62.1	60.9	85.1	88.1	89.4	90.7	90.3	90.1	90.1	91.2	89.1	.356	.582	.448	.633	.574	.712	.673	.623	.711
KPCA	32.8	55.7	68.4	58.0	60.3	x	x	x	x	84.0	88.9	89.1	87.9	88.6	x	x	x	x	.354	.637	.658	.609	.610	x	x	x	x
LLE	**44.8**	**70.1**	55.2	37.9	31.0	x	x	x	x	85.1	92.0	81.6	69.7	72.9	x	x	x	x	**.498**	**.724**	.611	.400	.290	x	x	x	x
MDS	39.1	51.7	51.1	49.4	50.6	49.4	54.0	51.7	60.9	86.2	88.6	87.9	86.0	87.9	82.9	88.1	86.7	89.0	.349	.509	.532	.498	.570	.519	.567	.571	.595
SE	43.1	57.5	44.8	39.7	32.8	x	x	x	x	85.2	90.3	78.5	72.9	55.5	x	x	x	x	.494	.643	.467	.409	.273	x	x	x	x
SPCA	34.5	57.5	53.4	60.3	62.1	63.8	62.1	63.8	57.5	84.3	90.2	87.8	90.7	91.3	91.1	90.6	89.7	88.9	.348	.600	.579	.619	.653	.662	.649	.631	.595
TSVD	29.9	57.5	69.0	57.5	59.2	x	x	x	x	80.8	89.4	91.5	89.5	89.0	x	x	x	x	.296	.622	.695	.634	.632	x	x	x	x
NMF	28.7	64.9	64.9	53.4	27.6	22.4	23.0	25.9	24.1	82.9	**92.3**	91.2	83.7	58.3	72.4	70.1	76.4	78.5	.279	.685	**.721**	.542	.226	.122	.156	.167	.176
PCA	33.9	66.7	57.5	62.6	62.1	x	x	x	x	83.3	91.8	88.8	90.4	90.6	x	x	x	x	.372	.676	.613	.660	.631	x	x	x	x
VAE	**44.8**	64.4	**70.7**	**74.1**	**80.5**	**81.0**	**75.9**	**77.0**	**74.1**	**86.3**	90.0	**93.5**	**93.9**	**95.9**	**95.8**	**94.6**	**95.4**	**94.4**	.486	.630	.705	**.741**	**.826**	**.792**	**.769**	**.785**	**.776**

Table A2. Results of different dimensions of different techniques for the CARCINOM dataset (174 observations, 9 182 features, 11 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	52.3	50.1	51.4	55.0	56.8	**60.4**	60.4	55.9	54.1	57.4	58.2	57.7	57.6	58.2	58.2	56.9	55.2	59.4	.265	.298	.216	.289	.298	.159	**.287**	.216	.292
PCA	52.3	61.3	54.1	53.2	55.0	x	x	x	x	57.4	58.8	57.5	56.9	55.6	x	x	x	x	.213	.319	.248	.192	.296	x	x	x	x
LLE	57.7	51.4	55.9	51.4	54.1	x	x	x	x	57.2	52.6	51.1	52.3	47.9	x	x	x	x	.255	.250	.118	.046	.054	x	x	x	x
MDS	40.5	49.5	51.4	52.3	56.8	55.9	60.4	55.0	55.0	53.0	55.8	56.7	55.2	58.2	**58.5**	57.3	57.6	57.2	.020	.146	.149	**.379**	.277	**.378**	.268	**.250**	.215
SE	**58.6**	**67.6**	49.5	55.0	45.9	x	x	x	x	57.7	**60.6**	44.9	57.6	41.4	x	x	x	x	.272	.254	.082	.250	.046	x	x	x	x
SPCA	56.8	55.9	55.0	55.0	58.6	55.0	55.0	55.0	55.0	**58.1**	58.0	57.6	57.6	56.9	57.6	57.8	57.6	57.6	.272	.267	.250	.250	.285	.250	.263	**.250**	.250
TSVDAE	56.8	51.4	55.0	54.1	55.0	x	x	x	x	57.6	55.3	57.6	57.7	57.7	x	x	x	x	.248	.184	.250	.260	.281	x	x	x	x
NMF	56.8	55.0	52.3	50.5	45.9	45.9	50.5	47.7	49.5	57.6	56.7	55.2	44.3	41.6	42.0	43.2	43.3	44.1	.248	.267	**.328**	.068	.046	.046	.067	.046	.077
PCA	56.8	55.0	54.1	51.4	55.9	x	x	x	x	57.6	57.6	57.4	57.1	58.9	x	x	x	x	.248	.250	.222	.196	**.443**	x	x	x	x
VAE	55.0	61.3	**62.2**	**62.2**	**63.1**	59.5	**61.3**	**60.4**	**64.0**	57.6	59.4	**60.4**	**59.7**	**59.8**	53.4	**58.9**	**58.1**	**60.7**	**.275**	**.322**	.220	.260	.319	.083	.285	.229	**.311**

Table A3. Results of different dimensions of different techniques for the CLL_SUB dataset (111 observations, 11 340 features, 3 classes)

Bold, shade, and x indicate best result of individual reduced dimensions, best among the applied reduced dimensions, and algorithms failed to provide reduced dimensions, respectively.

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	71.8	**72.9**	58.0	69.4	**70.6**	70.6	**72.9**	**72.9**	69.4	62.4	**68.6**	**76.2**	60.1	**68.2**	**78.9**	**69.8**	**63.6**	**66.2**	.181	.216	.219	.140	.109	.123	.083	.105	.148
KPCA	69.4	69.4	68.2	71.8	x	x	x	x	x	57.0	57.0	56.1	59.0	x	x	x	x	x	.239	.239	.228	.112	x	x	x	x	x
LLE	70.6	57.6	65.9	**74.1**	x	x	x	x	x	58.0	50.6	54.5	**61.2**	x	x	x	x	x	.101	.089	.000	.171	x	x	x	x	x
MDS	75.3	54.1	**72.9**	69.4	**70.6**	**78.8**	69.4	71.8	71.8	62.4	49.7	60.1	57.0	58.0	66.2	57.0	59.0	59.0	**.302**	.004	.145	.061	**.251**	**.217**	.008	.095	.076
SE	68.2	63.5	63.5	68.2	x	x	x	x	x	56.1	53.1	53.1	56.1	x	x	x	x	x	.066	.001	.060	.003	x	x	x	x	x
SPCA	68.2	**72.9**	68.2	68.2	69.4	76.5	70.6	68.2	**76.5**	56.1	60.1	56.1	56.1	57.0	63.6	58.0	56.1	63.6	.047	.173	.047	.228	.239	.316	.059	**.228**	**.231**
TSVD	69.4	70.6	70.6	72.9	x	x	x	x	x	57.0	58.0	58.0	60.1	x	x	x	x	x	.239	**.251**	.101	**.275**	x	x	x	x	x
NMF	72.9	67.1	**72.9**	56.1	58.0	56.1	56.1	57.0	58.0	60.1	55.3	60.1	56.1	58.0	56.1	56.1	57.0	58.0	.213	.183	.083	.022	.033	.033	.022	.008	.071
PCA	57.6	68.2	70.6	70.6	x	x	x	x	x	50.6	56.1	58.0	58.0	x	x	x	x	x	.020	.047	.059	.071	x	x	x	x	x
VAE	**76.5**	70.6	70.6	67.1	**70.6**	70.6	**72.9**	70.6	72.9	**63.6**	58.0	58.0	55.3	58.0	58.0	60.1	58.0	60.1	.213	**.251**	**.251**	.217	.035	.071	**.107**	.071	.213

Table A4. Results of different dimensions of different techniques for the GLI_85 dataset (85 observations, 22 283 features, 2 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	60.0	62.0	60.0	64.0	62.0	64.0	66.0	66.0	60.0	73.6	74.2	73.1	77.2	74.3	75.6	72.7	75.6	74.9	**.556**	.564	.495	.517	.508	**.584**	.499	.511	.490
KPCA	58.0	52.0	54.0	64.0	x	x	x	x	x	**75.6**	71.5	73.3	74.2	x	x	x	x	x	.501	**.576**	**.573**	.512	x	x	x	x	x
LLE	54.0	40.0	44.0	x	x	x	x	x	x	71.7	54.9	46.8	x	x	x	x	x	x	.476	.213	.172	x	x	x	x	x	x
MDS	46.0	54.0	42.0	56.0	60.0	62.0	58.0	68.0	54.0	71.3	72.3	48.7	71.1	73.1	73.6	72.7	75.5	72.3	.397	.467	.255	.479	.485	.494	**.524**	.490	**.518**
SE	56.0	46.0	42.0	x	x	x	x	x	x	73.2	52.5	52.8	x	x	x	x	x	x	.497	.277	.221	x	x	x	x	x	x
SPCA	60.0	56.0	56.0	58.0	**64.0**	**66.0**	66.0	54.0	**66.0**	73.1	74.9	72.7	75.6	74.4	75.4	**75.8**	74.3	**75.4**	.500	.489	.568	.501	**.509**	.524	.511	**.544**	.509
TSVD	60.0	60.0	**66.0**	62.0	x	x	x	x	x	73.6	73.6	**74.5**	73.6	x	x	x	x	x	.491	.484	.539	.508	x	x	x	x	x
NMF	56.0	60.0	50.0	34.0	52.0	54.0	54.0	54.0	60.0	64.9	70.4	63.8	31.5	65.1	71.4	71.1	71.8	73.1	.245	.537	.271	.125	.436	.482	.414	.429	.464
PCA	56.0	62.0	58.0	58.0	x	x	x	x	x	74.9	73.6	74.2	75.9	x	x	x	x	x	.489	.538	.538	.514	x	x	x	x	x
VAE	**62.0**	**68.0**	60.0	**72.0**	62.0	**66.0**	**68.0**	**70.0**	62.0	74.1	**75.0**	73.6	**80.1**	**74.8**	**76.7**	73.6	**77.2**	73.8	.508	.493	.458	**.545**	.465	.469	.424	.535	.486

Table A5. Results of different dimensions of different techniques for the GLIOMA dataset (50 observations, 4 434 features, 4 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	41.7	46.7	51.7	**51.7**	53.3	55.0	**53.3**	50.0	48.3	81.7	84.3	81.3	84.3	82.9	81.9	82.7	82.9	84.3	.436	.462	.502	.523	.486	.553	.481	**.542**	.447
KPCA	36.7	41.7	48.3	41.7	x	x	x	x	x	82.4	80.5	81.9	78.9	x	x	x	x	x	.396	.407	.471	.404	x	x	x	x	x
LLE	41.7	**51.7**	41.7	33.3	x	x	x	x	x	82.7	84.0	82.9	69.4	x	x	x	x	x	.426	.542	.387	.343	x	x	x	x	x
MDS	28.3	31.7	43.3	36.7	46.7	40.0	41.7	46.7	45.0	81.7	79.7	81.7	80.8	84.3	81.9	80.4	80.7	78.3	.298	.314	.436	.353	.493	.427	.398	.441	.404
SE	38.3	38.3	40.0	31.7	x	x	x	x	x	82.0	81.9	76.4	66.1	x	x	x	x	x	.421	.467	.438	.291	x	x	x	x	x
SPCA	35.0	41.7	43.3	43.3	43.3	48.3	**53.3**	48.3	45.0	81.4	84.3	80.7	81.3	82.1	82.4	84.0	84.4	83.7	.385	.482	.451	.412	.440	.449	.512	.523	.476
TSVD	38.3	43.3	43.3	41.7	x	x	x	x	x	**82.8**	83.1	81.5	82.8	x	x	x	x	x	.411	.502	.465	.451	x	x	x	x	x
NMF	**43.3**	**51.7**	**58.3**	28.3	26.7	26.7	30.0	26.7	25.0	82.2	81.5	83.3	37.5	33.0	34.7	47.6	58.0	49.9	**.462**	**.553**	**.600**	.303	.267	.243	.285	.266	.255
PCA	35.0	48.3	45.0	35.0	x	x	x	x	x	81.9	81.2	79.0	77.7	x	x	x	x	x	.379	.476	.447	.366	x	x	x	x	x
VAE	**43.3**	48.3	48.3	**51.7**	**55.0**	**56.7**	**53.3**	**51.7**	**50.0**	80.9	**85.5**	**85.0**	**85.9**	**85.5**	**86.9**	**85.5**	**85.7**	**85.3**	.448	.519	.514	**.543**	**.532**	**.588**	**.523**	.533	**.479**

Table A6. Results of different dimensions of different techniques for the NCI9 dataset (60 observations, 9 712 features, 9 classes)

Bold, shade, and x indicate best result of individual reduced dimensions, best among the applied reduced dimensions, and algorithms failed to provide reduced dimensions, respectively.

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	53.9	55.9	57.8	59.8	**62.7**	58.8	59.8	61.7	55.9	50.7	49.6	51.1	51.4	50.5	50.7	51.4	**52.8**	51.1	.007	.019	.010	.060	.034	.097	.055	.059	.018
KPCA	**58.8**	58.8	56.9	58.8	58.8	x	x	x	x	**51.1**	51.1	50.5	51.1	51.1	x	x	x	x	.026	.026	.014	.026	.026	x	x	x	x
LLE	55.9	**66.7**	**58.8**	53.9	53.9	x	x	x	x	50.2	**55.1**	**51.1**	49.8	49.8	x	x	x	x	.010	**.114**	.026	.047	.007	x	x	x	x
MDS	57.8	57.8	**58.8**	58.8	58.8	58.8	58.8	58.8	57.8	50.7	50.7	**51.1**	51.1	51.1	51.1	51.1	51.1	50.7	.018	.018	.026	.026	.026	.027	.026	.027	.019
SE	**58.8**	64.7	55.9	53.9	52.0	x	x	x	x	**51.1**	53.9	50.2	49.8	49.6	x	x	x	x	.055	.169	.029	.009	.036	x	x	x	x
SPCA	**58.8**	57.8	57.8	58.8	58.8	57.8	58.8	58.8	58.8	**51.1**	50.7	50.7	51.1	51.1	50.7	51.1	51.1	51.1	.026	.018	.018	.026	.026	.019	.026	.026	.026
TSVD	**58.8**	58.8	57.8	58.8	57.8	x	x	x	x	**51.1**	51.1	50.7	51.1	50.7	x	x	x	x	.026	.026	.019	.026	.019	x	x	x	x
NMF	**58.8**	57.8	52.0	50.0	52.0	53.9	58.8	58.8	**59.8**	**51.1**	50.7	49.6	49.5	49.6	49.8	51.1	51.1	**51.4**	**.027**	.057	**.036**	.034	.036	.004	.056	.056	**.078**
PCA	**58.8**	58.8	**58.8**	58.8	57.8	x	x	x	x	**51.1**	51.1	**51.1**	51.1	50.7	x	x	x	x	.026	.026	.026	.026	.019	x	x	x	x
VAE	55.9	56.9	55.9	**62.7**	**62.7**	**59.8**	**61.8**	**62.7**	58.8	50.2	50.5	50.2	**52.8**	**52.8**	**51.4**	**52.3**	**52.8**	51.1	.010	.013	.010	**.076**	**.051**	**.113**	**.067**	**.076**	.026

Table A7. Results of different dimensions of different techniques for the PROSTATE_GE dataset (102 observations, 5 966 features, 2 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	56.7	58.8	55.6	56.7	56.7	56.1	52.4	51.9	50.8	50.6	51.1	50.6	51.3	50.6	**50.6**	50.0	**50.5**	50.5	.008	.019	.010	.008	.012	.010	.002	.008	.010
KPCA	56.1	52.9	56.7	56.1	56.7	x	x	x	x	50.5	49.9	50.6	50.5	50.6	x	x	x	x	.010	.002	.012	.010	.012	x	x	x	x
LLE	**65.2**	52.4	54.0	**60.4**	**58.8**	x	x	x	x	**54.4**	49.8	50.1	**51.9**	**51.3**	x	x	x	x	**.076**	.015	.010	**.037**	**.029**	x	x	x	x
MDS	58.3	56.7	56.7	56.7	54.0	**56.7**	**56.1**	**56.1**	**56.7**	51.1	50.6	50.6	50.6	50.1	**50.6**	**50.5**	**50.5**	**50.6**	.019	.012	.012	.012	.007	.012	**.010**	**.010**	**.012**
SE	50.3	52.9	50.8	52.4	56.7	x	x	x	x	49.7	49.9	49.7	49.8	50.6	x	x	x	x	.000	.020	.005	.005	.018	x	x	x	x
SPCA	55.6	53.5	55.6	55.6	55.6	56.1	**56.1**	**56.1**	55.6	50.4	50.0	50.4	50.4	50.4	50.5	**50.5**	**50.5**	50.4	.008	.003	.008	.008	.008	.010	**.010**	**.010**	.008
TSVD	56.1	56.1	55.6	52.9	56.1	x	x	x	x	50.5	50.5	50.4	49.9	50.5	x	x	x	x	.010	.010	.008	.002	.010	x	x	x	x
NMF	56.7	56.7	**64.7**	51.9	52.4	51.3	51.3	52.4	51.3	50.6	50.6	**54.1**	49.8	49.8	49.8	49.8	49.8	49.8	.012	.012	**.066**	.023	.026	.023	.003	.005	.000
PCA	53.5	56.1	55.6	55.6	56.1	x	x	x	x	50.0	50.5	50.4	50.4	50.5	x	x	x	x	.003	.010	.008	.008	.010	x	x	x	x
VAE	58.8	**61.0**	57.2	57.8	54.0	52.4	52.9	51.9	51.9	51.3	**52.1**	50.8	50.9	50.1	49.8	49.9	49.8	49.8	.022	**.034**	.015	.012	.004	.001	.002	.006	.000

Table A8. Results of different dimensions of different techniques for the SMK_CAN_187 dataset (187 observations, 19 993 features, 2 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	44.4	57.3	52.6	52.6	48.5	48.0	57.3	57.3	44.4	66.3	68.0	72.7	69.2	70.1	69.2	70.9	68.0	67.4	.212	.418	.361	.293	.296	.290	.294	.449	.252
KPCA	42.7	42.1	46.8	46.2	48.0	x	x	x	x	66.4	66.2	70.1	69.3	69.2	x	x	x	x	.216	.261	.253	.231	.233	x	x	x	x
LLE	48.0	45.6	41.5	33.3	30.4	x	x	x	x	65.7	68.9	65.2	51.8	37.5	x	x	x	x	**.346**	.324	.187	.064	.055	x	x	x	x
MDS	39.8	42.1	43.9	42.7	43.9	44.4	45.0	47.4	45.0	64.6	68.5	67.4	60.4	66.2	66.3	63.2	69.7	67.6	.081	.212	.228	.295	.295	.296	.280	.254	.241
SE	48.0	41.5	40.4	37.4	35.1	x	x	x	x	**68.4**	62.0	61.2	54.3	49.7	x	x	x	x	.283	.141	.161	.100	.075	x	x	x	x
SPCA	43.3	44.4	46.2	45.6	47.4	46.2	46.8	48.5	**48.5**	67.9	67.0	66.9	67.2	69.4	68.0	69.4	69.2	**67.7**	.287	.277	.253	.274	.231	.267	.232	.259	.255
TSVD	**48.5**	45.0	46.2	48.5	52.6	x	x	x	x	69.9	66.9	65.6	68.3	68.3	x	x	x	x	.246	.241	.277	.298	.297	x	x	x	x
NMF	44.4	47.4	49.7	33.9	30.4	30.4	28.7	29.8	32.7	66.3	68.1	65.4	32.7	32.2	32.7	27.7	41.1	49.8	.241	.259	.190	.164	.114	.102	.055	.036	.037
PCA	43.9	46.2	46.8	43.9	45.0	x	x	x	x	66.3	69.2	67.8	67.7	67.8	x	x	x	x .234	.228	.232	.239	.241	x	x	x	x	
VAE	46.2	**60.8**	**59.6**	**57.3**	**57.9**	**57.3**	**64.3**	**63.2**	45.6	67.2	**75.9**	**74.2**	**72.0**	**72.7**	**70.9**	**74.7**	**77.2**	64.9	.196	**.459**	**.393**	**.310**	**.313**	**.312**	**.396**	**.552**	**.264**

Table A9. Results of different dimensions of different techniques for the TOX_171 dataset (171 observations, 5 748 features, 4 classes)

Bold, shade, and x indicate best result of individual reduced dimensions, best among the applied reduced dimensions, and algorithms failed to provide reduced dimensions, respectively.

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	61.0	75.0	77.0	73.0	78.0	82.0	78.0	77.0	84.0	87.8	93.9	94.7	92.3	95.0	95.2	93.7	93.5	94.0	.754	.870	.820	.847	.790	.800	.837	.795	.867
KPCA	**70.0**	71.0	69.0	74.0	78.0	x	x	x	x	92.8	93.2	92.3	94.3	94.3	x	x	x	x	.825	.797	.782	**.855**	.853	x	x	x	x
LLE	60.0	**80.0**	61.0	41.0	x	x	x	x	x	86.7	95.5	85.3	75.9	x	x	x	x	x	**.832**	**.893**	.681	.426	x	x	x	x	x
MDS	68.0	69.0	77.0	69.0	75.0	70.0	75.0	72.0	77.0	91.8	92.3	94.6	93.1	93.0	89.7	93.5	91.1	95.0	.742	.761	.829	.782	.836	.800	.797	.805	.871
SE	59.0	77.0	62.0	40.0	x	x	x	x	x	87.8	93.3	86.8	78.1	x	x	x	x	x	.730	.800	.660	.390	x	x	x	x	x
SPCA	69.0	71.0	72.0	71.0	79.0	73.0	**82.0**	73.0	73.0	**93.1**	92.8	91.8	93.5	95.2	93.6	**94.7**	93.9	93.2	.828	.825	.795	.809	**.880**	.829	.845	.830	.820
TSVD	58.0	68.0	**80.0**	**78.0**	76.0	x	x	x	x	90.2	91.7	**94.9**	93.5	93.7	x	x	x	x	.727	.774	**.846**	.816	.830	x	x	x	x
NMF	54.0	74.0	73.0	50.0	26.0	30.0	43.0	40.0	33.0	89.8	92.8	92.3	83.1	59.7	72.7	76.7	81.1	77.8	.695	.783	.844	.622	.252	.332	.525	.439	.408
PCA	61.0	63.0	72.0	**78.0**	67.0	x	x	x	x	91.2	90.1	92.2	**94.6**	91.6	x	x	x	x	.767	.761	.795	.830	.754	x	x	x	x
VAE	58.0	78.0	**80.0**	**78.0**	**83.0**	**84.0**	**82.0**	**78.0**	**87.0**	89.4	**94.9**	94.0	93.5	**95.4**	**95.9**	94.6	**94.8**	**96.3**	.635	.843	.822	.839	.793	**.866**	**.853**	**.853**	**.895**

Table A10. Results of different dimensions of different techniques for the ORLRAWS10P dataset (100 observations, 10 304 features, 10 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	82.0	84.0	89.0	84.0	83.0	94.0	82.0	91.0	89.0	95.	9 95.6	96.6	95.6	96.6	98.1	94.8	97.1	96.9	**.902**	.887	.896	.905	.890	.911	**.915**	.890	.938
KPCA	82.0	**86.0**	72.0	80.0	67.0	x	x	x	x	95.5	96.1	93.5	94.9	91.2	x	x	x	x	.877	.891	.836	.867	.845	x	x	x	x
LLE	56.0	72.0	61.0	40.0	x	x	x	x	x	88.1	90.9	83.6	69.3	x	x	x	x	x	.763	.855	.707	.457	x	x	x	x	x
MDS	71.0	79.0	85.0	81.0	78.0	89.0	**84.0**	82.0	84.0	93.0	94.8	96.3	95.0	94.0	96.5	**96.4**	95.4	95.2	.790	.851	.901	.867	.858	.912	.902	.896	.863
SE	56.0	82.0	57.0	44.0	x	x	x	x	x	87.6	95.9	85.4	73.4	x	x	x	x	x	.731	.909	.724	.475	x	x	x	x	x
SPCA	79.0	**86.0**	**91.0**	**94.0**	84.0	91.0	81.0	81.0	**94.0**	93.7	96.1	**97.1**	**98.1**	96.9	97.1	95.6	95.7	**98.1**	.839	.891	**.922**	**.950**	**.938**	.922	.876	**.905**	**.950**
TSVD	74.0	84.0	82.0	82.0	**87.0**	x	x	x	x	93.4	96.0	95.5	95.3	**96.8**	x	x	x	x	.802	.893	.887	.854	.914	x	x	x	x
NMF	75.0	81.0	72.0	56.0	45.0	40.0	37.0	44.0	39.0	94.0	95.5	91.4	86.1	79.7	77.0	75.8	82.9	80.1	.825	.890	.780	.653	.638	.507	.495	.536	.410
PCA	82.0	84.0	89.0	73.0	81.0	x	x	x	x	95.3	**96.6**	96.6	93.4	95.5	x	x	x	x	.864	**.920**	.913	.824	.864	x	x	x	x
VAE	**83.0**	**86.0**	90.0	89.0	**87.0**	**98.0**	81.0	**94.0**	84.0	**96.2**	95.7	96.6	96.5	96.5	**99.3**	95.2	**98.1**	96.7	.899	.878	.897	.912	.897	**.972**	.844	.891	.915

Table A11. Results of different dimensions of different techniques for the PIXRAW10P dataset (100 observations, 10 000 features, 10 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	23.8	42.3	39.2	46.2	38.5	40.8	40.0	30.8	34.6	80.7	82.6	83.5	85.5	83.3	83.1	84.4	83.5	83.3	.141	.352	.379	.451	**.453**	.395	.483	.296	.384
KPCA	23.1	24.6	30.8	31.5	27.7	x	x	x	x	82.3	81.8	83.8	83.2	81.9	x	x	x	x	.180	.174	.269	.285	.239	x	x	x	x
LLE	23.8	34.6	37.7	33.8	26.2	x	x	x	x	79.3	84.0	68.5	75.2	52.9	x	x	x	x	.174	.419	.354	.312	.248	x	x	x	x
MDS	23.8	27.7	28.5	24.6	29.2	28.5	29.2	26.2	26.9	**83.1**	83.2	81.8	80.7	83.3	82.9	82.9	83.3	82.6	.216	.276	.246	.189	.236	.269	.263	.230	.243
SE	23.8	28.5	30.0	28.5	24.6	x	x	x	x	82.3	83.5	81.7	76.6	66.0	x	x	x	x	.182	.248	.294	.225	.207	x	x	x	x
SPCA	20.8	29.2	30.0	26.2	30.8	27.7	29.2	29.2	26.9	80.7	83.7	81.6	81.9	81.9	80.9	83.2	81.9	82.6	.160	.247	.252	.256	.294	.230	.240	.268	.265
TSVD	21.5	22.3	25.4	26.2	23.8	x	x	x	x	82.6	80.3	82.9	82.9	81.8	x	x	x	x	.194	.131	.232	.224	.251	x	x	x	x
NMF	**24.6**	38.5	**54.6**	40.0	26.9	23.8	26.9	23.8	25.4	82.0	85.5	87.3	75.8	53.9	70.1	65.5	70.5	78.3	**.226**	.399	.571	.421	.263	.227	.238	.194	.213
PCA	**24.6**	25.4	27.7	29.2	28.5	x	x	x	x	82.3	83.1	82.3	83.0	80.6	x	x	x	x	.195	.208	.234	.280	.262	x	x	x	x
VAE	21.5	**46.2**	40.0	**50.0**	**39.2**	**42.3**	**40.8**	**33.8**	**37.7**	81.5	**86.5**	**85.7**	**87.5**	**84.4**	**85.6**	**86.5**	**84.2**	**84.6**	.150	**.463**	**.388**	**.517**	.446	**.478**	**.502**	**.370**	**.419**

Table A12. Results of different dimensions of different techniques for the WARPAR_10P dataset (130 observations, 2 400 features, 10 classes)

Bold, shade, and x indicate best result of individual reduced dimensions, best among the applied reduced dimensions, and algorithms failed to provide reduced dimensions, respectively.

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	34.3	58.6	54.7	86.9	91.8	68.4	48.7	42.8	42.8	83.1	88.3	87.4	88.6	98.4	88.6	84.1	83.5	83.9	.302	.610	.506	.739	.825	.591	.365	.307	.380
KPCA	22.4	29.5	30.5	35.7	32.4	30.0	x	x	x	82.4	83.1	84.0	82.7	83.4	83.7	x	x	x	.148	.342	.337	.382	.355	.329	x	x	x
LLE	26.7	51.4	54.3	34.8	29.5	23.8	x	x	x	81.8	86.7	87.9	66.9	77.6	68.2	x	x	x	.344	.616	**.698**	.329	.267	.175	x	x	x
MDS	24.3	29.0	31.4	29.5	32.9	31.9	31.0	29.0	29.5	82.3	83.8	83.5	82.8	83.6	83.9	83.4	83.7	82.1	.209	.321	.332	.308	.347	.333	.320	.345	.272
SE	21.9	29.5	30.5	32.9	28.6	21.0	x	x	x	80.9	82.7	80.1	80.4	75.6	59.8	x	x	x	.182	.285	.350	.291	.220	.141	x	x	x
SPCA	24.3	28.1	29.5	30.0	31.4	30.0	32.9	30.5	32.4	82.5	82.9	82.6	83.4	80.9	83.3	83.5	82.3	82.6	.226	.286	.300	.329	.356	.337	.341	.320	.352
TSVD	21.9	34.3	33.3	31.0	32.4	31.0	x	x	x	82.5	82.9	82.9	82.6	82.0	81.3	x	x	x	.156	.380	.360	.318	.329	.304	x	x	x
NMF	25.2	48.6	**67.1**	54.8	42.9	31.4	29.5	24.3	26.2	82.4	86.9	88.6	81.7	64.7	59.1	73.9	63.3	71.2	.238	.556	.689	.570	.464	.307	.235	.213	.204
PCA	23.3	30.5	33.3	31.4	30.5	32.4	x	x	x	82.4	82.3	84.2	80.8	83.4	82.4	x	x	x	.205	.300	.372	.335	.335	.335	x	x	x
VAE	**34.8**	**63.8**	58.6	**91.0**	**100.0**	**70.5**	**51.4**	**46.7**	**43.3**	**84.3**	**91.0**	**89.4**	**97.4**	**100.0**	**92.5**	**87.3**	**84.1**	**84.7**	**.369**	**.719**	.621	**.918**	**1.000**	**.782**	**.582**	**.474**	**.465**

Table A13. Result of different dimensions of different techniques for the WARPPIE dataset (210 observations, 2 420 features, 10 classes)

	Purity(%)									RI(%)									NMI (rate0-1)								
Algorithm	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500	2	10	20	50	100	200	300	400	500
AE	64.5	64.5	63.0	65.0	67.0	64.5	66.0	64.0	65.0	53.7	53.1	54.0	54.0	54.3	54.0	54.3	53.1	54.0	.053	.067	.081	.073	.090	.075	.073	.081	.081
KPCA	64.5	65.0	65.0	65.0	65.0	65.0	x	x	x	54.0	54.3	54.3	54.3	54.3	54.3	x	x	x	.077	.081	.081	.081	.977	.081	x	x	x
LLE	59.0	57.5	50.5	53.5	56.0	x	x	x	x	51.4	50.9	49.8	50.0	50.5	x	x	x	x	.016	.006	.057	.021	.008	x	x	x	x
MDS	56.5	64.5	65.0	63.0	64.5	64.5	64.5	59.0	65.0	50.6	54.0	54.3	53.1	54.0	54.0	54.0	51.4	54.3	.009	.077	.081	.039	.077	**.077**	**.077**	.016	.081
SE	59.0	53.5	54.0	55.0	53.5	x	x	x	x	51.4	50.0	50.1	50.3	50.0	x	x	x	x	.016	.007	.001	.008	.021	x	x	x	x
SPCA	64.0	64.5	64.5	64.5	64.5	64.5	59.0	64.5	59.0	53.7	54.0	54.0	54.0	54.0	54.0	51.4	54.0	51.4	.073	.077	.077	.077	.077	.077	.016	.077	.016
TSVD	64.5	59.0	65.0	65.0	65.0	64.5	x	x	x	54.0	51.4	54.3	54.3	54.3	54.0	x	x	x	.077	.016	.081	.081	.081	.077	x	x	x
NMF	**65.0**	**66.0**	63.0	58.5	56.5	55.5	56.5	58.5	56.5	**54.3**	**54.9**	53.1	51.2	50.6	50.4	50.6	51.2	50.6	**.081**	**.090**	.039	.015	.028	.020	.028	.013	.028
PCA	63.0	59.0	65.0	65.0	65.0	64.5	x	x	x	53.1	51.4	54.3	54.3	54.3	54.0	x	x	x	.039	.016	.081	.082	.082	.078	x	x	x
VAE	**65.0**	**66.0**	**65.5**	**66.0**	**69.0**	**66.0**	**66.5**	**66.0**	**67.0**	**54.3**	**54.9**	**54.6**	**54.9**	**57.0**	**54.9**	**55.2**	**54.9**	**55.6**	**.081**	**.090**	**.102**	**.091**	**.102**	**.091**	.076	**.091**	**.100**

Table A14. Results of different dimensions of different techniques for the ARCENE dataset (200 observations, 10 000 features, 2 classes)

Bold, shade, and x indicate best result of individual reduced dimensions, best among the applied reduced dimensions, and algorithms failed to provide reduced dimensions, respectively.

[9] Tipping, M. E.—Bishop, C. M.: Mixtures of Probabilistic Principal Component Analyzers. Neural Computation, Vol. 11, 1999, No. 2, pp. 443–482, doi: 10.1162/089976699300016728.

[10] Hyvärinen, A.—Oja, E.: Independent Component Analysis: Algorithms and Applications. Neural Networks, Vol. 13, 2000, No. 4-5, pp. 411–430, doi: 10.1016/s0893-6080(00)00026-5.

[11] Barber, D.: Bayesian Reasoning and Machine Learning. Cambridge University Press, New York, NY, United States, 2012, doi: 10.1017/CBO9780511804779.

[12] Cox, T.—Cox, M.: Multidimensional Scaling. In: Chen, C. H., Härdle, W. K., Unwin, A. (Eds.): Handbook of Data Visualization. Springer, Berlin, Heidelberg, Springer Handbooks Comp. Statistics, 2008, pp. 315–347, doi: 10.1007/978-3-540-33037-0_14.

[13] Cichocki, A.—Phan, A. H.: Fast Local Algorithms for Large Scale Nonnegative Matrix and Tensor Factorizations. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E92.A, 2009, No. 3, pp. 708–721, doi: 10.1587/transfun.e92.a.708.

[14] Wang, Y.—Yao, H.—Zhao, S.: Auto-Encoder Based Dimensionality Reduction. Neurocomputing, Vol. 184, 2016, pp. 232–242, doi: 10.1016/j.neucom.2015.08.104.

[15] Yue, T.—Wang, H.: Deep Learning for Genomics: A Concise Overview. 2018, arXiv: 1802.00810.

[16] Srivastava, N.—Hinton, G.—Krizhevsky, A.—Sutskever, I.—Salakhutdinov, R.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, Vol. 15, 2014, pp. 1929–1958.

[17] Zhao, W.: Research on the Deep Learning of the Small Sample Data Based on Transfer Learning. AIP Conference Proceedings, Vol. 1864, 2017, No. 1, Art. No. 020018, doi: 10.1063/1.4992835.

[18] Lamb, A.—Dumoulin, V.—Courville, A.: Discriminative Regularization for Generative Models. 2016, arXiv: 1602.03220.

[19] Mahmud, M. S.—Fu, X.: Unsupervised Classification of High-Dimension and Low-Sample Data with Variational Autoencoder Based Dimensionality Reduction. 2019 IEEE 4$^{\text{th}}$ International Conference on Advanced Robotics and Mechatronics (ICARM), Toyonaka, Japan, 2019, doi: 10.1109/icarm.2019.8834333.

[20] Hall, P.—Marron, J. S.—Neeman, A.: Geometric Representation of High Dimension, Low Sample Size Data. Journal of the Royal Statistical Society, Series B, Vol. 67, 2005, No. 3, pp. 427–444, doi: 10.1111/j.1467-9868.2005.00510.x.

[21] Jung, S.—Marron, J. S.: PCA Consistency in High Dimension, Low Sample Size Context. The Annals of Statistics, Vol. 37, 2009, No. 6B, pp. 4104–4130, doi: 10.1214/09-aos709.

[22] Li, X.—Lin, S.—Yan, S.—Xu, D.: Discriminant Locally Linear Embedding with High-Order Tensor Data. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), Vol. 38, 2008, No. 2, pp. 342–352, doi: 10.1109/tsmcb.2007.911536.

[23] Schölkopf, B.—Smola, A. J.—Müller, K.-R.: Kernel Principal Component Analysis. In: Burges, C. J. C., Schölkopf, B., Smola, A. J. (Eds.): Advances in Kernel Methods. MIT Press, Cambridge, MA, USA, 1999, pp. 327–352.

[24] Zou, H.—Hastie, T.—Tibshirani, R.: Sparse Principal Component Analysis. Journal of Computational and Graphical Statistics, Vol. 15, 2006, No. 2, pp. 262–286, doi: 10.1198/106186006x113430.

[25] Ng, A. Y.—Jordan, M. I.—Weiss, Y.: On Spectral Clustering: Analysis and an Algorithm. In: Dietterich, T. G., Becker, S., Ghahramani, Z. (Eds.): Advances in Neural Information Processing Systems 14 (NIPS 2001), MIT Press, 2001, pp. 849–856.

[26] Yeung, K. Y.—Ruzzo, W. L.: Principal Component Analysis for Clustering Gene Expression Data. Bioinformatics, Vol. 17, 2001, No. 9, pp. 763–774, doi: 10.1093/bioinformatics/17.9.763.

[27] Mishra, D.—Dash, R.—Rath, A. R.—Acharya, M.: Feature Selection in Gene Expression Data Using Principal Component Analysis and Rough Set Theory. In: Arabnia, H., Tran, Q. N. (Eds.): Software Tools and Algorithms for Biological Systems. Springer, New York, Advances in Experimental Medicine and Biology, Vol. 696, 2011, pp. 91–100, doi: 10.1007/978-1-4419-7046-6_10.

[28] Sato-Ilic, K.: Structural Classification Based Correlation and Its Application to Principal Component Analysis for High-Dimension Low-Sample Size Data. 2012 IEEE International Conference on Fuzzy Systems, 2012, doi: 10.1109/fuzz-ieee.2012.6251200.

[29] Yata, K.—Aoshima, M.: Effective PCA for High-Dimension, Low-Sample-Size Data with Noise Reduction via Geometric Representations. Journal of Multivariate Analysis, Vol. 105, 2012, No. 1, pp. 193–215, doi: 10.1016/j.jmva.2011.09.002.

[30] Ramsay, J. O.—Silverman, B. W.: Applied Functional Data Analysis: Methods and Case Studies. Springer, New York, Springer Series in Statistics, 2002, doi: 10.1007/b98886.

[31] Shen, D.—Shen, H.—Zhu, H.—Marron, J. S.: The Statistics and Mathematics of High Dimension Low Sample Size Asymptotics. Statistica Sinica, Vol. 26, 2016, No. 4, pp. 1747–1770, doi: 10.5705/ss.202015.0088.

[32] Pascual-Montano, A.—Carmona-Saez, P.—Chagoyen, M.—Tirado, F.—Carazo, J. M.—Pascual-Marqui, R. D.: bioNMF: A Versatile Tool for Non-Negative Matrix Factorization in Biology. BMC Bioinformatics, Vol. 7, 2006, Art. No. 366, doi: 10.1186/1471-2105-7-366.

[33] Berry, M. W.—Browne, M.—Langville, A. N.—Pauca, V. P.—Plemmons, R. J.: Algorithms and Applications for Approximate Nonnegative Matrix Factorization. Computational Statistics and Data Analysis, Vol. 52, 2007, No. 1, pp. 155–173, doi: 10.1016/j.csda.2006.11.006.

[34] Danaee, P.—Ghaeini, R.—Hendrix, D. A.: A Deep Learning Approach for Cancer Detection and Relevant Gene Identification. In: Altman, R. B., Dunker, A. K., Hunter, L., Ritchie, M. D., Murray, T. A., Klein, T. E. (Eds.): Proceedings of the Pacific Symposium on Biocomputing 2017 (Biocomputing 2017), 2017, pp. 219–229, doi: 10.1142/9789813207813_0022.

[35] Hinton, G. E.—Srivastava, N.—Krizhevsky, A.—Sutskever, I.—Salakhutdinov, R. R.: Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. 2012, arXiv: 1207.0580.

[36] Zhang, D.—Zhou, Z. H.—Chen, S.: Semi-Supervised Dimensionality Reduction. Proceedings of the 2007 SIAM International Conference on Data Mining (SDM 2007), 2007, pp. 629-634, doi: 10.1137/1.9781611972771.73.

[37] Rasmus, A.—Valpola, H.—Honkala, M.—Berglund, M.—Raiko, T.: Semi-Supervised Learning with Ladder Network. 2015, arXiv: 1507.02672.

[38] Quint, E.—Wirka, G.—Williams, J.—Scott, S.—Vinodchandran, N. V.: Interpretable Classification via Supervised Variational Autoencoders and Differentiable Decision Trees. 6th International Conference on Learning Representations (ICLR 2018), 2018.

[39] Halko, N.—Martinsson, P. G.—Tropp, J. A.: Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. SIAM Review, Vol. 53, 2011, No. 2, pp. 217–288, doi: 10.1137/090771806.

[40] Hoffman, M. D.—Blei, D. M.—Bach, F.: Online Learning for Latent Dirichlet Allocation. In: Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., Culotta, A. (Eds.): Advances in Neural Information Processing Systems 23 (NIPS 2010), 2010, pp. 856–864.

[41] Hoffman, M. D.—Blei, D. M.—Wang, C.—Paisley, J.: Stochastic Variational Inference. Journal of Machine Learning Research, Vol. 14, 2013, No. 4, pp. 1303–1347.

[42] Mairal, J.—Bach, F.—Ponce, J.—Sapiro, G.: Online Dictionary Learning for Sparse Coding. Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09), 2009, pp. 689–696, doi: 10.1145/1553374.1553463.

[43] Kingma, D. P.—Welling, M.: Auto-Encoding Variational Bayes. International Conference on Learning Representations, 2014, arXiv: 1312.6114, doi: 10.1561/2200000056.

[44] Kullback, S.—Liebler, R. A.: On Information and Sufficiency. The Annals of Mathematical Statistics, Vol. 22, 1951, No. 1, pp. 79–86, doi: 10.1214/aoms/1177729694.

[45] Rezende, D. J.—Mohamed, S.—Wierstra, D.: Stochastic Backpropagation and Approximate Inference in Deep Generative Models. Proceedings of the 31st International Conference on Machine Learning, Proceedings of Machine Learning Research, Vol. 32, 2014, No. 2, pp. 1278–1286, arXiv: 1401.4082.

[46] Ester, M.—Kriegel, H.-P.—Sander, J.—Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), AAAI Press, 1996, pp. 226–231.

[47] Pelleg, D.—Moore, A. W.: X-Means: Extending K-Means with Efficient Estimation of the Number of Clusters. Proceedings of the 17th International Conference on Machine Learning (ICML '00), 2000, pp. 727–734.

[48] Mahmud, M. S.—Huang, J. Z.—Fu, X.: Variational Autoencoder-Based Dimensionality Reduction for High-Dimensional Small-Sample Data Classification. Interna-

tional Journal of Computational Intelligence and Applications, Vol. 19, 2020, No. 1, Art. No. 2050002, doi: 10.1142/S1469026820500029.

[49] Masud, M. A.—Huang, J. Z.—Wei, C.—Wang, J.—Khan, I.—Zhong, M.: I-Nice: A New Approach for Identifying the Number of Clusters and Initial Cluster Centres. Information Sciences, Vol. 466, 2018, pp. 129–151, doi: 10.1016/j.ins.2018.07.034.

[50] Manning, C. D.—Raghavan, P.—Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, 2008.

[51] Rand, W. M.: Objective Criteria for the Evaluation of Clustering Methods. Journal of the American Statistical Association, Vol. 66, 1971, No. 336, pp. 846–850, doi: 10.1080/01621459.1971.10482356.

[52] Strehl, A.—Ghosh, J.: Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research, Vol. 3, 2002, pp. 583–617.

[53] Liu, B.—Wei, Y.—Zhang, Y.—Yang, Q.: Deep Neural Networks for High Dimension, Low Sample Size Data. Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-17), Melbourne, Australia, 2017, pp. 2287–2293, doi: 10.24963/ijcai.2017/318.

Mohammad Sultan Mahmud is currently Ph.D. candidate at the Shenzhen University, China. He received his Master's degree from the King Mongkut's University of Technology North Bangkok, Thailand, in 2014. He was awarded the Outstanding Doctoral Student of Shenzhen University in 2017 and Shenzhen Universiade International Scholarship in 2018. His current research focuses on big data mining and distributed and parallel computing.

Joshua Zhexue Huang received his Ph.D. degree from the Royal Institute of Technology, Sweden, in 1993. He is Distinguished Professor of the College of Computer Science and Software Engineering at Shenzhen University. Also, he is the Director of Big Data Institute and the Deputy Director of the National Engineering Laboratory for Big Data System Computing Technology. His main research interests include big data technology and applications. He has published over 200 research papers in conferences and journals. In 2006, he received the most influential paper award in the First Pacific-Asia Conference on Knowledge Discovery and Data Mining. He is known for his contributions to the development of a series of k-means type clustering algorithms in data mining, such as k-modes, fuzzy k-modes, k-prototypes, and w-k-means, that are widely cited and used, and some of which have been included in commercial software. He has extensive industry expertise in business intelligence and data mining, and has been involved in numerous consulting projects in Australia and China.

Xianghua Fu received his Ph.D. degree in computer science and technology from the Xi'an Jiaotong University, China, in 2005 and his M.Sc. degree from the Northwest A & F University, China, in 2002. Currently, he is Professor at the College of Big Data and Internet, Shenzhen Technology University, Shenzhen, China. He led a project of the National Natural Science Foundation, hosts a project of the Natural Science Foundation of Guangdong Province, and several projects of the Science and Technology Foundation of Shenzhen City. His research interests include machine learning, information retrieval, and natural language processing.

Rukhsana Ruby received her Master's degree from the University of Victoria, Canada, in 2009, and her Ph.D. degree from The University of British Columbia, Canada, in 2015. She has authored nearly 60 technical papers of well-recognized journals and conferences. Her research interest includes the management and optimization of next-generation wireless networks. She was a recipient of several awards or honors, notable among which are the Wait-listed for Canadian NSERC Postdoctoral Fellowship, the IEEE Exemplary Certificate (IEEE Communications Letters in 2018 and IEEE Wireless Communications Letters in 2018).

Kaishun Wu received his Ph.D. degree from the Hong Kong University of Science and Technology (HKUST), in 2011. From 2013, he is Distinguish Professor at the Shenzhen University, China. He has co-authored 2 books and published over 100 research papers in international journals and conferences, such as the IEEE Transactions on Mobile Computing, the IEEE Transactions on Parallel and Distributed Systems, ACM MobiCom, and IEEE INFOCOM. He holds 6 U.S. patents and has over 90 Chinese pending patents. He was a recipient of the 2012 Hong Kong Young Scientist Award and 2014 Hong Kong ICT awards and the 2014 IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award. He is Fellow of the IET and IEEE Senior Member.

Computing and Informatics, Vol. 40, 2021, 29–56, doi: 10.31577/cai_2021_1_29

BIG DATA ANALYTICS FOR ENERGY CONSUMPTION PREDICTION IN SMART GRID USING GENETIC ALGORITHM AND LONG SHORT TERM MEMORY

Sanju Kumari, Neeraj Kumar, Prashant Singh Rana

Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala, India
e-mail: {skumari_phd16, neeraj.kumar, prashant.singh}@thapar.edu

Abstract. Smart Grids (SG) have smart meters and advance metering infrasturutre (AMI) which generates huge data. This data can be used for predicting energy consumption using big data analytics. A very limited work has been carried out in the literature which shows the utilization of big data in energy consumption prediction. In this paper, the proposed method is based on Genetic Algorithm – Long Short Term Memory (GA-LSTM). LSTM memorises values over an arbitrary interval that manages time series data very effictively while GA is an evolutionary process that is used for optimization. GA combines with LSTM to process hyperparameters such as hidden layers, epochs, data intervals, batchsize and activation functions. Hence, GA creates a new vector for optimum solution that provides minimum error. These methods provide the best performance when compared with existing benchmarks. Moreover, GA-LSTM is used in a multi-threaded environment which increases the speed of convergence. Here, the multi-core platform is operated for solving one-dimensional GA-based inverse scattering problems. The result shows that GA-LSTM provides better convergence as compared to random approach techniques. For validating the results, Pennsylvania-New Jersey-Maryland Interconnection (PJM) energy consumption data has been used while adopting different performance evaluation metrics.

Keywords: Big data, deep learning, energy consumption prediction, genetic algorithm, load forecasting, long short term memory, multi-threading, smart grid

1 INTRODUCTION

Smart Grid (SG) is a technologically evolved electrical grid. It incorporates information technology into the existing grid and enables two way communication between the electric utility and the end consumer. The physical infrastructure is replaced with a digital one and conventional analog technologies are replaced with improved digital and power electronics. This technology makes the existing grid more efficient and reliable by reducing the number of outages that adds to the grid a self-healing or auto restore capability. Power is immediately rerouted when an outage occurs and power is restored to the affected area. Further, it promotes the use of renewable energy resources which reduces the carbon footprint. Also, SG being technologically advanced, consists of various energy measurement devices such as smart meter and advance metering infrastructure (AMI). These appliances generate huge data which can be termed as big data [24]. The generation of huge data also depends on other equipment such as supervisory control and data acquisition (SCADA) and phasor measurement unit (PMU), which generates data in seconds [11]. Since, there is a large number of measuring devices, data generation needs to be handled in a very efficient way. Therefore, big data management becomes an important task in SG. Moreover, many other tasks can be done using this data and one amongst them is energy consumption prediction.

The demand for energy increases due to economic and population growth. This growth can lead to an increased supply and demand gap, if not predicted well, beforehand. Hence, for proper utilisation of energy, big data analytics play a large role, and energy prediction can be one of the ways to reduce the demand and supply gap. Moreover, for the stability of demand and response, load forecasting has a necessary role to play in the SG system [32]. Many researchers have tried to achieve reliable and efficient energy management through big data techniques. For getting such type of energy management system, they combined data analytics and a scalable selection procedure so that the prediction of supply and consumption of energy could be stable. Big data analytics and cloud computing have been described for managing supply and demand of energy in SG [8]. For managing data, researchers illustrated various big data techniques in SG.

Big data analysis is a critical challenging task and can be overcome by various smart tools and techniques such as support vector machine (SVM) and decision tree analysis (DTA). In a similar line, Wang et al. discussed short-term load forecasting which is based on the recurrent neural network (RNN) and long short term memory (LSTM) [31]. RNN is rather an enhancement of the artificial neural network (ANN) and it is useful for processing the output directly to the first layer. In another case, LSTM is a part of deep learning and it overcomes the drawbacks of RNN. For energy forecasting, LSTM technique plays an important role by analysing the time series data. It uses big data strategies to reduce the storage space as well as analyse the data for taking decision on different models and make several frameworks [27]. Similarly, Pasini et al. suggested encoder-predictor for short-term load forecasting as an effective energy prediction [20].

Few authors used deep neural network (DNN) for energy forecasting. Amarasinghe et al. discussed demand side management using DNN [2]. In this paper, authors tried to discover an intelligent management of energy system and smart load distribution that focused on real time pricing. In another paper, Mohammad et al. defined the energy load forecasting model, which is based on DNN [19]. Furthermore, power demand forecasting using LSTM Neural Network is discussed in [4]. Here, LSTM provides a better performance as compared to the existing work. Few authors have analysed the DNN and Genetic Algorithm (GA) and concluded that this combination provides a better performance. In addition, various authors applied optimal RNN-LSTM model for energy forecasting. In this approach, Residual Network (ResNet) and LSTM have been used to develop the forecasting approach [5]. LSTM-RNN model is largely used in energy forecasting for small datasets. Using this approach, few authors used LSTM-RNN based day ahead load forecasting [28] using smart meter data of different localities. In a similar work, Sainath et al. discussed about short term load forecasting which is based on CNN and LSTM [25]. Many authors illustrated various applications, models and challenges in predicting energy. To overcome these challenges different machine learning models have been used. In [3], authors described statistical based modeling, machine learning and deep learning based model. Further, Diamantoulakis et al. suggested a prediction model for energy which is based on dynamically demand response in SG [1]. They suggested a dynamic energy managenment so that sufficient energy can be managed and further, cost can be reduced.

1.1 Related Works

Energy consumption prediction has a great role to play in maintaining the demand and supply gap in SG. It provides better decisions for power utility. Since, energy prediction is a time series data, it is desirable to work on techniques where challenges of big data can be handled by minimising the error between actual and predicted value. In this context, Rashid used smart meter data and developed big data analytics techniques for analyzing time series data. [23]. However, the author has taken a small dataset and compared it with other techniques which are not effectively considered. A very limited work with respect to energy forecasting using big data analytics has been done using the exiting methods such as the backward propogation neural network, support vector regression (SVR), generalized radial basis function neural network and multiple linear network. In a different work, Khuri et al. described 0/1 multiple knapsac problem [14] where proposed technology works on historical data. However, the authors have not used big data analytics. Few authors work on a similar line of energy management and they tried to improve the prediction of the energy consumption using CNN and Bi-directional LSTM (Bi-LSTM) neural network [16]. They applied electrical energy consumption prediction using Bi-LSTM model for improving results. In this approach, authors used a small dataset. Other researchers described dynamic test data generation using GA in energy prediction strategy [18]. However, none of them have used large datasets.

Sulaiman et al. used smart meter data and solved big data analytics using adaptive neuro-fuzzy inference system [29]. They used this data to predict the day ahead scheduling and verified the prediction accuracy to 84.03 %. In a very close work, Teres used MapReduce algorithm and developed histogram visualisation for SG [30]. However, the research was not intended towards energy prediction. Simhan et al. discussed cloud based approach for dynamic demand response for the SG [26]. However, authors did not focus on energy forecasting. In a different approach, Kaur et al. tried to elaborate LSTM based regression approach to solve the energy management of smart homes [13]. They verified the results with the existing techniques and for validation of the results, data was taken for 112 houses. Furthermore, Couceiro et al. made a stream analytics for energy prediction [7]. They used data streaming for handling large datasets for real time applications in power systems. However, their work was not validated to a real time data stream. A short term load forecasting using LSTM-RNN has been used in the SG [15]. Here, authors validated the result for a single household to forecast the load.

In very recent research, Zhang et al. used SVR and adaptive GA to optimize the parameters to get the best load forecasting model [33]. They performed and validated their results on a specific ratio value using very small datasets. In a similar work, Eseye et al. proposed machine learning tools based on binary GA [9]. They applied feature selection process and Gaussian process regression for measuring the fitness score. A similar approach is discussed in [17], where authors used hybrid model of GA and LSTM. They used half-hourly data from the australian energy market operator. However, their testing and training datasets were verified on small datasets. In another paper, authors used GA-ANN techniques for wind forecasting [10]. In this paper, the authors used meteorological data and compared double-stage back propagation trained ANN. In a similar work, Jaidee et al. presented a method for finding optimal parameters of a deep learning model by GA [12]. They tested the results with many other techniques including LSTM. However, their validation was limited to small datasets.

1.2 Motivation

Load forecasting is a difficult task in SG due to its complex and nonlinear relationship with different datasets. Different data mining and machine learning techniques have been adopted by the researchers but very few have taken large datasets to validate their proposal. Massive use of classification and regression analysis still poses a challenge at the implemetation level when large data is considered. From the literature review, it has been observed that very little work has been done with respect to big data techniques for energy prediction. It has also been observed that when large data is involved, time series data cannot be handled using conventional machine learning tools. Further, load forecasting techniques involve large datasets and to get early convergence we need some optimization tools, along with LSTM. Few authors proposed a different algorithm to develop load forecasting with big data but none have analysed the results in terms of multi-threading approach of

GA-LSTM which increases the speed of the convergence. Further, energy prediction is one of the techniques to understand the proper utilisation of the energy resources and therefore, we need to analyse the big data and use it for load forecasting. Proper load forecasting may reduce the supply and demand gap of electrical usage.

1.3 Contributions

In this paper, a multi-layer GA-LSTM model is proposed for energy prediction. It provides a better result as compared to existing techniques. The purpose of using GA is to optimize the parameters of the LSTM. To verify the effectivness of the proposed system, different parameters of LSTM have been used for reducing the errors. The major contributions of this paper are as follows.

- Multi-threaded based GA-LSTM technique is used for improving the performance of the algorithm with overall execution time.
- After identifying the lower and upperbound of the LSTM parameter, GA is used to optimize the LSTM for better performance.
- To validate the performance of GA-LSTM approach for large data, real time data of PJM has been used to validate the results with different evaluation metrics.
- To find the interval size of the optimal data, that gives minimum mean square error.

1.4 Organisation

Section 2 explains the dataset along with the performance and evaluation parameters. Section 3 provides the methodology of the proposed work. Section 4 outlines the results and discussions. Finally, the paper is concluded in Section 5.

2 DATASET AND ITS DESCRIPTION

2.1 Data Description

The dataset is a multivariate time-series data collected from Pennsylvania-New Jersey-Maryland Interconnection (PJM) which is a regional transmission organization (RTO) in the United States of America [21]. PJM is a part of the Eastern Interconnection grid operating an electric transmission system serving all parts of Delaware, Illinois, New Jersey and North Carolina. The hourly energy consumption data comes from PJM's website and are in megawatt-hours (MWh). The dataset is a daily and weekly based time series data. The dataset is of the PJM East that consists of data from 2002–2018 for the entire eastern region where 2002 to 2015 is used for training and 2015 to 2018 is used for testing [22].

Energy consumption has unique characteristics. The regions have changed over the years, so data may only appear for certain dates per region. GA-LSTM model is applied on these large datasets. The values of variables are compared between actual and predicted values. Since, hourly based data is very complex which is not suitable for LSTM model, therefore, focus was laid on daily and weekly based data. This data is compatible for GA-LSTM model, and provided more than 90 percent of result accuracy. For validation of the proposed work, three types of datasets are used and they are hourly, daily and weekly and the sample data is mentioned in Tables 1, 2 and 3, respectively.

SN	Date	Time (hrs)	Energy (MWh)
1	01/01/2002	1.00	14 107
2	01/01/2002	2.00	14 410
3	01/01/2002	3.00	15 174
4	01/01/2002	4.00	15 261
5	01/01/2002	5.00	14 774
6	01/01/2002	6.00	14 363
7	01/01/2002	7.00	14 045
8	01/01/2002	8.00	13 478
9	01/01/2002	9.00	12 892
10	01/01/2002	10.00	14 097

Table 1. Hourly sample dataset of energy consumption

SN	Date	Energy (MWh)
1	01/01/2006	363 822
2	02/01/2006	389 012
3	03/01/2006	431 551
4	04/01/2006	439 618
5	05/01/2006	388 212
6	06/01/2006	392 685
7	07/01/2006	394 595
8	08/01/2006	393 980
9	09/01/2006	417 416
10	10/01/2006	444 514

Table 2. Daily sample dataset of energy consumption

2.2 Performance Measures Used in This Energy Forecasting

2.2.1 Mean Absolute Error (MAE)

MAE is a measurement of errors between two variables such as x and y. The observations are expressed about the same event. It is expressed as per the following

SN	Year	Week	Energy (MWh)
1	2006	1	2 799 495
2	2006	2	2 986 229
3	2006	3	2 884 968
4	2006	4	2 644 030
5	2006	5	2 614 028
6	2006	6	2 614 028
7	2006	7	2 562 487
8	2006	8	2 562 487
9	2006	9	2 356 473
10	2006	10	2 349 789

Table 3. Weekly sample dataset of energy consumption

equation:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|a_i - p_i| \tag{1}$$

where n is number of observations, a is actual energy consumption and p is the predicted energy consumption.

2.2.2 Mean Square Error (MSE)

Mean square error (MSE) is an estimator which measures the average of the **squares of errors**. Here, average square provides the difference between the predicted value and the actual value. MSE is given as follows.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(a_i - p_i)^2 \tag{2}$$

where p_i indicates predicted value and a_i indicates actual value.

2.2.3 Median Absolute Error (MDAE)

The median absolute error is very crucial due to its robust nature of tackling outliers. Here, the loss is calculated by taking the median of all absolute differences between the actual and the predicted value. In the below equation, p_i is the predicted value of the i^{th} sample and a_i is the corresponding true value. MDAE estimated over n samples is defined as follows:

$$MDAE(a, p) = median(|a_1 - p_1|, \ldots, |a_n - p_n|). \tag{3}$$

2.2.4 Correlation

Correlation describes the statistical relationships between actual and predicted values. It is defined as follows:

$$r = \frac{\sum_{i=1}^{n}(a_i - \bar{a})(p_i - \bar{p})}{\sqrt{\sum_{i=1}^{n}(a_i - \bar{a})^2 \sum_{i=1}^{n}(p_i - \bar{p})^2}} \tag{4}$$

where r is the correlation, a is the actual value, p is the predicted value, $\bar{a}$ is the mean of all actual values, $\bar{p}$ is the mean of all predicted values and n is the number of instances. Correlation lies in the $[-1, 1]$ interval and is considered to have good correlations, if its value tends towards 1 or -1. In this paper, LSTM model is trained on 70 % of the dataset and testing is done on remaining 30 % of dataset. The trained LSTM model generates the predicted values that are compared with actual values. To understand the relationship between actual and predicted values, correlation is the best parameter. The correlation values lie between -1 and $+1$. The sign of the correlation denotes the nature of association and while the value denotes the strength of association.

2.2.5 Coefficient of Determination (R^2)

The coefficient of determination (R^2) summarizes the explanatory power of the regression model and is computed from the sums-of-squares terms and given as per the below equation:

$$R^2 = r * r \tag{5}$$

where r is the correlation as mentioned in Equation (4). R^2 lies in the $[0, 1]$ range and is considered to be good R^2, if its value tends towards 1.

3 METHODOLOGY

3.1 Proposed Work

The workflow of the complete system is shown in Figure 1. As can be seen from this figure, collected data is preprocessed and is divided into training and testing sets. Once the dataset is divided, LSTM model is trained with 70 % of the dataset and testing is done with remaining 30 % of the data. From the test data, prediction of consumed energy is obtained. Further, to improve the model, LSTM parameters are tuned with GA for calculating the evaluation points. The below subsections present modelling of LSTM, GA, GA-LSTM and multi-threading in GA-LSTM.

3.2 Long Short-Term Memory

LSTM is mainly used for time series dataset for prediction of energy. It works with the feedback connections and memorises previous information inside the network.

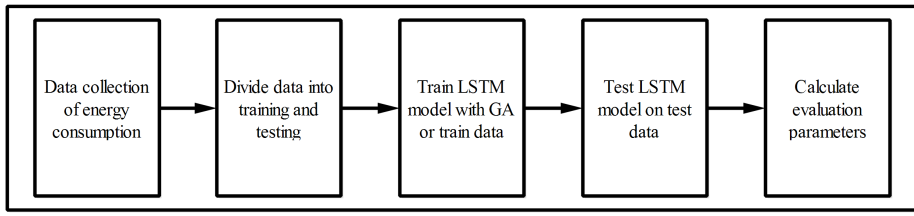


Figure 1. Workflow of the complete system

It has capability of solving time series and nonlinear prediction problems. The major problem of RNN is "long term dependancy", therefore, LSTM is used to overcome this problem. The cell state is the key of LSTM and it is like a conveyor belt. LSTM is capable of adding or removing the information and it is regulated by structures which are called gates. Gates are the mode where information is optionally chosen. These gates work with sigmoid activation function and a point to point multiplication operation. There are mainly three types of gates: input gate, output gate and forget gate. Tanh, sigma (σ) and Relu are the activation functions mainly used in the LSTM network. The below subsection describes the different techniques of LSTM for handling the large datasets.

3.2.1 Handling a Very Long Sequence Data with LSTM

LSTM is capable of learn and capturing of previous sequences of inputs. It can work nicely with one output, having many inputs but suffers if long input sequence exists. It is called sequence labeling or sequence classification. There are six modes of handling very long sequence data for classification problems. The starting point is to use the long sequence data as it is without any process. However, this may take long time to train. Further, attempt to back-propagate across extremely long input sequences may result in vanishing gradients, and in turn, an unlearnable model. A reasonable limit of 250–500 time steps is often used in practice with large LSTM models. Therefore, a way to handle these types of long sequence data is to simply truncate them. Here, removing a time steps from the beginning or at the end of input sequences is done. In some problem domains, it may be possible to summarize the input sequence. For example, in the case where input sequences are words, it may be possible to remove all words from input sequences that are above a specified word frequency such as and, &, the, and many more.

3.2.2 Process of LSTM

In this subsection, the step by step working of LSTM is explained. The first step in LSTM is to decide what information is to be selected from the cell state. This decision is taken by the forget gate which decides what information to keep and what

information to discard. Information from the input and previous hidden states is passed through a sigmoid function which squishes the values between 0 and 1. Values closer to 1 are kept and values closer to 0 are discarded.

The second step is to obtain the current cell state from the previous cell state and input. The previous hidden state and the input are passed through the input gate, which consists of the sigmoid function which squishes the values between 0 and 1 based on their importance. Values closer to 0 are not important while values closer to 1 are. The hidden state and the input are also passed through the tanh function which creates a candidate vector between 1 and -1, this regulates the network. The output of the input gate and the candidate vector is then multiplied. Finally, the obtained value is added to the product of the previous cell state and the forget vector to obtain the current cell state.

The third step decides what the new hidden state will be. The input and previous hidden state are passed through a sigmoid function to obtain the output. Next, the current cell state is passed through a tanh function. The obtained value and the output are then multiplied to decide what information the next hidden state carries. The product of this multiplication is the hidden state which is passed to the next LSTM cell along with the current cell state. The structure is shown in Figure 2.

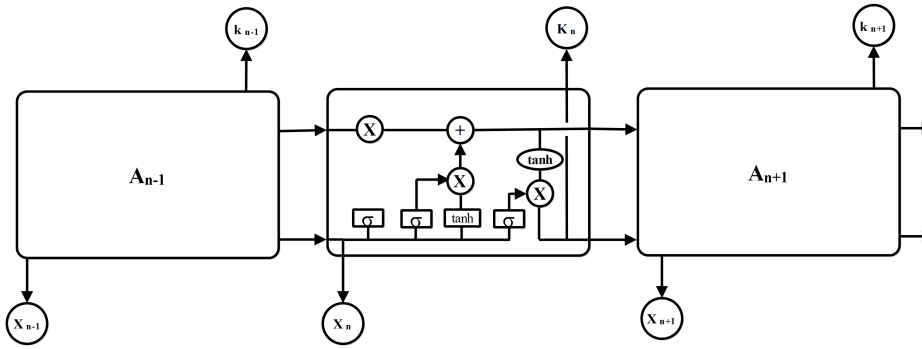


Figure 2. The operation of LSTM [6]

3.2.3 Modeling of LSTM

In this subsection, the mathematical modeling of LSTM cells is explained at every time step. LSTM cell contains several components such as forget gate F which decides what information should be thrown away or kept, a candidate layer $C?$ which holds all the possible values to be added to the cell state, an input gate I which is used to update the cell state and output gate O which decides what the next hidden state should be. Further, we represent the hidden state by H, and the cell state is represented by C and both of these are vectors. Current LSTM cell

is considered as the time step t. In the following equations '$*$' is an element-wise multiplication, '$+$' is an element-wise addition.

First, the input and previous hidden state are passed through the forget gate of the LSTM cell which has a sigmoid activation. It uses sigmoid activation because it needs to decide whether to forget information or not. The closer to 0 means to forget, and the closer to 1 means to keep.

$$F_t = \sigma(X_t * U_f + H_{t-1} * W_f) \tag{6}$$

where X_t is an input vector, U_f and W_f are the weight vectors for the forget gate and candidate gate respectively and H_{t-1} is the previous cell output or the hidden state. The new state of the LSTM is represented by follwing equation. We pass the hidden input and current input into tanh function to squish values between -1 and 1 which helps regulate the network.

$$C'_t = \tanh(X_t * U_c + H_{t-1} * W_t) \tag{7}$$

where C'_t is the current cell state at time step t, and it gets passed to next time step. H_{t-1} is the previous cell output and X_t is the input vector. The input gate is represented as per the below equation. We pass current input and previous hidden state into a sigmoid function that decides which values will be updated by transforming the values between 0 and 1. 0 means not important and 1 means important.

$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i) \tag{8}$$

where I_t is an input gate at time step of t, U_i and W_i are the weight vectors for the input gate and candidate gate, respectively, whereas H_{t-1} is the previous cell output. Output gate is represented as follows. Here the input vector and the previous hidden state are passed through a sigmoid function.

$$O_t = \sigma(X_t * (U_o + H_{t-1}) * W_o) \tag{9}$$

where O_t is an output gate at time step of t, X_t is an input vector, U_o and W_o are the weight vectors for the output gate and candidate gate, respectively, whereas H_{t-1} is the previous cell output. The current time step is mentioned as below.

$$C_t = f_t * C_{t-1} + l_t * C'_t \tag{10}$$

where C_t is current cell step at time step of t, f_t is a forget gate vector, I_t is the input gate. The current cell output is mentioned in Equation (11). This uses the output gate and cell state to give us the current hidden state.

$$H_t = O_t * \tanh(C_t) \tag{11}$$

where H_t is the current cell output at time step of t and $\tanh(C_t)$ is the activation function used to find the current cell state. Now with current memory state C_t, we

calculate new memory state from input state and C' layer.

$$C_t = C_t + I_t * C'_t \tag{12}$$

where C_t is the current cell state at time step t, and it gets passed to next time step and C'_t is new candidate gate. Now LSTM cell takes the previous memory state $C(t_1)$ and does element wise multiplication with forget gate F_t as per Equation (13).

$$C_t = C_{t-1} * F_t. \tag{13}$$

This output will be based on our cell state C_t but will be a filtered version. Therefore, we apply tanh to C_t and after this we make element wise multiplication with the output gate O and that will be our current hidden state H_t.

$$H_t = \tanh(C_t). \tag{14}$$

Now we pass C_t and H_t to the next time step and repeat the same process.

3.3 Genetic Algorithm (GA)

GA is based on the survival of the fittest, which was proposed by Darwin. Mainly five steps are involved in GA: initial population, selection operator, fitness function, crossover and mutation. The fitness function has great role in GA. Based on the requirements of LSTM, seven sets of chromosome samples are taken and they are data interval size, number of epochs, batchsize, number of hidden layers, dropout rate and number of units in each layer. The selected dimensions are used for processing GA-LSTM model. The results depend on fitness score which provides better result after comparing the value between predicted and actual value. Moreover, mutation and crossover have important role in this algorithm. Here, chromosomes work as a potential solution of target problem. It behaves as a binary string in a chromosome for processing the model. The chromosomes are generated randomly and the one which provides the best performance is selected. The basic process of the flow chart of a GA is shown in Figure 3.

3.4 Optimization in LSTM Network with GA

The operation of LSTM cell is shown in Figure 2 where three gates perform in coordination with each other. In these operation, LSTM is allowed to keep or forget information according to the requirements. This proposed work is divided into two stages. First stage is experimental part, where appropriate network parameters of the LSTM are designed. In the LSTM design, sequential input layer works on five hidden layers. By applying GA, optimal number of hidden neurons are found in each layer. GA searches the optimized hidden layers in LSTM model. In this model, tangent hyperbolic function is used for input nodes and hidden nodes. The range of tanh is $(-1$ to $1)$. The activation function of output node is designed as

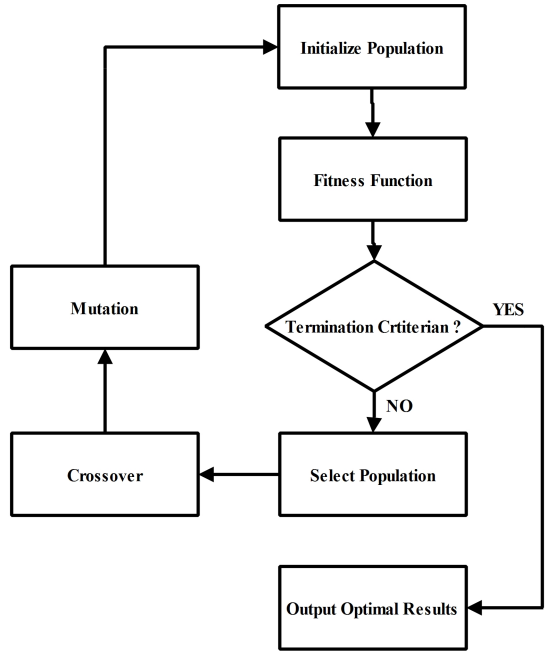


Figure 3. The flow chart of genetic algorithm

a non-linear function which works with the regression method. The objective of this model is to predict the energy consumption for the next year. The random values are set by the initial weight of the network.

In second stage, GA is combined with LSTM model, where fitness function is the main feature. GA is the evolutionary algorithm where initial population is selected on the basis of fitness function. At initial stage, population is generated randomly. After reproduction, best pairs of fitness score are selected. The experimental results depend on fitness score. Here, seven dimensions in one chromosome sample are created. Performance is measured through benchmark and GA-LSTM. This approach has an advantage in prediction of energy consumption with large dataset. The experimental result is compared with Mean Absolute Error (MAE), Mean Square Error (MSE), Median Absolute Error (MDAE), correlation, coefficient of determination and accuracy. GA-LSTM provides the optimal solution for large dimension data. Here, chromosomes are represented by strings of arrays and to obtain fitness value, MSE of the prediction model is used. The detailed algorithm is mentioned in Algorithm 1. This algorithm describes the use of GA to optimize the LSTM parameters. It uses crossover, mutation and selection of best chromosome that gives the best accuracy as fitness value. In step 1, GA parameters are initialized and in step 2 LSTM parameters are initialized. Similarly, Algorithm 2 describes random approach for LSTM parameters optimization. The fitness function (F) is defined as

Algorithm 1 Genetic Algorithm with LSTM

1. Initialize the GA parameters
   - cr = 0.9;
   - mr = 0.1;
   - iterations = 20;
   - popSize = 20;
2. Initialize LSTM parameters
   - d = 7;
   - dataInterval = $[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$;
   - nEpochs = $[50, 100, 150, 200, 250, 300, 350, 400, 450, 500]$;
   - batchSize = $[8, 16, 32, 64]$;
   - nHiddenLayer = $[2, 3, 4, 5]$;
   - dropoutRate = $[0.1, 0.2, 0.3, 0.4]$;
   - nUnits = $[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$;
   - nActivationFunction = ['relu', 'sigmoid', 'tanh']
3. t = 1
4. InitPop[$P(t)$]; Initializes the population
5. EvalPop[$P(t)$] = LSTM (chromosome); Evaluates the population

while stopping condition **do**
   Crossover()
   Mutation()
   MemoriseGlobalBest()
end while

6. Return the individual with the best fitness as the solution;

per the below equation.

$$F = \min(MSE(LSTM(x))) \tag{15}$$

where x is a vector of parameter and the sample chromosomes is like $x = [3, 30, 200, 32, \text{Relu}, 0.1, 30]$ which can be verified from Table 4. It returns the MSE between actual and predicted value of the testing dataset.

3.5 Multi-Threading in GA-LSTM

Multi-threading uses the CPU cache, translation lookaside buffer (TLB) cache and single core or multiple cores to carry out a wide range of tasks concurrently. It is a process in which the CPU provides multiple threads simultaneously for the execution of a task in a less amount of time. The CPU cache reduces the average data

Algorithm 2 Random approach with LSTM

1. Parameter initialization
   - iterations = 20;
   - bestchromosome = []
   - bestAccuracy = 0
2. Initialize LSTM parameters
   - d = 7;
   - datainterval = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100];
   - nEpochs = [50, 100, 150, 200, 250, 300, 350, 400, 450, 500];
   - batchSize = [8, 16, 32, 64];
   - nHiddenLayer = [2, 3, 4, 5];
   - dropoutRate = [0.1, 0.2, 0.3, 0.4];
   - nUnits = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100];
   - nActivationFunction = ['Relu', 'sigmoid', 'tanh'];
3. t = 1
4. While $t \leq$ iterations;

chromosome = Generate random set of LSTM parameters;
Evaluate accuracy = LSTM (chromosome);
if accuracy > bestAccuracy;
bestAccuracy = accuracy;
bestChromosome = chromosome;
$t = t + 1$

5. Return the bestAccuracy and bestChromosome as solutions.

access time from the main memory while TLB reduces the average time for memory allocation in the main memory. In GA-LSTM, data is loaded and the model is trained thereafter. These processes go step by step and the user needs to wait for their execution. But through multi-threading these tasks can be performed in parallel by running a number of threads which get queued and operate at a high speed without getting blocked. There are many benefits of multi-threading. Firstly, it eliminates the multiple processor subsystem and the hardware completely. Secondly, a single server can perform a number of tasks simultaneously by dispatching multiple threads at a time. This reduces the number of servers required while loading the large data. Thirdly, the applications run one after the other and wait for the former to get over. The latter applications do not get blocked, instead of that, they get queued and increase responsiveness to the operation. Finally, the memory to be allocated to processes is quite high if multi-threading is not used.

4 RESULTS

PJM dataset from 2002 to 2018 has been used to validate the results, wherein, dataset from 2002 to 2015 has been taken for training and dataset from 2015 to 2018 is taken for testing. GA-LSTM model is trained and tested and the validation of the proposed work is analysed. In the initial stage, the number of LSTM unit is formed into vectors of hidden layers, epochs, batch size, interval size and activation functions. In the proposed work, parameters of LSTM are optimized and verified for its effectiveness of the GA-LSTM model. The performance of the GA-LSTM network is measured using MAE, MSE, MDAE, correlation, coefficient of determination and accuracy. Comparsion of actual and predicted results was done and it was found that error is reduced using the proposed model. For validation of the proposed work, three types of datasets have been used hourly, daily and weekly whose sample data is mentioned in Section 3. Since, hourly based dataset is very complex and not suitable for GA-LSTM model, we have used daily and weekly based dataset in our work.

4.1 Experimental Setup and Simulation Parameters

The proposed algorithm uses Xeon Processor with 64 GB RAM (20 cores) and a 1 TB SSD. To increase the speed of the simulation, multi-threaded GA-LSTM algorithm is used. LSTM parmeters have been shown in Table 4. For better validation of the results, maximum of 5 hidden layers are used. It is seen from this table that as the data interval size increases, epoch is also increased. Further, three types of activation functions tanh, sigmoid and Relu are used. The purpose of using three types of activation functions is to verify the proposed methodology for large dataset. Further, these activation functions will give better choice while making crossover and mutation in GA. It can be seen in the table that dropout rate varies between 0.1 to 0.5 and the number of units are taken between 10 to 100 at an interval of 10.

SN	Name Parameters	Values
1	Number of hidden layers	$[2, 3, 4, 5]$
2	Data interval size	$[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$
3	Epochs	[50 to 500 with an interval of 50]
4	Batch size	$[8, 16, 32, 64]$
5	Activation Function	[Tanh, Sigmoid, Relu]
6	Dropout rate	$[0.1, 0.2, 0.3, 0.4, 0.5]$
7	Number of units	$[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$

Table 4. LSTM hyperparameters

GA parameters such as crossover, mutation, population size and number of iterations are mentioned in Table 5. It is seen from the table that the single point crossover is used. Another parameter is mutation where rate at single point is taken as 0.1. The size of the initial population is 100. Further, the maximum

number of iterations are taken as 20. The selection criteria used is roulette wheel. After optimizing the parameters of LSTM, the best parameters are found which are mentioned in Table 6. This table provides the best parameters for daily and weekly energy prediction and it can be observed that Relu activation function provides the best perfomance. Similarly, it can be seen that the optimized batch size is 16 for both daily and weekly energy prediction. Epochs are found to be 450 for both the cases. Optimized data interval size is 60 for both the cases. The details of other parameters of LSTM and Random are mentioned in Table 6.

SN	**Name of Parameters**	**Values**
1	Crossover rate	0.9 (Single point crossover)
2	Mutation rate	0.1 (Single point mutation)
3	Population size	100
4	Iteration	20

Table 5. GA parameters

Approach		LSTM		Random	
SN	**Parameters**	**Daily Energy Prediction**	**Weekly Energy Prediction**	**Daily Energy Prediction**	**Weekly Energy Prediction**
1	Number of hidden layers	4	3	3	2
2	Data interval size	60	60	50	40
3	Epochs	450	450	400	450
4	Batch size	16	16	8	16
5	Activation function	Relu	Relu	Relu	Relu
6	Dropout rate	0.3	0.2	0.3	0.2
7	Number of units	60	40	50	50

Table 6. Optimized parameters of LSTM and Random for daily and weekly energy consumption prediction using GA

Table 7 shows the experimental values of results with GA and Random approach. The performance metrics have been calculated for daily and weekly dataset with respect to MAE, MSE, MDAE, correlation, coefficient of determination (R^2) and accuracy. It can be seen from this table that accuracy of GA is 82.42 as compared to the random approach which is 51.26 for the daily dataset. Similarly for weekly dataset, accuracy of GA is 80.27 and random approach is 48.22. Further it can be observed that correlation and R^2 is better in GA as compared to random approach. The other evaluations parameters such as MAE, MSE and MDAE are also shown and it gives the best performance for the daily as well as weekly dataset in GA as compared to random approach. The acceptable error is mentioned as 10^3 and 10^4 for the daily and weekly dataset, respectively.

	Daily Energy Consumption		Weekly Energy Consumption	
Evaluation Parameters	**GA**	**Random**	GA	**Random**
Mean Absolute Error	5.34×10^2	1.27×10^3	1.35×10^3	2.23×10^4
Mean Squared Error	1.27×10^3	7.66×10^4	2.90×10^4	6.45×10^6
Median Absolute Error	7.46×10^1	9.80×10^2	4.20×10^2	8.31×10^3
Correlation	0.931	0.551	0.892	0.496
R^2	0.868	0.304	0.792	0.246
Accuracy	82.42*	51.26*	80.27@	48.22@

* with acceptable error of 10^3; @ with acceptable error of 10^4

Table 7. Performance and evaluation paramenters for daily and weekly dataset for Random and GA

4.2 Energy Predication on a Daily Dataset

GA-LSTM predication can be used for large datasets where GA is used to optimize the parameters of the LSTM. Figure 4 shows the energy consumption of actual versus predicted daily energy. Daily energy curve is given in MWh since PJM covers larger area of the USA. It is seen that predicted energy of PJM is very close to the actual energy. This prediction will help to schedule the generating units of PJM. LSTM uses 70 % of the data for training and 30 % for testing. This property of LSTM reduces the testing data. With GA approach, the energy consumption prediction on daily dataset gives much less error.

Figure 5 shows the convergence of daily energy prediction by using random approach and GA-LSTM approach. The convergence refers to different system moving towards performing the same task. Random sets approach is heuristic by nature hence, it is very helpful in discovering things themselves. In this graph, it is observed that random approach convergence takes larger iterations while GA approach takes 15 iterations to converge. The convergence graph is taken with MSE and it shows lower value as compared to random approach. This proves that GA provides optimized result with a fewer number of iterations and it converges at low iterations.

Figure 6 shows parameter sensitivity for different data interval size for daily energy consumption prediction. Parameter sensitivity analysis shows uncertainty in the output of a model. It can be used to validate with sources of uncertainty in the model input. Further, this is a method for finding or establishing the response of a model which changes when parameters are varied. This graph shows MSE versus data interval size. It is observed from the figure that MSE is low for data interval of size 60. Similarly, other optimized parameters can be seen from Table 6.

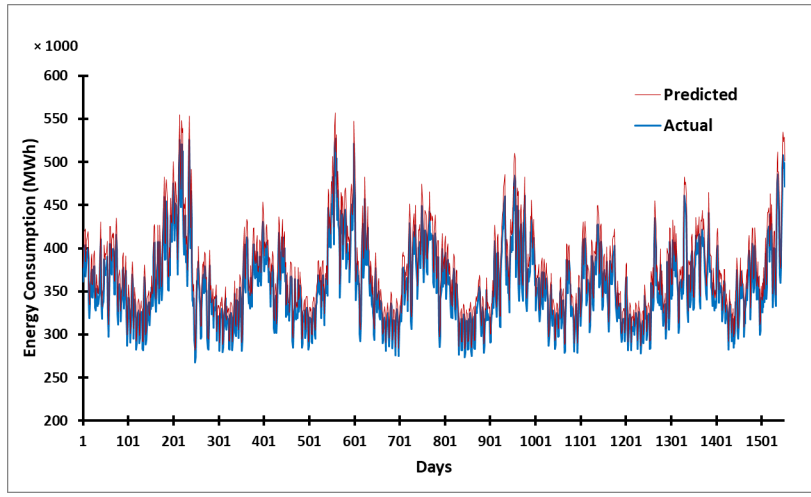


Figure 4. Actual vs. prediction of daily energy consumption

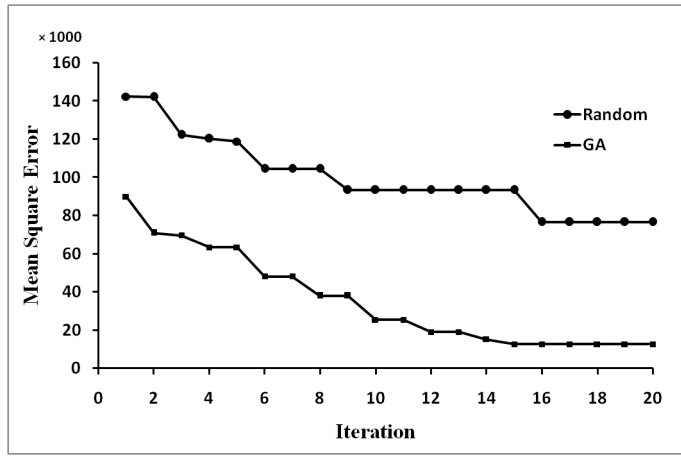


Figure 5. Convergence of daily energy consumption prediction

4.3 Energy Prediction on Weekly Dataset

Figure 7 represents weekly actual energy consumption versus predicted energy consumption. Weekly prediction is mostly done by the utility for week ahead scheduling of the generating units and it is one of the most widely used short term load forecasting in the SG. Since data is large, it can be seen from this figure that the energy is in 1 000 of MWh. It is also observed that actual versus predicted energy consumption are very close to each other with a very small error. Further, this energy

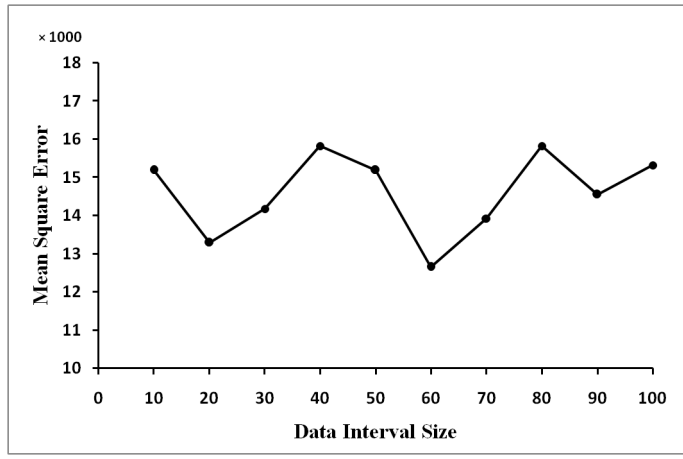


Figure 6. Parameter sensitivity for data interval size of daily energy consumption prediction

consumption prediction can be used to maintain the balance between demand and supply of the PJM. This prediction will save a large amount of money for utility and better utilization of the generating plants.

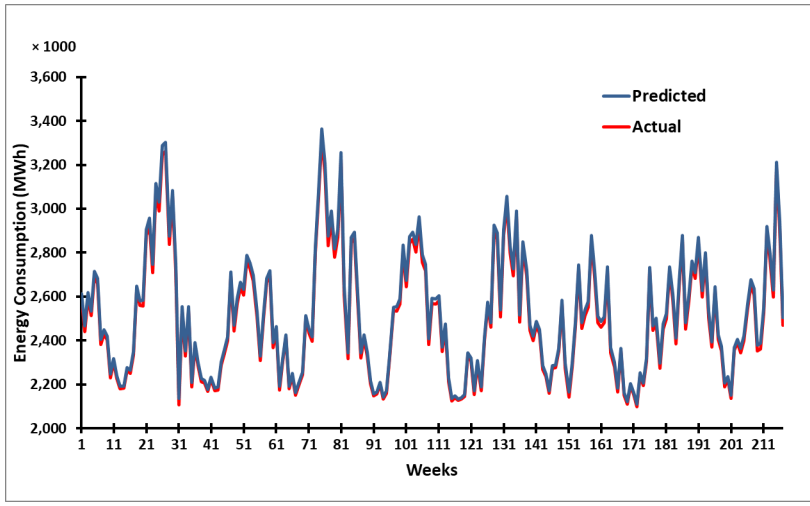


Figure 7. Actual vs. prediction of weekly energy consumption

The validation of the weekly energy forecasting is achieved through convergence graph. Figure 8 represents the convergence of weekly energy consumption prediction with GA and a random approach. This convergence graph is shown till 20^{th} iteration

and found that the random approach has slower convergence rate as compared to GA. Random approach converges at 15^{th} iteration while there is no certainity of convergence through the random approach. It is also observed that MSE has a higher value of the random approach as compared to GA.

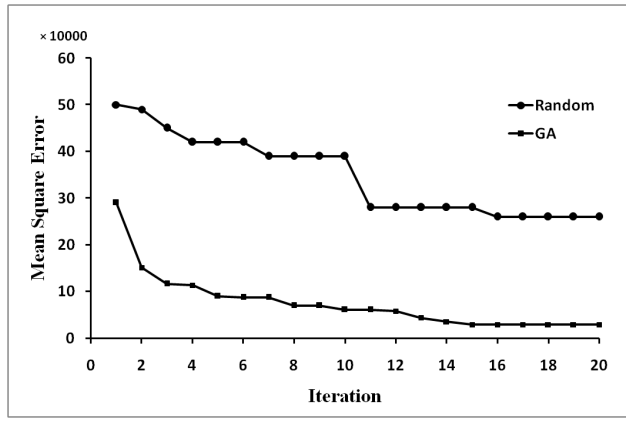


Figure 8. Convergence of weekly energy consumption prediction

Parameter sensitivity is a method for finding or establishing how responses of a model change when parameters are varied. There is great role of parameter sensitivity in optimization problem. Figure 9 shows parameter sensitivity with respect to MSE versus data interval size. It has been observed from the figure that MSE has a lower value at data interval of size 60. This proves that data interval size optimization is a very accurate method which will give a better convergence at lower iterations. Similarly, other optimized parameters can be seen from Table 6.

4.4 Multi-Threading

The competitive performance of multiple threads is shown in Table 8. Here, the program is run on a machine having 4 cores. Different number of threads are run which is starting from 1 to 8. As the number of threads are increasing from 1 to 4, the total execution time decreases, but as we increase the number of threads from 5 to 8, the total execution time increases and this is evident from Figure 10. Therefore, the optimal number of threads must be 4 to run the LSTM-GA program in a 4-core machine.

4.5 Variability of MSE with GA and Random Approach

To validate the performance of GA and random approach, box plot is shown in this subsection. Boxplot is a standardized way of displaying the variation of any quantity which emphasizes on stablity of the system. Further, we need to have

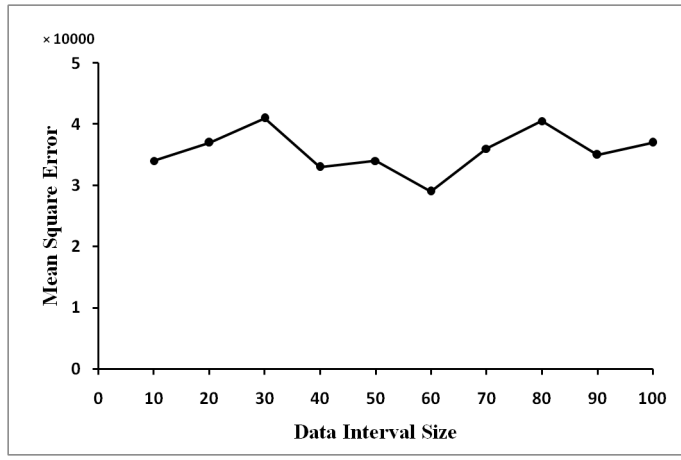


Figure 9. Parameter sensitivity for data interval size of weekly energy consumption prediction

Number of Threads	Time Taken in (sec)
1	822
2	545
3	476
4	364
5	390
6	490
7	600
8	900

Table 8. Time taken by GA-LSTM with different number of threads

information on the variability or dispersion of the data. A boxplot is a graph that gives a good indication of how the values in the data are spread out and also identify outliers. The variation of MSE with random approach and GA is shown to prove the efficiency of the proposed GA-LSTM algorithm. A total of 10 simulations are performed for daily and weekly energy consumption prediction. Figure 10 and Figure 11 present the box-plot for energy consumption for daily and weekly energy consumption prediction. Here, the variation of MSE in random approach is more than GA. It is found that GA variation is very less as compared to random approach which proves stability of MSE. Since there is less variation of MSE for the energy prediction using GA, it outperforms the random approach.

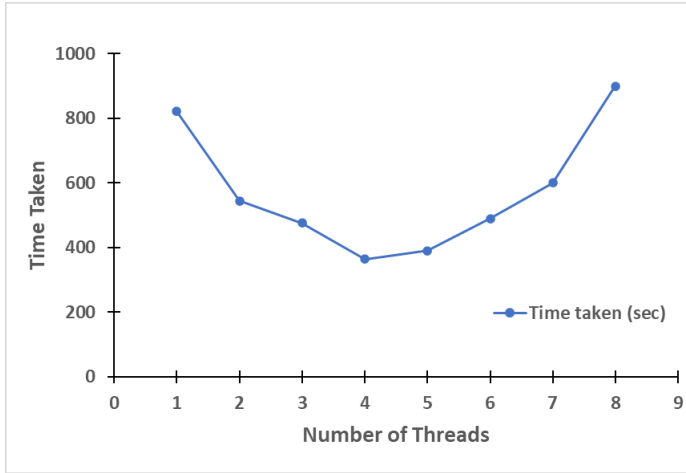


Figure 10. Multi-threading – number of threads vs. execution time

5 CONCLUSION

This paper introduces the prediction of energy consumption on real time large dataset obtained from PJM. It uses big data analytics using machine learning to predict the energy consumption for large dataset. To achieve this, mainly two mod-

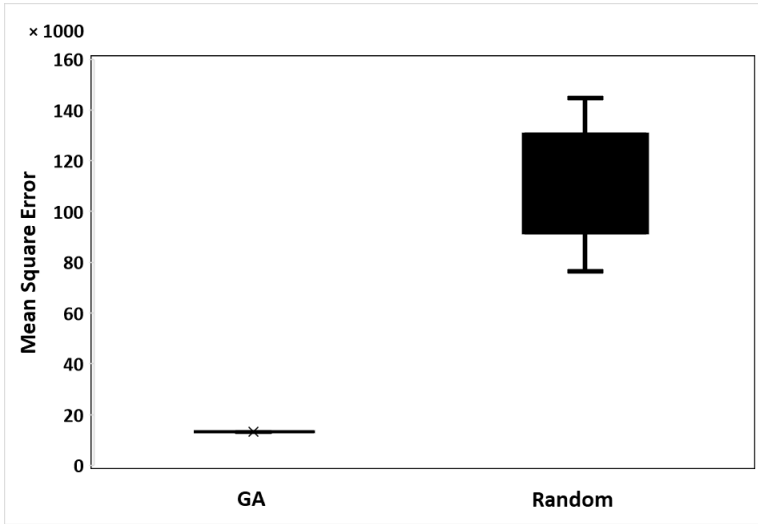


Figure 11. Comparison of genetic algorithm and random approach for mean absolute error – daily basis

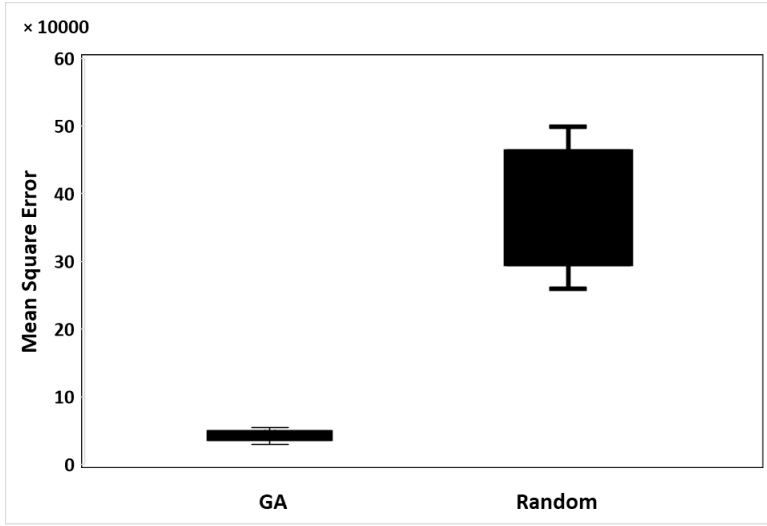


Figure 12. Comparison of genetic algorithm and random approach for mean absolute error – weekly basis

els – random and GA approach – are applied. On comparing the performance of both models, it was found that GA-LSTM outperforms the LSTM. Further, multi-threaded GA-LSTM is used to increase the speed of convergence. It has been observed that GA has higher accuracy as compared to random approach. The comparison is conducted experimentally for real datasets of PJM for daily and weekly energy consumption. It has been proved that GA-LSTM model provides optimized effective performance. The novelty of the paper lies in the multi-threaded based GA-LSTM technique used for improving the performance of the algorithm with overall low execution time. Further, after identifying the lower and upperbound of the LSTM parameter, GA is used to optimize LSTM for better performance. The results of the proposed work are verified with the variability of MSE and it was found that the proposed algorithm passes all the evaluation parameter checks.

REFERENCES

[1] Aman, S.—Frincu, M.—Chelmis, C.—Noor, M.—Simmhan, Y.—Prasanna, V. K.: Prediction Models for Dynamic Demand Response: Requirements, Challenges, and Insights. 2015 IEEE International Conference on Smart Grid Communications (SmartGridComm), 2015, pp. 338–343, doi: 10.1109/smartgridcomm.2015.7436323.

[2] Amarasinghe, K.—Marino, D. L.—Manic, M.: Deep Neural Networks for Energy Load Forecasting. 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), 2017, pp. 1483–1488, doi: 10.1109/ISIE.2017.8001465.

[3] Goodfellow, I.—Bengio, Y.—Courville, A.: Deep Learning. The MIT Press, 2017.

[4] Cheng, Y.—Xu, C.—Mashima, D.—Thing, V. L. L.—Wu, Y.: PowerLSTM: Power Demand Forecasting Using Long Short-Term Memory Neural Network. In: Cong, G., Peng, W. C., Zhang, W., Li, C., Sun, A. (Eds.): Advanced Data Mining and Applications (ADMA 2017). Springer, Cham, Lecture Notes in Computer Science, Vol. 10604, 2017, pp. 727–740, doi: 10.1007/978-3-319-69179-4_51.

[5] Choi, H.—Ryu, S.—Kim, H.: Short-Term Load Forecasting Based on ResNet and LSTM. 2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), 2018, pp. 1–6, doi: 10.1109/smartgridcomm.2018.8587554.

[6] Understanding LSTM Networks. Colah's Tutorial on LSTM. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2017.

[7] Couceiro, M.—Ferrando, R.—Manzano, D.—Lafuente, L.: Stream Analytics for Utilities. Predicting Power Supply and Demand in a Smart Grid. 2012 3rd International Workshop on Cognitive Information Processing (CIP), 2012, pp. 1–6, doi: 10.1109/cip.2012.6232904.

[8] Diamantoulakis, P. D.—Kapinas, V. M.—Karagiannidis, G. K.: Big Data Analytics for Dynamic Energy Management in Smart Grids. Big Data Research, Vol. 2, 2015, No. 3, pp. 94–101, doi: 10.1016/j.bdr.2015.03.003.

[9] Eseye, A. T.—Lehtonen, M.—Tukia, T.—Uimonen, S.—John, R. J.: Machine Learning Based Integrated Feature Selection Approach for Improved Electricity Demand Forecasting in Decentralized Energy Systems. IEEE Access, Vol. 7, 2019, pp. 91463–91475, doi: 10.1109/access.2019.2924685.

[10] Eseye, A. T.—Zhang, J.—Zheng, D.—Ma, H.—Gan, J.: Short-Term Wind Power Forecasting Using a Double-Stage Hierarchical Hybrid GA-ANN Approach. 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), 2017, pp. 552–556, doi: 10.1109/ICBDA.2017.8078695.

[11] Hu, J.—Vasilakos, A. V.: Energy Big Data Analytics and Security: Challenges and Opportunities. IEEE Transactions on Smart Grid, Vol. 7, 2016, No. 5, pp. 2423–2436, doi: 10.1109/tsg.2016.2563461.

[12] Jaidee, S.—Pora, W.: Very Short-Term Solar Power Forecasting Using Genetic Algorithm Based Deep Neural Network. 2019 4th International Conference on Information Technology (InCIT), 2019, pp. 184–189, doi: 10.1109/incit.2019.8912097.

[13] Kaur, D.—Kumar, R.—Kumar, N.—Guizani, M.: Smart Grid Energy Management Using RNN-LSTM: A Deep Learning-Based Approach. 2019 IEEE Global Communications Conference (GLOBECOM), 2019, pp. 1–6, doi: 10.1109/globecom38437.2019.9013850.

[14] Khuri, S.—Bäck, T.—Heitkötter, J.: The Zero/One Multiple Knapsack Problem and Genetic Algorithms. Proceedings of the 1994 ACM Symposium on Applied Computing (SAC '94), 1994, pp. 188–193, doi: 10.1145/326619.326694.

[15] Kong, W.—Dong, Z. Y.—Jia, Y.—Hill, D. Y.—Xu, Y.—Zhang, Y.: Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network. IEEE Transactions on Smart Grid, Vol. 10, 2019, No. 1, pp. 841–851, doi: 10.1109/tsg.2017.2753802.

[16] Le, T.—Vo, M. T.—Vo, B.—Hwang, E.—Rho, S.—Baik, S. W.: Improving Electric Energy Consumption Prediction Using CNN and Bi-LSTM. Applied Sciences, Vol. 9, 2019, No. 20, Art. No. 4237, doi: 10.3390/app9204237.

[17] Mamun, A.—Hoq, M.—Hossain, E.—Bayindir, R.: A Hybrid Deep Learning Model with Evolutionary Algorithm for Short-Term Load Forecasting. 2019 8th International Conference on Renewable Energy Research and Applications (ICRERA), 2019, pp. 886–891, doi: 10.1109/icrera47325.2019.8996550.

[18] Michael, C. C.—McGraw, G. E.—Schatz, M. A.—Walton, C. C.: Genetic Algorithms for Dynamic Test Data Generation. Proceedings of 12th IEEE International Conference Automated Software Engineering, 1997, pp. 307–308, doi: 10.1109/ASE.1997.632858.

[19] Mohammad, F.—Kim, Y. C.: Energy Load Forecasting Model Based on Deep Neural Networks for Smart Grids. International Journal of System Assurance Engineering and Management, Vol. 11, 2020, pp. 824–834, doi: 10.1007/s13198-019-00884-9.

[20] Pasini, K.—Khouadjia, M.—Samé, A.—Ganansia, F.—Oukhellou, L.: LSTM Encoder-Predictor for Short-Term Train Load Forecasting. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (Eds.): Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2019). Springer, Cham, Lecture Notes in Computer Science, Vol. 11908, 2019, pp. 535–551, doi: 10.1007/978-3-030-46133-1_32.

[21] PJM. Pennsylvania New Jersey Maryland Interconnection. `https://www.pjm.com`, 2020.

[22] PJM Dataset. PJM Time Series Analysis and Forecasting Data. `https://www.kaggle.com/brahimmebrek/pjm-east-eda-and-forecasting`, 2020.

[23] Rashid, M. H.: AMI Smart Meter Big Data Analytics for Time Series of Electricity Consumption. 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), 2018, pp. 1771–1776, doi: 10.1109/trustcom/bigdatase.2018.00267.

[24] Sagiroglu, S.—Terzi, R.—Canbay, Y.—Colak, I.: Big Data Issues in Smart Grid Systems. 2016 IEEE International Conference on Renewable Energy Research and Applications (ICRERA), 2016, pp. 1007–1012, doi: 10.1109/icrera.2016.7884486.

[25] Sainath, T. N.—Vinyals, O.—Senior, A.—Sak, H.: Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 4580–4584, doi: 10.1109/icassp.2015.7178838.

[26] Simmhan, Y.—Aman, S.—Kumbhare, A.—Liu, R.—Stevens, S.—Zhou, Q.—Prasanna, V.: Cloud-Based Software Platform for Big Data Analytics in Smart Grids. Computing in Science and Engineering, Vol. 15, 2013, No. 4, pp. 38–47, doi: 10.1109/mcse.2013.39.

[27] Stimmel, C.: Big Data Analytics Strategies for the Smart Grid. CRC Press, 2014.

[28] Sulaiman, S. M.—Jeyanthy, P. A.—Devaraj, D.: Artificial Neural Network Based Day Ahead Load Forecasting Using Smart Meter Data. 2016 Biennial International Conference on Power and Energy Systems: Towards Sustainable Energy (PESTSE), 2016, IEEE, pp. 1–6, doi: 10.1109/pestse.2016.7516422.

[29] Sulaiman, S. M.—Jeyanthy, P. A.—Devaraj, D.: Big Data Analytics of Smart Meter Data Using Adaptive Neuro Fuzzy Inference System (ANFIS). 2016 International Conference on Emerging Technological Trends (ICETT), 2016, pp. 1–5, doi: 10.1109/icett.2016.7873732.

[30] Teres, A. D.: Histogram Visualization of Smart Grid Data Using MapReduce Algorithm. 2019 2nd International Conference on Power and Embedded Drive Control (ICPEDC), 2019, pp. 307–312, doi: 10.1109/icpedc47771.2019.9036693.

[31] Wang, L.—Mao, S.—Wilamowski, B.: Short-Term Load Forecasting with LSTM Based Ensemble Learning. 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2019, pp. 793–800, doi: 10.1109/ithings/greencom/cpscom/smartdata.2019.00145.

[32] Wang, Y.—Chen, Q.—Kang, C.—Zhang, M.—Wang, K.—Zhao, Y.: Load Profiling and Its Application to Demand Response: A Review. Tsinghua Science and Technology, Vol. 20, 2015, No. 2, pp. 117–129, doi: 10.1109/tst.2015.7085625.

[33] Zhang, G.—Guo, J.: A Novel Method for Hourly Electricity Demand Forecasting. IEEE Transactions on Power Systems, Vol. 35, 2020, No. 2, pp. 1351–1363, doi: 10.1109/tpwrs.2019.2941277.

Sanju Kumari received her B.E. degree in computer science engineering from the Solapur University, Solapur, India in 2010 and M.E. degree in information technology from the Rajiv Gandhi Prodhiki Viswavidalaya, Bhopal, India in 2014. Presently, she is doing Ph.D. from the Thapar Institute of Engineering and Technology, Patiala, India. Her research interest is machine learning, deep learning, big data and smart grid.

Neeraj Kumar received his Ph.D. in CSE from the Shri Mata Vaishno Devi University, Katra (J & K), India, and was a postdoctoral research fellow in the Coventry University, Coventry, UK. He is working as Professor in the Department of Computer Science and Engineering, Thapar University, Patiala, India. He has published more than 100 technical research papers in leading journals and conferences from IEEE, Elsevier, Springer, John Wiley etc. He has guided many research scholars leading to Ph.D. and M.E. His research is supported by funding from TCS, DST and UGC.

Prashant Singh Rana is Assistant Professor in the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, India. He also worked as a Project Scientist at IIT Delhi, India. He received his Ph.D. from ABV Indian Institute of Information Technology and Management, Gwalior, India and his areas of research are machine learning, soft computing, combinatorial problems, and computational biology.

Computing and Informatics, Vol. 40, 2021, 57–82, doi: 10.31577/cai_2021_1_57

EFFECT OF TERM WEIGHTING ON KEYWORD EXTRACTION IN HIERARCHICAL CATEGORY STRUCTURE

Boonthida Chiraratanasopha, Salin Boonbrahm

Institute of Informatics, Walailak University
222 Thaiburi, Thasala District, Nakhonsithammarat 80161, Thailand
e-mail: {jboontida16, salil.boonbrahm}@gmail.com

Thanaruk Theeramunkong

Sirindhorn International Institute of Technology, Thammasat University
131 Moo 5, Tiwanont Road, Bangkadi Muang, Pathum Thani 12120, Thailand
&
Associate Fellow, The Royal Society of Thailand
Sanam Suea Pa, Dusit, Bangkok 10300, Thailand
e-mail: thanaruk@siit.tu.ac.th

Abstract. While there have been several studies related to the effect of term weighting on classification accuracy, relatively few works have been conducted on how term weighting affects the quality of keywords extracted for characterizing a document or a category (i.e., document collection). Moreover, many tasks require more complicated category structure, such as hierarchical and network category structure, rather than a flat category structure. This paper presents a qualitative and quantitative study on how term weighting affects keyword extraction in the hierarchical category structure, in comparison to the flat category structure. A hierarchical structure triggers special characteristic in assigning a set of keywords or tags to represent a document or a document collection, with support of statistics in a hierarchy, including category itself, its parent category, its child categories, and sibling categories. An enhancement of term weighting is proposed particularly in the form of a series of modified TFIDF's, for improving keyword extraction. A text collection of public-hearing opinions is used to evaluate variant TFs and IDFs to identify which types of information in hierarchical category structure are useful. By experiments,

we found that the most effective IDF family, namely TF-IDFr, is identity > sibling > child > parent in order. The TF-IDFr outperforms the vanilla version of TFIDF with a centroid-based classifier.

Keywords: Keyword extraction, text classification, term weighting, hierarchical category structure

Mathematics Subject Classification 2010: 68T50

1 INTRODUCTION

Relevant keywords are usually provided to documents in a collection, as a navigational clue when one would like to find documents that match with his or her intention. Since keywords provide a compact representation of the document, they are used in many applications [1], such as improvement of text categorization [2], knowledge map construction [3], incremental clustering [4, 5], automatic indexing, automatic summarization, automatic classification, automatic clustering, and automatic filtering [6]. In the past, automatic keyword generation was explored in three different approaches; keyword assignment [7, 8, 9], keyword extraction, and their hybrid method [10]. In keyword assignment, the set of words/terms that can be used as keywords, called the vocabulary, is predefined. Even the keywords generated from this approach is simple, consistent, and controllable, it is expensive to create and maintain the controlled vocabulary, and in many cases. On the other hand, keyword extraction identifies one or more words/terms that appear in and regard as the most significant in the document without predefined vocabulary and uses them as the keywords of the document. In the same way, it is a challenging task to assign keywords to a document collection, rather than to a document [11, 12].

However, naturally a keyword can be relative in the sense that it may be a good keyword for some situations but it may not be in the other, such as the word 'education' may be a good keyword for general news articles but it may not be a good keyword when we consider only news articles related to education since all news are commonly related education. Moreover, when documents are related by a kind of structure, keywords should be selected according to that structure. In the past, rather than a flat structure, a hierarchical (tree) structure is applied for managing a large set of documents. This structure was used in some works, including [13, 14, 15, 16, 17]. Handling a hierarchical-category structure is different from that in a flat-category structure since it includes constituent relations, such as parent/child relation, sibling relation, and root/leave category status and then relativeness needs to be considered during keyword extraction [18, 19].

Based on the above background, this paper presents a method to assign keywords to each document category, in a hierarchical structure. The method applies the

IDF enhanced with information obtained from hierarchical structure (later called a relative IDF: IDFr) in the weighting scheme of TFIDF, for assigning keywords for a document category. A text collection of public-hearing opinions is used to evaluate various combinations of TFs and IDFrs. To identify types of information in hierarchical category structure which are useful for improving the classification accuracy and keyword extraction.

In the rest, Section 2 presents related works. The proposed keyword extraction using hierarchical relations is described in Section 3. Section 4 provides dataset characteristics and experimental settings. In Section 5, the experimental results and their evaluation are given. Finally, a conclusion and future works are discussed in Section 6.

2 RELATED WORKS

Manual keyword assignment to books, articles, or other forms of publications is a tedious and time-consuming task. As for solutions, several works on automatic keyword extraction have been conducted in many applications, such as in medical texts [20], economic webpages [3], news articles [21] and academic publications [22]. In the past, two approaches in extracting keywords from a document are corpus-oriented methods [23, 24] and document-oriented methods [25, 26] The corpus-oriented approach assumes that the keyword construction relies on the comparison between documents in the corpus while the keywords are likely to be evaluated statistically for their discrimination within the corpus. In this approach, keywords that occur in many documents within the corpus are not likely to be selected due to their statistical insignificant or low discriminating power. On the other hand, in the document-oriented approach, keywords can be assigned to a document without comparison with other documents. The keywords can directly be extracted from the document by experience. Such document-oriented methods will extract the same keywords from a document regardless of the current state of a corpus, but keywords extracted by the corpus-oriented approach may not be the same for different corpora (different document sets).

In the same way, it is a challenging task to assign keywords to a document collection (cluster or class), instead of to a document [11, 12]. Similarly, two approaches on keyword extraction for a cluster/class are corpus-based and class-based keyword selection [12, 21]. The corpus-based keyword selection is applied in classification problems by filtering the low frequency features that appear, in the corpus, less than a threshold value [27]. On the other hand, the class-based keyword selection identifies important keywords (features) for each class with the class-based metric, such as ICF and mutual information, via comparison of statistics among clusters or classes. The above-mentioned works showed that information related to the structure of hierarchical categories could be used for performance improvement, particularly classification tasks. While the naive method to handle relations between documents is a flat category structure, where documents are grouped into a number

of classes (clusters or groups), a more expressive method is to arrange documents in a topic hierarchy with superclass/subclass relations [13, 14, 15, 16, 17].

To our best knowledge, there are few works on how to extract keywords for a category using relationship information among categories when documents are arranged in a hierarchical category structure. To enhance the conventional TFIDF term weighting, relationship information between categories in the hierarchical structure, including identity relation, super/sub-category (parent/child) relation, and sibling relation can be used.

3 KEYWORD EXTRACTION USING RELATIONS IN HIERARCHICAL STRUCTURE

3.1 Formulation of Keyword Extraction

This section presents a formal description of keyword extraction tasks. Based on the vector space model (VSM) [28], the keyword extraction task can be formulated as follows. Given a document collection $D = \{d_1, d_2, \ldots, d_{|D|}\}$ and the universal set of terms $T = \{t_1, t_2, \ldots, t_{|T|}\}$, a document $d_j \in D$ can be represented by a document vector $\vec{d_j} = \{w_{1j}, w_{2j}, \ldots, w_{|T|j}\}$, where w_{ij} is the weight of the i^{th} term t_i in the j^{th} document d_j. In addition, given a set of categories $C = \{c_1, c_2, \ldots, c_{|C|}\}$, the category model $M\colon D \times C \rightarrow \{T, F\}$ can be used to partition documents in a collection into a number of groups by assigning a Boolean value, $M(d_j, c_k) = T$, to each pair $\langle d_j, c_k\rangle \in D \times C$ if the document d_j is in the category c_k, otherwise $M(d_j, c_k) = F$. Moreover, $C_k = \{d \mid d \in D, M(d, c_k) = T\}$, where (1) any category pair is exclusive $C_i \cap C_j = \emptyset$ and (2) all categories form the document collection $(D = \cup_{k=1}^{|C|} C_k)$. Similarly, a category $c_k \in C$ can be represented by a category vector $\vec{c_k} = \{w'_{1k}, w'_{2k}, \ldots, w'_{|T|k}\} = \sum_{(d \in c_k)} \vec{d_j}$, where w'_{ik} is the weight of the i^{th} term t_i in the k^{th} category c_k. In this vector, we use a centroid vector [29]. The category vector can be calculated using the formula in Section 3.4.

The keyword extraction is a process to assign a set of non-trivial words/terms to each document d_j in the collection, i.e., $K(d_j) = \{k_{1j}, k_{2j}, \ldots, k_{p_j j}\}$, where k_{ij} is the i^{th} keyword of the j^{th} document d_j, p_j is the number of keywords in the document d_j and normally $p_j \ll |T|$. Similarly, a set of keywords can be assigned to a category (class) $K(c_k) = \{k'_{1k}, k'_{2k}, \ldots, k'_{s_k k}\}$ where k'_{ik} is the i^{th} keyword of the category c_k and s_k is the number of keywords for the category c_k where $s_k \ll |T|$. The keywords of either a document or a category can be straightforwardly obtained by selecting a few words with high weights (say top-n words) under the weighting method applied.

3.2 Categories in a Hierarchical Category Structure

Given a hierarchical structure, there are possible four types of relations among category; i.e., identity (I), parent (P), child (C), and sibling (S). The identity func-

tion $I\colon C \times C \to \{T, F\}$ describes the identity relation between two categories, where $I(c_i, c_j) = T$ if $c_i = c_j$. Otherwise, $I(c_i, c_j) = F$. The child function $H\colon C \times C \to \{T, F\}$ describes the child relation between two categories, where $H(c_i, c_j) = T$ if c_j is a child of c_i. Otherwise, $H(c_i, c_j) = F$. The parent function $P\colon C \times C \to \{T, F\}$ describes the parent relation between two categories, where $P(c_i, c_j) = T$ if $H(c_j, c_i) = T$, otherwise $P(c_i, c_j) = F$. The sibling function $S\colon C \times C \to \{T, F\}$ is a function to describe the sibling relation between two categories, where $S(c_i, c_j) = T$ if $\exists c_k \cdot P(c_i, c_k) \wedge P(c_j, c_k) \wedge (c_i \neq c_j)$, otw $S(c_i, c_j) = F$.

In this work, given a set of documents, each document d_j can be assigned only one single category c_k in the hierarchy, i.e. $C(d_j) = \{c_k \mid M(d_j, c_k) = T\} \wedge |C(d_j)| = 1.$, where $C(d_j)$ is the set of categories the document d_j is associated. Let $I(c_k), C(c_k), P(c_k)$, and $S(c_k)$ be the set of documents associated to the identity category, the child category, the parent category, and the sibling category of the category c_k. Their formulations can be described as follows. Here, $H^*(c_i, c_j) = T$ if there is a reachable child relation from the node c_j to its ancestor c_i.

$$I(c_k) = \bigcup_{(c_j = c_k) \vee (\exists c_j \cdot H^*(c_k, c_j))} \{d \mid (M(d, c_j) = T)\}, \tag{1}$$

$$C(c_k) = \bigcup_{(\exists c_j \cdot H^*(c_k, c_j))} \{d \mid (M(d, c_j) = T)\}, \tag{2}$$

$$P(c_k) = \bigcup_{(\exists c_j \cdot P(c_j, c_k))} \{d \mid (M(d, c_j) = T)\}, \tag{3}$$

$$S(c_k) = \bigcup_{(\exists c_j \cdot P(c_k, c_i) \wedge P(c_j, c_i) \wedge (c_j \neq c_k))} \{d \mid (M(d, c_j) = T)\}. \tag{4}$$

Here, a series of relative IDFs are proposed to reflect the identity, parent, child, and sibling relations, as well as the collection IDF (the conventional IDF). Figure 1 illustrates an example of the $IDFr$ family when we calculate $IDFr's$ ($IDF_I, IDF_C, IDF_P, IDF_S$) for a term according to the hierarchical category structure.

3.2.1 The Conventional IDF or Collection IDF (IDF)

In the field of text classification and information retrieval, the inverse document frequency (IDF) is a statistic popularly used to point out words/terms that commonly occur in several documents with less contribution to the content of the text. The collection IDF can be formulated as follows.

$$IDF(t_i) = \log\left(\frac{|D|}{1 + DF(t_i)}\right) \tag{5}$$

where $DF(t_i)$ is document frequency, i.e., the number of documents that include a term (t_i). The $IDF(t_i)$ is a logarithmic function of the ratio of the number of

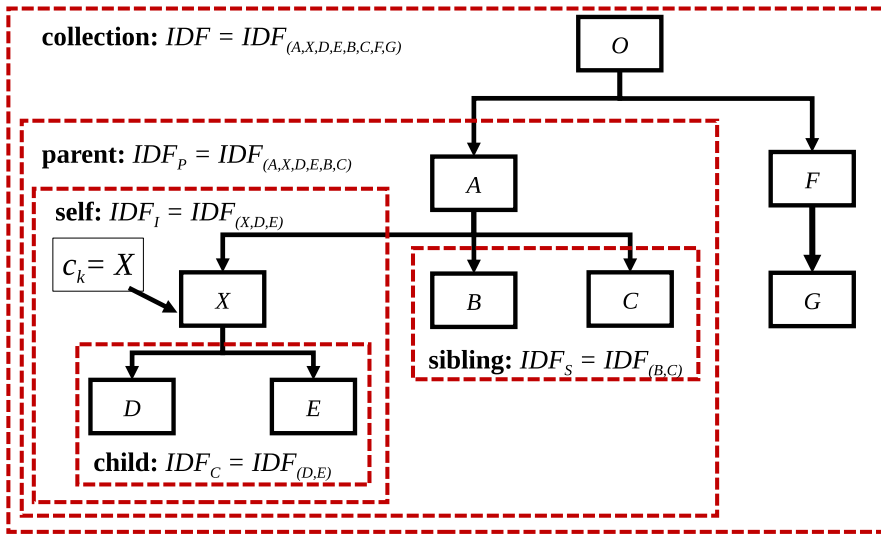


Figure 1. An example of the $IDFr$ family when we calculate $IDFr's$ $(IDF_I, IDF_C, IDF_P, IDF_S)$ for a term according to the hierarchical category structure. Here, the current node to be considered is $c_k = X$ and the other relative nodes are $I(c_k) = (X, D, E), C(c_k) = (D, E), P(c_k) = (A, X, D, E, B, C)$, and, $S(c_k) = (B, C)$.

the documents ($|D|$) in the collection divided by the number of the documents that contain the term (t_i) plus one, i.e. $DF(t_i)$ to prevent zero division. In Figure 1, IDF is calculated by taking the whole of documents in collection into account, that is $IDF = IDF_{(O)} = IDF_{(A,X,D,E,B,C,F,G)}$. Moreover, one (1) is added to the denominator.

3.2.2 The Identity IDF (IDF_I)

The identity IDF of the category X is the inverse document frequency of the documents in category X (i.e. $I(X)$) and the documents in the X's children categories (i.e. $C(X)$). For example, in Figure 1, the identity IDF of the category X is calculated from documents in the categories X, D, and E. The identity IDF of the term t_i in the category c_k, denoted by $IDF_I(t_i, c_k)$, is derived from Equation (6).

$$IDF_I(t_i, c_k) = \log\left(\frac{|D_I|}{1 + DF(t_i, I(c_k))}\right). \tag{6}$$

The identity IDF is the logarithmic value of the number of the documents in the category $c_k(|D_I|)$ divided by the number of the documents (in the category c_k) that contain the term (t_i), i.e. $DF(t_i, I(c_k))$.

3.2.3 Child IDF (IDF_C)

The child IDF is the inverse document frequency of the documents in all child categories of c_k; i.e., $C(c_k)$. In Figure 1, the child IDF of the category X is calculated from documents in the child categories; D, and E. The child IDF of the term t_i in the category c_k, denoted by $IDF_C(t_i, c_k)$, is derived from Equation (7).

$$IDF_C(t_i, c_k) = \log\left(\frac{|D_C|}{1 + DF(t_i, C(c_k))}\right). \tag{7}$$

The child IDF is the logarithmic value of the number of the documents in the collection of child categories $|D_C|$ divided by the number of the child documents of category $c_k, C(c_k)$, that contain the term (t_i), i.e. $DF(t_i, C(c_k))$.

3.2.4 Parent IDF (IDF_P)

This parent IDF is the inverse document frequency of the documents in the parent category of c_k; i.e., $P(c_k)$. In Figure 1, the parent IDF of the category X is calculated from documents in the parent category; A. The documents of the parent category A are the documents in A, X, D, E, B, and C. The parent IDF of the term t_i in the category c_k, denoted by $IDF_P(t_i, c_k)$, is derived from Equation (8).

$$IDF_P(t_i, c_k) = \log\left(\frac{|D_P|}{1 + DF(t_i, P(c_k))}\right). \tag{8}$$

The parent IDF is the logarithmic value of the collection of parent category $|D_P|$ divided by the number of the documents of the parent category of $c_k, P(c_k)$, that contains the term (t_i), i.e. $DF(t_i, P(c_k))$.

3.2.5 Sibling IDF (IDF_S)

The sibling IDF is the inverse document frequency of the documents in all sibling categories of c_k; i.e., $S(c_k)$. In Figure 1, the sibling IDF of the category X is calculated from documents in the sibling categories; B, and C. The sibling IDF of the term t_i in the category c_k, denoted by $IDF_S(t_i, c_k)$, is derived from Equation (9).

$$IDF_S(t_i, c_k) = \log\left(\frac{|D_S|}{1 + DF(t_i, S(c_k))}\right). \tag{9}$$

The sibling IDF is the logarithmic value of the collection of sibling categories $|D_S|$ divided by the number of the documents of the sibling category of $c_k, S(c_k)$, that contains the term (t_i), i.e. $DF(t_i, S(c_k))$.

3.3 Calculation of IDFr from Combining All IDF's Family

The previously mentioned calculations are used to inform statistic information based on a hierarchical structure. Hence, they are all needed to represent informative

values of different layers of categories. To summarize the information based on layer, details are given in Table 1.

	Parent IDF	Identity IDF	Sibling IDF	Child IDF
Top	X	O	O	O
Middle	O	O	O	O
Bottom	O	O	O	X

Table 1. Relative IDF by category type. O indicates that this type of relative IDF is calculable while X indicates that this type of relative IDF is not possible for this category in a hierarchy.

There are three layer types from a hierarchical structure. The first one is the top layer which is the root of their children categories. On the other hand, the bottom layer is the leaf of the tree where are very detailed of the root. In between the top and the bottom, the middle layer connects them. Apparently, there can be more than one middle layer.

For the top layer category that has no parent relation, the parent IDF cannot be calculated. The top category has a sibling relation for there are categories at the same level in the hierarchy. Hence top categories are for two relative relations which are its sibling relation (Sibling IDF) and its child relation (Child IDF) while it still needs Identity IDF to represent itself.

A middle-layer category has all possible relations in the hierarchical category structure. The super-type of the middle layer category is parent relation. The sub-type of the middle category is child relation while the categories in the same level of the same parent are its sibling relation. Therefore, the middle category in hierarchical structure has Parent IDF, Child IDF, and Sibling IDF respectively. In addition, the identity IDF is also required for its own standpoint. A base-layer category has no child relation, but it has parent relation and sibling relation. In this work, we use the IDFr's defined above to enhance the conventional TFIDF as shown in Equation (10), (11), (12), (13).

$$\vec{d_j} = [w_{ij}], \tag{10}$$

$$w_{ij} = TFIDF(t_i, d_j) \times IDF_r, \tag{11}$$

$$w_{ij} = N(t_i, d_j) \times IDF(t_i, d_j) \times IDF_r(t_i, d_j), \tag{12}$$

$$IDF_r(t_i, c_k) = IDF_I(t_i, c_k)^a \times IDF_P(t_i, c_k)^b \times IDF_S(t_i, c_k)^c \times IDF_C(t_i, c_k)^d \tag{13}$$

where $N(t_i, d_j)$ in Equation (12) is the number of term t_i in the document d_j. The document d_j depends on the category c_k being considered. In Equation (13), the $IDF_r(t_i, c_k)$, expresses the relative IDF of the term t_i in the category c_k, defined by the multiplication of the identify IDF, the parent IDF, the sibling IDF, and the child IDF with the powers of a, b, c and d, respectively. Such powers are hyperparameters in performance optimization.

By this parameter, we can exploit the relation in the hierarchical category structure to set term weighting for each term in the category. By employing the relation information, they should solve the difficulty of text classification in the hierarchy category which more complex and similar in its family categories. This can also help to identify and differentiate the importance of terms in a hierarchical category via specific term weighting. Moreover, it is expected to improve in a task of keyword extraction (KE) that uses a statistical approach by using hierarchical information.

We set up a situation for explanation in Figure 1. The calculation details of all possible related categories are declared in Table 2. In the table, several calculations are needed to represent a category. Despite many calculations, we expect the value of each IDFr to be able to inform the different term-weight based on a different layer. The calculation is language-free which means that it is not bound to any specific language. Thus, it can be used with any language.

Weighting Factors	Parent A	Sibling B, C	Children D, E
IDFr	$IDF_{(A,X,D,E,B,C)}$	$IDF_{(B,C)}$	$IDF_{(D,E)}$
TFIDF	$TF_{(X)} \times IDF_{(A,X,D,E,B,C)}$	$TF_{(X)} \times IDF_{(B,C)}$	$TF_{(X)} \times IDF_{(D,E)}$

Weighting Factors	Collection O	Itself X
IDFr	$IDF_{(O)}$	$IDF_{(X,D,E)}$
TFIDF	$TF_{(X)} \times IDF_{(O)}$	$TF_{(X)} \times IDF_{(X,D,E)}$

Table 2. IDFr and TFIDF with relations in each category X

However, there are limitations of this calculation. The first one is that the invented IDFr is suitable for hierarchical structures containing more than two depth layers. Moreover, the IDFr could not be applied to flat category and network category structure since it is specifically designed for acyclic top-down relation. For the information of term frequency in the whole collection, identity category, child category, parent category, and sibling category. All of these are explained by the formula in Equation (6), (7), (8), (9) respectively. Statistical calculations of each layer type are different; thus, they will be explained separately in each subsection below. In addition, an extra calculation including score normalization and smoothing is also explained. The base unit of calculation is the normalized TFIDF. The newly invented IDFr is an additional value which will be multiplied with the base TFIDF. Before applying IDF and IDFr, TF is performed with L2-normalization [30] to solve the problem of overweighting due to both higher term frequency and more unique terms. Since a long document gains two advantages over a short document by including higher term frequencies and more unique terms in document representation, a statistical frequency may be biased and led to unfairness in the calculation. The L2-Norm of TF is calculated by dividing all elements in a vector with the length of the vector that is $\sqrt{\sum N(w,d)^2}$ where $N(w,d)$ is the number of word(w) in document (d) of word-document vector.

To avoid division by zero, smoothing technique is suggested to apply in this task [31, 32]. It is common for zero to be assigned in a calculation since a document frequency $DF(t_i)$ value is the number of documents in the considered corpus containing the focused term (t_i) which may not exist in all documents. Thus, smoothing is necessary to prevent the possibility for division by zero. The smoothed inverse document frequency (IDF) is defined in Equation (5), in which $|D|$ is the number of documents in the corpus [31, 33].

3.4 Keyword Extraction for a Category

To obtain keywords from a category, terms in documents of the same categories are calculated as term-weighting. Keywords in this work are defined as condensed-summary terms representing a category. In this work, we apply a centroid based method to extract keywords and use the sum centroid as the representative of category c_k. The category vector is represented by a vector $\vec{c_k} = \{w'_{1k}, w'_{2k}, \ldots, w'_{|T|k}\}$. From the scores of hierarchical term-weighting, each term w'_{ik} in a category is assigned with a single score based on its category. The scores of a term are varied from a category to other categories depending on how significant from their existence. To select some as keywords to represent their category, in this work, the selection is based on the top rank. This method is to set n-best rank while n can be any number, and the terms which are in those top ranks are chosen as representative keywords.

4 DATASETS AND EXPERIMENTAL SETTINGS

4.1 The Dataset

The focused dataset in this work is a collection of public hearing opinion texts on how to reform Thailand, namely the "Thai reform" (`http://static.thaireform.org`). The full collection is composed of 64 850 opinions from 66 674 participants taken part in, from all 77 provinces in Thailand, arranged in eighteen reform issues (categories). The documents were assigned with one or more categories. Consequently, the summation of documents separated by categories is larger than the actual number of documents since some documents are counted more than one time. Among the eighteen categories, we select three major categories; i.e., educational and HR development (for short, E = Education with 9 314 documents), anti-corruption and anti-misconduct (for short, C = Corruption, with 4 367 documents), and local government (for short, G = Government, with 9 571 documents) for benchmarking since they are balanced in their three-level hierarchies. To simplify the process, two preferences are made to select major subcategories and their membership documents. Firstly, only documents assigned with a single category are considered. If the document was allocated to more than one category, it will be excluded for the dataset in the experiments. Hence, the experiment is conducted by comparing datasets in a pair to prevent the exclusion of documents. Secondly, we select the subcategories that their siblings are balanced in terms of the number of documents.

We evaluate our approach using documents in three category pairs (1) Reform-E-C, (2) Reform-E-G, (3) Reform-C-G, where E is 'educational and human resource development', C is 'anti-corruption and anti-misconduct', and G is 'local government'. Table 3 indicates the major characteristics of the data sets. The selected datasets have 3-depth level, the number of categories in the hierarchy structure for E-C, E-G, and C-G are 14, 16, and 16, respectively.

Data Sets	Reform-E-C	Reform-E-G	Reform-C-G
No. of docs	10 433	13 315	9 599
No. of categories	14	16	16
No. of levels	3	3	3
No. of features	6 772	7 241	6 188

Table 3. Characteristics of the three data sets

All documents in the Thai reform text database are written in the form of the central Thai (official Thai) sentences. Some are short sentences while the other are long sentences. For pre-processing, we manually edited frequently found typos and misspelling since they greatly affect further processes in terms of accuracy. Words in document are segmented using LongLexTo word segmentation engine. Then, non-text characters including symbols and numerical characters are removed. It is noted that stop words (functional words) are not removed and kept intact as they are.

4.2 Experimental Settings

There are four experiments as follows. The first experiment aims to investigate the effect of a single term weighting as Identity IDF (IDF_I), Parent IDF (IDF_P), Sibling IDF (IDF_S), and Child IDF (IDF_C) to term weighting on accuracy improvement of a standard centroid-based classifier. Only one term weighting factor is added, in turn, to the standard TFIDF as either a multiplier or a divisor. This experiment was designed to find the result of each for comparison. Moreover, the uses of a factor as a multiplier or a divisor were also compared. For dataset separation for training and testing, five-fold cross validation was applied. They were used in the centroid-based classification task to classify a raw text document in the testing set. The measurement in this experiment was accuracy and standard deviation.

In the second experiment, multiple term weighting factors were combined in different manners, and the efficiencies of these combinations were evaluated. This experiment investigated the combination of term weighting factors in improving the classification accuracy. Two following topics were considered in this experiment:

1. which factors are suitable to work together and
2. what is the appropriate combination of these factors.

In this experiment, five-fold cross validation is applied for the classification task, and the measurement is accuracy and standard deviation. At this experiment, the clas-

sifiers incorporate term weighting factors in their weighting, term weighting based on centroid-based classifiers (later called THCBs).

In the third experiment, we evaluate top keywords extraction by human experts. Three human experts evaluate the top keywords whether the obtained words are a keyword or not. In addition, bottom-n features are also selected to evaluate top keywords extraction. As the last evaluation, we select Top-100 ranked terms of each category to comparison the differential on terms between TFIDF weighting and TFIDFr weighting from THCB1 to clarify our category keywords by expert evaluation again. For all experiments, a centroid-based classifier and cosine similarity were used. The document-length normalization on TF is used before cooperating with IDF-IDFr in this work because it outperforms other in a preliminary result. One of the most important factors towards the meaningful evaluation is the way to set classifier parameters. Parameters that are applied to these classifiers are determined by some preliminary experiments since it performed well in ours pretests. For SCB, we apply the standard term weighting, TFIDF, the query weighting for THCBs is TFIDF by default. Smoothing techniques are applied in the term weighting process.

5 EXPERIMENTAL RESULTS AND EVALUATION

5.1 Effects of Single Term Weightings

In the first experiment, four term weighting factors, i.e., Identity IDF, Parent IDF, Sibling IDF, and Child IDF are individually evaluated by adding each term factor one by one to the standard TFIDF. For clarity, TF are also evaluated. The query weighting is TFIDF. The result is shown in Table 4. The bold indicates term weightings which achieve higher performance than the baseline TFIDF (SCB). Moreover, as we applied 5-fold cross validation, the number on the top-right superscript means the number of times (out of 5 times) that the classifier outperforms the standard classifier, i.e., SCB.

The result shows that the standard TFIDF (SCB) performs better than TF. With TFIDF (SCB) as a baseline, the average score of Identity IDF with a multiplier, Sibling IDF with a multiplier, and Child IDF with a multiplier is higher. Even average accuracy of Parent IDF is slightly lower than TFIDF, we found a good signal that if it is a multiplier (promoter), it is still likely to perform better than the divisor (demoter) like the other factors of IDFr. An interesting from observation in this experiment is all of term weighting factors has some effects on classification accuracy in roles of promoting the weight. Thus, it is conclusive that the multiplier (promoter) is better when applying to IDFr.

5.2 Effect of Multiple Term Weightings

The second experiment investigates the combination of term weighting factors in improving the classification accuracy. Although the previous experiment suggests

Method	Reform E-C	Reform E-G	Reform C-G	Avg.
TF	33.53 ± 2.06	36.76 ± 5.12	32.14 ± 2.58	34.14 ± 3.82
$TF \times IDF(SCB)$	35.54 ± 2.84	39.13 ± 6.61	34.50 ± 3.49	36.39 ± 4.74
$TF \times IDF \times IDF_I$	$\mathbf{36.90 \pm 2.86}^{(5)**}$	$\mathbf{41.25 \pm 10.01}^{(5)**}$	$34.18 \pm 3.69^{(3)}$	$\mathbf{37.44 \pm 6.63}^{(5)}$
$TF \times IDF \times IDF_S$	$\mathbf{36.38 \pm 3.13}^{(4)}$	$\mathbf{40.57 \pm 8.11}^{(5)**}$	$34.48 \pm 3.21^{(2)}$	$\mathbf{37.15 \pm 5.61}^{(4)}$
$TF \times IDF \times IDF_C$	$\mathbf{36.77 \pm 2.82}^{(4)}$	$38.24 \pm 6.57^{(2)}$	$34.50 \pm 2.72^{(2)}$	$\mathbf{36.50 \pm 4.39}^{(3)}$
$TF \times IDF \times IDF_P$	$\mathbf{35.60 \pm 2.92}^{(4)}$	$\mathbf{40.54 \pm 8.39}^{(2)}$	$32.95 \pm 4.12^{(1)}$	$36.36 \pm 6.16^{(3)}$
$TF \times IDF/IDF_C$	$34.50 \pm 2.15^{(1)}$	$35.14 \pm 5.62^{(0)}$	$31.45 \pm 2.72^{(0)}$	$33.70 \pm 3.90^{(0)}$
$TF \times IDF/IDF_I$	$32.08 \pm 1.43^{(0)}$	$34.47 \pm 3.54^{(0)}$	$30.62 \pm 2.04^{(0)}$	$32.39 \pm 2.84^{(0)}$
$TF \times IDF/IDF_S$	$30.76 \pm 2.27^{(0)}$	$34.81 \pm 4.23^{(0)}$	$30.56 \pm 2.11^{(0)}$	$32.04 \pm 3.46^{(0)}$
$TF \times IDF/IDF_P$	$31.52 \pm 0.87^{(0)}$	$34.40 \pm 4.02^{(1)}$	$29.47 \pm 2.35^{(0)}$	$31.80 \pm 3.29^{(0)}$

** $p < 0.05$ from the analysis of Wilcoxon Signed-Rank Test comparison with SCB

Table 4. The effect of single additional term weighting factors to TFIDF

the role of each term weighting factor, all possible combinations are explored in this experiment. Two following topics are focused:

1. which factors are suitable to work together and
2. what is the appropriate combination of these factors.

To the end, we perform all combinations of Identity IDF, Parent IDF, Sibling IDF, and Child IDF by varying the power of each factor between -1 and 1 with a step of 0.5 and using it to modify the standard TFIDF. The total number of combinations is 625 ($5 \times 5 \times 5 \times 5$). These combinations include TFIDF and eight single-factor term weightings. By the result, there are 67 patterns giving better performance than the baseline, TFIDF. The 20 best (top 20) and the 20 worst classifiers, according to average accuracy on three data sets, are selected for evaluation. Table 5 panel A (panel B) shows the number of the best (worst) classifiers for each power of IDFr family as Identity IDF, Parent IDF, Sibling IDF, and Child IDF. Characteristics and performances of the top 20 term weightings are shown in Tables 6 and 7.

Table 5 (panel A) confirms the same conclusion as the result obtained from the first experiment. The Identity IDF, Parent IDF, Sibling IDF, and Child IDF are suitable to be a promoter rather than a demoter. There are almost no negative results, except for Sibling IDF, and it is more obvious in top-19 cases of top-20, except the case of top 5. On the other hand, Table 5 (panel B) shows that the performance is low if Identity IDF, Parent IDF, Sibling IDF, and Child IDF as a demoter. Apparently, it is clear that using Identity IDF, Parent IDF, Sibling IDF, and Child IDF as a demoter make a negative impact on performance. This experiment can conclude that the results correspond to those of the first experiment.

Table 7 also emphasizes the classifiers that outperform the standard TFIDF (SCB) in all three data sets, with a mark '*'. There are fifteen classifiers that are superior. Moreover, as we applied 5-fold cross validation, the number on the

Term Weighting Factors	Power of the Factor					Total Methods
	−1	−0.5	0	0.5	1	
Panel A						
Identity IDF (IDF_I)	0 (0)	0 (0)	6 (9)	3 (8)	1 (3)	10 (20)
Parent IDF (IDF_P)	0 (0)	0 (0)	2 (9)	5 (8)	3 (3)	10 (20)
Sibling IDF (IDF_S)	0 (0)	1 (1)	5 (10)	4 (7)	0 (2)	10 (20)
Child IDF (IDF_C)	0 (0)	0 (0)	4 (10)	5 (9)	1 (1)	10 (20)
Panel B						
Identity IDF (IDF_I)	6 (10)	2 (4)	2 (3)	0 (2)	0 (1)	10 (20)
Parent IDF (IDF_P)	8 (16)	2 (3)	0 (1)	0 (0)	0 (0)	10 (20)
Sibling IDF (IDF_S)	9 (16)	1 (4)	0 (0)	0 (0)	0 (0)	10 (20)
Child IDF (IDF_C)	2 (5)	2 (4)	4 (6)	1 (2)	1 (3)	10 (20)

Table 5. Descriptive analysis of term weighting factors with different power of each factor. Panel A: the best 10 and panel B: the worst 10 (best 20 and worst 20 in parentheses).

top-right superscript means the number of times (out of 5 times) that the classifier outperforms the standard classifier, i.e., SCB.

This fact shows that there are some common term weighting factors that are useful generally in all data sets. The three best term weightings in this experiment are respectively as follows.

1. $TF \times IDF \times sqrt(IDF_P \times IDF_C)$,
2. $TF \times IDF \times sqrt(IDF_P \times IDF_S \times IDF_C)$,
3. $TF \times IDF \times sqrt(IDF_I \times IDF_P)$.

We found that there are at least two of four term weighting factors that cooperate to enhance the performance of classifiers. In a conclusion from this experiment, it is noticeable that Identity IDF, Parent IDF, Sibling IDF, and Child IDF should act as a promoter than a demoter. However, it is observed that the appropriate powers of term weighting factors depend on some characteristics of data sets.

There are a total of 625 classifiers from all combinations of power of factor (-1, -0.5, 0, 0.5, $1 = 5 \times 5 \times 5 \times 5$). To investigate all combinations, it needs the very high computation cost. Therefore, we exploit the result of the former experiments in suggesting the role of IDFr. All possible combinations subjected to this constraint include 225 classifiers (225 from $3(0, 0.5, 1) \times 5 \times 5 \times 3(0, 0.5, 1)$). From our classification accuracy result, we found that there are top-67 operation cases (of power of factor) that our method is superior than the baseline, TFIDF smoothing, on average from all three Reform datasets.

Moreover, for the combined IDFr factor, it seems the parent IDF and the child IDF are the most effective factor to improve the classification accuracy. That is the information from the parent and child category is helpful to distinguish the difference among classes in the hierarchy.

Methods	Power of				Term Weighting
	IDF_I	IDF_P	IDF_S	IDF_C	
THCB1*	0	0.5	0	0.5	$TF \times IDF \times sqrt(IDF_P \times IDF_C)$
THCB2*	0	0.5	0.5	0.5	$TF \times IDF \times sqrt(IDF_P \times IDF_S \times IDF_C)$
THCB3*	0.5	0.5	0	0	$TF \times IDF \times sqrt(IDF_I \times IDF_P)$
THCB4*	0	1	0	0.5	$TF \times IDF \times IDF_P \times sqrtIDF_C$
THCB5*	0	1	−0.5	0.5	$TF \times IDF \times IDF_P / sqrt(IDF_S) \times sqrt(IDF_C)$
THCB6*	0.5	0.5	0	0.5	$TF \times IDF \times sqrt(IDF_I \times IDF_P \times IDF_C)$
THCB7*	0.5	0	0.5	0	$TF \times IDF \times sqrt(IDF_I \times IDF_S)$
THCB8	1	0	0.5	0	$TF \times IDF \times IDF_I \times sqrt(IDF_S)$
THCB9*	0	1	0	1	$TF \times IDF \times IDF_P \times IDF_C$
THCB10*	0	0.5	0.5	0	$TF \times IDF \times sqrt(IDF_P \times IDF_S)$
THCB11	1	0	0	0	$TF \times IDF \times IDF_I$
THCB12	1	0.5	0	0	$TF \times IDF \times IDF_I \times sqrt(IDF_P)$
THCB13	0.5	0.5	0.5	0	$TF \times IDF \times sqrt(IDF_I \times IDF_P \times IDF_S)$
THCB14*	0.5	0	0	0.5	$TF \times IDF \times sqrt(IDF_I \times IDF_C)$
THCB15*	0	0	1	0.5	$TF \times IDF \times IDF_S \times sqrt(IDF_C)$
THCB16*	0	0	0.5	0.5	$TF \times IDF \times sqrt(IDF_S \times IDF_C)$
THCB17*	0	0.5	0	0	$TF \times IDF \times sqrt(IDF_P)$
THCB18*	0.5	0	0	0	$TF \times IDF \times sqrt(IDF_I)$
THCB19*	0.5	0	0.5	0.5	$TF \times IDF \times sqrt(IDF_I \times IDF_S \times IDF_C)$
THCB20	0.5	0	1	0	$TF \times IDF \times sqrt(IDF_I) \times IDF_S$
SCB	0	0	0	0	$TF \times IDF$

Table 6. The best-20 pattern of term weightings for experiment

5.3 Keyword Extraction in the Hierarchical Structure

This experiment is designed to find the potentials of keyword extraction used in the previous experiment. We select category keywords from THCB1 (using term weighting from ($TF \times IDF \times sqrt(IDF_P \times IDF_C)$) which is the best from all Reform dataset pairs regarding accuracy results. The steps in this experiment are as follows.

1. Selecting Top-100 keywords based on rank from each category of each dataset pair, namely Top-100 keywords from each of 14 categories of Reform-E-C and 16 categories from Reform-E-G, Reform-C-G.
2. Assigning those keywords as category keywords of their respective category.
3. Comparing the keywords from the proposed method with keywords from 3 human experts and calculating the results using precision (P), recall (R), and F (F1) score.

   (a) The number of keywords from the proposed method is limited to Top-10 to Top-50 for 10 different intervals into 5 groups.
   (b) The scores are in an average result of the 3 human experts.

Methods	Reform-			Avg.
	E-C	E-G	C-G	
THCB1*	$37.47 \pm 2.60^{(5)**}$	$40.47 \pm 7.86^{(5)**}$	$35.79 \pm 3.46^{(5)**}$	$37.91 \pm 5.20^{(5)}$
THCB2*	$37.22 \pm 2.42^{(5)}$	$40.54 \pm 8.43^{(4)}$	$35.86 \pm 3.49^{(4)}$	$37.87 \pm 5.44^{(5)}$
THCB3*	$37.24 \pm 2.72^{(5)}$	$41.45 \pm 9.52^{(4)}$	$34.89 \pm 3.94^{(3)}$	$37.86 \pm 6.35^{(5)}$
THCB4*	$37.35 \pm 2.28^{(5)}$	$40.80 \pm 8.64^{(4)}$	$35.30 \pm 3.22^{(4)}$	$37.82 \pm 5.59^{(5)}$
THCB5*	$37.19 \pm 2.43^{(5)}$	$40.47 \pm 7.89^{(4)}$	$35.62 \pm 3.46^{(4)}$	$37.76 \pm 5.22^{(5)}$
THCB6*	$37.71 \pm 2.40^{(5)}$	$40.69 \pm 9.09^{(4)}$	$34.69 \pm 3.33^{(2)}$	$37.70 \pm 5.90^{(4)}$
THCB7*	$37.14 \pm 2.71^{(5)**}$	$41.20 \pm 9.27^{(5)**}$	$34.72 \pm 4.03^{(3)}$	$37.69 \pm 6.24^{(5)}$
THCB8	$37.15 \pm 2.29^{(5)}$	$41.37 \pm 10.98^{(4)}$	$34.02 \pm 3.38^{(1)}$	$37.52 \pm 6.99^{(4)}$
THCB9*	$37.32 \pm 2.51^{(5)}$	$39.81 \pm 8.11^{(3)}$	$35.37 \pm 2.51^{(3)}$	$37.50 \pm 5.09^{(4)}$
THCB10*	$36.45 \pm 2.39^{(5)**}$	$41.45 \pm 8.52^{(5)**}$	$34.60 \pm 3.76^{(3)}$	$37.50 \pm 5.95^{(5)}$
THCB11	$36.90 \pm 2.86^{(5)**}$	$41.25 \pm 10.01^{(5)**}$	$34.18 \pm 3.69^{(3)}$	$37.44 \pm 6.63^{(5)}$
THCB12	$37.07 \pm 2.72^{(4)}$	$41.57 \pm 11.17^{(2)}$	$33.60 \pm 3.68^{(1)}$	$37.41 \pm 7.28^{(3)}$
THCB13	$37.12 \pm 2.33^{(5)}$	$41.26 \pm 10.10^{(3)}$	$33.85 \pm 3.59^{(1)}$	$37.41 \pm 6.65^{(3)}$
THCB14*	$37.09 \pm 2.58^{(4)}$	$39.96 \pm 7.94^{(4)}$	$34.99 \pm 3.41^{(3)}$	$37.35 \pm 5.26^{(4)}$
THCB15*	$36.82 \pm 2.66^{(4)}$	$39.83 \pm 8.33^{(3)}$	$35.38 \pm 3.13^{(4)}$	$37.34 \pm 5.32^{(4)}$
THCB16*	$37.10 \pm 2.61^{(5)}$	$39.73 \pm 7.75^{(4)}$	$35.18 \pm 3.24^{(4)}$	$37.34 \pm 5.09^{(5)}$
THCB17*	$36.31 \pm 2.71^{(5)**}$	$40.81 \pm 7.97^{(5)**}$	$34.79 \pm 3.85^{(3)}$	$37.30 \pm 5.61^{(5)}$
THCB18*	$36.46 \pm 3.01^{(5)}$	$40.64 \pm 8.23^{(5)}$	$34.80 \pm 3.98^{(3)}$	$37.30 \pm 5.74^{(5)}$
THCB19*	$37.28 \pm 2.80^{(5)}$	$40.01 \pm 8.95^{(3)}$	$34.57 \pm 3.06^{(3)}$	$37.28 \pm 5.75^{(4)}$
THCB20	$36.60 \pm 2.50^{(5)}$	$40.92 \pm 9.86^{(4)}$	$34.19 \pm 3.58^{(2)}$	$37.23 \pm 6.45^{(5)}$
SCB	35.54 ± 2.84	39.13 ± 6.61	34.50 ± 3.49	36.39 ± 4.74

** $p < 0.05$ from the analysis of Wilcoxon Signed-Rank Test for comparison between THCB methods and SCB

Table 7. Classification accuracy of the best-20 term weightings compared with SCB

The keyword comparing results are given below. Top-10 to Top-50 Keywords of Top-100 features and Bottom-10 to Bottom-50 words from Bottom-100 features from THCB1 on Reform-E-C.

From the results in Figure 2, the Top-10 and Top-20 keywords yield respectively the highest of 82.86 % and 72.14 % of precision which is higher than the average of all result as 68.38 %. The graph shows the descending since the more keywords in consideration, the more incorrect keywords are found. In terms of recall score, the graph is in opposite to the precision since the ascending indicates that the keywords are increasingly matched to the experts' opinion from the higher number of the given keywords. The best recall is from Top-50 for 42.14 % while the worst is from Top-10 as 11.73 %. In a comparison of the top and bottom group, the difference on all measurements was obvious that the top was much higher than the bottom. The difference of F-measure in average was greater than about 25 score in (37.08 % from average of top and 12.31 % from bottom).

From the results given in Figure 3, the average Precision and F-measure of Top10-50 was 67.10 % and 35.68 %, respectively, while the average Precision and F

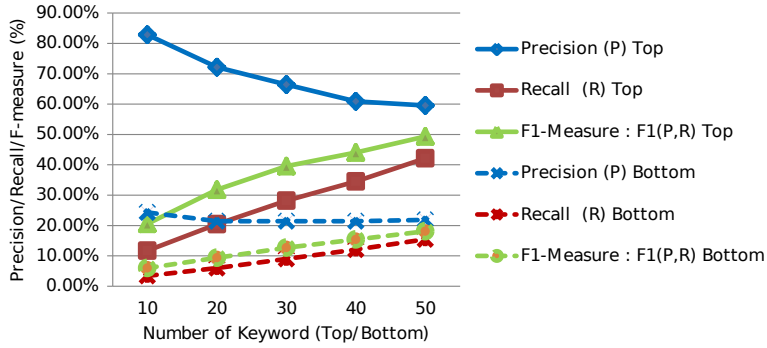


Figure 2. Top-10 to Top-50 Keywords of Top-100 features and Bottom-10 to Bottom-50 words from Bottom-100 features evaluations from Reform-E-C

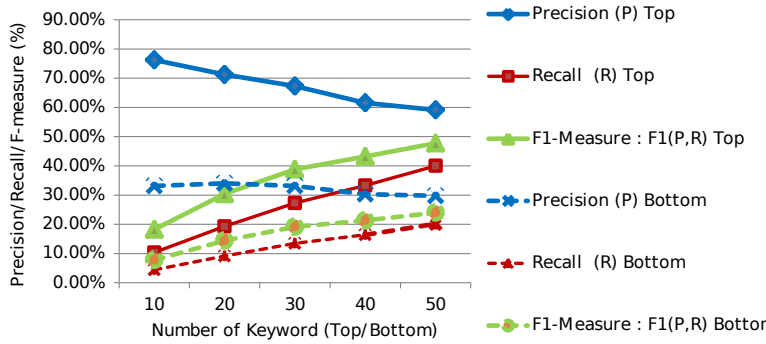


Figure 3. Top-10 to Top-50 keywords from Top-100 features and Bottom-10 to Bottom-50 words from Bottom-100 features in the Reform-E-G

measure of the bottom was 32.08 % and 17.34 %. Again, all measurements of the top group were significantly higher than those of the bottom group, but the gap was less once comparing to Figure 2. In case of the best in measurements, the best precision was obtained from Top-10 while the best recall and best F-measure was found in Top-50.

The results given in Figure 4 show a similar result as the result of other dataset pairs. The best F-measure from the top was from Top-50 and Bottom-50 for bottom group while the worst was obtained from Top-10 and Bottom-10.

From all results of dataset pairs, we observed the results and found two issues. The first one is that the list of extracted keywords contains a functional word (stop-word) as shown in Table 8 and leads to incorrect results. Since the stop words were

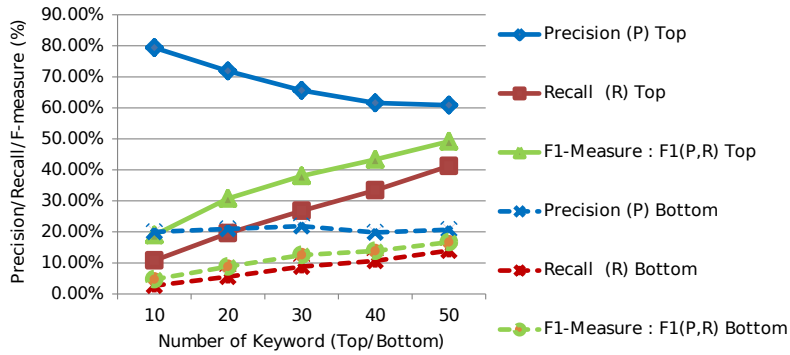


Figure 4. Top-10 to Top-50 keywords of Top-100 features and Bottom-10 to Bottom-50 words from Bottom-100 features evaluations from Reform-C-G

	Category E11		Category E21	
Rank	Extracted Keyword	Expert Decision	Extracted Keyword	Expert Decision
1	ศึกษา (study)	√	หลักสูตร (course)	√
2	โรงเรียน (school)	√	การเรียน (education)	√
3	การศึกษา (education)	√	การสอน (teaching)	√
4	เท่าเทียมกัน (equally)	√	คุณธรรม (virtue)	√
5	สถานศึกษา (school)	√	เพิ่ม (add)	√
6	ใน(in)	×	ใน (in)	×
7	อยาก (want)	×	วิชา (subject)	√
8	เด็ก (child)	√	จริยธรรม (morality)	√
9	ระดับ (Level)	√	เด็ก (child)	√
10	ด้าน (field)	×	ส่งเสริม (promote)	√
...	...	...	...	...
33	...		เรียน (learn)	√
46	...		การเรียนรู้ (learning)	√

Table 8. A list of Top-10 extracted and some lower ranked keywords from category E11 and E21

not considered as a keyword by human expert, this has direct effects on obtained precision. If the stop words were removed from the keyword list, the precision should be accordingly boosted and thus the F measure as well. The second issue was that there are same semantic extracted keywords with different wording and part-of-speech, for example, the term 'เรียน'(learn), 'ศึกษา'(study) and 'การเรียน'(education). The substring relations between words trigger decrement of term frequency and may lower the keywords that should occur frequently but they are substring or superstring of the others.

5.4 Keyword Extraction for Both TFIDF and TF-IDFr

Since the results of F-measure were slightly different among TFIDF and TF-IDFr, clarification on the effect of applying IDFr was investigated. Thus, we observed the keyword extraction results and listed out the difference between the two methods. According to the results of previous experiments, three pairs of datasets and three groups of terms based on term-weight ranking were still focused. The different keywords between the two methods which are TFIDF and TFIDF-IDFr were observed, and those keywords were judged by three humans whether they were appropriate terms as significant to represent the category or not. The positive ones were counted and calculated into probability in the range of 0 to 1 for the lowest to the highest, respectively. The different keyword evaluations of Top-100 ranked terms comparison between TFIDF and TF-IDFr weighting from Reform-E-C, E-G, and C-G.

From Figure 5, in average, terms from TF-IDFr were evaluated for more significantly suitable for being a keyword as 71 % while those from TFIDF were around 22 % for the Top-100 ranked terms. Furthermore, the different terms from IDFr of those in the middle and bottom group of Top-100 ranked terms were resulted similarly to the top group. These indicated that IDFr effectively assisted in keyword extraction in this dataset pair.

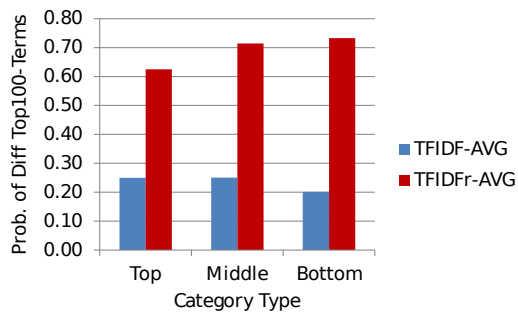


Figure 5. Keyword evaluations on different terms of Top-100 ranked terms comparison between TFIDF and TF-IDF-IDFr weighting with the average on probability by category type from Reform-E-C

Similarly, to the previous results, terms from IDFr were evaluated to be better in terms of term significance to represent hierarchical categories in every category. The most difference was found in the middle layer categories where all IDFr calculations can be applied. In average, TFIDFr generated different keywords which obtained higher evaluation as 0.64 compared to 0.25 from common TFIDF for dataset pair of E-G in Figure 6.

In this dataset pair of C-G in Figure 7, there is one case that different terms from common IDF evaluate equal to those of IDFr. However, the overall evaluation still insisted that TFIDFr performed better in all other cases. From all three dataset

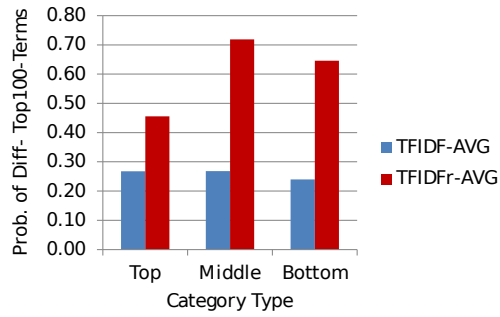


Figure 6. Keyword evaluations on different terms of Top-100 ranked terms comparison between TFIDF and TF-IDF-IDFr weighting with the average on probability by category type from Reform-E-G

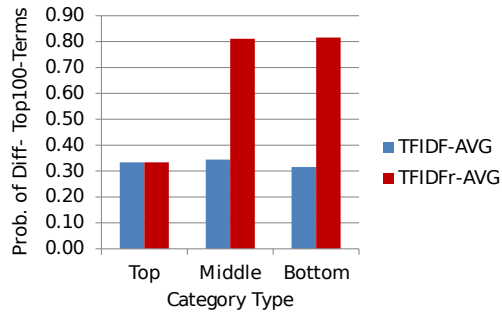


Figure 7. Keyword evaluations on different terms of Top-100 ranked terms comparison between TFIDF and TF-IDF-IDFr weighting with the average on probability by category type from Reform-C-G

pairs, different terms of TFIDFr were highly regarded from expert opinion on being more significant to represent a category. The average of all categories was about 0.69 from TFIDFr while 0.26 from the baseline TFIDF.

5.5 Experimental Summary

This section summarizes the results from the above four experiments.

1. The relative IDFs (IDFr), including parent IDF, child IDF, sibling IDF, and identify IDF, are shown to be effective for improving the classification accuracy and the keyword extraction. For the single IDFr factor, the most effective IDF family is ranked in the order of identity > sibling > child > parent. All types of IDFr's should be used as a multiplier (a promoter).

2. For the combined IDFr factor, it seems the parent IDFr and the child IDFr are the most effective factor to improve the classification accuracy.
3. For the keyword extraction, the Top-10, Top-20, Top-30, and Top-50 keywords extracted by our TF-IDFr weightings are manually assessed to be better representatives than the Bottom-10, Bottom-20, Bottom-30, and Bottom-50 keywords.
4. The average P, R, and F of the Top-10 to Top-50 keywords extracted by the best TF-IDFr weighting on all datasets are 67.78 %, 26.61 % and 36.27 %, respectively. The average P, R, and F of the Bottom-10 to Bottom-50 on all datasets are 24.95 %, 10.08 % and 13.66 %, respectively. This implies that the top keywords represent categories better than the bottom keywords do.
5. Comparing the keywords extracted by TFIDF and TF-IDFr weighting, the keywords from TF-IDFr weighting outperform the keywords from TFIDF weighting, for all categories in the hierarchical structure for all datasets.
6. As the error analysis, the two issues are (1) some stopwords are selected as keywords and the substring relations between words. The former incorrectly promotes stopwords as keywords and the latter triggers decrement of term frequency and may lower the keywords that should occur frequently, but they are substring or superstring of the others. If these issues can be solved, the results of keyword extraction should be improved.

6 CONCLUSIONS

The IDFr calculation is language-free which means that it is not bound to any specific language. In this work, some observations can be made as follows. Firstly, the Identity IDF, Parent IDF, Sibling IDF, and Child IDF should act as a promoter in an addition to TFIDF rather than a demoter since all of the results from multiplying are higher than applying a division. Secondly, from 225 of all tested combinations, there are 67 operation cases (29.8 %) that our method yielded superior results than the baseline, TFIDF smooth, on average classification accuracy on all three Reform datasets. Thirdly, in a keyword extraction task evaluated by three human experts, the average P, R, and F of the Top group (Top-10 to Top-50) from all dataset pair is 67.78 %, 26.61 %, and 36.27 % while the bottom group (Bottom-10 to Bottom-50) obtained 24.95 %, 10.08 % and 13.66 %, respectively. The results are conclusive that the proposed IDFr can extract a list of relevant keywords from hierarchy-based documents and effectively rank the relevant ones higher than the irrelevant terms.

Another keyword extraction task evaluated by three human experts, the average probability of the difference terms of Top-100 ranked terms of TF-IDFr weighting from all dataset pair is 0.47 on top category, 0.74 middle category, and 0.73 bottom category while TFIDF is 0.28 on top category, 0.29 middle category, and 0.25 bottom category hence, we can conclude that our method outperform TFIDF baseline

clearly. The incorrect results are the function words which a human disregards as a keyword. To solve the issue, stop word removal can be applied to boost keyword extraction performances. As our future work, the method to efficiently evaluate keywords is needed. It is worth studying the extraction of keywords in several tree-like or network-like structures, by exploiting semantic and higher level of information to improve keyword extraction. Multi-lingual keywords are another topic of interest.

Acknowledgments

The work is partly supported by the Research Fund, Thammasat University, Center of Excellence in Intelligent Informatics, Speech and Language Technology and Service Innovation (CILS), and Intelligent Informatics and Service Innovation (IISI) Research Center, the Thailand Research Fund under grant number RTA6080013, and the TRF Research Team Promotion Grant (RTA), the Thailand Research Fund under the grant number RTA6280015. The Thammasat University Fund on Research on Intelligent Informatics for Political Data Analysis, the Personnel Development Fund at Yala Rajabhat University, as well as the STEM Workforce Fund by National Science and Technology Development Agency (NSTDA).

References

[1] Siddiqi, S.—Sharan, A.: Keyword and Keyphrase Extraction Techniques: A Literature Review. International Journal of Computer Applications, Vol. 109, 2015, No. 2, pp. 18–23, doi: 10.5120/19161-0607.

[2] Hulth, A.—Megyesi, B. B.: A study on Automatically Extracted Keywords in Text Categorization. Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING ACL-44), 2006, pp. 537–544, doi: 10.3115/1220175.1220243.

[3] Wartena, C.—Alsina, M. G.: Challenges and Potentials for Keyword Extraction from Company Websites for the Development of Regional Knowledge Maps. Proceedings of the 5th International Conference on Knowledge Discovery and Information Retrieval (KDIR 2013) and the International Conference on Knowledge Management and Information Sharing (KMIS 2013) – Volume 1: SSTM, Vilamoura, Portugal, 2013, pp. 241–248, doi: 10.5220/0004660002410248.

[4] Rossi, R. G.—Marcacini, R. M.—Rezende, S. O.: Analysis of Statistical Keyword Extraction Methods for Incremental Clustering. Proceedings of the 10th of the Encontro Nacional de Inteligência Artificial e Computacional (ENIAC), Fortaleza, Brazil, 2013, pp. 1–12.

[5] Rossi, R. G.—Marcacini, R. M.—Rezende, S. O.: Analysis of Domain Independent Statistical Keyword Extraction Methods for Incremental Clustering. Learning and Nonlinear Models – Journal of the Brazilian Society on Computational Intelligence (SBIC), Vol. 12, 2014, No. 1, pp. 17–37, doi: 10.21528/lnlm-vol12-no1-art2.

[6] Zhang, C.—Wang, H.—Liu, Y.—Wu, D.—Liao, Y.—Wang, B.: Automatic Keyword Extraction from Documents Using Conditional Random Fields. Journal of Computational Information Systems, Vol. 4, 2008, No. 3, pp. 1169–1180.

[7] Lagus, K.—Kaski, S.: Keyword Selection Method for Characterizing Text Document Maps. Proceedings of the 9th International Conference on Artificial Neural Networks (ICANN '99), Vol. 1, 1999, pp. 371–376, doi: 10.1049/cp:19991137.

[8] Steinberger, R.: Cross-Lingual Keyword Assignment. Proceedings of the XVII Conference of the Spanish Society for Natural Language Processing (SEPLN-2001), Jaen, Spain, Art. No. 27, pp. 273–280.

[9] Schluter, N.: A Critical Survey on Measuring Success in Rank-Based Keyword Assignment to Documents. Proceedings of 22e Conférence sur le Traitement Automatique des Langues Naturelles (TALN '15), Caen, France, 2015, pp. 55–60.

[10] Medelyan, O.—Witten, I. H.: Thesaurus Based Automatic Keyphrase Indexing. Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '06), 2006, pp. 296–297, doi: 10.1145/1141753.1141819.

[11] Yamamoto, H.—Hanazawa, K.—Miki, K.—Shinoda, K.: Dynamic Language Model Adaptation Using Keyword Category Classification. Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010), Chiba, Japan, 2010, pp. 2426–2429.

[12] Özgür, A.—Özgür, L.—Güngör, T.: Text Categorization with Class-Based and Corpus-Based Keyword Selection. In: Yolum, P., Güngör, T., Gürgen, F., Özturan, C. (Eds.): Computer and Information Sciences – ISCIS 2005. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3733, 2005, pp. 606–615, doi: 10.1007/11569596_63.

[13] Theeramunkong, T.—Lertnattee, V.: Multi-Dimensional Text Classification. Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002), Taipei, Taiwan, Vol. 1, 2002, pp. 1002–1008, doi: 10.3115/1072228.1072383.

[14] Lertnattee, V.—Theeramunkong, T.: Multidimensional Text Classification for Drug Information. IEEE Transactions on Information Technology in Biomedicine, Vol. 8, 2004, No. 3, pp. 306–312, doi: 10.1109/TITB.2004.832542.

[15] Qiu, X.—Huang, X.—Liu, Z.—Zhou, J.: Hierarchical Text Classification with Latent Concepts. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011), Portland, Oregon, USA, Vol. 2, 2011, pp. 598–602.

[16] Silla Jr., C. N.—Freitas, A. A.: A Survey of Hierarchical Classification Across Different Application Domains. Data Mining and Knowledge Discovery, Vol. 22, 2011, No. 1–2, pp. 31–72, doi: 10.1007/s10618-010-0175-9.

[17] Shen, D.—Ruvini, J.-D.—Sarwar, B.: Large-Scale Item Categorization for e-Commerce. Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12), Hawaii, USA, 2012, pp. 595–604, doi: 10.1145/2396761.2396838.

[18] Wang, D.—Wu, J.—Zhang, H.—Xu, K.—Lin, M.: Towards Enhancing Centroid Classifier for Text Classification – A Border-Instance Approach. Neurocomputing, Vol. 101, 2013, pp. 299–308, doi: 10.1016/j.neucom.2012.08.019.

[19] Ling, W.—Dyer, C.—Black, A.—Trancoso, I.: Two/Too Simple Adaptations of Word2Vec for Syntax Problems. Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics-Human Language Technologies (NAACL HLT 2015), Denver, Colorado, 2015, pp. 1299–1304, doi: 10.3115/v1/n15-1142.

[20] Liu, Y.—Navathe, S. B.—Pivoshenko, A.—Dasigi, V. G.—Dingledine, R.—Ciliax, B. J.: Text Analysis of MEDLINE for Discovering Functional Relationships Among Genes: Evaluation of Keyword Extraction Weighting Schemes. International Journal of Data Mining and Bioinformatics, Vol. 1, 2006, No. 1, pp. 88–110, doi: 10.1504/IJDMB.2006.009923.

[21] Özgür L.—Güngör, T.: Two-Stage Feature Selection for Text Classification. In: Abdelrahman, O., Gelenbe, E., Gorbil, G., Lent, R. (Eds.): Information Sciences and Systems 2015. Springer, Cham, Lecture Notes in Electrical Engineering, Vol. 363, 2016, pp. 329–337, doi: 10.1007/978-3-319-22635-4_30.

[22] Karkali, M.—Plachouras, V.—Stefanatos, C.—Vazirgiannis, M.: Keeping Keywords Fresh: A BM25 Variation for Personalized Keyword Extraction. Proceedings of the 2nd Temporal Web Analytics Workshop (TempWeb '12), ACM, Lyon, France, 2012, pp. 17–24, doi: 10.1145/2169095.2169099.

[23] Das, B.—Pal, S.—Mondal, S. K.—Dalui, D.—Shome, S. K.: Automatic Keyword Extraction from Any Text Document Using N-gram Rigid Collocation. International Journal of Soft Computing and Engineering, Vol. 3, 2013, No. 2, pp. 238–242.

[24] Campos, R.—Mangaravite, V.—Pasquali, A.—Jorge, A. M.—Nunes, C.—Jatowt, A.: A Text Feature Based Automatic Keyword Extraction Method for Single Documents. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (Eds.): Advances in Information Retrieval (ECIR 2018). Springer, Cham, Lecture Notes in Computer Science, Vol. 10772, 2018, pp. 684–691, doi: 10.1007/978-3-319-76941-7_63.

[25] Haruechaiyasak, C.—Srichaivattana, P.—Kongyoung, S.—Damrongrat, C.: Automatic Thai Keyword Extraction from Categorized Text Corpus. Proceedings of the 1st International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI 2003), Chonburi, Thailand 2003.

[26] Beliga, S.—Meštrović, A.—Martinčić-Ipšić, S.: An Overview of Graph-Based Keyword Extraction Methods and Approaches, Journal of Information and Organizational Sciences, Vol. 39, 2015, No. 1, pp. 1–20.

[27] Özgür L.—Güngör, T.: Text Classification with the Support of Pruned Dependency Patterns. Pattern Recognition Letters, Vol. 31, 2010, No. 12, pp. 1598–1607, doi: 10.1016/j.patrec.2010.05.005.

[28] Salton, G.—Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, Vol. 24, 1988, No. 5, pp. 513–523, doi: 10.1016/0306-4573(88)90021-0.

[29] Jiang, C.—Zhu, D.—Jiang, Q.: A Dynamic Centroid Text Classification Approach by Learning from Unlabeled Data. Proceedings of the 3rd International Conference on Multimedia Technology (ICMT-13), Guangzhou, China, 2013, pp. 1420–1429, doi: 10.2991/icmt-13.2013.174.

[30] Lertnattee, V.—Theeramunkong, T.: Class Normalization in Centroid-Based Text Categorization. Information Sciences, Vol. 176, 2006, No. 12, pp. 1712–1738, doi: 10.1016/j.ins.2005.05.010.

[31] Ebert, S.—Adrian, B.: Detecting Documents with Complaint Character. Proceedings of Lernen, Wissen, Adaption (Learning, Knowledge, Adaptation) (LWA 2013), 2013, pp. 59–62.

[32] De Boom, C.—Van Canneyt, S.—Demeester, T.—Dhoedt, B.: Representation Learning for Very Short Texts Using Weighted Word Embedding Aggregation. Pattern Recognition Letters, 2016, Vol. 80, pp. 150–156, doi: 10.1016/j.patrec.2016.06.012.

[33] Manning, C. D.—Raghavan, P.—Schütze, H.: An Introduction to Information Retrieval. Cambridge University Press, 2009.

Boonthida Chiraratanasopha received her B.Sc. degree in computer science from the Prince of Songkla University in 1996 and M.Sc. degree in applied statistics from the National Institute of Development Administration in 1999. She works at the Yala Rajabhat University.

Salin Boonbrahm received her B.Sc. degree in mathematics from the Prince of Songkla University in 1981 and M.Sc. degree in applied statistics from the National Institute of Development Administration in 1984. She works at the Walailak University, Nakorn Si Thammarat, Thailand. Her current research interests include decision support system, human-computer interaction, augmented reality in education, library automation system.

Thanaruk Theeramunkong received his Bachelor's degree in electric and electronics, and the Master's and the doctoral degrees in computer science from the Tokyo Institute of Technology, Tokyo, Japan, in 1990, 1992, and 1995, respectively. He works at Sirindhorn International Institute of Technology (SIIT), Thammasat University, Pathumthani, Thailand. His current research interests include data mining, machine learning, natural language processing, information retrieval, and knowledge engineering.

Computing and Informatics, Vol. 40, 2021, 83–103, doi: 10.31577/cai_2021_1_83

MULTI-PLATFORM INTELLIGENT SYSTEM FOR MULTIMODAL HUMAN-COMPUTER INTERACTION

Mateusz Jarosz, Piotr Nawrocki, Bartłomiej Śnieżyński

Institute of Computer Science
Faculty of Computer Science, Electronics and Telecommunications
AGH University of Science and Technology
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: {mateusja, piotr.nawrocki, bartlomiej.sniezynski}@agh.edu.pl

Bipin Indurkhya

Institute of Philosophy
Jagiellonian University
Golebia 24, 31-007 Krakow, Poland
e-mail: bipin.indurkhya@uj.edu.pl

Abstract. We present a flexible human–robot interaction architecture that incorporates emotions and moods to provide a natural experience for humans. To determine the emotional state of the user, information representing eye gaze and facial expression is combined with other contextual information such as whether the user is asking questions or has been quiet for some time. Subsequently, an appropriate robot behaviour is selected from a multi-path scenario. This architecture can be easily adapted to interactions with non-embodied robots such as avatars on a mobile device or a PC. We present the outcome of evaluating an implementation of our proposed architecture as a whole, and also of its modules for detecting emotions and questions. Results are promising and provide a basis for further development.

Keywords: Human–computer interaction, multi-platform, intelligent system architecture, multimodal system, humanoid robot

1 INTRODUCTION

Social humanoid robots are becoming more and more commonplace. Their capabilities are also increasing rapidly: they are equipped with a variety of sensors to obtain information about the surrounding environment, including people, along with a variety of mechanisms to perform human-like actions naturally. Therefore, the importance of the Human–Robot Interaction (HRI) research is increasing.

There is a perceived need for a system that could be used during HRI experiments. Commercial manufacturers provide software frameworks for programming their respective robots and executing code (e.g. NAOqi). Systems developed by researchers are also becoming available [18, 20]. The problem is that such systems are usually closed, engineered to operate on a specific type of robot, and cannot be deployed on mobile devices.

One of the goals of our research is to improve the level of human–robot interaction (HRI) and design social robots with which humans can interact intuitively. To achieve this, we have designed and implemented an architecture that allows flexible interaction between humans and robots, which is the main contribution of this paper. In this architecture, the emotional state and the mood of the user are sensed on the basis of the users' dialogue with the robot, their posture and gesture, speech prosody, facial expression and eye gaze. An appropriate behaviour is then selected on the basis of a multi-path scenario. This architecture can be easily adapted to interactions with non-embodied robots such as avatars on mobile devices or PCs. Here, we describe our proposed architecture and report on a series of evaluation experiments. We also compare our architecture with other similar systems.

This article is structured as follows. Section 2 discusses related work. Section 3 contains a high-level description and implementation details of our proposed architecture. In Section 4 we describe evaluation experiments and their results. Section 5 compares our architecture with other similar approaches, while concluding remarks and directions for further research are presented in Section 6.

2 RELATED WORK

In recent years, many human–robot interaction architectures have been proposed based on behaviour trees, multimodal systems and adaptive systems. In this section we briefly describe some of these systems, along with their advantages and disadvantages. We focus on behaviour planning, decision-making using behaviour trees, and benefits of following a multimodal approach.

Alonso-Martin et al. [16] propose a multimodal emotion detection system as part of a larger Human–Robot Interaction system. It uses two channels of emotion detection, namely voice and face video, which are combined into one emotion value. A dialogue system is driven by this emotion value, acknowledging the intended effect

on the user. Our proposed architecture uses a similar approach, but we also incorporate information gleaned from the ongoing human–robot dialogue to determine the emotional state of the user.

In an earlier work, Breazeal [17], using her expressive anthropomorphic robot *Kismet*, studied emotions and expressive behaviours in regulating social interaction between a human and a robot in communicative and teaching scenarios. In this work, models of humanoid robot emotions and their scientific basis are described, and adapted for implementation in *Kismet*. They also use the prosody of the user's voice to detect their emotional stance. In the current prototype of our architecture we rely on face video for emotion detection, but we also plan to incorporate speech prosody in future versions. Moreover, at the moment we work with a non-expressive humanoid robot *Pepper*, but we plan to use expressive humanoid robots such as *Little Einstein* in the future.

More recently, Coronado et al. [20] proposed a robot programming framework and an interface for the development of usable and flexible end-user applications. The framework employs a component-based methodology, a block- and web-based interface, and a behaviour tree approach to designing robot behaviour, all of which can be combined to adopt the end-user development paradigm. This system is easy to use from the end-user perspective, and cross-platform tools like ROS and ZeroMQ are provided to enable the creation of platform-independent applications. It can also be expanded with new sensory devices or robots. Our architecture shares the multi-platform approach with the platform of Coronado et al., but the usability of our interface will be addressed in the future versions.

Beer et al. [18] developed a framework for Levels of Robot Autonomy (LORA), ranging from teleoperation (non-autonomous) to fully autonomous. Their framework proposes a 10-point taxonomy for LORA, and relates it to three HRI variables, namely acceptance, situation awareness, and reliability. However, compared to our architecture, this work focuses mostly on autonomous robot operation.

A human–robot interaction framework that outlines a general structure of future home service robots to assist humans in their home-based daily activities was proposed by Lee et al. [19]. The authors describe three main interaction modules: multimodal, cognitive, and emotional. The main function of the multi-modal interaction module is to make the interaction intuitive for the human user. The cognitive interaction module facilitates cooperative sharing of tasks, while the emotional interaction module maintains a close relationship between the human and the robot. Our framework is also multimodal in terms of accepting inputs from various kinds of sensors. We furthermore provide a cognition/perception module to infer higher-level conceptual information from the basic input devices. For example, we can extract the human's emotional state from a video feed (see Subsection 4.2), as long as the subject remains visible.

Ardila et al. implemented an adaptive controller for a robot arm [2], which can adapt motion trajectories to the environment and an overall robot interaction profile. This adaptive controller uses the PAD emotional model (Pleasure, Arousal and Dominance), where PAD values are used to change the strategy of robot movements.

This system generates affective motions in non-humanoid robots for more intuitive human–robot interaction. In our approach, we focus on adjusting behaviours instead of movements and our solution is capable of working on PCs, mobile devices and humanoid robots.

Rincon et al. [3] propose a novel cognitive-robot control architecture to adapt robot actions and motions to the dynamics of both the environment and the human. This solution involves incorporating "expressive states" in a cognitive model that adapt to yield optimal robot control. The authors also provide deep-learning algorithms for perception, cognitive models based on affects, and adaptive generalized predictive controllers (AGPC). Their system also relies on the PAD concept to represent the robot's emotional state. Adaptation is controlled by an AGPC, which changes according to the cognitive state of the robot. The AGPC cost functions are calculated using PAD values. An evaluation of the system showed that the robot was able to perform tasks continuously with expressive and personalized behaviours. This research focused on non-humanoid robots and adaptation of robot movements, whereas our system is targeted for PCs, mobile devices and humanoid robots, and we focus on adjusting behaviours as a whole rather than partially.

A framework based on an adaptive predictive control scheme and a fast dynamic and geometric identification process was proposed by Hagane et al. [4]. The approach was demonstrated with a force-controlled wall-painting task performed by a lightweight robot called KUKA. This research also includes a comparative analysis of the performance of generalized predictive control (GPC), adaptive proportional derivative gravity compensation, and adaptive GPC (AGPC). The results revealed that predictive controllers are more suitable than adaptive PD controllers with gravitational compensation, owing to the use of well-identified geometric and inertial parameters. This work also focused on non-humanoid robots and on performing movements adapted to the changing environment. In contrast, our research focuses on adapting more general behaviours, and is targeted not only for robots, but also for PCs and mobile devices.

Abiyev et al. proposed a novel behaviour tree (BT)-based control for decision-making in robot soccer [5]. The robot analyzes the current world state and decides how to act. The BT approach allows modelling of complicated situations with ease, which constitutes an advantage of this technique over finite state machines, which are widely used in robot control. An evaluation of the system performed by the authors reveals that BT performs well at the task of playing robot soccer. Though the use of BT in this system is similar to our approach, Abiyev's work is more focused on movements and goal-oriented decision-making, whereas our research is focused on adapting the behaviour flow in a multi-platform system.

Marzinotto et al. [6] proposed a unified BT framework along with notions of equivalence between BTs and Controlled Hybrid Dynamical Systems. They also demonstrate the applicability of their framework to real systems by scheduling open-loop actions in a grasping mission for the Nao robot. Their proposal to use BT for movement control is interesting, but differs from our approach in that we use a structure similar to BT for making decisions in a scenario tree.

Arriaga et al. propose a novel approach to emotion and gender recognition using only camera for human–robot interaction systems [11]. This approach uses a convolutional neural network (CNN) based on the simplified Google Xception model architecture. The system, as implemented by the authors, achieved 96 % accurancy in the IMDB gender dataset and 66 % in the FER-2013 emotion dataset, whereas humans achieve an emotion detection accuracy of 65 % ± 5 % in the same dataset, with the best solution peaking at 71 %. A major advantage of this model is its low computational cost, which is achieved by cutting the number of network parameters from 600 000 in the naive CNN implementation to 60 000, which corresponds to a tenfold reduction compared to their initial naive implementation, and 80-fold compared to the original CNN. This improvement enables the robot to run both networks at the same time and obtain results in real time. In our work, we rely on similar concepts for network design, but retrain the network with a larger data set, and change the face detection method to make it faster and avoid frames with undetected faces in high-framerate feeds.

Question detection in human–robot interaction is important: even if the robot cannot answer the question, it should react to it in some way. Ando et al. [12] propose a novel approach for question detection using lexical cues in addition to acoustic data. They also proposed their own framework for training the network, called feature-wise pre-training, which combines acoustic and phonetic features effectively. Their system achieved 66.8 % precision and 62.8 % recall for question detection. These are remarkable results, but we nevertheless decided to use CNN and mel spectrograms based on acoustic event detection for faster execution using the simplified Xception model.

Inception modules in conventional neural networks can be interpreted as an intermediate step between regular convolution and depthwise separable convolution operation (depthwise convolution followed by pointwise convolution). In this light, depthwise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation led Chollet [13] to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depthwise separable convolutions of an Xception architecture. This architecture achieved slightly better results on the imageNet dataset than the inception model. In our work, we use a simplified version of the Xception model (fewer layers) for emotion and question detection.

Zhang et al. described an approach to sound event detection using conventional neural networks [14]. Conventionally, sound event recognition methods based on informative front-end features such as MFCC, or with back-end sequencing methods such as HMM, tend to perform poorly in the presence of interfering acoustic noise. As noise corruption is usually unavoidable, Zhang et al. proposed to use CNN and spectrograms as a more robust solution. This method achieved excellent performance under noise-corrupted conditions compared to the conventional state-of-the-art approaches in standard evaluation tasks. In our research, we apply the same basic idea to recognise questions, treating the last two-thirds as a sound event.

Perzanowski et al. presented a multimodal system for human–robot interaction [7] using three different sources of commands: speech, gestures and PDA interaction. They assumed that with an integrated system, the user will be less concerned with the means of communicating, and can therefore concentrate on the tasks and goals at hand. Though this research is old, it still incorporates basic emotions and demonstrates the advantages of a multimodal approach. We apply the same principle in designing our system.

Many research papers have been published in the area of human–robot interaction, adaptive movement or movement of robot parts, including the benefits of behaviour trees and multimodal systems. Among the important directions of HRI research is detection and analysis of emotions with the use of voice and face image. An important aspect, which, as indicated by a study of the available literature, has not been further explored, is the possibility of extending analysis of emotions to include information obtained from the dialogue between a robot and a human, as well as data from various types of sensors (for example, determining the robot's position). Detecting emotions, including, primarily, image analysis performed for this purpose, may exploit machine learning methods such as CNN. It is important that these methods are trained on as large a data set as possible in order to obtain the best possible results. As a result, the emotional attitude of the user may be accurately determined. In parallel, an important conclusion from the presented analysis is the need for universality of the developed solutions. Many of the systems analyzed in this section are designed only for a specific device or class of devices. Evidently, there is no existing multimodal human-computer interaction system that incorporates – whether fully or partially – behaviour trees for adaptation of behaviours, and is capable of operating on multiple platforms (such as robots and mobile devices). This is precisely what motivated us to initiate the research presented in this paper.

3 ARCHITECTURE FOR MULTIMODAL HUMAN-COMPUTER INTERACTION

Our architecture for multimodal human-computer interaction (see Figure 1) consists of the following modules: Perception modules, Actions, Activity scripts and Behaviour planner. Activity scripts represent scenarios, and are stored in the JSON format. These scripts are used by the Behaviour planner to enact a scenario. The Behaviour planner is the main module responsible for executing activity scripts and choosing appropriate paths in the script according to data from perception modules. Perception modules are responsible for extracting higher-level data about the user's interaction with the robot, based on raw data from the robot's sensors. In the future, we plan to augment this with external sensors. In the current prototype, there are four perception modules: facial expression, gaze tracking, dialogue monitoring, and speech prosody. The action module is responsible for executing simple actions, such as saying something, moving forward, changing the direction of the robot's gaze,

making hand gestures, or displaying a video. Different actions can be combined for more complex expressive movements.

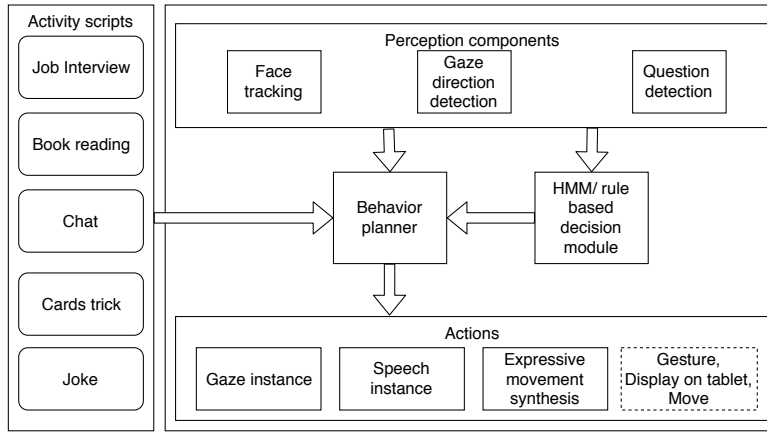


Figure 1. Framework architecture

Our proposed architecture has three main features:

Multi-platform: We designed our architecture to be able to work with different robotic platforms. At the moment, we mostly work with the Pepper robot, but we are planning to use other robots, such as Little Einstein, and also non-embodied avatars running on computers or mobile devices. To incorporate this multi-platform capability, we designed our architecture in such a way that only the part responsible for robot/avatar actions and extracting sensory data is dependent on the platform. All other parts of the system can be executed on any platform equipped with a Python interpreter.

Multimodality: Obtaining information from multiple sensors and analysing it simultaneously can result in significant computing load, especially for a mobile device with limited resources. To address this issue we decided to move most of the analysis tasks to the cloud as microservices. This approach has another benefit in that different platforms can share the same functionality through the cloud to avoid duplication. Moreover, as new features are added to the cloud, they become available across all platforms. In this way, we were able to deploy e.g. emotion and question detection facilities.

Intelligent system: Through a number of in-the-wild studies on child-robot interaction [21, 22], we have found that in order for a robot to behave naturally towards a human it needs to have a basic understanding of what humans are saying and where they are looking, along with some awareness of the surrounding environment. For example, the robot should be aware when a human is asking it something and should answer appropriately. We plan to achieve this

by using a multi-path scenario and an event system with break points. This is explained in detail below.

Initially, we tried a simple approach, shown on the left in Figure 2, where there was only one scenario, with the robot executing one behaviour. Clearly, this is not very flexible, for the robot should be able to take different actions and execute different behaviours depending on the context and user actions. To address this issue, we created a module containing several activity scripts. These were organised in a tree structure, with each node equipped with a condition, which determines when the given path should be followed, and a sequence of actions (e.g. speak, move, etc.) that the robot will perform while traversing the node. There are break points between actions where events can be executed. Each event consist of a type and a series of actions. Activity scripts also allow parallel looped paths for executing continuous or regularly repeating actions during a scenario. We can express our scenario structure formally in the following way:

$$scenario = (N, E, A, C, Ne, s) \tag{1}$$

where N is a set of nodes, and E is a set of edges. Nodes contain one or more actions from set A, a condition from set C and events from the set of events Ne: for all $n \in N$, node n is labeled by (ne, a, c), where $ne \in Ne$, $a \in A$, $c \in C$. Every event $ne \in Ne$ is labeled by action and condition: $(a, c) \in A \times C$. Every condition assumes the form of K-SAT:

$$\left(x_1^1 \wedge x_1^2 \wedge \cdots \wedge x_1^{n_1}\right) \vee \left(x_2^1 \wedge x_2^2 \wedge \cdots \wedge x_2^{n_2}\right) \vee \cdots \vee \left(x_k^1 \wedge x_k^2 \wedge \cdots \wedge x_k^{n_k}\right) \tag{2}$$

where x_i^j is a basic formula of the form $a == | <= | >= |! = | > | < b$, a and b are constant values or variables representing object states, results of detecting humans or emotions, etc. Ne are events defined per node. Every event has a set of actions and a triggering condition. *Scenario* has one starting node $s \in N$ and can have many leaf nodes and cycles.

Algorithm 1 shows how action scripts work. The MAIN procedure contains the main decision loop, where we browse the graph for child nodes, evaluating the condition and then choosing the best node to follow, e.g. the first one for which the condition evaluates to true. In addition to sequential nodes, we also identify parallel ones. Such nodes are run in separate threads and they perform repeating tasks with little to no impact on the main flow of the script, e.g. moving the robot's head in a repeating pattern. Parallel nodes can be stopped at any time, or terminate themselves, but are usually stopped at the end of the script. The next step involves execution of nodes (RUN procedure), with an event mechanism that operates as follows. Before taking any action, we register, in a decision module, all events defined for the given node. Event conditions are monitored in a parallel thread (see the PROCESS_EVENTS procedure). If a condition evaluates to true, a corresponding event is executed. Execution is synchronised with the main flow of scenario by locking a mutex which corresponds to the node. Actions are then performed sequen-

tially. While this goes on, a decision module can decide to trigger – based on data from perception components – one of the registered events. Such triggered events can start immediately, if the currently running action permits this, or they can be executed after the current action has ended. After executing all actions, the events are deregistered, and will not be called unless registered again.

Algorithm 1 Behavior planner algorithm

```
procedure MAIN
    graph ← read_node_graph()
    node ← get_current_node(graph)                          ▷ get root node
    while graph.has_next() do                    ▷ if current node has children
        graph.set_current_node(node)
        RUN(node)
        successors ← get_successors(node)
        pss ← get_passing_sequential_successors(successors)
        pps ← get_passing_parallel_successors(successors)
        for each parallel_successor ∈ pps do
            start_thread(parallel_successor)
        node ← get_best_successor(pss)
procedure RUN(node)
    if check_node_start_condition(node) == True then
        for each event ∈ get_events(node) do
            add_event_to_event_monitoring_list(event)
        Execute in a new thread
          PROCESS_EVENTS(event_monitoring_list, thread)
        for each action ∈ get_actions(node) do
            lock.acquire                ▷ Check if some event is not triggered
            action.run()
            lock.release
        for each event ∈ get_events(node) do
            remove_event_from_event_list(event)
        Stop thread
procedure PROCESS_EVENTS(event_monitoring_list, thread)
    while thread is running do
        for each event ∈ event_monitoring_list do
            if check_event_condition(event) == True then
                lock.acquire
                execute_action(event)
                lock.release
```

This architecture is loosely based on the concept of a behaviour tree [6]. Though behaviour trees have a formal definition and are compact, they are difficult to use for new users due to their non-intuitive structure. In contrast, directional graphs with

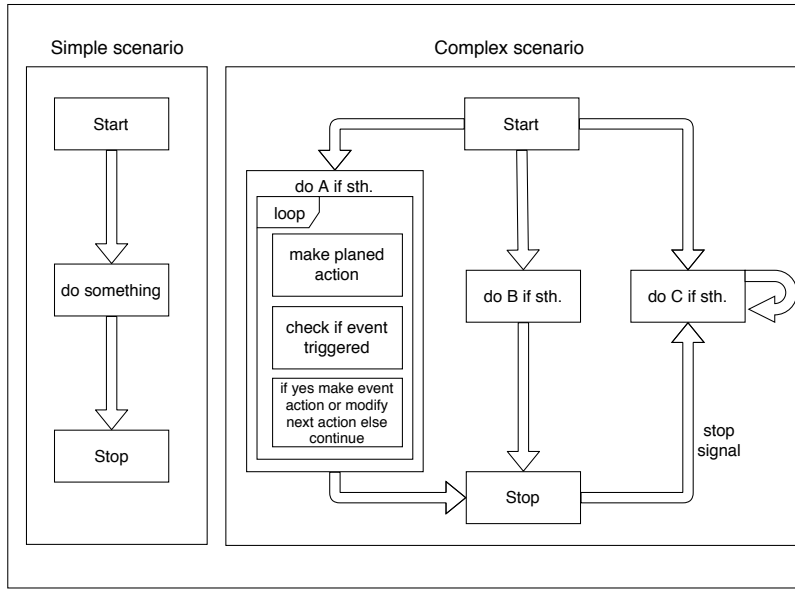


Figure 2. Comparison between a simple scenario and a complex scenario

entry conditions are natural for computer scientists and can be readily used with myriads of algorithms. Therefore, our representation is similar to a state machine or a classical directional graph with conditional nodes.

In order to improve performance and to incorporate a multi-platform approach, all the heavy computing perception modules, except the basic question detection module, were implemented as microservices. In our previous research [10], we tested several approaches to detecting the user's gaze direction with a camera mounted on a robot or a mobile device. Based on these results, we decided to use the OpenFace library [8] for estimating the user's gaze direction. Our solution also employs a system we developed ourselves for determining the user's gaze direction based on facial landmark [9] and pupil detection.

A simple approach to detecting questions is based on the assumption that the final second of a question exhibits higher pitch then the preceding several seconds. This solution is simple enough to be computed locally rather than by uploading the recording to the network, which might result in higher costs and increased complexity. Therefore, this module was not implemented as a microservice.

A more complex approach would involve text-based question detection based on a transcript returned by Google Cloud, followed by a dialog-act classification. This method was not implemented in our framework, because initial tests revealed high latency and cumulative error introduced by two main components: Polish speech-to-text conversion and dialog-act classification. Other important issues included the cost of using Google Cloud services, privacy issues (uploading the voice feed

to Google Cloud) and the difficulty of implementing continuous detection (Google Cloud limits streaming duration to 1 minute). Consequently, we decided to use dialog-act classification using Mel Spectrograms of length 3 s, with one-second overlap, as input for a CNN classifier. The CNN architecture is based on the Xception network with five Xception blocks and 200 000 parameters: we found that this works well for rapid computation on relatively slow devices.

The training data set for the CNN was prepared as follows. First, we partitioned audiobooks into distinct sentences and questions. All pieces containing more than one sentence were then removed. Finally, we augmented the sound samples by cropping 10 % of the data randomly, increasing volume by a factor between 2 and 5 (randomly), masking 10 % of possible data and replacing it with 0 s, adding 0.03 of random noise, using VTLP (Vocal Tract Length Perturbation), changing pitch, and varying speed by a factor between 0.5 and 2.0. For all data files, no augmentation was applied in the first 20 % and the final 20 % of the audio file (as in [23]), and we took the last 3 seconds of the data to extract its spectrogram. The training data set was in Polish and consisted of 10 500 samples (3 600 before augmentation), equally divided into questions and non-questions. The testing data set contained 7 500 samples with 1 150 questions (20 % of questions in the training data set prior to augmentation), similar to real-life dialogue.

For emotion detection, we retrained the network used in [11] by adding one Xception block and extending the training set threefold, with additional 100 000 images from "The Ryerson Audio-Visual Database data set" [15]. We also found that using CNN to find the face in an image, instead of using a hog classifier, results in a significant improvement in the face detection rate.

The main module for the robot is written in Python, and consists of the following sub-modules: The Action-script reader reads an action script and returns a directional graph with one root and many possible paths to the end. The Configuration manager is responsible for preparing the system environment. The Behaviour engine is responsible for executing the graph produced by the Action-script reader. It decides which of the multiple paths is to be followed based on data from the Analysis module. The Analysis module collects data from the perception modules. It can perform local processing to further perform statistical analysis of the raw sensory data robot and/or combine data from different sensors synchronously or asynchronously, using sensor fusion algorithms. This module is also responsible for triggering events based on the collected/analysed data. Actions are primitive operations stored as objects inside the graph produced by the Action-script reader.

4 EVALUATION

To evaluate the efficacy of our architecture, especially with regard to emotion and question detection modules, we conducted an experiment along with a number of field studies. Below, we describe these tests and present their results.

4.1 Verification of Scenario Execution

The first experiment involved an interview setting: the Pepper robot was the interviewer and a human participant was the interviewee. The robot asked a number of predetermined questions, and listened to responses from the interviewee. Figure 3 shows results before and after experiment surveys. In the aftermath of the interview, subjects claimed to be less concerned about the robot.

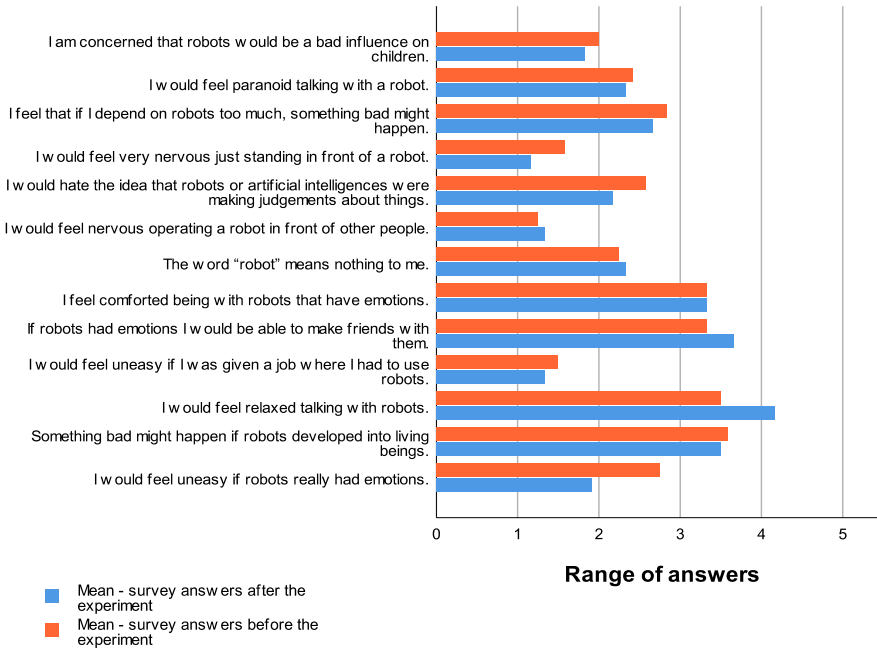


Figure 3. Results before and after experiment survey (1)

Subsequently, we conducted a number of field studies (in-the-wild studies) in which the Pepper robot interacted with children. Here, we discuss results from two such studies. The first study involved kindergarten children (age 4–6) at a Polish school in Kraków. The interaction scenarios were designed to match the children's capabilities, and included a drawing activity, a reading session (Pepper reads to the children), dancing with the robot, and a question-answer session using the Wizard-of-Oz paradigm (meaning that a human experimenter answered the children's questions that were delivered through Pepper).

We conducted another such workshop with older children (age 5–13), where we introduced a rock-paper-scissors game with Pepper, along with an extended question-answer session. All experiments proceeded without any problems affecting either the software or the robot, and were well received by the children.

The presented experiments were performed using a simplified version of our software (without emotion recognition and question detection); thus, to confirm that the system works well when equipped with new modules, we conducted a small study with the same setup as in the interview experiment, including emotion recognition and question detection. Participants gave positive responses when robot reacted to their behaviours. The questionnaires filled out by participants before and after the interview (see Figure 4), reveal an increase in the level of anxiety. This was due to the robot's reaction to the behaviour of the participants, e.g. having detected that a participant was sad, the robot asked if everything was okay. Such interruptions could have prevented the participant from answering a previously asked question, which made the participant feel a bit confused. This minor problem will be addressed in a future version of our system. There were no other problems. The system performed well and participants did not report any other issues.

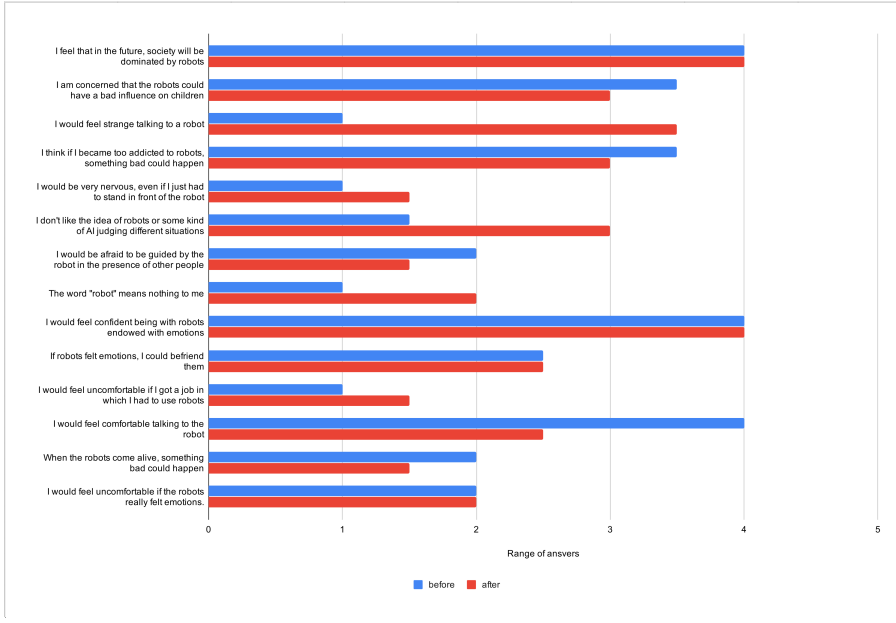


Figure 4. Results before and after experiment survey (2)

4.2 Evaluation of the Emotion Classification Module

As mentioned above, we used a modification of the network proposed in [11] and retrained it on a larger dataset. We also changed the face detection classifier from the haar classifier in the OpenCV library to the CNN classifier in the dlib library. The main reason for this change was better performance of the CNN classifier compared to the haar classifier. In our tests, haar dropped many frames without detecting

a face even though the face was clearly visible: in one experiment, the haar classifier found the face in only 4 of 81 frames in one file, whereas CNN recognised the face in 75 of 81 frames. The average rate of frames with undetected faces was 15 % when using haar, but drooped to 8 % when using the CNN classifier.

For improving the efficacy of emotion detection we retrained our model on a larger dataset. We added one additional Xception block to the network to increase the model's capacity, and added face scans extracted from the Ryerson Audio-Visual Database [15] to the FER-2013 dataset. This caused an unbalance in the dataset due to the lack of faces displaying disgust or surprise in the Ryerson Audio-Visual Database. Figure 5 shows the confusion matrix of the retrained neural network on the test dataset: we can observe that while results are generally promising, disgust detection has a lower accuracy compared to other emotions like anger and sadness.

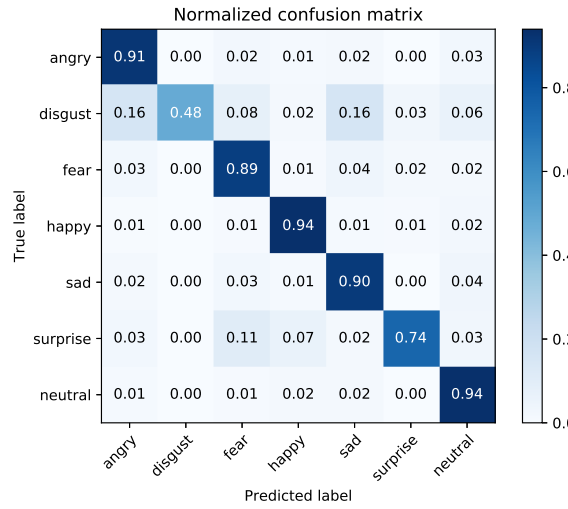


Figure 5. Confusion matrix on the emotion test dataset

The average accuracy of the model is 86 % with 83 % recall.

4.3 Evaluation of Question Discovery Module

First we attempted a simple approach, resulting in peak accuracy of 70 % for one class and 30 % for the other class, time or 55 % for both classes combined. A text-based approach was abandoned after preliminary testing. An evaluation (confusion matrix) of the question detection module using the CNN and Mel spectrograms is shown in Figure 6.

On average, our model achieved 70 % precision and 80 % recall due to an unbalanced test data set; however, when weighted with the number of samples, the results improved to 88 % precision and 82 % recall.

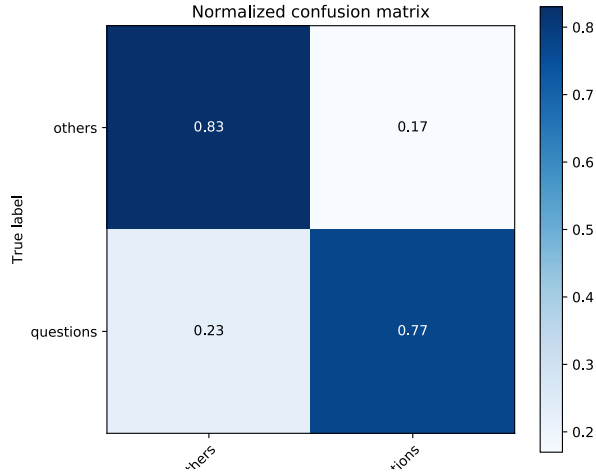


Figure 6. Confusion matrix on audio test dataset

5 COMPARISON WITH OTHER FRAMEWORKS

In this section we compare the main features of our architecture with other existing systems for HRI. We selected external systems on the basis of the following criteria: functional similarity, creation in the last 7 years (most are 2-3 years old) and popularity (most have more then 10 citations). Other important factors included a clear description of the system and a similar range of applications. We could not test all other systems ourselves, as all information we have is based on published papers and therefore the number of features which can be compared is limited to those mentioned in such papers. We chose six features which are common to all systems and mentioned in their respective publications. The first and second features, "Target environment" and "Compatible robots and environments", correspond to one of our main goals: multi-platform operation. As the next feature we chose "Understanding emotions" – we believe that recognising human emotion and exploiting that information in the course of conversation is important and represents the future of HRI, similar to "Question detection". Next, we focused on the presence of a GUI as it can greatly lower the skill threshold for interaction scenarios and is important for that reason. The final two features, "Allows modification of scenario according to changing environment" and "Decision algorithm" correspond to our other goals, namely adaptivity and "intelligence" of the system. Some systems include other important features, such as gesture recognition, but those features are only present in a minority of the analyzed solutions, so we decided to omit them.

Table 1 briefly summarizes the main features of five popular HRI systems, as well as of our solution.

	Our solution	Alonso-Martin [16]	Ardila [2]	Coronado [20]	Röning [24]	Liu [25, 26]
Target environment	Multi-platform	Multi-platform	Single-platform	Multi-platform	Single-platform	Multi-platform
Compatible robots and environments	Pepper/nao family, virtual avatar on mobile and PC	Pioneer robot/ROS compatible	Robo arm	Nao	Minotaurus	Nao, mobile robot
Understanding emotions	Seven discrete emotions for better interaction	Seven discrete emotions for better interaction	PAD (Pleasure, Arousal, and Dominance) for modifying movements	None	Six discrete emotions for better interaction	Six discrete emotions for better interaction; also the robot can show emotions
Question detection	Yes	No	No	No	Yes	No
GUI	No	No	No	Yes	No	No
Allows modification of scenario according to changing environment	Yes	Yes	Yes	Yes	Yes	Yes
Decision algorithm	Conditional Tree	Rules	Robust Generalised Predictive Control	Behaviour trees	N/A	N/A

Table 1. Comparison of HRI system features

All systems enable making adjustments in the scenario depending on changes in the environment, for example user emotions. Our system, as well as the systems described by Alonso-Martin, Röning and Liu use discrete emotion representation, while the Ardila system uses a continuous pleasure arousal model. Both approaches have advantages – in our opinion, discrete representation is simpler and easier to understand. Moreover, in the Ardila system emotions mean the state of the robot, while in other systems they express the state of the human in the environment, and the Liu system not only recognizes human emotions, but can also display robot emotions. Only the Coronado system does not employ an emotion detection module. We think that emotions are one of most important channels in human communication; what is more, humans tend to be more open when talking with a robot, showing their emotions freely. Due to those facts, we believe that an emotion detection module is an essential component of an HRI system. Our system includes a question detection module; Alonso-Martin and Coronado systems can use one, if available, due to their modular architecture. The question detection module enables more natural conversation between the human and the robot, and is also simpler and less demanding then a speech-to-text solution with an advanced chatbot to interpret and respond to user speech. The Röning system uses such an online chatbot to answer user questions. Our system can be run on most platforms and can interact not only with robots, but also with virtual agents. Alonso-Martin and Liu systems are also compatible with many robots.

Our system uses conditional trees whereas the Alonso-Martin system uses rules, the Adrila system relies on a special algorithm, the Coronado system uses behaviour trees and the remaining two systems use an unknown decision mechanism. In our opinion our approach is equivalent to the rule-based approach in terms of simplicity, and incorporates the advantages of the hierarchical approach, such as behaviour trees. Only the last system has a graphical interface, but we plan to create one for our system in the near future. From this simple comparison we can see that our solution is needed because its features are not replicated by other freely available systems. What is more, our system is more flexible than other solutions since it is able to operate on most platforms and to be started on many systems.

6 CONCLUSIONS

Our goal was to create an architecture for human–robot interaction which is intuitive and incorporates the emotional state of the user. We developed an approach based on behaviour trees for controlling the flow of interaction. To evaluate our architecture, we implemented a prototype system and conducted a number of experiments in varying conditions. We also carried out a detailed comparison of our system with other similar systems. The results demonstrate the flexibility of our architecture, which allows a robot to react to human questions in an appropriate way. We also achieved good performance on the test data set with our recognition module.

The architecture is universal (can be applied in many scenarios), distributed and heterogeneous: less demanding services can be located on the robot platform or mobile device, while more complex ones (e.g. using neural networks to process images or signals) can be offloaded to the public cloud or a local PC. This enables efficient processing of multi-modal data. Owing to the microservice approach, the system can be adapted to work in other environments (e.g. Virtual Machines instead of the cloud and PC) and new hardware platforms (other robots or VR avatars). Currently a single control module (Behaviour planner) is responsible for executing a synchronous main scenario, represented by a graph, and processing asynchronous events in parallel. In the future, this approach can be scaled up and several instances of the Behaviour planner, controlling different robots, can be deployed.

In the future we also plan to conduct more extensive tests of the system and generate an improved version based on user feedback.

Acknowledgements

The research presented in this paper was supported by funds from the Polish Ministry of Science and Higher Education allocated to the AGH University of Science and Technology. Mateusz Jarosz's work was supported in part by the National Center for Research and Development (NCBR) under Grant No. POLTUR2/5/2018. We wish to thank Anna Kolota for her assistance with testing.

REFERENCES

[1] Fine, S.—Singer, Y.—Tishby, N.: The Hierarchical Hidden Markov Model: Analysis and Applications. Machine Learning, Vol. 32, 1998, No. 1, pp. 41–62, doi: 10.1023/A:1007469218079.

[2] Ardila, L. R.—Coronado, E.—Hendra, H.—Phan, J.—Zainalkefli, Z.—Venture, G.: Adaptive Fuzzy and Predictive Controllers for Expressive Robot Arm Movement During Human and Environment Interaction. International Journal of Mechanical Engineering and Robotics Research, Vol. 8, 2019, No. 2, pp. 207–219, doi: 10.18178/ijmerr.8.2.207-219.

[3] Rincon, L.—Coronado, E.—Law, C.—Venture, G.: Adaptive Cognitive Robot Using Dynamic Perception with Fast Deep-Learning and Adaptive On-Line Predictive Control. In: Uhl, T. (Ed.): Advances in Mechanism and Machine Science (IFToMM WC 2019). Springer, Cham, Mechanisms and Machine Science, Vol. 73, 2019, pp. 2429–2438, doi: 10.1007/978-3-030-20131-9_240.

[4] Hagane, S.—Ardila, L. K. R.—Katsumata, T.—Bonnet, V.—Fraisse, P.—Venture, G.: Adaptive Generalized Predictive Controller and Cartesian Force Control for Robot Arm Using Dynamics and Geometric Identification. Journal of Robotics and Mechatronics, Vol. 30, 2018, No. 6, pp. 927–942, doi: 10.20965/jrm.2018.p0927.

[5] Abiyev, R. H.—Akkaya, N.—Aytac, E.: Control of Soccer Robots Using Behaviour Trees. 2013 9th Asian Control Conference (ASCC), IEEE, Istanbul, Turkey, 2013, pp. 1–6, doi: 10.1109/ascc.2013.6606326.

[6] Marzinotto, A.—Colledanchise, M.—Smith, C.—Ögren, P.: Towards a Unified Behavior Trees Framework for Robot Control. 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 2014, pp. 5420–5427, doi: 10.1109/icra.2014.6907656.

[7] Perzanowski, D.—Schultz, A. C.—Adams, W.—Marsh, E.—Bugajska, M.: Building a Multimodal Human–Robot Interface. IEEE Intelligent Systems, Vol. 16, 2001, No. 1, pp. 16–21, doi: 10.1109/mis.2001.1183338.

[8] Wood, E.—Baltrusaitis, T.—Zhang, X.—Sugano, Y.—Robinson, P.—Bulling, A.: Rendering of Eyes for Eye-Shape Registration and Gaze Estimation. Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pp. 3756–3764, doi: 10.1109/iccv.2015.428.

[9] Baltrusaitis, T.—Robinson, P.—Morency, L.-P.: Constrained Local Neural Fields for Robust Facial Landmark Detection in the Wild. Proceedings of the IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 2013, pp. 354–361, doi: 10.1109/iccvw.2013.54.

[10] Jarosz, M.—Nawrocki, P.—Placzkiewicz, L.—Sniezynski, B.—Zielinski, M.—Indurkhya, B.: Detecting Gaze Direction Using Robot-Mounted and Mobile-Device Cameras. Computer Science, Vol. 20, 2019, No. 4, doi: 10.7494/csci.2019.20.4.3435.

[11] Arriaga, O.—Valdenegro-Toro, M.—Plöger, P.: Real-Time Convolutional Neural Networks for Emotion and Gender Classification. 2017, arXiv: 1710.07557.

[12] Ando, A.—Asakawa, R.—Masumura, R.—Kamiyama, H.—Kobashikawa, S.—Aono, Y.: Automatic Question Detection from Acoustic and Phonetic Features Using Feature-Wise Pre-Training. Proceedings of INTERSPEECH, 2018, pp. 1731–1735, doi: 10.21437/interspeech.2018-1755.

[13] Chollet, F.: Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1800–1807, doi: 10.1109/cvpr.2017.195.

[14] Zhang, H.—McLoughlin, I.—Song, Y.: Robust Sound Event Recognition Using Convolutional Neural Networks. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 559–563, doi: 10.1109/icassp.2015.7178031.

[15] Livingstone, S. R.—Russo, F. A.: The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A Dynamic, Multimodal Set of Facial and Vocal Expressions in North American English. PLoS ONE, Vol. 13, 2018, No. 5, Art. No. e0196391, doi: 10.1371/journal.pone.0196391.

[16] Alonso-Martin, F.—Malfaz, M.—Sequeira, J.—Gorostiza, J. F.—Salichs, M. A.: A Multimodal Emotion Detection System During Human–Robot Interaction. Sensors, Vol. 13, 2013, No. 11, pp. 15549–15581, doi: 10.3390/s131115549.

[17] Breazeal, C.: Emotion and Sociable Humanoid Robots. International Journal of Human-Computer Studies, Vol. 59, 2003, No. 1-2, pp. 119–155, doi: 10.1016/s1071-5819(03)00018-1.

[18] Beer, J. M.—Fisk, A. D.—Rogers, W. A.: Toward a Framework for Levels of Robot Autonomy in Human–Robot Interaction. Journal of Human–Robot Interaction, Vol. 3, 2014, No. 2, pp. 74–99, doi: 10.5898/jhri.3.2.beer.

[19] Lee, K. W.—Kim, H. R.—Yoon, W. C.—Yoon, Y. S.—Kwon, D. S.: Designing a Human–Robot Interaction Framework for Home Service Robot. IEEE International Workshop on Robot and Human Interactive Communication (ROMAN 2005), 2005, pp. 286–293, doi: 10.1109/ROMAN.2005.1513793.

[20] Coronado, E.—Mastrogiovanni, F.—Venture, G.: Development of Intelligent Behaviors for Social Robots via User-Friendly and Modular Programming Tools. 2018 IEEE Workshop on Advanced Robotics and Its Social Impacts (ARSO), Genova, Italy, 2018, pp. 62–68, doi: 10.1109/arso.2018.8625839.

[21] Zguda, P.—Kołota, A.—Jarosz, M.—Sondej, F.—Izui, T.—Dziok, M.—Belowska, A.—Jędras, W.—Venture, G.—Śnieżynski, B.—Indurkhya, B.: On the Role of Trust in Child-Robot Interaction. 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), New Delhi, India, 2019, pp. 1–6, doi: 10.1109/ro-man46459.2019.8956400.

[22] Zguda, P.—Kołota, A.—Jarosz, M.—Sondej, F.—Izui, T.—Dziok, M.—Indurkhya, B.: "Why Don't You Have a Wife?!" Free Format Dialogue in CRI. 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Late Breaking Results Poster, IEEE, 2019.

[23] Edward, M.: NLP Augmentation. Available at: `https://github.com/makcedward/nlpaug`, 2020.

[24] Röning, J.—Holappa, J.—Kellokumpu, V.—Tikanmäki, A.—Pietikäinen, M.: Minotaurus: A System for Affective Human–Robot Interaction in Smart Environments. Cognitive Computation, Vol. 6, 2014, No. 4, pp. 940–953, doi: 10.1007/s12559-014-9285-9.

[25] Liu, Z.—Wu, M.—Cao, W.—Chen, L.—Xu, J.—Zhang, R.—Mao, J.: A Facial Expression Emotion Recognition Based Human–Robot Interaction System. IEEE/CAA Journal of Automatica Sinica, Vol. 4, 2017, No. 4, pp. 668–676, doi: 10.1109/jas.2017.7510622.

[26] Liu, Z. T.—Pan, F. F.—Wu, M.—Cao, W. H.—Chen, L. F.—Xu, J. P.—Zhang, R.—Zhou, M. T.: A Multimodal Emotional Communication Based Humans–Robots Interaction System. 2016 35th Chinese Control Conference (CCC), IEEE, 2016, pp. 6363–6368, doi: 10.1109/chicc.2016.7554357.

Mateusz Jarosz is Ph.D. student in the Institute of Computer Science at the AGH University of Science and Technology, Krakow, Poland. His research interests include human–robot interaction, gaze patterns and machine learning. He is currently working on an NCBR-supported research project in cooperation with Turkish scientists.

Piotr Nawrocki is Associate Professor in the Institute of Computer Science at the AGH University of Science and Technology, Krakow, Poland. His research interests include distributed systems, mobile systems, cloud computing, artificial intelligence and service-oriented architectures. He has participated in several EU research projects including MECCANO, 6WINIT and UniversAAL. He is a member of the Polish Information Processing Society (PTI).

Bartłomiej Śnieżyński received his Ph.D. degree in computer science in 2004 from the AGH University of Science and Technology, Krakow, Poland. In 2004, he worked as Postdoctoral Fellow under the supervision of Professor R. S. Michalski at the Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA. Currently, he is Associate Professor in the Institute of Computer Science at AGH. His research interests include machine learning, multi-agent systems and knowledge engineering. He is a member of the Polish Information Processing Society (PTI) and the Polish Artificial Intelligence Society (PSSI).

Bipin Indurkhya is Professor of cognitive science at the Jagiellonian University, Krakow, Poland. His main research interests are social robotics, usability engineering, affective computing and creativity. He received his Master's degree in electronics engineering from the Philips International Institute, Eindhoven (The Netherlands) in 1981, and his Ph.D. in computer science from the University of Massachusetts at Amherst in 1985. He has taught at various universities in the US, Japan, India, Germany and Poland; and has led national and international research projects with collaborations from companies like Xerox and Samsung.

Computing and Informatics, Vol. 40, 2021, 104–144, doi: 10.31577/cai_2021_1_104

MULTI-OBJECTIVE TASK SCHEDULING USING SMART MPI-BASED CLOUD RESOURCES

Mehran Mokhtari

Department of Computer, Sari Branch, Islamic Azad University, Sari, Iran
e-mail: mehrmokhtari@yahoo.com

Peyman Bayat*

Department of Computer, Rasht Branch, Islamic Azad University, Rasht, Iran
e-mail: bayat@iaurasht.ac.ir

Homayun Motameni

Department of Computer, Sari Branch, Islamic Azad University, Sari, Iran
e-mail: h_motameni@yahoo.com

Abstract. Task Scheduling and Resource Allocation (TSRA) is the key focus of cloud computing. This paper utilizes Smart Message Passing Interface based Approach (SMPIA) and the Roulette Wheel selection method in order to determine the best Alternative Virtual Machine (AVM). To do so, the Virtual MPI Bus (VMPIB) is employed for efficient communication among Virtual Machines (VMs) using SMPIA. In this matter, SMPIA is applied on different resource allocation and task scheduling strategies. MakeSpan (MS) was chosen as an optimization factor and solutions with minimum MS value as the best task mapping performance and reduced cloud consumption. The simulation is conducted using MATLAB. The analysis proves that applying SMPIA reduced the Total Execution Time (TET) of resource allocation, maximum MS time, and increase the Resource Utilization (RU), as compared to non-SMPIA for Greedy, Max-Min, Min-Min algorithms. It is observed that SMPIA can outperform non-SMPIA. The effect of SMPIA is more

* Corresponding author

obvious as change in the MS and the number of cloud workloads increase. Furthermore, regarding the TET and MS of the tasks, the SMPIA can significantly reduce the starvation problem as well as the lack of sufficient resources. In addition, this approach improves the system's performance more than the previous methods, what reflects effectiveness of the proposed approach concerning the Message Passing Interface (MPI) communication time in the network virtualization. The mentioned text mining work was prepared concurrently after practical evaluation.

Keywords: Cloud computing, SMPIA, TSRA, resource allocation scheduling, roulette wheel, text mining, AVM, starvation

1 INTRODUCTION

Cloud computing is well known as a model that aims at providing resources and services through a network. In this matter, the Task Scheduling and Resource Allocation (TSRA) present the key focus of cloud computing [1]. Message Passing Interface (MPI) is a standard communication protocol, which has become a legal standard for communication between processes and implements a parallel programming using the MPI [2, 3]. Typically, High-Performance Computing (HPC) applications employ the MPI communication [4, 18]. Here, it is worthwhile to mention that as the mapping tasks problem onto resources (workflow tasks scheduling) is known as NP-complete in the cloud computing, several scheduling algorithms have been developed to solve it [5, 13, 14, 15, 16, 29, 38]. The main objective of the workflow scheduling problem is to reduce Total Execution Time (TET) as well as to generate a balance between the resources consumption and the Quality of Service (QoS) [8, 9].

This paper proposes a Smart MPI Approach (SMPIA) to improve MPI communication time and the lack of sufficient resources, as well as to reduce the resource involvement level and starvation problem (large waiting time) of the tasks. Therefore, this paper addresses two issues:

1. whether it is possible to reduce latency of MPI communication time and starvation problem by reducing the average TET and completion time, and
2. whether it is possible to reduce the resource consumption time and involvement level while ensuring efficiency.

To solve the first problem, the SMPIA and the Roulette Wheel selection method are utilized to determine the probability of choosing Alternative Virtual Machines (AVMs). In addition, several AVMs are employed instead of one Virtual Machine (VM) in sub-networks inter-connected. To answer the second question, a suitable model is developed to calculate the Resource Utilization (RU) in order to decrease the level of resource involvement in the cloud and resource consumption time in

such a way that its inputs were MakeSpan (MS) parameters that improved RU and resource consumption time.

To accomplish this aforementioned aim, several experiments are implemented on different TSRA strategies using SMPIA in the Ministry of Communication and Information Technology of Iran (MCITI) dataset. After that, the performance metrics including MS, TET, and RU are measured. The obtained results of the experiment indicate that the SMPIA outperform standard algorithms including Min–Min algorithm [34] in terms of MS. Moreover, the SMPIA performs better than the algorithm developed in [4, 6, 20, 30, 35, 17, 31, 33, 36, 37], in terms of ET, MS and RU, respectively.

This paper aims to allocate the appropriate AVMs collection based on the minimum MS time and optimal mapping of current flow onto the selected AVM. Through choosing the appropriate AVMs and mapping the flow onto them in the shortest MS time in Virtual MPI Bus (VMPIB), this approach can improve MPI communication time, starvation problem and performance in the cloud computing. To do so, this approach encompasses three phases of calculation of resource-ranking, resource selection, and optimal task-resource mapping. In the following, the main contribution and motivation of this paper is described.

1.1 Contribution and Motivation

The main contributions of this paper are as follows:

1. This paper develops a novel method and technique for efficient communication between VMs in MPI-based cloud resources using VMPIB. The key idea is the phases of the resource-ranking, AVM selection, and mapping of the current flow onto the selected AVM. In this way, the probability of choosing an AVM for workflows was determined using the Roulette Wheel selection method.
2. This paper improves the resource consumption time and task scheduling in the cloud computing. In this regard, the parameters of load, capacity, and amount of computed load, Execution Speed (ES) and execution time of each flow on the VM are employed to calculate the minimum MS. Besides, the parameters CPU processing speed of each flow on the VM are considered to calculate the execution time and TET. It is worth noting that this approach is different from the previous conducted approaches because the effect of MPIA is more obvious as changes increase in the MS and the number of cloud workloads.
3. This paper enhances RU, reduction resource consumption, optimal task-resource mapping model in the cloud computing. To accomplish the aim, the memory capacity parameters, memory capacity used, total processing capacity, and processed capacity on the VMs, are exploited to calculate the RU. It is worthwhile to mention that this approach is different from the former reviewed approaches, because decreasing the RU level using heuristic (Max-Min, Min-Min) and Greedy algorithm indicates that mapping the flows onto the appropriate AVM is carried out properly. Analysis of the results demonstrates that the proposed approach

enhances performance in terms of MS up to 55.94 %, while it is up to 55.59 % in terms of the TET based on which RU and involvement level are enhanced up to 12.80 %. The need to conduct this research is to delay MPI communication and starvation problem in the VMPIB. It should be mentioned that utilizing SMPIA on a telecommunications transaction application increases its efficiency.

The motivation for this research is the implementation of text mining on the telecommunications transactions application in order to reduce MPI processing time and solving starvation problem of the tasks. The main novelty of this research is to implement text mining on the telecommunications transaction application in order to achieve proper processing time as well as to manage the proposed cloud system.

The remainder of this paper is organized as follows: the related works are presented in Section 2. A case study is described in Section 3. Process smart MPIA and job migration in the text mining is found in Section 4. Solving the resource allocation scheduling problem using SMPIA is provided in Section 5. Solving the starvation problem using SMPIA is performed in Section 6. Evaluation of SMPIA is given in Section 7. Discussions and analysis are in Section 8. Ultimately, conclusions and future research are presented in Section 9.

2 RELATED WORKS

In this section, some studies conducted on the scheduling and method of resource allocation in the cloud are presented for the optimal resource use.

The papers of [2, 4, 6, 7, 10, 11, 12, 19] confirmed that the implementation of MPI applications is appropriate on the cloud. In [17], the ranking of each task in Heterogeneous Earliest Finish Time (HEFT) algorithm was performed based on the average of task connections and cost of computing between the current task and its substitution. In [20], the Eager Map algorithm for solving mapping problems was proposed in cluster nodes and cores, which was based on a Greedy heuristic in order to match application communication patterns to hardware hierarchies. A novel dynamic task scheduling algorithm was developed in [26] based on an improved genetic algorithm. Then, some experimental results indicate that the proposed algorithm could effectively improve throughput of the cloud computing systems so that it could significantly reduce the execution time of task scheduling. In [27], a novel task-scheduling algorithm termed as Genetic Algorithm-based Customer-Conscious Resource Allocation and Task Scheduling (GACCRATS) is proposed for the heterogeneous multi-cloud environment in order to cope with the gap between frequently changing customer requirement and available infrastructure for the services. After that, the simulation results were compared with the existing scheduling algorithm. The aim was to task-resource mapping of the multi-cloud federation in order to achieve minimum MS time and maximum customer satisfaction. In [28], the problem of allocating Data Center (DC) resources is considered for the cloud enterprise customers who required the guaranteed services on demand. For the higher traffic situation, the heuristic approach was much more suitable, which was analyzed

and then the results are presented for up to 3,200 servers. The proposed heuristic was fast to solve large-scale problems where the Mixed-Integer Linear Programming (MILP) problem was difficult to solve. They developed a novel MILP model as well as alternately a heuristic that was solved in this framework at each review point. In other words, more frequency options for a server mean higher reduction in the energy consumption. According to findings of [28], there are several future directions to address, which do not allow partial fulfillment of a request if there is a lack of sufficient resources to consider fully a request. Furthermore, they planned to add performance evaluation on the loads to a DC based on its geographical distance from different Virtual Networks (VNs). Moreover, they planned to explore different allocation policies so that the service performance was comparable for different VN customer groups. In [32], Foundations of Machine Learning (FOML) algorithm is compared to Min–Min, Max–Min, Sufferage and Enhancement HEFT (E-HEFT) algorithm. In [33], the simulation results show that the Segmented Min-Min (SMM) algorithm with the high number of tasks and machines is the best. In [34], the obtained results indicated that the Max-Min Scheduling Improved Algorithm (MM-SIA) had the lowest completion time of all VMs, as compared to three algorithms such as Max-Min, Min-Min, and Round Robin. In [35], an approach presented called Optimized Process Placement (OPP) found the best placement scheme comparing to all collective communications on all message sizes.

2.1 Comparison the SMPIA with Benchmarks, Algorithms and Methods

In [16], compared to the previous methods, the heuristic method could improve the task response time and resource allocation up to 50 %. The proposed heuristic approach performed task scheduling and resource allocation efficiently with high utility. In this way, the maximum RU was achieved with computing resources such as CPU, memory and bandwidth. Existing systems such as [16] considered three resources of CPU, memory, and bandwith in evaluating their performance. In our proposed system, the parameters such as (CPU, memory, bandwith, storage), capacity (memory, CPU), load (VMs, flow), number (VMs, flows), ES (flow, CPU), DC ID, server ID, CPU number, CPU Freq (HZ), VM ID, flow ID were considered as input parameters to calculate VM capacity. Also, the experimental results show that the SMPIA outperforms three standard algorithms, i.e., Greedy, Min–Min and Max–Min algorithm and improved these algorithms in terms of TET, MS, and RU metrics. The obtained result indicated the lower of the starvation problem between tasks and resources, the lower the level of resource involvement and resource consumption in MPI communications. Besides, the related performance parameters in SMPIA were calculated based on the mean values, as obtained after 50 times of the program execution. This paper conforms results [1, 8, 9, 10, 14, 16, 22, 23, 24, 25, 30, 31, 36] avoiding Service Level Agreements (SLA) violation and QoS dropping.

In this paper, the SMPIA could improve the resource allocation time and completion time up to 55.80% and 55.94 %, respectively, which caused reducing the resource consumption and resource involvement levels up to 11.80 %. At the mean-

time, the RU parameter confirms the rate of resource consumption and the extent of the involvement of hardware resources in the cloud to percentages. In this paper, both high and low percentages do not indicate its high or low quality. Table 1 reports the details of comparison performance parameters the SMPIA with benchmarks, algorithms and methods for optimization.

3 CASE STUDY

In this study, the general model of the telecommunication cloud system was designed. In this model, as illustrated in Figure 1, the VMPIB and the cloud management center were considered. To define the concept of the VMPIB, we considered a topology associated with the connected graph $G = (D, V)$, where $D = \{DC_1, DC_2, DC_3, \ldots, DC_d\}$ and $V = \{VM_1, VM_2, VM_3, \ldots, VM_m\}$. In this matter, we bring the following assumptions to investigate which resource can be allocated to the flow. The function $\varphi : V \to D$ was considered to control the dependencies of flows and tasks of all flows including the set $F = \{F_1, F_2, F_3, \ldots, F_f\}$. In addition, the function $\theta : F \to V$ and $T = \{T_1, T_2, T_3, \ldots, T_t\}$ is regarded as well. Here, D is the total number of DC; V refers to the total number of VMs; T denotes the set of t transaction, it was defined as the total of transactions. F means the total of the flows depending on each other. θ is a function to determine which flow could be executed by VM in Virtuall MPI Bus. In addition, φ is function, in which VM is assigned to each DC in the Virtuall MPI Bus. To do so, we bring the following assumptions:

Definition 1 (Cloud management center)**.** The cloud management center contains the cloud manager (Administrator), the workflow progress manager, and the initial scheduler. It performs the schedule work, schedule workflows, and resources, and then sets the initial values. The job scheduler schedules the workflows and resources, and then adjusts the initial values (Figure 1).

Definition 2 (Cloud provider)**.** The cloud provider was composed from DC and servers, in which each DC had a number of servers, each of which had several VMs in the VMPIB (Figure 1).

Definition 3 (Cloud VMs)**.** A set of VMs, they receive and process the superclouds as the resources. In this work, each VM has an ID and capacity. We have a list of VMs and IDs for the VMs. The VMs in a VMPIB include two-way communication with each other and a server. It should be noted that each VM could process only one type of flow in large numbers. Each DC, server, and VM has an ID in the cloud.

Definition 4 (Job)**.** A transaction involves a number of jobs. In this way, the flows in a VMPIB must pass through a number of jobs to perform a transaction. Each input and output flow was exhibited as an arrow, whereas each job was depicted as a red, yellow and blue circle in Figures 2 and 3. Similar to array cells, a number of jobs generate a task.

Problem	Technique	Platform	Technology	Reference	Metrics	Improvement
Performance	PingPong TCP	CPU, RAM, VM	MPI	[2]	Latency	30 %
Performance	PingPong Open-MX	CPU, RAM, VM	MPI	[2]	Latency	36 %
Performance	Alltoall	CPU, RAM, VM	MPI	[2]	Latency	35 %
Performance	Alltoallv	CPU, RAM, VM	MPI	[2]	Latency	35 %
Performance	HEAT	CPU, RAM, VM	MPI	[2]	Elapse time	30 %
Performance	HEAT	CPU, RAM, VM	MPI	[2]	Latency	30 %
Performance	Allgather	CPU, RAM, VM	MPI	[2]	Latency	33 %
Performance	Allgatherv	CPU, RAM, VM	MPI	[2]	Latency	31 %
Performance	Reduce	CPU, RAM, VM	MPI	[2]	Latency	35 %
Performance	Reduce-Scatter	CPU, RAM, VM	MPI	[2]	Latency	32 %
Performance	Exchange	CPU, RAM, VM	MPI	[2]	Latency	45 %
Performance	Sendrecv	CPU, RAM, VM	MPI	[2]	Latency	35 %
Performance	LAMPICS, MPAR	CPU, RAM, VM	MPI	[3]	Latency	26.5 %
Performance	MCM	CPU	MPI	[4]	ET	30 %
Performance	MCM_CG Class B	CPU	MPI	[4]	ET	16.2 %
Performance	MCM_CG Class C	CPU	MPI	[4]	ET	12.7 %
Performance	NPA	CPU, RAM, Bandwidth	Simulations	[6]	ET	28.3 %
Performance	NPA	CPU, RAM, Bandwidth	MPICH2 on AEC2	[6]	ET	25.4 %
Performance	N-body	CPU, RAM, Bandwidth	MPI	[6]	Performance	41.6 %
Performance	CG	CPU, RAM, Bandwidth	MPI	[6]	Performance	14.3 %
Performance	CMPI	CPU, RAM, Bandwidth	MPI	[6]	TET	14.3 %
Performance	CMPI	CPU, RAM, Bandwidth	MPI	[6]	NCT	33.2 %
Performance	MCM, NAS-CG CLASS B	CPU, RAM, VM	OpenMPI, MPI	[9]	Latency	29.26 %
Performance	NPB	CPU, RAM, VM	MPI	[12]	ET	20 %
TSRA	Heuristic, BATS+BAR	CPU, RAM, Bandwidth, VM	CCS	[16]	Response time	50 %

Continue in next pages

Table 1. Comparison performance parameters of SMPIA with benchmarks, algorithms and methods to optimization

TMLB	LB*	VM	CCS	[17]	MS	15 %
TMLB	Eager Map HPC	CPU, RAM	OpenMPI, MPI	[20]	ET	10.3 %
TSRA	IGATS	CPU, RAM, Bandwidth, Server, DC	CCS	[26]	ET, Response time	Effective improvement
TSRA	GA, TLBO, GACCRATS, COTS	CPU, RAM, DC, Server	CCS	[27]	MS, customer satisfaction	MCS, Minimize MS
Resource allocation	Heuristic approach, MILP	CPU, Server	CPU, Bandwidth	[28]	Energy consumption	Maximum Reduce energy consumption
Performance, load balancing	Min-Max	CPU	CCS	[30]	TET	9 %
Performance, load balancing	Min-Max	CPU	CCS	[30]	TET	7 %
Performance, load balancing	Min-Max	CPU	CCS	[30]	ART	9 %
Performance, Task scheduling	PA-MMSIA	Tasks/Virtual resources	CCS	[31]	ACT	20 %
Performance, Task scheduling	PA-LBIMM	Tasks/Virtual resources	CCS	[31]	MS	13.2 %
Performance, Task scheduling	PA-LBIMM	Tasks/Virtual resources	CCS	[31]	ARU	1.16 %
Load balancing, Energy consumption	FOML	CPU, RAM, VM	Cloud computing,	[32]	ACT	16 %
Performance, Scheduling	SMM	VM	CCS	[33]	MS	6.8 %
Load balancing	LBIMM	CPU, RAM, VM	CCS	[34]	MS	12.5 %
Load balancing	Min-Min	CPU, RAM, VM	CCS	[34]	MS	19.625 %
Performance	OPP_cyclic	CPU, RAM	MPI	[35]	ET	53.6 %

Continue in next page

Performance	OPP_block	CPU, RAM	MPI	[35]	ET	20.4 %
WSRA	OWM	CPU, VM	Grid environments	[36, 37]	Average MS	26 %
Task scheduling	MCC, MEMAX, CMMN	CPU, RAM	CCS	[38]	MS, ACU	MS, cloud utilization
TSRA	Greedy-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA	This paper	TET	51.10 %
TSRA	Greedy-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA	This paper	MS	38.84 %
TSRA	Greedy-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA,	This paper	RU	12.28 %
TSRA	Max-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA,	This paper	TET	49.91 %
TSRA	Max-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA,	This paper	MS	55.94 %
TSRA	Max-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA,	This paper	RU	11.80 %
TSRA	Min-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA	This paper	TET	55.80 %
TSRA	Min-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA	This paper	MS	53.04 %
TSRA	Min-Min-SMPIA	CPU, RAM, VM, Bandwidth, DC, Server	CCS, SMPIA	This paper	RU	11.86 %

Definition 5 (Job migration)**.** Job transfer for running from one AVM to another one (Table 3, Figures 4 and 5).

Definition 6 (Task)**.** In this paper, the blue circles form a task in Figures 2 and 3. For example, in Figure 3, flow number 8 is for a task called a tender, which is divided into general deal (i.e. flow 11) and limited deal (i.e. flow 12).

Definition 7 (Transactions)**.** Transactions (including deals and tenders) are categorized into two categories of small and big. A transaction involves a number of jobs. To perform a transaction, the flows in a VMPIB must go through a number of jobs. Each input and output flow was shown as an arrow, in which each job is depicted as a circle. Besides, the big and small transactions (i.e. deals and tenders) were shown as small and big workflows. The related details are illustrated in Figures 2 and 3. Small and big transactions were composed as 28 and 14 jobs, respectively.

Definition 8 (MPI table)**.** MPI table is the same as Virtual MPI page table. Were these pages are placed in physical memory is determined by the page table. An address obtained with the ampersand operator in program language is not a physical address, but a virtual address. Its initial values are set using the scheduling algorithms at the start of the algorithm through the specified data tables, which are updated during program execution and its values change (Figure 8).

Definition 9 (Distributed MPI table)**.** The type of the network is MPI. Because the servers process in parallel in the MPI network that are distributed in the network. The memory allocated to MPI tables is distributed so that the pages are stored in the buffer of the physical memory.

Definition 10 (MPI's flows)**.** Each transaction (deals and tenders) that enters the cloud system contains a number of flows. The transaction flows (deals and tenders) that are exchanged (sending or receiving) between the VMs for processing are called MPI's flows in the VMPIB. In this regard, some application developers encounter the transmissibility problem in communication networks, which led to the definition of a standard for messaging, so-called the MPI. MPI is a standard interface that is independent from hardware, platform, and message-based for parallel applications. Although the MPI sub-layer can be a proprietary protocol, it does not see the protocols of some applications, except MPI.

Thus, MPI is a middle ware and a simple interface. In MPI, it is assumed that communication takes place between a specific group of processes, in which each group contains an ID. On the other hand, each process contains a local ID in each group. The ID group and ID process of either source or destination identify a message uniquely which is utilized instead of the transfer layer address. In this paper, we have a list of MPI flows and their IDs. Each flow contains the amount of load, ES, and execution time on the VM. As illustrated in Figure 2, the MPI's flows of referrals to the supplier dataset (for providing the qualified supplier) and the review

of technical non-approval (for receiving new requests) are designed as circular while other flows are linearly defined. Note that the small and big transactions were 28 and 14 jobs composed, respectively.

Definition 11 (Input parameters)**.** It includes flow (ID, load, speed, number), CPU (ID, number, speed Freq (HZ), DC (ID), server (ID)), bandwidth (Mb/s), VM (ID, load, size, number) and flow (ID, load, number) (Figure 4).

Definition 12 (Output parameters)**.** It includes capacity (bit/Byte) metrics of VMs, current flow and VM with minimum Computational Cost (CC), next flow and VM with low load variance (Figure 4).

Definition 13 (Execution time)**.** Execution time can be modeled as follows:

1. The amount of computational load of the current flow (L_{CF});
2. The speed of CPU execution (ES).

It is the same as TET and performance metric parameter.

Definition 14 (Completion time)**.** The completion time can be modeled as follows:

1. Time spent by execution flow_k of transaction_j on VM_i namely T_{ijk}.
2. The parameter determining the CC of flow_k on VM_i namely CC_i.
3. The parameter ES_i determines the ES of CPU on VM_i.

Definition 15 (Efficiency)**.** Efficiency can be modeled as RU and performance metric.

Definition 16 (Task scheduling)**.** Allocating the flow to the selected VM in the shortest time. Each task is a transaction that consists of a number of jobs. For example, the task scheduling for a tender in Figure 3 means two general and limited tenders, which each run with the VM in the shortest time, in which each workflow consists of a number of tasks while each task consists of a number of jobs and flows.

Definition 17 (Resource allocation)**.** The best AVM is allocated to the current flow.

Definition 18 (Workflow)**.** For example, the tender task of three jobs and two flows was illustrated in Figure 3. Each workflow is a transaction. The cloud input includes two workflows, big and small deals. Here, in order to employ each workflow to be considered as a service in the process of purchasing deals and tenders, a comprehensive and centralized telecommunication supply chain system is introduced. The aims of designing this application are as follows: cost reduction, decreasing administrative bureaucracy, time productivity, accuracy in performing works, integration and focus on the field of supply, reporting system and preparing the management dashboard, the mechanized management of stakeholders and

suppliers, and improving communication and coordination process. This practical application includes all requirements of the supply chain such as the process of ordering and purchasing of inquiries and tenders, service contracts, and communication with the personnel system, etc. To design this practical application, Key Performance Indicators (KPIs) were utilized to decrease the amount of stagnant items in warehouses, to decrease the cycle time of work processes, percentage of centralized purchases, and identifying the products required by the regions to save on the purchases. Many distribution systems and applications are developed on the simple message model provided by the transmission layer. In this simulation, 31 telecommunication regions distributed in 31 provinces were chosen, each of which participated in the tender process of purchasing telecommunication equipment. MPI in the cloud was utilized to improve the tender processing time, to reduce the process transfer delays, to allocate resources to customers at the suitable time, to improve execution time and completion time. The MPI communications are among the VMs, servers, and DCs. The objective of MPI communications is to get the cloud out of the centralized management. This method, in addition to reducing execution time, completion time, the level of engagement, and resource consumption, has also improved the productivity. In this matter, supplying the resources by allocating resources at the right time has also increased productivity.

Definition 19 (WaaS)**.** Workflow as a Service (WaaS) is an emerging concept that offers workflow execution as a service to the scientific community. Note that WaaS is categorized as either Platform as a Service (PaaS) or Software as a Service (SaaS) on the cloud stack service model. With the emergence of WaaS in the cloud, it is more challenging to predict workflow scheduling and estimate the runtime of tasks. In this way, processing a large volume of data needs to predict real-time changes for the resource performance [21].

Definition 20. The comprehensive design and implementation of a comprehensive and centralized supply chain. In order to decrease the costs and administrative bureaucracy as well as to obtain the productivity on time, punctuality, integration and focus in the field of procurement, reporting system and management dashboard, mechanized management of stakeholders and suppliers, improving the communication process and coordination, an application supply chain (including the comprehensive design and implementation of a comprehensive and centralized supply chain) was introduced. This application encompasses all the requirements of the supply chain such as the process of ordering and purchasing inquiries and tenders, service contracts, and communication with the personnel system, etc. Some KPIs were employed to design this application in order to decrease the amount of stagnant items in the warehouses, the time of the work process cycle, the percentage of centralized purchases, as well as to identify the goods required by the regions to save on purchases.

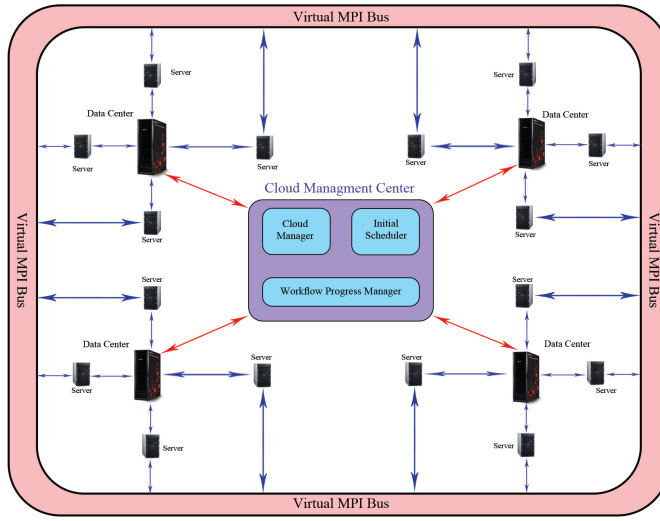


Figure 1. Model of cloud computing based on VMPIB

3.1 Problem Description

In the most basic cloud service model, an infrastructure is considered as a service, in which computing resources can be provided as VMs [2]. The main problem in our work is to allocate suitable AVMs and then optimally map the flows onto them in the minimum processing time in order to

1. simultaneously optimize the performance parameters;
2. to solve the TSRA problem in cloud computing using SMPIA;
3. to address the starvation problem of the tasks (large waiting times for small and big jobs) and the lake of sufficient resources.

To this end, the non-SMPIA and the SMPIA are developed. In the non-SMPIA (Greedy, Max-Min, Min-Min), first, the current flow was chosen based on these aforementioned algorithms from the expected flows. Afterwards, the selected flow was sent to the selected resource of these algorithms. Then, the SMPIA was applied to each of these algorithms in order to optimize TET, completion time, and RU. After optimization, if the VM could not perform the current flow (e.g. the queue was full, the system crashed, the system was disconnected, etc.), the SMPIA was applied. The SMPIA was implemented in three phases: calculation of the AVM ranking, selection of the AVM, and flow mapping onto the AVM. Regarding the MPI management of the VMs, the rank of AVMs was calculated based on the number of its connections with other AVMs based on MS's calculation.

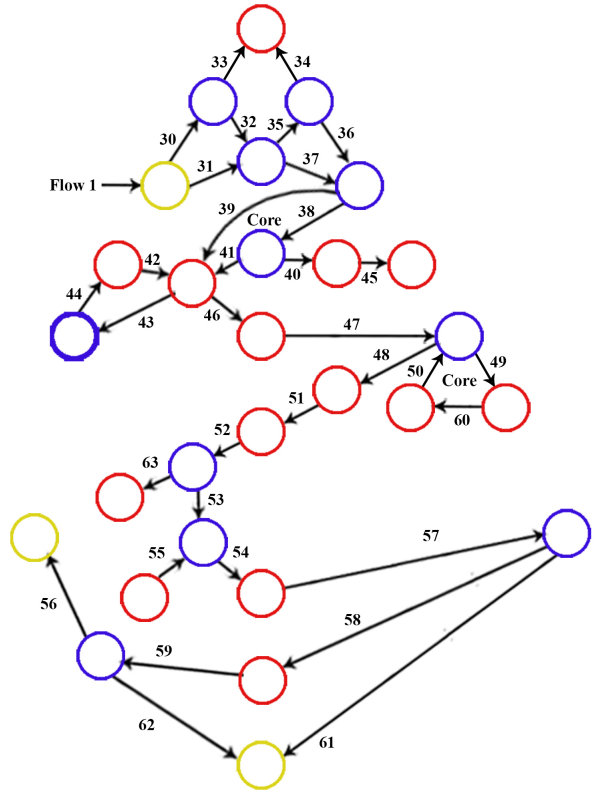


Figure 2. WaaS of small transaction

To calculate the performance parameters, Equations (1), (2), (3), (4) were employed for TET, MS, RU, and average utilization, respectively. On the other hand, the programs were executed at least 50 times in a system to compute the average of the parameters, with the identical specifications. At the end, the mean values were recorded. Regarding the above-mentioned information, all the equations are quickly solved as well.

The execution time is equal to TET using Equation (1):

$$\text{Problem ET} = \frac{\text{The value of calculation load}}{\text{CPU execution speed}} = \frac{L_{cpu}}{ES_{cpu}}. \tag{1}$$

The MS is equal to maximum MS of all tasks using Equation (2):

$$\text{Problem MS} = \max\left(\sum_{i=1}^{m}\sum_{j=1}^{t}\sum_{k=1}^{f}\left(\frac{(CC)_i}{(ES)_i}\right) \times T_{ijk}\right). \tag{2}$$

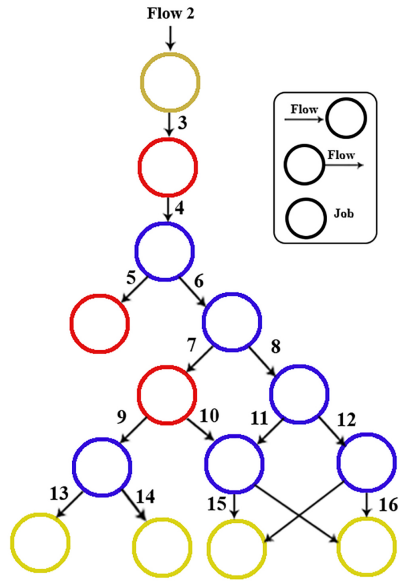


Figure 3. WaaS of big transaction

T_{ijk} is time spent by execution flow_k of transaction_j on VM_i, otherwise, it is zero [22]. CC_i is the parameter determining the CC of flow_k on VM_i. ES_i is the parameter determining the ES of flow_k on VM_i.

The RU is calculated through the following Equation (3):

$$\text{Problem RU}_i = \frac{CP_i}{TPC_i} + \frac{MC_i - MCU_i}{MC_i}. \tag{3}$$

CP_i is the capacity processed in VM_i, TPC_i refers to the total processing capacity of VM_i, MC_i denotes the memory capacity of VM_i, MCU_i means the memory capacity used in VM_i

$$\text{Ave RU} = \frac{\sum_{i=1}^{m} RU_i}{m} * 100. \tag{4}$$

We considered the following hypotheses using Equations (5), (6), (7), (8), (9), (10), (11):

$$L(VM_i^t) = N(T,t)/S(VM_i^t). \tag{5}$$

$L(VM_i^t)$ means the load on a VM_i can be calculated as the number of tasks at the time t, $N(T,t)$ is the number of tasks at the time t in the service queue of VM_i, $S(VM_i^t)$ denotes the service rate of VM_i at the time t.

$$C(VM_i^t) = Pe_{numi} \times Pe_{mipsi} + VM_{bwi}. \tag{6}$$

Pe_{numi} is the number of processors in VM_i, Pe_{mipsi} is million instructions per second of all the processors in VM_i, VM_{bwi} is the capability of communicational bandwidth of VM_i.

$$(CC)_i = L(VM_i^t)/C(VM_i^t). \tag{7}$$

$C(VM_i^t)$ is the capacity on a VM_i at the time t.

$$L = \sum_{i=1}^{m} L(VM_i^t). \tag{8}$$

L is the loads exerted on all VMs in a DC.

$$C = \sum_{i=1}^{m} C_i. \tag{9}$$

C is the capacity of all VMs in a DC.

$$CC = L/C. \tag{10}$$

$$U = \sum_{j=1}^{n} \text{ and } C_{\max} = \max\{C_j,\ j = 1, \ldots, n\}. \tag{11}$$

U denotes maximum MS is defined as the maximum total time of completion of all workflows.

4 PROCESS SMART MPIA AND JOB MIGRATION IN THE TEXT MINING

The general process of doing work is described in such a way that after designing the general cloud model, the initial values of input parameters are valued by the job scheduler. After that, there is the waiting step for a request (flow) to enter. An all-broadcast query message is sent to all VMs that can respond to the current flow. The selection of the best VM is conducted based on the algorithms. Afterwards, the current flow is sent to the VMs using the function "AddJobVM" to execute so that its status is reported to the cloud management. The function "RetRelatedVMs" takes the ID of current flow ready on the queue and returns the VM that running the flow. In this way, the VM executes the current flow and stores the status of modes. If the query is not done, the next job will be executed. An inquiry message is sent to those flows which do the next job. Then, the selection of the best flow is carried out. At the end, the flow is sent to the selected VM. In this way, it is determined by which VM each of the different flows is executed using the algorithms. To updating the time step in this text mining work, the MPI run time of transactions in the dataset is sorted using the function "SortTransTime". The time function "AddMinutes" receives a time to generate time steps and then adds it to the current time and displays the new time in the output. The current and new

time are compared using another time function "CompareDateTime", and then the another new time is generated. In this condition, if the current time is longer than the new transaction time, then the new request is entered into the ready list. The flow executing method is that one flow is given for execution and the then next flow is received (Figure 4).

Any VM that wants to accept a flow to run and requires the cooperation with other VMs to run, it can communicate with other VMs using MPI-defined communications and send them the message whether they want to run the flow or not. As such, each cooperation VM that accepts the execution of the flow, the flow is sent to it. After running the flow by the cooperation VM, the cloud manager is informed that the flow has been completed. This is a voluntary choice of VMs to run the flows. To describe MPI smartly, each VM contains a list (or Table) of other VMs. Over time, each VM prepares a list of its neighbors based on the number of connections made to other VMs. In other words, this list contains those VMs that have more connection and send more workflows, which can help to run flows in the future. The VM that has received the flow while cannot run the flow for any reason, by checking the list of cooperators VM, it can select a cooperation VM that contains the relevant conditions to accept the flow so that it sends the flow with it. The cooperation VMs are called as AVMs in SMPIA (Figure 4).

It should be noted that if the VM did not work (e.g. the queue was full, the system crashed, the system was disconnected, etc.), in which it could not execute the current flow; as a result, the SMPIA would be applied to each of the algorithms. After updating the time step (by the time function), the list of transactions is checked. If the current time is greater than the transaction time as well as if the new transactions are entered into the list, one of them (either big or small) is chosen to be executed. On the other hand, if the small transaction is chosen, the first flow in the program is equal to 1 whereas if the big transaction is selected, the first flow is equal to 2. After transferring the flows of the transactions to the selected VMs, the SMPIA is applied and an appropriate AVM is selected based on the proposed approach. At the meantime, if the AVM is appropriate ($MS_{\min}$), the transaction is processed, and then the obtained results are saved and the stop condition is rechecked (were all transactions carried out?). Otherwise, the job migration is carried out (i.e. transferring the job from one AVM to another one) and the proposed approach is again applied to select another appropriate AVM (Figure 4).

The nfs-kernel-server and MPICH-3.0.4 packages are installed to implement MPI, due to a special feature in the master system. After that, Htop software is installed to monitor those processes running in parallel in MPI. Then, in the Hosts file, the master system is defined for IP systems. The copy of the program file was compiled in the "Mirror" folder using the MPI compiler. At the end, the program file is copied to the mirror folder and compiled it using the MPI compiler. Then, the program compiled by MPICH was executed using the following command:

```
/nfsshare$ mpirun –f hosts –n number. /MPI_sample.
```

It should be mentioned that instead of *number*, the number of processors desired to be involved with the program could be entered. Moreover, the program names are entered instead of *MPI_sample*. After that, the program run well on all systems and the performance of the processors was observed on each of the Slave computers with the help of *htop*. Ultimately, the user code was copied in the AVM buffer by MPI, and then it was prepared for parallel execution. To better demonstrate the above procedure, Figure 4 illustrates the framework of process SMPIA and job migration in text mining work. Here, Figure 5 illustrates the relationship between VMs and AVMs in VMPIB.

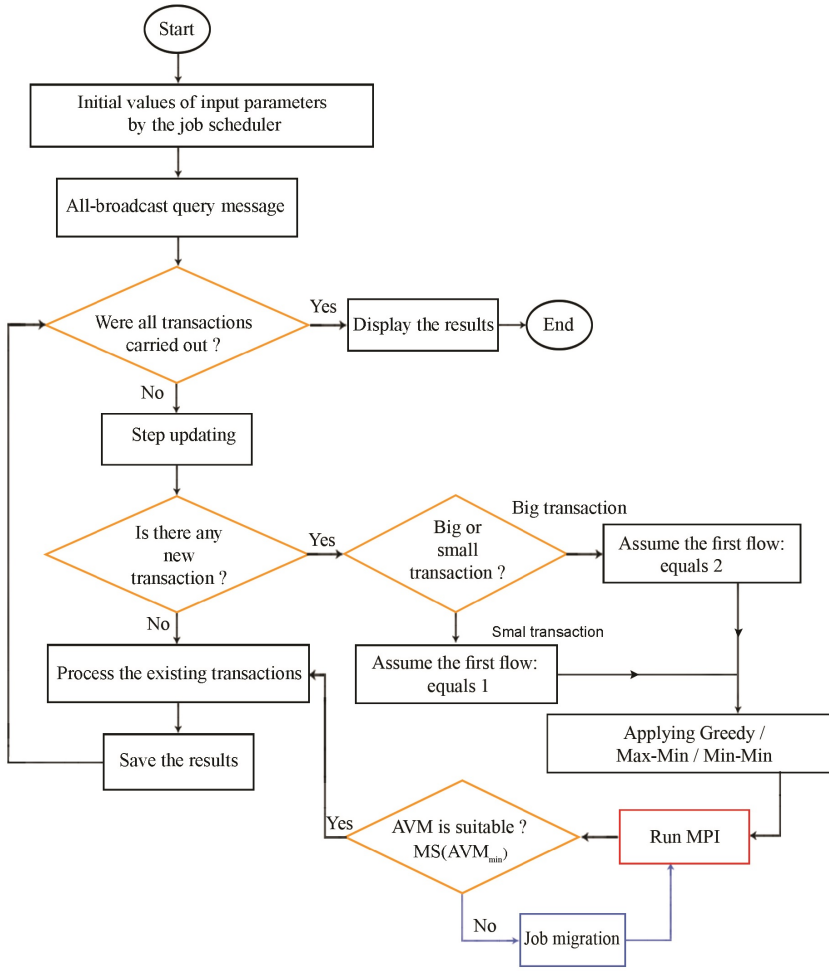


Figure 4. The framework of our process placement SMPIA and job migration

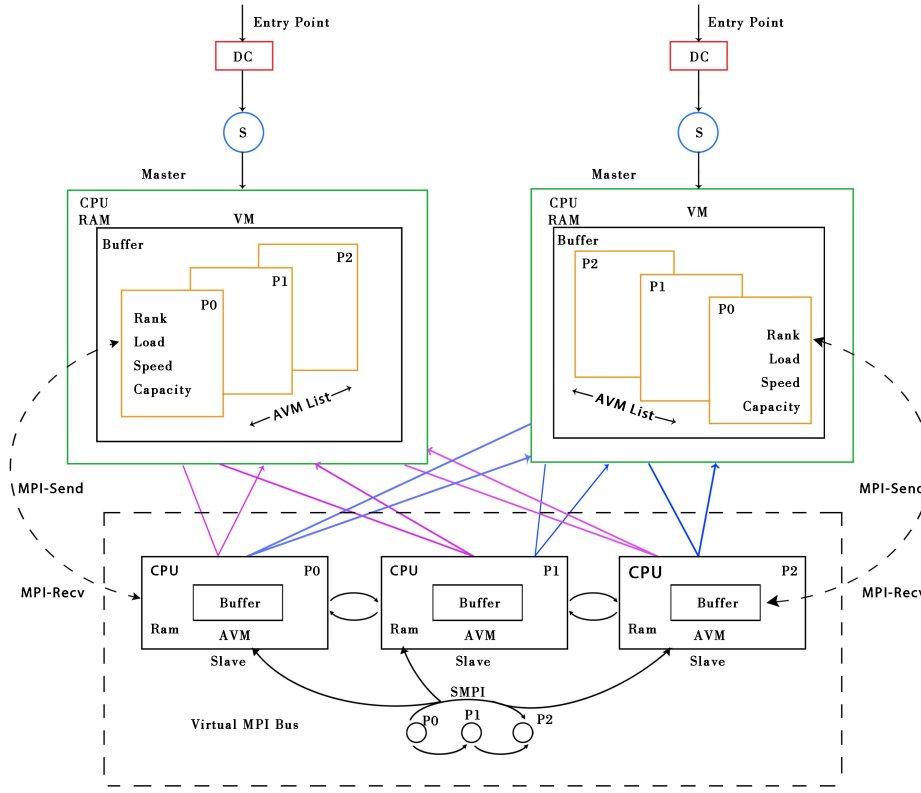


Figure 5. The relationship between VMs and AVMs in VMPIB

The SMPIA was performed in the three phases to pick up the current MPI's flows from the current VM and to transfer it to the AVM at VMPIB using the following phases: AVM rank computation phase, AVM rank-based selection phase, and MPI's flows mapping phase.

4.1 AVM Rank Computation Phase

4.1.1 Inactive MPI Process

In this case, all VMs were ranked based on the number of their connections with other VMs in the VMPIB. Note that any VM had a distributed MPI table to compute the ranking of VMs and manage MPI table. After that, the adjacent VMs were sorted out by managing the VM table based on the number of connections with other VMs in the VMPIB.

4.1.2 Active MPI Process

As discussed before, the proposed method was utilized to optimize the task scheduling and resource allocation by the MPIA (active MPI) and the non-MPIA (inactive MPI). In this way, if the number of transaction is greater than zero, a value is set for the method variable as follow: with "method = 1" the Greedy algorithm or "method = 2" the Max-Min algorithm or "method = 3" the Min-Min algorithm is determined. Note that MPI contains two processes; one is defined as Inactive MPI process due to it is executed prior the application execution; and the other is defined as Active process due to it is executed parallel user's application. Although the MPI was available in both approaches, it was smart in the proposed approach, because, with "EnableMPI = 1" the MPI is Active and "EnableMPI = 0" the MPI is Inactive. The value 1 refers to Enable MPI while the value 0 is used to Desable MPI. This paper takes advantage the Greedy, and fixed heuristic algorithms such as Max-Min, Min-Min [14, 17, 23] to find the optimal solution. These algorithms were implemented in the cloud platform on the datasets of MCITI.

In this case, all AVMs were ranked based on the number of connection with other AVM before they reach to the MPI table. The MPI table will be taken into account for VM configuration in order to choose the best AVM for current MPI's flows as well as to avoid congested of the MPI's flows. The probability of choosing an AVM for MPI's flows was determined using the Roulette Wheel selection method. According to this method, the probability of selecting AVM is equal to *the ratio of AVM rank i to total ranking of all AVMs*. The probability of selecting any $AVM_i(P_i)$ can be computed using Equation (12). Obviously, the probability of selecting an AVM with higher rank would be higher. In this matter, the MPI is defined as the communication of the VM with other AVMs in the distributed network system. In other words, the AVMs that were in touch along with most of them as well as those received more flows, were then introduced to run the next MPI's flow on the VM list. In each VM, the list of AVMs was provided in a table, based on which the minimum rank of each AVM was equal to 1. P_i or Run Times (RTs) of AVMs was calculated according to Pseudo-Code derived in Table 2.

$$P_i = \frac{\text{The ratio of AVM rank } i}{\text{Total ranking of all AVMs}} = \frac{f_i}{\sum_{j=1}^{NC} f_j}. \tag{12}$$

The Pseudo-Code of MPI algorithm is derived in Table 3.

Besides, the rank of each AVM varies based on MS's calculation. For each rank that MPI gives to the VM, $(MPI - VM_{rank})$, the rank for AVMs (AVM_{rank}) is periodically obtained according to the MS's change. As such, the pseudo-code of Send and Receive in MPI communications is provided in Table 4 for the user codes in the buffer.

```
(01)  Start
(02)  Read AVMs                          / * AVMs denotes AVMs in a data center */
        /* resource list contains AV[m] ← {AVM_1, AVM_2, AVM_3, ..., AVM_m} */
(03)  AVMsL ← Length(AVMs)
      /* AVMsL denotes the length of AVMs or the amount of current workload on
      AVMs */
(04)    For i ← 1 to AVMsL
(05)      Index ← R_AVMs(i)
                  /* Index denotes array index, R_AVMs denotes Related AVMs */
(06)      A_1 ← L_flow
                                        /* L_flow denotes MPI's flow load */
(07)      B_1 ← L_AVMs(i, Index)
                          /* L_VMs total computational loads on all of the AVMs */
(08)      C_1 ← S_AVM(Index, 4)
                                  /*S_AVM denotes size (capacity) of AVM */
(09)      D_1 ← FID
           /* FID denotes MPI's flow id, flow list contains F[f] ← {F_1, F_2, ..., F_f} */
(10)      E_1 ← ES(D_1, Index)
            /* ES denotes execution speed of the D_1 MPI's flow on AVM index */
(11)      TT ← ((A_1 + B_1)/C_1)/E_1          /* TT denotes temp time */
(12)      RT ← TT                               /* RT denotes run time */
(13)      Display "RT"
(14)    End for
(15)  End
```

Table 2. Pseudo-code for run times of AVMs

4.2 AVM Rank-Based Selection Phase

In the second phase, the list of AVMs would be checked by the MPI management and the AVM via the highest priority was selected. In this approach, the effect of selecting an AVM on the number of next flows was assessed, which would lead to an increase in their ranks. In this phase, the AVM's MS was computed as well. By comparing the obtained results of the implemented SMPIA and the applied algorithms, the effect of changing the selection of an AVM was precisely explored on both decreasing and increasing the time of MS. The best AVM was selected based on Pseudo-Code in Table 5. In Table 6, SMPIA is implemented for AVMs.

The third phase consists of SMPI functions that are responsible to allocate MPI's flows to the selected AVMs using the proposed method.

4.3 MPI's Flows Mapping Phase

In phase III, the flows were mapped onto the AVMs using these algorithms as well as the defined functions. According to the calculated MS, the flow from the VMs

(01) Start
(02) Upload MPI_{AVMs}(Table)
/* MPI_{AVMs}(Table) denotes the MPI table of AVMs */
(03) Based on the rank of each AVM, the following tasks are done, respectively.
(04) Inquire L_{AVM}
(05) Inquire ES_{AVM}
/* ES_{AVM} denotes the execution speed of current MPI's flows on the AVM */
(06) Inquire T_{CF}
/* T_{CF} denotes the execution time of current flow on the AVMs */
(07) Inquire C_{AVM} /* C_{AVM} denotes the capacity of the AVM*/
(08) Inquire C_{VM}
(09) Calculate MS_{AVM}
/* MS_{AVM} denotes the MS of the AVM for the current MPI's flow */
(10) Calculate MS_{VM}
/* MS_{VM} denotes the MS of the VM for the current MPI's flow */
(11) If $MS_{AVM} < MS_{VM}$
/* This is a scientific contribution of the paper */
(12) $AVM \leftarrow CF$ /* CF denotes current flow */
(13) $R_{AVM} \leftarrow R_{AVM} + 1$ /* R_{AVM} denotes the rank of AVM */
(14) Go to steep 19
(15) Else
(16) $R_{AVM} \leftarrow R_{AVM} - 1$
(17) Go to steep 04 /* Job migration */
(18) End if
(19) End

Table 3. Pseudo-code of MPI algorithm

via higher MS would be transferred to the AVMs via lower MS in order to reduce the mapping time (TET and completion time), resource involvement, and resource consumption. After that, the SMPIA was applied onto these algorithms. In this phase, the current flow of the current VM was removed and then mapped onto the selected AVM using the combination of the mentioned algorithms with the SMPIA. After the run of the flow accepted by the AVM, more flows were accepted by the AVM to run. In the main program, an AVM was selected to the each of input MPI's flows. Then, an ID for each of AVM was determined. Moreover, a function namely "Size" received the input MPI's flows and then calculated theirs number and capacity. The process of selecting the AVM and assigning the MPI's flow to the selected AVM was performed using a combination of these algorithms with SMPI method. Here, it should be mentioned that when the implementation of the MPI's flow was accepted by the AVM, the replication of a MPI's flow to run by the AVMs intelligently would be more. The cost function of the SMPIA calculation based on Pseudo-Code was presented as $O(P^3)$ in Table 7.

```
(01)  Start
(02)    If (SelAVM_rank < MPI − VM_rank)
(03)      {
(04)        Send (VM_cf, SelAVM_id)
(05)        Recv (VM_cf, SelAVM_id)
(06)      }
(07)    Else
(08)      {
(09)        Recv(VM_cf, SelAVM_id)
(10)        Send(VM_cf, SelAVM_id)
(11)      }
(12)    End if
(13)  End
```

Table 4. Pseudo-code of send and receive

5 SOLVING THE TSRA AND STARVATION PROBLEM USING SMPIA

First, the non-SMPIA was implemented to each of the Greedy, Max-Min and Min-Min. In the following, SMPIA were applied to each algorithm in order to assess the performance of the proposed approach in the Greedy, Max-Min, and Min-Min cloud systems. Two approaches were executed parallel to each other. Any kinds of

```
(01)  Start
(02)  Input: ORT                                   /* ORT denotes other run time */
(03)  Output: OAVM                          /* OAVM denotes other virtual machine */
(04)  ORT ← −1
      /* There is always an input to the numbers of tasks plus one, so entries: Number
         of jobs +1 */
(05)  OAVM(ID) ← −1              /* OAVM (ID) denotes the ID of Other AVM */
(06)  Read VMs
(07)  VMsL ← Length(VMs)
(08)    For i ← 1 to VMsL
(09)      If AVM_i(T_CF) < TRT
                                 /* T_CF denotes execution time of current MPI's flow */
            /* AVM_i(T_CF) denotes the execution time of current flow in AVM_i */
                                                   /* TRT denotes temp run time */
(10)        TRT ← AVM_i(T_CF)
(11)        ORT ← TRT
(12)        OAVM(ID) ← AVM_i(RT)
(13)      End if
(14)    End for
(15)  End
```

Table 5. Pseudo-code of choose the best AVMs

```
(01)  Start
(02)  Input: FID, TCL, TRT.
(03)  Output: OAVM (ID), ORT.
(04)      OAVM(ID) ← −1
      /* There is always an input to the numbers of tasks plus one, Entries: Number
      of jobs +1 */
(05)      ORT ← −1
(06)      Related_AVMsS ← Related_AVMs(FID)
      /* Related_AVMS denotes size (capacity) of AVMs, Related_AVMs denotes all
      of Related AVMs */, /* Related_AVMs denotes the related AVMs */
(07)      RT ← zeros (C (Related_AVMs))
(08)  AVMs_Count ← zeros (C (Related_AVMs))
            /* AVMs_Count denotes the number to each corresponding AVM */
(09)    For i ← 1 to AVMs_Count
(10)      Index ← Related_AVMS(i)
(11)      A1 ← TCL
                    /* TCL denotes task computation load of the MPI's flow */
(12)      B1 ← AVMs_Load(1, Index)
                          /* AVMs_Load denotes the total load of the AVM */
(13)      C1 ← AVMS(Index, 4)
                               /* AVMS denotes size (capacity) of all AVMs */
(14)      D1 ← FID
(15)      E1 ← ES(D1, Index)
(16)      TT ← ((A1 + B1)/C1)/E1
(17)      RT(i) ← TT
(18)    End for
(19)    For i ← 1 to AVMs_Count
(20)      If (RT(i) < TRT)
(21)          TRT ← RT(i)
(22)          OAVM(ID) ← Related_AVMS(i)
(23)      End if
(24)    End for
(25)  End
```

Table 6. Pseudo-code of SMPIA implementation for AVM

changes in the state of successor flows in successor VMs (changes in load, capacity, etc.) affected the next flows of subsequent AVM. The SMPIA is applied on different TSRA strategies as follows:

5.1 Solving the TSRA Problem with Applying SMPIA onto the Greedy Algorithm

In the non-MPIA, the best VM was chosen according to the lowest load variance is calculated with the Greedy algorithm for predecessor flows through the Equation (13):

```
(01)  Start
(02)  Input: IA and PRs
      /* IA denotes individual array, PRs denotes processes that are running */
(03)  Output: EV
      /* EV denotes an evaluation value function that the same as maximum MS */
(04)  TT ← EI(IA, PRs)
                                  /* EI denotes evaluation individual or temp time */
      /* There is always an input to the numbers of tasks plus one, so, the job of
      evaluation individual function is to receive an array called individual and the
      execution MPI's flows (processing requests), and calculate how much time in-
      dividual needs to execute. that's mean: Entries: Number of jobs +1, This is
      a scientific contribution of the paper */
(05)  TL ← Lengh(IA)
                                  /*  TL denotes task length of the individual array  */
                          /*  Task or transaction list contains T[t] ← {T1, T2, ..., Tt} */
                             /*  Any array is a task and any the cell of array is a job */
(06)  EV ← −1
(07)    For i ← 1 to TL
(08)      Index ← IA(i)
(09)      DO
(10)        {
(11)        A1 ← PRs(i, 4)
(12)        B1 ← LAVMs(1, Index)
(13)        C1 ← SAVM(Index, 4)
                                        /* SAVM denotes the size (capacity) of AVM */
(14)        D1 ← PRs(i, 3)
(15)        E1 ← ES(D1, Index)
(16)        TT ← ((A1 + B1)/C1)/E1
(17)        If TT > EV              /* Founding the maximum run time (MS) */
(18)          EV ← TT
(19)        End if
(20)        }
(21)      While EV! = −1
(22)        If EV ← −1
(23)          Print "it has not changed"
(24)        End if
(25)      End do
(26)    End for
(27)  End
```

Table 7. Pseudo-code of the cost function to SMPIA

$$Var(x) = \frac{\sum(x - \bar{x})^2}{m - 1}. \tag{13}$$

In the third phase for the next flows the appropriate AVM was chosen according to the calculation of the MS formula for all AVMs in the SMPIA. The shortest MS time caused the selection of one of the AVMs. By choosing AVM in VMPIB, the speed of execution tasks could be enhanced, and the mean of the MS and TET parameters were minimized for the considered workflows. The variables and definitions used in the paper are listed in Table A1 of Appendix A. IDs is assigned to AVMs by the "For Loop" of the Greedy algorithm using Pseudo-Code in Table 8. The steps of applying the SMPIA onto the Greedy algorithm in the cloud system are exhibited in Figure 6.

```
(01)  Start
(02)  Input: AVMsS, InProcessReqs
      /* AVMsS denotes the size (capacity) of the AVMs and is global variable, the
      InProcessReqs denotes the number of MPI's flows (requests) */
(03)  Output: SAVM(ID)          /* SAVM(ID) denotes ID of selected AVMs */
(04)    For ipc ← 1 to InProcessCount
                  /* InProcessCount denotes the number of the InProcessReqs */
(05)        FlowID ← InProcessReqs(ipc, 3)
(06)        AVMsCount ← Length(AVMS)
              /* AVMsCount denotes the lengths (AVMs counter) of AVMs */
(07)        For i ← 1 to AVMsCount
(08)          If (FlowID == AVMS(i, 3))
(09)            SAVM(ID) (ipc) ← i
(10)          End if
(11)        End for
(12)      End for
(13)  End
```

Table 8. Pseudo-code of assign IDs to AVMs in the Greedy-SMPIA

5.2 Solving the TSRA Problem with Applying SMPIA onto the Max-Min and Min-Min Algorithms

In the non-SMPIA, the best VM was selected with the minimum completion time onto the Max-Min and Min-Min algorithms for the predecessor flows. Meanwhile, in the third phase, the appropriate AVM was chosen for the subsequent flows according to the calculation of the MS formula for all AVMs in the SMPIA. Both the Equations (14) and (15) are calculated for the flow of k on all VMs in Max-Min and Min-Min, respectively. By calculating these equations for all VMs, the minimum value would be found, which is clear in the located VM. The variables and definitions used in the paper are listed in Table A1 of Appendix A. The steps of applying

the SMPIA onto these Max-Min and Min-Min algorithms in the cloud system are exhibited in Figure 7.

$$Fitness = (\text{Max-Min})\, VM_{\min} = \frac{L_{CF}(k) + VMLoad(i)}{ES(k,i)}, \tag{14}$$

$$Fitness = (\text{Min-Min})\, VM_{\min} = ((A_1 + B_1)/C_1)\,/E_1. \tag{15}$$

Obviously, the current flow mappings onto the chosen AVM in the Max-Min and Min-Min algorithms were almost the same, the only subtle difference was that, in the Min-Min, the flow with low execution time could be assigned to the AVM with the minimum completion time.

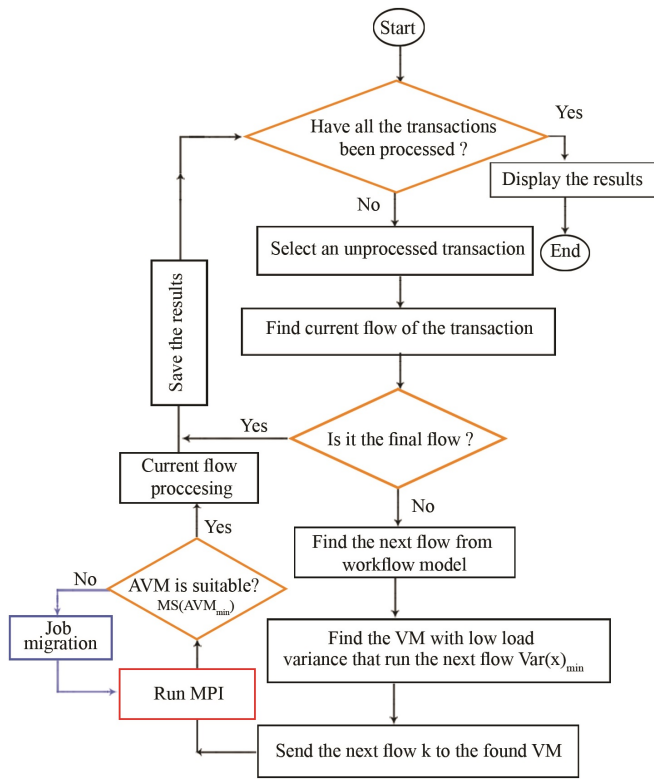


Figure 6. Flowchart of applying SMPIA to Greedy algorithm

The SMPIA was executed for Greedy, Max-Min and Min-Min algorithms based on pseudo-code in Table 9.

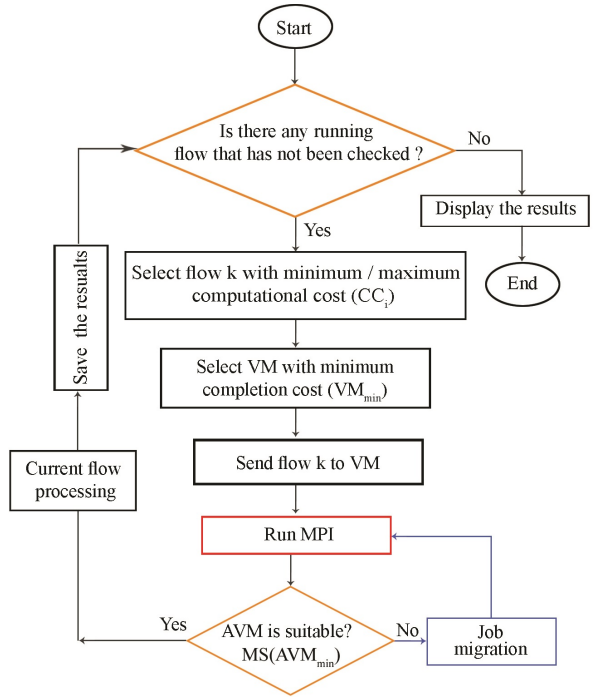


Figure 7. Flowchart of applying SMPIA to Max-Min and Min-Min algorithms

6 SOLVING THE STARVATION PROBLEM USING SMPIA

In this paper, the SMPIA is applied onto the Greedy, Max-Min and Min-Min algorithms. Then, the results of solving the starvation problem (i.e. large waiting times for small and big jobs) and the lack of sufficient resources are achieved as follows:

1. In comparison with the Greedy algorithm and the SMPIA, the best AVMs were specified in a short time so that the correspondence successor flows were executed at less time. Meanwhile, three parameters were simultaneously improved in Greedy algorithm (see Table 10).
2. In comparison with the Min-Min algorithm and the SMPIA, the TET and the completion time were considered priority. At first, those tasks were scheduled that had a minimum TET and a minimum completion time. In this way, the starvation was fixed for big tasks using the SMPIA, which was advantageous for bigger tasks in subsequent flows. The predecessor flows were processed earlier in small jobs. In addition, the subsequent flows with SMPIA were processed in big jobs earlier. Note that when the number of big tasks became more than the number of small tasks (i.e. increasing in the workload), this issue was observed

```
(01)  Start
(02)  L_SelAVMs ← (L_NF + L_SelAVMs)
                        /* L_SelAVMs denotes the load of the selected AVMs */
                          /* L_NF denotes the load of the next MPI's flow */
(03)  TRT ← (L_NF/C_SelAVM)
                        /* C_SelAVM denotes the size (capacity) of the SelAVM */
(04)  RT(SelAVM) ← TRT
                        /* RT(SelVM) denotes the run time of selected AVM */
(05)      If Enable-MPI = 1 and method = type of method (number: 1 or
          2 or 3)
                          /* Enables MPI and select the type of algorithm */
(06)          Calculate OAVM(ID), ORT          /* for next MPI's flow */
(07)          If OAVM(ID) > 0
(08)              SelAVM ← OAVM(ID)
(09)              TRT ← ORT
(10)              RT(SelAVM_i) ← TRT
                      /* RT(SelAVM_i) denotes run time of selected AVM_i */
(11)          End if
(12)      End if
(13)  End
```

Table 9. Pseudo-code of SMPIA to Greedy, Max-Min and Min-Min algorithms

noticeably. In this way, three parameters were simultaneously improved in the Min-Min (as listed in Table 10).

3. In comparison with the Max-Min algorithm and the SMPIA, the TET and completion time were prioritized as well. At first, those tasks were scheduled that had maximum TET and minimum completion time. This issue was the advantage of smaller tasks in the subsequent flows where the starvation was fixed in Max-Min for small jobs. It should be noted that the TET of the predecessor flows became longer especially for small jobs. This issue would lead to creating a change in the selection of AVMs, which was effective to decrease the minimum MS. Afterwards, the predecessor flows in small tasks were processed, but with smart MPI, then, the subsequent flows in small tasks were processed earlier. When the number of big tasks became more than the number of small tasks (i.e. increasing in the workload), this issue became noticeable. Subsequently, an increase was obtained in the number of flows of the AVMs due to the smartness of AVMs, which had minimum MS. At the meantime, the MS was reduced by changing the selection of AVMs, therefore, the performance of the Max-Min algorithm became more efficient, which was chosen to assign the next flow. Here, three parameters were improved simultaneously in the Max-Min. The greater change in AVM selection, the greater the increase or decrease in MS time would be. According to the previous discussions [38], the Minimum Completion Cloud (MCC), MEdian MAX (MEMAX) and Cloud Min–Max Normalization (CMMN)

generate a balance between MS and average cloud utilization in order to achieve a trade-off between them through solving the problem of starvation. According to discussions the SMPIA reduces the starvation problem by considering the TET and MS of the tasks. Other details are provided in Table 10.

7 EVALUATION OF SMPIA

Max-Min and Min-Min [24] and Greedy algorithms [25] were utilized extensively and successfully to map independent tasks onto resources in computational systems. They had $O(N^2 * M)$ [24] and $O(M * N)$ [25], respectively, in which N represents the number of tasks and M represents the number of processors.

In order to evaluate the proposed approach in the distributed system, some performance parameters such as TET, MS, and RU were utilized. Moreover, Greedy, Max-Min, and Min-Min algorithms were employed to investigate the proposed approach in the distributed system. The performance of the SMPIA was assessed by calculating the performance of the TET, MS, and RU parameters in both non-MPIA and SMPIA. Note that both parameters of the number of records and number of VMs were assumed to be constant. Some practical tests were implemented on the actual data in a homogenous environment including 4 DCs, 22 servers, 132 VMs, 132 flows, and 324 telecommunication equipment. In the following, programs were executed at least 50 times in a system with identical specifications to compute the average of parameters. Ultimately, the mean values were recorded as well. To do so, 201 535 records were collected on transactions (including deals and tenders) of the telephone company from 2011 to 2017. The simulation and implementation were carried out using MATLAB software. Besides, the considered experiments were done on a system the follow features: CPU 1.83 GHz, Core i7 4 GB RAM. First, the non-SMPIA was implemented to each of the Greedy, Max-Min and Min-Min. In the following, MPIA were applied to each algorithm in order to assess the performance of the proposed approach in the Greedy, Max-Min, and Min-Min cloud systems. Two approaches were executed parallel to each other. Any kinds of changes in the state of successor flows in successor VMs (changes in load, capacity, etc.) affected the next flows of subsequent AVM.

7.1 Evaluation of Total Execution Time

In this process, the execution time was calculated as the TETs using Equation (1). As illustrated in Figure 8, the TET decreases at 132 cloud workloads with SMPIA. Moreover, the maximum percent of improvement TET is 55.80 % at 132 cloud workloads in Min-Min algorithm; but SMPIA performs better than the non-SMPIA. In addition, the TET in Greedy-SMPIA and Max-Min-SMPIA improved in comparison with Greedy and Max-Min algorithms. This decrease reflects the impact of the SMPIA to optimize the TET parameter as well as to improve the performance of the proposed system. The implementation of next flows of transactions (i.e. deals and

tenders) with a minimum execution time was prioritized, due to the use of AVMs and ranking based on having the most number of connections with other AVMs. The initiating of any requests of transactions (deals and tenders) in the cloud-based system was carried out in a due time and in a short while. In this way, each request was answered in a short time. Other details are provided in Table 10.

7.2 Evaluation of Maximum Makespan

In this process, the MS was calculated as the maximum MS using Equation (2). As can be observed from Figure 9, the MS time was decreased to 132 cloud workloads with the MPIA. The maximum percent of improvement MS time is 55.94 % at 132 cloud workloads in Max-Min algorithm; but SMPIA outperforms the non-SMPIA. The maximum percent of improvement in Min-Min is 53.04 % but SMPIA performs better than non-MPIA. Furthermore, the maximum completion time of transactions (deals and tenders) in the proposed system with Greedy was less than other algorithms. The processing of each job was performed faster and completed in a short time. This decrease reflects the impact of the SMPIA to optimize the completion time parameter as well as to improve the system performance. Note that changing the choice of AVMs can be effective to reduce MS. In addition, the execution of next flows of transactions (deals and tenders) via a minimum completion time was prioritized because of the simultaneous use of AVMs and the obtained ranks based on the highest number of connections. After all, the requests for transactions (deals and tenders) were answered faster and the last jobs were completed sooner. The aforementioned results confirmed the effect of proposed approach on the appropriate distribution of load on the proposed system resources. Other details are provided in Table 10.

7.3 Evaluation of Resource Utilization

Equation (3) is formed to calculate the RU. As exhibited in Figure 10, the RU increases at 132 cloud workloads with SMPIA. Besides, the maximum percent of improvement RU is 12.28 % at 132 cloud workloads in Greedy algorithm; but SMPIA performs better than non-SMPIA in this way. The utilization of cloud resources for the Greedy has increased up to 87 % (12.28 %), which revealed the impact of the SMPIA to optimize the RU parameter. Greedy scored better value in utilization of resources than Max-Min and Min-Min. Note that any increase in the utilization of cloud resources emphases that the system was performing more efficiently using the SMPIA. Other details are listed in Table 10.

8 DISCUSSIONS AND ANALYSIS

The TET and the completion time of the workflows with the Min-Min and Max-Min algorithms were noticeably decreased using the application of MPIA. In addition,

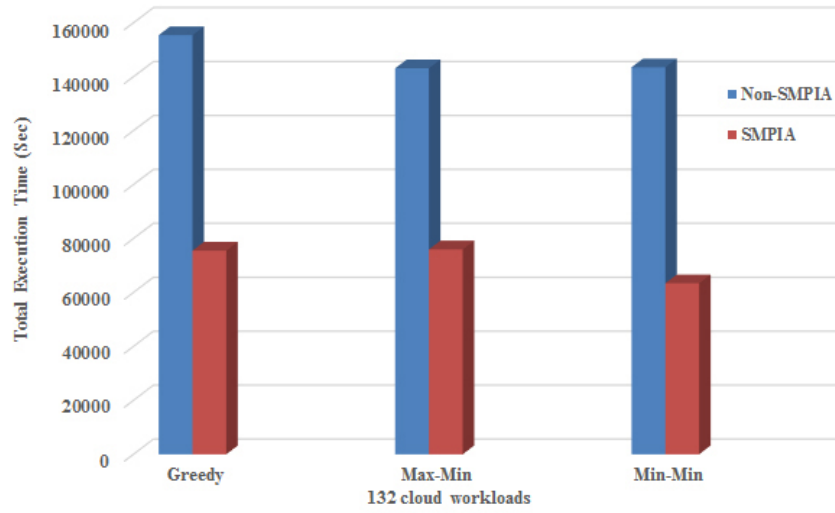


Figure 8. Effect of 132 cloud workloads on total execution time with Non-SMPIA and SMPIA

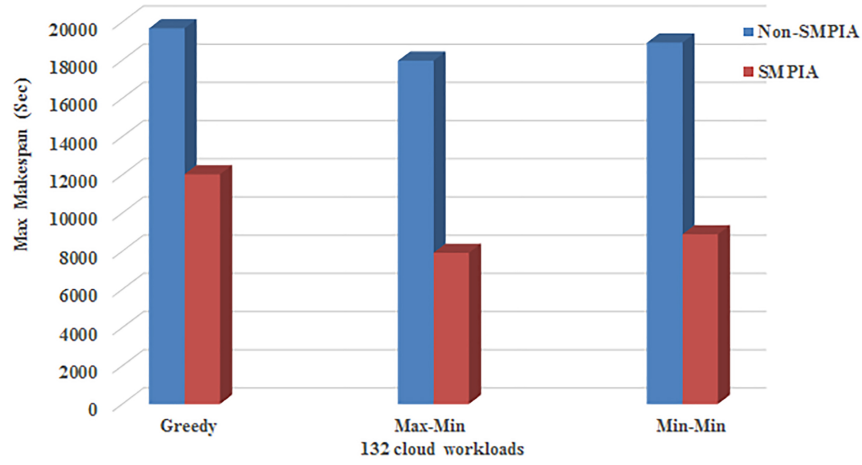


Figure 9. Effect of 132 cloud workloads on makespan with Non-SMPIA and SMPIA

Dataset	Non-SMPIA			SMPIA			Improvement (%)		
	TET	MS	RU [%]	TET	MS	RU [%]	TET [%]	MS %	RU [%]
MCITI	155 445	19 668	74.72 %	75 523	12 027	87 %	51.10 %	38.84 %	12.28 %
MCITI	143 172	17 976	75.72 %	76 008	7 919	87.52 %	49.91 %	55.94 %	11.80 %
MCITI	143 517	18 921	76.87 %	63 430	8 884	88.73 %	55.80 %	53.04 %	11.86 %

Table 10. Comparison performance parameters with Non-SMPIA and SMPIA

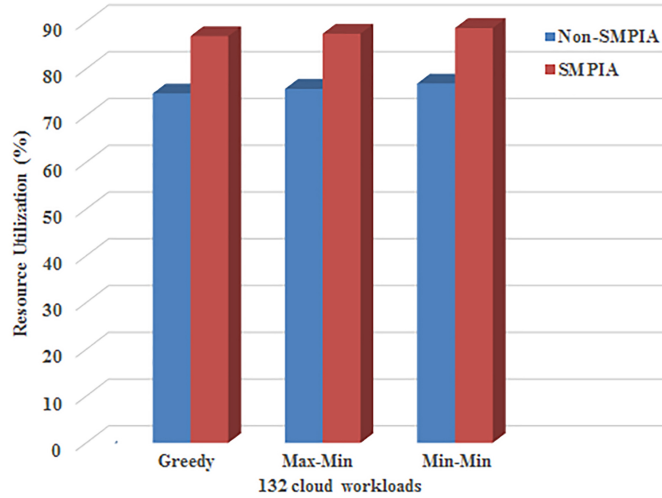


Figure 10. Effect of 132 cloud workloads on resource utilization with Non-SMPIA and SMPIA

the RU was meaningfully increased as well. It should be noted that changing the choice of an AVM to execution flows in the Min-Min and Max-Min algorithms was further evident on a number of workflows. This is due to the fact that in SMPIA, when AVM accepts the execution of the flow, it will again have a request to run by other AVMs. The effect of changing the choice of AVMs in MS time variations was increased with Min-Min and Max-Min algorithms. By choosing AVM in VMPIB, the speed of execution tasks could be enhanced, and the mean of the MS and TET parameters were minimized for the considered workflows.

The achieved results of calculating the time complexity of algorithms, the cost function of the SMPIA, and comparing the simulation, indicated that the Max-Min algorithm performance was noticeably better than the other algorithms; hence, the Max-Min algorithm was chosen to allocate the next job. Comparing the TET and MS of algorithms confirmed that the TET and MS with the SMPIA decreased, as compared to ones of the non-SMPIA. In addition, implementing the proposed approach decreased the TET from 143 517 seconds in the Min-Min algorithm to 63 430 seconds (55.80 %). Meanwhile, the MS time from the 17 976 seconds in Max-Min algorithm decreased to 7 919 seconds (55.94 %). Furthermore, the MS time in Min-Min decreased from 18 921 seconds to 8 884 seconds (53.04 %). Moreover, the RU rate in Max-Min algorithm increased from 75.72 % to 87.52 % (11.80 %). Accordingly, the executing of a workflow was purposefully enhanced. The cloud computing metrics (execution time and MS) and cloud providers (e.g. RU) were considered as part of the Multi-Objective Optimization (MOO) of real environments. Concerning the results of SMPIA in three parts, particularly those obtained with the proposed algorithm, optimal utilization of resources was provided to enhance the

system efficiency. Comparison of the results indicated the significant performance of the proposed approach by improving the efficiency and proper distribution of load in the cloud.

8.1 Resource Utilization with Total Execution Time

As illustrated in Figure 11, TET in non-SMPIA was 59.2 % greater than SMPIA at 74.72 % involvement levels and RU; nevertheless, TET in SMPIA was 59.2 % less than the non-SMPIA at 88.73 % involvement levels and RU. That is, the performance of the SMPIA was better than one of the non-SMPIA.

8.2 Resource Utilization with Makespan Time

In Figure 12, MS in non-SMPIA is 54.84 % greater than SMPIA at 74.72 % involvement levels and RU; nonetheless, MS in SMPIA is 54.84 % lesser than non-SMPIA at 88.73 % involvement levels and RU. Thus, the SMPIA outperforms the non-SMPIA in terms of the performance.

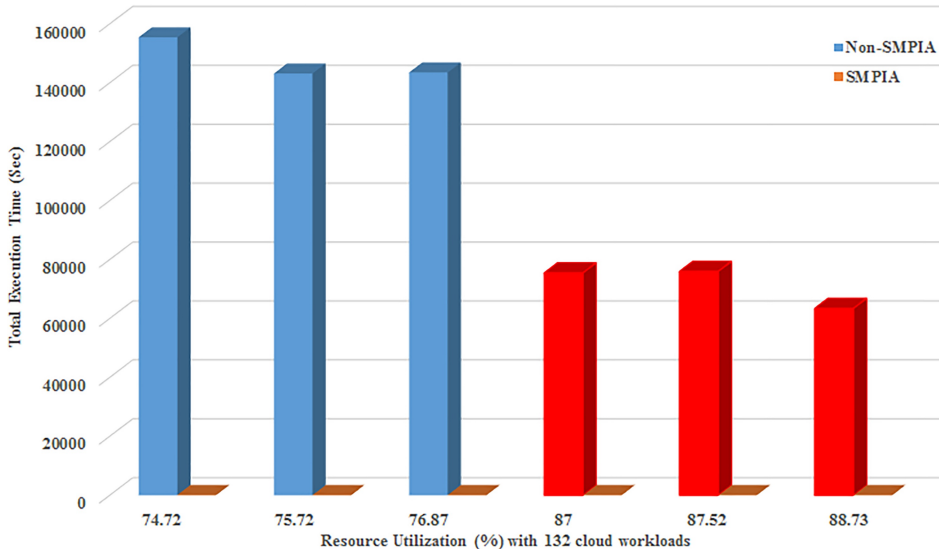


Figure 11. Effect of total execution time on resource utilization with SMPIA and non-SMPIA

9 CONCLUSIONS AND FUTURE RESEARCH

In this paper, a SMPIA with the probability of choosing the AVM was developed for the workflows using the Roulette Wheel selection method. This approach was

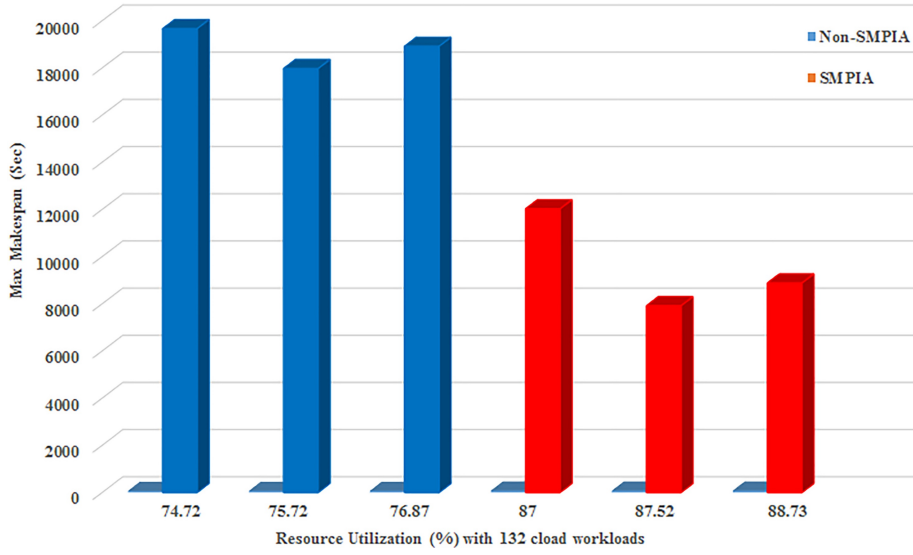


Figure 12. Effect of makespan on resource utilization with SMPIA and non-SMPIA

provided in three phases: resource-ranking, resource selection, and optimal task-resource mapping. In this way, MS and TET were significantly reduced using this approach, based on which the RU rate was simultaneously enhanced. The simulation results in the MATLAB confirmed that the minimum MS value was considered as the best solution for the optimal resource allocation and task mapping in the MPI based on the cloud resources. The best mode was belonged to Max-Min, which with SMPIA the MS up to 55.94 % and TET by up to 49.91 % improved. MS was chosen as an optimization factor, in which those solutions containing the minimum MS were chosen as the best task-resource mapping performance, which could reduce the cloud resource consumption. According to the obtained result, regarding the TET and MS of the tasks, the proposed SMPIA could reduce the starvation problem (large waiting time for small and big tasks) and the lack of sufficient resources. In addition, a multi-objective improvement approach was developed with a less complex time $O(p^3)$ for the SMPIA. An analysis of experimental actual data in the environment confirmed that the SMPIA was more efficient than the non-SMPIA and previous methods in the proposed system. As such, through the selection of AVMs and the computing the rank of them, the volume of jobs was properly distributed on the resources so that the operational efficiency of the system increased. In the following, an efficient task scheduling model was proposed using the SMPIA and Roulette Wheel selection method. However, some limitations of this paper could be the traffic congestion caused by MPI communications in the virtual network, the hidden topology of the network from the users' point of view, the delay caused through sending and receiving flows in the VMPIB. The mentioned system was

ready for the use immediately after practical application evaluation. Therefore, for future research, the performance prediction of SMPIA application will be developed using a fuzzy SMPIA to propose a minimum MS.

Acknowledgements

This project was conducted at Islamic Azad University – Sari Branch, Iran, with the cooperation of MCITI and Iran Telecommunication Company. The document of the research contract between the student, the professors, and Islamic Azad University – Sari Branch was recorded in the electronic book of Notary Public Office No. 29 of Sari under No. 15117, dated 10/07/2017.

APPENDIX

A VARIABLES AND DEFINITIONS USED IN THE PAPER

Variable	Definition
ORT	The Other Run Time is the execution time of another AVM while its initial value is equal to -1.
EI	The job of Evaluation Individual function is to receive an array called Individual and the execution flows, and calculate how much time Individual needs to execute, which is the same as the temp time.
EV	An Evaluation Value function that calculates maximum run time (MS).
ES(k, i)	The speed of execution flow$_k$ on VM_i
MS	MS is defined as completion time of the last job. Maximum MS is defined as the maximum total time of completion of all workflows. MS.
ET	Execution Time is defined as the TET of every single task from the beginning to the end.
RU	RU shows the extent of the involvement of hardware resources in the cloud to percentages, which are defined as the amount of resource involved.
TET	TET of all tasks from the beginning to the end.
TT	Temp Time, which takes to run an array called individual (the initial value is -1), $AVM_{\min}$
VMLoad(i)	The amount of current workload on AVM_i
L_{CF}	The computational Load of Current Flow.
A_1	The computational load (workloads) of the flows (process requests).
B_1	The total load of the AVMs.

C_1	The total capacity of the AVMs.
D_1	Flow type.
E_1	The execution speed of the flow on the VM.
OAVM(ID)	The Other AVM ID is the ID of another AVM and its initial value is equal to -1.
$RT(AVM_i)$	The Run Time of AVM_i.
ARU	Average RU.
$\overline{x}$	Mean values.
P	The number of execution transactions during the current time step.
m	The number of machines.
ART	Average Response Time.
ACT	Average Completion Time.
Zeros	Takes the array of the corresponding VMs and calculates matrix size (execution time) of each MPI's flow of AVMs in them and put it in the RTs.
TMLB	Task Mapping and Load Balancing.
CCS	Cloud Computing and Simulation.
NCT	Network Communication Time.
SMM	Segmented Min-Min.
WSRA	Workflow Scheduling and Resource Allocation.
ACU	Average Cloud Utilization.
MCS	Maximize Customer Satisfaction.
MCM	MPI Communication Management.
LAMPICS	Latency-Aware-MPI-Cloud-Scheduler.
MPAR	MPI-Performance-Aware-Reallocation.
IGATS	Improved Genetic Algorithm Task Scheduling.
CG	Conjugate Gradient.
NPA	Network Performance Awareness.
NPB	NAS Parallel Benchmarks.
CMPI	Cloud-MPI.

Table A1: List of variables and definitions

REFERENCES

[1] Ma, T.—Chu, Y.—Zhao, L.—Ankhbayar, O.: Resource Allocation and Scheduling in Cloud Computing: Policy and Algorithm. IETE Technical Review, Vol. 31, 2014, No. 1, pp. 4–16, doi: 10.1080/02564602.2014.890837.

[2] Gomez-Folgar, F.—Valin, R.—García-Loureiro, A. J.—Pena, T. F.—Zablah, J. I.—Ferreiro, R. V.: Cloud Computing for Teaching and Learning

MPI with Improved Network Communications. In: Mikroyannidis, A., Hernández Rizzardini, R., Schmitz, H.-C. (Eds.): Workshop on Cloud Education Environments (WCLOUD 2012). CEUR Workshop Proceedings, Vol. 945, 2012, pp. 22–27.

[3] Gomez-Folgar, F.—Indalecio, G.—Seoane, N.—Pena, T. F.—García-Loureiro, A. J.: MPI-Performance-Aware-Reallocation: Method to Optimize the Mapping of Processes Applied to a Cloud Infrastructure. Computing, Vol. 100, 2018, No. 2, pp. 211–226, doi: 10.1007/s00607-017-0573-6.

[4] Espínola, L.—Franco, D.—Luque, E.: MCM: A New MPI Communication Management for Cloud Environments. Procedia Computer Science, Vol. 108, 2017, pp. 2303–2307, doi: 10.1016/j.procs.2017.05.069.

[5] Zhuang, W.—Huang, L.: Overview of Cloud Computing Resource Allocation and Management Technology. 2019 6th International Conference on Systems and Informatics (ICSAI), Shanghai, China, 2019, pp. 713–718, doi: 10.1109/icsai48974.2019.9010101.

[6] Gong, Y.—He, B.—Zhong, J.: Network Performance Aware MPI Collective Communication Operations in the Cloud. IEEE Transactions on Parallel and Distributed Systems, Vol. 26, 2015, No. 11, pp. 3079–3089, doi: 10.1109/tpds.2013.96.

[7] He, Q.—Zhou, S.—Kobler, B.—Duffy, D.—McGlynn, T.: Case Study for Running HPC Applications in Public Clouds. Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC '10), 2010, pp. 395–401, doi: 10.1145/1851476.1851535.

[8] Amiri, M.—Mohammad-Khanli, L.: Survey on Prediction Models of Applications for Resources Provisioning in Cloud. Journal of Network and Computer Applications, Vol. 82, 2017, pp. 93–113, doi: 10.1016/j.jnca.2017.01.016.

[9] Kumar, K. D.—Umamaheswari, E.: Resource Provisioning in Cloud Computing Using Prediction Models: A Survey. International Journal of Pure and Applied Mathematics, Vol. 119, 2018, No. 9, pp. 333–342, https://acadpubl.eu/jsi/2018-119-9/articles/9/32.pdf.

[10] Rad, P.—Boppana, R. V.—Lama, P.—Berman, G.—Jamshidi, M.: Low-Latency Software Defined Network for High Performance Clouds. 2015 10th System of Systems Engineering Conference (SoSE), IEEE, 2015, pp. 486–491, doi: 10.1109/sysose.2015.7151909.

[11] Espínola, L.—Franco, D.—Luque, E.: Improving MPI Communications in Cloud. ACM-W Europe WomENcourage Celebration of Women in Computing, 2016, https://womencourage.acm.org/archive/2016/poster_abstracts/womENcourage_2016_paper_20.pdf.

[12] Antonenko, V.—Chupakhin, A.—Petrov, I.—Smeliansky, R.: Improving Resource Usage in HPC Clouds. In: Korenkov, V., Strizh, T., Nechaevskiy, A., Zaikina, T. (Eds.): Proceedings of the XXVII International Symposium on Nuclear Electronics and Computing (NEC 2019). CEUR Workshop Proceedings, Vol. 2507, 2019, pp. 180–184, http://ceur-ws.org/Vol-2507/180-184-paper-31.pdf.

[13] Xie, Z.—Shao, X.—Xin, Y.: A Scheduling Algorithm for Cloud Computing System Based on the Driver of Dynamic Essential Path. PloS One, Vol. 11, 2016, No. 8, Art. No. e0159932, doi: 10.1371/journal.pone.0159932.

[14] Almezeini, N.—Hafez, A.: An Enhanced Workflow Scheduling Algorithm in Cloud Computing. Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016), Vol. 2, 2016, pp. 67–73, https://www.scitepress.org/Papers/2016/59083/59083.pdf, doi: 10.5220/0005908300670073.

[15] Samadi, Y.—Zbakh, M.—Tadonki, C.: E-HEFT: Enhancement Heterogeneous Earliest Finish Time Algorithm for Task Scheduling Based on Load Balancing in Cloud Computing. 2018 International Conference on High Performance Computing and Simulation (HPCS), IEEE, 2018, pp. 601–609, doi: 10.1109/hpcs.2018.00100.

[16] Gawali, M. B.—Shinde, S. K.: Task Scheduling and Resource Allocation in Cloud Computing Using a Heuristic Approach. Journal of Cloud Computing, Vol. 7, 2018, No. 1, Art. No. 4, doi: 10.1186/s13677-018-0105-8.

[17] Singhal, S.—Patel, J.: Load Balancing Scheduling Algorithm for Concurrent Workflow. Computing and Informatics, Vol. 37, 2018, No. 2, pp. 311–326, doi: 10.4149/cai_2018_2_311.

[18] Sakellariou, R.—Zhao, H.: A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems. Proceedings of IEEE 18th International Parallel and Distributed Processing Symposium, 2004, pp. 111, doi: 10.1109/IPDPS.2004.1303065.

[19] Espínola, L.—Franco, D.—Luque, E.: DA-MCM: A Dynamic Application-Aware Mechanism for MPI Communications in Cloud Environments. Proceedings of the 2018 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), The 2018 World Congress in Computer Science, Computer Engineering and Applied Computing (CSCE'18), 2018, pp. 211–217, https://csce.ucmss.com/cr/books/2018/LFS/CSREA2018/PDP3183.pdf.

[20] Cruz, E. H. M.—Diener, M.—Pilla, L. L.—Navaux, P. O. A.: EagerMap: A Task Mapping Algorithm to Improve Communication and Load Balancing in Clusters of Multicore Systems. ACM Transactions on Parallel Computing (TOPC), Vol. 5, 2019, No. 4, pp. 1–24, doi: 10.1145/3309711.

[21] Hilman, M. H.—Rodriguez, M. A.—Buyya, R.: Task Runtime Prediction in Scientific Workflows Using an Online Incremental Learning Approach. 2018 IEEE/ACM 11th International Conference on Utility and Cloud Computing (UCC), 2018, pp. 93–102, doi: 10.1109/ucc.2018.00018.

[22] Dhinesh Babu, L. D.—Krishna, P. V.: Honey Bee Behavior Inspired Load Balancing of Tasks in Cloud Computing Environments. Applied Soft Computing, Vol. 13, 2013, No. 5, pp. 2292–2303, doi: 10.1016/j.asoc.2013.01.025.

[23] Malik, B. H.—Amir, M.—Mazhar, B.—Ali, S.—Jalil, R.—Khalid, J.: Comparison of Task Scheduling Algorithms in Cloud Environment. International Journal of Advanced Computer Science and Applications, Vol. 9, 2018, No. 5, pp. 384–390, doi: 10.14569/ijacsa.2018.090550.

[24] Tabak, E. K.—Cambazoglu, B. B.—Aykanat, C.: Improving the Performance of Independent Task Assignment Heuristics MinMin, MaxMin and Sufferage. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, 2014, No. 5, pp. 1244–1256, doi: 10.1109/tpds.2013.107.

[25] Madni, S. H. H.—Latiff, M. S. A.—Abdullahi, M.—Abdulhamid, S. M.—Usman, M. J.: Performance Comparison of Heuristic Algorithms for Task Scheduling in IaaS Cloud Computing Environment. PloS ONE, Vol. 12, 2017, No. 5, Art. No. e0176321, doi: 10.1371/journal.pone.0176321.

[26] Ma, J.—Li, W.—Fu, T.—Yan, L.—Hu, G.: A Novel Dynamic Task Scheduling Algorithm Based on Improved Genetic Algorithm in Cloud Computing. In: Zeng, Q.-A. (Ed.): Wireless Communications, Networking and Applications (WCNA 2014). Springer India, New Delhi, Lecture Notes in Electrical Engineering, Vol. 348, 2016, pp. 829–835, doi: 10.1007/978-81-322-2580-5_75.

[27] Jena, T.—Mohanty, J. R.: GA-Based Customer-Conscious Resource Allocation and Task Scheduling in Multi-Cloud Computing. Arabian Journal for Science and Engineering, Vol. 43, 2018, No. 8, pp. 4115–4130, doi: 10.1007/s13369-017-2766-x.

[28] Maswood, M. M. S.—Develder, C.—Madeira, E.—Medhi, D.: Energy-Efficient Dynamic Virtual Network Traffic Engineering for North-South Traffic in Multi-Location Data Center Networks. Computer Networks, Vol. 125, 2017, pp. 90–102, doi: 10.1016/j.comnet.2017.04.042.

[29] Pande, S. K.—Panda, S. K.—Das, S.: A Customer-Oriented Task Scheduling for Heterogeneous Multi-Cloud Environment. International Journal of Cloud Applications and Computing (IJCAC), Vol. 6, 2016, No. 4, Art. No. 1, 17 pp., doi: 10.4018/ijcac.2016100101.

[30] Zhou, Z.—Zhigang, H.: Task Scheduling Algorithm Based on Greedy Strategy in Cloud Computing. The Open Cybernetics and Systemics Journal, Vol. 8, 2014, No. 1, pp. 111–114, doi: 10.2174/1874110x01408010111.

[31] Chen, H.—Wang, F.—Helian, N.—Akanmu, G.: User-Priority Guided Min-Min Scheduling Algorithm for Load Balancing in Cloud Computing. 2013 National Conference on Parallel Computing Technologies (PARCOMPTECH), IEEE, 2013, pp. 1–8, doi: 10.1109/parcomptech.2013.6621389.

[32] Liang, B.—Dong, X.—Wang, Y.—Zhang, X.: A Low-Power Task Scheduling Algorithm for Heterogeneous Cloud Computing. The Journal of Supercomputing, 2020, Vol. 76, No. 9, pp. 7290–7314, doi: 10.1007/s11227-020-03163-8.

[33] Wu, M. Y.—Shu, W.—Zhang, H.: Segmented Min-Min: A Static Mapping Algorithm for Meta-Tasks on Heterogeneous Computing Systems. Proceedings of 9th Heterogeneous Computing Workshop (HCW 2000), IEEE, 2000, pp. 375–385, doi: 10.1109/hcw.2000.843759.

[34] Hung, T. C.—Hieu, L. N.—Hy, P. T.—Phi, N. X.: MMSIA: Improved Max-Min Scheduling Algorithm for Load Balancing on Cloud Computing. Proceedings of the 3rd International Conference on Machine Learning and Soft Computing (ICMLSC 2019), 2019, pp. 60–64, doi: 10.1145/3310986.3311017.

[35] Zhang, J.—Zhai, J.—Chen, W.—Zheng, W.: Process Mapping for MPI Collective Communications. In: Sips, H., Epema, D., Lin, H. X. (Eds.): Euro-Par 2009 Parallel Processing (Euro-Par 2009). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 5704, 2009, pp. 81–92, doi: 10.1007/978-3-642-03869-3_11.

[36] Arabnejad, H.—Barbosa, J.: Fairness Resource Sharing for Dynamic Workflow Scheduling on Heterogeneous Systems. 2012 IEEE 10th International Symposium

on Parallel and Distributed Processing with Applications, 2012, pp. 633–639, doi: 10.1109/ispa.2012.94.

[37] Hsu, C. C.—Huang, K. C.—Wang, F. J.: Online Scheduling of Workflow Applications in Grid Environments. Future Generation Computer Systems, Vol. 27, 2011, No. 6, pp. 860–870, doi: 10.1016/j.future.2010.10.015.

[38] Panda, S. K.—Jana, P. K.: Efficient Task Scheduling Algorithms for Heterogeneous Multi-Cloud Environment. The Journal of Supercomputing, Vol. 71, 2015, No. 4, pp. 1505–1533, doi: 10.1007/s11227-014-1376-6.

Mehran Mokhtari is Ph.D. student in computer (software engineering, Islamic Azad University, Sari Branch, Sari, Iran). Two Master of computer (Software Engineering from Islamic Azad University, Damghan and South Tehran Branch, Damghan and Tehran, Iran). Bachelor of computer science (Mathematics: Application in Computer, Islamic Azad University, Lahijan Branch, Lahijan, Iran). He has more than 7 research projects, more than 3 books, more than 30 paper published in conference and journals. Interests: ability to programing with the network simulator-ns-2 &&3 for more than 15 years, WSN, ICT, wireless network, cloud computing, QoS in VPN, installation and implementation of OSPF protocol on network topology, project risk management, performance analysis.

Peyman Bayat is Assistant Professor and Faculty Member of Computer Engineering Department, Islamic Azad University Rasht Branch, Rasht, Iran. Ph.D. in computer (Computer Systems Engineering from University Putra Malaysia). Master of computer (Computer Software Engineering, IAU of Arak). Bachelor of electronic (Electronic Engineering, IAU of Arak). He has more than 13 research projects, more than 7 books, more than 38 articles published in journals. Interests: philosophy and arts.

Homayun Motameni is Associated Professor and Faculty Member of Computer Engineering Department. Islamic Azad University Sari Branch, Sari, Iran. Ph.D. in computer (Software from Islamic Azad University, Science and Research Branch, Tehran, Iran). Master of computer science (Machine Intelligence from Islamic Azad University, Science and Research Branch, Tehran, Iran). Bachelor of computer (Software from Shaheed Behest University of Tehran). He has more than 10 research projects, more than 6 books, more than 60 conference papers, and more than 100 articles published in journals. Interests: software engineering, performance analysis, evolutionary computation, fuzzy systems.

Computing and Informatics, Vol. 40, 2021, 145–168, doi: 10.31577/cai_2021_1_145

CREDIT RISK ASSESSMENT OF BANKS' LOAN ENTERPRISE CUSTOMER BASED ON STATE-CONSTRAINT

Renjing Liu, Xuming Yang, Xinyu Dong, Boyang Sun

School of Management, Xi'an Jiaotong University
No. 28, Xianning West Road
Xi'an, China
e-mail: 18223205757@163.com

Abstract. Commercial banks are facing increasingly complex enterprise loan customers and businesses. It is important for banks' enterprise loan business to efficiently assess credit risks. Our study builds an enterprise credit risk assessment model based on the state and constraint of bank and customer, and get empirical researches with RF, SVM and DT algorithms. The results show that our model has excellent performance with accuracy 99 % and great characteristic importance in the evaluation of enterprise credit risk. The study can provide important decision-making reference for bank loan business and enrich the theoretical system of bank credit risk research.

Keywords: Credit risk assessment, state and constraint, enterprise loan, machine learning

Mathematics Subject Classification 2010: 68U35

1 INTRODUCTION

As it is universally acknowledged that commercial loan is the core source of bank profit and takes the most important position in banking business. However, in recent years, the non-performing loan ratio of commercial banks has been constantly on the rise. Taking China as an example, according to the announcement of China Banking Regulatory Commission in August 2020, the non-performing loan ratio

of China's commercial banks has risen to 1.94 %, with an increase of 7.2 % over last year. As one of the core subjects of commercial loans, enterprises have complex market economy network, huge loan amount, increasingly difficult asset quality management, and high credit risk. Enterprise's loans are known as the saying "one loan losses can lose nine loan profits". At the same time, in the current financial environment with the integration of Internet and digital technology, the bank's traditional business model that used to rely on the expansion of corporate credit scale to increase profits has squeezed the profit space of banks and accumulated a large number of customer risks [1]. With the continuous innovation and development of financial technology and the accumulation of massive customer data information in the banking industry, it has become one of the most urgent and important tasks for commercial banks to effectively control risks by means of digital technology based on the digital transformation of the banking industry driven through big data [2]. It is of important value for commercial bank credit business development to combine financial technology and big data of banks to get scientific and efficient evaluation of enterprise credit risk, which can help bank timely and objective assessment of enterprise customers and credit conditions, also provide more powerful decision support for loan business.

The integration of big data analysis and other new information technologies into the financial field has triggered a new round of digital reform in the financial industry that attracted a lot of scholars' interest in the research of corporate loan credit evaluation. Using the real-time data of bank customers to assess the credit risk of enterprises through big data analysis [3, 4] and machine learning [5, 6] methods are also increasingly intensified. At present, relevant researches are mainly carried out from two aspects. Firstly, based on the in-depth study of enterprise credit risk assessment model, the evaluation model was constantly improved and optimized by adding or introducing characteristic factors which affect enterprise credit, so as to improve the performance effect of assessment. For example, Minnis and Sutherland added tax and other indicators into the model construction to improve the effectiveness of the model for predicting corporate default risk [7]. However, too many characteristic indicators can cause the model to suffer from "dimensional disaster" [8]. In this regard, Tong's improvements put forward the LSOMAP-RVM credit model to evaluate the credit risk of domestic listed companies and solve the high-dimensional problem through algorithm and index fusion [9], but its application scenario was extremely limited. Only focusing on extraction of feature elements and optimization of credit model will, to a certain extent, result in all "preferences" of certain aspects such as enterprise state [10], managerial ability [11] overall industry characteristics [12] in the evaluation results, and fail to comprehensively evaluate enterprise credit risk.

Secondly, based on the research on the improvement of enterprise credit risk assessment methods, some machine learning algorithms were improved to overcome data and algorithm problems in enterprise credit assessment, such as data imbalance [13] and algorithm stability [14]. For example, Tian et al. built a new fuzzy set and the most advanced credit risk assessment algorithm model using the kernel-free

QSSVM basic model to deal with information label error [15]. Bu et al. created a new mixed information method, using mixed integrated information to predict enterprise credit risk, and demonstrated its advantages in short-term credit assessment [16]. Huang et al. improved the robustness of the probabilistic neural network model by determining the dimensions in advance [17]. The research based on the improvement of evaluation method can solve the data processing problems in the practice of algorithm evaluation, but the internal relationship mining of enterprise credit characteristic indexes are not enough.

Both the model optimization and the algorithm improvement in the existing research on enterprise credit risk evaluation focus more on the perspective of financial market overall credit risk and enterprise financial operation direction of the microcosmic perspective, but pay few attention to the enterprise and the bank individual state and inherent relationship, which can lead to some specific "bias" and "distortion" for credit evaluation results. Meanwhile it will also result in data structure problems such as unbalanced data. However, the generation of enterprise credit risk is not only closely related to the operation state of enterprises, but also correlated with the operation state and risk control ability of banks [10, 18]. Different from the previous single enterprise credit evaluation model of customer perspective, this paper analyzes the internal connection and restriction relationship between banks and their enterprise customers, and the model of enterprise credit risk evaluation is constructed based on the respective state and constrains of both bank and enterprise, and the SMOTE sampling method is used to overcome the imbalance data. Then we use the random forests and support vector machine (SVM) algorithms to evaluate the customer's credit state of the enterprise. The results show that the credit evaluation model based on the perspective is of high rationality and credibility. Compared with other scholar's researches, our relevant parameters all have 10 % to 20 % improvement, and data imbalance problem is solved well. The conclusion of the study can provide a certain bank loan management decision-making reference, at the same time enrich research perspectives and research model of enterprise credit assessment.

The rest of our article is structured as follows. We firstly introduce the algorithm basis used in our study in Section 2, including three machine learning algorithms and SMOTE algorithm. Next, based on the bank constraint theory, we construct a credit risk assessment model from both sides of the bank and the enterprise in Section 3. Then we get an empirical analysis of our enterprise credit risk assessment model with three machine learning algorithms and SMOTE algorithm with the data of a commercial bank in Section 4. Finally, in Section 5, we offer some concluding remarks.

2 ALGORITHMS BASIS

With the development of bank digital transformation and the generation of massive data, it has obvious advantages for machine learning algorithms in complex rela-

tional data analysis [19]. Machine learning can be divided into supervised learning and unsupervised learning. In this paper, the classification algorithm with supervised learning is selected. According to research of Choi et al. about all kinds of machine learning algorithms [6], meanwhile thinking about the bank and customer data missing, imbalance and associated features and so on, we finally choose random forest (RF) and support vector machine (SVM) algorithm, to evaluate the credit risk of bank corporate loans, and select a decision tree algorithm for supplementary research. The following is a brief introduction of these algorithms.

2.1 Random Forest Algorithm

2.1.1 The Decision Tree

Decision tree, known as classification tree, is the underlying tree structure applied to random forest algorithm. There are many kinds of decision trees, and the typical binary CART decision tree is widely used in classification problems. CART decision trees utilize Gini minimization criteria for feature selection and recursive modeling. For the training data set D with K categories, C_K represents the sample subset of class K, $|C_K|$ and $|D|$ are respectively the size of and D, then the Gini coefficient of set D is

$$Gini(D) = 1 - \mathrm{K}\sum_{K=1}^{K}\left(\frac{|C_K|}{|D|}\right)^2. \tag{1}$$

Suppose the discrete feature A is used to segment the data, then D is divided into D_1 and D_2 according to the value of A

$$\begin{aligned} D_1 &= \{D \mid A = a\}, \\ D_2 &= \{D \mid A \neq a\}. \end{aligned} \tag{2}$$

Then, under the condition of discrete feature A, Gini index of set D is:

$$Gini(D) = \frac{|D_1|}{D} Gini\left(\mathrm{D}_1\right) + \frac{|D_2|}{D} Gini\left(D_2\right). \tag{3}$$

Gini coefficient represents sample impurity degree, so the attribute with small Gini index is preferred during tree construction. While the type of attribute is greater than two, the Gini coefficient will be calculated for each combination of two categories of classification, and the classification combination that minimizes Gini index will be automatically selected. When the Gini coefficient of sample sets in nodes is less than the reference threshold, the number of samples is less than the reference threshold, or there are no more features, the algorithm converges.

2.1.2 Random Forest

Stochastic forests are essentially integrated learning derived from decision trees, proposed by Tin Kam Ho of Bell Laboratories in 1955. It works by generating

multiple single classification trees that can be learned and predicted independently. The steps to establish each classification tree are as follows:

1. Assuming that the sample set size is N, bootstrap sampling is adopted. N training samples are random and put back from the sample set to be the training set without a classification tree.
2. Assuming that the feature dimension of each sample is M, m features are randomly selected from M features as feature subset (m is far less than M), and the tree is divided from these m features each time, so as to calculate the optimal splitting mode.
3. Random sampling can ensure that there is no overfitting, so each decision tree is allowed to grow completely without pruning until it meets the predetermined requirement.

Random forest is mainly based on bagging thoughts [23], as shown in Figure 1. Every time there are replacement samples from the population N, about 2/3 of the samples forming the training set. The remaining one-third of the sample is called out of bag (OOB), and the OOB is excluded from each tree. Then the OOB error estimation model is used to estimate the accuracy and internal error of the prediction. Since OOB error rate is an unbiased estimate of random forest generalization error, its junction effect is approximately equal to k-fold cross validation. Therefore, the random forest does not need to be cross-verified, and at the same time, the model can be well generalized [24].

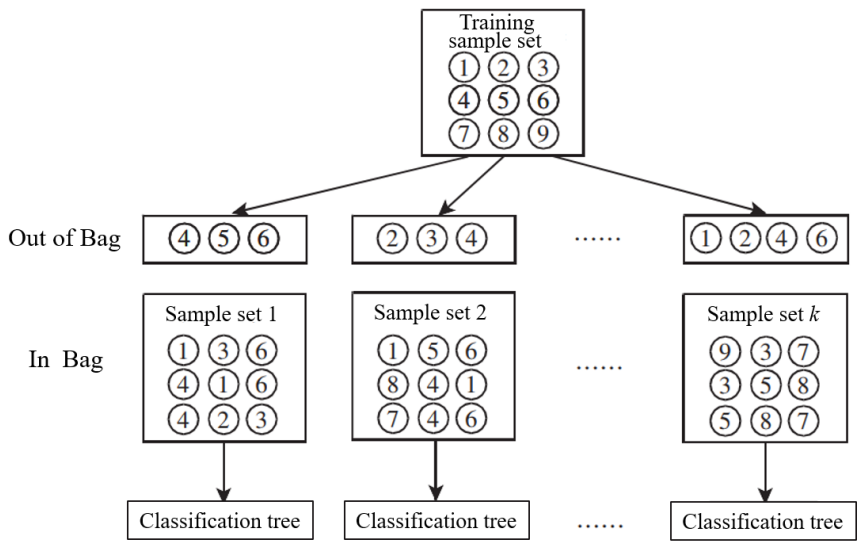


Figure 1. Schematic diagram of the Bagging process

2.2 Support Vector Machine

Support vector machine (SVM) is a machine learning algorithm based on the principle of structural risk minimization and statistical theory [25]. It is widely used in statistical regression and classification problems. For the credit classification problem of bank corporate customers, SVM has good classification performance and learning ability, meanwhile it has strong nonlinear approximation ability and can better overcome dimensional disasters, which can help process bank customer data very efficiently.

2.2.1 Linear SVN Model

First, choosing the hyperplane of the classifier in the sample space:

$$w^T x + b = 0. \tag{4}$$

The distance from any point in the sample space to the hyperplane is:

$$\mathrm{r} = \frac{\left|w^T x + b\right|}{||w||}. \tag{5}$$

Using hyperplane to classify samples:

$$\begin{cases} w^T x + b \geq +1, y_i = +1, \\ w^T x + b \leq -1, y_i = -1. \end{cases} \tag{6}$$

As shown in Figure 2, the samples are closest to the hyperplane so that the above equation holds are called support vectors. The sum of the distances from the support vectors to the hyperplane is the "interval":

$$r = \frac{2}{||w||}. \tag{7}$$

To make the maximum interval of the partition hyperplane of the classifier

$$\begin{cases} \min_{w,b} \frac{1}{2}||w||^2, \\ y_i \left(w^T x + b\right) \geq 1, i = 1, 2, \ldots, m. \end{cases} \tag{8}$$

2.2.2 Nonlinear SVM Model

However, many sample spaces are not linearly separable in reality, so a nonlinear transformation method is needed to transform the problem into a linear separable problem in the feature space of a certain dimension, so as to train linear support vector machines in the feature space of a higher dimension.

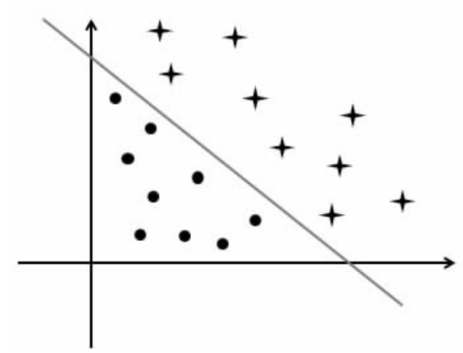


Figure 2. Schematic diagram of linear SVM model

The common transformation method is to use kernel function to transform. As shown in Figure 3, the given data set is shown in the figure on the left, and the hyperplane is divided into ellipses, which cannot be linearly divided. At this point, input vectors can be mapped into the high-dimensional feature space by introducing kernel functions, and be converted into the linear form of the figure on the right, so as to be converted into the form of linear SVM. Common nonlinear kernel functions are as follows:

1. Polynomial kernel function

$$\mathrm{k}\left(x_i, x_j\right) = \left(x_i^T x_j\right)^d. \tag{9}$$

2. Radial basis kernel function

$$\mathrm{k}\left(x_i, x_j\right) = \exp\left(\frac{||x_i - x_j||^2}{2\sigma^2}\right). \tag{10}$$

3. Sigmoid and functions

$$\mathrm{k}\left(x_i, x_j\right) = \tanh\left(\beta x_i^T x_j + \theta\right). \tag{11}$$

2.3 SMOTE Algorithm

Synthetic minority oversampling technique (SMOTE) is an improved scheme based on random oversampling algorithm. Its basic idea is to analyze minority samples and artificially synthesize new samples based on minority samples to add to the data set, which can well solve the problem of over-fitting and low model generalization resulted from simple random oversampling. The details are shown in Figure 4, and the algorithm flow is as follows.

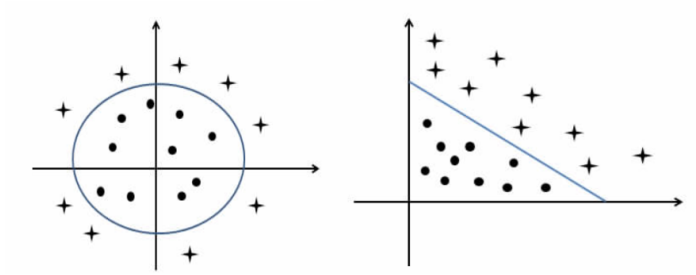


Figure 3. Schematic diagram of non-sexual SVM model

1. For each minority sample P, K nearest neighbor is obtained from the minority samples around it.
2. A minority class sample P_{bour} is selected among K nearest neighbors randomly.
3. The composite sample P_{new} is obtained by interpolation between P and P_{bour}, as shown in Formula (12).

$$P_{new} = P + rand(0,1) \times (P_{bour} - P) \tag{12}$$

where rand$(0,1)$ is the random number between $[0,1]$.

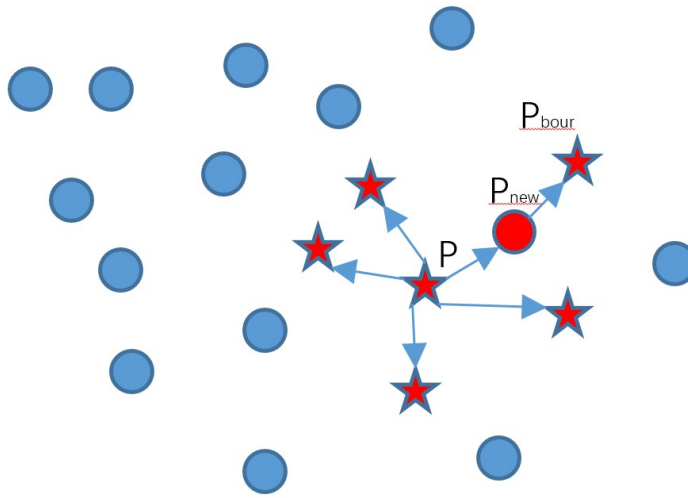


Figure 4. Schematic diagram of SMOTE algorithm sampling

3 THE CONSTRUCTION OF CREDIT RISK ASSESSMENT MODEL BASED ON STATE-CONSTRAINT

In general, the research on enterprise credit risk assessment focuses more on the assessment of enterprises as on an independent entity [3, 4, 15, 9], therefore, the influence of credit risk bearers will be ignored to a certain extent. However, in fact, the bearing party of credit risk will also affect the credit risk value of relevant enterprises. For example, when enterprises are faced with loans from banks and other financial institutions, their risk measurement is inconsistent because banks have a more complete credit system and solvency [26, 27]. Therefore, the credit risk of an enterprise is closely related to the enterprise itself, the loan bank and the whole market industry. Based on the bank constraint theory, this section constructs a credit risk assessment model from both sides – from the bank and from the enterprise side.

The theory of constraints (TOC) [28] was first proposed by Dr. Goldratt in his optimization production technique (OPT), which requires firms to establish management system to identify and eliminate constraints in the process of achieving goals. TOC emphasizes the importance of treating the enterprise as a system, considering and dealing with problems from the perspective of overall benefits. The enterprise credit risk involves not only the enterprise itself, but also the bearers of credit risk (generally referring to banks or investment companies) and the whole industry, etc. These factors will jointly act on the formation of enterprise credit risk, forming the whole integration of enterprise credit risk [27, 29].

The essence of the credit risk management of the bank's enterprise loan customers is to set a series of constraints for the enterprise, so as to reduce the probability of the loan enterprise to break the promise. For an enterprise, the higher the constraint force is, the smaller the risk of dishonesty will be. As for enterprise constraints, there are generally two aspects: self-constraints and external constraints [30]. From the subjective point of view, under the guidance of banks, enterprises will form a self-restraint system to restrict the occurrence of dishonest behaviors, including the loss of credibility at the present stage and the influence of dishonesty at the future stage, etc. These constraints can restrict enterprises' willingness of dishonest behaviors to a certain extent [31]. From an objective point of view, corporate dishonesty is subject to the relevant constraints of external banks, such as floating interest rate, overdue penalty, etc.

In addition, establishing relevant constraints and the guarantee of constraints need to depend on the operational state of both parties, and only a good operational state can ensure the operation of the constraint mechanism [32]. However when an enterprise is in a poor state of operation, even if it has no intention to break its promise, it has to break its promise because of its bad enterprise assets state. Of course, if the bank has a good risk control system, the warning is made in advance and this may help the enterprise to solve the problem, and the enterprise may still solve the loan repayment after the recovery.

To sum up, according to the research fully based on a domestic commercial bank loan business and the state-constraint theory, we have integrated the enterprise itself, risk takers, the overall industry and the environment generated by credit risk into a complete system. From the perspective of the states and constraints of both the enterprise and the bank, there are 15 characteristic indicators in 4 aspects selected from the bank's corporate loan database to construct a corporate loan credit risk assessment model, which is shown in Figure 5. The description and interpretation of the characteristics indicators is in Table 1. The following will introduce the characteristic indicators of the model.

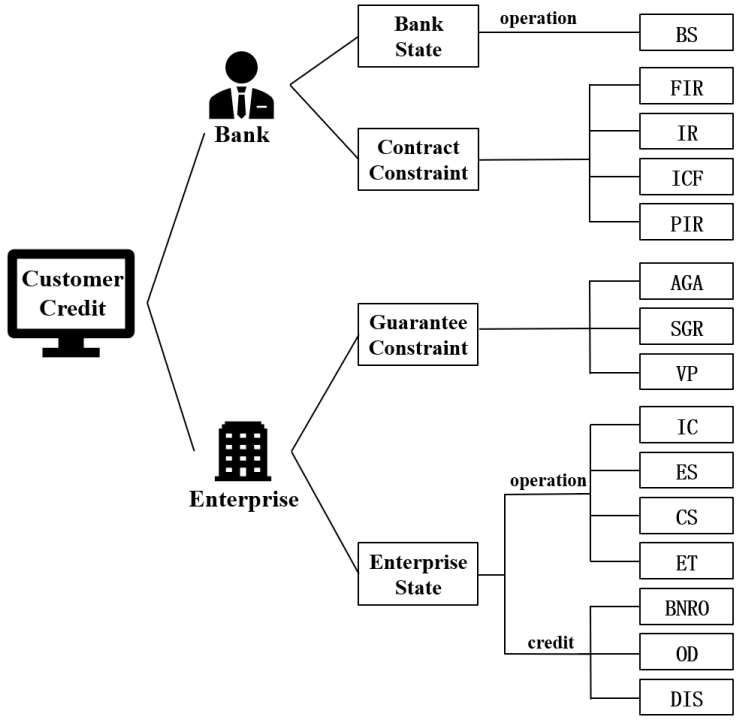


Figure 5. Credit risk assessment model based on state-constraint

3.1 Characteristic Indicators of Enterprise

3.1.1 Enterprise State

The enterprise state is an index reflecting the viability and operating state of an enterprise. The more ideal the state of an enterprise is, the less its credit risk will be, because "no enterprise wants to deliberately fight against the bank under nor-

Feature Dimension	**Characteristics**	**Characteristics Abbreviation**	**Characteristics Description**
Label	Customer Credit	CC	Dichotomous variable $CC \in 0, 1$
Bank State	Bank State	BS	Multiple categorical variables $BS \in [0, 100]$
Contract Constraint	Flexible Interest Rate	FIR	Continuous variable $FIR \in [0, +\infty)$
	Interest Rate	IR	Continuous variable $IR \in [0, +\infty)$
	Interest Calculation Frequency	ICF	Multiple categorical variables $ICF \in 0, 1, 2, 3$
	Penalty Interest Rate	PIR	Continuous variable $PIR \in [0, +\infty)$
Guarantee Constraint	Account of Guarantee Amount	AGA	Continuous variable $AGA \in [0, +\infty)$
	Security Guarantee Reliability	SGR	Continuous variable $SGR \in [0, +\infty)$
	Value of Pledges	VP	Continuous variable $VP \in [0, +\infty)$
Enterprise State	Industry Categories	ICF	Multiple categorical variables $CC \in 0, 1, 2 \cdots 17,$
	Enterprise Scale	ES	Multiple categorical variables $ES \in 0, 1, 2, 3$
	Customer State	CS	Multiple categorical variables $CS \in 0, 1, 2, 3$
Enterprise credit State	Extension Times	ET	Continuous variable $ET \in [0, +\infty)$
	Borrow New to Returnthe Old Times	BNRO	Continuous variable $BNRO \in [0, +\infty)$
	Overdue Days	OD	Continuous variable $OD \in [0, +\infty)$
	Debit Interest State	DIS	Dichotomous variable $DIS \in 0, 1$

Table 1. Description the characteristics indicators of the state-constrained enterprise credit evaluation model

mal circumstances". For credit risk assessment, the enterprise state mainly includes operation state and credit state.

The enterprise state is a comprehensive reflection index to measure the business state and financial state of the enterprise. In this paper, it mainly includes three categories of industries: enterprise scale (ES), customer state (CS) and industry category (IC). IC means the industry category in which the enterprise is located, which can reflect the overall market state of the current industry. It covers 18 main industry categories. ES represents the size of the enterprise and re-

flects the development state and overall size of the enterprise at the present stage. It is divided into four grades: large, medium, small and micro. The CS indicates the bank's assessment of the business state of the enterprise at the present stage. There are four levels of development, consolidation, adjustment and elimination.

Enterprise credit state is a comprehensive response index to measure the past dishonest behavior and credit state of enterprises, mainly including extension times (ET), Borrow New to Return the Old Times (BNRO), Overdue Days (OD) and Debit Interest State (DIS). ET is the total number of times in the past that the loan of the enterprise could not be repaid upon maturity and was approved to extend the repayment time, which can reflect the dishonest behavior and dishonest intention of the enterprise in the past. BNRO is the total number of times it fails to repay the loan on time and applies for a new loan again to repay part or all of the original loan after the loan is due (including the maturity after the extension). The larger BNRO is, the higher the credit risk of the company. OD means the total number of overdue days in the past when the enterprise loan is due and cannot be repaid on time, which can very cleanly reflect the state and degree of corporate credit default. DIS means that the company has defaulted or not on loan interest in the bank in the past, which can reflect the credit risk state of the company.

3.1.2 Guarantee Constraint

For the loan enterprise, its main constraint is the guarantee constraint in the loan. Guarantee constraint is the funds and assets paid by enterprises for guarantee when they draw loans, and it is one of the most important constraints for banks to restrict the credit loss of enterprises. Loan guarantee generally has the value equal to the enterprise loan fund, once the enterprise breaks the promise, the bank can collect the loan guarantee to repay part or the whole loan. Therefore, guarantee constraint plays an important role in alleviating enterprise credit risk. The guarantee constraint mainly includes three indexes – the guarantee amount (AGA), security guarantee reliability (SGR) and value of pledges (VP).

AGA is the total amount of the relevant account connected by the loan account, which can reflect the financial stability and reliability of the loan account. The higher the connected amount is, the more reliable the financial constraint of the loan account will be. SGR is the product of the enterprise security guarantee coefficient and the total amount of the security guarantee, which can reflect the security of the guarantee fund and the reliability of the guarantor, and high reliability can effectively restrict the occurrence of misconduct. VP means the total amount of assets mortgaged to the bank when the enterprise draws a loan, which can reflect the minimum guarantee of the loan. The existence of collateral can well restrict the enterprise to break the promise and reduce the risk of the bank's dead loan.

3.2 Characteristic Indicators of Bank

3.2.1 Bank State

Bank state (BS) in our study is the comprehensive evaluation score made by the head office of bank for each branch in the enterprise loan business, which reflects the credit evaluation ability, risk control ability and loan recovery ability of the bank in the enterprise loan business. Banks with a higher state/status of the bank have better risk control ability and means, and the cost and impact of their dishonesty is higher when facing such banks. Therefore, the state of the bank can indirectly affect the credit of the enterprise from the external environmental conditions.

3.2.2 Contract Constraint

For the bank of enterprise loan business, the main constraint it can carry out on the enterprise is the contract constraint of loan. Contract constraint is the related constraint of the contract signed by the bank and the enterprise when it grants loans to enterprises. Contract constraint has explicit legal effect and is of great importance to the dishonesty of enterprises. Contract constraints mainly include four indicators, such as flexible interest rate (FIR), interest rate (IR), interest calculation frequency (ICF) and penalty interest rate (PIR). FIR is the maximum floating interest rate provided by the bank according to the credit evaluation of the loan enterprise, which can encourage the enterprise to repay the loan on time and restrain the enterprise's dishonest behavior to some extent. IR is the loan interest rate set by the bank when it lends money to the enterprise. IR with different credit levels is different, so it can restrain the subsequent influence of the enterprise due to its bad records. ICF is the frequency of interest calculation agreed by both parties when the bank makes a loan, and faster frequency is with relative higher interest and faster reimbursement frequency. PIR means the overdue penalty interest rate that the loan enterprise does not repay the loan in accordance with the contract. PIR is generally higher than the contract interest rate, so it can restrain the enterprise from breaking the promise and generate excess penalty interest.

4 EMPIRICAL ANALYSIS OF ENTERPRISE CREDIT RISK ASSESSMENT

4.1 Data Source and Data Preprocessing

The research data in our study are from the enterprise-loan database of a commercial bank in China. After sorting and screening, 1 467 enterprise loan data are obtained, among which 119 are in default and 1 348 are in normal credit. Then based on the enterprise credit risk assessment model, the integrated screening was carried out, and each data obtained contained a total of 16 indicators, and the data set was

further preprocessed including missing items processing, numerical processing and standardized processing.

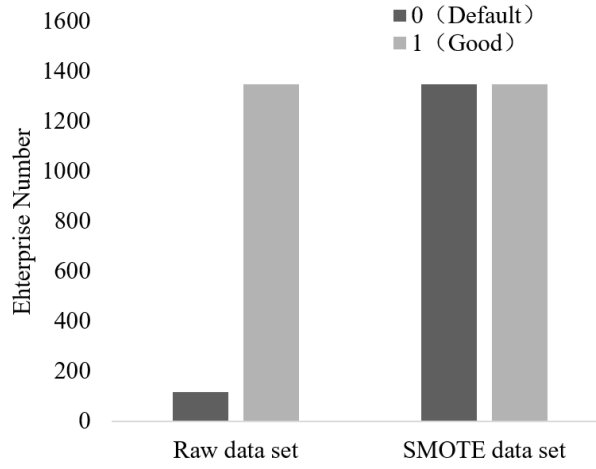


Figure 6. Credit distribution of enterprise customers

As shown in Figure 6 (left), the gap between the data set of default enterprises and normal enterprises is very large, in this case the unbalanced data problems in machine learning will affect the learning performance. Comparing with methods of under-sampling and over-sampling [33], we selected a SMOTE over-sampling algorithm dealing with unbalanced data sets, to obtain the new data set distribution as shown in Figure 6 on the right. Since the balance coefficient of the data set is lower than 15, the machine learning performance of the original data set is still reliable, so the subsequent analysis in this paper adopts two data sets for the comparative analysis.

4.2 Indicators Correlation Analysis

In this section, we used correlation analysis to test the rationality and correlation of the selection of indicators for the construction of the enterprise credit risk assessment model based on the state-constraint theory. Because the distribution of some indicators was unknown and does not show a normal distribution, meanwhile model included classification indicators, the Spearman rank correlation coefficient was finally used for the analysis.

As shown in Table 2, in the 15 indicators of the model, all indicators except the PIR were significantly related to customer credit (CC) at the 95 % confidence level, and 13 indicators were significantly related to CC at the 99 % confidence level. Significant correlation indicates that there is a significant correlation between the 15 indicators selected in the model and the dependent variable CC, which means

that the model construction is reasonable.

Index Name	**Case Number**	**Correlation Coefficient**	**Significance (P)**
CC	1 467	1.000**	0
BS	1 467	0.393**	0
FIR	1 467	0.075**	0.004
IR	1 467	0.201**	0
ICF	1 467	0.053*	0.042
PIR	1 467	0.006	0.808
AGA	1 467	0.117**	0
SGR	1 467	0.122**	0
VP	1 467	0.094**	0
ICF	1 467	0.194**	0
ES	1 467	0.137**	0
CS	1 467	0.245**	0
ET	1 467	0.230**	0
BNRO	1 467	0.279**	0
OD	1 467	0.652**	0
DIS	1 467	0.643**	0

Table 2. Spearman correlation coefficient between each indicator and CC

4.3 Selection of Model Evaluation Indexes

In machine learning, the commonly used indexes include accuracy, precision, recall, specificity, F1-value, AUC, etc. In our study, the emphasis of enterprise credit risk assessment is put to accurately identify enterprises with high credit risk, so as to provide decision-making support for bank enterprise credit. Therefore, we comprehensively selected four model evaluation indexes, including accuracy, recall rate, precision and F1-value.

Real Situation	Predicted Results	
	Case	Not the Case
Case	TP	FN
Not the Case	FP	TN

Table 3. Results of dichotomous problems refer to table

Accuracy. The accuracy rate represents the proportion of all correctly classified enterprises in all enterprise sampIs caseles, and the value interval is $[0, 1]$. The closer to 1 it is, the higher the identification accuracy of enterprise credit risk assessment is. Its calculation formula is as follows:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}. \tag{13}$$

Recall rate. Recall rate means the proportion of enterprises that will be identified as high credit risk enterprises in the truly high credit risk enterprises, and the value interval is [0,1]. The closer to 1 it is, the stronger the ability to identify enterprises with high credit risk is. Its calculation formula is as follows:

$$Recall = \frac{TP}{TP + FN}. \tag{14}$$

Precision. Precision means the true proportion of all enterprises with high credit risk identified by the model as high credit risk, and the value interval is [0,1]. The closer to 1 it is, the higher the credibility of the identification result is. Its calculation formula is as follows:

$$Precision = \frac{TP}{TP + FP}. \tag{15}$$

F1-value. F1-value is the harmonic average of recall rate and precision, which can comprehensively reflect the overall effect of recall rate and precision, so it is often used as the comprehensive evaluation parameter of machine learning. Its value interval is $[0, 1]$, the closer to 1 it is, the overall performance of the model is better. Its calculation formula is as follows:

$$F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}. \tag{16}$$

4.4 Empirical Research Results of Credit Evaluation

In this section, three algorithms including random forest (RF), support vector machine (SVM) and decision tree (DT) on R language platform are respectively used to empirically analyze the enterprise credit risk data of banks. Each experiment contained the original data set and the data set processed by SOMTE. Due to the sufficient data sample size, in order to improve the test credibility of the model, the ratio of analysis training set and sample set in RF and SVM is 6:4. In the DT algorithm, the ten fold cross validation method was used to train and test the model. The overall experimental results and comparative analysis are shown in Table 4.

4.4.1 Results of RF-Based Enterprise Credit Assessment Analysis

Regarding the parameter setting of the RF algorithm, the number of decision trees contained in the random forest was finally set to ntree = 600, and the number of variables used in the binary tree in the node mtry = 3.

The results are shown in Table 4. In the experiment with the original set, the overall sample corporate credit classification accuracy rate and F1-value was 99 % and 97 %, indicating that the overall recognition performance of the model was very good, and the high-risk enterprise samples credit classification recall and precision

Research	Characteristics	Algorithm	Accuracy	Recall	Precision	F1
Our study	State-constraint characteristics	RF	**0.99**	**0.96**	**0.99**	**0.97**
		SVM	**0.99**	**0.94**	**0.99**	**0.96**
		DT	0.95	0.64	0.89	0.74
		SMOTE +RF	0.94	0.87	0.90	0.88
		SMOTE +SVM	**0.99**	**0.99**	**0.99**	**0.99**
		SMOTE +DT	**0.99**	**0.99**	**0.99**	**0.99**
Qiu W et al. (2019) [35]	Credit history and company information characteristics	RF	0.84	0.75	0.50	0.60
		GBDT	0.87	0.76	0.59	0.66
		XGBOOST	0.82	0.85	0.47	0.61
Jain et al. (2020) [37]	Trade and loan characteristics	DT	**0.99**	0.78	0.81	0.79
		RF	**0.99**	0.78	**0.97**	0.86
		XGBOOST	**0.99**	0.83	0.95	0.89
Wang F et al. (2020) [34]	Online supply chain characteristics	LS-SVM	0.97	**0.96**	**0.97**	**0.96**
Jingming L et al. (2020) [36]	Enterprise competence characteristics	GSO-ELM	0.91	/	0.91	/

The top 5 values of each evaluation index are bolded;
GBDT- Gradient Boosting Decision Tree;
LS-SVM- Least Squares SVM;
GSO-ELM- Group Search Optimizer- Extreme Learning Machine.

Table 4. Empirical analysis results of enterprise credit evaluation

was 96 % and 99 %, indicating that the model's ability to identify key risks was outstanding as well. The result is significantly better than similar studies of Qiu [35] and Jain [37] with the same RF algorithm but different model characteristics, as well as better than research results of Qiu [35] and Li [36] with their advanced methods but different model characteristics, which shows that our credit risk assessment model on state-constraint of both bank and enterprise is reliable with pretty performance. In the experiment of SMOTE set, the overall classification accuracy rate and F1-value dropped to 94 % and 88 %, and the recall and precision dropped 87 % and 90 %, and the overall risk identification ability of the model drops significantly. After inspection and analysis, we believe that the reason for the decline in the performance of the SMOTE data set model may be that the imbalance of the total sample set is relatively small, and the RF algorithm itself has greater adaptability and tolerance for data imbalance, and high-risk enterprise sample groups have high similarities. Using the SMOTE algorithm will oversample a small number of samples with high risk to form new samples with a smaller gap from the original samples, resulting in similar "overfitting" problems, leading to learner model performance

worse. At the same time, we found that this problem does not exist in other two algorithms.

4.4.2 Results of SVM-Based Enterprise Credit Assessment Analysis

Regarding the parameter setting of SVM, by comparing the classification accuracy of high-risk enterprise samples in the SVM algorithm, the radial basis kernel function (RBF) was finally selected as the kernel function of the SVM model. There are two important parameters in RBF: gamma and cost. Gamma is a parameter of the kernel function that controls the shape of the segmented hyperplane. The larger the gamma, the more support vectors and the wider the range of training samples. Cost represents the cost parameter of the model's error cost. The greater the cost, the greater the model's penalty for errors, the more complex the generated classification boundary, and the smaller the error in the corresponding training set, but it is also possible that the too small cost can lead to overfitting problems. Considering the balance of learning performance and efficiency, we finally let the gamma parameter range to $(10^{-6}, 10)$ and the cost parameter range to $(10^{-6}, 10^{10})$.

In the SVM algorithm experiment, the overall classification accuracy and F1-value of the original data set was 99 % and 96 %, meanwhile the recall and precision was 94 % and 99 %, indicating that the learner's recognition ability for the samples of low-risk enterprises was higher than that of high-risk enterprises. But the comprehensive effect of our model is still pretty fine, which is a little senior than research of Wang et al. with LS-SVM in 2020 [34]. In the SMOTE set, overall classification accuracy and precison of the dataset was still 99 %, but recall rate increased from 94 % to 99 %, and F1-value reached to 99 %. This indicated that SMOTE had a certain improving effect on the SVM model, and our final results were also better than other researches.

4.4.3 Results of DT-Based Enterprise Credit Assessment Analysis

Since the SMOTE algorithm had different effects in the RF and SVM models, we added the experiment of the decision tree algorithm to obtain more reliable results. In the DT experiment, because there were too many categorical feature variables, the CHAID decision tree suitable for processing multivariate and categorical variables was selected. The parent node and child node of the minimum number of cases were 100 and 50, respectively, and the maximum number of classes was 3.

As can be seen from the results in Table 4, the overall classification accuracy of the original data set was 95 %, but the F1-value was only 74 %, and the recall and precision was only 64 % and 89 %, indicating that the decision tree model has poor recognition ability for high-risk groups, even though this result was better than result of Qiu et al. with GBDT [35]. In SMOTE set, four evaluation indexes all increased to 99 %, especially the recall rate suggested a 33 % increase over the original set. This result confirms our analysis in the stochastic forest algorithm, and indicates that SMOTE has an obvious effect in treating the imbalance data set,

which can resolve the data imbalance and improve the performance of the learner to some extent.

4.5 Importance of Risk Assessment Characteristic Indicators

The stronger the ability of the risk assessment characteristic indicators to distinguish between enterprise customer credit (default and normal), the higher its importance, that is, the characteristic indicators have obvious individual effects on customer credit assessment, and play a significant role in risk credit risk assessment. Figure 7 shows the accuracy reduction characteristics importance based on the RF algorithm and the characteristics importance of the Gini coefficient. Although the importance of a few indicators is inconsistent, the overall results were regionally consistent. In our study, the 15 characteristic indicators based on the state-constraint model all had a certain level of credibility. Among them, the top 5 indicators such as OD, DIS, CS, IC and IR were ranked among the top 5 with the importance of accuracy contribution over 0.015 and the importance of Gini coefficient over 8, which showed a strong ability to distinguish enterprise customer credit. PIR and BNRO were both at the bottom, which showed poor ability to distinguish customer credit. The importance of the Gini coefficient for the other eight indicators, all exceeded 4, which was at an intermediate level, and played a certain role in promoting customer credit differentiation.

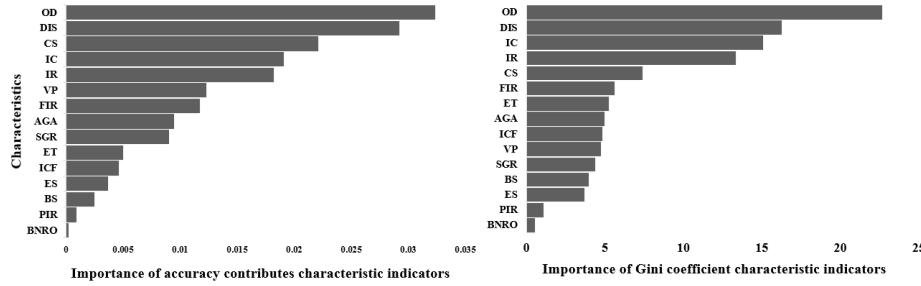


Figure 7. Importance of characteristic indicators of a risk assessment model

5 CONCLUSION AND DISCUSSION

The digital transformation of the financial industry is constantly developing. Commercial banks and other financial institutions will inevitably face increasingly complex enterprise loan customers and businesses. The scientific and digital assessment of enterprise loan customer credit risk is crucial. Based on the theory of constraint, our study extracts characteristic indicators from the state and constraint of both the bank and enterprise and builds an enterprise credit risk assessment model. In

our empirical research, the comprehensive evaluation recognition rate of the overall sample and high-risk credit enterprises can both reach up to 99 %. The results based on RF and SVM in our study are better than others' researches with same methods but different model characteristics, as well as better than others' researches with both different methods and model characteristics, which shows that our model has excellent performance and reliability for the evaluation of enterprise credit risk. At the same time, it shows that the unbalanced data processing based on the SMOTE algorithm does not significantly improve the performance of the RF algorithm, but it has a significant improvement in the performance of SVM and DT algorithms, which can well overcome the problem of data imbalance. Considering both the performance and robustness of the state-constrain model, SVM seems to be a better choice for the credit risk assessment.

In addition, about 90 % of the characteristic indicators in our research model have a significant correlation with customer credit at a 99 % confidence level. At the same time, the importance of the Gini coefficient is great enough, indicating that the characteristic indicators are highly distinguishable among customer credits. The ability to accurately assess credit risk of our model is strong, and the model construction in this article is very reasonable. This study's credit risk assessment model and the importance of its characteristic indicators can help banks to better understand the internal relationship between banks and enterprise customers, so that banks can better review and control enterprise credit risks in the corresponding feature dimensions. Thereby our study can provide important decision-making references for banks and enrich theoretical system of the credit risk research.

There are some limitations in our study. Firstly, the data used in our study comes from a commercial bank in China, and the data information is limited, although we have done a lot of exploration and attempted to find the optimal algorithm for the model. Due to the nature of machine learning and the limited data it is hard to figure out the internal relationship between model data and machine learning algorithms and give a fixed optimal algorithm. At the same time, the applicability of the conclusion in different countries and different regulatory policies needs to be further explored. In addition, we only study the credit risk assessment of commercial bank customers, but do not expand the study to other non-bank financial institutions. In the future, we will try to focus on customer credit risk of banks and financial institutions in different countries, continuously improve the research model, and explore the universality of credit risk assessment of our study.

REFERENCES

[1] Katre, S. M.: Analysis of Problems Faced by Public Sector Banks and Cooperative Banks and Strategies to Overcome w.s.r. to Ahmednagar City. IBMRD's Journal of Management and Research, Vol. 1, 2012, No. 1, pp. 84–87.

[2] Laurenceson, J.—Chai, J. C. H.: State Banks and Economic Development in China. Journal of International Development, Vol. 13, 2001, No. 2, pp. 211–225, doi: 10.1002/jid.727.

[3] Luvizan, S. S.—Nascimento, P. T.—Yu, A.: Big Data for Innovation: The Case of Credit Evaluation Using Mobile Data Analyzed by Innovation Ecosystem Lens. 2016 Portland International Conference on Management of Engineering and Technology (PICMET), IEEE, 2016, pp. 925–936, doi: 10.1109/picmet.2016.7806738.

[4] Hurley, M.—Adebayo, J.: Credit Scoring in the Era of Big Data. The Yale Journal of Law and Technology, Vol. 18, 2016, pp. 148–216.

[5] Golbayani, P.—Wang, D.—Florescu, I.: Application of Deep Neural Networks to Assess Corporate Credit Rating. 2020, arXiv: 2003.02334v1.

[6] Choi, J.—Suh, Y.—Jung, N.: Predicting Corporate Credit Rating Based on Qualitative Information of MD & A Transformed Using Document Vectorization Techniques. Data Technologies and Applications, Vol. 54, 2020, No. 2, pp. 151–168, doi: 10.1108/dta-08-2019-0127.

[7] Minnis, M.—Sutherland, A.: Financial Statements as Monitoring Mechanisms: Evidence from Small Commercial Loans. Journal of Accounting Research, Vol. 55, 2017, No. 1, pp. 197–233, doi: 10.1111/1475-679x.12127.

[8] Oseledets, I. V.—Tyrtyshnikov, E. E.: Breaking the Curse of Dimensionality, Or How to Use SVD in Many Dimensions. SIAM Journal on Scientific Computing, Vol. 31, 2009, No. 5, pp. 3744–3759, doi: 10.1137/090748330.

[9] Tong, G.—Li, S.: Construction and Application Research of Isomap-RVM Credit Assessment Model. Mathematical Problems in Engineering, Vol. 2015, 2015, Art. No. 197258, doi: 10.1155/2015/197258.

[10] Yin, W.—Liu, X.: Bank Versus Nonbank Financial Institution Lending Behaviour: Indictors of Firm Size, Risk or Ownership. Applied Economics Letters, Vol. 24, 2017, No. 18, pp. 1285–1288, doi: 10.1080/13504851.2016.1273473.

[11] Bonsall, S. B.—Holzman, E. R.—Miller, B. P.: Managerial Ability and Credit Risk Assessment. Management Science, Vol. 63, 2016, No. 5, pp. 1425–1449, doi: 10.1287/mnsc.2015.2403.

[12] Bourgain, A.—Pieretti, P.—Zanaj, S.: Financial Openness, Disclosure and Bank Risk-Taking in MENA Countries. Emerging Markets Review, Vol. 13, 2012, No. 3, pp. 283–300, doi: 10.1016/j.ememar.2012.01.002.

[13] Huang, Y. M.—Hung, C. M.—Jiau, H. C.: Evaluation of Neural Networks and Data Mining Methods on a Credit Assessment Task for Class Imbalance Problem. Nonlinear Analysis: Real World Applications, Vol. 7, 2006, No. 4, pp. 720–747, doi: 10.1016/j.nonrwa.2005.04.006.

[14] Lou, Y.: The Research on Corporate Credit Risk Evaluation Model Based on Fuzzy Neural Network. Journal of Central South University, Vol. 19, 2013, No. 5 (in Chinese).

[15] Tian, Y.—Sun, M.—Deng, Z.—Luo, J.—Li, Y.: A New Fuzzy Set and Nonkernel SVM Approach for Mislabeled Binary Classification with Applications. IEEE Transactions on Fuzzy Systems, Vol. 25, 2017, No. 6, pp. 1536–1545, doi: 10.1109/tfuzz.2017.2752138.

[16] Bu, D.—Kelly, S.—Liao, Y.—Zhou, Q.: A Hybrid Information Approach to Predict Corporate Credit Risk. The Journal of Futures Markets, Vol. 38, 2018, No. 9, pp. 1062–1078, doi: 10.1002/fut.21930.

[17] Huang, X.—Liu, X.—Ren, Y.: Enterprise Credit Risk Evaluation Based on Neural Network Algorithm. Cognitive Systems Research, Vol. 52, 2018, pp. 317–324, doi: 10.1016/j.cogsys.2018.07.023.

[18] Kim, J. B.—Song, B. Y.—Stratopoulos, T. C.: Does Information Technology Reputation Affect Bank Loan Terms? The Accounting Review, Vol. 93, 2018, No. 3, pp. 185–211, doi: 10.2308/accr-51927.

[19] Manogaran, G.—Chilamkurti, N.—Hsu, C. H.: Special Issue on Advancements in Artificial Intelligence and Machine Learning Algorithms for Internet of Things, Cloud Computing and Big Data. International Journal of Software Innovation, Vol. 7, 2019, No. 2.

[20] Janitza, S.—Strobl, C.—Boulesteix, A.-L.: An AUC-Based Permutation Variable Importance Measure for Random Forests. BMC Bioinformatics, Vol. 14, 2013, No. 1, Art. No. 119, doi: 10.1186/1471-2105-14-119.

[21] Gislason, P. O.—Benediktsson, J. A.—Sveinsson, J. R.: Random Forests for Land Cover Classification. Pattern Recognition Letters, Vol. 27, 2006, No. 4, pp. 294–300, doi: 10.1016/j.patrec.2005.08.011.

[22] Rodriguez-Galiano, V. F.—Ghimire, B.—Rogan, J.—Chica-Olmo, M.—Rigol-Sanchez, J. P.: An Assessment of the Effectiveness of a Random Forest Classifier for Land-Cover Classification. ISPRS Journal of Photogrammetry and Remote Sensing, Vol. 67, 2012, pp. 93–104, doi: 10.1016/j.isprsjprs.2011.11.002.

[23] Breiman, L.: Random Forest. Machine Learning, Vol. 45, 2001, No. 1, pp. 5–32, doi: 10.1023/A:1010933404324.

[24] Qin, X.—Li, Q.—Dong, X.—Lv, S.: The Fault Diagnosis of Rolling Bearing Based on Ensemble Empirical Mode Decomposition and Random Forest. Shock and Vibration, Vol. 2017, 2017, Art. No. 2623081, doi: 10.1155/2017/2623081.

[25] Chapelle, O.: Training a Support Vector Machine in the Primal. Neural Computation, Vol. 19, 2007, No. 5, pp. 1155–1178, doi: 10.1162/neco.2007.19.5.1155.

[26] Lookman, A. A.: Bank Borrowing and Corporate Risk Management. Journal of Financial Intermediation, Vol. 18, 2009, No. 4, pp. 632–649, doi: 10.1016/j.jfi.2008.08.006.

[27] Laeven, L.—Levine, R.: Bank Governance, Regulation and Risk Taking. Journal of Financial Economics, Vol. 93, 2009, No. 2, pp. 259–275, doi: 10.1016/j.jfineco.2008.09.003.

[28] Stoi, R.—Kühnle, B. A.: Theory of Constraints. Controlling – Zeitschrift für Erfolgsorientierte Unternehmenssteuerung, Vol. 14, 2002, No. 1, pp. 55–56, doi: 10.15358/0935-0381-2002-1-55.

[29] Lee, J.—Naranjo, A.—Sirmans, S.: Exodus from Sovereign Risk: Global Asset and Information Networks in the Pricing of Corporate Credit Risk. The Journal of Finance, Vol. 71, 2016, No. 4, pp. 1813–1856, doi: 10.1111/jofi.12412.

[30] Wagner, J.: Credit Constraints and Exports: Evidence for German Manufacturing Enterprises. Applied Economics, Vol. 46, 2014, No. 3, pp. 294–302, doi: 10.1080/00036846.2013.839866.

[31] Hall, K.: The Psychology of Corporate Dishonesty. Australian Journal of Corporate Law, Vol. 19, 2006, p. 268–286.

[32] Sassi, S.—Gasmi, A.: The Effect of Enterprise and Household Credit on Economic Growth: New Evidence from European Union Countries. Journal of Macroeconomics, Vol. 39, 2014, Part A, pp. 226–231, doi: 10.1016/j.jmacro.2013.12.001.

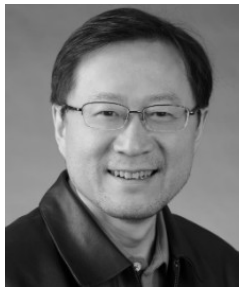
[33] Yap, B. W.—Khatijahhusna, A. R.—Rahman, H. A. A.—Fong, S.—Khairudin, Z.—Abdullah, N. N.: An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets. In: Herawan, T., Deris, M., Abawajy, J. (Eds.): Proceedings of the First International Conference on Data Engineering (DaEng-2013). Springer, Singapore, Lecture Notes in Electrical Engineering, Vol. 285, 2014, pp. 13–22, doi: 10.1007/978-981-4585-18-7_2.

[34] Wang, F.—Ding, L.—Yu, H.—Zhao, Y.: Big Data Analytics on Enterprise Credit Risk Evaluation of e-Business Platform. Information Systems and e-Business Management, Vol. 18, 2020, pp. 311–350, doi: 10.1007/s10257-019-00414-x.

[35] Qiu, W.—Li, S.—Cao, Y.—Li, H.: Credit Evaluation Ensemble Model with Self-Contained Shunt. 2019 5th International Conference on Big Data and Information Analytics (BigDIA), 2019, pp. 59–65, doi: 10.1109/bigdia.2019.8802679.

[36] Li, J.—Li, X.—Dai, D.—Ruan, S.—Zhu, X.: Research on Credit Risk Measurement of Small and Micro Enterprises Based on the Integrated Algorithm of Improved GSO and ELM. Mathematical Problems in Engineering, Vol. 2020, 2020, Art. No. 3490536, doi: 10.1155/2020/3490536.

[37] Jain, V.—Agrawal, M.—Kumar, A.: Performance Analysis of Machine Learning Algorithms in Credit Cards Fraud Detection. 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2020, pp. 86–88, doi: 10.1109/icrito48877.2020.9197762.



Renjing Liu received his graduation degree from the Xinjiang University of Mathematics, Xinjiang, China, in 1987. He is Professor in the Xi'an Jiaotong University, Xi'an China. His current research interests include artificial intelligence, complex system management, multi project management, risk management, business intelligence.

Xuming Yang received his graduation degree from the Chongqing University of Industrial Engineering, Chongqing, in 2019. He is a graduate student of Xi'an Jiaotong University, Xi'an, China. His current research interests include information management and business intelligence.

Xinyu Dong is a graduate student of the Dietrich School of Arts and Sciences of University of Pittsburgh. Her current research interests include big data analysis and artificial intelligence.

Boyang Sun is a graduate student of the School of Management of Xi'an Jiaotong University, Xi'an, China. His current research interests include information management and data mining.

Computing and Informatics, Vol. 40, 2021, 169–194, doi: 10.31577/cai_2021_1_169

ASSESSMENT OF THE VIABILITY OF A BIOMETRIC CHARACTERISTIC IN THE CONTEXT OF BIOMETRIC AUTHENTICATION ON MOBILE DEVICES

Piotr Nawrocki, Wojciech Kubaty

AGH University of Science and Technology
Faculty of Computer Science, Electronics and Telecommunications
Institute of Computer Science
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: piotr.nawrocki@agh.edu.pl, wkubaty@gmail.com

Abstract. The issue of safe utilization of mobile devices is becoming an increasingly important problem, among others due to the widespread use of such devices to access sensitive data (such as electronic documents or banking data). In our work we analyze the use of biometric techniques in order to secure a mobile device, with particular emphasis on the viability of selected biometric characteristics. For this purpose, we investigate the possibility of applying machine learning models to assess the authenticity of a biometric characteristic. Results of our tests have shown that the most effective method of assessing the viability of a biometric characteristic involves blink and smile detection.

Keywords: Biometrics, viability of a biometric characteristic, face recognition, biometric authentication, mobile device

1 INTRODUCTION

Due to the widespread use of mobile devices, it is becoming more and more important to ensure their appropriate level of security. In particular, it is important to protect the identity of the mobile device user, including protection of personal information, banking data or other sensitive data. For this purpose, various security methods have been developed over the years. The most common method of

securing access to a mobile device is a 4-character PIN password. However, with the ongoing technical progress (including machine learning methods, and, in particular, data processing by deep neural networks), biometric methods are more and more frequently applied to secure access to mobile devices, including, primarily, the facial recognition method. This method must work in real time; moreover, it should work reliably in various environment conditions and despite changes in the appearance of the face itself. Therefore, with the growing popularity of biometric methods that provide high accuracy, scalability and system security, it becomes important to ensure that the biometric characteristic used for authorization remains authentic. The problem of assessing the viability of a biometric characteristic, making sure that the characteristic comes from a real and physically present person, is complex and not yet fully explored. Therefore, in our article, we focus on this issue and examine the effectiveness of various methods of assessing the viability of biometric characteristics.

The structure of the article is as follows: Section 2 presents an overview of research in the field of detecting attacks on the credibility of a biometric characteristic; Section 3 describes evaluation of methods of assessment of the viability of a biometric characteristic; Section 4 presents the results of experiments and Section 5 contains conclusions.

2 RELATED WORK

A biometric system should not only be able to correctly recognize the face, but also be resistant to various types of attacks. The assessment of the viability of a biometric characteristic consists in making sure that the characteristic considered by the system comes from a real person who is physically present at the site of the given activity. Due to the large variety of attacks, the problem of detecting malicious access has not yet been fully resolved. The following types of attacks can be distinguished:

1. static – using a printed or displayed target;
2. dynamic – using facial recording, often with changing perspective or expressions;
3. using 3D masks [5] – ranging from the simplest paper masks to very detailed ones made of resin or silicone.

Given the wide range of potential attacks, many solutions have been developed to detect them. The focus was mostly on detecting only one type of attack. Initially, hand-designed methods were used, such as LBP [18, 4] and its various modifications [6, 24]. Recently, neural networks have been gaining popularity. They perceive features of the image that are difficult to describe using simple algorithm. There are many ways to detect an attack, differing mainly in the degree of interaction with the user. An ideal biometric system would be able to detect an attack attempt in a very short time and without any user interaction. Unfortunately, such

methods require additional sensors, which mobile devices are not always equipped with.

The simplest solutions rely on a single photo showing the the user's face and make the appropriate classification by analyzing this image. Another group of solutions is based on the analysis of a sequence of consecutive photos. We can distinguish solutions based on unintentional facial or camera movements, and those that require the user to perform the prescribed motion. The former group is not very invasive, but it does not provide as much information as a forced interaction. By forcing specific movement larger changes are produced between consecutive images; what is more, an attack becomes more difficult to carry out. The following ways of recognizing attacks can be distinguished: analyzing facial expression, analyzing the 3D structure and mimicry of the face, and using image texture analysis.

Utilization of blinking as a simple method of assessing facial vitality was discussed in [11] and [21]. Furthermore, [8] presents a method of assessing vitality by analyzing unintended eye movements and blinking. When eyes are detected, the image is normalized and successive frames of the image sequence are compared. If the differences are large enough, the characteristic is classified as alive.

The possibility of using lip movement analysis while uttering a given phrase was tested in [13], focusing on the correlation between the assumed utterance and the dynamics of the mouth movement. Due to their simplicity and feasibility of implementation on mobile devices, methods based on analysis of the degree of eye closure and smile were tested in this study.

The 3D analysis group includes *Optical Flow* analysis described in [3], where it was noticed that the movement of objects can be divided into four types: rotation, displacement, resizing and changing perspective. The first three types are common to 2D and 3D objects, while the final type is specific only to 3D objects – thus, analysis of such changes may allow us to detect attacks. The paper discussed the idea of using light neural networks to classify Optical Flow maps. This method, in conjunction with facial landmark detection, is also applied in [10]. It can be assumed that as the face moves, different parts of the face move differently. The model uses Gabor decomposition and an SVM classifier. This technique is also used in the method proposed in [12]. However, there is a high probability of rejecting a characteristic if it does not exhibit this kind of movement. In [28] a method has been proposed to detect characteristic points of the face in several photos taken from different angles, by forcing the user's head to move appropriately. The classifier assesses whether the arrangement of points is characteristic of a real face or whether it is an attack. This solution is independent of the user's phone model and environmental conditions, which has been proven by carrying out tests on several databases. However, the proposed approach requires some user engagement.

One of the first publications on image texture and component analysis is [16], where attacks involving a printed photo of a face using the Fourier transform are detected. This solution is based on two observations: the real image contains more high-frequency components than the false image, and even if the face is moving, the standard deviation in successive frames remains small. With the development of

printers and improvements in printing precision, this approach has become ineffective. The study of textures is a very popular and widely studied topic. In [4] and [18], the possibility of using LBP in various variants in static images was checked. The method consists in creating local histograms of differences in the values of neighboring pixels, using a window with a predetermined size, for example 3×3. It can divide the image into blocks and perform separate operations, and then combine the resulting histograms into one.

The solution proposed in [6] – Dynamic Textures – is an extension of LBP and is based on the analysis of microtextures in time and space. The best results have been obtained using a nonlinear SVM classifier. In [9] both solutions were tested – the discrete two-dimensional Fourier transform and texture detection via LBP, achieving the best results by combining these two methods. The research was conducted on attacks using paper masks, but the method used only one static image at a time. The computed values produced vectors that were used to train the SVM classifier. With the development of machine learning, solutions based on convolutional neural networks have emerged. An example of such a solution is [1], where one photo is required, and the developed method involves non-linear blurring of the image in such a way that the contours of the real face remain visible while the fake face disappears. The prepared images are then classified by the convolutional neural network.

Dataset	**Date**	**Number of Videos**	**Number of People**
NUAA	2010	(photos) 12 000	15
CASIA-FASD	2012	600	50
Replay-Attack	2012	1 200	50
MSU-USSA	2016	9 000	1 000
CASIA-SURF	2018	21 000	1 000
ROSE-Youtu	2018	3 350	20

Table 1. Comparison of datasets used for training and testing new attack detection solutions

There is a group of solutions which combine several methods, often using additional sensors, for example image depth [17], light reflections at different frequencies, or detection of the intensity of skin changes caused by blood flow [19, 17]. The recently created CASIA-SURF [30] database contains images of 1 000 people's faces in visible light, infrared and a depth map. On its basis, several works [22, 29, 26] have been published, with researchers reporting very high effectiveness. Different neural network architectures were compared with the best results for combining the three modalities. Unfortunately, such sensors are not widely available on mobile devices. HOG-based methods (*Histogram of Oriented Gradients*) were also used. In [14] information about the character's surroundings is used and attack detection is performed in a similar way to what an actual person would do – it points out if someone is trying to cheat the system by holding, for example, a photo of the face.

In [7], a comprehensive solution using only RGB images was proposed, using EfficientNet [27] networks and on MobileNetV2 [25], adding the last few layers in

such a way as to obtain a binary classification result on the output. It was shown that although the MobileNetV2-based model achieved slightly worse results, it also required a smaller number of network parameters (267 thousand, compared to the the EfficientNet-based model at over 5.5 million parameters), and therefore has greater potential for use on mobile devices. The ROSE-Youtu [15] training set, which was applied in this study, contains attempts of attacks by 20 people using both paper printouts and masks, as well as reconstructed recordings. In [2] the authors propose to combine two solutions. In the former (*patch-based*) random, small, local facial characteristics were examined, increasing the amount of data to train the model. The procedure enables the use of the full resolution of the characteristic image, as opposed to the holistic approach, which often scales the image, and thus forfeits some of its quality. The second aproach (*depth-based*) resulted in a method for creating a comprehensive face depth map, assuming that a real face has more depth than the flat characteristics used in attacks.

Several sets – CASIA-FASD [31], MSU-USSA [23] and Replay-Attack [4] – were used, producing 2.67 %, 0.35 % and 0.79 % EER (*Equal Error Rate*) respectively. Table 1 compares the datasets used to detect attacks.

3 ASSESSMENT OF THE VIABILITY OF A BIOMETRIC CHARACTERISTIC

In order to analyze the means of assessing the viability of a biometric characteristic, various methods were tested in this study, both specific to facial biometrics and enabling detection of attacks regardless of the biometric method used.

3.1 Assessment of Characteristic Viability on the Basis of Facial Movement in Three-Dimensional Space

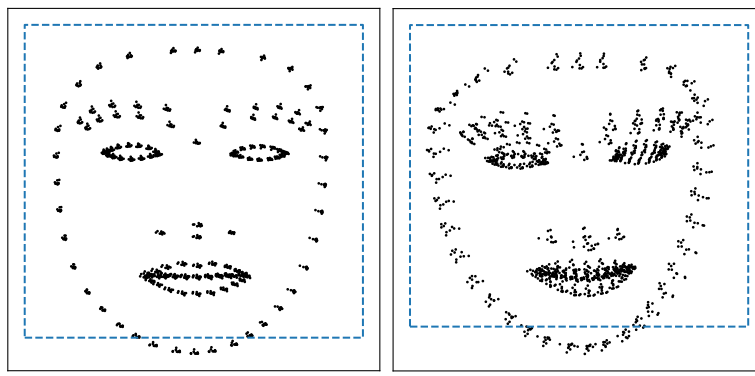
Due to the specificity of face recognition biometrics, methods involving analysis of successive frames of the recording, in particular the movement of characteristic points of the face, were proposed in this paper to assess the viability of the characteristic. The implementation, number and exact location of points may vary, but the most common ones include eyes, mouth and nose. The method proposed in this work bases on the observation that the movement of the face in three-dimensional space differs from the movement of a two-dimensional object, which is a photo of a face printed or displayed on an electronic device. Evaluation of changes in the grid of characteristic points can be performed by analyzing the distance measure. For two matrices, A_1 and A_2, representing the distances between each two characteristic points of the face, the distance is given by the formula:

$$M = \left|\left| \frac{A_1}{||A_1||_F} - \frac{A_2}{||A_2||_F} \right|\right|_F \tag{1}$$

where $||\cdot||_F$ is the Frobenius norm. For the A matrix with dimensions of $m \times n$ it assumes the form:

$$||A||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}. \tag{2}$$

For two identical grids of characteristic points, the value of the M distance will be zero. The value increases along with an increase in differences between the grids. In fact, lack of precision in detecting the characteristic points causes slight changes in distance for meshes which retain similarity[1], For example, by changing the position in the frame, zooming and rotating it, the value will be close to zero. Rotation of the head changes the position of the characteristic points in relation to each other, which implies an increase in distance. Figure 1 shows the position of the characteristic points of the face for two ranges of motion, taking into account 10 successive frames of the recording.



a) Minor head movements b) Significant head movements

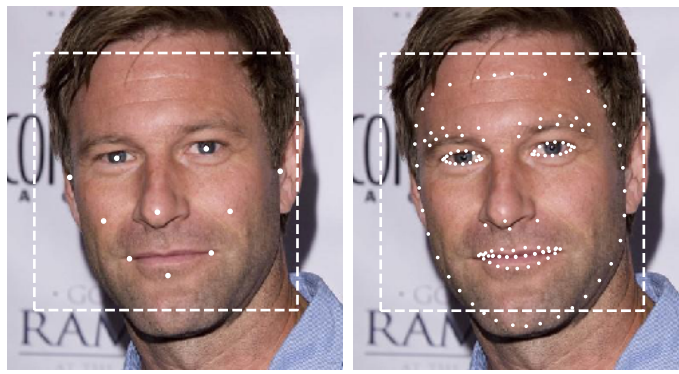
Figure 1. Comparison of the relative coordinates of 131 characteristic points of the face. The edge of the detected face is marked with a dashed line.

The NUAA Imposter DB[2] was used to check the value of the distance for faces that do not change position, as well as faces which exhibit movement. Each recording in the database consists of a sequence of between several dozen and several hundred frames, and within each sequence all frames show the same person. The recordings contain presentation attacks involving various transformations of printed photographs and sequences of real people. The attack set was divided into a part which contains similarity transforms, and a part which involves changes in perspective and bending of the photo.

[1] Similarity – a geometric transformation that maintains the ratio of the distance between points.

[2] NUAA Imposter DB set – `http://parnec.nuaa.edu.cn/xtan/data/NUAAImposterDB.html`

Each photo was subjected to face detection and extraction of characteristic points on a mobile device, resulting in a list of coordinates relative to the upper left corner of the detected face. The *Firebase ML Kit*[3] library was used for this purpose. It enables the use of two characteristic point detection algorithms which return 10 or 131 face characteristic points respectively.



a) b)

Figure 2. Photos of faces with 10 and 131 landmarks respectively

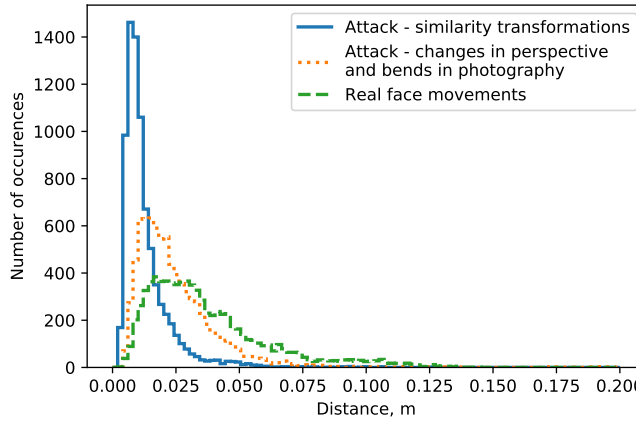
Figure 2 shows the points used in both methods. The positions of some facial features (mainly the ears) which are not directly visible, were approximated. Their positions in subsequent frames differ significantly, so only the remaining eight points were used in further studies. Moreover, in the second drawing (131 points) poor precision of the facial contours can be observed. These contours are probably averaged over existing points rather than independently detected.

To calculate the M distance for data from the NUAA database, both face characteristic point detection algorithms from the Firebase ML Kit library were used. For each frame in the sequence characteristic points of the face were generated. Subsequently, a matrix of mutual distances between each pair of characteristic points was computed. Following final transformations, based on the (1) formula, a distance measure was obtained, expressed as a single numerical value.

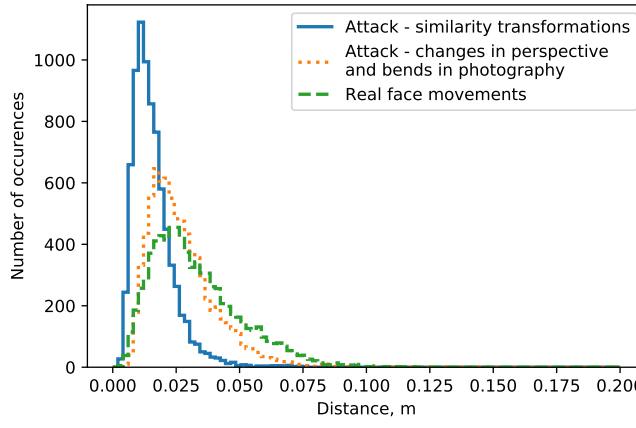
Figure 3 presents the distance histograms according to the M distance for both face characteristic point detection algorithms and two types of attacks: the first one based solely on the similarity of still photos and the second one based on distortions and projective transformations. Additionally, the movement distance was included for real subjects. The lowest value of M corresponds to the distortion-free set, while for attacks carried out with distortions and actual subjects very similar values were obtained. Moreover, no significant differences were noticed between the algorithms

[3] Firebase ML Kit library – `https://firebase.google.com/docs/ml-kit/detect-faces`

for detecting characteristic points of the face; thus, further analyses involved the algorithm which returns fewer points.



a) 8 points



b) 131 points

Figure 3. Histogram showing the M distance for the NUAA set for two face characteristic point detection algorithms

Based on the above distance measure, the theoretical classification abilities of the logistic regression model to distinguish attacks from real subjects were tested. Figure 4 a) shows the ROC curve (*Receiver Operating Characteristic*). It takes into account the relationship between FPR[4] and TPR[5] The expected ROC curve should be more convex and the surface area beneath it as large as possible. In this case, the

[4] FPR (*False Positive Rate*) – the percentage of wrong confirmations
[5] TPR (*True Positive Rate*) – sensitivity or true positive percentage

ROC curve shows that only presentation attacks in which there is no facial movement can be detected relatively well. Unfortunately, for this collection, the differences in the movement of the characteristic points of real people's faces in relation to the distortions of photographs are too small to be able to clearly separate the classes from each other.

Due to the lack of unambiguous results assessing the effectiveness of the distance and the shortage of data sets of appropriate size and quality, which would include significant facial movement within one sequence of photos, it was decided to use a data set not specialized for this purpose. The YouTube Faces database[6] containing recordings of 1 595 different people (over 600 000 photos) was used. In order to divide the set into sequences containing significant movement and those containing slight movement, each photo in the sequence was analyzed for facial rotation. The FSA-NET[7] was used to assess the face rotation angle, resulting in a three-dimensional face rotation vector. The threshold for considering facial movement as significant was approximated based on the previously described NUAA dataset, which also assessed the facial rotation angle.

Table 2 presents statistics of the NUAA set for attacks using similarity-based transformations. It has been shown that changes in position in relation to the X and Y axes are relatively small and most do not exceed 7° along any of the axes. The X axis corresponds to the absolute value of face rotation vertically, while the Y axis corresponds to horizontal rotation.

	Face rotation angle for the NUAA set	
	X axis	Y axis
Average	1.97	2.29
Standard deviation	1.76	1.68
50 %	1.29	2.06
75 %	3.36	3.61
99 %	6.55	5.48

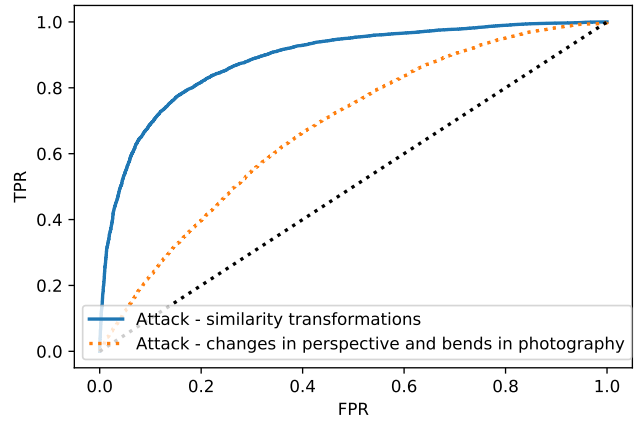
Table 2. Changes in the angle of face rotation for the NUAA dataset

Due to the small characteristic size, frames with a minimum 10° change in the face rotation angle were included in the distinct movement collection. Sequences where movement fell below this value were treated as attack simulations, reflecting the instability and imprecision of facial landmark detection models.

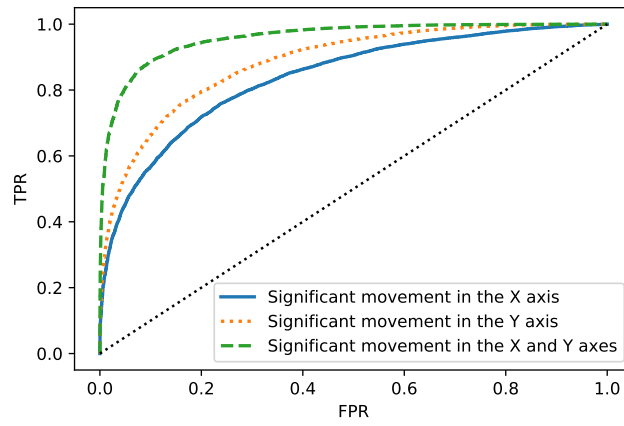
Table 3 summarizes the ROC curve parameters for the logistic regression classifier. Its graph is shown in Figure 4 b). The most significant movement (at least 10° along the X and Y axes) corresponded to the highest average distance values and therefore the best separation between this set and the set where there was no significant movement. For this reason, the EER error was the smallest for this class, equalling 4.99 %. It follows that the combination of facial movement along two axes

[6] YouTube Faces database – `https://www.cs.tau.ac.il/~wolf/ytfaces/`

[7] FSA-NET network – `https://github.com/shamangary/FSA-Net`



a) NUAA dataset



b) YT Faces dataset

Figure 4. ROC curves for NUAA and YT Faces datasets

may be the best indicator of the viability of the characteristic, but may also be the least convenient for the user to perform. Figure 5 presents a histogram showing the M distance values for the YT Faces set.

	X Axis Movements	**Y Axis Movements**	**X & Y Axes Movements**
EER	19.17 %	9.08 %	4.99 %
AUC[8]	0.88	0.97	0.99

Table 3. Efficiency of classification using logistic regression

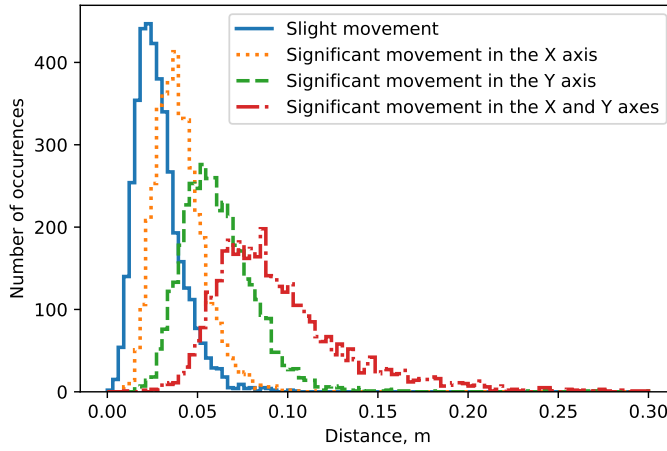


Figure 5. Histogram showing the M distance for the YT Faces set

3.2 Simulating Face Movement by Changing Perspective

Facial movement can also be simulated by changing the perspective under which the 2D characteristic is presented, thus simulating facial rotation. In order to create a model capable of recognizing such attacks a new set of sequences was generated providing it is a faithful representation of real-world attacks. For this purpose, the Facescrub[9] collection was used [20]. For each subject, on the basis of their single photo and projective transformation, new images were created, imitating changes in the angle at which the false characteristic is presented. Each photo was transformed along the X axis, Y axis and both axes simultaneously. The corresponding edges of the photo were enlarged in one of three scales: 120 %, 160 % and 200 %. An example of a Y-axis transformation simulating horizontal face rotation is presented in Figure 6.

The effectiveness of face detection was checked along with its characteristic points depending on the degree of transformation. Results are presented in Table 4. It can be seen that as the degree of transformation increases, the effectiveness of face detection decreases. Moreover, for such transformations the angle of face rotation predicted by the FSA-NET network was also checked. As the degree of transformation increases, the projected angle of rotation increases. The M distance value for the original photo and the generated transformation was analyzed, with the corresponding histograms presented in Figure 7. Results indicate that the distance value may exceed the value obtained when comparing faces under natural movement.

Additionally, the similarity of the transformed photos to the originals was checked by applying the previously described facial recognition model. Figure 8 presents

[9] Facescrub DB set – `http://vintage.winklerbros.net/facescrub.html`

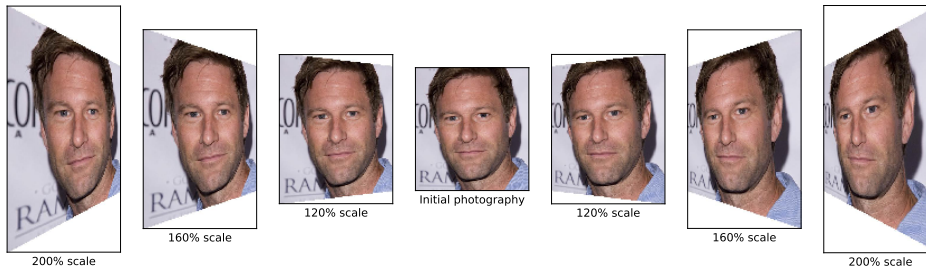
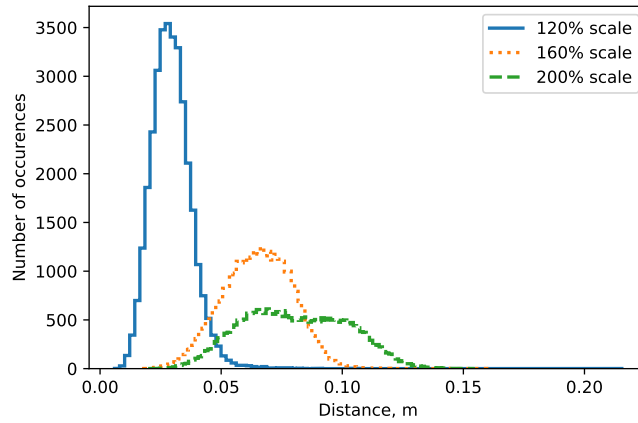
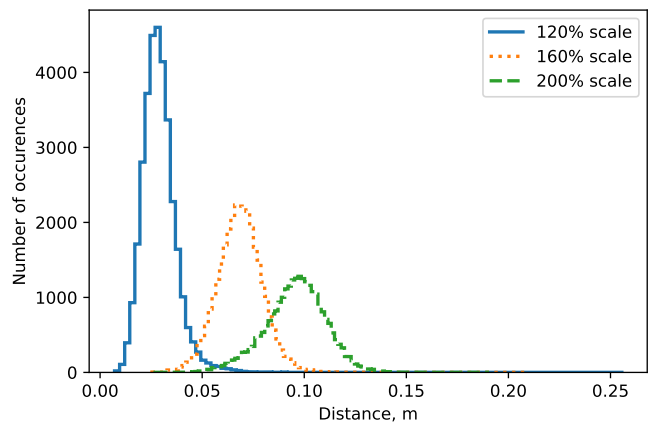


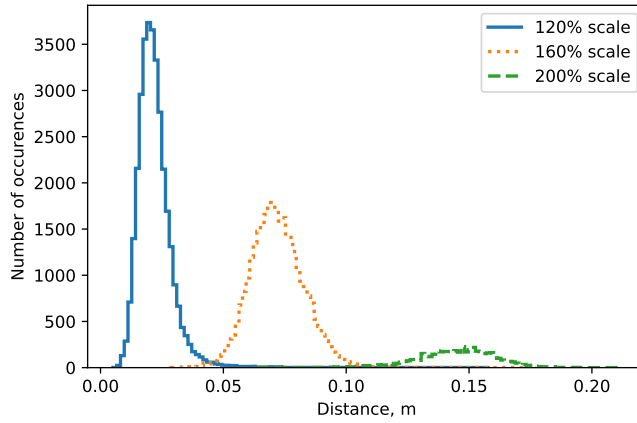
Figure 6. An example of a generated sequence simulating an attack using a face photo perspective change



a) X axis



b) Y axis



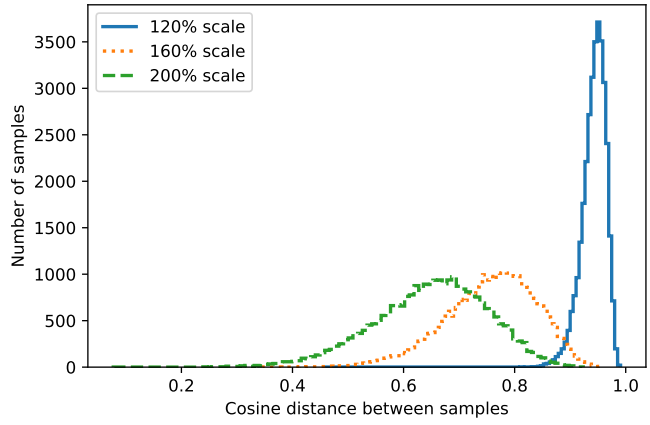
c) X and Y axes

Figure 7. Histograms showing the M distance as a function of the degree of projective transformation

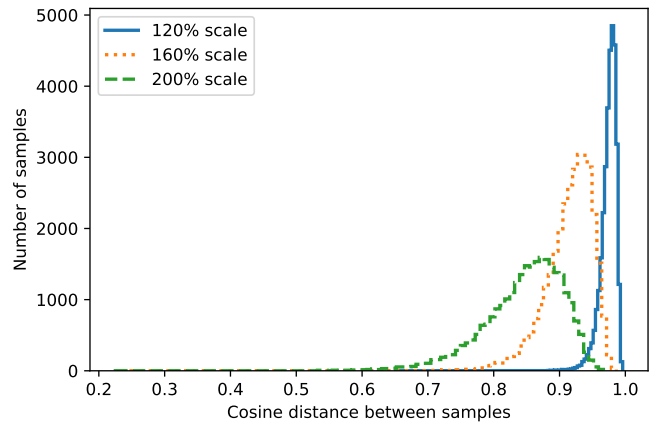
Transform		Average Rotation Angle [°]		Average Distance M Value	Face Detection Efficiency [%]
Axis	Scale [%]	X	Y		
X	120	1.16	3.15	0.029	98.24
	160	4.50	2.52	0.065	96.11
	200	7.16	2.89	0.080	79.53
Y	120	2.22	3.04	0.022	98.86
	160	1.80	6.64	0.068	93.58
	200	2.58	9.51	0.095	78.19
X/Y	120	1.93	1.70	0.029	98.80
	160	7.39	5.96	0.071	92.12
	200	16.75	15.74	0.144	14.64

Table 4. Comparison of rotation angle, average distance value, and face detection accuracy for different projective transforms

the statistical distribution of the cosine distance between the compared characteristics. It is evident that as the transformation scale increases, the distance and thus similarity both decrease. Nevertheless, for the previously determined threshold, the vast majority of characteristics are still positively classified. This means that the facial recognition module is not very sensitive to changes of perspective under which a potential false characteristic is presented, but this has a significant impact on the selection of the method for assessing the viability of the characteristic.



a) X axis



b) Y axis

Figure 8. Histograms showing the value of the cosine distance between the original photograph and the projective transformation at different scales and axes

3.3 Classification of the Movement of Characteristic Points of the Face Using Machine Learning Techniques

A high M distance value is not sufficient as an indicator of the viability of a given characteristic. It only reveals the magnitude of changes in between two sets of facial landmarks. The greater the distance value, the more significant the facial movement, but it is not known whether this is caused by actual changes in the appearance of the subject or merely by distortions and changes in perspective. The corresponding assessment can be performed through more detailed analysis of the movement of characteristic points of the face, using machine learning methods. The previously

generated projective transformations and sequences from the YT Faces database were used to classify the face of a real subject and to attack the presentation. Instead of calculating a one-dimensional distance value, a matrix of differences in the distance of characteristic points of the face was used to classify the movement. Four machine learning models from the *scikit-learn*[10] library were tested: two support vector machines (*LinearSVC* and *SVC* with *rbf* kernel), *GaussianNaiveBayes* algorithm, and the multilayer perceptron *MultiLayerPerceptron*. Classification accuracy was checked using the above models for two distance ranges, which may correspond to two scales of projective transformations (120 % and 160 % respectively). Percentage values were determined arbitrarily in the course of experiments and while they were not adjusted to distance ranges, they nevertheless correspond to them with high accuracy. Face detection accuracy dropped dramatically following 200 % projective transformation, and the accuracy of detecting attacks was very high; thus a realistic attack is regarded as unlikely.

Results are presented in Table 5. The best results were obtained for a nonlinear support vector machine. It has been shown that classification accuracy increases along with distance and thus with the scope of transformations. It is therefore easier to detect simulation of motion through perspective changes if the angle at which the characteristics are presented is larger. Therefore, classification should be performed only after exceeding a certain distance value (in this case – 0.05). Movement of the subject's face will therefore have to be significant enough to exceed the predefined threshold, permitting accurate analysis.

Classifier	**Accuracy**	
	$M \in (0.01; 0.05)$	$M \in (0.05; 0.1)$
Linear SVM	85.19 %	90.03 %
Nonlinear SVM	90.76 %	96.90 %
Naive Bayes Classifier	85.14 %	86.61 %
Multilayer Perceptron	85.37 %	95.28 %

Table 5. Classification performance using different machine learning methods for different M distance values

3.4 Classification of Face Movement Using Optical Flow and Mobile Neural Networks

Classification of facial motion by detecting movement of characteristic points of the face requires the user to perform considerable movement with their head. However, for practical reasons, it is advisable to minimize such movement. For this reason, the applicability of a method based on Optical Flow maps was tested. This technique can be used to detect the direction of movement of objects between successive frames

[10] Scikit-learn – `https://scikit-learn.org/stable/`

in a video sequence. In this work, classification is done through a neural network, without assuming any particular motion model.

The data used to train neural networks came from the previously mentioned NUAA Imposter DB database. The Farneback algorithm from the OpenCV library was used to create Optical Flow maps. Among the algorithm's parameters there is the size of the analyzed window within which motion is sought. Window sizes of 10, 15 and 20 pixels were checked. For each pixel, the value and direction of movement are calculated. These values are converted to the HSV color space and then to RGB. Modified EfficientNet neural networks were used for classification. Dropout and Softmax layers were added to the base. Learning accuracy was checked using *Transfer Learning* (trained networks on ImageNet) and teaching the networks from scratch. Color pictures with dimensions of 224×224 formed the input to the network. 10-fold cross-validation was used to evaluate the effectiveness of neural network learning. Sequences from the initial set were divided into 10 disjoint parts. Nine of these comprised the training set and one formed the test set. The learning process was repeated ten times, with a different part used as the test set each time. Finally, the average learning success rate for the test set was measured.

Regardless of the value of the window, training the model from scratch was more effective. Moreover, the highest classification efficiency occurs for a window size of 15. Therefore, this value was chosen for further analysis.

In addition, in order to increase the amount and diversity of data, less popular data sets were also used: the BioID[11], which contains 1 521 photos of 23 subjects, and Kaggle DeepFake[12] from which 3 310 10-frame sequences were obtained. The attack collection was also expanded with Optical Flow maps generated on the basis of 40 customized videos displayed on the monitor. Increasing the size of the training sets resulted in increased stability of classification and enabled the authors to forgo cross-validation. Sequences from the initial set were split into 90/10 subsets, with 90 % of data forming the training set and the remaining 10 % used as the test set. The effectiveness of classification was checked both with the use of *Transfer Learning* and by training the network from scratch. As before, classification efficiency turned out to be better for networks trained from scratch.

Table 6 summarizes the effectiveness of classification for different variants of network training. Although training a network from scratch increases the required time, it does not affect the response time of the model on a mobile device. More important is the effectiveness of classification, which turned out to be higher when the network was trained from scratch – which is why this neural network model was used for further tests on mobile devices.

[11] BioId database – `https://www.bioid.com/facedb/`

[12] Kaggle DeepFake DB – `https://www.kaggle.com/c/deepfake-detection-challenge`

Dataset	**Accuracy**	
	Transfer Learning	Without Transfer Learning
NUAA (window = 10)	83.85 %	95.55 %
NUAA (window = 15)	87.46 %	95.55 %
NUAA (window = 20)	84.59 %	93.61 %
Combination of datasets (window = 15)	95.58 %	98.94 %

Table 6. Effectiveness of classification of Optical Flow maps of the EfficientNet network using *Transfer Learning* and training the network from scratch

4 EVALUATION

For practical analysis of the issue of assessing the viability of a biometric characteristic, we have developed a dedicated mobile application used in experiments. Models whose training is described in the previous chapter have been used.

4.1 Implementation

User authentication consists of the facial recognition process and the characteristic viability evaluation process. In a dedicated mobile application the user can choose a specific method of assessing the viability of the characteristic:

- method based on the analysis of the movement of characteristic points of the face (1);
- Optical Flow method using vertical or horizontal face rotation (2);
- Optical Flow method which involves bringing the lens closer to the face (3);
- method based on blink and smile detection (4).

In addition to a real-time preview of the camera image, the user's screen displays a message highlighting the need to perform a given action. Depending on the selected characteristic viability evaluation mode, the screen may also show additional information in graphic form, e.g. detected facial markings or indicators showing the angle of face rotation. The process of acquiring a face photo is automated. After analyzing a given frame, a decision is made whether the requested action has been successfully performed by the user. Upon acceptance, the frame is remembered and the status of the authentication process is updated. A corresponding message is also displayed to the user on the screen. After performing all actions, the collected frames are analyzed by the characteristic viability detection module and the face recognition module.

In the method based on the movement of characteristic points of the face (1), the user must perform any face rotation, e.g. a rotation to the right. Each frame of the recording is analyzed and the face is detected, followed by its characteristic points. A list of the most recent 10 frames is kept. For each pair of consecutive frames

Attempt	**Characteristic Authenticity Prediction** [%]		
	Genuine Characteristic	Characteristic Displayed on Electronic Device Screen	Characteristic Printed on A4 Paper Sheet
1	69.49	11.35	9.44
2	45.91	17.21	2.69
3	51.44	11.31	19.05
4	29.41	5.31	5.79
5	56.81	8.38	2.60
6	78.64	24.15	7.43
7	51.58	7.95	6.67
8	25.05	7.89	22.54
9	21.49	11.81	3.64
10	63.08	18.48	11.08
Average	**49.29**	**12.38**	**9.09**

Table 7. Characteristic authenticity prediction – method based on changing the distance from the tested object

containing facial landmarks, the distance is calculated according to the M distance formula (described in the previous section). If the value of the metric exceeds 0.05 for at least 10 pairs of frames, classification is performed. The decision to accept a characteristic is made by the two classifiers which exhibit the best training efficiency, i.e. the nonlinear classifier *SVM* and the multilayer perceptron *MLP*. The classifiers were trained in Python using the scikit-learn library and converted using the *sklearn-porter*[13]. Appropriate parameters necessary for initialization of classifiers are saved in the *json* format and loaded from the device's internal memory. A characteristic is classified positively if the mean value returned by both classifiers is positive ($> 50\,\%$).

For the method involving rotation of the face (2), the selection of appropriate frames used to create Optical Flow maps is performed by analyzing the face rotation angle. Depending on the selected mode, the *Firebase ML Kit* library is used to analyze the horizontal rotation angle, or *dlib* together with the *OpenCV* library to detect the vertical rotation angle. The rotation angle is analyzed in real time. A point indicator is displayed on the user's screen, showing the current degree of facial rotation, along with four target points. The points are arranged in a straight line vertically or horizontally, depending on the selected mode – two per side. By rotating the face, the user changes the position of the pointer. If it reaches the indicated target point, the frame is saved. After completing the task, Optical Flow maps are computed between frames using the *OpenCV* library. Subsequently, they are classified by the EfficientNet mobile neural network in *tflite* format using the *TensorFlow Lite* library. If the average of all images is above the set threshold, the viability test is accepted.

[13] sklearn-porter library – `https://github.com/nok/sklearn-porter`

Attempt	**Characteristic Authenticity Prediction** [%]		
	Genuine Characteristic	Characteristic Displayed on Electronic Device	Characteristic Printed on A4 Paper Sheet
1	72.85	43.84	27.00
2	43.55	51.30	23.29
3	55.66	13.60	35.91
4	43.12	54.39	9.45
5	77.12	9.47	23.27
6	21.32	24.10	18.95
7	68.72	22.59	38.78
8	68.97	44.94	46.04
9	46.31	43.60	26.69
10	79.49	22.20	40.98
Average	**57.71**	**33.03**	**29.04**

Table 8. Characteristic authenticity prediction – method based on horizontal face rotation

In order to minimize the need to rotate the face while ensuring that movement is registered between each two frames of the recording, the classification capabilities of Optical Flow maps were tested using the natural vibrations of the user's hand, perspective changes in the process of zooming in and out, and involuntary changes in facial expressions (3). On the device screen the currently detected face is marked with a frame, with rectangles depictng two target frames – a smaller one and a larger one. The larger square takes up approx. 80 % of the screen width, while the smaller square takes up approx. 60 %. The user's task is to move the mobile device in such a way that the detected face is inside the smaller and outside the larger rectangle respectively. Once the requirements are met, the frames are saved for further analysis. In total, three frames are obtained in this method - one with the face fitting inside the smaller rectangle, the second when the face protrudes beyond the larger rectangle, and the third in between. An Optical Flow map is computed for each pair of consecutive frames and then classified in the same way as for the facial rotation method.

Contrary to the previous methods, which require face rotation or close-up, method (4) analyzes changes in facial expressions. The user is asked to blink first and then to smile. Detection of these activities is performed using the *Firebase ML Kit* library, which assesses the likelihood of a blink and of a smile for each eye. If an action occurs in the analyzed frame, it is saved for further analysis. Appropriate information about the required next action is displayed at the top of the screen.

4.2 Experiments

For each method of assessing the viability of a characteristic and the type of attack, 10 measurements were performed. Table 7 presents the characteristic authenticity

Attempt	**Characteristic Authenticity Prediction** [%]	
	Genuine Characteristic	Characteristic Printed on A4 Paper Sheet
1	67.1	40.43
2	51.12	28.21
3	53.06	27.83
4	59.17	16.18
5	46.19	8.05
6	40.83	31.19
7	21,53	23.99
8	58.74	14.5
9	76.61	35.14
10	88.71	14.01
Average	**56.31**	**23.95**

Table 9. Characteristic authenticity prediction – method based on the vertical face rotation

assessment for the method based on changes in distance from the tested object. The greater the value, the higher the likelihood that the characteristic is genuine, while lower values suggest an attempted attack. It can be seen that the attacks obtained a much lower mean value than real characteristics, which confirms the statistical effectiveness of the model. However, the tendency of the model to reduce the likelihood that the characteristic is authentic can also be noticed.

Table 8 shows the results of characteristic authenticity prediction for the horizontal face rotation method. The mean value for all three test modes turned out to be greater than for the method based on changes in distance from the test object. In this case, facial movement was more significant, suggesting greater authenticity of the characteristic. In addition, the true characteristic has a greater authenticity estimate than the corresponding attack attempts. Unfortunately, in some cases the authenticity estimate of the counterfeit characteristic is greater than that of the genuine characteristic.

The characteristic authenticity prediction for the vertical face rotation method is shown in Table 9. The test was performed only for the real characteristic and for an attack which exploits a printed photograph. It was not possible to perform tests on an electronic device in this form. The vertical face rotation angle evaluation algorithm used on a mobile device turned out to be unable to detect faces merely via changes in perspective. The flexible nature of paper, however, allowed for more extensive manipulation and enabled a successful initial test. Paradoxically, this turns out to be an effective test of the authenticity of a characteristic displayed on a mobile device.

The characteristic authenticity prediction for the method based on blink and smile detection is shown in Table 10. The attack could not be successfully performed with a single face image, therefore only the true characteristic is included in this

Attempt	**Characteristic Authenticity Prediction** [%]
	Genuine Characteristic
1	84.42
2	66.49
3	77.82
4	72.85
5	67.21
6	70.32
7	82.55
8	62.59
9	95.28
10	76.88
Average	**75.64**

Table 10. Characteristic authenticity prediction – method based on blink and smile detection

table. The reported values turned out to be the highest among all methods, which suggests that changes in facial expressions are the most significant for the method using Optical Flow maps. In all cases, both a face with closed eyes and a smiling face were correctly identified.

Assessment of characteristic authenticity for a method based on analysis of the movement of characteristic points of the face, using the *SVM* classifier and the *MLP* classifier, is presented in Table 11. As can be seen, in most cases the classifiers returned results which were definitive, but divergent from each other.

Attempt	**Characteristic Authenticity Prediction** [%]					
	Genuine Characteristic		Characteristic Displayed on Electronic Device		Characteristic Printed on A4 Paper Sheet	
	SVM	MLP	SVM	MLP	SVM	MLP
1	60	80	0	0	0	0
2	100	100	0	0	0	40
3	100	100	20	20	0	0
4	100	100	0	0	0	0
5	100	100	0	100	10	20
6	0	100	0	0	0	0
7	100	100	10	10	0	0
8	100	100	0	0	0	0
9	100	100	80	60	10	20
10	100	100	40	30	0	0
Average	**86**	**98**	**15**	**22**	**2**	**8**

Table 11. Characteristic authenticity prediction – method based on the analysis of facial landmark movements

Method	Prediction Accuracy [%]		
	Genuine Characteristic Acceptance	Rejection of the Characteristic Displayed on Electronic Device	Rejection of the Characteristic Printed on A4 Paper Sheet
Distance change	60	100	100
Horizontal face rotation	60	80	100
Vertical face rotation	70	–	100
Blink and smile	100	–	–
Facial landmark movement analysis	90	100	90

Table 12. Summary of accuracy of characteristic authenticity prediction for all tested methods

A summary of the performance of all tested methods is presented in Table 12. The effectiveness with which the authenticity of a real characteristic is confirmed and false characteristics detected was calculated under the assumption that for methods based on Optical Flow, the input is considered genuine if the average value is at least 50 %. In turn, for the method based on the analysis of the movement of characteristic points of the face, the average value returned by both classifiers, i.e. *SVM* and *MLP*, must be at least 0.5. All tested methods showed good effectiveness. Assuming that only one photo of the face is used for the attack, the most effective method involved detection of blinks and smiles. Presentation attacks cannot be carried out without photo manipulation. In this method, the use of Optical Flow maps does not yield any additional benefits, but can be applied when an attack exploits a paper mask with cutouts for the mouth and eyes.

5 CONCLUSION

In this work we focused on the issue of assessing the viability of a biometric characteristic on a mobile device – focusing on the possibility of using machine learning models to assess the authenticity of such a characteristic. For this purpose, we analyzed the effectiveness of detecting the viability of a biometric characteristic during attacks using a single photograph of the subject's face. Our analysis and experiments indicate that the effectiveness of assessment is largely influenced by the hardware properties of mobile devices. On the one hand, such devices have various types of sensors that can be used to assess the viability of a biometric characteristic; however, on the other hand, limited hardware resources (such as memory, CPU power or screen space) may affect the convenience and efficiency with which biometric characteristic viability assessment is performed, especially using complex machine learning algorithms. As a result of the presented tests, we concluded that the most effective method of assessing the viability of a biometric characteristic is

the one which involves blink and smile detection. However, each of the analyzed methods (including the most effective one) was susceptible – to some extent – to attack. Foolproof attack detection and unquestionable assessment of the viability of a biometric characteristic are impossible to achieve; however, any proposed method used should be accurate enough to effectively discourage potential attack attempts.

Acknowledgements

The research presented in this paper was supported by funds from the Polish Ministry of Science and Higher Education allocated to the AGH University of Science and Technology. Wojciech Kubaty's work was supported in part by the National Center for Research and Development (NCBR) under Grant No. CYBERSECIDENT/382354/II/NCBR/2018.

REFERENCES

[1] Alotaibi, A.—Mahmood, A.: Deep Face Liveness Detection Based on Nonlinear Diffusion Using Convolution Neural Network. Signal, Image and Video Processing, Vol. 11, 2017, No. 4, pp. 713–720, doi: 10.1007/s11760-016-1014-2.

[2] Atoum, Y.—Liu, Y.—Jourabloo, A.—Liu, X.: Face Anti-Spoofing Using Patch and Depth-Based CNNs. 2017 IEEE International Joint Conference on Biometrics (IJCB), 2017, pp. 319–328, doi: 10.1109/btas.2017.8272713.

[3] Bao, W.—Li, H.—Li, N.—Jiang, W.: A Liveness Detection Method for Face Recognition Based on Optical Flow Field. 2009 International Conference on Image Analysis and Signal Processing, IEEE, 2009, pp. 233–236, doi: 10.1109/iasp.2009.5054589.

[4] Chingovska, I.—Anjos, A.—Marcel, S.: On the Effectiveness of Local Binary Patterns in Face Anti-Spoofing. Proceedings of the International Conference of Biometrics Special Interest Group (BIOSIG), IEEE, 2012, pp. 1–7.

[5] Erdogmus, N.—Marcel, S.: Spoofing in 2D Face Recognition with 3D Masks and Anti-Spoofing with Kinect. 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS), pp. 1–6, doi: 10.1109/btas.2013.6712688.

[6] de Freitas Pereira, T.—Komulainen, J.—Anjos, A.—De Martino, J. M.—Hadid, A.—Pietikäinen, M.—Marcel, S.: Face Liveness Detection Using Dynamic Texture. EURASIP Journal on Image and Video Processing, Vol. 2014, 2014, No. 1, Art. No. 2, doi: 10.1186/1687-5281-2014-2.

[7] Ghofrani, A.—Toroghi, R. M.—Tabatabaie, S. M.: Attention-Based Face AntiSpoofing of RGB Images, Using a Minimal End-2-End Neural Network. 2019, arXiv: 1912.08870.

[8] Jee, H. K.—Jung, S. U.—Yoo, J. H.: Liveness Detection for Embedded Face Recognition System. World Academy of Science, Engineering and Technology, International Journal of Computer and Information Engineering, Vol. 2, 2008, No. 6, pp. 2142–2145, doi: 10.5281/zenodo.1060812.

[9] Kim, G.—Eum, S.—Suhr, J. K.—Kim, D. I.—Park, K. R.—Kim, J.: Face Liveness Detection Based on Texture and Frequency Analyses. 2012 5th IAPR International Conference on Biometrics (ICB), IEEE, 2012, pp. 67–72, doi: 10.1109/icb.2012.6199760.

[10] Kollreider, K.—Fronthaler, H.—Bigun, J.: Evaluating Liveness by Face Images and the Structure Tensor. 4th IEEE Workshop on Automatic Identification Advanced Technologies (AutoID '05), 2005, pp. 75–80, doi: 10.1109/autoid.2005.20.

[11] Kollreider, K.—Fronthaler, H.—Bigun, J.: Verifying Liveness by Multiple Experts in Face Biometrics. 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008, pp. 1–6, doi: 10.1109/cvprw.2008.4563115.

[12] Kollreider, K.—Fronthaler, H.—Bigun, J.: Non-Intrusive Liveness Detection by Face Images. Image and Vision Computing, Vol. 27, 2009, No. 3, pp. 233–244, doi: 10.1016/j.imavis.2007.05.004.

[13] Kollreider, K.—Fronthaler, H.—Faraj, M. I.—Bigun, J.: Real-Time Face Detection and Motion Analysis with Application in "Liveness" Assessment. IEEE Transactions on Information Forensics and Security, Vol. 2, 2007, No. 3, pp. 548–558, doi: 10.1109/tifs.2007.902037.

[14] Komulainen, J.—Hadid, A.—Pietikäinen, M.: Context Based Face Anti-Spoofing. 2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS), 2013, pp. 1–8, doi: 10.1109/btas.2013.6712690.

[15] Li, H.—Li, W.—Cao, H.—Wang, S.—Huang, F.—Kot, A. C.: Unsupervised Domain Adaptation for Face Anti-Spoofing. IEEE Transactions on Information Forensics and Security, Vol. 13, 2018, No. 7, pp. 1794–1809, doi: 10.1109/tifs.2018.2801312.

[16] Li, J.—Wang, Y.—Tan, T.—Jain, A. K.: Live Face Detection Based on the Analysis of Fourier Spectra. Biometric Technology for Human Identification, Proceedings of the SPIE, Vol. 5404, 2004, pp. 296–303, doi: 10.1117/12.541955.

[17] Liu, Y.—Jourabloo, A.—Liu, X.: Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 389–398, doi: 10.1109/cvpr.2018.00048.

[18] Määttä, J.—Hadid, A.—Pietikäinen, M.: Face Spoofing Detection from Single Images Using Micro-Texture Analysis. 2011 International Joint Conference on Biometrics (IJCB), IEEE, 2011, pp. 1–7, doi: 10.1109/ijcb.2011.6117510.

[19] Nowara, E. M.—Sabharwal, A.—Veeraraghavan, A.: PPGSecure: Biometric Presentation Attack Detection Using Photopletysmograms. 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), 2017, pp. 56–62, doi: 10.1109/FG.2017.16.

[20] Ng, H. W.—Winkler, S.: A Data-Driven Approach to Cleaning Large Face Datasets. 2014 IEEE International Conference on Image Processing (ICIP), 2014, pp. 343-347, doi: 10.1109/icip.2014.7025068.

[21] Pan, G.—Sun, L.—Wu, Z.—Lao, S.: Eyeblink-Based Anti-Spoofing in Face Recognition from a Generic Webcamera. 2007 IEEE 11th International Conference on Computer Vision, 2007, pp. 1–8, doi: 10.1109/iccv.2007.4409068.

[22] Parkin, A.—Grinchuk, O.: Recognizing Multi-Modal Face Spoofing with Face Recognition Networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 1617–1623, doi: 10.1109/cvprw.2019.00204.

[23] Patel, K.—Han, H.—Jain, A. K.: Secure Face Unlock: Spoof Detection on Smartphones. IEEE Transactions on Information Forensics and Security, Vol. 11, 2016, No. 10, pp. 2268–2283, doi: 10.1109/tifs.2016.2578288.

[24] Patel, K.—Han, H.—Jain, A. K.—Ott, G.: Live Face Video vs. Spoof Face Video: Use of Moiré Patterns to Detect Replay Video Attacks. 2015 International Conference on Biometrics (ICB), IEEE, 2015, pp. 98–105, doi: 10.1109/icb.2015.7139082.

[25] Sandler, M.—Howard, A.—Zhu, M.—Zhmoginov, A.—Chen, L. C.: MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520, doi: 10.1109/cvpr.2018.00474.

[26] Shen, T.—Huang, Y.—Tong, Z.: FaceBagNet: Bag-of-Local-Features Model for Multi-Modal Face Anti-Spoofing. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 1611–1616, doi: 10.1109/cvprw.2019.00203.

[27] Tan, M.—Le, Q. V.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In: Chaudhuri, K., Salakhutdinov, R. (Eds.): Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, Proceedings of Machine Learning Research, Vol. 97, 2019, pp. 6105–6114, arXiv: 1905.11946.

[28] Wang, T.—Yang, J.—Lei, Z.—Liao, S.—Li, S. Z.: Face Liveness Detection Using 3D Structure Recovered from a Single Camera. 2013 International Conference on Biometrics (ICB), IEEE, 2013, pp. 1–6, doi: 10.1109/ICB.2013.6612957.

[29] Zhang, P.—Zou, F.—Wu, Z.—Dai, N.—Mark, S.—Fu, M.—Zhao, J.—Li, K.: FeatherNets: Convolutional Neural Networks as Light as Feather for Face Anti-Spoofing. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 1574–1583, doi: 10.1109/cvprw.2019.00199.

[30] Zhang, S.—Wang, X.—Liu, A.—Zhao, C.—Wan, J.—Escalera, S.—Shi, H.—Wang, Z.—Li, S. Z.: A Dataset and Benchmark for Large-Scale Multi-Modal Face Anti-Spoofing. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 919–928, doi: 10.1109/CVPR.2019.00101.

[31] Zhang, Z.—Yan, J.—Liu, S.—Lei, Z.—Yi, D.—Li, S. Z.: A Face Antispoofing Database with Diverse Attacks. 2012 5th IAPR International Conference on Biometrics (ICB), IEEE, 2012, pp. 26–31, doi: 10.1109/icb.2012.6199754.

Piotr Nawrocki is Associate Professor in the Institute of Computer Science at the AGH University of Science and Technology, Krakow, Poland. His research interests include distributed systems, mobile systems, cloud computing, artificial intelligence and service-oriented architectures. He has participated in several EU research projects including MECCANO, 6WINIT and UniversAAL. He is a member of the Polish Information Processing Society (PTI).

Wojciech Kubaty received his M.Sc. in 2020 in computer science from the AGH University of Science and Technology, Kraków, Poland. His interests include practical use of mobile technologies and machine learning algorithms. He is currently working for one of the biggest and fastest growing companies developing mobile applications in Poland.

Computing and Informatics, Vol. 40, 2021, 195–215, doi: 10.31577/cai_2021_1_195

MORE EFFICIENT ON-THE-FLY VERIFICATION METHODS OF COLORED PETRI NETS

Cong He, Zhijun Ding*

Department of Computer Science and Technology
Tongji University
No. 4800, Caoan highway
Shanghai, China
e-mail: {1930766, dingzj}@tongji.edu.cn

Abstract. Colored Petri Nets (CP-nets or CPNs) are powerful modeling language for concurrent systems. As for CPNs' model checking, the mainstream method is unfolding that transforms a CPN into an equivalent P/T net. However the equivalent P/T net tends to be too enormous to be handled. As for checking CPN models without unfolding, we present three practical on-the-fly verification methods which are all focused on how to make state space generation more efficient. The first one is a basic one, based on a standard state space generation algorithm, but its efficiency is low. The second one is an overall improvement of the first. The third one sacrifices some applicability for higher efficiency. We implemented the three algorithms and validated great efficiency of latter two algorithms through experimental results.

Keywords: Model checking, CPN, on-the-fly, LTL, state space

Mathematics Subject Classification 2010: 93-A30

1 INTRODUCTION

CPNs are powerful graphical language for modeling concurrent systems introduced by Jensen in 1981 [9]. As a kind of high-level Petri nets, CPN is a Petri net that extends the type of place (token) to describe different data types. Moreover, arcs

* Corresponding author

in CPN are labelled with arc expression functions to describe data operations; transitions in CPN are labelled with guard functions to describe branch conditions. In this way, CPN combines the capabilities of Petri nets and a high-level programming language. Success stories of CPN can be found in many industrial domains, such as network protocols [14], systematic softwares [11, 15], embedded systems [3], e-commerce systems [20], etc.

Explicit-state on-the-fly verification [4, 8, 7] is an universal optimization approach for model checking. It integrates state space generation, product automaton construction and counterexample detection (in LTL (Linear Temporal Logic) model checking, a counterexample is an accepting cycle in product automaton). An advantage of this approach is that the algorithm can give an answer without generating full state space. Though success stories of on-the-fly in P/T nets clearly demonstrate its effectiveness and applicability, there are few works dedicated to directly applying on-the-fly in checking CPN models. As for checking CPN models, the mainstream approach is unfolding [16, 13, 12, 2], which transforms a CPN into an equivalent P/T net and implements model checking on the latter. With unfolding, one can directly apply all successful optimization techniques which are difficult to extend to CPNs on the equivalent net, like Data Decision Diagram (DDD) [5, 1], P-invariants [18]. However, a big disadvantage of unfolding is that the equivalent P/T nets transformed from a CPN tends to be too enormous to be handled, with much more places and transitions. Also, the transformed P/T nets cannot directly describe the system to be verified. If a counterexample is detected by verification process, it is difficult to be directly reflected into the system, which is not friendly for debugging.

Concerned with checking CPN models without unfolding, we present a basic on-the-fly method, named *full-info algorithm* (*FullInfo*). It is based on the standard state space generation algorithm [10]. Its core idea is that once a new reachable state m is generated, it calculates a set of all enabled binding elements (we call the set ENBE) in m and stores ENBE together with marking. ENBE serves two purposes. One is to calculate successors of m. Another one is to help check atomic propositions carried by a Büchi automaton state during the generation of product automaton states (or product states for short). For example, some atomic propositions may check enabling of transitions, and enabling of transitions can be reflected by ENBE (for a transition t, if there exists an enabled binding element of t in ENBE, t is enabled. Otherwise, t is not enabled). The algorithm is simple to implement but may generate much redundant information during on-the-fly. A great characteristic of on-the-fly is that it terminates exploration upon a counterexample is detected. Thus, in most cases, the set ENBE of every state m is not fully utilized, cause many successors of m have not been calculated before termination. Also, for some special LTL formulas whose atomic propositions are all related to numbers of tokens (this kind of LTL formulas are called LTLCardinality[1] formulas), ENBE can help

[1] This terminology originated from MCC (Model Checking Contest) which is an annual competition for model checking. `https://mcc.lip6.fr/`.

nothing, because checking these atomic propositions never refers to information in ENBE. This leads to waste of computing resources and low efficiency.

Besides *FullInfo*, we introduce two more efficient state space generation methods integrated into on-the-fly, namely, *minimum representative algorithm* (*MinRep*) and *dynamic exploration algorithm* (*DynExp*). *MinRep* is inspired by *canonical representative* in [17]. Its core idea is that for every enabled transition t in a newly calculated reachable state m, only a representative of enabled binding element of t is initially calculated. While in *DynExp*, none enabled binding element is initially calculated in every newly generated reachable state m. Every enabled binding element in m is calculated on demand when a new successor of m needs generating to start a new path. However, without complete enabled transitions, *DynExp* is hard to check atomic propositions related to enabling of transitions. Thus, it is limited to LTLCardinality formulas.

In short, the main contributions of this paper are summarized as follows:

1. Concerned with LTL model checking of CPN without unfolding, we present an efficient on-the-fly verification method, named *MinRep*. It is an overall improvement of *FullInfo*.
2. For LTLCardinality formulas, we present another more efficient on-the-fly verification method, named *DynExp*.
3. We implemented *FullInfo*, *MinRep* and *DynExp* and did a number of experiments to demonstrate high efficiency of the latter two algorithms.

The rest of this paper is organized as follows: In Section 2, we introduce the definition of Colored Petri Nets and Linear Temporal Logics. In Section 3, we briefly introduce standard state space generation and on-the-fly verification. Then in Section 4, we specify a binding elements calculation problem from the core part of state space generation and on-the-fly. In Sections 5, 6, 7, we elaborate on *FullInfo*, *MinRep* and *DynExp*. Their strengths and weaknesses are discussed as well. Implementation and experimental results are given in Section 8. Finally, in Section 9, we present our conclusion.

2 PRELIMINARIES

2.1 Colored Petri Nets

In this section, definitions of multi-set and non-hierarchical CPN are cited [10] and definitions of LTL are cited [6, 21]. As a matter of convenience, $Bool = \{false, true\}$ is the set of Boolean types, where *true* and *false* are two predicates respectively. $Type[v]$ is the data type of variable v. $Type[ex]$ is the type of expression ex. $EXPR_V$ is an expression constituted by elements from set V.

Definition 1 (Multi-set)**.** Let $S = \{s_1, s_2, s_3, \dots\}$ be a non-empty set. A multi-set m is a function over $S : S \to \mathbb{N}$ that maps each element $s \in S$ into a non-negative

integer $m(s) \in \mathbb{N}$ called the number of appearances (coefficient) of s in m. A multi-set m can also be written as a sum (the operator '++' is a natural addition '+' when two elements s_1, s_2 are the same data type, otherwise '++' is just a junction symbol without real meaning):

$$^{++}\sum_{s \in S} m(s)'s = m(s_1)'s_1 ++ m(s_2)'s_2 ++ m(s_3)'s_3 ++ \ldots$$

Operators: addition ($++$), scalar multiplication ($**$), comparison ($\ll=$), size ($|m|$) and subtraction ($--$) are defined as follows:

- addition: $\forall s \in S$, $(m_1 ++ m_2)(s) = m_1(s) + m_2(s)$,
- scalar multiplication: $\forall s \in S$, $(n ** m)(s) = n * m(s)$,
- comparison: $m_1 \ll= m_2 \Leftrightarrow \forall s \in S$, $m_1(s) \leq m_2(s)$,
- size: $|m| = \sum_{s \in S} m(s)$,
- when $m_1 \ll= m_2$, subtraction is defined as: $\forall s \in S$, $(m_2 -- m_1)(s) = m_2(s) - m_1(s)$.

Definition 2 (Non-hierarchical CPN)**.** A non-hierarchical CPN is a nine-tuple $N = (P, T, A, \Sigma, V, C, G, E, I)$, where P, T, A are finite sets of places, transitions and arcs such that $P \cap T = \emptyset$, $A \subseteq P \times T \cup T \times P$, Σ is finite set of non-empty color sets, V is a finite set of typed variables such that $Type[v] \in \Sigma$ for all variables $v \in V$, $C : P \to \Sigma$ is a color set function that assigns a color set to each place, $G : T \to EXPR_V$ is a guard function that assigns a guard to each transition t such that $Type[G(t)] = Bool$, $E : A \to EXPR_V$ is an arc expression function that assigns an arc expression to each arc a such that $Type[E(a)] = C(p)_{MS}$ where p is the place connected to the arc a, $I : P \to EXPR_\emptyset$ is an initialization function that assigns an initialization expression to each place p such that $Type[I(p)] = C(p)_{MS}$. The variables of a transition t are denoted $Var(t)$, $Var(t) \subseteq V$. $Var(t)$ includes all the variables appearing in t's guard $G(t)$ and arc expressions $E(a)$ for all $a \in A$, a is connected to t.

Definition 3 (Enabling and firing rules)**.** Let $N = (P, T, A, \Sigma, V, C, G, E, I)$ be a non-hierarchical CPN. A *marking* of N is a function M that maps each place $p \in P$ into a multi-set of tokens $M(p) \in C(p)_{MS}$. A *binding* of a transition t is a function b that maps each variable $v \in Var(t)$ into a value $b(v) \in Type[v]$. The set of all bindings for a transition t is denoted $B(t)$, called t's *binding space*. A *binding element* is a pair (t, b) such that $t \in T, b \in B(t)$. The set of all binding elements for a transition t is denoted $BE(t)$, called t's *binding element space*. $BE(t)$ is defined by $BE(t) = \{(t, b) \mid b \in B(t)\}$. The set of all binding elements in a CPN is denoted BE, called *binding element space*. A binding element $(t, b) \in BE$ is enabled in a marking M if and only if the following two properties are satisfied (denotation $G(t)\langle b \rangle$ expresses the evaluation of transition t's guard in the binding b and it is either *true* or *false*; denotation $E(p, t)\langle b \rangle$ expresses the evaluation of arc a's arc expression and it is a multi-set):

1. $G(t)\langle b\rangle = true$;
2. $\forall p \in P, E(p,t)\langle b\rangle \ll = M(p)$.

When (t,b) is enabled in M, it may occur and is leading to a marking M' (written $M \xrightarrow{(t,b)} M'$), such that $\forall p \in P, M'(p) = (M(p) - - E(p,t)\langle b\rangle) + + E(t,p)\langle b\rangle$. A transition t is enabled in a marking M if and only if $\exists (t,b) \in BE(t)$, (t,b) is enabled in M.

Definition 4 (State space)**.** For a marking M and a marking M', if there exists an enabled binding element (t,b) such that $M \xrightarrow{(t,b)} M'$, M' is said to be immediately reachable from M; if there exists an sequence of binding element $(t_1,b_1)(t_2,b_2)\dots(t_n,b_n)$ such that $M \xrightarrow{(t_1,b_1)} M_1 \xrightarrow{(t_2,b_2)} M_2 \dots \xrightarrow{(t_n,b_n)} M'$, M' is said to be reachable from M, written $M \xrightarrow{*} M'$. The state space of a CPN consists of the set $R(m_0) = \{m \mid m_0 \xrightarrow{*} m\}$ of states reachable from the initial state. Each state $m \in R(m_0)$ is called a *reachable state*[2].

2.2 Linear Temporal Logics

Linear Temporal Logic (abbreviated as LTL) is used to describe properties of a system execution. It consists of a non-empty finite set of atomic propositions AP, Boolean operators $\neg$ (negation), $\vee$ (disjunction) and $\wedge$ (conjunction), and temporal operators X (next), U (until), R (release), F (eventually) and G (always). In LTL model checking, the negation of a formula will be transformed into a Büchi automaton. There are many approaches to construct a Büchi automaton from the LTL formula [6, 19].

Definition 5 (Syntax of LTL)**.** The syntax of LTL is defined as follows:

$$\phi ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid X\varphi \mid \varphi U \psi \mid \varphi R \psi \mid F\varphi \mid G\varphi$$

where p is an atomic proposition and ϕ, φ, ψ are well-formed LTL formulas. Referring to MCC and Wolf's [21] provisions for atomic propositions, we make the following provisions for an atomic proposition p:

$$\begin{aligned} p ::= & TRUE \mid FALSE \mid FIREABLE(t)\,(t \in T) \mid DEADLOCK \\ & \mid k_1 p_1 + \dots + k_n p_n \leq k\,(k_i, k \in \mathbb{Z}, p_i \in P) \end{aligned}$$

Let state m be the current state, $FIREABLE(t)$ holds if only if t is enabled in m, $DEADLOCK$ holds if and only if there are no transitions are enabled in m, $k_1 p_1 + \dots + k_n p_n \leq k$ holds if and only if $k_1 M(p_1) + \dots + k_n M(p_n) \leq k$ in m.

[2] State is a snapshot of a system, marking is a distribution of tokens. Though a state m can be uniquely identified by a marking M, they are different concepts. Throughout the paper, we use lower case m (subscripts or superscripts will be used if necessary) to represent a state, upper case M(subscripts or superscripts will be used if necessary) to represent the marking of m.

Definition 6 (Semantics of LTL)**.** Let AP be a non-empty finite set of atomic propositions, $\xi = x_0x_1x_2\ldots$ be a sequence over alphabet 2^{AP}, ϕ, φ, ψ be LTL formulas. We write ξ_i for the suffix of ξ starting at x_i. The semantics $\xi \models \phi$ (ξ models ϕ) is defined as follows:

- $\xi \models p$, iff $p \in x_0$ for $p \in AP$,
- $\xi \models \neg\phi$, iff $\xi \not\models \phi$,
- $\xi \models \varphi \vee \psi$, iff $\xi \models \varphi$ or $\xi \models \psi$,
- $\xi \models X\phi$, iff $\xi_1 \models \phi$,
- $\xi \models \varphi U \psi$, iff $\exists i \geq 0$, $\xi_i \models \psi \wedge (\forall j < i, \xi_i \models \varphi)$.

Other operators ($\wedge$, R, F, G) can be derived from the above operators (X, U, $\neg$): $\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$; $\varphi R \psi \equiv \neg(\neg\varphi U \neg\psi)$; $F\phi \equiv (TRUE)U\phi$; $G\phi \equiv \neg(F\neg\phi)$.

3 STANDARD STATE SPACE GENERATION AND ON-THE-FLY

3.1 Standard State Space Generation

The standard state space generation [10] works on three sets: Node, Unprocessed, Edges. Node stores reachable states. Unprocessed consists of states whose successors have not yet been calculated. Edges stores arcs. As illustrated in Algorithm 1, the algorithm firstly initializes Node, Unprocessed with initial state m_0 and Edges with empty set. Then it selects a reachable state m in Unprocessed and calculates all enabled binding elements in m. Each enabled binding element that occurs will lead to a reachable state m' and an arc from m to m'. If m' has not yet been encountered, it will be added into Node and Unprocessed. The algorithm terminates with full state space.

3.2 On-the-Fly

On-the-fly method was first proposed in [4]. The main idea is integrating state space generation, product automaton construction and detecting counterexamples (in LTL model checking, a counterexample is an accepting cycle in product automaton). In more detail, for a given product state $p :: (m, b)$ (a product state is composed by a reachable state m and an automaton state b), it calculates a successor m' of m, and a successor b' of b. if all atomic propositions carried by b' are satisfied in m', then a product state $p' :: (m', b')$ is generated. If some conditions are triggered, on-the-fly will implement counterexample detection, i.e., if on-the-fly finds the successor p' of p is an encountered product state where it may form a cycle, then on-the-fly will check that. This idea can be illustrated by Algorithm 2. Line 1 is state space generation, lines 3–4 are the product state generation and the line 6 is counterexample detection. As for counterexample detection, there are several ways to do that.

Algorithm 1 Standard state space generation

```
1: NODES ← {m_0}
2: UNPROCESSED ← {m_0}
3: EDGES ← ∅
4: while UNPROCESSED ≠ ∅ do
5:     Select a Marking m in UNPROCESSED
6:     UNPROCESSED ← UNPROCESSED − {m}
7:     for all binding elements (t, b) such that (t, b) is enabled in m do
8:         Calculate m' such that m --(t,b)--> m'
9:         EDGES ← EDGES ∪ {(m, (t, b), m')}
10:        if m' ∉ NODES then
11:            NODES ← NODES ∪ {m'}
12:            UNPROCESSED ← UNPROCESSED ∪ {m'}
13:        end if
14:    end for
15: end while
```

Like nested depth-first search algorithm [4], TCHECK[3] algorithm [7] and DCHECK algorithm [7].

Algorithm 2 on-the-fly

Input: $p :: (m, b)$: a product state
Output: *true* or *false*: checking result

```
1: for m' ← NEXTSUCCESSOR(m) ≠ 'no more' do
2:     for all b' ∈ SUCCESSOR(b) do
3:         if m' satisfies all atomic propositions carried by b' then
4:             Generate a produce state p' :: (m', b')
5:             if p' has been encountered then
6:                 Accepting cycle detection
7:                 if ∃ an accepting cycle then
8:                     Terminate with false
9:                 end if
10:            else
11:                on-the-fly(p')
12:            end if
13:        end if
14:    end for
15: end for
```

[3] The main procedure of TCHECK and DCHECK are non-recursive functions, and they work much more efficiently than nested depth-first search. *FullInfo*, *MinRep* and *DynExp* are integrated into TCHECK algorithm. More details can be referred to [7].

4 ENABLED BINDING ELEMENTS CALCULATION PROBLEM

Enabled binding elements are vitaly important during state space generation. Firstly, all successors of a reachable state are controlled by enabled binding elements (lines 7–8 in Algorithm 1 and line 1 in Algorithm 2). Secondly, enabled binding elements plays a part in product state generation (line 3 in Algorithm 2), because some atomic propositions may check enabling of some transitions, i.e., $FIREABLE(t)$ atomic propositions. The core problem that we encounter is: given a reachable state m, in which way to explore binding element space BE to find enabled binding elements in m to calculate successors of m and generate product states. Different solutions to this problem lead to huge different performances. Intuitively, we may come up with that upon a new reachable state m is generated, explore BE exhaustively at once to get *all* enabled binding elements ENBE in m and store ENBE in case to use. In this way, every time on-the-fly backtracks to m, the process can easily fetch a next enabled binding element from ENBE to calculate another successor of m. This is exactly how *FullInfo* works. We will detail it in the next section.

5 FULLINFO

The core idea of *FullInfo* is very simple: upon getting a new reachable state m, it calculates a set of all enabled binding elements in m called ENBE and stores ENBE together with marking M immediately. Then it uses ENBE to generate different successors of m and product states. The technical difficulties lie in how to get all enabled binding elements and how to manage them.

5.1 How to Get All Enabled Binding Elements

Traversing t's binding space $B(t)$ is essentially a combination problem that assigns a value $b(v) \in Type[v]$ to each variable $v \in Var(t)$. The binding space $B(t)$ can be depicted as a tree (we name it t's *binding space tree*). Assume that $t \in T$ is a transition, $|Var(t)| = k$ such that $Var(t) = \{v_1, v_2, \ldots, v_k\}$ and for each variable v_i, $|Type[v_i]| = n_i$ such that $Type[v_i] = \{c_{i1}, c_{i2}, \ldots, c_{in_i}\}$, then $B(t)$ can be depicted as a tree in Figure 1. The depth of this tree is equal to the number of variables in $Var(t)$. All direct successors of a node are overall mapping cases of next variable. For example, the direct successors of node '$v_1 = c_{11}$' list complete mapping cases of next variable v_2, which is from $v_2 = c_{21}$ to $v_2 = c_{2n_2}$ (we use horizontal ellipsis to represent all omitted nodes in its layer and vertical ellipsis to represent all omitted child nodes of one node). A path from *root* node to a leaf node is a specific binding of t, and all paths like this constitute t's binding space $B(t)$. We use a recursive function to traverse this tree to get all enabled binding elements. The function is presented in Algorithm 3. When the depth is lower than $|Var(t)|$, the function tries to assign a color to the variable which corresponds to the depth and then recurves down. If the depth is equal to or greater than $|Var(t)|$ which means the function

reaches a leaf node and a complete binding b is obtained, then it begins to check the enabling of (t, b). The specific checking procedure lies in lines 2–9. If (t, b) is enabled, it will be added into ENBE. After implementing this function on each transition $t \in T$, the complete ENBE will be obtained.

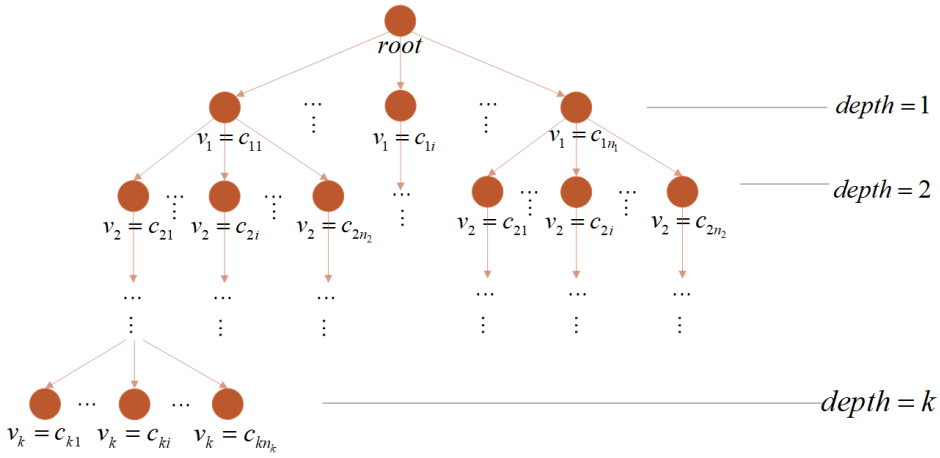


Figure 1. Binding space tree

Algorithm 3 getENBE$(m, t, b, depth)$

Input: m: reachable state, t: Transition, b: Binding, $depth$: int
Output: ENBE: a set stores enabled binding elements

```
1:  if depth ≥ |Var(t)| then
2:      if ¬G(t)⟨b⟩ then
3:          return
4:      end if
5:      for all p ∈ •t do
6:          if ¬(E(p,t)⟨b⟩ ≤ M(p)) then          ▷ M is m's marking
7:              return
8:          end if
9:      end for
10:     ENBE.ADD(b)
11: else
12:     for all c ∈ Type[v_depth] do
13:         b[depth] ← c
14:         getENBE(m, t, b, depth + 1)
15:     end for
16: end if
```

5.2 How to Manage All Enabled Binding Elements

In this subsection, we focus on how to take advantage of ENBE to serve for successor reachable states generation and product states generation. We use a two-level queue as data structure for ENBE. The first level queue stores enabled transitions, each enabled transition has a second level queue consisting of its bindings which render it enabled. Figure 2 is an example of

$$\text{ENBE} = \{(t_1, b_{11}), (t_1, b_{12}), (t_2, b_{21}), (t_2, b_{22}), (t_2, b_{23}), (t_3, b_{31})\}.$$

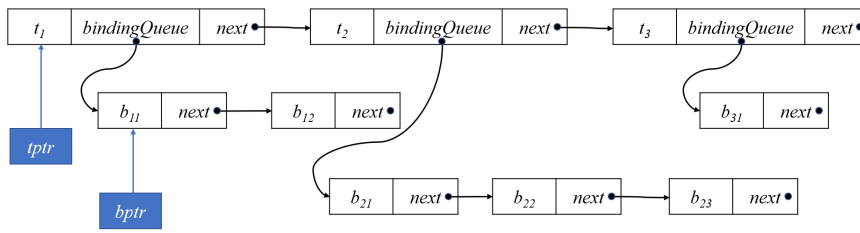


Figure 2. Data structure for ENBE

In the data structure of ENBE, there are two pointers, *tptr* and *bptr*, respectively pointing to an enabled transition and a binding of it. They are used to represent an enabled binding element, i.e., in Figure 2 they represent (t_1, b_{11}). Each time an enabled binding element occurs, *bptr* will move to next binding of the current queue. If next binding does not exist, i.e., it reaches the tail of the queue, *tptr* will move to next transition and *bptr* will point to the head of its *bindingQueue*. By this, the process can obtain different successors of a reachable state and this procedure is one possible way how line 7 in Algorithm 1 and line 1 in Algorithm 2 work.

Another crucial role of ENBE is to help generate a product state. For example, let $F\alpha$ be a LTL formula, where α is an atomic proposition $FIREABLE(t_2)$. During checking process, every state needs to check if t_2 is enabled in it. To do this, every state just needs to check its ENBE. If t_2 appears in the first level queue, it is enabled, otherwise it is not.

With *FullInfo*, we can basically solve the *enabled binding elements calculation problem*. The two core parts, successor reachable states generation and product states generation, can be done easily with the aid of ENBE. But a conspicuous disadvantage is that it may generate much redundant information. Or in other words, many states' ENBE may not be fully utilized. For example, if on-the-fly reports a checking result without generating the whole state space, that means there must exist some states where some enabled binding elements have not yet occurred and these enabled binding elements remain to be redundant. Another case is when none of atomic propositions of the LTL formula is form of $FIREABLE(t)$ or $DEADLOCK$, ENBE can do nothing to help in the product state generation. This disadvantage is

particularly obvious when a CPN's binding element space is huge or when on-the-fly detects a counterexample along a path with few backtrackings. Here, Figure 3 is an example to demonstrate the second case. Figure 3 is a partial state space of a CPN. We use solid cycles to represent reachable states, arrows marked by transitions to represent enabled transitions in a reachable state, solid squares to represent enabled binding elements which have occurred and hollow squares to represent enabled binding elements which have not yet occurred. If on-the-fly detected a counterexample $S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_0$ after generating S_0, S_1, S_2 and then terminates, then the computing resources allocated for calculating the hollow squares are wasted because they had never been used during checking process. If on-the-fly went deeper along this path and detected a counterexample, the waste would be worse. Therefore, we need another algorithm to solve the *enabled binding elements calculation problem.*

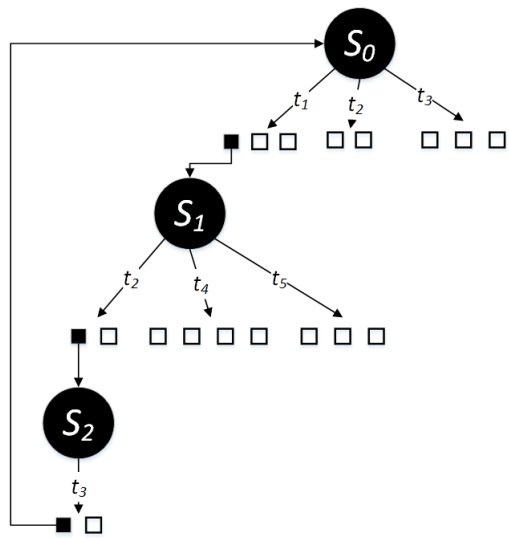


Figure 3. Partial state space

6 MINREP

In this section, we develop another solution to the *enabled binding elements calculation problem.* According to the definition of enabling of transitions (in Definition 3), if there exists one enabled binding element of a transition t, t is proven to be enabled. Thus, as for checking atomic propositions during product state generation, it is unnecessary to calculate complete ENBE in each reachable state. The core idea of *MinRep* is to specify an order over $B(t)$ such that $(B(t), \prec)$ for each transition t. And for t's binding element space, this algorithm only initially calculates one en-

abled representative which is the smallest enabled one in $BE(t)$. Certainly, if t is not enabled, there will not be such a representative. This idea is inspired by *canonical representative* [17].

Before presenting the order $(B(t), \prec)$, we firstly specify an order $(C, \prec)$ over each color set $C \in \Sigma$. Here are the orders:

1. $(C, \prec)$: $\forall c_i, c_j \in C$, $c_i \prec c_j$ iff $i < j$. Here the index i, j can be arbitrarily defined. Typecially we use the index in data structure storing color set C, i.e., sequence table.
2. $(B(t), \prec)$: $Var(t) = \{v_1, v_2, \ldots, v_n\}$, $\forall b_i, b_j \in B(t)$, $b_i = \langle c_{i1}, c_{i2}, \ldots, c_{in}\rangle$, $b_j = \langle c_{j1}, c_{j2}, \ldots, c_{jn}\rangle$, $b_i \prec b_j$ iff $\exists k, 1 \leq k \leq n, c_{ik} \prec c_{jk}$ and $\forall m, 1 \leq m < k, c_{im} = c_{jm}$. $(B(t), \prec)$ can be regarded as a lexicographical order induced by the vector of binding.

Calculating representative is similar to Algorithm 3. The minor difference is that for *MinRep*, upon getting an enabled binding element, it terminates. More specifically, we just need to insert a terminate clause after line 10. All representatives are organized in a set, and we name it Ent. Here we use a queue to organize Ent. Figure 4 is an example of

$$\text{Ent} = \{(t_1, b_{11}), (t_2, b_{21}), (t_3, b_{31})\}.$$

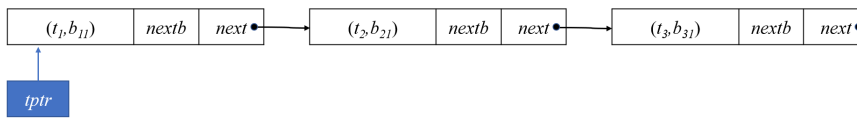


Figure 4. Data structure for Ent

In this data structure, *tptr* is a pointer pointing to a transition occurring last time and *nextb* is a binding that is prepared to calculate a successor reachable state next time (initially it equals to the binding part of the related representative). When on-the-fly backtracks to a reachable state m, the program uses *nextb* to calculate a successor m' of m and tries to check bindings behind $(tptr- > nextb)$ to update *nextb*. If there are no more bindings related to *tptr*, *tptr* will move to next representative, which indicates the binding element space of prior transition has been explored exhaustively. The successor generation procedure is presented in Algorithm 4 where $(m.tptr- > t)$ is the transition of corresponding representative which pointed by *tptr*. Function UPDATE(m) is to update $(tptr- > nextb)$. It regards the binding vector as a n-digit number, where n equals to $|Var(m.tptr- > t)|$, and tries to implement increment '++' on this number. During the increment operation, it may trigger carries (lines 7–8) like numbers. Function NEXTSUCCESSOR(m) is to get another successor of m according to binding element $(m.tptr- > t, m.tptr- > nextb)$.

As for checking atomic propositions, ENT together with marking is right enough. As for $FIREABLE(t)$ propositions, *MinRep* checks ENT of current reachable state m. If there is a related representative of t, t is enabled, otherwise, it is not. As for *DEADLOCK* propositions, *MinRep* also checks ENT of current reachable state m. If ENT is empty, the propositions are satisfied, otherwise, it is not. As for $k_1p_1 + \cdots + k_np_n \leq k$ propositions, *MinRep* checks them by the marking M of current reachable m. if $k_1M(p_1) + \cdots + k_nM(p_n) \leq k$ holds, the propositions are satisfied, otherwise, it is not.

Algorithm 4 MinRep

Input: m: current reachable state
Output: m': successor of m or 'no more'

```
1: function UPDATE(m)
2:     while m.tptr ≠ NULL do
3:         (t', b') ← (m.tptr− > t, m.tptr− > nextb)
4:         n : int ← |Var(t')|                          ▷ Var(t') = {v_1, v_2, ..., v_n}
5:         for i from n to 1 do
6:             c: color ← NEXTCOLOR(b[i])
7:             if c = 'no more' then
8:                 b'[i] ← first color of Type[v_i]
9:                 continue
10:            else
11:                b'[i] ← c
12:                if (t', b') is enabled in m then
13:                    m.tptr− > nextb ← b'
14:                    return
15:                end if
16:            end if
17:        end for
18:        tptr ← tptr− > next
19:    end while
20:    return
21: end function
22:
23: function NEXTSUCCESSOR(m)
24:    if tptr = NULL then
25:        return 'no more'
26:    else
27:        Calculate m' such that m --(m.tptr−>t, m.tptr−>nextb)--> m'
28:        UPDATE(m)
29:        return m'
30:    end if
31: end function
```

For each reachable state m, *MinRep* only calculates partial information from m's binding element space which is just enough to handle all kinds of atomic propositions. Compared with *FullInfo*, redundant information is much less and efficiency would be higher. But it has the same disadvantage that if all atomic propositions of the LTL formula are neither form of *DEADLOCK* nor *FIREABLE*(t), i.e., LTLCardinality formulas, ENT can help nothing and remain to be redundant. So we need another more efficient algorithm to handle LTLCardinality formulas.

7 DYNEXP

In this section, we develop another algorithm dedicated to handling LTLCardinality formulas. As for checking atomic propositions of this formula type, it is unnecessary to calculate any enabled binding elements. The sole function of enabled binding elements here is to calculate successors. In order to be more efficient, *DynExp* will not initially calculate any enabled binding element in each newly calculated reachable state m. Instead of calculating all enabled binding elements at once, enabled binding elements are obtained dynamically. Once an enabled binding element is obtained, the algorithm will let it occur immediately and calculate a successor m' of m, then continue on-the-fly on m'. To obtain different successors of a given reachable state m when on-the-fly backtracks to m, *DynExp* extends the orders defined in Section 6 to the whole binding element space BE such that $(BE, \prec)$, and each reachable state would record the binding element that occurred last time, called *lastbe*. In this way, when on-the-fly backtracks to m, it can check binding elements behind *lastbe* until an enabled one is detected or there are no more in BE.

Before presenting the order $(BE, \prec)$, we firstly specify an order $(T, \prec)$ over transition set T. They are defined as follows:

1. $(T, \prec)$: $\forall t_i, t_j \in T$, $t_i \prec t_j$ iff $i < j$. Here the index value i, j can be arbitrarily defined. Typically we use its index value in a specific data structure that stores the transition set.
2. $(BE, \prec)$: $\forall (t_i, b_{ik}), (t_j, b_{jm}) \in BE$, $(t_i, b_{ik}) \prec (t_j, b_{jm})$ iff $t_i \prec t_j$ or $i = j$, $b_{ik} \prec b_{jm}$.

With the aid of order $(BE, \prec)$ and *lastbe*, the algorithm can iterate over BE exhaustively to get different enabled binding elements in each reachable state. We specify three functions to implement the idea, namely, NEXTBINDING$((t, b))$, NEXTTRANSITION$((t, b))$ and NEXTSUCCESSOR(m). They are illustrated in Algorithm 5. Function NEXTBINDING$((t, b))$ is similar to function UPDATE(m) illustrated in Algorithm 4. It is to fetch (t, b)'s next binding of transition t according to order $(BE, \prec)$. Where function NEXTCOLOR(c) is to get color c's next color in c's color set C by order $(C, \prec)$. Function NEXTTRANSITION$((t, b))$ is very simple. Its job is to fetch next transition of t according to order $(T, \prec)$, and initiate the binding (lines 23–25). If there are no more transitions, 'NULL' will be returned.

Algorithm 5 DynExp

Input: m: current reachable state
Output: m': successor of m

```
1: function NEXTBINDING((t, b))
2:     (t', b') ← (t, b)
3:     n : int ← |Var(t)|
4:     for i from n to 1 do
5:         c: color ← NEXTCOLOR(b[i])
6:         if c = 'no more' then
7:             b'[i] ← first color of Type[v_i]
8:             continue
9:         else
10:            b'[i] ← c
11:            return (t', b')
12:        end if
13:    end for
14:    return 'no more'
15: end function
16:
17: function NEXTTRANSITION((t, b))
18:    (t', b') : binding element
19:    t' ← t.index++
20:    if t' = 'no more' then
21:        return 'no more'
22:    else
23:        for i from 1 to |Var(t')| do
24:            b'[i] ← first color of Type[v_i]
25:        end for
26:    end if
27:    return 'no more'
28: end function
29:
30: function NEXTSUCCESSOR(m)
31:    repeat
32:        repeat
33:            (t, b) ← NEXTBINDING(m.lastbe)
34:            if (t, b) is enabled in m then
35:                Calculate m' such that m --(t,b)--> m'
36:                return m'
37:            end if
38:        until (t, b) = 'no more'
39:        (t, b) ← NEXTTRANSITION(m.lastbe)
40:    until (t, b) = 'no more'
41:    return 'no more'
42: end function
```

Function NEXTSUCCESSOR(m) keeps calling NEXTBINDING($m.lastbe$) to iterate over $B(t)$, trying to find an enabled one. If there are no more or do not exist at all, it will move to next transition by calling NEXTTRANSITION($m.lastbe$). Upon obtaining an enabled binding element (t, b), (t, b) will occur immediately leading to a successor m' of m and terminates this function. Or if there are no more enabled binding element, NEXTSUCCESSOR(m) will return 'no more'.

As for generating product states, we have nothing to worry about, because checking LTLCardinality formulas just need information of markings and markings are never absent.

Because upon getting a successor, the algorithm terminates exploring binding element space and continues on-the-fly, any enabled binding element and corresponding successor are calculated on demand during the checking process. Hence, no redundant information is generated and *DynExp* would be more efficient than *FullInfo* and *MinRep*. However, it is limited to LTLCardinality formulas. It sacrifices applicability for greater efficiency.

8 EXPERIMENT

We implemented all three algorithms in C++, and they are all integrated into the non-recursive on-the-fly TCHECK. The source code is available from:

- *FullInfo*: `https://github.com/Tj-Cong/EnPAC_CPN`,
- *MinRep*: `https://github.com/Tj-Cong/EnPAC_CPN_F`,
- *DynExp*: `https://github.com/Tj-Cong/EnPAC_CPN_C`.

We get testing data from MCC. There are two kinds of models provided by MCC:

- Academic models: these were designed in universities by researcher, to benchmark some tools, to illustrate a typical situation or within the context of academic projects and cooperations.
- Industrial models: these where designed within the context of industrial projects.

Both kinds of models have practical meanings and each model is provided with a file describing it which can be found from `https://mcc.lip6.fr/models.php`. Each model can result in several instances due to the scaling parameter (the parameters are often indicated at the end of its instance name). There are two kinds of LTL formulas. One is called LTLCardinality formulas whose atomic propositions are all form of $k_1p_1 + \cdots + k_np_n \leq k$. Another kind is called LTLFireability formulas whose atomic propositions are all form of $FIREABLE(t)$. Based on this testing data, we have done two sets of experiments. One is designed to measure the performance on LTLcardinality formulas. Another one is designed to measure the performance on LTLfireability formulas. They are both implemented on a Linux PC with Intel(R) Core(TM) i7-7700HQ CPU @ 2.80 GHz and 16 GB RAM. Operating system is Ubuntu 18.04 LTS.

As for the first experiment, all three algorithms were tested on four different instances with different size of binding element space. Each instance is checked by two formulas. Testing time for each formula is limited to 300 seconds, and if one algorithm does not finish checking one formula within 300 seconds, the corresponding table entry will be marked as "?". Memory for each formula is limited to 16 GB and if one algorithm cannot finish checking one formula within 16 GB, the corresponding table entry will be marked as "Overflow". Of course, the three algorithms are set to traverse paths in the same order. The result is presented in Table 1. |States| represents the number of states explored by on-the-fly before termination. The unit of time is seconds; the unit of memory is MB; the size of binding element space $|BE|$ is calculated by:

$$|BE| = \sum_{t \in T} \left(\prod_{v \in Var(t)} (|Type[v]|) \right).$$

From Table 1, we can find that *DynExp* always consumes the least time and memory. Thus, we can conclude that *DynExp* is the most efficient algorithm for LTLcardinality formulas, no matter with respect to memory consumption or time used. *MinRep* ranks the second and *FullInfo* is the least efficient. Beginning from Formula 2 of DWM-COL-40, memory for *FullInfo* overflows. And beginning from Formula 1 of GRA-COL-11, time for *MinRep* runs out of 300 seconds. When $|BE|$ goes larger, the advantage of dynamic exploration becomes more salient. By checking Formula 1 and Formula 2 of every instance, we can also find out that the more states on-the-fly explores, more obvious the advantage of *DynExp* is.

Models	ALD-COL-10[1]		DWM-COL-40[2]		GRA-COL-11[3]		DVM-COL-16[4]	
|Places|	20		11		5		6	
|Transitions|	15		8		7		7	
|BE|	132		12 800		2 705 087		4 433 952	
formulas	1	2	1	2	1	2	1	2
|States|	38 115	43 109	20 936	1 251 201	286 755	545 605	178 433	457 369
Time (FullInfo)	2.510	3.193	36.417	?	?	?	?	?
Time (DynExp)	0.899	1.644	8.089	28.381	27.462	80.120	10.377	57.256
Time (MinRep)	0.905	2.751	14.4249	151.37	> 300	> 300	> 300	> 300
Memory (FullInfo)	506.367	529.363	2 766.367	Overflow	Overflow	Overflow	Overflow	Overflow
Memory (DynExp)	433.348	449.348	504.344	8 080.348	871.244	1387.344	1181.348	2,518.352
Memory (MinRep)	433.348	449.348	508.348	8 252.348	?	?	?	?

[1]The full name is AirplaneLD-COL-0010.
[2]The full name is DatabaseWithMutex-COL-40.
[3]The full name is GlobalResAllocation-COL-11.
[4]The full name is DrinkVendingMachine-COL-16.

Table 1. Comparison on LTLCardinality formulas

As for the second experiment, *FullInfo* and *MinRep* were implemented on four different instances with different size of binding element space. Same as the first experiment, each instance is checked by two formulas and each formula is limited to 300 seconds and 16 GB. The result is presented in Table 2. Obviously, *MinRep* works more efficiently than *FullInfo*. Comparing Formula 2 of FR-COL-G005 with

Formula 1 of GRA-COL-9, we can find that for *MinRep*, the memory consumption is much lower in GRA-COL-09, while for *FullInfo*, the memory consumption is much higher in GRA-COL-9, because the size of binding element space is much bigger. We can also conclude that when $|BE|$ goes larger, the advantage of *MinRep* becomes more obvious. Also, the more states on-the-fly explores, more salient the advantage of *MinRep* is.

Models	ALD-COL-50[1]		DWM-COL-40[2]		FR-COL-G005[3]		GRA-COL-9[4]	
|Places|	20		11		104		5	
|Transitions|	15		8		66		7	
|BE|	612		12 800		134 480		1 003 437	
formulas	1	2	1	2	1	2	1	2
|States|	7	209	2 077	9 596	115 121	31 812	36424	?
Time (FullInfo)	0.104	0.121	3.054	13.600	22.739	57.739	?	?
Time (MinRep)	0.091	0.105	1.628	5.092	3.609	7.097	113.148	>300
Memory (FullInfo)	327.359	334.357	630.359	2 222.359	1 791.363	3 894.363	Overflow	Overflow
Memory (MinRep)	327.351	327.351	347.355	405.355	712.351	1 553.348	400.348	?

[1]The full name is AirplaneLD-COL-0050.
[2]The full name is DatabaseWithMutex-COL-40.
[3]The full name is FamilyReunion-COL-L00200M0020C010P010G005.
[4]The full name is GlobalResAllocation-COL-09.

Table 2. Comparison on LTLFireability formulas

From the two experiments, we can conclude that no matter what kind of LTL formulas is, *MinRep* is always more efficient that *FullInfo*. While for LTLCardinality formulas, *DynExp* works best.

9 CONCLUSIONS

We have presented a basic state exploration method and two more efficient ones under the framework of on-the-fly. The basic one, *FullInfo*, simply calculates all enabled binding elements for every newly generated reachable state. It is easy to implement but efficiency is low. *MinRep* is 'semi-dynamic'. It calculates all enabled transitions for every newly generated reachable state, but for each enabled transition, only a minimum representative of enabled binding elements is initially calculated, others are calculated dynamically on demand. While *DynExp* is 'fully-dynamic'. Every enabled binding element is calculated on demand by on-the-fly. As for applicability, *FullInfo*=*MinRep*>*DynExp*. As for efficiency, *DynExp*>*MinRep*>*FullInfo*.

Acknowledgement

This work is partially supported by the National Key Research and Development Program of China under the Grant No. 2018YFB2100800 and the National Natural Science Foundation of China under Grant No. 61672381, and in part by the Fundamental Research Funds for the Central Universities under Grant No. 22120180508.

REFERENCES

[1] Amparore, E. G.—Donatelli, S.—Beccuti, M.—Garbi, G.—Miner, A. S.: Decision Diagrams for Petri Nets: A Comparison of Variable Ordering Algorithms. In: Koutny, M., Kristensen, L., Penczek, W. (Eds.): Transactions on Petri Nets and Other Models of Concurrency XIII. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 11090, 2018, pp. 73–92, doi: 10.1007/978-3-662-58381-4_4.

[2] Bergenthum, R.—Lorenz, R.—Mauser, S.: Faster Unfolding of General Petri Nets Based on Token Flows. In: van Hee, K. M., Valk, R. (Eds.).: Applications and Theory of Petri Nets (PETRI NETS 2008). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 5062, 2008, pp. 13–32, doi: 10.1007/978-3-540-68746-7_6.

[3] Christensen, S.—Jørgensen, J. B.: Analysing Bang & Olufsen's Beolink® Audio/Video System Using Coloured Petri Nets. In: Azéma, P., Balbo, G. (Eds.): Application and Theory of Petri Nets 1997 (ICATPN 1997). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1248, 1997, pp. 387–406, doi: 10.1007/3-540-63139-9_47.

[4] Courcoubetis, C.—Vardi, M. Y.—Wolper, P.—Yannakakis, M.: Memory Efficient Algorithms for the Verification of Temporal Properties. In: Clarke, E. M., Kurshan, R. P. (Eds.): Computer-Aided Verification (CAV 1990). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 531, 1991, pp. 233–242, doi: 10.1007/BFb0023737.

[5] Couvreur, J.-M.—Encrenaz, E.—Paviot-Adet, E.—Poitrenaud, D.—Wacrenier, P.-A.: Data Decision Diagrams for Petri Net Analysis. In: Esparza, J., Lakos, C. (Eds.): Applications and Theory of Petri Nets 2002 (ICATPN 2002). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2360, 2002, pp. 101–120, doi: 10.1007/3-540-48068-4_8.

[6] Gastin, P.—Oddoux, D.: Fast LTL to Büchi Automata Translation. In: Berry, G., Comon, H., Finkel, A. (Eds.): Computer Aided Verification (CAV 2001). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2102, 2001, pp. 53–65, doi: 10.1007/3-540-44585-4_6.

[7] Geldenhuys, J.—Valmari, A.: More Efficient On-the-Fly LTL Verification with Tarjan's Algorithm. Theoretical Computer Science, Vol. 345, 2005, No. 1, pp. 60–82, doi: 10.1016/j.tcs.2005.07.004.

[8] Gerth, R.—Peled, D. A.—Vardi, M. Y.—Wolper, P.: Simple On-the-Fly Automatic Verification of Linear Temporal Logic. In: Dembinski, P., Sredniawa, M. (Eds.): Protocol Specification, Testing and Verification XV (PSTV 1995). Springer, Boston, MA, IFIP Advances in Information and Communication Technology, 1996, pp. 3–18, doi: 10.1007/978-0-387-34892-6_1.

[9] Jensen, K.: Coloured Petri Nets and the Invariant-Method. Theoretical Computer Science, Vol. 14, 1981, No. 3, pp. 317–336, doi: 10.1016/0304-3975(81)90049-9.

[10] Jensen, K.—Kristensen, L. M.: Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer, Berlin, Heidelberg, 2009, doi: 10.1007/b95112.

[11] Jørgensen, J. B.—Bossen, C.: Requirements Engineering for a Pervasive Health Care System. Proceedings of the 11th IEEE International Conference on Requirements

Engineering (RE 2003), Monterey Bay, CA, USA, September 2003, pp. 55–64, doi: 10.1109/ICRE.2003.1232737.

[12] Kordon, F.—Linard, A.—Paviot-Adet, E.: Optimized Colored Nets Unfolding. In: Najm, E., Pradat-Peyre, J. F., Donzeau-Gouge, V. V. (Eds.): Formal Techniques for Networked and Distributed Systems – FORTE 2006. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 4229, 2006, pp. 339–355, doi: 10.1007/11888116_25.

[13] Kozura, V. E.: Unfoldings of Coloured Petri Nets. In: Bjørner, D., Broy, M., Zamulin, A. V. (Eds.): Perspectives of System Informatics (PSI 2001). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2244, 2001, pp. 268–278, doi: 10.1007/3-540-45575-2_27.

[14] Kristensen, L. M.—Jensen, K.: Specification and Validation of an Edge Router Discovery Protocol for Mobile Ad Hoc Networks. In: Ehrig, H., Damm, W., Desel, J., Große-Rhode, M., Reif, W., Schnieder, E., Westkämper, E. (Eds.): Integration of Software Specification Techniques for Applications in Engineering, Priority Program SoftSpez of the German Research Foundation. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3147, 2004, pp. 248–269, doi: 10.1007/978-3-540-27863-4_15.

[15] Kristensen, L. M.—Jørgensen, J. B.—Jensen, K.: Application of Coloured Petri Nets in System Development. In: Desel, J., Reisig, W., Rozenberg, G. (Eds.): Lectures on Concurrency and Petri Nets (ACPN 2003). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3098, 2004, pp. 626–685, doi: 10.1007/978-3-540-27755-2_18.

[16] McMillan, K. L.: Using Unfoldings to Avoid the State Explosion Problem in the Verification of Asynchronous Circuits. In: von Bochmann, G., Probst, D. K. (Eds.): Computer Aided Verification (CAV 1992). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 663, 1993, pp. 164–177, doi: 10.1007/3-540-56496-9_14.

[17] Schmidt, K.: Integrating Low Level Symmetries into Reachability Analysis. In: Graf, S., Schwartzbach, M. I. (Eds.): Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2000). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1785, 2000, pp. 315–330, doi: 10.1007/3-540-46419-0_22.

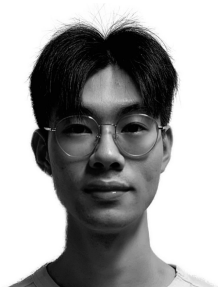
[18] Schmidt, K.: Using Petri Net Invariants in State Space Construction. In: Garavel, H., Hatcliff, J. (Eds.): Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2003). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2619, 2003, pp. 473–488, doi: 10.1007/3-540-36577-x_35.

[19] Tian, C.—Song, J.—Duan, Z.—Duan, Z.: LtlNfBa: Making LTL Translation More Practical. In: Liu, S., Duan, Z. (Eds.): Structured Object-Oriented Formal Language and Method (SOFL + MSVL 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9559, 2016, pp. 179–194, doi: 10.1007/978-3-319-31220-0_13.

[20] Wang, Z.—Luan, W.—Du, Y.—Qi, L.: Composition and Application of Extended Colored Logic Petri Nets to E-Commerce Systems. IEEE Access, Vol. 8, 2020, pp. 36386–36397, doi: 10.1109/access.2020.2974883.

[21] Wolf, K.: How Petri Net Theory Serves Petri Net Model Checking: A Survey. In: Koutny, M., Pomello, L., Kristensen, L. (Eds.): Transactions on Petri Nets and

Other Models of Concurrency XIV. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 11790, 2019, pp. 36–63, doi: 10.1007/978-3-662-60651-3_2.



Cong He received his B.Sc. degree in computing science and technology from the Tongji University, Shanghai, China, in 2019. He is currently pursuing his M.Sc. degree in the Department of Computer Science and Technology, Tongji University, Shanghai, China. His current research interests include Petri nets and model checking.

Zhijun Ding received his M.Sc. degree from the Shandong University of Science and Technology, Tai'an, China, in 2001, and his Ph.D. degree from the Tongji University, Shanghai, China, in 2007. He is currently Professor with the Department of Computer Science and Technology, Tongji University. He has published over 100 papers in domestic and international academic journals and conference proceedings. His research interests are in formal engineering, Petri nets, services computing, and mobile internet.

Computing and Informatics, Vol. 40, 2021, 216–248, doi: 10.31577/cai_2021_1_216

FORMAL APPROACH BASED ON PETRI NETS FOR MODELING AND VERIFICATION OF VIDEO GAMES

Franciny M. Barreto

Federal University of Jataí
Department of Computer Science
BR 364, km 195, 3800
Jataí – GO, Brazil
e-mail: franciny@ufg.br

Stéphane Julia

Federal University of Uberlândia
Faculty of Computation
2121 João Naves de Ávila Ave
Uberlândia – MG, Brazil
e-mail: stephane@ufu.br

Abstract. Video games are complex systems that combine technical and artistic processes. The specification of this type of system is not a trivial task, making it necessary to use diagrams and charts to visually specify sets of requirements. Therefore, the underlying proposal of this work is to present an approach based on the formalism of Petri nets for aiding in the design process of video games. The activities of the game are represented by a specific type of Petri net called Work-Flow net. The definition of a topological map can be represented by state graphs. Using Colored Petri nets, it is possible to define formal communication mechanisms between the model of activity and the model of the map. The simulation of the timed models allows then to produce an estimated time that corresponds to the effective duration a player will need to complete a level of a game. Furthermore, a kind of Soundness property related to gameplay in a game Quest can be verified through state space analysis. For a better understanding of the approach, the video game Silent Hill II is used.

Keywords: Petri nets, video games, WorkFlow net, state graph, soundness verification, simulation, CPN tools

Mathematics Subject Classification 2010: 68-M99

1 INTRODUCTION

A video game is a synthesis of code, images, music and animation that come together in the form of entertainment. The creation of video games differs from the creation of classic software because of the pre-production phase and due to the extensive use and integration of multimedia resources. In addition, the creation of video games involves some activities that do not necessarily exist when considering the process of traditional software development. Some of these activities are called game design and level design.

According to [10], game design is a difficult task that combines technical and artistic processes. The game design phase defines the main aspects related to the universe of the game, such as epoch and style, goal to be achieved, etc. In the level design phase, the main actions and objects of the game are defined [4].

A game has to be interactive, entertaining and give controlled freedom to the player. It is important to propose challenges that are neither easy nor too difficult to solve. Furthermore, it is of fundamental importance to ensure that the game experience leads to a succession of goals within a reasonable time [10]. To accomplish all these requirements is not a trivial task, even more in complex games.

Over the years, video games have evolved and become more complex. The increasing complexity of game development highlights the need for tools to improve productivity in terms of time, cost, and quality [13]. It is important to adopt Software Engineering practices to address the challenges that game developers face.

Some studies have shown the use of UML diagrams to show how different objects in a game will interact according to some actions that will be performed by the player [1, 14]. UML diagrams are interesting as they produce an execution structure of the game. However, they do not present in an explicit way the possible scenarios that exist in a mission or at a game level [3].

On the other hand, some studies show the use of formal methods in game modeling, like the ones presented in [2, 10, 3]. In [10], for example, a new type of Petri net called transactions net is presented. The transactions net allows modeling logical and temporal transactions while the topological map of the game is modeled by a type of graph called hypergraph. The authors created a specific communication mechanism between the transactions net and the hypergraph to establish the influence that a model has over the other. However, formal analysis is applied only in the transactions net (because it is represented by a Petri net).

The study in [3] presents a new approach based on WorkFlow nets to specify the scenarios existing at a quest level. In this approach, the sequent calculus of linear

logic was used to prove the correctness of the scenarios a player can execute within a quest of a game. Such an approach only considers models based on the activities of the player and ignores completely the topological map vision of the game.

The research presented in this paper shows an approach where the scenario of a video game is represented by the combination of two types of Petri nets: WorkFlow nets and State Graphs. The WorkFlow nets are used to represent the activities that exist at a game level. The topological map that represents the areas of the virtual world where the player can progress is then represented by a State Graph. To specify the communication between both models, a synchronous communication mechanism is considered. A time version of the models is also presented in order to estimate the time duration a player will need to complete a specific level of the game. The modeling, analysis and simulation of the video game Silent Hill will be implemented on CPN (Colored Petri nets) Tools.

2 THEORETICAL FOUNDATIONS

2.1 Petri Nets

A Petri net is a graphical and mathematical modeling tool that allows one to model, analyze and control discrete event systems that involve parallel activities, concurrency between processes and asynchronous communication mechanisms [9]. Petri nets are formally defined as a directed bipartite graph with two types of nodes called *places* and *transitions*. These nodes can be connected by directed arcs. An arc can only connect a place to a transition or a transition to a place [20]. In graphical notation, the places are represented by circles and transitions by rectangles. Formally, Petri nets can be defined as follows [20].

Definition 1 (Petri nets)**.** A Petri net is a triple $PN = (P, T, F)$, where:

- $\mathbf{P}$ is a finite set of places of PN.
- $\mathbf{T}$ is a finite set of transitions of PN.
- $\mathbf{F} \subset (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P})$ represents a set of directed arcs that connect places to transitions and transitions to places.

According to [20], the concepts of input place and output place are then defined in terms of flow relation $\mathbf{F}$ as follows:

- A place p is an input place of a transition t if $(p, t) \in F$. The pre-set' $= \{p \mid (p, t) \in F\}$ defines all input places of a transition t.
- A place p is an output place of a transition t if $(t, p) \in F$. The post-set' $= \{p \mid (t, p) \in F\}$ defines all output places of a transition t.

In a Petri net, the occurrence of an event in the system is represented by the transition to which this event is associated. A transition t can only be fired if each input place of t contains at least one token. A token indicates whether the

condition associated with a place is satisfied. At any time, a place contains zero or more tokens, drawn as black dots. In Figure 1, for example, there are two tokens in the place *wait*, which means two clients are waiting to use the x-ray machine. One token in place *free* indicates that the x-ray machine is free and can be used. In that way, the transition *enter* is enabled and can be fired because it exists at least one token in each of its input places.

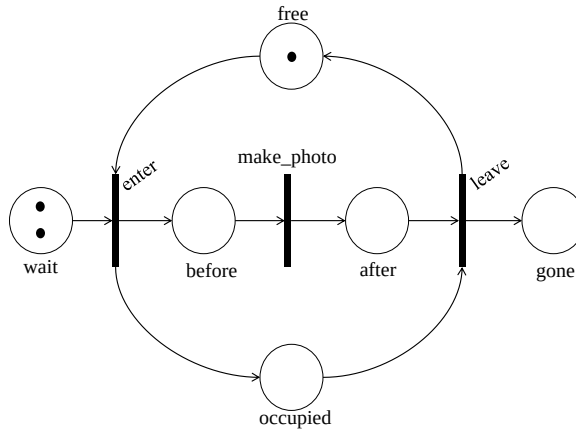


Figure 1. A Petri net for the business process of an x-ray machine

The state of a Petri net, often referred to as marking denoted by M, corresponds to a distribution of tokens over the places of the net. The notation (PN, M) is used to denote a Petri net PN with an initial marking M. Important properties of Petri nets depend directly on the initial marking of the net. Such properties are defined in [9] and are presented as follows.

- Reachability: a marking M_n is said to be reachable from a marking M_0 (initial marking) if there exists a sequence of transition firings that transforms M_0 to M_n. This property ensures if certain states will be reached or not.
- Liveness: a Petri net (PN, M) is said to be live if, and only if, for every reachable state M' and every transition t there is a state M'' reachable from M' which enables t. This property guarantees that the system is deadlock free.
- Boundedness: a Petri net (PN, M) is bounded if, and only if, for each place p there is a natural number n such that for every reachable state, the number of tokens in p is less than n. The net is called *safe* if for each place the maximum number of tokens does not exceed 1.
- Reversibility: a Petri net (PN, M_0) is said to be reversible if it is always possible to return to the initial marking through a sequence of firings, regardless of the marking considered.

2.2 WorkFlow Nets

A Petri net that models a workflow process is called a WorkFlow net (WF-net) ([19, 16]). A WF-net satisfies the following properties:

1. It has only one source place named i and only one sink place named o. These are special places such that the place i has only outgoing arcs and the place o has only incoming arcs.
2. A token in i represents a case that needs to be handled and a token in o represents a case that has been handled.
3. Every task t (transition) and condition p (place) should be in a path from place i to place o.

The formal definition of a WorkFlow net is presented below.

Definition 2 (WorkFlow net)**.** A Petri net $PN = (P, T, F)$ is a WorkFlow net if, and only if:

- There is one source place $i \in P$ such that $\bullet i = \phi$.
- There is one sink place o ∈ P such that $o\bullet = \phi$.
- Every node $x \in P \cup T$ is on a path from i to o.

A WF-net has one input place i and one output place o because any case handled by the procedure represented by the WF-net is created when it enters the workflow management systems and is deleted once it is completely handled by the workflow management systems, i.e., the WF-net specifies the life-cycle of a case. The third requirement in Definition 2 has been added to avoid "dangling tasks and/or conditions"; in other words, tasks and conditions which do not contribute to the processing of cases [19].

A business process specifies which tasks need to be performed and in which order to execute them. Modeling a business process in terms of a WF-net is quite straightforward: tasks are modeled by transitions, conditions are modeled by places, and cases are modeled by tokens [19]. An example of WorkFlow net is presented in Figure 2.

Figure 2 illustrates the workflow process which takes care of the processing of claims related to a car damage presented in [18]. The tasks required to process the claims are: *check_insurance*, *contact_garage*, *pay_damage* and *send_letter*. The tasks *check_insurance* and *contact_garage* determine whether the claim is justified. These tasks may be executed in any order. If the claim is justified, the damage is paid (task *pay_damage*). Otherwise a letter of rejection is sent to the claimant (task *send_letter*). The tasks are modeled by transitions. The two tasks *check_insurance* and *contact_garage* may be executed in parallel. Thus, there are two additional transitions: *fork* and *join*. The places *p1*, *p2*, *p3*, *p4* and *p5* are used to route a case through the procedure in a proper manner.

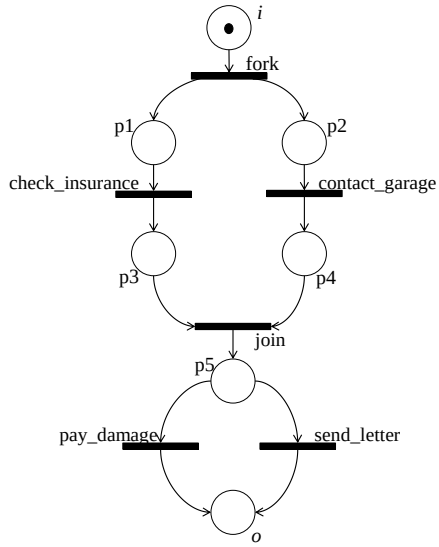


Figure 2. Example of a WorkFlow net

2.3 Soundness Property

Soundness is a correctness criterion defined for WF-nets. A WF-net is Sound if, and only if, the following three requirements are satisfied [16]:

1. For each token put in the place i, one and only one token will appear in place o.
2. When the token appears in place o, all the other places are empty for this case.
3. For each transition (task), it is possible to move from the initial marking to a marking in which that transition is enabled, i.e., there are no dead transitions.

The Soundness property is related to the dynamics of a WF-net. The first requirement states that starting from the initial marking i, it is always possible to reach the marking with one token in place o. The second requirement states that the moment a token is put in place o, all the other places should be empty. The third requirement states that for each transition t, it is possible to reach (starting from i) a marking where t is enabled [19].

Following, the formal definition of soundness property in the WF-net context, proposed in [19, 21], is presented.

Definition 3 (Soundness)**.** A process modeled by a WF-net $PN = (P, T, F)$ is Sound if, and only if:

- For every marking M reachable from marking i, there exists a firing sequence

leading from marking M to marking o. Formally:

$$\forall_M (i \xrightarrow{*} M) \rightarrow (M \xrightarrow{*} o).$$

- Marking o is the only marking reachable from marking i with at least one token in place o. Formally:

$$\forall_M (i \xrightarrow{*} M \wedge M \geq o) \rightarrow (M = o).$$

- There are no dead transitions in $(PN,\, i)$. Formally:

$$\forall_{t \in T} \exists_{M,M'} i \xrightarrow{*} M \xrightarrow{*} M'.$$

In [19], a method was proposed to verify the soundness property of a WF-net. Given a WF-net $PN = (P, T, F)$, one must decide whether PN is Sound. For this, an extended net $\overline{PN} = (\overline{P}, \overline{T}, \overline{F})$ is created. $\overline{PN}$ is the Petri net obtained by adding an extra transition t^* which connects places o and i. The extended Petri net $\overline{PN} = (\overline{P}, \overline{T}, \overline{F})$ is defined as follows ([17]):

- $\overline{P} = P$,
- $\overline{T} = T \cup t^*$,
- $\overline{F} = F \cup \{\langle p, t^* \rangle, \langle t^*, i \rangle\}$.

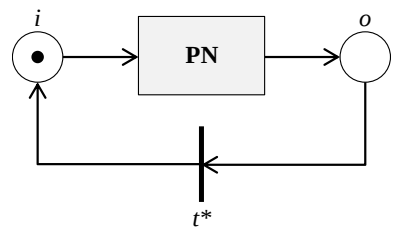


Figure 3. Example of extended Petri net

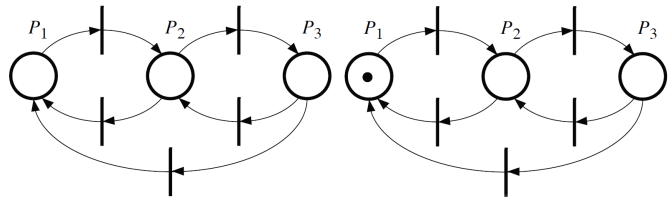
Figure 3 illustrates the relation between PN and $\overline{PN}$. The following theorem can be proven.

Theorem 1. A WorkFlow net PN is Sound if, and only if, $(\overline{PN}, i)$ is live and bounded.

The proof of this theorem can be found in [17]. Thus, the verification of the Soundness property boils down to checking whether the extended Petri net $\overline{PN}$ is live and bounded. This means that standard Petri-net-based analysis tools can be used to decide Soundness. An overview about WF-net can be found in [18, 19, 16].

2.4 State Graphs

An unmarked PN is a state graph if and only if every transition has exactly one input and one output place [11], as illustrated in Figure 4 a).



a) Example of a state graph unmarked

b) Example of a state graph marked with one token

Figure 4. Example of a state graph

A marked Petri net known as a state graph will be equivalent to the state graph in the classical sense (representing an automaton which is in only one state at a time) if, and only if, it contains exactly one token located in one of the places of the set P [11]. Figure 4 b) shows an example of a state graph with one token. It is important to note that in a state graph, the weight of all the arcs is 1.

2.5 Colored Petri Nets

Petri Nets are traditionally divided into low-level Petri nets and high-level Petri nets. Low-level Petri nets are characterized by simple tokens (natural numbers associated to places) that generally indicate the active state of a system or the availability of a resource. High-level Petri nets are aimed at practical use, in particular because they allow the construction of compact and parametrized models.

Classic Petri nets belong to the class of low-level Petri nets. They allow the representation of parallelism and synchronization, thus they are appropriate for the modeling of distributed systems. However, when classic Petri nets are used for the modeling of process, the size of very large and complex systems became an issue of major complication. In that way, one disadvantage is that classic Petri nets fall short if they are used to precisely model complex systems, making then unsuitable for the modeling of systems having large state spaces or a complex temporal behavior [15]. Then, many extensions of basic Petri net models arose from the need to represent these complex systems. One of them is the Colored Petri Net (CPN).

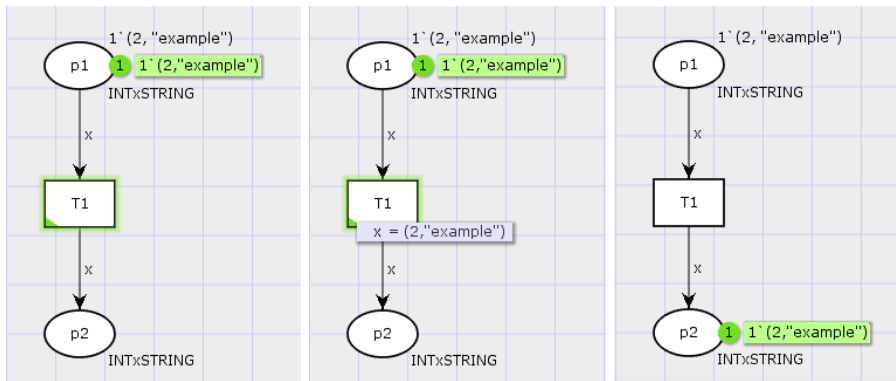
The idea of CPN is to put together the ability to represent synchronization and competition of Petri nets with the expressive power of programming languages with their data types and the concept of time. Colored Petri Nets (CPNs) belong then to the class of high-level Petri Nets, and they are characterized by the combination of Petri nets and a functional programming language [8], called CPN ML. Thus, the

formalism of Petri nets is well suited for describing concurrent and synchronizing actions in distributed systems, whereas the functional programming language can be used to define data types and manipulation of data [6].

The formal definition of CPN [6] is presented bellow.

Definition 4 (Colored Petri Net)**.** A non-hierarchical Colored Petri Net (CPN) is a nine-tuple CPN = (P, T, A, Σ, V, C, G, E, I) where:

- **P** is a finite set of places.
- **T** is a finite set of transitions T such that $P \cap T = \emptyset$.
- **A** $\subseteq p \times t \cup t \times p$ is a set of directed arcs.
- Σ is a finite set of non-empty color sets.
- **V** is a finite set of typed variables such that Type[v] $\in \Sigma$ for all variables $v \in V$.
- **C**: $p \rightarrow \Sigma$ is a color set function that assigns a color set to each place.
- **G**: $t \rightarrow \text{EXPR}_v$ is a guard function that assigns a guard to each transition t such that $Type[G(t)] = Bool$.
- **E**: $a \rightarrow \text{EXPR}_v$ is an arc expression function that assigns an arc expression to each arc a such that $Type[E(a)] = C(p)_{MS}$, where p is the place connected to the arc a.
- **I**: $p \rightarrow \text{EXPR}_\emptyset$ is an initialization function that assigns an initialization expression to each place p such that $Type[I(p)] = C(p)_{MS}$.



a) Example of a simple CPN b) During the firing of transition *T1* c) After the firing of transition *T1*

Figure 5. Elements of a Colored Petri Net

The states of a CPN are represented by means of places. Every place has a type associated which determines the kind of data that the place may contain. Each place will contain a varying number of tokens. Every token has a data value, that

is known as a color [15], and belongs to the type associated with the place. These colors do not just mean colors or patterns, they can represent complex data types [6]. Figure 5 illustrates an example of the basic elements of a CPN.

The CPN in Figure 5 a) has two places called *p1* and *p2*. The places have the type (color set) $INT \times STRING$, as well as the variable x associated to the arcs of the net. This type is formed by cartesian product of the color sets *INT* (integers) and *STRING*. The places accept only tokens of this same type. In that way, the inscription *1'(2, "example")* corresponds to one token whose attributes are given by the integer *2* and the string *"example"*.

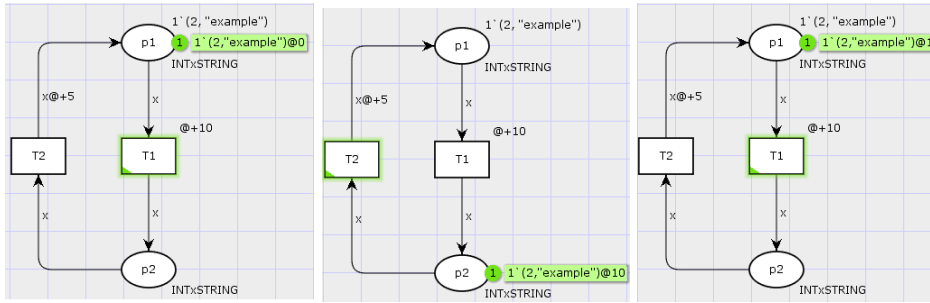
In the example of Figure 5 a) the transition *T1* will be enabled only if there is at least one token in place *p1* (input place of *T1*). During the firing of a transition in a CPN model, the variables of its input arcs will be replaced with the token value. Figure 5 b) shows that variable x was associated with the value (2, "example"). After firing *T1* the place *p2* has one token. Since there is no token in its input place, *T1* is not enabled anymore (Figure 5 c)).

CPN models allow adding time information to investigate the performance of systems. For this, a global clock used to represent model time was introduced. The clock values may either be discrete or continuous [5]. In a timed CPN model the token can carry a time value, called timestamp. To calculate the timestamps to be given to a token it is necessary to use time delay inscriptions attached to the transition or to the individual output arcs [6]. A time inscription on a transition applies a time delay to all output tokens created by that transition. On the other hand, a time inscription on an output arc applies a time delay only to tokens created at that arc [6].

To exemplify a firing transition in a timed CPN, consider the Figure 6 a). Transition *T1* represents an operation which takes 10 time units. Thus, *T1* creates timestamps for its output tokens by using time delay inscriptions attached to the transition (inscription $@ + 10$). In that way, every time *T1* is fired its output token timestamps will be increased by 10 time units. Figure 6 b) illustrates *T1* after firing. The outgoing arc to *p1* has a time delay expression $@ + 5$. Thus, the timestamp given to the tokens created on this output arc is the sum of the value of the global clock and the result of evaluating the time delay inscription of the arc, as it can be seen in Figure 6 c).

Another advantage of CPN model is that it can be structured into different related modules. The concept of module in CPN is based on a hierarchical structuring mechanism which supports bottom-up as well as top-down working style. The basic idea behind hierarchical CPN is to allow the modeler to construct a large model by combining a number of small CPN into a single model [5]. According to [15] it facilitates the modeling of large and complex systems, such as information systems and business processes.

CPN hierarchy also offers a concept known as fusion places. This concept allows the modeler to specify that a set of places are considered to be identical. Such places are called fusion places and a set of fusion places is a fusion set. Anything that happens to one place of the set also happens to the other places of the set.



a) Initial marking of a timed CPN model b) After the firing of transition *T1* c) After the firing of transition *T2*

Figure 6. An example of a timed Colored Petri Net

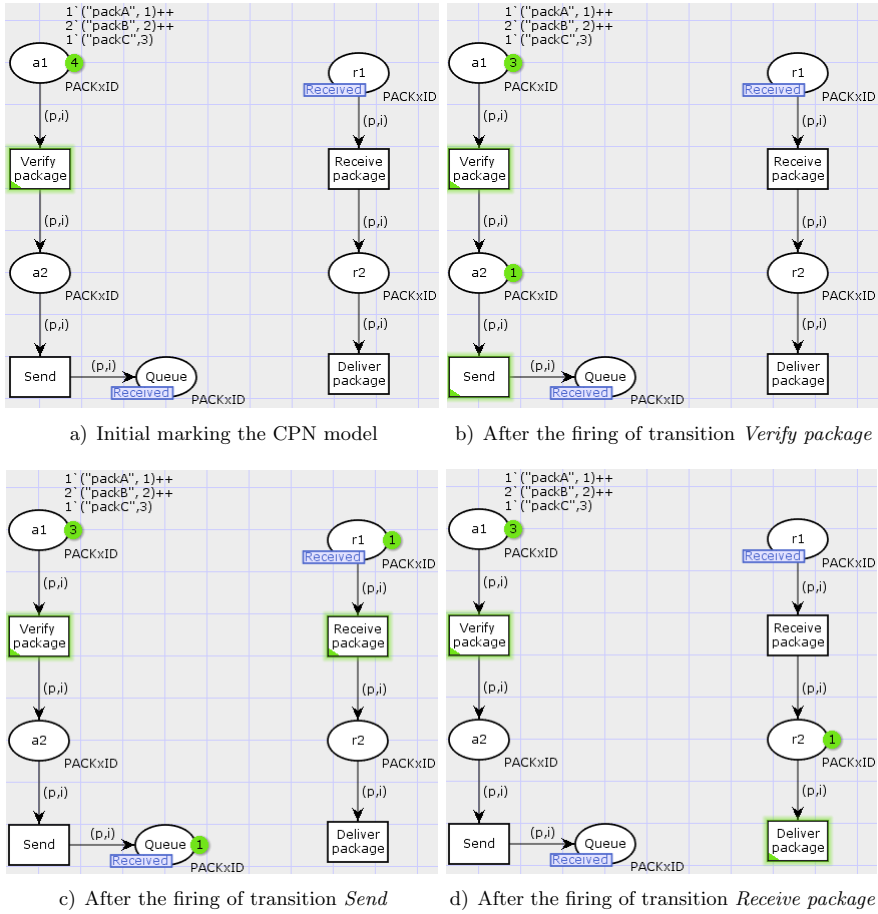
Thus, when a token is added/removed from one of the places, an identical token will be added/removed in all the other places of the fusion set.

Figure 7 illustrates an example of CPN with fusion places. This model represents a procedure where packages are verified and sent to another procedure to be delivered. Figure 7 a) shows the initial marking of the net. After firing the transition *Verify package*, the package will be sent, represented by transition *Send* (Figure 7 b)). After firing the transition *Send*, the token will be added in place *Queue* which represents a queue. The places *Queue* and *r1* are defined as fusion places and they are marked with a fusion tag named *Received*. This tag represents to which fusion set these places belong. Thus, if *Queue* has one token, the place *r1* will also have one token as illustrated in Figure 7 c). After firing transition *Receive package*, the token in *r1* will be consumed (Figure 7 d)) and the transition *Deliver package* can be fired.

When all members of a fusion set belong to a same page (the same part of a CPN model) and that this page only has one instance, a fusion place is nothing more than a drawing convenience that allows the user to avoid too many crossing arcs in his visual model [6]. Thus it is possible to simplify the net graphical structure without changing its meaning.

The practical application of CPN modeling and analysis heavily relies on the existence of computer tools supporting the creation and manipulation of CPN models. CPN Tools [5] is a tool suited for editing, simulating and providing state space analysis of CP-net models.

In this paper, the CPN Tools is used to represent graphically and analyze the proposed models, and to simulate the timed versions of the model. By performing analysis and simulation of the proposed models, it will make possible to investigate different game scenarios and to explore qualitatively and quantitatively the global behavior of the game.



a) Initial marking the CPN model

b) After the firing of transition *Verify package*

c) After the firing of transition *Send*

d) After the firing of transition *Receive package*

Figure 7. An example of a Colored Petri Net with fusion places

3 RELATED WORK

In [10], a new method to aid in the creation process of video games is presented. The authors approach models to represent the logic of the game (sequence of actions) and virtual space. To model the logic of the game, the authors used the formalism of Petri nets. In this approach, a Petri net called the Transaction Net determines the beginning and the end of each player action. Thus, each model represents a set of activities that can be performed by the player in the game. To model the virtual space, the authors used hyper-graphs. Each node of the hyper-graph represents a region of the game where the actions are performed, and the hyper-edges represent the paths between those regions. In order to unite the two structures and represent the game in a global context, the authors created a mechanism called

connections. This mechanism attempts to replace a hyper-edge of the hyper-graph by the reachability tree of a Petri net. Each time a task is performed, a place on the topological map is released and the hyper-edge replaced. The verification of the game model is treated according to each formalism.

In [3] a new approach based on a particular type of Petri net, called WorkFlow net, is presented to specify existing scenarios in a game. In this approach, the authors used a WorkFlow net to represent the flow of activities that must be performed by the player in order to achieve a specific goal in the game. This flow of activities is associated with the notion of quest, i.e., a mission that the player must perform. According to [3], in terms of the model, each quest is a subprocess of a larger WorkFlow Net. The integration of several quests form a net of quests and it is through this that the overall result of the analysis of the game model is established. As each quest is a subprocess, in case of a change in a quest already studied, a new study of good properties will be done only for the quest that has changed. In this approach, the authors performed a qualitative analysis using linear logic. According to the authors, the translation of the models into linear logic trees has the objective of proving the soundness property of the net that corresponds to the consistency of the scenario modeled from the point of view of the game.

The concept of Petri nets in game modeling is also used in [12] to present an approach that operates in the early stages of game design, detecting structural errors in singleplayer and multiplayer games. To differentiate the singleplayer games from multiplayer, [12] used Colored Petri nets that can assign types to the tokens and use guard functions in the transitions. All elements of the game are then mapped to appropriate Petri net constructors. For example, the rooms of the game (e.g., a room or a depot) are represented by places in the Petri net, as are actions (e.g., opening a door) and variables representing boolean values. Players are represented by the tokens and the events of the game by the transitions. The creation of the Petri net is added as an extension of the StoryTec tool [12]. Thus, any game that is created with this tool can be automatically transformed into a Colored Petri net. The generated net is then exported to the XML format (Extensible Markup Language) that can be read by the CPN Tools. According to [12] when model analysis is done in CPN Tools it is possible to identify structural errors like deadlocks (situations in which players cannot change the state of the game), livelocks (situations in which players can change the state of the game but cannot reach the end), unreachable scenes (situations in which players cannot reach a scene under any circumstances) and impossible actions (situations in which actions never can be achieved due to unsatisfied conditions).

Most of the work related to this research show the use of Petri nets as efficient modeling language to formally specify and analyze video games. The approach presented in [10], for example, is interesting as it defines diagrams to represent aspects of the game that are important for its creation process, thus facilitating the study of gameplay. However, verification happens in the logical model or in the topological map since they are distinct formalisms that do not allow the integration of the two models in a single view. In the work presented in [3] the authors did

not present a model to represent the virtual world map of the game where the player's actions are performed. In addition, the authors presented a qualitative analysis using a formalism different from the Petri nets. Finally, in [12], despite the automatic generation of the model, this approach presented produced a complex model. Thus, the model needs to go through optimization strategies to deal with the state explosion problem and be formally analyzed.

4 MODELING GAME LEVELS

The video game Silent Hill II [7] will be used to illustrate the approach presented in this paper. The modeling method presented in this section uses the case of a real game. However, the same modeling principle can be applied to any singleplayer game that has activities and the presence of a virtual topology.

4.1 Logical Model

For the representation of the logical model, it is necessary to consider the concept of level. The use of the term level in video games has been employed for a long time and, generally, can be approached in two ways. The first associates the term level with the difficulty of the game phase. The second associates level to a certain stage of the game. In this work, the second approach is used. Thus, a level is considered a part of the game.

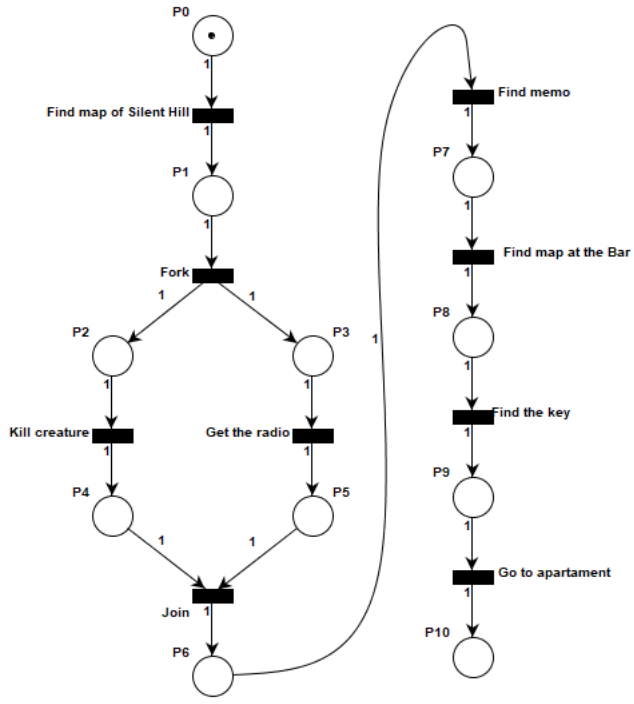
Most games can be structured into a group of levels. Every game has one main goal that the player must reach in order to win and each level has a specific goal associated to the level. To achieve this goal the player must perform a certain sequence of tasks. These tasks must be performed by the player sequentially or simultaneously to a certain extent. In addition, some tasks are mandatory and some optional. After all level tasks are performed correctly, the goal is reached and the player can go to the next level.

Since a game consists of several levels and a level consists of a set of activities, the WorkFlow nets seem suitable to produce a logical model of a game level. Indeed, a game level is similar in its structure to the classic representation of a workflow process. Both have a beginning and a final goal that will be reached after performing some activities. Thus, WorkFlow nets will be well adapted to model the routing structure of the activities a player will have to performed in a specific level of a video game.

To model a game level, it is necessary to identify first the activities of the level. To complete the first level of Silent Hill II, the player must perform the following sequence of activities:

1. Find the Silent Hill map on the parking observation deck.
2. Kill the creature and get the radio at the tunnel.
3. Find the memo at the trailer.

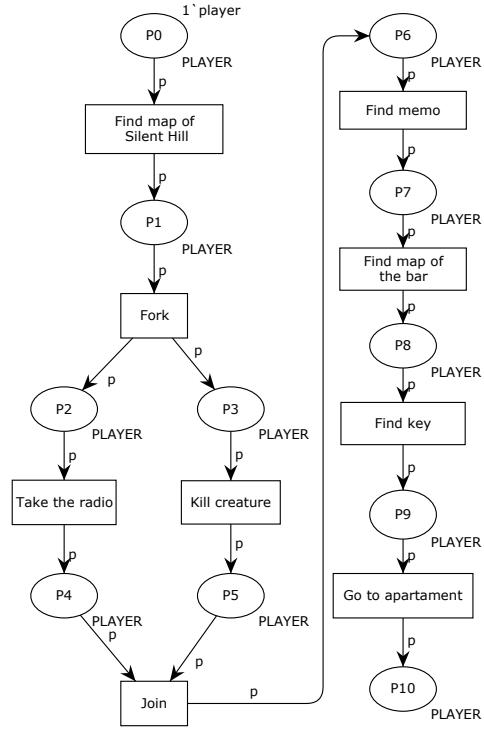
4. Find the second map at Neely's Bar.
5. Find the key on a corpse at Martin street.
6. Go to Wood Side Apartment.



a) Logical model of the first level of Silent Hill II

After performing these activities, the player will complete the first level and will be able to move to the next level. The WorkFlow net in Figure 8 a) represents the logical model of the first level of the game.

The activities are associated with the transitions and the conditions with the places. The initial place is *P0* and represents the beginning of the first level. The token in *P0* represents the player. The place *P10* represents the end of the level. The first task that the player has to perform is *Find map of Silent Hill*. The tasks *Get the radio* and *Kill creature* belong to a parallel routing. That means the player can execute them in any order. *Find memo*, *Find map of the bar*, *Find key* and *Go to Apartment*, are tasks which have to be executed in sequence and then belong to a sequential routing of the WF-net. *Go to Apartment* corresponds to the last task of the first level. After completing all the tasks, the player ends the first level, reaching the final place *P10*.



b) Logical model of the first level of Silent Hill II represented by a Colored Petri net model of the CPN Tools

Figure 8. Logical model of Silent Hill II

Figure 8 b) shows the Colored Petri net model of the Workflow net presented in Figure 8 a) and adapted to be directly implemented on the CPN Tool. In the presented approach, the same color set *PLAYER* is associated to all places of the model. The inscription *player* then represents the value attached to the token that represents a specific player in the corresponding level. The variable p associated to the arcs of the model belongs to the same type *PLAYER* and can receive tokens of the same color set. Thus, the token *1'player* can go through the entire CPN, from the beginning to the end, representing the evolution of the player in the game.

In most of games, there exist activities based on finding objects, solving puzzles and interacting with game characters named NPC (Non-Player Character). Some specific interactions propose challenges where the player must win a competition against a NPC in order to perform the next activity. In case of failure (death of the player during a fight for example) the player can have to perform the same activity more than once. In the first level of the Silent Hill II, such a situation exists. The activity *Kill creature* (Figure 8 b)) consists in fighting a monster (a NPC). If the

player does not win, he will have to perform the same activity over again, i.e., the player will have to fight repeatedly against the monster until he manages to defeat him.

The logical model of the level, the activity *Kill creature* is part of a parallel route where the other activity is *Take the radio*. Thus, these activities can be executed in any sequence. In particular, four option can be considered:

1. The player successfully performs the activity *Take the radio* first, and then the activity *Kill creature*. After that, he moves to the next activity following the game flow;
2. The player successfully performs the activity *Kill creature* first and then the activity *Take the radio*. After that, he moves to the next activity following the game flow;
3. The player performs the activity *Kill creature* and loses. As a consequence, the game is restarted to the checkpoint that corresponds to the conclusion of the previous activity (*Find map of Silent Hill*);
4. The player successfully performs the activity *Take the radio* first, and then does not manage to complete the activity *Kill creature*. As a consequence, the game is restarted to the checkpoint that corresponds to the conclusion of the previous activity (*Find map of Silent Hill*).

The Figure 9 illustrates the final logical model of the level. The activities *Take the radio* and *Kill creature* are represented by the transitions of the same name. The transition *L2* represents the option 3 and the transition *L1* the option 4. *L2* is fired when places P2 and P3 are marked (which corresponds to option 3). *L1* is fired if a token is in *P3*. After the firing of *L1*, *L3* is fired and a token is produced in place *P11* and in place *P4* (which corresponds to option 4). Such a control structure corresponds to a kind of iterative route in a Workflow net.

It is important to note that each activity of a game takes a minimum duration to be performed. The duration depends, basically, on the player experience. More experienced players tend to perform the game challenges more easily than the less experienced ones. A player experience can come from experience based on previous games or after repeating the same challenge over and over again. Thus, the more a player executes the proposed activities, the more experience he will obtain. After gaining some experience, it is more likely that a player will perform given activities again in a most efficient way, spending less time every time he runs it.

The duration of each game activities is important because from it, it will be possible to calculate the average time required for a level to be completed by a player. Therefore, it is necessary to represent the duration of the activities on the logical model in an explicit manner. For this, a random time function associated to each activity of a level will be considered. Such a function will simulate then the duration a player needs to perform a specific activity.

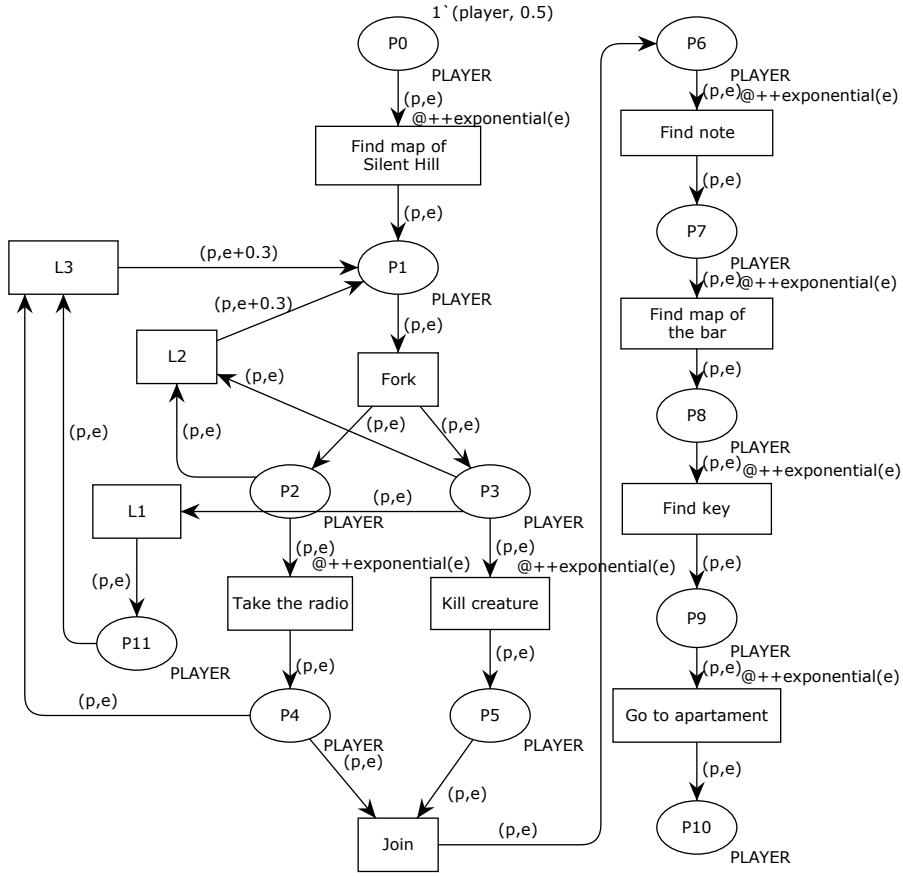


Figure 9. First level of Silent Hill II with an iterative activity

In the proposed approach, a negative-exponential probability distribution will be used. This function is one of the most widely used random distribution functions for simulating inter-arrival times in most simulation problems [6]. The function is based on an exponential parameter r, where r is a positive real number. The CPN Tool function *exponential*(r) produces a value based on the distribution exponential with mean $1/r$, for $r > 0$.

In Figure 9, the exponential function is associated with each activity. The parameter of the function is given by the real parameter e. This parameter is based on the experience of the player. Initially, e has the value 0.5[1]. This value may change

[1] At first, this value was defined only for test purposes. As future work, it will be interesting to consider a more accurate study based on statistical data to determine e with more precision.

according to the evolution of the game. The inscription $@++exponential(e)$ assigns a delay to the token used to fire the transition corresponding to the considered activity.

When a player repeats a same activity, as it is the case with the activity *Kill creature* of the logical model, his experience increases. Such a statement is expressed by the inscription $(p, e + 0.3)$ associated to the outgoing arcs of transitions *L3* and *L2*. The value of the parameter e is increased by 0.3. In that way, the duration (mean duration) to perform the activity *Kill creature* each time the player is killed will be shorter, expressing the accumulated experience.

4.2 Topological Model

It is in the virtual world of a game that activities are performed. Thus, it is also important to describe the topological properties of the virtual game world along with the evolution of the player within it [10].

A game has a set of areas where the player has to fulfill specific challenges. These areas do not change during the game.

It was in [10] that a formal definition of the topological map concept existing in video games appears for the first time. The model proposed by the authors was based on a kind of graph with the possibility of adding dynamically some arcs connecting adjacent areas after the player managed to liberate some kind of passage. The problem with this kind of approach is clearly the difficulty of formally implementing communication mechanisms between the logical model and the topological model. In particular, when the logical model is represented by a Petri net, it will be particularly difficult to implement formal communication mechanisms between the logical model and a topological model represented by a kind of graph without the notion of dynamic marking that exists in Petri net models. Therefore, in order to produce formal communication mechanisms between the logical and the topological model of video games, the approach presented in this article considers the representation of the topology of the virtual world of the game using a kind of Petri net model called state graph.

The semantic of a state graph makes the modeling of the topological map of a game easy. Each region of the virtual world is called an area. In the corresponding state graph, a specific area is modeled by a specific place. The boundaries between areas are represented by simple transitions. The location of the player is then represented by a token in a specific place and the arc orientation represents in which direction the player may go between two adjacent areas.

To illustrate this approach, the map of the first level of Silent Hill II in Figure 10 can be considered. The numbers represent special subsets of the virtual world where the player has to fulfill specific tasks. These regions do not change during the game. They are named as follows:

1. Observation deck,
2. Forest,

3. Church,
4. Backyard,
5. Tunnel,
6. Trailer,
7. Bar,
8. Martin Street,
9. Wood Side Apartment.

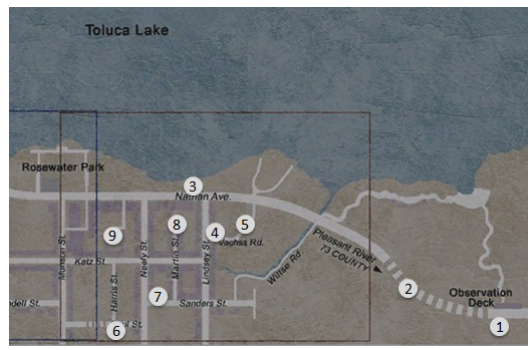


Figure 10. Virtual world areas of the first level of Silent Hill II

In Figure 11, the formal representation of the topological world of the first level of Silent Hill II is presented. Each region of the map is represented by places of the same name in the state graph. The transitions represent the boundaries between adjacent areas. The token in place *Observation Deck* represents the current location of the player at the beginning of the game. In the topological map, firing a transition means that the player moves from one area to another.

Figure 12 shows the same topological world of Figure 11, but adapted to the Colored Petri net of the CPN Tools. In Figure 12, all the areas of the game have the type *PLAYER* and the token is represented by the inscription *1'player* at the place *Observation Deck*.

On the topological map, a transition firing means that the player moved from one area to another. He can pass through areas quickly or slowly. Thus, minimum and a maximum durations for passing from one area to another have to be considered in the time model. To simulate the player's movement on the map, a random time function is then added to each transition of the state graph.

The function $uniform(a, b)$ produces a random number between parameters *a:real* and *b:real*. The probability distribution is uniform, i.e., any value between parameters a and b has the same probability to accur. For $b > a$, $uniform(a, b)$ produces then a value from a uniform distribution with mean $(a + b)/2$ [15]. The

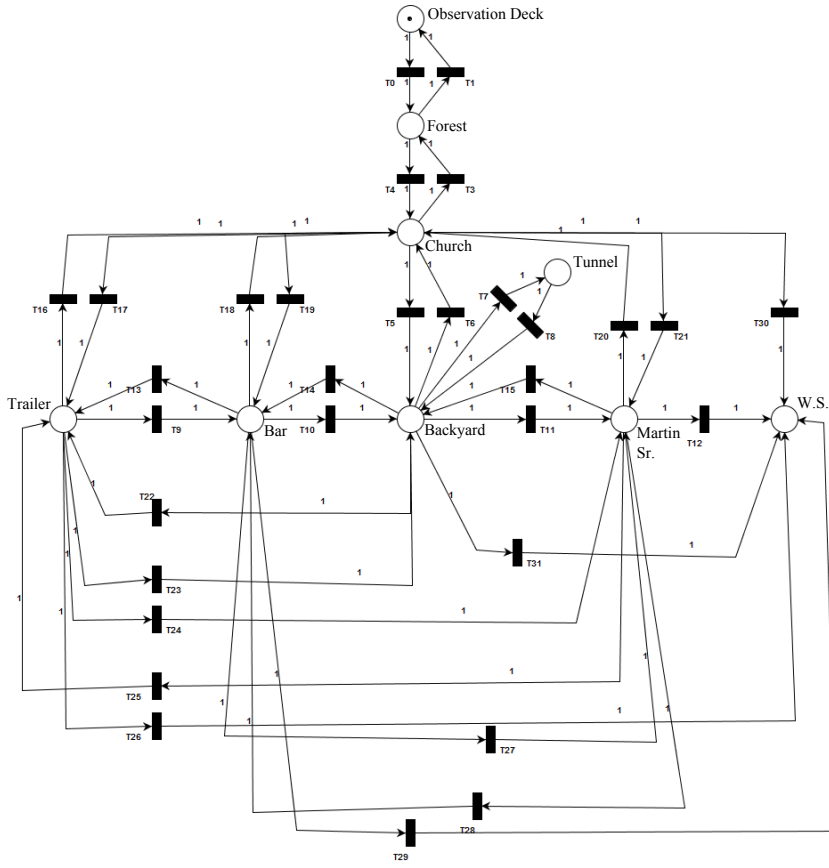


Figure 11. Topological map of Silent Hill II represented by a State Graph

uniform function seems suitable to represent the duration that exists on the topological map as it has a minimum and maximum parameter, which seem consistent with how long a player will spend while moving from one area to another.

Figure 13 illustrates the time topological model. In order to keep the model as clean as possible, the inscription $@++Time()$ was associated with each transition of the topological model. $Time()$ corresponds to a simplified notation for the function $uniform(a, b)$ and assigns a delay to the token used to fire the transition. The two parameters, a and b, and the random value produced, are all real numbers. The chosen parameters for this example are[2] $a = 0.1$ and $b = 1.0$. In that way, in case of

[2] These parameters can be updated according to the specifications of the game designer.

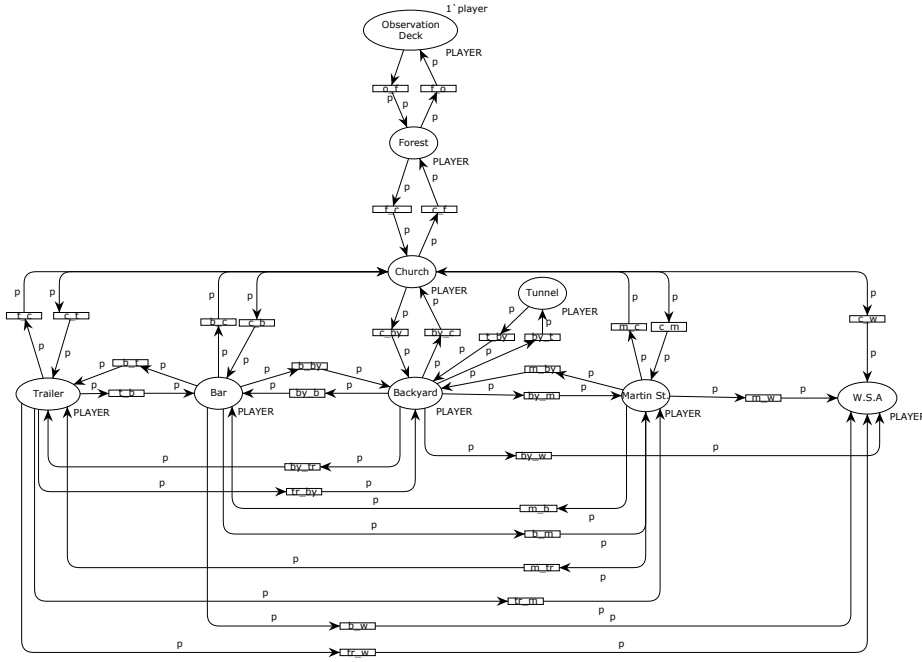


Figure 12. Topological map of the first level of Silent Hill II adapted to CPN Tools

adjacent areas, the player will spend between 0.1 and 1.0 time units to move from one area to another.

4.3 Communication Mechanisms

Generally, at the beginning of the game the player cannot access all the connected areas. To pass from one area to another the player has to respect some requirements (conditions) that will depend on some of the activities associated to the logical model of the level. Such conditions in most of the cases will correspond to find an object (like a key for example) necessary for passing from one area to another. Eventually, some conditions will simply correspond to the completion of a task the player must perform (like kill a creature for example), not necessarily producing a specific game item. In any case, such situations will imply in a kind of communication between the logical model and the topological model.

For example, in the game Silent Hill II, to pass from the area *Observation Deck* to the area *Forest*, the player must find the map of Silent Hill. Thus, the first activity on the logical model is *Find the map*. The player can only performed this activity if he is in the corresponding area, which is *Observation deck*. Thus, it is necessary to establish a communication mechanism between both models in order to guarantee that the player is at the right place (where he must be to perform the activity) at

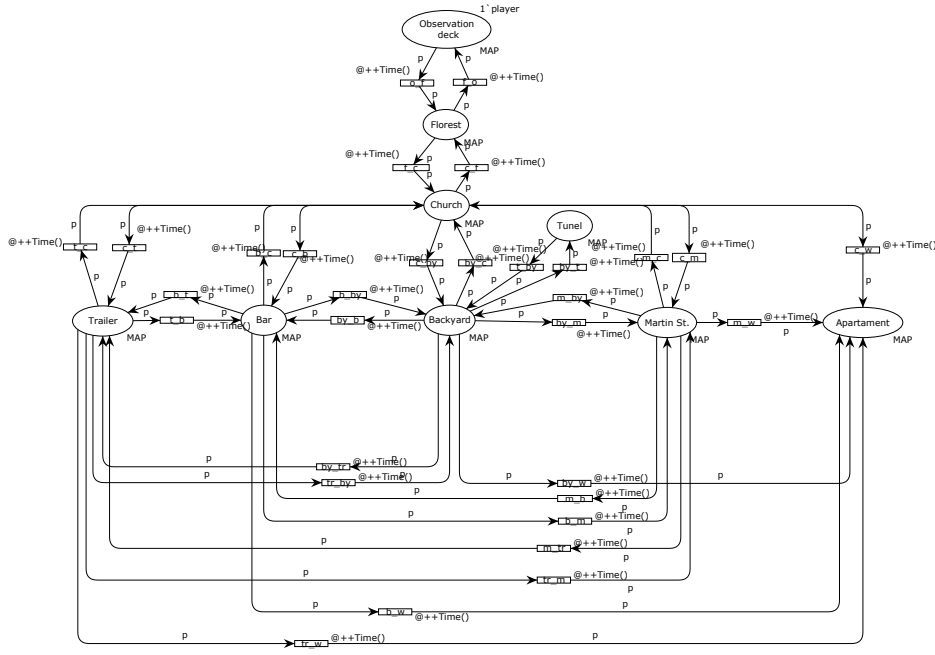


Figure 13. Timed topological model of the first level of Silent Hill II

the right time (when he must perform the activity in accordance with the logical model which fixes the sequence of activities of the level).

Figure 14 is an example of communication mechanism between the logical model and the topological model. To execute the first activity *Find map of Silent Hill*, the player needs to be on the area *Observation deck* (the place *Observation deck* must be marked). After the completion of the activity *Find map of Silent Hill*, the first condition of the level is verified. The first condition of the game is represented by the places *Map found* (in the logical model) and *Condition 1: find the map* (in the topological model). Once with the map, the player will be able to pass from *Observation Deck* to *Forest*. After a condition becomes true (a token produced in a condition place), the corresponding condition place will continue marked all the time. This means that once a passage between two adjacent areas is liberated, it will continue open until the level is completed.

Such a procedure to link both models can be seen as a kind of synchronous communication mechanism. In order to keep both models distinct (at least visually), an interesting approach is to use the fusion place concept proposed by the CPN Tools. Fusion places in our approach are used to visually implement the communication mechanisms between the logical model and the topological map, as shown in Figure 15. Figure 15 is then equivalent to Figure 14 using the CPN Tools concept of fusion places, maintaining both models distinct.

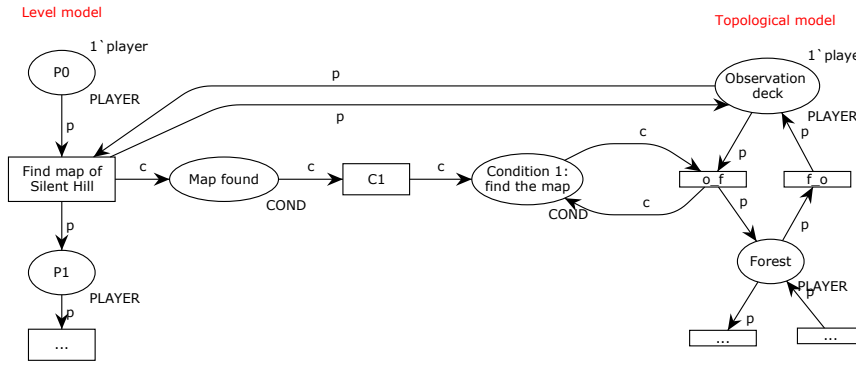


Figure 14. Communication mechanism between the logical model and the topological model

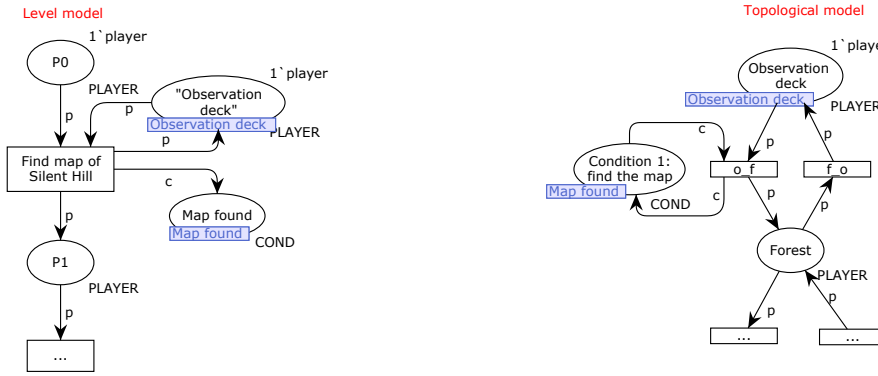


Figure 15. Communication mechanism implemented with the fusion place concept

To distinguish condition places from other places in the model, the color set *COND* was created. This type is associated with all places that represent a condition in the topological model. In Figure 15, the places *Map found* and the place *Condition 1: find the map* have the type *COND*. These places are marked with the tag *Map found* (represented by a blue rectangle). The members of a fusion set have the same fusion tag. When a token is produced in one of the places of a same fusion set, the same token is then produced in all other places of the fusion set. For example, in Figure 15, after the firing of transition *Find map of Silent Hill*, a token is produced in the place *Map found* of the logical model. The same token is then automatically reproduced (cloned) in the corresponding place *Condition 1: find the map* of the topological model.

Figure 16 represents both models (logical and topological) with their synchronous communication mechanisms of the first level of Silent Hill II. It is important

to notice that both models, as well as the communication mechanisms, use a same formalism (a Colored Petri net); then, it will be possible to implement some kind of qualitative and quantitative analysis techniques on the global model (logical + topological) of the level of the game.

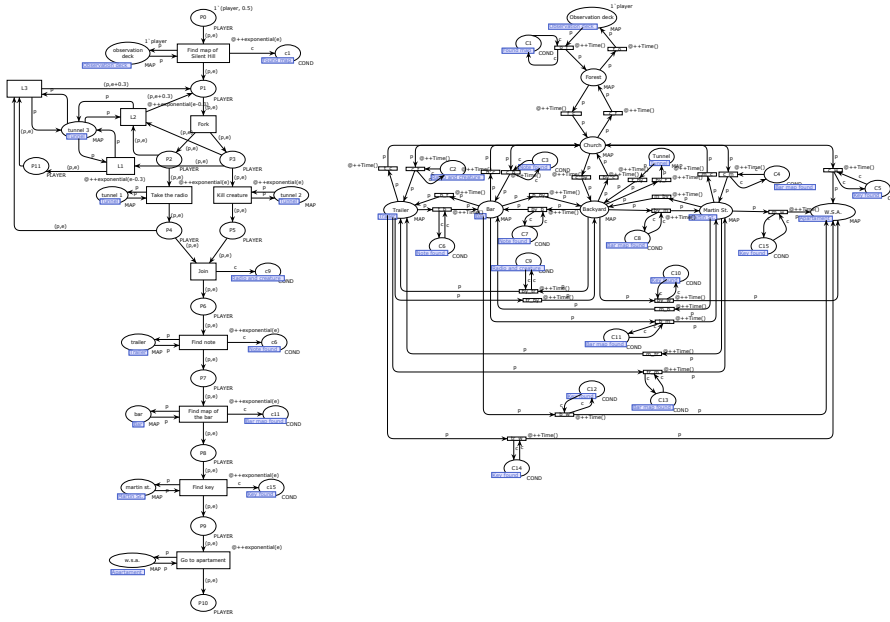


Figure 16. Full communication between the logical model and the topological model. The left side of the figure represents the activities of the first level of Silent Hill II. The right side represents the areas of the virtual world.

5 GAMEPLAY ANALYSIS

One of the advantages of using Petri nets is the existence of various analysis techniques. It is possible to analyze the state space of a Petri net by applying reachability analysis and to use some techniques to investigate some good properties of a Petri net. Simulation can also be used, and the emphasis in that kind of validation is to detect errors and to increase confidence in the correctness of a system [6].

In order to analyze the model of a game, two analysis techniques are used in this work. The first (qualitative analysis) is based on the analysis of the state space of the model. The second one (quantitative analysis) is based on simulation.

5.1 Soundness Verification

The idea behind a state space is to construct a graph which has a node for each reachable marking and an arc for each occurring binding element [5]. In other words, a state space represents all possible executions of the system under consideration and can prove that the system contains a certain formally specified property [6].

To construct the state space, the approach proposed by [19] (presented in Section 2.3) was used. This approach consists of creating an extended Petri net $\overline{PN}$ obtained by adding an extra transition $t*$ which connects the end places of the models to the start places of the same models. If $\overline{PN}$ is live and bounded, then the corresponding PN model can be considered Sound from the point of view of game modeling. The Soundness criterion will correspond then to the correctness of the model. Thus, for applying such a method of analysis, some modifications to the model presented in Figure 16 are needed.

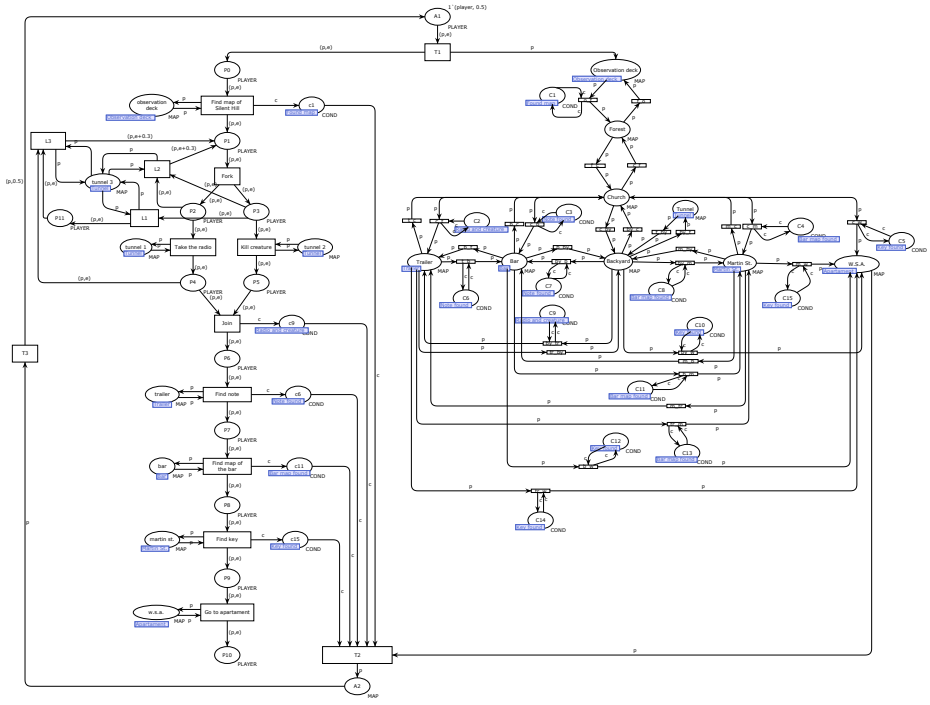


Figure 17. Global model for analysis

The modified model for analysis is presented in Figure 17. The time functions were omitted from the model since the notion of time is unnecessary for traditional state space exploration in the Petri net theory. Common *start* and *end* places, named *A1* and *A2* respectively, are created. The transition *T1* is a fork which produces one token in the start place of the level model (*P0*) and one token in the place *Observation deck* that represents the area of the game where the player will be at the beginning of the level. The transition *T2* is a join which has the purpose to consume the tokens that, in the Sound case, must be present at the end place of the logical model, at the place representing the area where the player will be at the end of the level, and in the marked condition places common to both models (used as places communication between the logical model and the topological model). The firing of transition *T3* will reinitiate the augmented model only if the non augmented model is Sound (all activity transitions of the logical model can be achieved by the player who can normally explore the various areas of the topological map). In practical terms, the augmented model will be reinitiated if there is no dead transition and no token duplication.

The qualitative analysis of the models is implemented using the state space analysis functionality of the CPN Tools. This functionality allows to record the results of the analysis in a report file. The first part of the report corresponds to the statistical information obtained after applying the state space analysis on the augmented model of Figure 17. The SCC Graph (Strongly Connected Components) indicates that there exists only one strongly connected component in the obtained reachability graph after applying state space analysis. The second part of the state space report contains information about the boudedness property. According to the boundedness report, the game model (logical model + topological map) is bounded. The last part of the report indicates the live transition instances. All the transitions of the augmented model are live.

According to the state space report, the augmented Petri net model is bounded and live. Thus, according to the theorem presented in Section 2.3, the non augmented model is Sound. From the point of view of the game, it means that all activities will be performed eventually and all areas of the topological map will be accessed by the player, what is as a matter of fact expected from a video game. The second analysis of the models is based on simulation.

5.2 Simulation

Simulation supports validation [15]. It can be used to explore a finite number of executions of the system under consideration. Thus, simulation is suitable for detecting errors and for obtaining increased confidence in the correctness of a system [6]. The CPN Tools simulator supports interactive and automatic simulation. An interactive simulation provides a way to investigate different scenarios in detail and check how the model works. In the automatic mode, simulation can be performed in play mode or fast forward mode. The end of a simulation is determined by simulation stop criteria and produces a simulation report.

In this paper, the automatic simulation is used in order to estimate the time a player will need to complete a level of a game. All the simulations presented are performed automatically 10 times. Therefore, a replication functionality of the CPN Tools, expressed by the inscription *CPN'Replications.nreplications 10*, is used.

First, the logical model, presented in Figure 9, is simulated alone (without considering the topological model). Figure 18 shows the simulation report. The minimum duration for performing all activities of the level corresponds to approximately 8.54 time units, and the maximum duration corresponds to approximately 23.41 time units. The mean time to performed all tasks of the level corresponds to approximately 13.76 time units.

```
Simulation no.: 1
Steps.........: 9
Model time....: 8.54950963137
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 2
Steps.........: 9
Model time....: 15.30329904
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 3
Steps.........: 9
Model time....: 9.77368666598
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 4
Steps.........: 15
Model time....: 17.0892393767
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 5
Steps.........: 17
Model time....: 23.4159232825
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 6
Steps.........: 9
Model time....: 13.9048727582
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 7
Steps.........: 15
Model time....: 8.68495576436
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 8
Steps.........: 13
Model time....: 15.4814450027
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 9
Steps.........: 9
Model time....: 15.6245852868
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds

Simulation no.: 10
Steps.........: 9
Model time....: 9.91201704388
Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds
```

Figure 18. Simulation report of the Timed Logical Model

Figure 19 illustrates the simulation report of the topological model alone (presented in Figure 13). The topological model represents all the areas of the first level of Silent Hill II. According to the simulation report, the minimum duration to cover all the areas of the map corresponds to approximately 1.17 time units. In contrast, the maximum duration to cover all the areas of the map corresponds to approximately 10.50. The mean duration corresponds to approximately 5.01 time units.

In a game, the player can only perform an activity if he is in the appropriate area. And some areas are accessed only after performing a specific activity. It is necessary then to keep both models together with the existing interaction represented by the communication mechanisms to estimate the global duration of the level in

```
Simulation no.: 1                          Simulation no.: 6
Steps.........: 7                          Steps.........: 18
Model time....: 3.10039547966              Model time....: 10.5031058175
Stop reason...: No more enabled transitions! Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds          Time to run simulation: 0 seconds

Simulation no.: 2                          Simulation no.: 7
Steps.........: 4                          Steps.........: 8
Model time....: 1.17036978075              Model time....: 4.20277828118
Stop reason...: No more enabled transitions! Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds          Time to run simulation: 0 seconds

Simulation no.: 3                          Simulation no.: 8
Steps.........: 11                         Steps.........: 4
Model time....: 6.05321861787              Model time....: 1.83511203897
Stop reason...: No more enabled transitions! Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds          Time to run simulation: 0 seconds

Simulation no.: 4                          Simulation no.: 9
Steps.........: 11                         Steps.........: 15
Model time....: 5.32683792618              Model time....: 8.49571707258
Stop reason...: No more enabled transitions! Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds          Time to run simulation: 0 seconds

Simulation no.: 5                          Simulation no.: 10
Steps.........: 7                          Steps.........: 12
Model time....: 4.17942236777              Model time....: 5.24148291281
Stop reason...: No more enabled transitions! Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds          Time to run simulation: 0 seconds
```

Figure 19. Simulation report of the Timed Topological Model

a more realistic way. The timed global model presented in Figure 16 is simulated. According to the replication report in Figure 20, the minimum duration of the level corresponds to approximately 59.13 time units, and the maximum duration corresponds to approximately 210.44. The mean duration of the level corresponds to approximately 87.16 time units.

When the models are simulated separately, the estimated duration refers only to the property of a specific model that expresses only one vision of the game (activities or map). This does not happen when considering the global model because one model influences the other. For example, when the topological model is simulated alone, all areas of the map can be accessed. This is not possible when the global model is simulated because the player needs to perform certain activities to obtain access to some areas. When the activity model is simulated alone all the activities can be performed almost immediately, which again is not possible when the global model is simulated. In fact, the player needs to be in a specific area of the map to perform a specific activity of the game and need a certain time to move from one area to another. In this way, both models need to be considered together by means of a communication mechanism in order to produce a realist simulation and obtain an accurate estimate time when considering the level of a video game.

```
Simulation no.: 1                    Simulation no.: 6
Steps.........: 124                  Steps.........: 368
Model time....: 74.8600106922        Model time....: 210.443008263
Stop reason...: No more enabled transitions!   Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds    Time to run simulation: 0 seconds

Simulation no.: 2                    Simulation no.: 7
Steps.........: 153                  Steps.........: 87
Model time....: 94.7624598182        Model time....: 59.6436906794
Stop reason...: No more enabled transitions!   Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds    Time to run simulation: 0 seconds

Simulation no.: 3                    Simulation no.: 8
Steps.........: 97                   Steps.........: 174
Model time....: 59.1337012618        Model time....: 105.075118921
Stop reason...: No more enabled transitions!   Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds    Time to run simulation: 0 seconds

Simulation no.: 4                    Simulation no.: 9
Steps.........: 117                  Steps.........: 110
Model time....: 70.8650821122        Model time....: 65.2153802654
Stop reason...: No more enabled transitions!   Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds    Time to run simulation: 0 seconds

Simulation no.: 5                    Simulation no.: 10
Steps.........: 114                  Steps.........: 109
Model time....: 71.4459040729        Model time....: 60.1723798904
Stop reason...: No more enabled transitions!   Stop reason...: No more enabled transitions!
Time to run simulation: 0 seconds    Time to run simulation: 0 seconds
```

Figure 20. Simulation report of the Global Model

6 CONCLUSION

This article presented the modeling and analysis of video games using the formalism of Petri nets. A video game is composed of activities and a virtual world. To represent the activities of a game the WorkFlow net was used. The areas of the virtual world (topological map) was represented by a state graph. A timed version of the model was presented too. In particular, random time functions were used to simulate the time a player needs to complete a level of a game. The fact of using a same formalism (Petri Nets) for both models (logical model + topological model) permitted to consider a kind of synchronous communication mechanism used to show the influence of one model over the other. The software CPN Tools was used to implement the approach.

Two forms of analysis were presented. A qualitative analysis based of a state space construction was presented to verify the Soundness of the model. The main purpose of the state space analysis is to show that all the activities of the level model can be executed eventually and that all the areas of the virtual world can be accessed by the player. A quantitative analysis was presented to calculate the game play duration. The simulation results show in particular the influence that a model has over the other, thus making it possible to produce the effective duration a player will need to complete a specific game level.

Comparing this approach with other works dealing with game modeling, its main advantage is that the use of the same formalism to specifying different views (logical + topological) of a game allows the use of the CPN Tools to implement a kind of qualitative and quantitative analysis of the gameplay. In this approach,

there is no need to translate the model to another formalism or deal with the problem of state explosion. In addition, such an approach has the advantage to verify the correctness of a video game still in the level design phase (before its implementation). In that way, the CPN Tools was an interesting option as it is capable of producing graphical models and provides efficient analysis functionalities.

As a future work proposal, it will be interesting also to investigate the behavior of multiplayer games and to model the interactions that exist between several players that collaborate to reach a common goal.

REFERENCES

[1] Ang, C. S.—Rao, G. S. V. R. K.: Designing Interactivity in Computer Games: A UML Approach. International Journal of Intelligent Games and Simulation, Vol. 3, 2004, No. 2, pp. 62–69.

[2] Araújo, M.—Roque, L.: Modeling Games with Petri Nets. Proceedings of the 2009 DiGRA International Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory (DiGRA 2009), London, UK, 2009.

[3] de Oliveira, G. W.—Julia, S.—Passos, L. M. S.: Game Modeling Using Workflow Nets. 2011 IEEE International Conference on Systems, Man, and Cybernetics, 2011, pp. 838–843, doi: 10.1109/icsmc.2011.6083757.

[4] Gal, V.—Le Prado, C.—Natkin, S.—Vega, L.: Writing for Video Games. Proceedings Laval Virtual (IVRC), 2002.

[5] Jensen, K.: An Introduction to the Practical Use of Coloured Petri Nets. In: Reisig, W., Rozenberg, G. (Eds.): Lectures on Petri Nets II: Applications (ACPN 1996). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1492, 1998, pp. 237–292, doi: 10.1007/3-540-65307-4_50.

[6] Jensen, K.—Kristensen, L.: Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer, Berlin, Heidelberg, 2009, doi: 10.1007/b95112.

[7] Konami: Silent Hill 2. Computer Game, Developed and Published by Konami, 2001.

[8] Milner, R.—Harper, R.—MacQueen, D.—Tofte, M.: The Definition of Standard ML (Revised Edition). The MIT Press, 1997, doi: 10.7551/mitpress/2319.001.0001.

[9] Murata, T.: Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, Vol. 77, 1989, No. 4, pp. 541–580, doi: 10.1109/5.24143.

[10] Natkin, S.—Vega, L.—Grünvogel, S.: A New Methodology for Spatiotemporal Game Design. In: Mehdi, Q., Gough, N. (Eds.): Proceedings of 5th Game-On International Conference on Computer Games: Artificial Intelligence, Design and Education (CGAIDE 2004), 2004, pp. 109–113.

[11] David, R.—Alla, H.: Discrete, Continuous, and Hybrid Petri Nets. Springer, 2010, doi: 10.1007/978-3-642-10669-9.

[12] Reuter, C.—Göbel, S.—Steinmetz, R.: Detecting Structural Errors in Scene-Based Multiplayer Games Using Automatically Generated Petri Nets. Proceedings of

the 10th International Conference on the Foundations of Digital Games (FDG 2015), Pacific Grove, USA, 2015.

[13] Reyno, E. M.—Cubel, J. A. C.: Automatic Prototyping in Model-Driven Game Development. Computers in Entertainment, Vol. 7, 2009, No. 2, Art. No. 29, doi: 10.1145/1541895.1541909.

[14] Rucker, R. V. B.: Software Engineering and Computer Games. Pearson Education, 2003.

[15] van der Aalst, W.: Timed Coloured Petri Nets and Their Application to Logistics. International Symposium on Physical Design, 1992.

[16] van Der Aalst, W.—van Hee, K. M.: Workflow Management: Models, Methods, and Systems. MIT Press, 2004.

[17] van der Aalst, W. M.: Structural Characterizations of Sound Workflow Nets. Computing Science Reports, Vol. 96, 1996, No. 23, pp. 18–22.

[18] van der Aalst, W. M.: Verification of Workflow Nets. In: Azéma, P., Balbo, G. (Eds.): Application and Theory of Petri Nets 1997 (ICATPN 1997). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 1248, 1997, pp. 407–426, doi: 10.1007/3-540-63139-9_48.

[19] van der Aalst, W. M.: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers, Vol. 8, 1998, No. 1, pp. 21–66, doi: 10.1142/s0218126698000043.

[20] van der Aalst, W. M.—Stahl, C.: Modeling Business Processes: A Petri Net-Oriented Approach. The MIT Press, 2011, doi: 10.7551/mitpress/8811.001.0001.

[21] van der Aalst, W. M.—ter Hofstede, A. H.: Verification of Workflow Task Structures: A Petri-Net-Based Approach. Information Systems, Vol. 25, 2000, No. 1, pp. 43–69, doi: 10.1016/s0306-4379(00)00008-9.

Franciny M. Barreto works at the Federal University of Jataí as Assistant Professor. She received her Bachelor degree in computer science in 2012 from the Federal University of Goiás (Brazil), her Master degree in computer science in 2015, and her Ph.D. in 2020, both of them from the Federal University of Uberlândia (Brazil).

Stéphane Julia received his Diploma in electrical and automatic control engineering in 1992, his Master degree in automatic and industrial computing in 1993 and his Ph.D. in industrial computing in 1997, all of them from the Paul Sabatier University of Toulouse (France). He is currently Full Professor of computer science at the Federal University of Uberlândia (Brazil). His research interests include the use of Petri nets in software engineering and the modeling and analysis of workflow management systems.