Penerbit Akademia Baru

Journal of Advanced Research Design

Journal homepage:
https://akademiabaru.com/submit/index.php/ard
ISSN: 2289-7984

Regularized Stacked Autoencoder with Dropout-Layer to Overcome Overfitting in Numerical High-Dimensional Sparse Data

Abdussamad[1,*], Hanita Daud[1], Rajalingam Sokkalingam[1], Iliyas Karim Khan[1], Abdus Samad Azad[1], Muhammad Zubair[2], Farrukh Hassan[3]

[1] Fundamental and Applied Science Department Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak, Malaysia
[2] Department of Computer Sciences, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak, Malaysia
[3] Department of Computing and Information System, School of Engineering and Technology, Sunway University, 47500 Petaling Jaya, Selangor, Malaysia

ARTICLE INFO

Article history:
Received 2 August 2024
Received in revised form 14 August 2024
Accepted 4 April 2025
Available online 30 April 2025

Keywords:
Sparse data; deep learning; regularized stacked autoencoder

ABSTRACT

High-dimensional sparse numerical data are normally encountered in machine learning, recommender systems, finance and medical imaging. The problem with this type of data is that it has high dimensions (many features) and highly sparse (most values are zero), which is prone to overfitting. The data visualization can be achieved through a neural network architecture called stacked autoencoders. These multilayer autoencoders are designed to reconstruct input data, but overfitting is a major problem. To overcome this problem novel L1 Regularization-dropout technique is introduced to reduce overfitting and boost stacked autoencoder performance. L1 regularization penalizes large weights, simplifying data representations whereas the dropout technique randomly turns off neurons during training and makes the model dependent only on the selected turn-on neurons. The model employs batch normalization to improve the performance of the autoencoder. The approach was implemented on a high-dimensional sparse numerical dataset in the field of cybersecurity to minimize the loss function, measured by Mean Square Error (MSE) and Mean Absolute Error (MAE). The findings were compared to the conventional stacked autoencoder. The study revealed that the suggested method effectively mitigated the issue of overfitting. Stacked autoencoders, when combined with L1 regularisation and the dropout approach, are very successful in handling high-dimensional sparse numerical data in a diverse range of applications.

1. Introduction

Recently, the analysis of high-dimensional data has become essential because of its use in several domains such as bioinformatics, image processing, natural language processing and cybersecurity. A dataset is classified as high-dimensional when the number of features (p) surpasses the number of observations (N). For instance, in the case of a dataset with 6 features (p = 6) and only 3 observations (N = 3), it can be classified as high-dimensional due to the presence of more features than data points

* *Corresponding author*
E-mail address: abdussamad_22009779@utp.edu.my

https://doi.org/10.37934/ard.129.1.6074

[9]. High-dimensional data may be categorized into two different types: sparse data and dense data. A data set is dense if most of the values are non-zero, otherwise, it's sparse [1,10]. This distinction is crucial as it impacts how data is processed and analysed. Sparse data is prevalent in many internet-scale applications, including search engines, recommendation systems and online advertising. However, most deep learning frameworks are designed for dense data and struggle to perform effectively on sparse datasets [12]. The exponential increase in the quantity, variety, complexity and dimensions of digital data presents unique challenges, particularly in the domain of high dimensional sparse data [4]. Various applications, including biology, computer vision and text processing, frequently utilize sparse, high-dimensional; vectors for data representation [14].

Several methodologies have been proposed to address these challenges, for instance, Liu *et al.,* [14] proposed a new approach for learning similarity measures in high-dimensional sparse data that aims to circumvent the limitations of traditional methods. While these approaches offer innovative solutions, they also have limitations. One major drawback is computational inefficiency when handling large, complex datasets, along with limited theoretical guarantees. Another significant development is the industrial deep learning framework (XDL), a distributed, scalable and high-performance system designed specifically for high-dimensional data. Despite this, XDL lacks open-source availability, hence limiting its accessibility for academic study and affecting contributions from the wider research community [10]. The Fast Autoencoder (FAE) model examines High-dimensional Structural (HiDS) data and offers reduced computing expenses. Nevertheless, there is a lack of empirical evidence to support its effectiveness across various datasets, which gives rise to inquiries over its applicability and constraints in dealing with limited data [11]. The SL-LF model, which focusses on the smooth L1-norm, is specifically developed to predict missing data in matrices that are both high-dimensional and sparse. However, it has challenges in adjusting its hyperparameters automatically and achieving optimal performance while sticking to nonnegative restrictions [23]. On the other hand, the multi-metric latent factor (MMLF) technique enhances performance by revealing hidden patterns in detailed data, but it also brings further complexity owing to its elaborate design [24]. Deep learning has been more popular in the field of high-dimensional data analysis due to its capacity to identify low-dimensional subspaces. Deep feedforward networks and convolutional neural networks have been extensively used in image processing, natural language interpretation and robotic control, yielding remarkable achievements [7,8]. A multivariate function can be modelled in such deep feedforward networks by a hierarchy of features, each represented as the result of applying to an input series of desirable nonlinear projections devised so that high dimensionality doesn't create problems. However, a deep network is usually trained using large-scale data, which may be too expensive to put into practical use in engineering problems [5,17]. Compared to traditional machine learning (ML) techniques, DL is a unique research direction in the field of ML that has shown remarkable success in many applications. Feature engineering in standard machine learning has become a significant bottleneck, limiting the amount of human labour that can be effectively applied [13]. In contrast, deep learning (DL) algorithms excel at handling complex relationships because they can hierarchically extract information from raw data through multiple levels of nonlinear processing [26]. The development of graphics processing units (GPUs) and improvements in computing capacity have made it easier to train deep learning algorithms. Methods such as the Stacked Autoencoder (SAE) are very successful in learning important data aspects, which makes them beneficial for applications like categorization. SAEs are prone to overfitting, particularly when trained on limited datasets, because of their complex structures including several parameters [12].

This study presents a Regularised Stacked Autoencoder (RSAE) model specifically developed to tackle the problem of overfitting that often arises in high-dimensional sparse data. The method

employed the use of L1 regularization together with dropout layers to enforce sparsity and decrease the complexity of the model, hence enhancing its generalization abilities. The RSAE model demonstrates exceptional performance on a cybersecurity dataset, indicating its potential for use in other sectors with high-dimensional and sparse data, such as image processing, natural language processing and biology.

The contributions of this paper are as follows:

i. The RSAE model, which combines L1 regularization with dropout layers, substantially improves the performance of the Stacked Autoencoder (SAE) in dealing with high-dimensional sparse data.
ii. The RSAE model outperforms existing techniques, including the classic SAE, by effectively reducing overfitting and achieving lower error metrics.
iii. The rest of this paper is structured as follows: Section 2 explores the detailed workings of the Stacked Autoencoder (SAE) enhanced with L1 regularization.

Table 1
Comparative summary for state-of-the-art approaches

Literature	Method	Limitation	Conclusion
Kuan *et al.,* [14]	Frank-Wolfe	Scalability and generalization are limited because of high computing costs, reliance on labelled data and probable overfitting.	This approach increases similarity learning in sparse data, resulting in better performance but requiring additional scalability enhancements.
Jiang *et al.,* [10]	XDL Framework	Large-scale dataset optimization is complex and requires a lot of processing power.	Demonstrates good handling of high-dimensional sparse data, with potential for industrial-scale use.
Jiang *et al.,* [11]	Fast Deep AutoEncoder	Computationally intensive, reconstruction accuracy and efficiency must be carefully tuned.	Effectively handles high-dimensional sparse matrices in recommender systems, improving speed and scalability.
Wu *et al.,* [23]	Robust Latent Factor Analysis	Hyperparameter selection can be critical and optimal performance may need significant adjustment.	Accurately and robustly represents high-dimensional sparse data, boosting data analysis and modelling precision.
Wu *et al.,* [24]	Multi-Metric Latent Factor Model	The integration of many measurements is complex and parameter adjustment may be tough.	Improves analysis of high-dimensional sparse data by using numerous metrics to increase accuracy and understanding.
Zhang *et al.,* [27]	Stacked Sparse Autoencoder (SSAE) and Improved Gaussian Mixture Model (GMM)	The model is computationally demanding and necessitates significant parameter adjustment, which might affect scalability and performance in big or noisy datasets.	The model successfully enhances intrusion detection accuracy in high-dimensional data by utilizing the Stacked Sparse Autoencoder and Improved Gaussian Mixture Model, however, it may be restricted by computational complexity and tuning issues.

2. Methodology

2.1 Data Prepossessing

Raw datasets have numerous problems, such as outliers, missing values, various feature dimensions and in-comparability [2]. Data can only be used as input into the model once it has been cleaned up and prepossessed [15]. Furthermore, since the SSAE network's input is a numeric matrix, we must translate the symbolic characteristics into numerical features. Additionally, a maximum-

minimum normalization technique is used for the original feature values to put them in the same order of magnitude and make comparisons easier [22].

2.2 Dataset

UNSW-NB15 is the dataset chosen for RSAE evaluation. It was created in 2015 by the Australian Centre for Cyber Security (ACCS) laboratory utilizing the IXIA Perfect Storm tool [19]. Table 2 contains a breakdown of the specific features that make up a total of 49 features in the dataset. Total traffic samples in all are 2,540,044 across 4 CSV files. We create a training set and a testing set from the 2,540,044 original traffic samples consequently. The dataset was uploaded to Google Drive for effective data management and accessibility. Using Google Colab, the experimental setup made use of the processing capacity of the Google Cloud-activated free GPU environment. This improved efficiency by facilitating easy access to the dataset and speeding up processing power.

Table 2
The UNSW-NB15 dataset features

Feature category	Feature name
low features	scrip,sport,dstip,dsport,proto
base features	state,dur,sbytes,dbytes,sttl,dttl,sloss,dloss,service,sload,dload,spkts,dpkts
content features	swin, dwin,stcpb,dtcpb,smeansz, dmeansz, trans_depth,res_bdy_len
time features	sjit,djit,stime,ltime,sintpkt,dintpkt,tcprtt,synack,ackdat
additional generated features (general purpose features)	is_sm_ips_ports,ct_state_ttl,ct_flw_http_mthd,is_ftp_login,ct_ftp_cmd
additional generated features (connection features)	ct_srv_src,ct_srv_dst,ct_dst_ltm,ct_src_ltm,ct_src_dport_ltm,ct_dst_sport_ltm,ct_dst_src_ltm
labelled features	attack_cat,Labe

2.3 Numeralization

We use one-hot encoding to perform Numeralization. The symbolic features the dimensional dataset contains include “proto”, “service”, “state” and “attack _ cat” Consequently, the dataset's feature dimensions are expanded upon the conclusion of the numerical processing [25].

2.4 Normalization

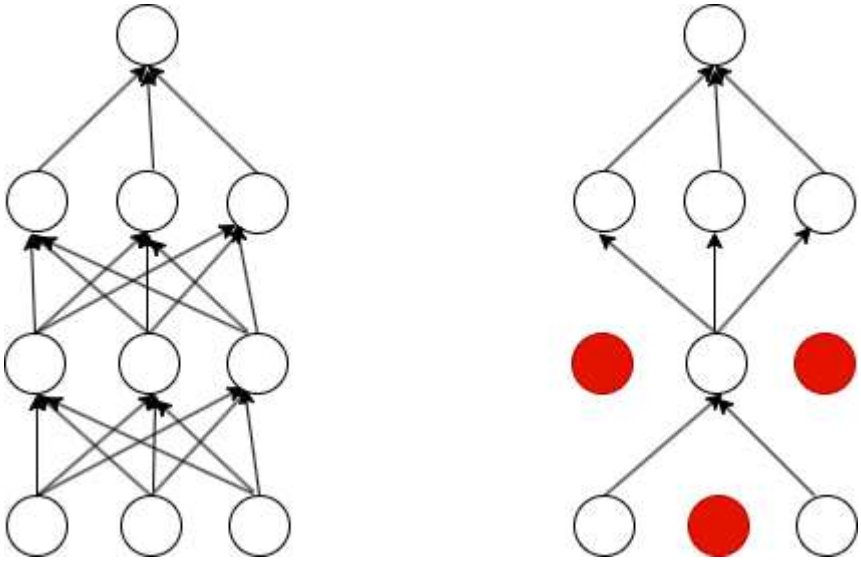
The maximal-minimum normalization approach provided in Eq. (1) is used to normalize the feature values in the dataset to make it easier to compare the findings [27]. The value of x is scaled into the numeric range [0,1] using the min-max normalization method,

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)} \quad (1)$$

Where X' = normalized value,
X = Original value
$\min(X)$ = minimum value of X
$\max(X)$ =maximum value of X

2.5 Dropout

A neural network model can use a dropout strategy to learn more robust features and reduce the amount of interdependent learning among the neurons. In a neural network, dropping-out units are referred to as "dropouts." A unit can be temporarily removed from the network by dropping it, together with all its incoming and outgoing connections [21]. Random units can be dropped from the network. The dropout technique is used in this work to educate unsupervised learning to prevent over-fitting or the extraction of the same features again. With the use of dropout, specific nodes are set to zero values in a training run and dropped from the network. Therefore, they do not affect its prediction and in the backpropagation. Consequently, a new slightly altered network structure is created in each run and the network learns to provide quality predictions without specific inputs. On setting up the dropout layer a so-called drop probability must be specified. This defines the number of nodes that will be assigned 0 in a layer. It should be mentioned that the dropout feature is only used during the training phase and is disabled during testing [3]. Figure 1 shows the difference between standard neural networks and those after applying dropout.



(a) Standard neural network (b) After applying dropout

Fig. 1. Dropout applied to a standard neural network

2.6 Autoencoder Model

The input layer, hidden layer and output (reconstruction) of the unsupervised three layer network known as Autoencoder are depicted in Figure 2 and Figure 3 with the representation of network. [18]. Nonlinear transformation from a high dimensional space into a low-dimensional one can accomplished by the autoencoder by sequentially mapping the synthetic feature vectors to abstract feature vectors [6]. The autoencoder can be divided into two stages: encoding and decoding which can be defined as:

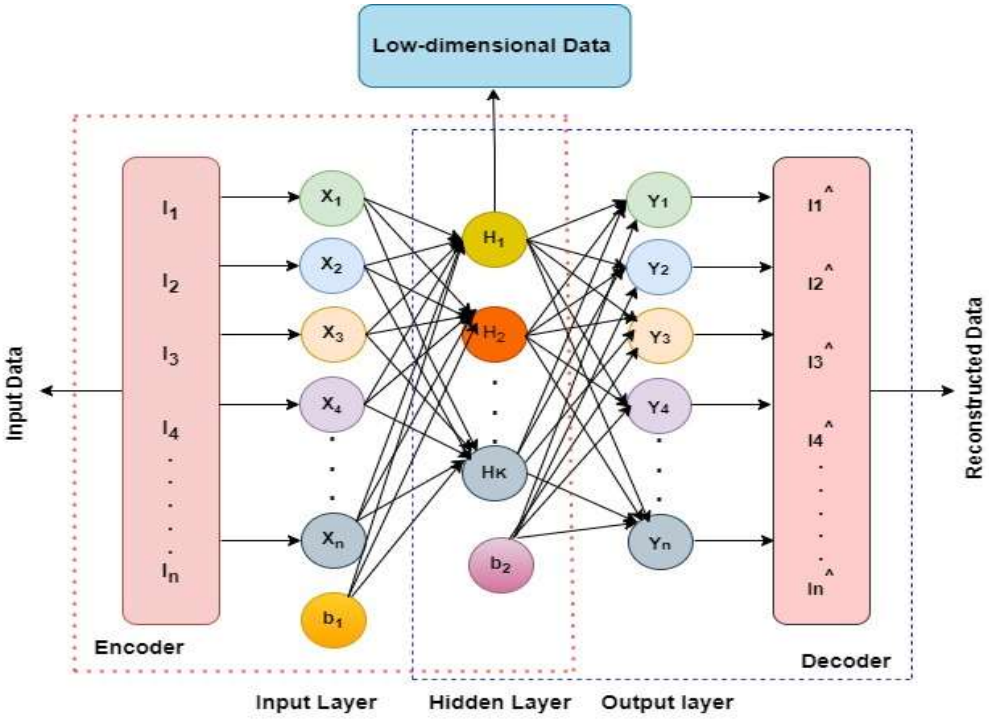


Fig. 2. Basic autoencoder model

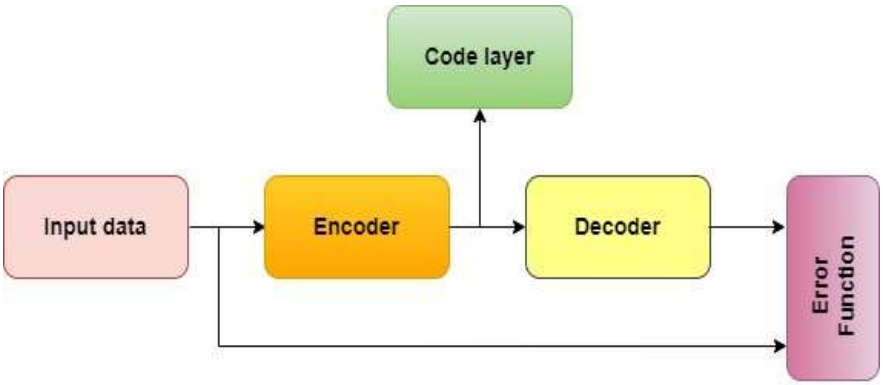


Fig. 3. Autoencoder model representation

The encoding process from the input layer to the hidden layer is as in Eq. (2),

$$H = f\theta_1(X) = \sigma(W_{ij}X + b_1) \quad (2)$$

The procedure for the decoding from the reconstruction layer to concealed layer is as in Eq. (3),

$$Y = f\theta_2(H) = \sigma(W_{jk}X + b_2) \quad (3)$$

The input data vector in this formula denoted by $X = (x_1, x_2, x_3, \dots \dots x_n)$, the reconstruction vector of the input data is represented by $Y = (y_1, y_2, y_3, \dots \dots y_n)$ and the low dimensional output from the hidden layer is denote by $H = (h_1, h_2, h_3, \dots \dots h_m)$. Thus, $X \in R^n$, $Y \in R^n$, $H \in R^m$ (where n is the input vector's dimension and m are the number of hidden units). The weight connection matrix between the input layer and hidden layer is denoted by $W_{ij} \in R^{m \times n}$. The weight connection matrix between the output layer and hidden layer is denoted by $W_{jk} \in R^{n \times m}$. $W_{ij} = {W_{jk}}^T$ often

occurs in the experiment to reconstruct the input data as precisely as feasible while minimizing the resource consumption during model training. $b_1 \in R^{n\times 1}$ and $b_1 \in R^{m\times 1}$ are the bias vectors of input layer and hidden layer respectively. $f\theta_1(\cdot)$ and $f\theta_2(\cdot)$ are the activation functions of hidden layer neuron and output layer neurons respectively. We use Relu activation function and sigmoid activation function in this paper as in Eq. (4) and (5) respectively,

$$f\theta_1(\cdot) = max(0, x) \tag{4}$$

$$f\theta_2(\cdot) = \frac{1}{1+e^{-x}} \tag{5}$$

The Autoencoder makes the reconstruction of original data through training by minimizing the resulting error between reconstructed output and actual values. At this stage we assume that the data provided by hidden layer units aggregates all information which was present in initial dataset and is optimal low-dimensional representation of it. Eq. (6) illustrates the application of the mean squared-error function in the reconstruction error function $J_E(W, b)$ between H and Y, where N is the number of input samples.

$$J_E(W, b) = \frac{1}{2N}\sum_{r=1}^{N}\|Y^r - X^r\|^2 \tag{6}$$

2.7 Stacked Autoencoder (SAE)

The concept of sparse coding to model the computational learning of basic cell receptive fields in the primary visual cortex of mammals was first introduced by Olshausen *et al.,* [20]. For instance, the input data is transferred to the output layer by straightforward copying because of the autoencoder's inevitable issue. In this instance, the autoencoder does not extract any useful features, even though the original input data can be reconstructed properly. To make the autoencoder generate more concise and efficient low-dimensional data features under sparse constraints to better depict the input data, the author used a method of adding L1 penalty terms on hidden layers in an effort. The term "L1-norm," also known as "Lasso regression," refers to the weight vector $W's$ sum of the absolute values of each of its elements. It is defined as follows: $L1(W) = \|W\| = \sum_i \| W \|_i$,. It can therefore be applied to select more significant representations. Choosing features that provide greater value to the model during training is hampered by an abundance of characteristics in the sample. As a result, we eliminate the connections that add very little to the model and do not affect the classification performance at all. With high dimensional data, it can extract more valuable features in less time.

The mean square error term and the regularization term make up the first and second terms of the error function at this point. As may be seen in Eq. (7):

$$J_E(W, b) = \frac{1}{2N}\sum_{r=1}^{N}\|Y^r - X^r\|^2 + \alpha \sum \| W^r{}_{ij} \| \tag{7}$$

Here, α is a user-adjustable hyperparameter that allows us to precisely manage L1 regularization. This new regularization method is incorporated into our autoencoder architecture to enhance feature learning and minimize overfitting. Layers of encoding and decoding inside the design itself aid in constructing hierarchical representations from incoming input. We use dropout layers after each encoding layer, where neurons are regularly removed from the training population to avoid

overfitting. A RSAE neural network's structure, which is made up of several regularized auto-encoders connected end to end, is depicted in Figure 4.

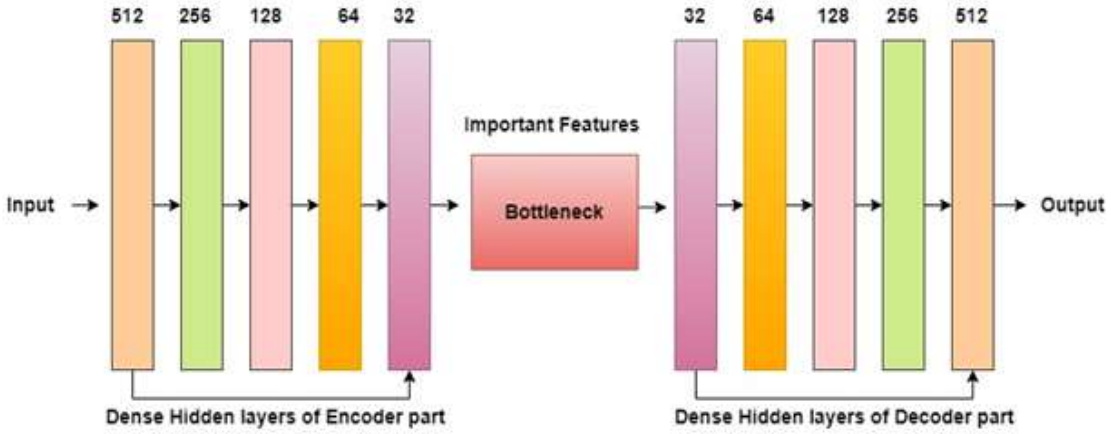


Fig. 4. Regularized Stacked Autoencoder model

Higher-level feature representations of the input data are produced by the subsequent layer of the sparse self-encoder using the output from the preceding layer. The optimal connection weights and bias values of the stacked sparse auto-encoded network are obtained through sequential training of each layer using a greedy layer pretrain method. For the best parameter model, RSAE is then tuned using error back-propagation way until a satisfactory result of the output value between the input and required data. For the error function given in Eq. (6):

$$\frac{\partial}{\partial \mathrm{W}^{r}{}_{ij}} J_E(W,b) = \frac{1}{2N}\frac{\partial}{\partial \mathrm{W}^{r}{}_{ij}} \sum_{r=1}^{N} \|Y^r - X^r\|^2 + \alpha \cdot \mathrm{sign}(\mathrm{W}^{r}{}_{ij}) \tag{8}$$

$$\frac{\partial}{\partial \mathrm{b}^{r}} J_E(W,b) = \frac{1}{2N}\frac{\partial}{\partial \mathrm{b}^{r}} \sum_{r=1}^{N} \|Y^r - X^r\|^2 \tag{9}$$

Consequently, the following Eq. (10) and (11) is the weight and bias update processes,

$$\mathrm{W}^{k}{}_{ij} = \mathrm{W}^{k}{}_{ij} - \mu \frac{\partial}{\partial \mathrm{W}^{k}{}_{ij}} J_E(W,b) \tag{10}$$

$$\mathrm{b}^{r} = \mathrm{b}^{r} - \mu \frac{\partial}{\partial \mathrm{b}^{r}} J_E(W,b) \tag{11}$$

Where, Y^r and X^r are respectively the original vector and its corresponding reconstruction vectors. μ represents the learning rate.

Due to the sparse structure of the RSAE network, distinct learning rates to the individual parameters. For features that aren't used regularly, such as the goal of releasing fewer updates. However, most widely used gradient descent algorithms, including mini-batch and stochastic gradient descent, employ the same learning rate for every parameter that needs to be updated, making it challenging to choose the right learning rate and rapidly arrive at a local minimum [16]. Therefore, we employ the adaptive moment estimation (Adam) gradient descent approach described by Zhang [28] to perform dynamic adaptive adjustment of various parameters to train a better RSAE network model. By calculating the gradient first-order moment estimate m_t and second-order moment estimate v_t as shown in Eqs. (12) to (14), the Adam algorithm allows for the dynamic

adjustment of various parameters. $\beta_1$and β_2 stand for the first order and second-order exponential damping decrements, respectively. The gradient of the parameters at the time step t in the loss function $J_E(W, b)$ is denoted by g_t.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \cdot g_t \quad (12)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \cdot {g_t}^2 \quad (13)$$

$$g_t \leftarrow \nabla_\vartheta J_t(\vartheta_j - 1) \quad (14)$$

Computer bias-corrected for m_t andv_t as in Eqs. (15) and (16) respectively,

$${m_t}' = \frac{m_t}{1-{\beta_1}^t} \quad (15)$$

$${v_t}' = \frac{v_t}{1-{\beta_2}^t} \quad (16)$$

The update step size is denoted by τ and ϵ is constant to prevent the denominator from zero as in Eq. (17),

$$\vartheta_{t-1} = \vartheta_t - \frac{\tau}{\sqrt{{v_t}'+\epsilon}} \cdot {m_t}' \quad (17)$$

3. Results

3.1 Model Parameters and Sensitivity Analysis

In this work, a RSAE architecture is used to obtain significant features and reconstitute input information. RSAE includes an encoder and a decoder, both containing five interrelated layers. The encoding layers sequentially reduce the input data size, with dense units featuring rectified linear activation in addition to batch normalization and dropout mechanisms for countering overfitting. According to section 2.1 after the sample in the UNSW-NB15 dataset is pre-processed the features are extended from 49 dimensions to 202 dimensions. Consequently, the author decides that the RSAE has 202 input layer neurons. Extensive testing and a literature study led to the selection of hyperparameters, such as the learning rate, number of neurons in hidden layers, batch size and L1 regularization strength (alpha). We used grid search techniques to find the best values for these parameters, guaranteeing a reasonable trade-off between model complexity and performance. Furthermore, investigations demonstrate that the five-layer RSAE network's hidden structure is the best experimental model which is shown in Table 3. The dense layer with 32 units and ReLU activation is the critical space where important features are captured in the model. This layer is also regularized via L1 regularization with various values of alpha, demonstrating to what extent the Regularized Stacked Autoencoder. Mean Squared Error (MSE) and Mean Absolute Error (MAE) are used as the reconstruction losses with the Adam optimizer, learning rate 0.0001 and 100 epochs. Sigmoid activation is applied to the final layer output values, which are constrained between 0 and 1. 128 samples are organized into batches to avoid overfitting and the regularization term improves model robustness. Having used key metrics, such as MSE and MAE, throughout both training and validation phases, model performance is measured. The experimental model parameters are presented in Table 3.

Table 3
Hyperparameter summary of RSAE

Algorithms	Parameter	Value
RSAE	The number of nodes in the input layer	202
	Number of neurons in the initial hidden layer	512
	Number of neurons in the second hidden layer	256
	Number of neurons in the third hidden layer	128
	Number of neurons in the fourth hidden layer	64
	Number of neurons in the fifth hidden layer	32
	Learning rate	0.0001
	Alpha	0.0001,0.001,0.01,0.1,1
	Batch size	128
	Epochs	100
	Activation functions	ReLU, Sigmoid
Adam	First-order exponential damping decrement	0.9
	Second-order exponential damping decrement	0.999
	Non-zero constant	10^{-8}

An extensive sensitivity analysis was conducted to evaluate the impact of varying the L1 regularization intensity (alpha) on the performance of the model. This research examined the impact of different alpha values on the MSE and MAE of the training and validation datasets. The results show that an alpha value of 0.0001 yields the best performance and the lowest errors in both measurements. Conversely, when alpha values were higher, there was a notable rise in error rates, indicating that excessive regularisation negatively impacts the performance of the model. Higher alpha values have a detrimental effect on the model because they excessively restrict it, resulting in less accuracy and stability. Conversely, lower alpha values help reduce errors and improve the model's ability to generalize.

A comprehensive sensitivity analysis was carried out to assess how changing the L1 regularization intensity (alpha) might affect the model's performance. This study examined the effects of various alpha values on the training and validation datasets MSE and MAE. The findings show that the highest performance is achieved with alpha values of 0.0001 and 0.001, with the lowest errors in both measures. Higher alpha values, on the other hand, significantly increased error rates, suggesting that over-regularization harms model performance. This implies that greater alpha values tend to unduly limit the model, which negatively affects its accuracy and resilience, while lower values aid in decreasing error and enhancing model generalization.

3.2 Quantitative Results

The RSAE model's ability to accurately identify structural similarities is shown in Figure 5 and Figure 6, which display the training and validation Loss. These figures demonstrate the RSAE's effective generalization and its ability to avoid overfitting. The MSE and MAE metrics are used to assess the reconstruction quality of the autoencoder under various alpha configurations. These metrics serve to illustrate the training and validation performance of the model as it acquires knowledge. As the number of epochs rises, the MSE and MAE for the training data drop, suggesting that the model is successfully acquiring knowledge from the data. Similarly, the MSE and MAE for the validation data fall as the number of epochs increases, indicating that the model effectively applies its knowledge to new input. In general, lower values of MSE and MAE imply a more optimal fit.

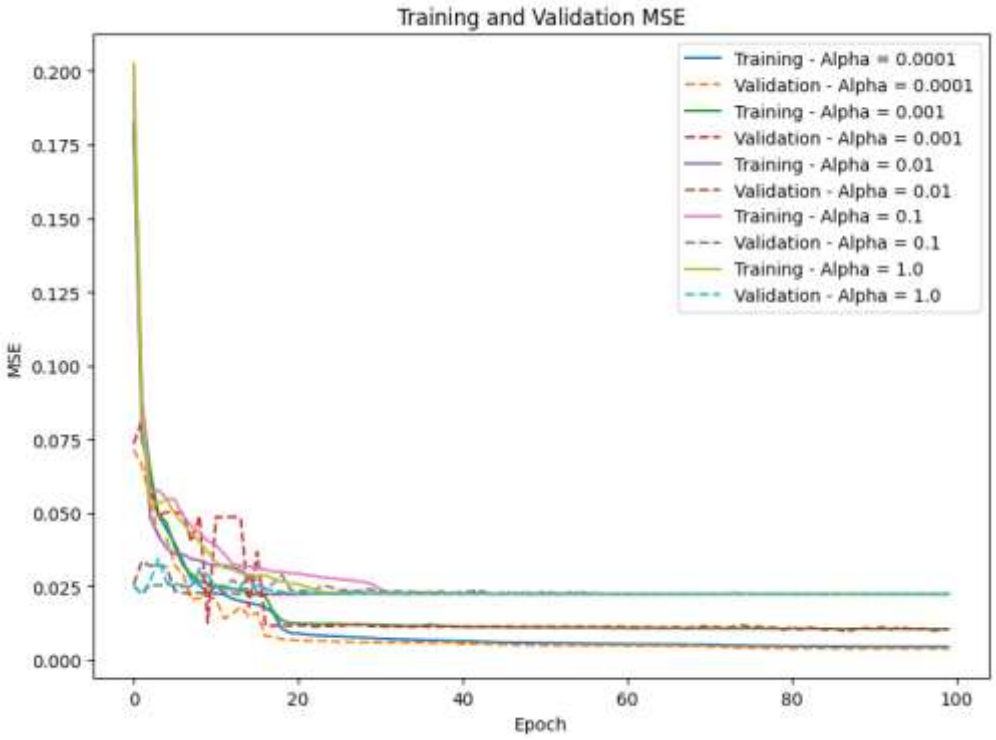


Fig. 5. Training and validation MSE loss with L1 regularization

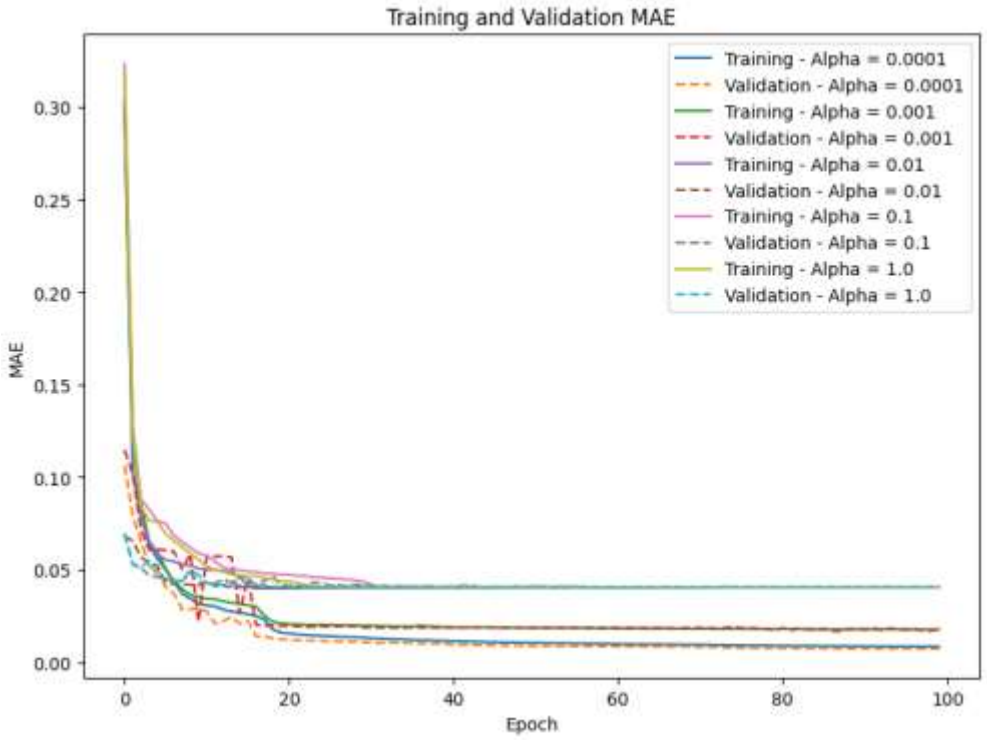


Fig. 6. Training and validation MAE loss with L1 regularization

Table 4 presents a concise overview of how different alpha values affect the performance of the RSAE model. Specifically, it examines the MSE and MAE for both the training and validation datasets. Alpha is a crucial hyperparameter that determines the level of L1 regularization applied to the model. Increasing alpha values enhance regularization, hence mitigating overfitting but probably restricting the model.

The sensitivity analysis assessed how different alpha values influence model performance. The results indicate that alpha values of 0.0001 provide the best performance, with the lowest training and validation errors. Conversely, higher alpha values (0.01, 0.1, 1.0) significantly increased both MSE and MAE, indicating that over-regularization negatively affects the model's accuracy and robustness. Thus, lower alpha values are more effective in reducing errors and improving the model's generalization ability.

Table 4
Sensitivity analysis of RSAE performance metrics with Varying α

Alpha (L1 strength)	Training MSE	Validation MSE	Training MAE	Validation MAE
0.0001	0.0043	0.0038	0.0083	0.0072
0.001	0.0108	0.0099	0.0181	0.0167
0.01	0.0225	0.0225	0.0405	0.0405
0.1	0.0225	0.0225	0.0405	0.0405
1.0	0.0225	0.0225	0.0405	0.0405

To validate the RSAE model with classical stacked autoencoder it is shown in Figure 7 that the model lacks regularization The model overfits the training data when MSE and MAE from training is much lower than that of validation. This means that the model is learning specific features of what it was trained on too well and therefore doesn't generalize to unseen data as a result.

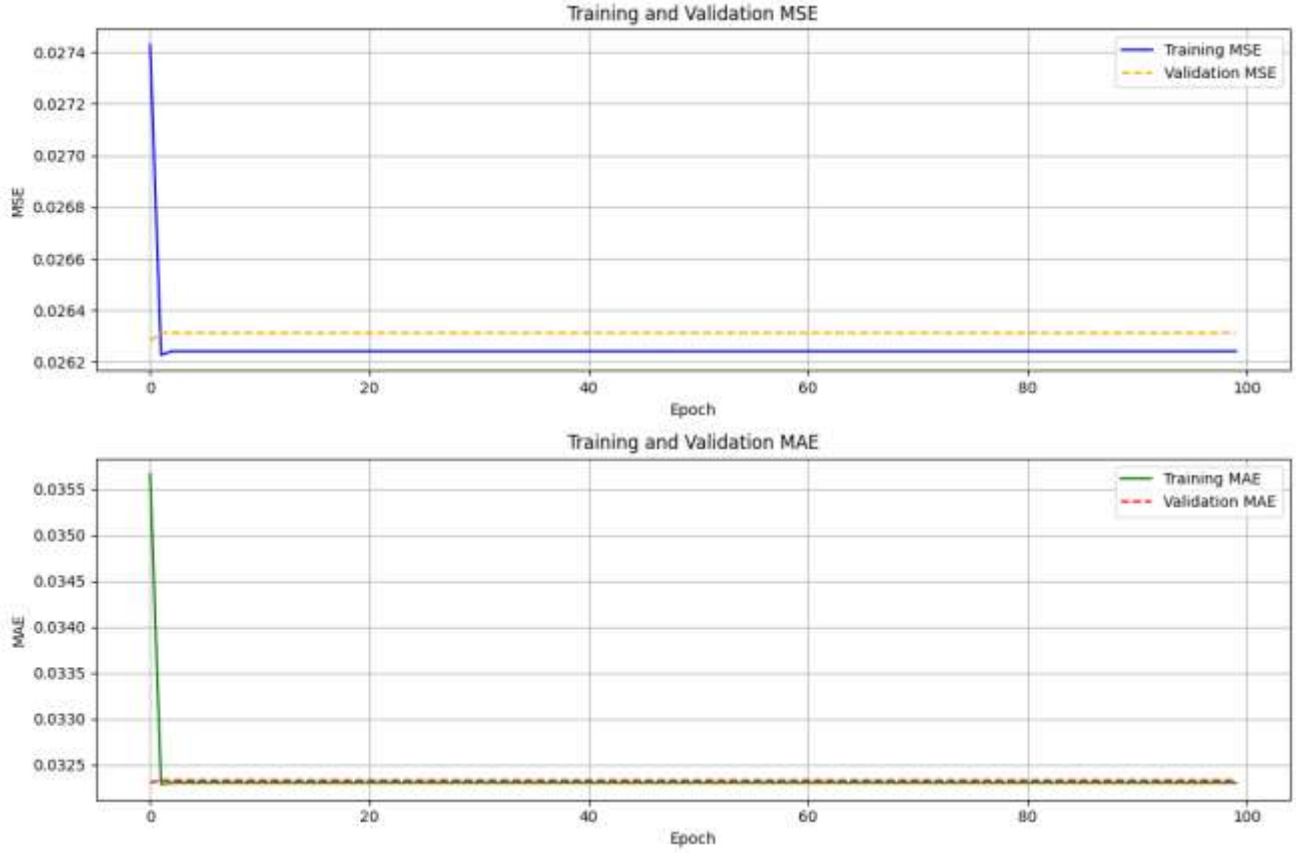


Fig. 7. Training and validation loss without L1 regularization

Table 5 summarizes the outcomes for MSE and MAE of classic stacked autoencoder respectively.

Table 5
SAE without regularization

S/No	MSE	MAE
1	0.0262	0.0329

A comparison of the training and validation loss curves shows that L1 regularization successfully prevents overfitting and enables stack autoencoders to be used on high-dimensional sparse data with considerably greater generalizability. L1 therefore provides a paradigm for an example-oriented model that also can understand patterns, making it extendable to actual applications. So, it promotes sparsity, feature selection and generalizability.

4. Discussion

The outcomes of the proposed RSAE model demonstrate its ability to mitigate overfitting and enhance performance on the cybersecurity dataset. These developments have significant practical implications in several domains, in addition to being technological.

4.1 Cybersecurity Context

Considering cybersecurity, the better performance of the RSAE model results in more consistent threat detection. Less overfitting helps the model distinguish between actual actions and potential hazards, hence lowering false positives and negatives. More accurate identification of anomalies and hazards follows from this, which is vital for quick reaction and avoidance of security breaches. Moreover, more effective use of computer resources made possible by higher model efficiency helps to potentially lower running expenses and shorten the time required for threat detection and response.

4.2 Financial Sector

The RSAE model may significantly enhance fraud detection in financial organizations, where security and accuracy are of utmost importance. By enhancing the model's ability to identify unusual patterns in transaction data, banks and financial institutions may enhance their ability to safeguard against fraudulent activities and insider threats. This, in turn, will bolster financial transaction security and preserve sensitive information.

4.3 Health Sector

The RSAE model's advancements enhance patient data privacy in the healthcare industry. Enhanced anomaly detection capabilities ensure the secure storage and adherence to regulations, such as HIPAA, of sensitive health information. Not only does this safeguard patient confidentiality, but it also fosters confidence in digital healthcare solutions.

4.4 Manufacturing

In an industry where operational technology and critical infrastructure are increasingly being attacked by hackers, the RSAE model's increased performance may help prevent costly disruptions.

By shielding industrial control systems from potential cyber threats, the method promotes operational continuity and safety while minimizing the significant financial and safety risks associated with cyberattacks.

5. Conclusions

RSAE models can extract complex characteristics from high-dimensional, sparse data. RSAE models learn well from training data and can efficiently generalize to new samples. One method for preventing overfitting is L1 regularisation, which penalizes the absolute value of the weights. This regularization strategy reduces certain weights to zero, allowing the model to learn sparse representations. This may help the model avoid learning unnecessarily intricate properties, resulting in a more intelligible structure. Studies have proven that this simplified regularization technique provides outstanding performance on high-dimensional sparse data using RSAE models with L1. Future research focuses on improving the RSAE model's performance and adapting it to new and changing cyber threats. Exploring its use for prediction tasks like binary classification, as well as increasing its usage in diverse situations, would help boost its usability and efficacy in cybersecurity and other fields.

Acknowledgement

This project was funded by a YUTP-FRG Grant under the cost centre: 015LC0-442, Universiti Teknologi PETRONAS, Malaysia.

References

[1] Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado *et al.,* "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* (2016).

[2] Abhaya, Abhaya and Bidyut Kr Patra. "An efficient method for autoencoder based outlier detection." *Expert Systems with Applications* 213 (2023): 118904. https://doi.org/10.1016/j.eswa.2022.118904

[3] Aouedi, Ons, Kandaraj Piamrat and Dhruvjyoti Bagadthey. "A semi-supervised stacked autoencoder approach for network traffic classification." In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, pp. 1-6. IEEE, 2020. https://doi.org/10.1109/ICNP49622.2020.9259390

[4] Ayesha, Shaeela, Muhammad Kashif Hanif and Ramzan Talib. "Overview and comparative study of dimensionality reduction techniques for high dimensional data." *Information Fusion* 59 (2020): 44-58. https://doi.org/10.1016/j.inffus.2020.01.005

[5] Baldi, Pierre, Peter Sadowski and Daniel Whiteson. "Searching for exotic particles in high-energy physics with deep learning." *Nature communications* 5, no. 1 (2014): 4308. https://doi.org/10.1038/ncomms5308

[6] Daneshfar, Fatemeh, Sayvan Soleymanbaigi, Ali Nafisi and Pedram Yamini. "Elastic deep autoencoder for text embedding clustering by an improved graph regularization." *Expert Systems with Applications* 238 (2024): 121780. https://doi.org/10.1016/j.eswa.2023.121780

[7] Han, Jiequn and Arnulf Jentzen. "Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations." *Communications in mathematics and statistics* 5, no. 4 (2017): 349-380. https://doi.org/10.1007/s40304-017-0117-6

[8] Erfani, Sarah M., Sutharshan Rajasegarar, Shanika Karunasekera and Christopher Leckie. "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning." *Pattern Recognition* 58 (2016): 121-134. https://doi.org/10.1016/j.patcog.2016.03.028

[9] Ghaddar, Bissan and Joe Naoum-Sawaya. "High dimensional data classification and feature selection using support vector machines." *European Journal of Operational Research* 265, no. 3 (2018): 993-1004. https://doi.org/10.1016/j.ejor.2017.08.040

[10] Jiang, Biye, Chao Deng, Huimin Yi, Zelin Hu, Guorui Zhou, Yang Zheng, Sui Huang *et al.,* "Xdl: an industrial deep learning framework for high-dimensional sparse data." In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*, pp. 1-9. 2019. https://doi.org/10.1145/3326937.3341255

[11] Jiang, Jiajia, Weiling Li, Ani Dong, Quanhui Gou and Xin Luo. "A fast deep autoencoder for high-dimensional and sparse matrices in recommender systems." *Neurocomputing* 412 (2020): 381-391. https://doi.org/10.1016/j.neucom.2020.06.109
[12] Jin, Lina, Jiong Yu, Xiaoqian Yuan and Xusheng Du. "Fish classification using DNA barcode sequences through deep learning method." *Symmetry* 13, no. 9 (2021): 1599. https://doi.org/10.3390/sym13091599
[13] Ketkar, Nikhil. "Introduction to tensorflow." In *Deep Learning with Python: A Hands-on Introduction*, pp. 159-194. Berkeley, CA: Apress, 2017. https://doi.org/10.1007/978-1-4842-2766-4_11
[14] Liu, Kuan, Aurélien Bellet and Fei Sha. "Similarity learning for high-dimensional sparse data." In *Artificial Intelligence and Statistics*, pp. 653-662. PMLR, 2015.
[15] Lai, Xiaochen, Xia Wu, Liyong Zhang, Wei Lu and Chongquan Zhong. "Imputations of missing values using a tracking-removed autoencoder trained with incomplete data." *Neurocomputing* 366 (2019): 54-65. https://doi.org/10.1016/j.neucom.2019.07.066
[16] Kim, Jihyun and Howon Kim. "An effective intrusion detection classifier using long short-term memory with gradient descent optimization." In *2017 International Conference on Platform Technology and Service (PlatCon)*, pp. 1-6. IEEE, 2017. https://doi.org/10.1109/PlatCon.2017.7883684
[17] Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver and Daan Wierstra. "Continuous control with deep reinforcement learning." *arXiv preprint arXiv:1509.02971* (2015).
[18] Pinaya, Walter Hugo Lopez, Sandra Vieira, Rafael Garcia-Dias and Andrea Mechelli. "Autoencoders." In *Machine learning*, pp. 193-208. Academic Press, 2020. https://doi.org/10.1016/B978-0-12-815739-8.00011-0
[19] Moustafa, Nour and Jill Slay. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." In *2015 military communications and information systems conference (MilCIS)*, pp. 1-6. IEEE, 2015. https://doi.org/10.1109/MilCIS.2015.7348942
[20] Olshausen, Bruno A. and David J. Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images." *Nature* 381, no. 6583 (1996): 607-609. https://doi.org/10.1038/381607a0
[21] Spoorthy, G. and S. G. Sanjeevi. "Multi-criteria–recommendations using autoencoder and deep neural networks with weight optimization using firefly algorithm." *International Journal of Engineering* 36, no. 1 (2023): 130-138. https://doi.org/10.5829/IJE.2023.36.01A.15
[22] Vaziri, Pouya, Sanyar Ahmadi, Fatemeh Daneshfar, Behnam Sedaee, Hamzeh Alimohammadi and Mohammad Reza Rasaei. "Machine learning techniques in enhanced oil recovery screening using semisupervised label propagation." *SPE Journal* 29, no. 09 (2024): 4557-4578. https://doi.org/10.2118/221475-PA
[23] Wu, Di and Xin Luo. "Robust latent factor analysis for precise representation of high-dimensional and sparse data." *IEEE/CAA Journal of Automatica Sinica* 8, no. 4 (2020): 796-805. https://doi.org/10.1109/JAS.2020.1003533
[24] Wu, Di, Peng Zhang, Yi He and Xin Luo. "A Multi-Metric Latent Factor Model for Analyzing High-Dimensional and Sparse data." *arXiv preprint arXiv:2204.07819* (2022).
[25] Yan, Binghao and Guodong Han. "Effective feature extraction via stacked sparse autoencoder to improve intrusion detection system." *IEEE Access* 6 (2018): 41238-41248. https://doi.org/10.1109/ACCESS.2018.2858277
[26] Zhang, Guoqiang Peter. "Neural networks for classification: a survey." *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)* 30, no. 4 (2000): 451-462. https://doi.org/10.1109/5326.897072
[27] Zhang, Tianyue, Wei Chen, Yuxiao Liu and Lifa Wu. "An intrusion detection method based on stacked sparse autoencoder and improved gaussian mixture model." *Computers & Security* 128 (2023): 103144. https://doi.org/10.1016/j.cose.2023.103144
[28] Zhang, Zijun. "Improved adam optimizer for deep neural networks." In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pp. 1-2. Ieee, 2018. https://doi.org/10.1109/IWQoS.2018.8624183