Mehran University Research Journal of Engineering and Technology

https://doi.org/10.22581/muet1982.3005

2024, 43(3) 66-77

# An improved long short-term memory with denoising autoencoder for solving text classification problems

Amani Saleh Alija *, Nor Adnan Yahaya

*Faculty of Information Technology, Malaysia University of Science and Technology, Malaysia*

* Corresponding Author: Amani Saleh Alija, Email: amalija647@gmail.com

Received: 30 August 2023, Accepted: 08 May 2024, Published: 01 July 2024

K E Y W O R D S

Text Classification

Deep Learning

RNN

LSTM

DAE

A B S T R A C T

In the past decade, text classification has been a popular research field for automatically categorizing documents into relevant categories. Predicting classes for input samples based on their features is the goal of classification. To limit the dimension of the feature space for precise text categorization, feature selection approaches are widely used for recognizing important and affective features while ignoring unimportant, weak, and chaotic components. This paper proposes a new model referred to as DAE-LSTM for reducing data dimension through the use of a denoising autoencoder. (Long Short-Term Memory (LSTM) is one of the popular text classification approaches, particularly when dealing with sequential datasets. Additionally, the Rectified Linear Unit (Relu) activation function was used instead of the hyperbolic tangent activation function (tanh) to improve the accuracy of the model. Four benchmark datasets were used to evaluate the proposed model against 4 cutting-edge text classification methods, including conventional Bidirectional LSTM, Bidirectional GRU, CNN-LSTM, and CNN-GRU. The suggested model has been found to perform much better than current methods using several kinds of performance assessment measures.

## 1. Introduction

Classification is the categorization of objects or documents for decision-making purposes. It is among the most cited studies in the field of artificial neural networks (ANN) [1]. The worldwide web and the Internet contain a vast quantity of structured and unstructured digital data. In fact, in the 20th century, a huge amount of data has already been created since the early 1960s. Numerous businesses generate vast sums of digital data daily, and individuals routinely submit numerous papers online. Digital libraries, blog archives, discussion boards, news stories, and biological databases all contain this knowledge. The accurate categorization of such data is a difficult undertaking that takes a lot of work. Machine learning, on the other hand, enables the automatic examination of data by identifying patterns for categorization with a minimum of human involvement. Text classification (TC) is a method that is often used for automatically classifying an unknown text or document by recommending the most probable class or predefined categories it pertains. It is an integral component of text mining, which employs knowledge engineering techniques to develop an automatic text classification system. The fundamental objective of text mining is to extract pertinent information from textual resources and perform operations including retrieval, categorization, and summarization. Combining machine learning and data mining techniques allows for the automatic classification and discovery of patterns in various text types. As the amount of electronic information and text generated each day continues to increase, it becomes essential to organize massive quantities of data for analysis and processing [2]. Text filtering, text clustering, document organization, news item classification, web spam, website classification, data mining, and the search for

intriguing content on the web have all benefited from TC techniques. Applications for classification algorithms include sentiment analysis [3], text clustering [4], [5], spam extraction [6], website classification [7], disease report finding [8], generation of text summaries and classification them [4], [9].

Many of the most well-known text categorization techniques have been proposed and widely implemented. Naive Bayes, Support Vector Machine (SVM), k-NN, Entropy, and Decision Tree are some machine learning algorithms used for text classification [2]. However, there are benefits and drawbacks to each type of classification algorithm. Hence, the effectiveness of different categorization algorithms for different scenes is determined by their strengths and shortcomings [10]. To show superior classification performance, recurrent neural networks (RNNs) have been used in many data mining applications recently. RNNs have shown good performance in capturing the semantics for sentiment classification and are capable of capturing temporal dependencies in sequence information [11]. RNNs have the advantage of handling temporal sequential data with varied lengths, allowing for flexibility in analyzing varying-length reviews. Long-term memory (LSTM) and gated recurrent unit (GRU) are 2 variants of RNN. Text data contains numerous features and is highly dimensional. A single layer is not effective for extracting informative features of textual data. Thus, previous studies have focused on mining the most valuable features and characteristics from the raw textual data in existing deep-learning models by employing numerous layers. Although LSTM is shown to be effective in text classification tasks, and many studies have proposed approaches for improving its accuracy, it still has some flaws and shortcomings. As such, more effort is needed to make it better.

The main aim of this study is to modify the fundamental architecture of LSTM to reduce the complexity and increase its performance, rather than to improve the overall goal of RNN. Towards this end, we have accomplished the following. First, there is an improved dimensionality reduction capability of the LSTM model using a denoising autoencoder to address text categorization issues that are due to LSTM's inability to capture the feature set with usable data. Secondly, the proposed equation in this research uses Rectified Linear Unit (Relu) activation rather than the hyperbolic tangent activation function (tanh)in the LSTM architecture. For deep learning models, Relu has shown good performance as compared to other sigmoid non-linearities.

## 2. Related Work

Cai, Y et al [12] put forward 3 methods for sharing information using recurrent neural networks and use the multi-task learning framework for concurrent learning amongst inter-related tasks. A unified system is created wherein all interrelated tasks are merged and trained collectively. In the first model, a solitary shared layer is responsible for all tasks. In contrast, the second approach incorporates multiple layers that can access data from other layers to handle different tasks. A shared layer is employed in the 3rd approach for all tasks, alongside the allocation of a dedicated layer for each task. Additionally, a gating mechanism is introduced to give the model the ability to use the mutual information selectively. The whole model is trained simultaneously for all underlying tasks. During the experiments, 4 benchmark text classification datasets were utilized for the performance evaluation of the proposed models. The obtained results indicate that models can improve the accuracy by combining the tasks, as demonstrated through the experiments on these benchmark text classification tasks.

Zhen-Zhong et al [10] conducted a study focusing on the classification of news articles. Their proposed approach suggests utilizing a Latent Dirichlet Allocation-based (LDA) model for news text categorization. Due to the significantly large size of news texts, this model leverages a topic model to reduce the text size while extracting relevant features. Concurrently, the study also explored the application of the Softmax regression algorithm, a widely used classifier for addressing multi-class text-related challenges in our daily lives. Hence, the proposed new text classification system can handle a large amount of text data quickly and predict categorization labels accurately.

The task of text classification is crucial for various applications, including information retrieval, and sentiment analysis. These tasks rely on the accurate categorization of textual data to enable effective information retrieval, content filtering, and sentiment assessment [13]. Due to its significance, text classification has gained substantial attention from academia and industry. One key challenge in text categorization is feature representation, which commonly follows the bag-of-words (BoW) paradigm. This approach involves extracting features such as unigrams, bigrams, n-grams, or carefully designed patterns to capture the essence of the text. Researchers have dedicated efforts to address this problem and enhance feature representation in text categorization tasks. To handle more discriminative features, numerous feature selection approaches are

used, including frequency, Mutual Information (MI), Probabilistic Latent Semantic Analysis (pLSA) [14], and Latent Dirichlet Allocation (LDA) [15].

Lai et al [16] presented a non-human-designed recurrent convolutional neural network for text categorization. For better word representations, they employed recurrent structures to capture contextual information, which reduces noise as compared to window-based artificial neural networks. Additionally, a max-pooling layer is utilized to identify the most significant words in text categorization, thus capturing the essential components of the texts. The researchers validate their hypothesis by conducting experiments on four benchmark datasets. The experimental results demonstrate that the proposed technique surpasses existing state-of-the-art methods, particularly on document-level datasets.

Finally, according to [17], a Deep Text User Interest System (DTUIS) is a deep learning model that utilizes Word2Vec and CNN for user interest classification by combining text classification and sentiment analysis.

In our study, we have trained a model for generating a summary of Twitter users' interests considering their social textual data in 5 categories: religion, cuisine, fashion, travel, and sports. Consequently, the researchers can discern the specific topics that pique consumers' interest. Drawing inspiration from the remarkable achievements of deep learning, we also leveraged pre-trained word embedding models, Word2vec for the underlying classification problem. This enables the system to generate vector representations of words, which serve as inputs for a suitable CNN architecture. By employing this architecture, the system can perform deep feature extraction, effectively capturing significant patterns and features from the text data.

## 3. Deep Learning Models

Deep learning models evolved from artificial neural networks and have gained success in many fields. This section investigates the deep learning models utilized for text classification tasks.

### *3.1 Recurrent Neural Network*

Recurrent Neural Networks (RNNs) was first proposed by Hopfeld in 1983 [18]. They are a robust type of artificial neural network that allows previous outputs to be utilized as inputs. It is the superior version of the feed-forward neural network that can mitigate the problem of a variable-length sequence utilizing a recurrent hidden state that considers the prior time sequences during activation [19]. While performing sequential information, the RNN employs recurrent hidden states, the activation of which is highly reliant on the previous time step, and the existing state has dependency over the current input. As a result, the present hidden state fully utilized the previous data. As a result, standardization is achieved. RNN can handle input of varying lengths and calculates sequence data in processes that are dynamic. The standard RNN architecture is given in Fig. 1. For a given input sequence X = [x1, x2,... xt...xT] of length T and a process output vector $O_t$ , the hidden state $U_o$ at time step t is determined using the following equations:

$$O_t = \varphi(W_x x_t + U_0 h_{t-1}) \quad (1)$$

$$H_t^l = \varphi(w_x h_{t-1}^l + U_H h_{t-1}^l) \quad (2)$$

where $W_x$ are the weights connecting input and hidden layers, and φ represents the activation function, i.e., sigmoid. $U_H$ is the hidden layers weights matrix and $Wx$, $U_o$, are parameters.

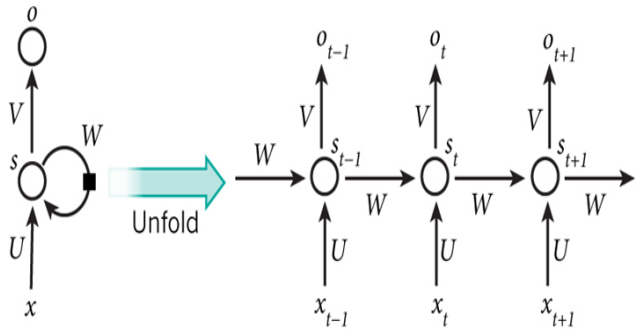


**Fig. 1.** Traditional Structure of RNN

The input unit of the time step is $X_{t-1}$, $h_t$ is the activation state of the step t. $O_t$ displays the output at time t; the output is chosen based on the network's requirements. Weights are the trainable parameters and denoted by U and W. RNNs, on the other hand, are difficult to train and suffer from disappearing and bursting gradients.

### *3.2 Long Short-Term Memory (LSTM)*

The LSTM is a variant of RNN that captures the long-term dependency. LSTM was introduced by Hochreiter and Schmidhuber in 1997. In the recurrent hidden layer of LSTM, there is a gated mechanism and particular units called memory blocks that assist in solving the two most prominent difficulties of standard RNNs, which are: vanishing gradient and explosion or exploding. Fig. 2 shows the architecture of LSTM gates.

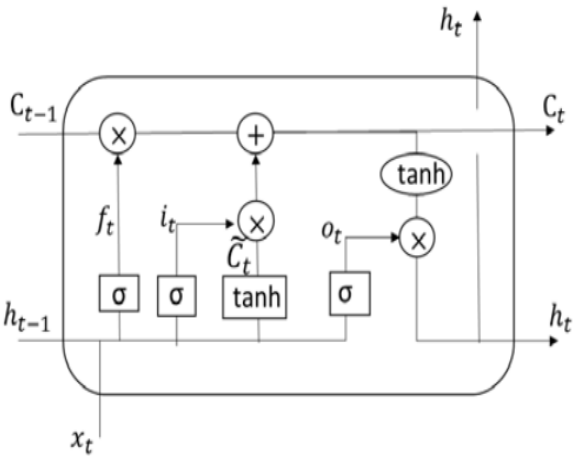


**Fig. 2.** The Architecture of LSTM Gates

In LSTM, the information can be added or deleted to the cell-by-cell state, which is considered the essence of LSTM. and electricity allows information to flux over the door technique to perfect this goal. There are three gates in LSTM as shown in Fig. 2: a forget gate ($f_t$), an input gate ($i_t$) and an output gate ($o_t$) through time step t [20]. firstly, forget gates decide the information to be removed from the cell state, and what information to update is the responsibility of the input gate. The output gate decides the network output. The Eq. (1)-(6) defines the state of each node in this operation.

$$f_t = \sigma(w_{xf}x_t + U_{hf}h_{t-1} + b_f) \quad (3)$$

$$i_t = \sigma(w_{xi}x_t + U_{hi}h_{t-1} + b_i) \quad (4)$$

$$O_t = \sigma(w_{xo}x_t + U_{ho}h_{t-1} + b_0) \quad (5)$$

$$\tilde{a}_t = tanh(w_{\widetilde{xa}}x_t + U_{\widetilde{ha}}h_{t\_1} + b_{\tilde{a}}) \quad (6)$$

$$h_t = O_t * tanh(c_t) \quad (7)$$

$$c_t = f_t * x_{t-1} + i_t * \tilde{a}_t \quad (8)$$

$c_t$ represents the memory cell content which was updated in Eq 8, $x_t$ represents the current input, $h_{t-1}$ represents the previous layer's hidden state, $f_t$ represents forget gate's output, $w$ and $U_{ho}$ are weights, b is a bias matrix, and $h_t$ denotes the hidden state of the next layer, σ denotes the sigmoid activation function. Weights and bias of the network can be computed throughout the training phase are $W_i$ , $W_o$, $W_f$ , $W_{\tilde{a}} \in R^{m*p}$, $U_i$ , $U_o$, $U_f$ , $U_{\tilde{a}} \in R^{m*m}$, $b_i$ , $b_0$,$b_f$ , $b_{\tilde{a}} \in R^{m*1}$.

$W_i$ , $W_o$, $W_f$ , $W_{\tilde{a}}$ are the parameters between the input and hidden layers, $U_i$ , $U_o$, $U_f$ , $U_{\tilde{a}}$ are weights between hidden and output layers. LSTM network has a strong patterning capacity for extracting long-term dependencies between sequence parts and is gaining prominence.

LSTM is an effective approach adopted in the field of text classification problems. Previous studies have improved the accuracy of text classification using LSTM. However, there is still room for improvement by removing drawbacks and required efforts to eliminate the limitation of standard LSTM. The first drawback is, having the nature of non-application independent as well as data dimensionality reduction corresponding to the given textual data [21]. This results in the incapability to acquire the minute details when possessing important information. Furthermore, it also gives a low quality of results when solving the classification problem. Secondly, LSTM also does not give accurate results when combined with other text classification approaches. The major cause of these issues is the complex representation of the variety of features that differentiate different classes.

### *3.3 The Basic Autoencoder*

Autoencoder (AE) is one of the essential types of unsupervised artificial neural networks (ANN) that play a vital role to reduce the dimensionality of data which can efficiently press and encode features after extraction and learn to regenerate the data from the latent representation as it gives the better latent representation of the inputs as compared to the inputs as near to its initial inputs as feasible [22].

AE has two stages encoding and decoding. The encoding phase generates the latent representation, and the decoding phase regenerates this new representation back to its original inputs as closely as possible. The primary goal of an autoencoder is to generate high-quality features from large textual data without any need for reducing the dimensionality. Currently, conventional AE is a great and efficient unsupervised model for encoding related information. The model trained using latent representations generated by the conventional AE is robust to existing noise and shows delicate fluctuations in the input training data. It learns the non-linear relationship in the data using a nonlinear activation function and multiple layers. Fig. 3. shown simple representation of the Autoencoder model architecture.

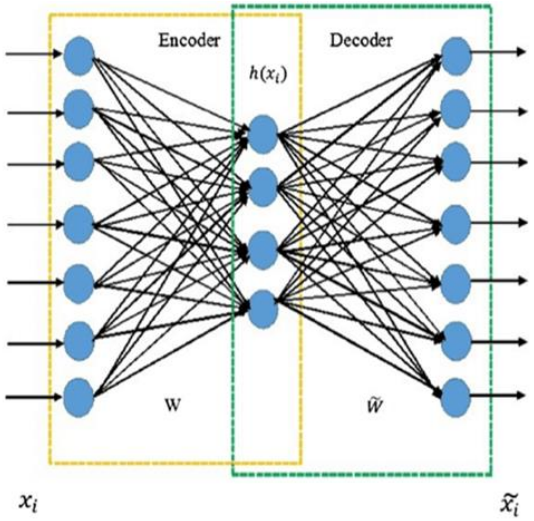


**Fig. 3.** Standard Autoencoder Architecture

The autoencoder has two stages: the encoder and the decoder. The former translates the input features set into a latent intermediates representation to be considered as an input to the hidden layer. Its main objective is to reduce the curse of dimensionality of the input data and can be represented as:

$$y = E(m) = k_e( W_x + b_h ) \quad (9)$$

where $k_e$ is the encoding activation function, $W_x$ are the weights while b represents the biases corresponding to the input features, and E(m) refers to the output of the input layer which is given as input to the next hidden layer. Similarly, the decoding stage regenerates the original input using different parameters from the latent representation as shown in Eq. (10).

$$z = F(n) = k_d( W_y + b_r) \quad (10)$$

where F(n) is the hidden layer output, W represents input weights of the hidden layer, b represents to the hidden layer, and $k_d$ represents the activation function.

The primary aim of reconstruction is to regenerate outputs almost similar to the original inputs by reducing the difference by using the following parameters:

$$\theta = [ W, b_h, b_r ] \quad (11)$$

Suppose, Xi = [x1, x2, x3, …xn] is the input features in the training phase, then following the reconstruction error can be optimized using the following function:

$$J_{AE}(\theta) = \Sigma_{x \in D_i} G(x, r) \quad (12)$$

Also, for a linear representation, where G represents the reconstruction error, while for a nonlinear representation, Euclidean distance is used as a cross-entropy loss for reconstruction. To handle the overfitting as well as controlling the huge weights obtained from the above Eq. (12). The Eq. (12) can be represented in a simple form as:

$$J_{AE\_}W_d(\theta) = \Sigma_{x \in D_i} G(x, r) + \frac{1}{2}(\alpha \|w\|) \frac{2}{2} \quad (13)$$

Here, α represents a weight decay coefficient for preventing the weights from getting very large.

*3.4 The Denoising Autoencoder*

Denoising autoencoder (DAE) is a minor modification to typical AEs that are trained to denoise an artificially distorted version of their input rather than recreate it [23]. A DAE must extract more valuable features to tackle the significantly more challenging denoising problem, in contrast to an overcomplete regular AE, which can learn worthless identity mapping with ease. Reconstructing noisy data from the original data is the primary function of DAEs. DAEs are used to both encode noisy data and retrieve the original input data from the output data that has been rebuilt. Data encoders that are stacked in various layers create stacked DAEs [24]. Notably, Xia et al [25] incorporated a weighted reconstruction loss function over the traditional DAE for speech-enhancement system noise categorization. to build the model, they stacked numerous weighted DAEs. In their trials, they carried out 50 steps using a variable number of input nodes, between 50 and 100. White noise with signal-to-noise ratios (SNRs) of 6, 12, and 18 dB was chosen from the NTT database and used to train the model, the un-noisy data consisted of 8 languages. The model was evaluated on data that was 8 minutes long after being trained on data that was 1 hour long.

According to Caterini et al [26], DAEs have attracted the attention of numerous researchers in a variety of contexts. In addition, by introducing noise to the training data, it learns a more robust representation of the input signal and exhibits stronger generalization ability than conventional encoders. The denoising autoencoder structure is shown in Fig. 4.

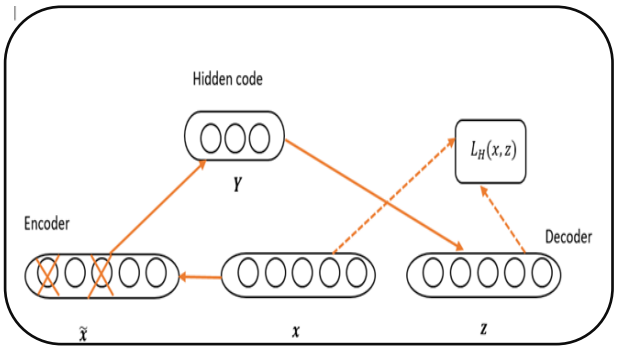


**Fig. 4.** Denoising Autoencoder Structure

As can be seen from Fig. 4. above, the encoder and decoder are also the main two parts of the DAE.

Data dimensions are reduced using the encoder. The coding result y is ultimately acquired through the actions of the activation function and linear transformation, which is subsequently input into the decoder.

$$y = f_\theta(x) = s(W\tilde{x} + b) \quad (14)$$

The primary role of the decoder, which is represented as the following function, is to map and reconstruct the data from the hidden layer back to z, which is represented as the following function:

$$z = f_\theta(y) = s(\tilde{w}y + b) \quad (15)$$

Where W is a weight matrix of $d \times d'$ and, b is a bias parameter. s is the activation function; all the parameters of the model are obtained by continuous reverse finetuning to obtain the minimum average reconstruction error. The expression of reconstruction error is as follows:

$$L_H = \|x - z\|^2 \quad (16)$$

## 4. Model Architecture

This section describes the proposed improved model architecture that integrates DAE into LSTM. This DAE-LSTM model architecture seeks to solve the issues and limitations of the standard LSTM linked to dimensionality reduction as well as the complexity of text classification problems. In this architecture, we combine denoising auto-encoder (DAE) layers with LSTM to improve LSTM's capability of dimensionality reduction. The essential computational process of developing the DAE-LSTM model for text classification contains four main phases for the training of both DAE layers and the LSTM model. In the first stage, the parameter sets of DAE as well as LSTM are initialized. Subsets of the parameters of DAE, such as activation functions, and learning rate are selected and then fed with input data. In the second stage, the encoding process takes place when translating input to the latent representation for the hidden layer as an input. The decoding procedure happens during information transformation from hidden to output layers optimizing the reconstruction error. The information is propagated to the third stage if the reconstruction error does not exceed the threshold value. Improved results were obtained based on a certain threshold value (find the best threshold value). The model modifies the threshold at each epoch to achieve the ultimate accuracy; the threshold did not further grow or drop. During the third stage, the DAE layer's output is transformed and becomes input to the LSTM for learning prominent and useful features. The embedding of the hidden layers of LSTM is directed towards the fourth stage, where the intermediate resultant features vector obtained from LSTM layers is passed through the Softmax layer for generating the predictions. Fig. 5. depicts the core architectural difference between the standard LSTM and DAE-LSTM model. The conventional LSTM architecture has already been discussed in Sect. 3.2. There are two major modifications have been made to improve the performance of standard LSTM. 1) a denoising auto-encoder is injected into an LSTM architecture, 2) tanh is replaced with Relu activation in candidate state, which the modification shown in Eq. (17)

$$\tilde{a}_t = Relu(w_{\widetilde{xa}} x_t + U_{\widetilde{ha}} h_{t-1} + b_{\tilde{a}}) \quad (17)$$

Relu has provided better performance as compared to other sigmoid nonlinearities for deep learning approaches. Fig. 5 shows the modification of the DAE-LSTM model.

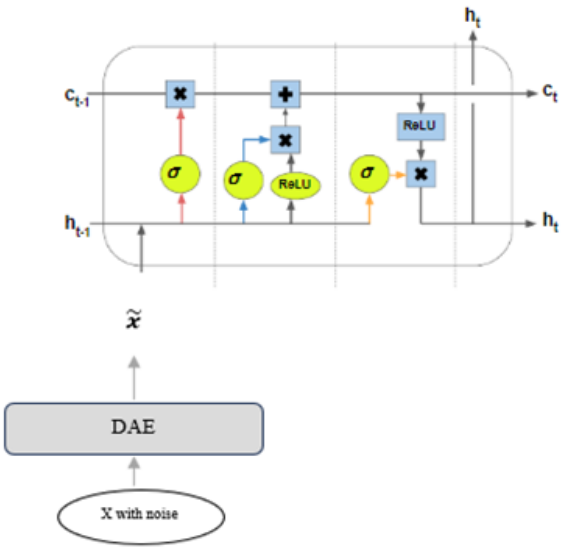


**Fig. 5.** The DAE-LSTM Architecture with Relu Function

## 5. Experiments

### *5.1 Dataset*

Training a machine learning model for text classification starts with the process of data collection. To train and evaluate our proposed DAE-LSTM model, we have used the 4 publicly benchmark datasets, i.e., 20 newsgroups, Reuters21578, Amazon reviews, and IMDB, freely available at UCI machine learning repository in various formats, i.e., pdf, CSV, doc, etc. Furthermore, for consistency, these datasets, as shown in Table 1 below, were selected and split into training and testing sets for comparing the proposed model with the four chosen existing models. It is important to note that in performing the comparison, we have to retrain these four other models with these benchmark datasets under our experimental setup.

**Table 1**

Descriptions summary of our experimental statistics datasets

| Datasets | Train Samples | Test Samples | Average lengths | Classes | Tasks |
|---|---|---|---|---|---|
| Amazon review | 300,000 | 125,000 | 156 | 5 | Sentiment analysis |
| IMDB | 35,000 | 15,000 | 55 | 02 | Sentiment analysis |
| Reuters21578 | 8088 | 2700 | 87 | 15 | Text classification |
| 20NG | 7500 | 2500 | 96 | 20 | Text classification |

*5.2 Flow Chart of the DAE-LSTM Model Development Process*

Fig. 6 shows the flowchart of the process for training and testing the proposed DAE-LSTM model for solving text classification problems. This process comprises three phases. The first phase involves data preprocessing for experimental purposes where spaces and punctuation are discarded to extract words. Keywords are chosen and unnecessary words are removed from these datasets. Porter stemmer is used to remove the inflectional form and to make the word its base form. This step is crucial because the quality of the input data is very important for training models using deep learning approaches. In the second phase, we initialize the parameters for the proposed DAE-LSTM model. In this phase, denoising auto-encoder layers were applied for feature reduction purposes while LSTM layers were used for learning the prominent features in the training process and optimizing the model parameters. The integration of the two introduces a novel technique of long short-term memory (LSTM) based on DAE that replaces tanh with the Relu activation function. The third phase is the validation process. To evaluate the performance of the proposed model different performance evaluation metrics are used for solving text classification issues. These comprise accuracy, precision, recall, F-measure, specificity, error rate/convergence, time complexity, and Mean Squared Error (MSE). DAE-LSTM model has numerous parameters that can influence its training. Depending on those parameters, the performance of the model can vary notably. the training hyperparameters were chosen in the proposed model DAE-LSTM, as shown in Table 2. Finally, upon passing this evaluation phase, a trained and tested DAE-LSTM model was produced with the configuration ready for deployment in performing the classification of further unseen texts.

**Table 2**

Hyperparameter for the DAE-LSTM model

| Hyperparameter | Values |
|---|---|
| Learning rate | 0.001 |
| Epochs | 12 |
| Optimizer | adam |
| Batch size | 32 |
| Activation function | Relu, softmax |
| Dropout | 0.5 |

*5.3 Experimental Setup*

To improve the performance of the proposed DAE-LSTM model, after cleaning the dataset by pre-processing techniques such as eliminating stop words from the sequential input and punctuation, the information was maintained through the training of sentence embeddings. Then, the Word2vec model was used to get 300-dimensional word vectors for all the words as performed in [27]. Some researchers implemented well-structured training strategies for getting word embeddings (word representations) to improve the accuracy of sentence classification tasks [28]. Adam algorithm with a learning rate of 0.001 and a mini-batch of 64 is used during the training phase. In our research, the common embeddings for all benchmark datasets are more useful to apply for getting good generalization ability of the model. The dropout strategy [29] was applied to avoid the overfitting problem, with a dropout rate of 0.5. All the experiments were implemented using Python 3.7 with Anaconda running on Intel Core i7-3770, 3.40 GHz, (CPU)," and 8 GB of Random-Access Memory (RAM). Other required libraries that are used in this research work are, Sklearn, Numpy, Scipy, Pandas, and Keras packages, running on Microsoft Windows 10 64-bit operating system. For quick implementation of the proposed model and comparative traditional deep learning models executed, an efficient open-source software library for numerical computational is applied which allows a simple and fast development for both Central Processing Unit (CPU) and Graphics Processing Unit (GPU) support. The performance of all methods was based on pre-defined assessment measures as mentioned in the scope of this research.

*5.4 Performance Evaluation*

There are different performance evaluation metrics used for solving text classification issues, this research uses accuracy, precision, recall, F-measure, and error rate/convergence, to evaluate the performance. The formulas are as follows.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$\text{Precision} - \frac{TP}{(TP + FP)}$$

$$\text{Sensitivity} = \text{Recall} = \frac{TP}{(TP + FN)}$$

$$\text{Error rate} = \frac{(FP + FN)}{(TP + TN + FP + FN)}$$

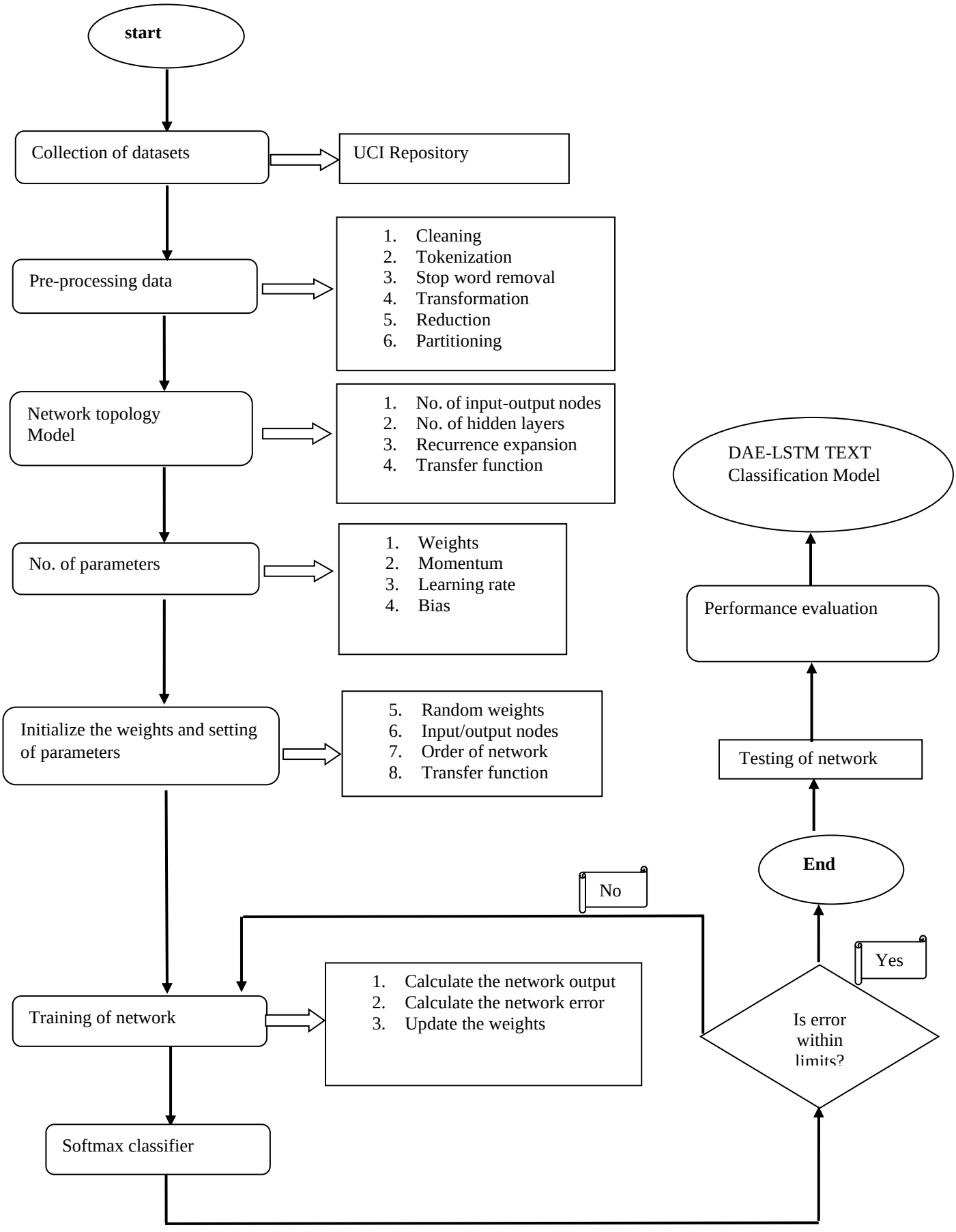


**Fig. 6.** The Flowchart of The DAE-LSTM Model Development Process

## 6. Results and Analysis

In the analysis of the results, the empirical results of the proposed and comparative approaches were briefly analyzed based on the selected datasets.

### *6.1 Accuracy Results of All Models*

This section presents improved results of a proposed (DAE-LSTM) model on the four benchmark datasets and compares them with the standard text classification approaches containing BiLSTM, BiGRU, CNN-LSTM, and CNN-GRU. These four models were retrained and tested under the same experiment setup and datasets before comparing them with the DAE-LSTM model. In the evaluation, we have also considered other important factors, such as the curse of dimensionality as well as imbalanced classes. Evaluation results of the proposed and the conventional LSTM are shown in separate figures. Several performance measures are used for performance evaluation. Table 3 shows the accuracy of the proposed model as well as other approaches. These results indicate that with the increase in complexity and size of data, the proposed model improves its performance.

The obtained result shows the significance and effectiveness of the proposed model for complex data by giving improved accuracy on the benchmark datasets comparatively as shown in Table 3. The improvement is due to the model's ability for robustness to the noise, and high-quality latent representation generation for giving as an input to the LSTM. For consistency, each experiment was repeated 5 times for each of the models. These results conclude that the proposed model outperforms the existing models, BiLSTM, BiGRU, CNN-LSTM, and CNN-GRU.

**Table 3**

Accuracy of the proposed and existing models (%)

| Datasets | BiGRU | BiLSTM | CNN-LSTM | CNN-GRU | DAE-LSTM |
|---|---|---|---|---|---|
| Reuter21578 | 0.8787 | 0.8760 | 0.8795 | 0.8790 | 0.8893 |
| 20newgroup | 0.9062 | 0.9013 | 0.9134 | 0.9089 | 0.9203 |
| Amazon reviews | 0.8319 | 0.8488 | 0.8501 | 0.8541 | 0.8603 |
| IMDB | 0.8636 | 0.8699 | 0.8721 | 0.8765 | 0.8937 |

### *6.2 Precision and Recall*

Precision is the ratio of the number of correctly predicted positive observations to the total the total positive observations. Recall is the ratio of the number of correct predictions on positive instances to the total number of actual class instances. Generally, both measures are used for binary classification problems, therefore, we show average measures by first calculating these metrics for each class individually and then taking the average of them to obtain the final values. Table 4 shows the precision while Table 5 shows a recall of the existing as well as proposed models.

The last row of table 4, shows the improved results of the proposed model compared to the existing models. In the first two datasets (reuters21578 and 20newgroup) the CNN-GRU and CNN-LSTM performed nearly to the proposed DAE-LSTM due to the less complexity of the datasets. Also, along with the increase in dataset complexities, the proposed model exhibits better performance in comparison to BiGRU, and BiLSTM.

**Table 4**

Precision-based evaluation of proposed and existing models on benchmark datasets

| | Datasets | | | |
|---|---|---|---|---|
| Models | Reuter21578 | 20NG | IMDB | Amazon review |
| CNN-GRU | 0.85 | 0.89 | 0.86 | 0.80 |
| CNN-LSTM | 0.86 | 0.89 | 0.85 | 0.81 |
| BiLSTM | 0.85 | 0.88 | 0.83 | 0.80 |
| BiGRU | 0.84 | 0.86 | 0.84 | 0.79 |
| DAE-LSTM | 0.88 | 0.92 | 0.89 | 0.83 |

**Table 5**

Recall-based evaluation of proposed and existing models on benchmark datasets

| | Datasets | | | |
|---|---|---|---|---|
| Models | Reuter21578 | 20NG | IMDB | Amazon review |
| CNN-GRU | 0.84 | 0.89 | 0.79 | 0.77 |
| CNN-LSTM | 0.85 | 0.90 | 0.80 | 0.79 |
| BiLSTM | 0.84 | 0.88 | 0.82 | 0.81 |
| BiGRU | 0.83 | 0.87 | 0.87 | 0.79 |
| ES-GRU | 0.87 | 0.92 | 0.89 | 0.82 |

*6.3 Convergence Rate*

This section compares the convergence rate of the proposed model and existing models. “Fig. 7, 10,11, 12, and 13” show the convergence rate of these models while learning classification for benchmark datasets for 12 iterations. After 6 iterations, all models maintain their error rates. The proposed model shows slightly inferior performance as compared to the existing models for Amazon review and router21578 datasets. For the complex datasets, i.e., IMDB and 20newsgroup as compared to the other simple datasets, the proposed model is consistent in convergence at good value throughout the 12 iterations while others are affected at certain extent.

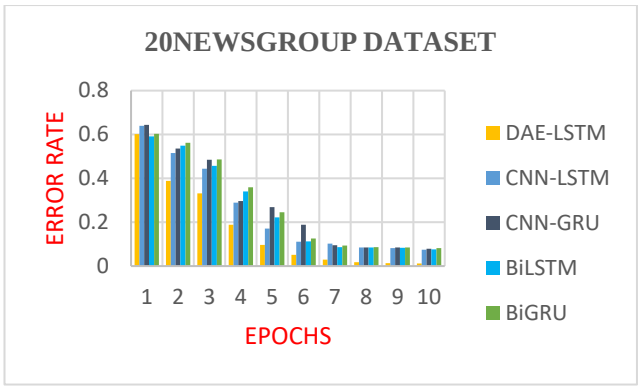


**Fig. 7.** Convergence Rate of Models on 20newsgroup Dataset

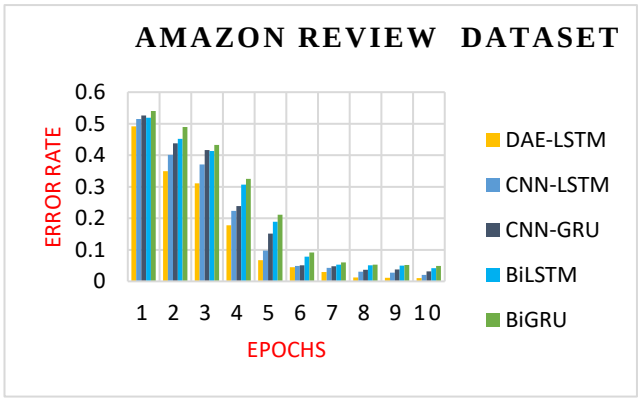


**Fig. 8.** Convergence Rate of All Models on Amazon Dataset

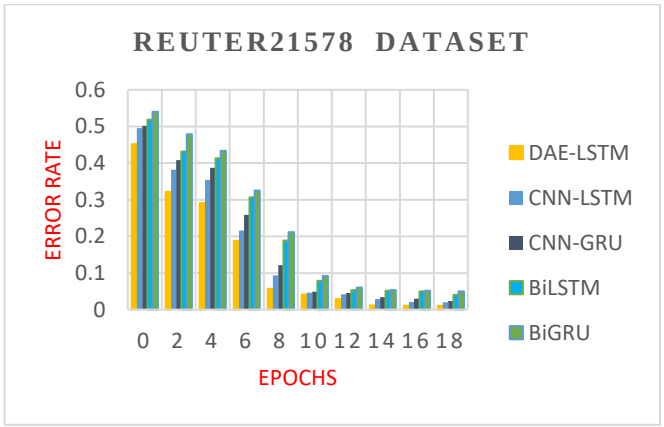


**Fig. 9.** Convergence Rate of All Models on Reuter21578 Dataset

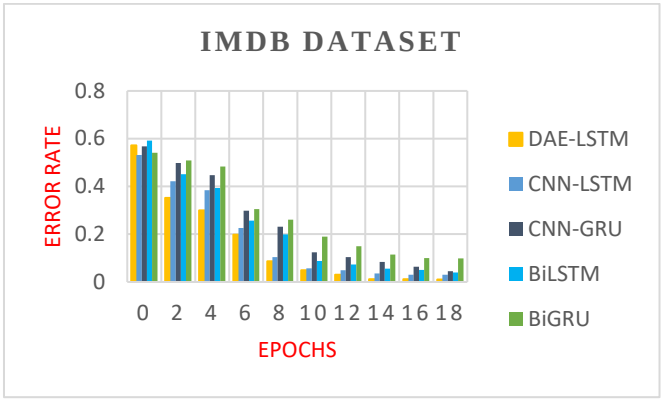


**Fig. 10.** Convergence Rate Of All Models On The IMDB Dataset

## 7. Conclusion

Natural Language Processing (NLP) is the common and pertinent field of text classification. LSTM is one popular and widely used deep learning model in NLP. In this study, we proposed an enhanced variant of LSTM based on a denoising autoencoder, DAE-LSTM for solving the problems of text classification. This research has made three contributions. First, the integrated DAE-LSTM approach for learning the latent representation of the input features can significantly reduce dimensions to mitigate the curse of dimensionality. Secondly, we have also shown the utility of replacing the tanh with Relu activation and a Softmax output layer for getting classification outputs for input data. Finally, the proposed model has gone through sufficient experiments for comparative analysis and evaluation to ensure its effectiveness and applicability. In summary, the proposed model has been shown to outperform other state-of-the-art models on four benchmark datasets (20NG, Amazon review, IMDB, and Reuters21578) and achieved outstanding classification accuracy with less execution time. For future work, the proposed model will be implemented for other text and data mining problems, such as information retrieval and machine translation for evaluating its performance in general.

## 8. References

[1] N. Shahid, T. Rappon, and W. Berta, "Applications of artificial neural networks in health care organizational decision-making: A scoping review", PloS one, vol. 14, no. 2, p. e0212356, 2019.

[2] Z. Ullah, F. Saleem, M. Jamjoom, and B. Fakieh, “Reliable prediction models based on enriched data for identifying the mode of childbirth by using machine learning methods: development study”, Journal of Medical Internet Research, vol. 23, no. 6, pp. e28856, 2021.

[3] D. Alessia, F. Ferri, P. Grifoni, and T. Guzzo, "Approaches, tools and applications for sentiment analysis implementation", International Journal of Computer Applications, vol. 125, no. 3, 2015.

[4] W. Sharif, N. A. Samsudin, M. M. Deris, and M. Aamir, “Improved relative discriminative criterion feature ranking technique for text classification”, Int. J. Artif. Intell, vol. 15, no. 2, pp. 61-78, 2017.

[5] Y. Zhou and X. Jiang, "Dissecting android malware: Characterization and evolution", 2012 IEEE symposium on security and privacy, 2012: IEEE, pp. 95-109.

[6] R. O. Midigo, W. Mwangi, and G. O. Okeyo, "Biterm for spam filtering in short message service text", International Journal of Computer Science Issues (IJCSI), vol. 14, no. 1, p. 79, 2017.

[7] R. Rajalakshmi and C. Aravindan, "Naive Bayes approach for website classification", International Conference on Advances in Information Technology and Mobile Communication, 2011: Springer, pp. 323-326.

[8] L. Leape et al., "Transforming healthcare: a safety imperative", BMJ Quality and Safety, vol. 18, no. 6, pp. 424-428, 2009.

[9] J. Su et al., "Wheat yellow rust monitoring by learning from multispectral UAV aerial imagery", Computers and electronics in agriculture, vol. 155, pp. 157-166, 2018.

[10] S. Zhen-zhong, M. Xiao-ning, W. Wei, and X. Li-qun, "Deformation prediction model of concrete arch dam based on improved particle swarm optimization algorithm", Earth and Space 2010: Engineering, Science, Construction, and Operations in Challenging Environments, 2010, pp. 443-451.

[11] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning", arXiv preprint arXiv:1605.05101, 2016.

[12] Y. Cai, Q. Huang, Z. Lin, J. Xu, Z. Chen, and Q. Li, "Recurrent neural network with pooling operation and attention mechanism for sentiment analysis: A multi-task learning approach", Knowledge-Based Systems, vol. 203, p. 105856, 2020.

[13] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms", Mining text data, pp. 163-222, 2012.

[14] L. Cai and T. Hofmann, "Text categorization by boosting automatically extracted concepts", Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, 2003, pp. 182-189.

[15] S. Hingmire, S. Chougule, G. K. Palshikar, and S. Chakraborti, "Document classification by topic labeling", Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, 2013, pp. 877-880.

[16] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification", Proceedings of the AAAI conference on artificial intelligence, 2015, vol. 29, no. 1.

[17] A. H. Ombabi, O. Lazzez, W. Ouarda, and A. M. Alimi, "Deep learning framework based on Word2Vec and CNN for users interests classification", 2017 Sudan conference on computer science and information technology (SCCSIT), 2017: IEEE, pp. 1-7.

[18] S. Ranjit, S. Shrestha, S. Subedi, and S. Shakya, "Comparison of algorithms in foreign exchange rate prediction", 2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS), 2018: IEEE, pp. 9-13.

[19] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling", arXiv preprint arXiv:1412.3555, 2014.

[20] M. Abdel-Nasser and K. Mahmoud, "Accurate photovoltaic power forecasting models using deep LSTM-RNN", Neural computing and applications, vol. 31, pp. 2727-2740, 2019.

[21] K. Murphy and C. Lepers, "Fault Prediction for Heterogeneous Networks using Machine Learning: a Survey", Authorea Preprints, 2023.

[22] L. Zhang, G.-J. Qi, L. Wang, and J. Luo, "Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data", in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2547-2555.

[23] Z. Bao and L. Muñoz-González, "Mitigating evasion attacks against machine learning systems through dimensionality reduction and denoising", 2018.

 

[24] M. Chen, K. Q. Weinberger, Z. Xu, and F. Sha, "Marginalizing stacked linear denoising autoencoders", The Journal of Machine Learning Research, vol. 16, no. 1, pp. 3849-3875, 2015.

[25] B. Xia and C. Bao, "Wiener filtering based speech enhancement with weighted denoising auto-encoder and noise classification", Speech Communication, vol. 60, pp. 13-29, 2014.

[26] A. L. Caterini, A. Doucet, and D. Sejdinovic, "Hamiltonian variational auto-encoder", Advances in Neural Information Processing Systems, vol. 31, 2018.

[27] S. JP, V. K. Menon, S. KP, and A. Wolk, "Generation of cross-lingual word vectors for low-resourced languages using deep learning and topological metrics in a data-efficient way", Electronics, vol. 10, no. 12, p. 1372, 2021.

[28] Q. T. Ain et al., "Sentiment analysis using deep learning techniques: A review", International Journal of Advanced Computer Science and Applications, vol. 8, no. 6, 2017.

[29] S. Wang et al., "Defensive dropout for hardening deep neural networks under adversarial attacks", 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018: IEEE, pp. 1-8.

 