Advances in Intelligent Systems Research, volume 172
Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019)

Sequential Models for Text Classification Using Recurrent Neural Network

Winda Kurnia SARI[1], Dian Palupi RINI[2], Reza Firsandaya MALIK[3*], Iman Saladin B. AZHAR[4]

[1]*windakurniasari@unsri.ac.id, Faculty of Computer Science, Universitas Sriwijaya, Indonesia*
[2]*dprini@unsri.ac.id, Faculty of Computer Science, Universitas Sriwijaya, Indonesia*
[3]*rezafm@unsri.ac.id, Faculty of Computer Science, Universitas Sriwijaya, Indonesia*
[4]*imansaladin@ilkom.unsri.ac.id, Faculty of Computer Science, Universitas Sriwijaya, Indonesia*
**Corresponding author: rezafm@unsri.ac.id*

ABSTRACT
Neural network-based applications are recently shown promising results for text classification. However, it is still challenging for the model to contemplate local features and word contingent on the information of the sentence. This work proposed a deep learning approach to generate a more precise sentence that leverages the preceding texts when classifying a subsequent one. One of the deep learning methods used is Recurrent Neural Network (RNN) with the architecture Long Short-Term Memory (LSTM). By training four variant models of 1-layer LSTM for each balance dataset in pre-processing process with 20,000, 25,000, 30,000, 35,000, 40,000, and 45,000 using optimizer Adam and RMSProp. The results show that; first, the more data input, the higher the accuracy it gets and the second is Adam can perform better as optimizer than RMSProp in this research. The highest Precision, Recall, and F1-score obtain are 97.
Keywords: *sequential, text classification, glove, multilabel.*

INTRODUCTION

Text classification is one of the main parts in Natural Language Processing (NLP) work which has applied in many fields, such as sentiment analysis [1], document classification [2], text categorization [3], information retrieval [4].

Previous research in text classification has proposed traditional machine learning methods such as Naïve Bayes [5], K-Nearest Neighbor [6], Support Vector Machine [7], and Logistic Regression [8]. The traditional machine learning algorithm has solved in text classification work, but it occurs limitations when processing multi-label data and large datasets [8].

In recent activity, neural network-based models are becoming more popular [9]. Deep learning is an artificial neural network with multiple layers both input and output. One of the deep learning methods used in this study is Recurrent Neural Network (RNN) with architecture Long Short-Term Memory (LSTM). The most popular architecture used in NLP is RNN because the structure of recurrent is compatible with long variable text processing. However, RNN has a problem called gradient vanishing and exploding during training, so LSTM purposes to solve the problem [10].

MODEL

Word Representation

Word representation is also known as word embedding, a group of language modeling and feature learning technique in NLP where words from the corpus are plotted into vectors. One of the word embeddings used in this study is Global Vector (Glove) which achieved by plotting words into a valuable space where the distance between words is related to semantic similarity [11]. Word embedding with Glove use for training is glove.6B.50d which means 6 billion tokens and 400,000 vocab with 50d vectors.

Optimization

In this study use two deep learning optimizers, Adam and RMSProp. Adam can control sparse gradient issues [12].

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

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

where *m* and *v* refer averages for the first two moments of the gradient, *g* shows gradient on current mini-batch. Optimizer RMSPRop can adapt the learning rate for each hyper-parameter. It intends to divide the learning rate for weight.

$$v(w,t) := \gamma v(w,t-1) + (1-\gamma)(\nabla Q_i(w))^2 \tag{3}$$

Recurrent Neural Network (RNN)

The traditional RNN has a problem called gradient vanishing and exploding during training. RNN is one of the deep learning categories because data is processed automatically and without defining features [13]. RNN does not just throw away information from the past in the learning process. This distinguishes RNN from a common Artificial Neural Network. RNN is a part of the Neural Network for processing sequential data. How RNN can store information from the past is looping inside its architecture, which automatically keeps information from the past stored. RNN can use the internal states (memory) to process the input sequence. It makes RNN capable of tasks such as Natural Language Processing (NLP) [14], speech recognition [15], handwriting recognition [16]. Two architectures of RNN are Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

Basic equation of RNN,

$$s_t = \tanh(U_{x_t} + W_{s_{t-1}}) \tag{4}$$

$$\hat{y}_t = softmax(V_{s_t}) \tag{5}$$

Long Short-Term Memory (LSTM)

Long-Short Term Memory (LSTM) as seen in Fig. 1 is one of RNN architectures, it has become a popular tool among NLP researchers for the supreme capability to model and learn from sequential data. Models LSTM has shown outstanding results in many domains such as language modeling, tagging problem, and sequence-to-sequence predictions. LSTM aims to solve the problem of RNN called gradient vanishing and exploding [10]. LSTM replaces hidden vectors from recurrent neural networks with memory blocks equipped with gates. The LSTM gates perform three layers; input gate, forget gate, and output gate.
The steps of LSTM cell and its gates as down below:

Input Gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{6}$$

$$\check{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{7}$$

Forget Gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{8}$$

Memory State:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{9}$$

Output Gate:

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o) \tag{10}$$

$$h_t = o_t * \tanh(C_t) \tag{11}$$

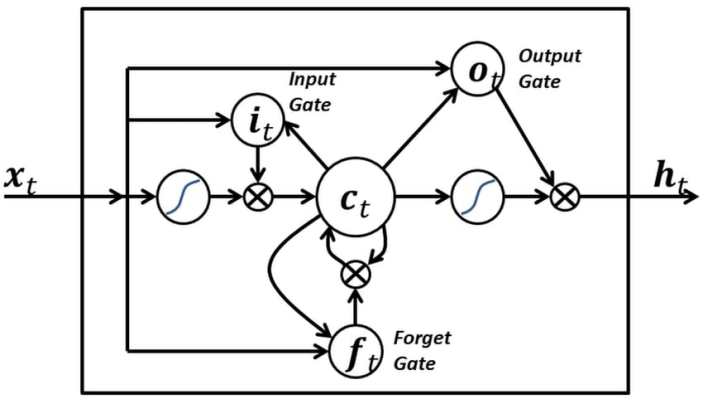


Figure 1 Architecture of LSTM

Experiments

Dataset

Dataset used in this study is based on paper Zhang et.al., 2015 [17] AGNews dataset. AGNews is a classification of topics in four categories of Internet news articles consisting of titles and descriptions classified into four classes: World, Entertainment, Sports, and Business. The dataset is shown in Table 1, with the following content specifications:

Table 1. Detail of Dataset

Label	Contains
0	152,469
1	115,967
2	108,344
3	45,639
Total	**422,419**

This study prepares the pre-processing data with balancing dataset by splitting to be 20,000, 25,000, 30,000, 35,000, 40,000, and 45,000 data for each label in Fig. 2.

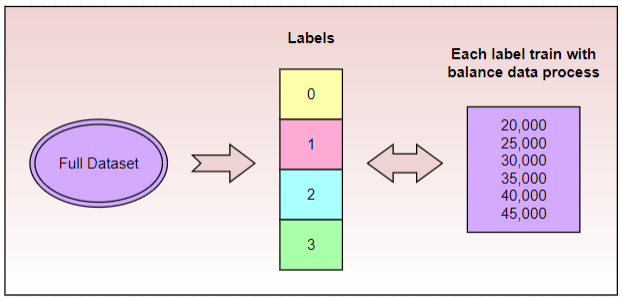


Figure 2. Balance Data Process from Dataset

Research Methodology

The stages in research methodology used to assist in arranging this study need a clear framework. The research methodology is shown in Fig. 3 which contains literature reviews, data preparation, pre-processing data, RNN architectures, classification with LSTM, result analysis, and summary.

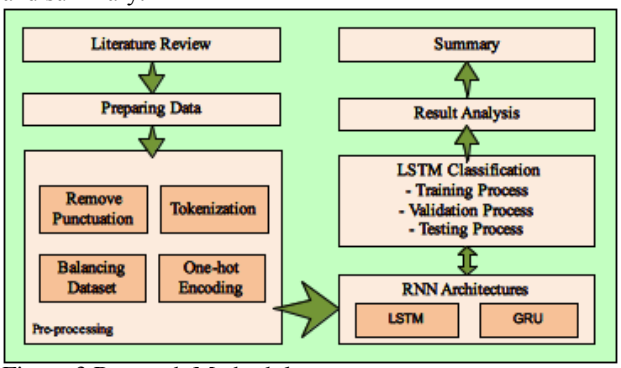


Figure 3 Research Methodology

PRE-PROCESSING PROCESS

Tokenization and Remove Punctuation

Tokenization is the process of breaking the stream of text into words, phrases, symbols, or other meaningful elements called tokens. Tokenizing means splitting the text into units that have minimal meaning. This is a mandatory step before any type of processing. This process will divide the text into sentences and sentences into typographic tokens. That means separating punctuation. The features generated from tokenizing will be training data.

Balance Dataset

In this process, the data is balanced on the dataset, by entering the amount of data that is not more than the smallest number of labels, 45,000. In this study, data balancing has been carried out in the form of 20,000, 25,000, 30,000, 35,000, 40,000 and 45,000. For example, in 45,000 balance data there will be as many as 45,000 rows on each label 0, 1, 2, and 3 etc. It can be seen in Fig. 4.

```
num_of_categories = 45000
shuffled = data.reindex(np.random.permutation(data.index))
e = shuffled[shuffled['CATEGORY'] == 'e'][:num_of_categories]
b = shuffled[shuffled['CATEGORY'] == 'b'][:num_of_categories]
t = shuffled[shuffled['CATEGORY'] == 't'][:num_of_categories]
m = shuffled[shuffled['CATEGORY'] == 'm'][:num_of_categories]
concated = pd.concat([e,b,t,m], ignore_index=True)
#Shuffle the dataset
concated = concated.reindex(np.random.permutation(concated.index))
concated['LABEL'] = 0
```

Figure 4 Process of Balancing Dataset

One-hot encoding

After the process of balancing data based on the input amount of data, next is labeling data with one-hot encoding. Machine learning algorithms cannot work with categorical data directly. Categorical data must be converted into numbers. This applies because the research working with a sequence classification type problem and using deep learning methods such as Long Short-Term Memory Recurrent Neural Networks. Fig. 5 shows the process.

```
concated.loc[concated['CATEGORY'] == 'e', 'LABEL'] = 0
concated.loc[concated['CATEGORY'] == 'b', 'LABEL'] = 1
concated.loc[concated['CATEGORY'] == 't', 'LABEL'] = 2
concated.loc[concated['CATEGORY'] == 'm', 'LABEL'] = 3
print(concated['LABEL'][:10])
labels = to_categorical(concated['LABEL'], num_classes=4)
print(labels[:10])
if 'CATEGORY' in concated.keys():
    concated.drop(['CATEGORY'], axis=1)
'''
 [1. 0. 0. 0.] e
 [0. 1. 0. 0.] b
 [0. 0. 1. 0.] t
 [0. 0. 0. 1.] m
'''
```

Figure 5 Process of One-Hot Encoding

Training Process

Dataset AGNews split into 80% for training and 20% for testing. From splitting training data 80%, 10% is used for the validation process. The amount of each dataset is randomly split with an automatic data split which can be seen in Table 2.

Table 2. Split Dataset

Balance Dataset	Training	Testing
20.000	57.600	16.000
25.000	72.000	20.000
30.000	86.400	24.000
35.000	100.000	28.000
40.000	115.200	32.000
45.000	129.600	36.000

TRANING SEQUENTIAL MAODELS

Hyper-parameter trained in this study use learning rate 0.001, Neuron 50, dropout 0.2, batch size 32, and embedding dimension of Glove 50d. The detail hyper-parameter use with LSTM in this study can be shown in Table 3.

Table 3. Training Sequential Models

	Optimizer	Loss Function	Activation Function	
			Hidden	Output
Model 1	Adam	Categorical Cross Entropy	Relu	Softmax
Model 2	Adam	Categorical Cross Entropy	Tanh	Softmax
Model 3	RMSProp	Categorical Cross Entropy	Tanh	Softmax
Model 4	RMSProp	Categorical Cross Entropy	Relu	Softmax

RESULTS ANALYSIS

From training four models of LSTM with 1-Layer for each balance dataset, the results show that the more data input can increase the accuracy, precision, recall, and F1-score, also decrease the overfitting during training. In this study, Model 1 and Model 2 have shown the best results than Model 3 and Model 4 which is the difference in the optimizer. The accuracy results can be shown in Table 4. It shows that overall in Model 1 and 2 get higher training accuracy than Model 3 and 4. As known that Model 1 and 2 use Adam optimizer with activation Tanh and Relu for each training, while Model 3 and 4 use RMSProp optimizer for training model with the same parameters. It indicates good performance results of those models. For the highest training accuracy shows in Table 4 is 35,000 data on Model 2 with 96.53. Table 5 shows the results comparison to previous researchs.

Table 4. Training Accuracy

Data	Model 1	Model 2	Model 3	Model 4
20,000	95.59	95.74	91.61	91.57
25,000	95.94	96.01	92.00	91.66
30,000	96.35	96.50	92.67	92.35
35,000	96.27	96.53	92.99	92.08
40,000	96.02	96.03	92.54	92.19
45,000	96.02	96.10	92.61	92.17

Table 5. Results Comparison

Model	AGNews
Bag-of-words (Zhang et al.,2015)	88,8
Small word CNN (Zhang et al.,2015)	89,13
Large word CNN (Zhang et al.,2015)	91,45
LSTM (Zhang et al.,2015)	86,06
Deep CNN (29 layer) (Conneau et al.,2017)	91,27
SWEM (Shen et al.,2018)	92,24
fastText (Joulin et al.,2016)	92,5
LEAM (Wang et al., 2018)	92,45
LEAM (linear) (Wang et al., 2018)	91,75
Proposed Model*	**94**

As a comparison, the differences among the accuracy of training, validation, and testing can be shown in Fig. 6. From the proposed model, the confusion matrix results obtain the highest is in 45,000 data with Adam as an optimizer. The training accuracy may get higher in Model 1 and 2 with Adam than Model 3 and 4, but the testing accuracy shows that Model 3 and 4 get not much change for accuracy.

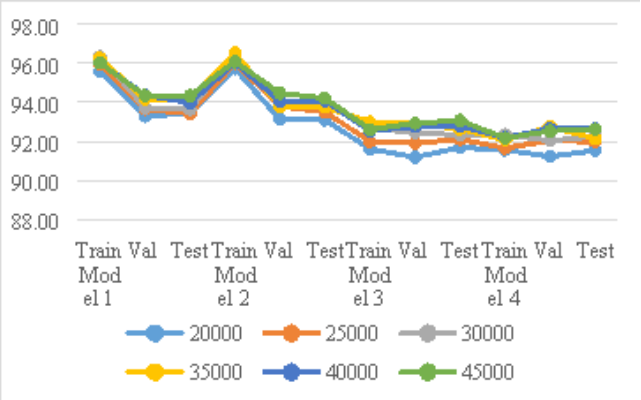
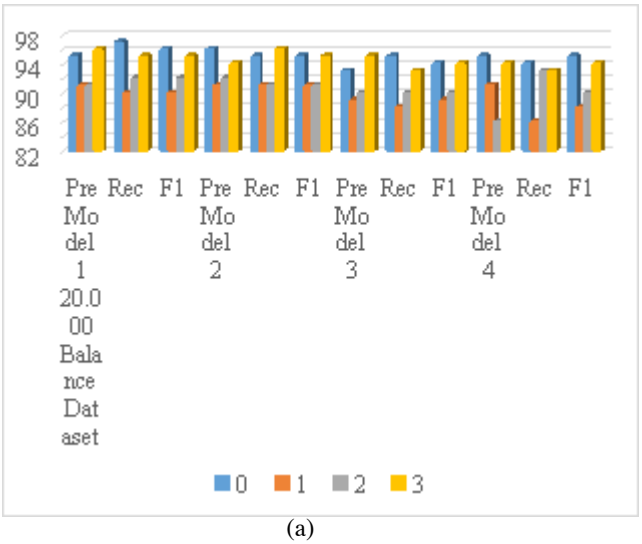
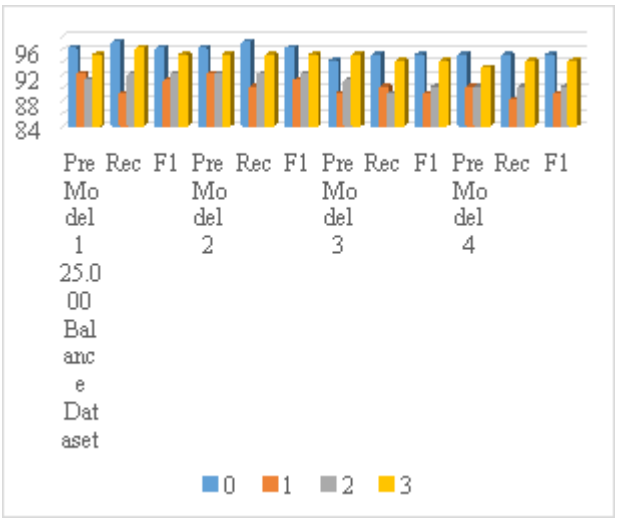


Figure 6 Accuracy of Training, Validation, and Testing

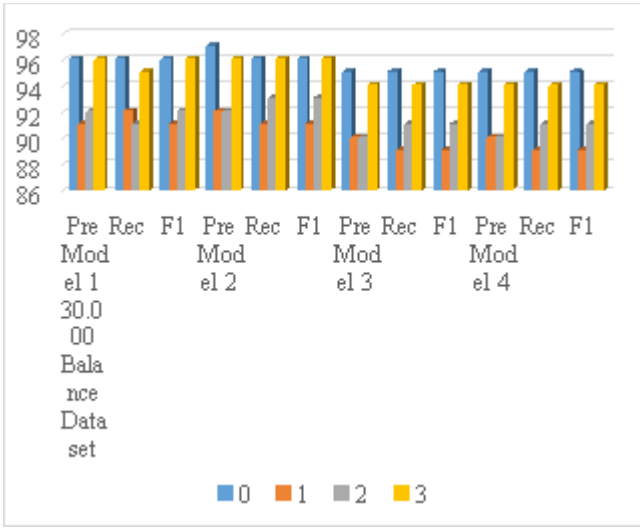
Fig. 7 shows result of metrics performance balancing data from dataset with 20,000, 25,000, 30,000, 35,000, 40,000, and 45,000 data. The maximum value of precision, recall, and F1-score get in this experiment is between 94-97, while the minimum is between 86-89. This means sequential models proposed have good performance.



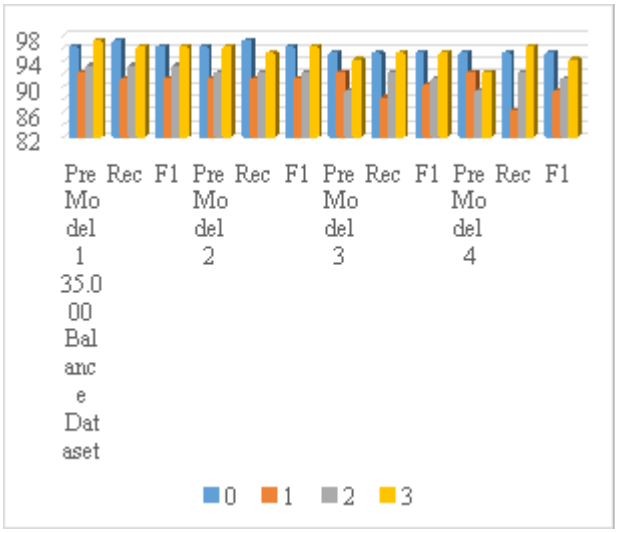
(a)



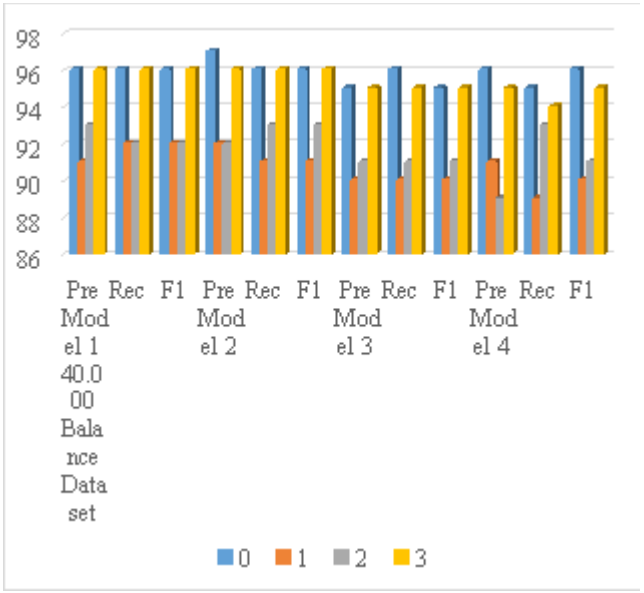
(b)



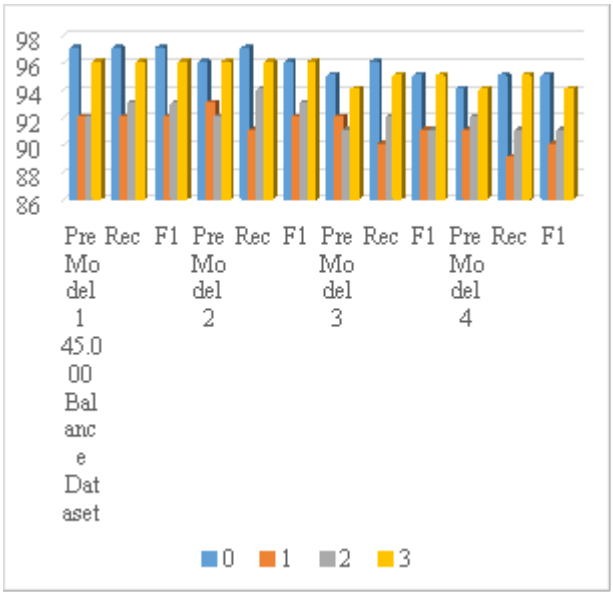
(c)



(d)



(e)



(f)

Figure 7 Precision, Recall, F1-Score Comparison in 20,000 Data (a), 25,000 Data (b), 30,000 Data (c), 35,000 (d), 40,000 (e), and 45,000 (f)

SUMMARY

Based on the experiments above, it can be concluded that the RNN-LSTM with sequential models proposed has succeeded in classifying the text. By training 4 models of 1-Layer LSTM with different hyper-parameters, the highest precision, recall, and f1-score results on Adam optimizer are 97. While the highest accuracy is 96.53%.

REFERENCES

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, Ng, Y. Andrew, and C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, Proceedings of the 2013 conference on empirical methods in natural language processing. 2013, pp. 1631-1642.

Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, Hierarchical attention networks for document classification, In Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies. pp. 1480-1489.

G. Chen, D. Ye, Z. Xing, J. Chen, and E. Cambria, 2017, Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 2377-2383). IEEE.

J. Lilleberg, Y. Zhu, and Y. Zhang, Support vector machines and word2vec for text classification with semantic features, In 2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC). IEEE, pp. 136-140.

L. Jiang, C. Li, S. Wang, and L. Zhang, Deep feature weighting for naive Bayes and its application to text classification, Engineering Applications of Artificial Intelligence, 52, 2016, pp. 26-39.

M. Azam, T. Ahmed, F. Sabah, and M. I. Hussain, Feature Extraction based Text Classification using K-Nearest Neighbor Algorithm, IJCSNS Int. J. Comput. Sci. Netw. Secur, 18, 2018, pp. 95-101.

M. Fanjin, H. Ling, T. Jing, and W. Xinzheng, The Research of Semantic Kernel in SVM for Chinese Text Classification, In Proceedings of the 2nd International Conference on Intelligent Information Processing, p. 8. ACM, 2017.

M. Gao, T. Li, and P. Huang, Text Classification Research Based on Improved Word2vec and CNN, In International Conference on Service-Oriented Computing, pp. 126-135. Springer, Cham, 2018.

Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882. 2014.

Y. Yan, Y. Wang, WC. Gao, BW. Zhang, C. Yang, and XC. Yin, LSTM2: Multi-Label Ranking for Document Classification, Neural Processing Letters 47, no. 1, 2018, pp. 117-138.

A. Alberto, O. Alfonso,T. António, M. Carmen, H. Carlos, P. Fernando, B. Fernando, M. Nuno. Advances in Speech and Language Technologies for Iberian Languages: Third International Conference, IberSPEECH 2016, Lisbon, Portugal, November 23-25, 2016, Proceedings. Cham: Springer. p. 165. ISBN 9783319491691.

D. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.

T. Wiatowski, and H. Bölcskei, 2017, A mathematical theory of deep convolutional neural networks for feature extraction. IEEE Transactions on Information Theory, 64(3), pp.1845-1866.

K. Kowsari, D.E. Brown, M. Heidarysafa, K.J. Meimandi, M.S. Gerber, and L.E.Barnes, 2017, Hdltex: Hierarchical deep learning for text classification. In 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 364-371). IEEE.

H. Zen, and H. Sak. 2015. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4470-4474). IEEE.

Graves, Alex, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. IEEE transactions on pattern analysis and machine intelligence 31, no. 5 (2008): 855-868.

X. Zhang, J. Zhao, and Y. LeCun, 2015, Character-level convolutional networks for text classification. In Advances in neural information processing systems (pp. 649-657).