Hindawi
Scientific Programming
Volume 2022, Article ID 6028693, 8 pages
https://doi.org/10.1155/2022/6028693

Research Article

Natural Language Processing with Improved Deep Learning Neural Networks

YiTao Zhou

Hubei Research Center for Language and Intelligent Information Processing, Wuhan University, Wuhan 430072, China

Correspondence should be addressed to YiTao Zhou; jamesz722@whu.edu.cn

Received 10 October 2021; Revised 15 December 2021; Accepted 21 December 2021; Published 7 January 2022

Academic Editor: Rahman Ali

As one of the core tasks in the field of natural language processing, syntactic analysis has always been a hot topic for researchers, including tasks such as Questions and Answer (Q&A), Search String Comprehension, Semantic Analysis, and Knowledge Base Construction. This paper aims to study the application of deep learning and neural network in natural language syntax analysis, which has significant research and application value. This paper first studies a transfer-based dependent syntax analyzer using a feed-forward neural network as a classifier. By analyzing the model, we have made meticulous parameters of the model to improve its performance. This paper proposes a dependent syntactic analysis model based on a long-term memory neural network. This model is based on the feed-forward neural network model described above and will be used as a feature extractor. After the feature extractor is pretrained, we use a long short-term memory neural network as a classifier of the transfer action, and the characteristics extracted by the syntactic analyzer as its input to train a recursive neural network classifier optimized by sentences. The classifier can not only classify the current pattern feature but also multirich information such as analysis of state history. Therefore, the model is modeled in the analysis process of the entire sentence in syntactic analysis, replacing the method of modeling independent analysis. The experimental results show that the model has achieved greater performance improvement than baseline methods.

1. Introduction

The study of grammar in computational linguistics refers to the study of specific structures and rules contained in language, such as finding the rules of the order of words in sentences and classifying words [1]. Linear laws in these languages can be expressed using methods such as Language Model and Part-of-Speech Tagging. For the nonlinear information in the sentence, we can use Syntactic Structure or Dependency Relation between words in the sentence to express. Although this analysis and expression of sentence structure may not be the ultimate goal of natural language processing problems, it is often an important step to solve the problem [2], which is used in such as search query understanding [3], Question Answering, QA [4] and Semantic Parsing and other issues have important applications. Therefore, as one of the key technologies in many natural language application tasks, Syntactic Parsing [5] has always been a hot issue in the field of natural language processing research, and it has significant research significance and application value.

Syntactic analysis is mainly divided into two types: syntactic structure parsing and dependency parsing [2]. The main purpose of syntactic structure analysis is to obtain a sentence parsing tree, so it is often referred to as full syntactic parsing, sometimes referred to as full parsing. The main purpose of dependency syntax analysis is to obtain a tree structure representation of the dependency relationship between words in a sentence, which is called a dependency tree.

In the 1940s, researchers introduced the term "neural network" in order to express biological information processing systems [6]. The simplest one, the feed-forward neural network, also known as the multilayer perceptron model, has achieved good results in many application tasks, but due to the high computational complexity of the model, training is more difficult. With the continuous improvement of computer performance, it is possible to train large-scale

and deep neural networks. As a result, the Deep Learning method has made a huge breakthrough in the research of multiple fields of machine learning. Deep learning learns from large-scale data to intricate structural representations. This learning is achieved by adjusting network parameters through error-driven optimization algorithms between different layers of artificial neural networks through back-propagation. In recent years, deep convolution network has made great breakthroughs in graphics and image processing, video and audio processing, and other fields. At the same time, recursive networks have also achieved good results in sequence data such as text and voice [7].

The recurrent neural network initially achieved good results in handwritten digit recognition [8]. The well-known word vector algorithm Word2Vec was originally obtained from the language model learned from RNN [9]. Due to the gradient disappearance defect of recurrent neural network (RNN), Long Short-Term Memory (LSTM) was proposed [10]. Due to the recent popularity of deep learning methods, LSTM has also been applied to work such as dialogue systems [11] and language models [12]. The neural network model with attention mechanism proposed recently [13] has attracted the attention of researchers. This attention mechanism has been successfully applied to machine translation [14] and text summaries [15] and has achieved certain results.

The main contributions of this paper are the following:

(i) We propose a feed-forward neural network in which the parameters propagate unidirectionally

(ii) We use a neural network model as a classifier and use the reverse propagation algorithm as the learning algorithm

(iii) We proposed a well-organized dataset to evaluate the proposed framework

Rest the paper is structured as follows: Section 2 describes related work and critically analyzes and compares the work done so far. Section 3 is about the proposed methodology describing the materials and methods adopted in this study. Section 4 is about the validity of the proposed methodology and experimentation and discussions made about the results produced. The work done is finally concluded in Section 5.

2. Related Work

Concepts such as neural networks originated in the 1940s. After the 1980s, backpropagation was successfully applied to neural networks. In 1989, the backpropagation algorithm was successfully applied to the training of a convolutional neural network. As of 2006, the graphics processing unit was used in the training of convolution neural networks. As a result, a new upsurge of neural network research has been set off. The early neural network models of the 1940s were very simple, usually only had one layer and could not be learned. It was not until the 1960s that early neural networks were used for supervised learning, and the model became slightly more complicated and had a multilayer structure. In 1979, Fukushima [16] first proposed the concepts of convolution neural networks and deep networks. After that, related pooling and other methods were proposed one after another. In 1986, the backpropagation algorithm was proposed by Rumelhart et al. [17]. It greatly promoted the development of neural network research. The second is the emergence of several public datasets. The majority of the public datasets make the neural network no longer a toy model. In the field of computer vision, there is the famous ImageNet [18]. In the field of natural language processing, there is the dataset published by Twitter 2 and the data of Weibo 3 in the Chinese field.

Bengio et al. [19] proposed the use of a recurrent neural network to build a language model. The model uses the recurrent neural network to learn a distributed representation for each word while also modeling the word sequence. This model has achieved better results in experiments than the optimal n-gram model of the same period and can use more contextual information. Bordes et al. [20] proposed a method for learning Structured Embeddings using neural networks and a knowledge base. The experimental results of this method on WordNet and Freebase show that it can embed structured information. Mikolov et al. [21] proposed continuous bag of words (CBOW): In this model, to predict the words in a sentence, the concept of word position in a sentence is used; this work also proposes a skip-gram model, which can use a word in a certain position in a sentence and predicts the words around it. Based on these two models, Mikolov et al. [21] open-sourced the tool word2vec4 to train word vectors, which has been widely used. Kim [22] introduced the convolution neural network to the sentence classification task of natural language processing. This work uses a convolution neural network with two channels to extract features from sentences and finally classify the extracted features. The experimental results show that the convolution neural network has a significant effect on the feature extraction of natural language. Similarly, Lauriola et al. [23] has critically studied and analyzed the use of deep learning in Natural Language Processing (NLP) and the models, techniques, and tools used so far have been summarized. Fathi and Shoja [24] also discuss the application of deep neural networks for natural language processing.

Tai et al. [25] proposed a tree-like long and short-term memory neural network. Because traditional recurrent neural networks are usually used to process linear sequences, and for data types with internal structures such as natural language, this linear model may lose some information. Therefore, this model uses long and short-term memory neural networks in the analysis tree and has achieved good results in sentiment analysis.

In summary, the key limitations of existing deep learning-based approaches to natural language processing include the following: deep neural network models are difficult to train because they need large amounts of data, training requires powerful, expensive video cards, lack of a uniform representation method for different forms of the data, such as text and image and the ambiguity resolution in natural language text at the word, phrase, and sentence level. Moreover, deep learning algorithms are not good at

inference and decision making, cannot directly handle symbols, they are data-hungry and not suitable with small data size, difficult to handle long-tail phenomena, black-box nature of the models makes them difficult to understand, and computational cost of the learning algorithms is high. Apart from the limitations, the good about deep neural networks include the following: efficiency in pattern recognition, data-driven approach, performance being high in many problems, little or no domain knowledge needed in system construction, the feasibility of cross-modal processing, and gradient-based learning.

3. Material and Method

In this section, we are going to discuss the recurrent neural network-based model.

3.1. Feed-Forward Neural Network. As the first proposed neural network structure, the feed-forward neural network is the simplest kind of neural network. Inside it, the parameters propagate unidirectionally from the input layer to the output layer, as shown in Figure 1 as a schematic diagram of a four-layer feed-forward neural network.

3.2. Recurrent Neural Network. Recurrent Neural networks have been a hot research field in neural network research in recent years. The reason why Recurrent Neural Networks have become a research flashpoint is that the Feed-forward Neural Network or Multilayer Perceptron cannot grip data with time series relationships well. The time recursive structure of the Recurrent Neural Network permits it to learn the time series information in the data so that it can well solve this kind of job (see Figure 2).

For each moment, the activation value of the hidden layer is calculated recursively as follows (t from 1 to N, n from 2 to N, N is the number of hidden layers):

$$\begin{aligned} h_t^1 &= \sigma\left(W_{ih^1}x_t + W_{h^1h^1}h_{t-1}^1 + b_h^1\right), \\ h_t^n &= \sigma\left(W_{ih^n}x_t + W_{h^{n-1}h^n}h_t^{n-1} + W_{h^nh^n}h_{t-1}^n + b_h^n\right). \end{aligned} \tag{1}$$

Among them, W is the parameter matrix (for example, W_{ih^n} represents the connection weight matrix from the input layer to the Nth hidden layer), b is the bias vector, and σ is the activation function.

Calculate the output sequence of the hidden layer, and you can use the following formula to calculate the output sequence:

$$\begin{aligned} \widehat{h}_t &= b_y + \sum_{n=1}^{N} W_{h^ny}h_t^n, \\ y_t &= y\left(\widehat{y}_t\right). \end{aligned} \tag{2}$$

The output vector y_t is used to estimate the probability distribution $Pr(x_{t+1}|y_t)$ of the input x_{t+1} at the next moment. The loss function $\mathscr{L}(X)$ of the entire network is expressed by the following formula:

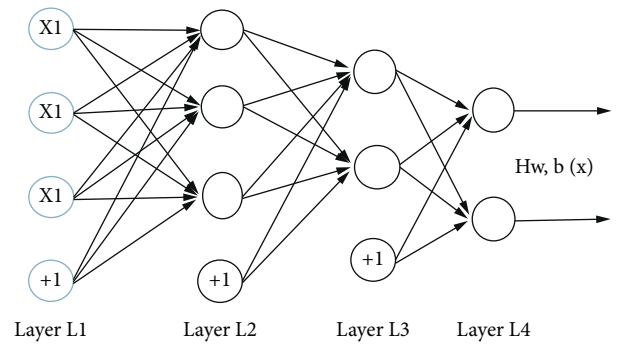

Figure 1: Schematic diagram of feed-forward neural network.

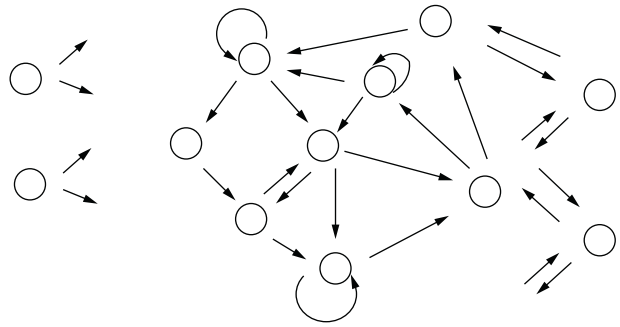

Figure 2: Schematic diagram of recurrent neural network.

$$\mathscr{L}(X) = -\sum_{t=1}^{T} \log \Pr\left(x_{t+1}|y_t\right). \tag{3}$$

Similar to the feed-forward neural network, the partial derivative of the loss function to the network parameters can be obtained by using backpropagation through time, and the gradient descent method is used to learn the parameters of the network, which is shown in Figure 3.

Due to the advantages of recurrent neural networks in time series, in recent years, many researchers in the field of natural language processing have applied recurrent neural networks to research such as machine translation, language model learning, semantic role tagging, and part-of-speech tagging and achieved good results.

3.3. Realization of Learning Algorithm and Classification Model. As an essential part of the syntactical analyzer, the role of the classification model is to predict the analytical action. The role of the learning algorithm is the parameters of the learning model from training data. In this model, we use a neural network model as a classifier, obviously use the reverse propagation algorithm as a learning algorithm. In this section, the precise implementation of the classification model will be introduced, and some details of the model learning will be described later.

The role of the embedded layer of the network is to convert the sparse representation of the feature into a dense representation. The embedding layer is divided into three parts: word embedding layer, part-of-speech embedding

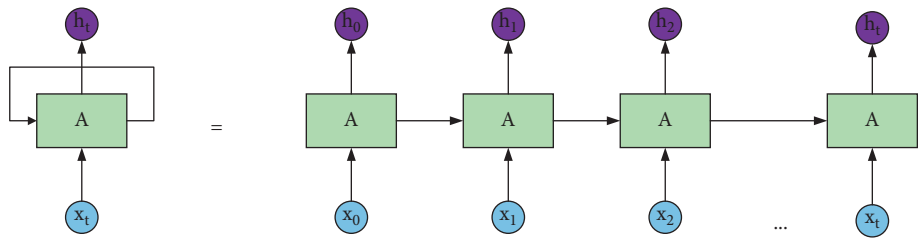

Figure 3: Schematic diagram of unfolded recurrent neural network.

layer, and dependency arc embedding layer. The three embedding layers obtain input from three different features corresponding to the input layer. It is worth noting that compared with the size of the dictionary, the value set of part of speech and dependency arc is relatively small, so the dimension of part of speech and arc embedding in the embedding layer is smaller than the dimension of word embedding. Specifically, the word feature in the analysis pattern c is mapped to d_w as a dimensional vector $e^w \in R^{dw}$, and the embedding matrix is $E^w \in R_{d_w \times N_w}$. Among them, N_w is the dictionary size.

Similarly, part-of-speech features and dependency arc features are mapped to $e^p \in R^{d_p}$ and $e^l \in R^{d_l}$ after the conversion is completed, the layer outputs 48 dense features, each of which is a real vector.

The hidden layer in the model connects the 48 output features x_h of the embedding layer end-to-end to form a feature vector and perform linear and nonlinear transformation operations on it. Specifically, the nonlinear transformation function is a cubic activation function:

$$h = \left(W_1 x_h + b_1\right)^3. \tag{4}$$

Among them, $W_1 \in R^{d_h \times d_{x_h}}$ is the parameter matrix of the hidden layer, and $d_{x_h} = 18 * d_w + 18 * d_p + 12 * d_l$, b_1 is the bias vector.

The last layer of the network is the softmax layer, whose role is to predict and analyze the probability distribution of actions:

$$\begin{aligned} o &= W_2 h + b_2, \\ p_a &= \frac{\exp\left(o_a\right)}{\sum_{a \in \tau} \exp\left(o_a\right)}. \end{aligned} \tag{5}$$

Among them, W_2 is the parameter matrix of the softmax layer, b_2 is the bias vector and τ is the set of all actions in the dependency syntax analysis system.

After obtaining the probability distribution of the analysis action predicted by the model, the loss function of the network can be calculated. The same as the general multiclassification problem, we use the cross-entropy loss function:

$$C = -\frac{1}{n} \sum_i \left[y_i \log p_i + \left(1 - y_i \log\left(1 - p_i\right)\right)\right]. \tag{6}$$

In fact, the classification task is to select a correct action from multiple analysis actions, so the loss function is simplified as follows:

$$L\left(\Theta\right) = -\sum_{i \in A} \log\left(p_i + \frac{1}{2}\lambda \left\|\Theta\right\|\right), \tag{7}$$

where A is the correct analysis sequence action set of the batch, λ is the regularization parameter, and Θ is the model parameter.

The classifier in the dependency syntax analyzer is a neural network classifier, and its learning algorithm is the same as the general neural network learning algorithm, which is a backpropagation algorithm. Using the backpropagation algorithm, the gradient of the loss function to the parameters can be obtained, and then the gradient descent method is used to update the parameters of the model.

4. Experiments and Discussion

In this section, we are going to discuss the dataset and the experimental setup and evaluate the framework.

4.1. Long and Short-Term Memory Neural Network. The recursive neural network is used to translate the input sequence to an output sequence, such as a sequence identification problem or sequence forecast problem. However, many of the actual use tasks expose difficulty in training recursive neural networks. Sequences in these issues often extent a lengthier time interval. Bengio et al., since the gradient of the recursive neural network, will ultimately "disappear," the recursive neural network that wants to learn a long-distance memory is more difficult, as shown in Figure 4.

To solve this problem, Hochreiter and SchmidHuber [10] proposed Long Short-Term Memory, LSTM. In this model, the concept of "door" is added so that the network can choose when "Forget" increasing new "memory."

As a variant of the recursive neural network, the long-term memory neural network in the design is to solve the gradient disappearance of ordinary recursive neural networks. The usual recursive neural network reads an input vector x_t from a vector sequence $(x_1, x_2, \ldots, x_n)$ and calculates a new hidden layer state h_t. However, the problem of gradient disappearance results in an ordinary recursive neural network that cannot be modeled on long-distance dependence. Long short-term memory neural networks introduced "Memory Cell" and three "Control Gate," which used to control when to choose "memory," when to choose "Forget."

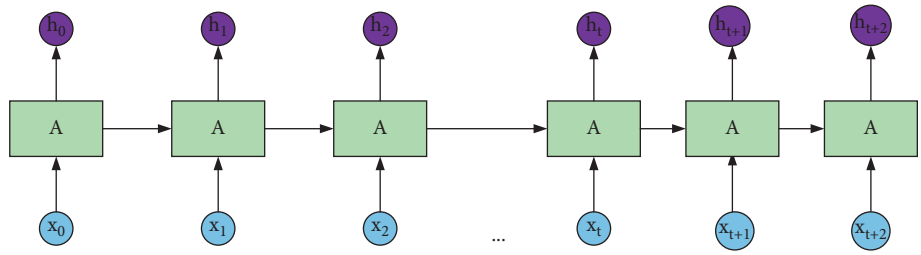

Figure 4: Ordinary recurrent neural networks cannot handle long-distance dependencies.

Specifically, the long and short-term memory neural network uses an input gate, a forget gate, and an output gate. Among them, it determines the proportion of the current input that can enter the memory unit, and the forget gate controls the proportion of the current memory that should be forgotten.

For example, at time t, the long and short-term memory neural network is updated in the following way:

At time t, given input x_t, calculate the value of input gate i_t, forget gate and candidate memory $\widetilde{C}_t$ according to the following formula:

$$\begin{aligned} i_t &= \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right), \\ \widetilde{C}_t &= \tan h\left(W_c x_t + U_c h_{t-1} + b_c\right), \\ f_t &= \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right), \end{aligned} \tag{8}$$

where σ is the component-wise logistic function and $\odot$ is the component-wise product.

At the same time, the value and output value of the new memory cell are given as follows:

$$\begin{aligned} C_t &= i_t \odot \widetilde{C}_t + f_t \odot C_{t-1}, \\ o_{t=\sigma}&\left(W_o x_t + U_o h_{t-1} + V_o C_t + b_o\right), \\ h_t &= o_t \odot \ \tan h\left(C_t\right). \end{aligned} \tag{9}$$

4.2. Experimental Data. Since batch training is required, and the analysis sequence lengths of sentences of different lengths are not the same, we have adopted a mask method for training. Even so, because the length of some sentences is too long, other sentences in the batch have been processed and have been waiting for the long sentence to appear. Therefore, to train the model more quickly, we removed sentences with more than 70 words in the training process. Such sentences have a total of 76 sentences, accounting for 0.2% of the number of sentences in the training dataset. We believe that this will not affect the effect of the final model. After removing part of the training data and verification data, the actual data used is shown in Table 1.

4.3. Evaluation Index. The analysis of phrase structure usually uses accuracy, recall, and $F1$ value for evaluation:

(1) *Accuracy*. The accuracy rate in phrase structure analysis refers to the percentage of the number of correct phrases in the analysis result to the number of phrases in the analysis result:

$$P = \frac{\text{Number of correct phrases in the analysis result}}{\text{The total number of phrases in the analysis results}}. \tag{10}$$

(2) *Recall Rate*. The accuracy rate in phrase structure analysis refers to the percentage of the number of correct phrases in the analysis result to the total number of phrases in the test set:

$$R = \frac{\text{Number of correct phrases in the analysis result}}{\text{The total number of phrases in the test set}}. \tag{11}$$

(3) $F1$ value.

$$F1 = \frac{2 \times P \times R}{P + R}. \tag{12}$$

Table 1: Statistics of the data used in this article.

Data set	A	B	C	D	E (%)
Training set	33288	33251	99.89	76	99.8
Development set	1850	1848	99.89	1	99.9
Test set	1850	1848	99.89	—	100

A is the total number of sentences; B is the number of projectable sentences; C is the percentage of projectable sentences; D is the number of sentences up to 70; E is the percentage of sentences used to projectable sentences.

4.4. Experimental Results and Analysis. In addition to the comparison with the baseline method, this topic is also compared with two other classic dependency parsers: Malt Parser and MST Parser. For Malt Parser, we used the stackproj and nivreeager options for training, which correspond to the arc-standard analysis algorithm and the arc-eager analysis algorithm, respectively. For MST Parser, we report the results in Chen and Manning (2014). The test results are shown in Table 2.

It can be seen from the table that the dependency syntax analyzer based on the long and short-term memory neural network has achieved certain effects in modeling the analysis sequence of sentences. This model has achieved 91.9% UAS accuracy and 90.5% LAS accuracy on the development set of Penn Tree Bank, which is about 0.7% improvement over the greedy neural network dependency parser of the baseline method. On the test set, our model achieved a UAS accuracy

Table 2: Test results on WSJ.

Analyzer	Development set		Test set	
	UAS	LAS	UAS	LAS
Malt: standard	90.5	88.9	89.2	87.4
Malt: eager	90.2	88.8	89.4	87.5
MST parser	91.4	88.1	90.7	87.6
Baseline method	91.2	89.9	90.1	88.3
Greedy feature extractor	91.4	89.8	90.2	88.5
This model	91.9	90.5	90.7	89.0

Table 3: Test results on WSJ23.

Model	Accuracy	Recall rate	$F1$ value	Effective output	Word count does not match	Output structure error
Single follower	0.610	0.606	0.608	934	607	99
Double follow	0.827	0.826	0.827	1347	274	19

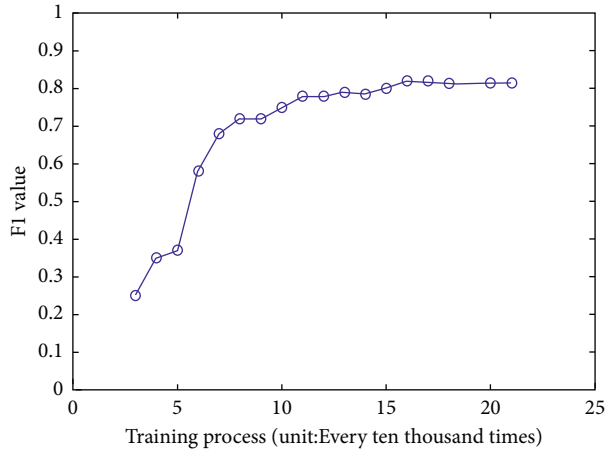

Figure 5: $F1$ value changes with the training process.

rate of 90.7% and an LAS accuracy rate of 89.0%, which is about 0.6% improvement over the greedy neural network-dependent syntax analyzer of the baseline method.

Compared with the most representative transfer-based dependency parser, Malt Parser, our method has a relative improvement of about 1.4%; compared with the famous graph model-based MST Parser, our model can obtain 0.5 on the development set. % Improvement, the UAS accuracy rate on the test set is comparable, and the LAS accuracy rate has been improved by 1.4%.

The experimental results show that, compared with the greedy feed-forward neural network, the dependency syntax analysis model based on the long and short-term memory neural network performs better. Different from the greedy model, this model uses long and short-term memory neural networks to model the entire sentence and can use historical analysis information and historical pattern information to help classify analysis actions, thereby improving the performance of the dependent syntax analyzer.

The results of testing on the Pennsylvania Tree Bank are shown in Table 3. In the testing process, this article uses the column search technique, and the corresponding beam size is 12.

It can be seen from the data in the table that the dual attention mechanism can effectively reduce the number of errors in the output results. In the effective output, the $F1$ value of the model reached 0.827, and its change with the training process is shown in Figure 5. Various errors change with the training process, as shown in Figure 6.

By linearizing the phrase structure tree in natural language, the phrase structure analysis task is transformed into a sequence-to-sequence conversion task. A simple implementation of the sequence-to-sequence model is carried out, and it is found that the end-to-end analysis still needs the rule restriction on the decoder side. To this end, we propose a dual attention mechanism model, that is, a sequence-to-sequence model that introduces attention mechanisms at the input and output at the same time. Experiments show that after the introduction of the dual attention mechanism

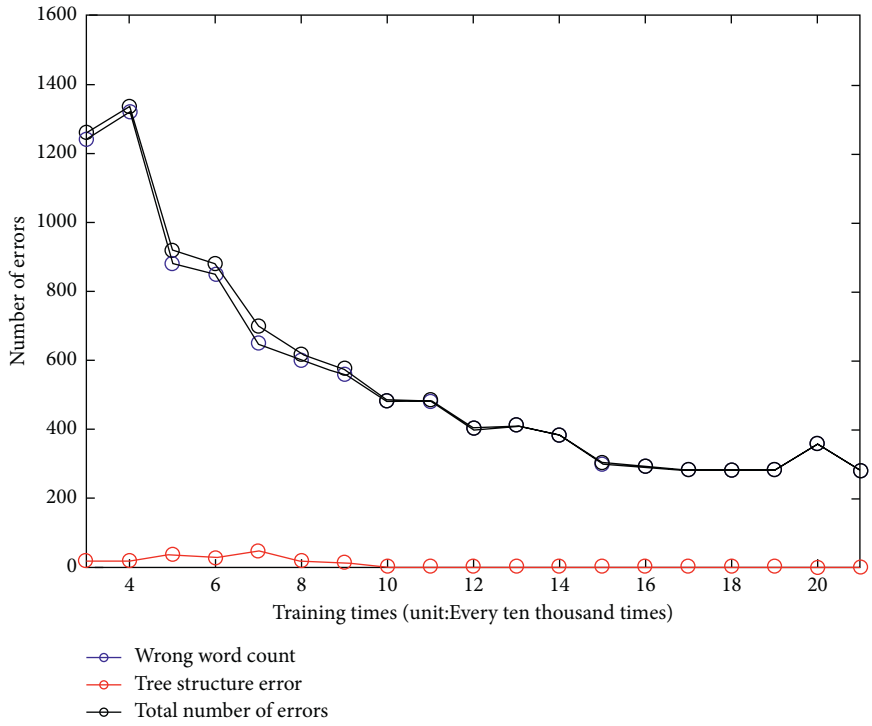

Figure 6: Variety of errors with the training process.

Table 4: Symbols used in the papers.

Symbol/abbreviation	Abbreviation definition and meaning
RNN	Recurrent neural network
LSTM	Long short-term memory
MST parser	Maximum spanning tree parser
CBOW	Continuous bag of words

model, the performance of the model on the test set is greatly improved in Table 4.

5. Conclusions

Syntactic analysis is an indispensable part of tasks such as question answering systems, search string comprehension, semantic analysis, and knowledge base construction. This paper studies a neural network model of dependency syntactic analysis based on transfer learning. This model uses a feed-forward neural network as the classifier in the dependency syntax analyzer and adjusts its parameters by analyzing the model to achieve better results. The experimental results show that after improvement, the effect of the model is increased by 0.1 to 0.2 percentage points. We propose a dependency syntax analysis model based on long and short-term memory neural networks. This model is based on the neural network model and used as a feature extractor. Specifically, the model is based on the characteristics of the long and short-term memory neural network and uses it to memorize the analysis state and analysis history in the transfer-based dependency syntactic analysis process so that the model can capture and utilize more historical information. In addition, the model models the analysis process of the entire sentence in the dependent syntax analysis and improves the greedy model to model the independent analysis state. The experimental results show that compared with the baseline method, the model obtains an improvement of 0.6 to 0.7 percentage points.

Through the work experience and error analysis, we can further study the dependency syntax analysis model based on the long and short-term memory neural network, and we found that the attention mechanism can be introduced into the model.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] C. D. Manning and H. Schutze, *Foundations of statistical natural language processing*, MIT Press, Cambridge, MA, USA, 1999.

[2] C. Q. Zong, *Statistical Natural Language Processing*, Tsinghua University Press, Beijing, China, 2008.

[3] J. Liu, P. Pasupat, and Y. Wang, "Query Understanding Enhanced by Hierarchical Parsing structures," in *Proceedings of the Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pp. 72–77, IEEE, Okinawa, Japan, December, 2017.

[4] K. Tymoshenko and A. Moschitti, "Assessing the impact of syntactic and semantic structures for answer passages reranking," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, Melbourne, Australia, October 2015.

[5] W. Monroe and Y. Wang, *Dependency Parsing Features for Semantic Parsing*, Stanford University, Stanford, CA, USA, 2014.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[8] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[9] T. Mikolov, W. t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751, Association for Computational Linguistics, Atlanta, Georgia, June 2013.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pp. 3104–3112, Curran Associates, Inc., Montreal, Canada, December 2014.

[12] M. Sundermeyer, R. Schlüter, and H. Ney, *LSTM Neural Networks for Language Modeling*ISCA Archive, Portland, OR, USA, 2012.

[13] V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pp. 2204–2212, Curran Associates, Inc., Montreal, Canada, December 2014.

[14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, https://arxiv.org/abs/1409.0473.

[15] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," 2015, https://arxiv.org/abs/1509.00685.

[16] K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[17] D. E. Rumelhart and J. L. M Cc Lelland, "Parallel distributed processing: explorations in the microstructure of cognition," *Language*, vol. 22, no. 4, pp. 98–108, 1986.

[18] O. Russakovsky, J. Deng, H. Su et al., "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[19] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[20] A. Bordes, J. Weston, and R. Collobert, "Learning structured embeddings of knowledge bases," in *Proceedings of the Conference on Artificial Intelligence*, AAAI, San Francisco, CA, USA, 2011.

[21] T. Mikolov, K. Chen, and G. Corrado, "Efficient estimation of word representations in vector space," 2013, https://arxiv.org/abs/1301.3781.

[22] Y. Kim, "Convolutional neural networks for sentence classification," 2014, https://arxiv.org/abs/1408.5882.

[23] I. Lauriola, A. Lavelli, and F. Aiolli, "An introduction to deep learning in natural language processing: models, techniques, and tools," *Neurocomputing*, vol. 470, 2021 Jul 22.

[24] E. Fathi and B. M. Shoja, "Deep neural networks for natural language processing," *Handbook of Statistics*, vol. 38, pp. 229–316, 2018 Jan 1.

[25] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," 2015, https://arxiv.org/abs/1503.00075.