Coreference disambiguation based on two-layer label dependence analysis

Hongfei Xu*[,a], Yingna Li[b]
[a] Faculty of Information Engineering and Automation, Kunming University of Science and Technology, 727 Jingming South Rd.,Chenggong District, Kunming, China 650500; [b] Yunnan Key Laboratory of Computer Technology Applications, 727 Jingming South Rd.,Chenggong District, Kunming, China 650500.

ABSTRACT

A two-layer labeling-based dependency analysis model is proposed to introduce the location features of words for the co-reference disambiguation of entities. Firstly, two layers of labels are used for labeling, the second layer labels the location information of words, the features of sentences are learned using a bidirectional long short-term memory network, the dependency syntax analysis based on deep graph decoding obtains the dependency tree of sentences, and the two layers of labels are fused to improve the performance and accuracy of coreference disambiguation. Experiments are conducted on the text dataset of power security entity relationship extraction, and the results show that the two-layer labeled dependency analysis has an improved effect on co-finger disambiguation, which verifies the effectiveness of the model for cofinger disambiguation on the experimental dataset.

Keywords: Coreference resolution, Dependency Analysis, Double-layer label, BiLSTM

1. INTRODUCTION

Denotation is a linguistic structure that exists in natural language expressions and is divided into various types: noun denotation, pronoun denotation, inter-phrase denotation, etc. Denotation is the value of the interpretive association of one linguistic component to another, i.e., the description of one component to another. The denotation disambiguation is an important task in the field of natural language processing, and performing the denotation disambiguation task directly enhances the effect of entity relationship extraction and has an impact on the subsequent work such as knowledge mapping. Coreferential disambiguation is divided into Coreference and Anaphora, in which the prior and the illuminant point to the same entity, and the prior and the illuminant are equivalently positioned in the sentence and have no close contextual semantic connection; in which the prior and the illuminant are semantically connected in the sentence, and the prior precedes the illuminant and points to different entity in different contexts. The current research on denotative disambiguation mainly focuses on coreferential disambiguation, and this paper also focuses on coreferential disambiguation.

Dependency analysis can be used in the study of coreference disambiguation by generating a dependency tree to represent the sentence structure explicitly through the analysis of dependency relations. Dependency syntax is a method of identifying word-to-word dependencies in a sentence. The basis of dependency syntax is that the words in a sentence are interconnected, and the main purpose is to analyze the connection of words in the sentence structure. Usually, the two words of coreference have the same dependency structure and have equivalent relationship in the sentence structure, so it is feasible to generate the dependency tree by dependency syntactic analysis first and then perform coreference disambiguation by judging the candidate disambiguation objects. In this paper, we improve the dependency analysis based on deep graph decoding by using a two-layer labeling method to label the sentences and introduce the position information of words in the sentences, and test it on the dataset, and achieve good results that outperform other commonly used models by comparing them with each other.

*xu19991110@163.com

2. RELATED WORK

2.1 Coreference resolution

There are also three types of research on coreference disambiguation, rule-based research methods, machine learning-based research methods, and deep learning-based research methods. Hobbs' plain algorithm adds semantic knowledge to the rules, which has a great improvement effect on the model[1]. However, rule-based methods have some general problems, which are complicated to implement, prone to errors, poor generalization ability, and the model effect is limited by the formulation of rules. With the rise of machine learning algorithms, scholars have tried to use machine learning methods for the task of coreferential disambiguation, using machine learning methods to predict output information by using partial input information. soon et al. used partial semantic features as input to the model for learning to complete coreferential disambiguation[2]; Luo et al. applied Bell trees to obtain referential features in the representation of phrases[3], but the machine learning methods also suffer from The quality of the input features suffers from the dependence on the set threshold, among other problems. Thus, researchers have applied the emerging deep learning techniques in coreference disambiguation by applying linear transformations to the intrinsic logic in the data using deep learning algorithms to abstract more implicit information, and then fitting and optimizing them to capture more over-dimensional features such as utterance information, semantic information, structural information, and dependency information in the dataset, reducing the dependence on manual input features, and also having strong generalization ability to make the models to be used in different tasks. Neural network models such as CNN (Convolutional Neural Network) and LSTM (Long Short-Term Memory Neural Network) have superb learning and characterization capabilities, which provide a good basis for the study of co-referential disambiguation. Wiseman et al. used RNN (Recurrent Neural Network) to learn clustering of entity pointing object words to solve the pronoun referential disambiguation problem[4]; Clark et al. used reinforcement learning to construct a co-referential disambiguation system[5]. Lee et al. proposed an end-to-end neural network model for denotational disambiguation without using artificial feature inputs, incorporating attention mechanisms and long- and short-term memory networks to achieve better results[6].

2.2 Dependency analysis

Dependency analysis is divided into transfer-decoded dependency analysis and graph-decoded dependency analysis, and graph-decoded dependency analysis is an important method of dependency analysis, which can find the global optimal dependency syntax tree. The analysis algorithm finds the optimal dependency tree based on the score of each dependency tree. Deep graph decoding dependency analysis is the use of deep learning techniques for syntactic analysis. deep learning applications for syntactic analysis are relatively recent. Collobert et al. used deep learning for discriminative syntactic analysis, decomposed the analysis tree of the sentence using a deep convolutional model, and then analyzed each layer of the tree[7]; Durrett et al. combined field random fields and neural network models, and then used this method for syntactic analysis[8]; Zheng et al. proposed convolutional neural network for syntactic analysis, which was based on word for sentence analysis, and achieved good results[9]; Chen et al. proposed two recurrent neural networks and used them on transfer-based dependent syntactic analysis for dense feature learning to improve the performance of the model[10].

3. DEPENDENCY ANALYSIS BASED ON DOUBLE-LABELING

3.1 Double-layer label

Traditional coreference elimination uses single-layer tags or does not use any tags for syntactic analysis and semantic analysis, and eliminates misreference and misassociation through simple syntactic information and partial semantic information, which can be applied to Chinese text information extraction in general fields. Moreover, the presence of a large number of long continuous entities in the power safety text will also have a direct impact on the extraction results.

In this paper, we use two layers of tags to label the relationship of sentence entities. The first layer of tags uses sentence component tags to label the components in a sentence, which focuses on the component information of the sentence; the second layer of tags labels the position of words, which focuses on the position information of words in the sentence.

For a sentence, tag one labels the entity and relation categories of the sentence, and tag two labels the position information of the entity relation in the sentence. For the second layer of labels, two labeled words are considered as adjacent in the position of the sentence when there is no valid adjacent label in the middle of the two valid adjacent labels.

3.2 Dependency analysis

Dependency syntactic analysis is different from the traditional compositional syntactic analysis in that the focus of dependency legal analysis is not on the components of the phrase, but on the direct use of the words themselves and the binary dependencies between them, enabling a more direct analysis of the subject-predicate and other components of the sentence, and the results obtained using dependency syntactic analysis show the relationships between words more directly.

Syntactic relations are the basis of dependency relations, which are binary relations between words, and each dependency relation is composed of the central word and its dependencies, and this kind of is good for distinguishing between long continuous entities and non-continuous entities with unknown referents that appear in the entity relations extracted from the power text.

Dependency syntax considers the verb in the predicate as the center of a sentence, and the words of other components are directly or indirectly related to the central word, and dependency syntax analysis analyzes the sentence into a dependency syntax tree, which describes the dependency relationship between each word and also points out the syntactic collocation relationship of the words associated with the semantics

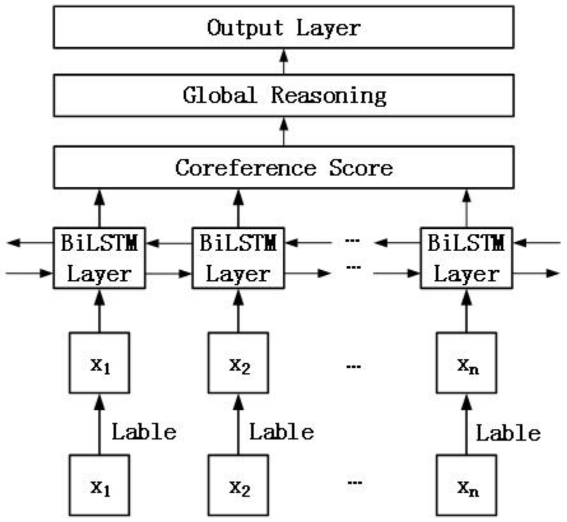

Figure 1. Model structure diagram.

In this paper, we use the set of candidate entity relations to be disambiguated by entity relation extraction, and then use the graph decoding dependency analysis based on the deep learning model to optimize and reason about the set of entity relations to be disambiguated by using the position information of the fused entity relations in the sentence, and then use the scoring function to rank and determine the disambiguation result.

The model processes the sentences with extracted entity relations with the input sequence x=(w_0,w_1,w_2,...,w_n) (w0 is the added word root ROOT), $\tau(x)$ represents the set of all possible dependency trees of the sentence, and the process of dependency analysis of sentences is shown in Equation 1.

$$\hat{y} = \underset{y \in \tau(x)}{\operatorname{argmax}} Score(x, y) \tag{1}$$

The process of dependency analysis is to find the optimal dependency tree. *Sore*(*x,y*) is the scoring function for the dependency tree, which is usually obtained by decomposing the subgraphs *c* of the dependency tree and scoring them independently, and summing the independent scores.

$$Score(x, y) = \sum_{c \in y} Score(x, c) \tag{2}$$

The scoring model uses a multilayer perceptron (MLP), and the graph decomposition method uses a first-order decomposition strategy to decompose the dependency tree into dependency edges.

$$Score(w_h, w_m, \theta) = \mathrm{MLP}(\phi(w_h, w_m); \theta) \tag{3}$$

This *Φ* represents the characteristics of dependent edges, using a low-dimensional dense embedding representation, and *θ* is a network parameter model, the specific process is shown in Equation 4-8..

$$Score(w_h, w_m, \theta) = W_o^d h + b_o^d,\ \ d \in \{0,1\} \tag{4}$$

$$h = g(W_h^d a + b_h^d),\ \ d \in \{0,1\} \tag{5}$$

$$g(l) = \tanh(l^3 + l) \tag{6}$$

$$a = \phi(w_h, w_m) \tag{7}$$

$$\theta = \{W_h^d, W_o^d, b_h^d, b_o^d\},\ \ d \in \{0,1\} \tag{8}$$

The W_h^d and b_h^d is the hidden layer parameter,the W_o^d and b_o^d is the output layer parameter, and the value of *d* is 0 or 1. When the value is 0, it means the left-dependent edge parameter, and when the value is 1, it is the right-dependent edge parameter. The computed value is a multidimensional vector, and the number of dimensions depends on the number of dependent labels. The value of the *i* dimension of the vector is the score of the corresponding dependent edge lb_i, as shown in Equation 9.

$$Score(w_h, w_m, lb_i) = Score(w_h, w_m, \theta)[i] \tag{9}$$

The long short-term memory network is introduced to learn the features using a bidirectional long short-term memory network, and the structure of the long short-term memory network is shown in Figure 2.

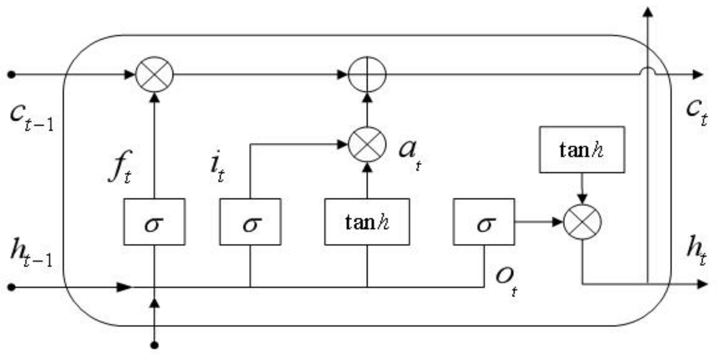

Figure 2. Long short-term memory network model.

For a sequence of vectors $X=(x_1,x_2,...,x_n)$, each moment corresponding input has a corresponding output vector $h_i(i=1,2,...,n)$, recursively learning the sequence information as shown in Equation 10.

$$\begin{aligned} h_i &= \mathrm{LSTM}(h_{i-1}, x_i, \theta) \\ &= \mathrm{LSTM}(\mathrm{LSTM}(h_{i-2}, x_{i-1}, \theta), x_i, \theta) \\ &= \cdots \\ &= \mathrm{LSTM}(\mathrm{LSTM}(\cdots(\mathrm{LSTM}(h_0, x_1, \theta), \cdots) x_{i-1}, \theta) x_i, \theta) \end{aligned} \tag{10}$$

For the embedding vectors e_{w_i} and e_{t_i} of words and word categories in the sentences are spliced and transformed into input vectors by a single-layer ReLU network, and then the long-short time encoding vectors in both forward and reverse directions are summed up[11], and the output of the computationally shared long short-term memory model is obtained, and the output of the bidirectional long short-term memory network is represented by v_i, as shown in Equations 11-12.

$$x_i = \mathrm{ReLU}(W[e_{w_i}; e_{t_i}] + b) \tag{11}$$

$$v_i = \mathrm{BiLSTM}(x_i) = h_i^F + h_i^B \tag{12}$$

By learning the features, we are able to use not only the information in the dependency library, but also the contextual information of the sentence to derive the dependency of entity relations in the sentence. When two entities have the same dependency relationship in the sentence adjacent to each other, it is determined that there is a coreference relationship between the two entities and the merging between entities is performed.

4. EXPERIMENTAL RESULTS AND ANALYSIS

In this paper, we use our own power security text dataset with extracted entity relations for model experiments. The dataset sentences contain <entity-relationship-entity> triad, and the text is includes direct access to word type information and word location information, it introduced information about the position of the sentence. 10,000 data items are used, and the training set and test set are divided according to 4:1.

The evaluation metrics for dependency analysis are unmarked dependency correct rate (UAS), marked dependency correct rate (LAS), dependency correct rate (DA), root correct rate (RA), and perfect match rate (CM). In this paper, we use dependency syntactic analysis for co-reference elimination, mainly for the correctness of the dependency trees that the model can analyze, so the most commonly used UAS and LAS as the evaluation index of the model can evaluate the effectiveness of the model in generating dependency trees.

UAS (unmarked dependency correctness rate): the percentage of correct dependency core words found in the test set to the total number of words.

LAS (marked dependency correctness rate): the percentage of words with correct dependencies found in the test set to the total number of words.

Since subgraph decomposition affects the efficiency of the core scoring algorithm for graph decoding dependency analysis, the dependency analysis experiments were compared according to the models of different methods of subgraph decomposition, and the results are shown in Table 1.

Table 1. Dependence analysis comparison experimental results.

Models	**Method**	**UAS**	**LAS**
MSTParser	Traditional low order decomposition	77.61	76.49
Fourth-order dependency-dependency model[12]	Traditional higher order decomposition	79.56	——
Multi-layer perceptron dependency model[13]	The phrase embedded indicates	78.99	77.63
Our Model	Based on BiLSTM	**80.21**	**79.59**

MSTParser is a dependency syntactic analysis tool, non-projective dependency parser that searches for maximum spanning trees on directed graphs, a traditional model for dependency analysis using traditional first-order subgraph decomposition; the fourth-order dependency model uses a fourth-order decomposition model with subgraphs consisting of up to four dependency edges; the multilayer perceptron dependency model was the first to introduce multilayer perceptrons into the model, eliminating the setting of combinatorial The multilayer perceptron dependency model was the first to introduce multilayer perceptrons into the model, eliminating the setting of combinatorial features and using atomic features, using segments as the basic atomic features into the model. Dependency analysis using the deep graph decoding model achieves the best results on the power security extraction text dataset, compared with other deep dependency analysis models and traditional dependency analysis models.

The evaluation metrics of co-dependency decoding still use three metrics, Precision (P), Recall (R), and F1 value. The accuracy rate reflects the accuracy of the model for co-index elimination, i.e. the accuracy rate; the recall rate reflects the completeness of the model for co-index elimination, i.e. the completeness rate; and the F1 value is based on the combined accuracy and completeness rate.

The experiments were conducted before and after using the labels, and also compared with the use of word type labels only, and the results are shown in Table 2.

Table 2. Results of coreference resolution experiments.

Method	Precision	Recall	F1
Unused labels	59.2	58.6	58.9
Use the word type tag	65.9	75.0	70.2
Double-layer label	**70.3**	**76.2**	**73.1**

From the experimental results, it can be seen that the accuracy, recall and F1 values of the co-reference elimination are improved when the model uses labels, with the recall improving the most. After the model uses word type labels and location labels, there is a significant improvement in accuracy compared to the model that uses only word type labels. The model shows the effectiveness of co-reference disambiguation using a deep graph decoding dependency analysis model based on double-layer labels on the power security extraction text dataset.

5. SUMMARY

In this paper, the traditional labeling method is improved by using double labeling technique for text labeling, and the location label of words is introduced, which is complementary to the feature learning of common finger elimination and can improve the effect of common finger elimination in practical applications. The model uses a deep graph decoding model for dependency analysis, which can make up for the lack of model expression capability compared with traditional dependency analysis methods, and the application of bi-directional long short-term memory networks can also simplify the input of manual features and optimize the limitations existing in traditional methods, with practical effects and significance.

REFERENCES

[1] Hobbs, J. R., Resolving pronoun references[J]. Journal of Lingua, 44(4):311-338 (1978).

[2] Soon, W. M., Ng, H. T., Lim, D. C. Y., A Machine Learning Approach to Coreference Resolution o f Noun Phrases[J]. Computational Linguistics, 27(4):521-544 (2001).

[3] Luo, X., Ittycheriah, A., Jing, H., et al., A Mention-Synchronous Coreference Resolution Algorithm Based on the Bell Tree[C]// Meeting of the Association for Computational Linguistics. DBLP (2004).

[4] Wiseman, S., Rush, A. M., Shieber, S. M., Learning global features for coreference resolution[J]. a r Xiv preprint arXiv:1604.03035 (2016).

[5] Clark, K., Manning, C. D., Entity-Centric Coreference Resolution with Model Stacking[C]// Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (2015).

[6] Lee, K., He, L., Lewis, M., et al., End-to-end Neural Coreference Resolution: Association for Computational Linguistics, 10.18653/V1/D17-1018[P] (2017).

[7] Collobert, R., Deep learning for efficient discriminative parsing. In Proceedings of the fourteenth international conference on artificial intelligence and statistics (pp. 224-232). JMLR Workshop and Conference Proceedings. (2011).

[8] Durrett, G., Klein, D., Neural crf parsing[J].arXiv preprint arXiv:1507.03641 (2015).

[9] Zheng, X., Peng, H., Chen, Y., et al., Character-based parsing with convolutional neural network. In Proceedings of the 24th International Conference on Artificial Intelli gence.Austin,Texas,USA:AAAI Press, pp.1054–1060 (2015).

[10] Chen, X., Zhou, Y., Zhu, C., Qiu, X., Huang, X. J., Transition-based dependency parsing using two heterogeneous gated recursive neural networks. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1879-1889 (2015).

[11] Wang, W., Chang, B., Graph-based dependency parsing with bidirectional LSTM. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2306-2315 (2016).

[12] Ma, X., Zhao, H., Fourth-order dependency parsing. In Proceedings of COLING 2012: posters, pp. 785-796 (2012).

[13] Pei, W., Ge, T., Chang, B., An effective neural network model for graph-based dependency parsing. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 313-322 (2015).