Learning Entity Representation for Named Entity Disambiguation

Rui Cai, Houfeng Wang, and Junhao Zhang

Key Laboratory of Computational Linguistics (Peking University),
Ministry of Education, China
cairui@pku.edu.cn, wanghf@pku.edu.cn, zhang.junhao@pku.edu.cn.

Abstract. In this paper we present a novel disambiguation model, based on neural networks. Most existing studies focus on designing effective man-made features and complicated similarity measures to obtain better disambiguation performance. Instead, our method learns distributed representation of entity to measure similarity without man-made features. Entity representation consists of context document representation and category representation. Document representation of an entity is learned based on deep neural network(DNN), and is directly optimized for a given similarity measure. Convolutional neural network(CNN) is employed to obtain category representation, and shares deep layers with DNN. Both models are trained jointly using massive documents collected from Baike[1]. Experiment results show that our method achieves a good performance on two datasets without any manually designed features.

Keywords: Entity Disambiguation, Entity Representation, Deep Neural Network

1 Introduction

Named entity disambiguation(NED) is an important research task in information extraction(IE) and natural language processing(NLP). Given a set of documents and a knowledge base, it's needed to determine whether an identical mention occurring in different documents refers to the same thing in real world, and link them to knowledge base entries.

At the heart of NED is the notion of similarity. The majority of exiting approaches follow [2] and employ different ways to measure the similarity of mention and candidate entities. To obtain better disambiguation performance, most studies focus on designing more effective features and similarity measures. For example, [1] propose a learning to rank algorithm for entity linking which utilizes hand-crafted features.

To avoiding excessive task specific feature engineering, neural network architecture have been applied to various natural language processing tasks[10] and can learn representations useful to specific task. Existing neural networks for paragraph embedding[19] and sentence modelling[12] ignore the situation

[1] http://baike.baidu.com/

of ambiguous mention and performs well in many tasks, but they are not suitable for disambiguation task. In the present work, we propose a model learning document representation(**LDR**) of the context of ambiguous mention based on DNN. Context representation are optimized for a given similarity measure, and allow us to compare context and entity at some higher level. Features are learned leveraging large scale annotation of Baike, without any manual designed efforts.

Convolutional neural network(CNN) has been applied to solve sentence classification([11]) and achieve a good performance. A CNN classifying entity is trained for learning category representation(**LCR**). The output of CNN is a vector of probabilities, each element of which corresponds to a specific category and can be regarded as a distributed representation of category. Inspired by multi-task learning[10], **LDR** and **LCR** share deep layers and achieve better performance when they are trained jointly. The final similarity score is the combination of document similarity and category similarity, utilizing document representation and category representation separately.

Compared with English NED, the Chinese NED is even more challenging. Many common words and geographic names can often be used as a person name in Chinese. The lack of morphology information (such as the capital word for named entity) in Chinese also increases the difficulty [22]. Our method is applied to the Chinese named entity recognition and disambiguation(NERD) Task in the CIPS-SIGHAN joint conference on Chinese language processing(CLP2012). This task focuses on NERD in Chinese language, and presents some challenges unique to Chinese introduced previously.

Experiment are conducted at two parts. We test our method on a Chinese dataset for personal name disambiguation. Then we apply our method on data of CLP-2012 Chinese Named Entity Recognition and Disambiguation task.

2 Learning Entity Representation

Given an ambiguous mention string m with its context document$d$, a list of candidate entities $e_i \in C(m)$ in knowledge base, for each candidate entity we compute a score $sim(d, e_i)$ indicating how likely m refers to e_i. The linking result is $e = argmax_{e_i} sim(d, e_i)$.

Our algorithm consists of two parts. Document representation learning(**LDR**) is based on DNN and the network weights are fined-tuned to optimize the document similarity score $sim_{doc}(d, e)$. Category representation learning(**LCR**) based on CNN is trained by maximizing a likelihood over the training data. The final similarity score $sim(d, e)$ combines document similarity $sim_{doc}(d, e)$ and category similarity $sim_{cat}(d, e)$.

2.1 Learning Representation of Document

The context document is a variable length sequence of words, and there are two common approaches to obtain fixed length input: a window approach, and a convolutional approach. Window approach assumes the meaning of a mention

depends mainly on its neighboring words. Convolutional approach is a simple and effective methods for compositionality learning in vector based semantics(Mitchell and Lapata[14]). Entity disambiguation task requires the consideration of the whole context, the natural choice becomes convolutional approach.

Collobert[10] introduce a model to learning word embedding based on the syntactic contexts of words. Formally, in entity disambiguation task, we have a word dictionary D of size $|D|$. Each word w in D is represented as a d-dimensional vector $Embed(w)$. For a word $w \in D$, the corresponding word embedding $Embed(w) \in R^d$ is retrieved by lookup table layer shown in Figure 1:

$$Embed(w) = LT_w.w \tag{1}$$

Where $LT_w \in R^{d \times |D|}$ is a matrix of parameters to be learnt, $Embed(w) \in R^d$ is the w^{th} column of LT_w. Given any sequence of n words $w_{1:n}$ in D, lookup table layer applies the same operation for each word in sequence, and the output is represented as:

$$Embed(w_{1:n}) = Embed(w_1) \oplus Embed(w_2) \oplus ... \oplus Embed(w_n) \tag{2}$$

where $\oplus$ is the concatenation operator.The concatenation of word embedding

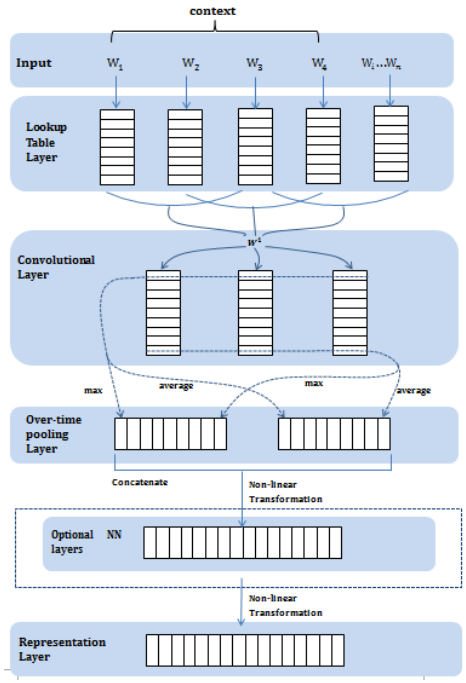

Fig. 1. Learning continuous representation of context

can be fed to further neural network layers.

A convolution operation involves a filer $\mathbf{w} \in \mathbf{R}^{hd}$, which is applied to a window of h words to produce a new feature. For example, a feature x_i is generated from a windows of words $Embed(w_{i:i+h-1})$ by

$$x_i = f(\mathbf{w} \cdot Embed(w_{i:i+h-1}) + b) \tag{3}$$

Here $b \in \mathbf{R}$ is a bias term and f is a non-linear function such as the hyperbolic tangent. This filter is applied to each possible in the sequence to produce a feature map

$$\mathbf{x} = [x_1, x_2, ..., x_{n-h+1}] \tag{4}$$

with $\mathbf{x} \in \mathbf{R}^{n-h+1}$. Our model uses multiple filters to obtain multiple global features. We then apply an average and a max pooling operation over the feature map, and take the average value $\bar{x} = mean\{\mathbf{x}\}$ and the maximum value $\hat{x} = max\{\mathbf{x}\}$. Local feature vectors extracted by the convolutional layers are combined to obtain a global feature vector by pooling operation, with fixed-size independent of context length. Global feature vector are fed to stacked full connected hidden layers, and the number of hidden layers depends on specific disambiguation task. As shown in Figure 1, a representation layer is stacked on hidden layers, the output of which is the continuous representation of context document.

We optimize the document representation towards the similarity score $sim(d, e)$, with large annotation as supervision. The similarity score of (d, e) pair is defined as the dot product of $f(d)$ and $f(e)$ shown in Figure 2:

$$sim(d, e) = \text{Dot}(f(d), f(e)) \tag{5}$$

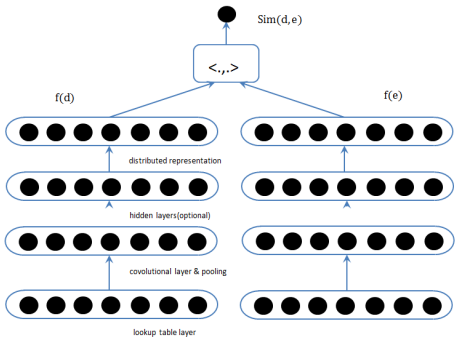

Fig. 2. Network structure of training stage

Our goal is to rank the correct definition higher than the rest candidates relative to the context of mention. For each training instance (t, d), we contrast

it with one of its negative candidate pair (t, d'). This gives the pairwise ranking criterion:

$$\mathcal{L}(d, e) = \{0, 1 - sim(t, d) + sim(t, d')\} \tag{6}$$

Alternatively, we can contrast with all its candidate pairs (t, d_i). That is, we raise the similarity score of correct pair $sim(t, d)$ and penalize all the rest $sim(t, d_i)$. The loss function is defined as negative log of softmax function:

$$\mathcal{L}(t, d) = -\log \frac{\exp sim(t, d)}{\sum_{d_i \in C(m)} \exp sim(t, d_i)} \tag{7}$$

Finally, we seek to minimize the following training objective across all training instances. We find that penalizing more negative examples, convergence speed can be greatly accelerated. In our experiment, the $softmax$ loss function consistently outperforms pairwise loss function, which is taken as a default setting.

For regularization, we employ dropout on the penultimate layer(Hinton et al.[18]). Dropout prevents co-adaptation of hidden units by randomly dropping out a proportion p of the hidden units during foward-backpropagation. That is, given the penultimate layer $\mathbf{z}$, for output unit y in forward propagation, dropout uses

$$y = \mathbf{w} \cdot (\mathbf{z} \circ \mathbf{r}) + b \tag{8}$$

where $\circ$ is the element-wise multiplication operator and $\mathbf{r} \in \mathbf{R}^m$ is a 'masking' vector of Bernoulli random variables with probability p of being 1. Gradients are backpropagated only through the unmasked units. At test time, the learned weight vectors are scaled by p such that $\hat{\mathbf{w}} = p\mathbf{w}$, and $\hat{\mathbf{w}}$ is used to score unseen context.

2.2 Learning Category Representation

The category informations of ambiguous mention m are helpful to disambiguation task. Take the personal name disambiguation for example, the document containing ambiguous name and the correct candidate entity are used to share a same category such as *politician*. Our approach for learning category representation(**LCR**) is inspired by the method for sentence classification. The model architecture is a slight variant of the CNN architecture of (Collobert et al.[10]). The output of softmax layer is a multi-dimensional category vector $\mathbf{c}$, and $\mathbf{c}_i$ is the probability of being the i-th category.

Training dataset is denoted as $\mathcal{D}$, and each training instance is a pair of context document and category denoted as (d, c). In the case of multi-class classification, it is very common to use the negative log-likelihood as the loss, which is equivalent to maximizing the likelihood of the dataset $\mathcal{D}$:

$$\mathcal{L} = -\log \prod_{i=0}^{|D|} P(C = c^{(i)} | d^{(i)}) \tag{9}$$

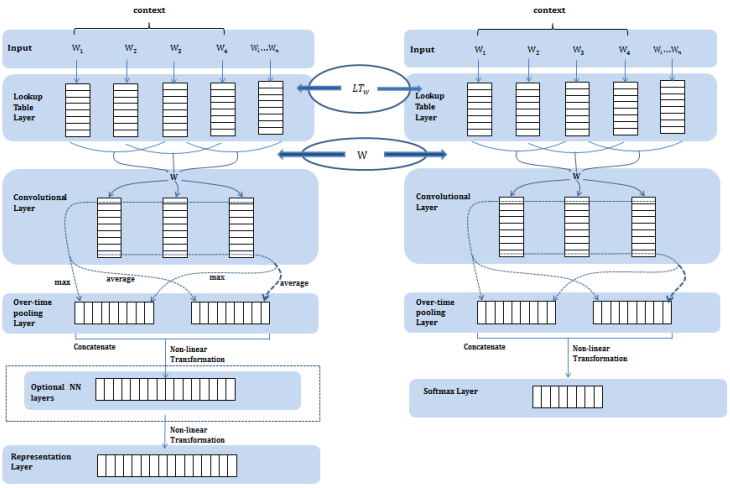

Fig. 3. LCR in the right section shares lookup table layer and convolutional layer with **LDR**

The CNN architecture automatically learns features for classification task in the deep layers of its architecture. Learning continuous representation of the context and its category are two highly related task, and it make sense that features useful for one task might be useful for another one. As shown in Figure 3, the lookup table layers and convolutional layers in two models share the same architecture. Our work is inspired by the multitask learning, which is a procedure of learning several tasks at the same time with the aim of mutual benefit(Collobert et al.[10]). Training is achieved in a stochastic manner by looping over two tasks.

The category similarity score of (d, e) pair is defined as the dot product of two category representations.The final similarity score $sim(d, e)$ becomes the combination of context similarity $sim_{doc}(d, e)$ and category similarity $sim_{cat}(d, e)$:

$$sim(d, e) = sim_{doc}(d, e) + \lambda sim_{cat}(d, e) \tag{10}$$

The value of λ is chosen depends on the performance of experiments.

3 Experiments

3.1 Disambiguation of Entities in Web Pages

Training settings: For the lookup table layer, word vectors are initialized using the publicly available word2vec vectors that were trained on gigaword. The vectors have dimensionality of 200 and were trained using the continuous bag-of-words architecture. For the convolutional layer, We use filter window of 3 with 200 feature maps. Representation layer has 200 units with sigmoid activation. We use learning rate of 1e-3 and dropout rate of 0.5. The mini-batch size is set to 20.

Dataset : We constructed a dataset of web pages obtained from Baike, where every name string m corresponds to a describing page which lists multiple web pages in the reference. For each name, we create a collection of six web pages: given a describing page in Baike, only one page corresponds to the same entity with it, whereas the rest of pages are not. Totally 113,681 names are picked out and are split into three set: 80% for training, 10% for validation, and 10% for testing.

We experiment with several variants of the model:

LDR: The model introduced in section 3.1 with pre-trained vectors from word2vec and applies two pooling operations on convolutional layer. Rank candidate entities with document similarity.

LDR-rand: Same as above but word vectors are randomly initialized and then modified during training.

LDR-max-pooling: The model only applies max-pooling operation on convolution layer.

LDR-average-pooling: The model only applies average-pooling operation on convolution layer.

LDR+LCR-joint: The model introduced in section 3.2. Learning representations of document and category of entities jointly. The final similarity score becomes the combination of document similarity and category similarity.

LDR+LCR: Similar with above but LDR and LCR model are trained separately.

#	**Methods**	**Precision**	
		with dropout	without dropout
1	LDR	0.865	0.873
2	LDR-rand	0.843	0.849
3	LDR-max-pooling	0.851	0.865
4	LDR-average-pooling	0.838	0.844
5	LDR+LCR-joint	0.887	**0.891**
6	LDR+LCR	0.885	0.889

Table 1. Comparison results of several variants of model

Results of several variants of the model was shown in Table 1. Our model with all randomly initialized words(LDR-rand) does not perform well on its own. Different global features are extracted by max and average operation, and a better performance is achieved in LDR by combining both of them. Performance gains through the use of category representation for similarity score, and the result of LDR+LCR suggests that category information is helpful to disambiguation task, even though LCR is trained for entity classification individually. Joint training is employed in LDR+LCR-joint, and improves the performance of LDR+LCR to 89.1%.

We compare our approach with following baseline methods:

*Vector Averaging:*This method computes the weighted average of all word vectors in the document. We use idf-weighting as the weighting function.

*Bag-of-words:*A document is represented as the bag of its words, disregarding grammar and word order but keeping multiplicity.

*Bag-of-bigrams:*Similar to bag-of words, a document is represented as the bag of its bigrams. Both methods are performed with TF-IDF weighting.

Topic model: A document is represented as a distribution on multiple topics utilizing the toolkit JGibbLDA[1]. The similarity of two documents is measured by the similarity of two topic distributions.

Paragraph Vector: Proposed by [19], an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of text. This algorithm represents each document by a dense vector which is trained to predict words in the document.

#	**Methods**	**Precision**
1	Vector Averaging	0.734
2	Bag-of-words	0.758
3	Bag-of-bigrams	0.767
4	Topic Model	0.
5	Paragraph Vector	0.
6	LDR+LCR	0.889
7	LDR+LCR-joint	0.891

Table 2. Comparison results of models computing features for documents

The result of our method and other baselines are reported in Table 2. In this task, we find that TF-IDF weighting performs better than raw counts and therefore we only report the results of methods with TF-IDF weighting. Our method significantly outperforms bag-of-words and bag-of-bigrams methods, for they lose the ordering words and ignore semantics of the words. Topic model and paragraph vector achieved better performance than BoW model and that suggests that they are useful for capturing the semantics of the document. Both topic model and Paragraph Vector are trained unsupervisedly, while in our method the features for documents are optimized for specific disambiguation task.

3.2 Experiments on NERD Dataset

The named entity recognition and disambiguation(NERD) in CIPS-SIGHAN 2012 is the task detecting entity mentions from raw text and classifying each mention to its real world entity. There are 16 names in the sample data and 32 names in the test data. For each name N, there is a document collection T and knowledge base(KB) which contains several persons, organizations or locations who share the same name N. For each document in T, the task is to find the target

[1] http://jgibblda.sourceforge.net/

entity of the name N in KB; if the target entity of the name N in document is not contained in KB, then the system needs to determine whether N is a common word or not; if not, we need to cluster these documents into subsets, each of which refers to one single entity.

We defined two pseudo entities for each name using the same way as Han [22]. The *other* pseudo entity describes the situation that name is used as a common word and *out* pseudo entity represents the target entities which are not contained in KB. For each document D in test dataset, name N in D is linked to the entity E in KB with highest similarity score $sim(D, E)$. We compare our methods with top three systems participated NERD task in CIP-SIGHAN 2012. The result in Table 3 shows that our method outperforms the best team.

#	Methods	Precision(%)	Recall(%)	F(%)
1	LDR+LCR-joint	83.77	81.52	82.63
2	Han et al.[22]	79.48	80.98	80.22
3	Hao et al.[23]	72.56	79.23	75.75
4	Liu et al.[26]	67.18	85.62	75.39

Table 3. Comparison with top three teams in NERD task

[22] adopt a two-stage method which can incorporate classifying and clustering techniques. [23] proposed an extraction algorithm of keyword features based on the distribution of unlabeled data for this task. [26] extracted 19 kinds of attributes for personal named entity, and trained Support Vector Machine(SVM) to classify documents. All teams above extract attributes information from document using toolkit of named entity recognition(NER) and then design features for name N manually. Our method outperforms the best team by 2.41% without any manual designed features.

4 Related Work

4.1 Named Entity Disambiguation

In recent years , many methods have been proposed for entity disambiguation or linking which dates back to Bunescu and Pasca[2]. Bunescu and Pasca[2] built a system to compare the context of mention to the Wikipedia categories of an candidate entity.(Cucerzan[3]) extended this framework with more carefully designed feature for similarity measure. Many similarity measures have been employed in previous work, such as cosine similarity, Kullback-Leibler divergence,Jaccard distance and so on(Hoffart et al [5]; Bunescu and Pasca[2];Cucerzan[3]) The straightforward solution that measures the relatedness between the context of mention and definitions of candidate entities in knowledge base is local and it lacks collective notion of coherence between Wikipedia pages.Moreover, these man-made features are often duplicate and over-specific.

The straightforward solution that measures the relatedness between the context of mention and definitions of candidate definitions in knowledge base is local and it lacks collective notion of coherence in knowledge base. To overcome the disadvantages of local disambiguation, some works focus on collective disambiguation(Han et al.[4];Ratinov et al.[7];Hoffat et al.[5]).Collective methods disambiguate all mentions in a text simultaneously based on the coherence among decisions. Graph-based methods has been proposed for collective disambiguation(Han[4]),and Page-Rank algorithm is applied to rank nodes in the graph(Alhelbawy [6]). Moro[9] presented Babelfly, a unified graph-based approach to name entity and word sense disambiguation. However, Ratinov[7] has shown that local methods based on similarity of context and entity definition achieve a good performance and are hard to beat, even though collective disambiguation can be improved slightly.

Many efficient methods have been proposed for CLP-2012 named entity recognition and disambiguation task.Han[22] adopt a two-stage method which can incorporate classifying and clustering techniques.Hao[23] proposed an extraction algorithm of keyword features based on the distribution of unlabeled data for this task.[26] extracted 19 kinds of attributes for personal named entity, and trained Support Vector Machine(SVM) to classify documents.

4.2 Representation Learning

To fight the curse of dimensionality, Bengio[8] learns simultaneously a distributed representations for words with the probability function for word sequences based on neural networks. Tang and Wei[13] develop three neural networks to learn sentiment specific word embedding, which encodes sentiment information in the the continuous representation of words. However, most existing algorithms for learning continuous word representations typically only model the syntactic context of words but ignore the ambiguity of words.

Various neural models have been described to achieve phrase level or sentence level representation(Mitchell and Lapata [14];Zanzotto et al.[16];Mikolov et al.[17]).A simple approach of sentence models is that of Neural Bag-of Words(NBoW) models. For instance, the weighted average of all words in document is used as input of following fully connected layers, which loses the word order information in context. Recursive Neural Network(RecNN)(Socher et al.[15]) combining the word vectors in an order given by parse tree, which has been shown work for only sentences because it relies on parsing. Paragragh Vector(Mikolov et al.[19]) has been proposed to construct representations of input sequences of variable length. It does not require task-specific tuning of the word weighting function or does it rely on the parse tree. The paragraph vector containing ambiguous mention is trained in an unsupervised way, which means that the representation of sentence or paragraph isn't suitable for disambiguation task.

To learn entity representations for entity disambiguation, (He et al.[20]) first employed Stacked Denoising Auto-encoders to learn an initial document representation and a supervised fine-tuning stage follows to optimize the representation towards the similarity measure. The input of each document is a binary

bag-of-words vector which loses the word order information. The input layers has 100,000 units and it makes the training process more difficult and time-consuming.

5 Conclusion and Future Work

We propose a approach that automatically learns context-entity similarity for entity disambiguation based on deep neural networks. The intermediate representation are learned using large scale annotations in Baike, without any man-made feature. Experiment reveals the importance of context modeling in entity disambiguation task. In the following work, we are trying to extend our method to other disambiguation task such as word sense disambiguation(WSD) and personal name disambiguation. Representations of ambiguous word and personal name will be learned for a novel similarity measure.

Acknowledgements

Our work is supported by National High Technology Research and Development Program of China (863 Program) (No.2015AA015402), National Natural Science Foundation of China (No.61370117 & No.61433015) and Major National Social Science Fund of China (No.12&ZD227).

References

1. Zheng, Z., Li, F., Huang, M., Zhu, X. (2010, June). Learning to link entities with knowledge base. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (pp. 483-491). Association for Computational Linguistics.
2. Bunescu, R. C., & Pasca, M. (2006, April). Using Encyclopedic Knowledge for Named entity Disambiguation. In EACL (Vol. 6, pp. 9-16).
3. Cucerzan, S. (2007, June). Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In EMNLP-CoNLL (Vol. 7, pp. 708-716).
4. Han, X., Sun, L., & Zhao, J. (2011, July). Collective entity linking in web text: a graph-based method. In Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval (pp. 765-774). ACM.
5. Hoffart, J., Yosef, M. A., Bordino, I., Frstenau, H., Pinkal, M., Spaniol, M., ... & Weikum, G. (2011, July). Robust disambiguation of named entities in text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (pp. 782-792). Association for Computational Linguistics.
6. Alhelbawy A, Gaizauskas R. Graph ranking for collective named entity disambiguation[C]. Annual Meeting of the Association for Computational Linguistics. 2014: 75-80.
7. Ratinov, L., Roth, D., Downey, D., & Anderson, M. (2011, June). Local and global algorithms for disambiguation to wikipedia. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 1375-1384). Association for Computational Linguistics.

8. Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. The Journal of Machine Learning Research, 3, 1137-1155.
9. Moro, A., Raganato, A., & Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics, 2, 231-244.
10. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12, 2493-2537.
11. Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
12. Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188.
13. Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). Learning sentiment-specific word embedding for twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (pp. 1555-1565).
14. Mitchell, J., & Lapata, M. (2010). Composition in distributional models of semantics. Cognitive science, 34(8), 1388-1429.
15. Socher, R., Lin, C. C., Manning, C., & Ng, A. Y. (2011). Parsing natural scenes and natural language with recursive neural networks. In Proceedings of the 28th international conference on machine learning (ICML-11) (pp. 129-136).
16. Zanzotto, F. M., Korkontzelos, I., Fallucchi, F., & Manandhar, S. (2010, August). Estimating linear models for compositional distributional semantics. In Proceedings of the 23rd International Conference on Computational Linguistics (pp. 1263-1271). Association for Computational Linguistics.
17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in Neural Information Processing Systems (pp. 3111-3119).
18. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
19. Le, Q. V., & Mikolov, T. (2014). Distributed representations of sentences and documents. arXiv preprint arXiv:1405.4053.
20. He, Z., Liu, S., Li, M., Zhou, M., Zhang, L., & Wang, H. (2013, August). Learning Entity Representation for Entity Disambiguation. In ACL (2) (pp. 30-34).
21. Song, Y., Huang, J., Councill, I. G., Li, J., & Giles, C. L. (2007, June). Efficient topic-based unsupervised name disambiguation. In Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries (pp. 342-351). ACM.
22. Han, Z. P. L. S. X. (2012). SIR-NERD: A Chinese Named Entity Recognition and Disambiguation System using a Two-Stage Method. CLP 2012, 115
23. Hao, Z., Wong, D. F., & Chao, L. S. (2012). A Template Based Hybrid Model for Chinese Personal Name Disam-biguation. CLP 2012, 121.
24. Tian, W., Pan, X., Yu, Z., Xian, Y., & Yang, X. (2012). Chinese name disambiguation based on adaptive clustering with the attribute features. CLP 2012, 132
25. Han, W., Liu, G., Mao, Y., & Huang, Z. (2012). Attribute based Chinese Named Entity Recognition and Disambigua-tion. CLP 2012, 127.
26. Han, W., Liu, G., Mao, Y., & Huang, Z. (2012). Attribute based Chinese Named Entity Recognition and Disambigua-tion. CLP 2012, 127.