

Singleton Detection for Coreference Resolution via Multi-window and Multi-filter CNN

Kenan Li^{1,3}, Heyan Huang^{1,2(✉)}, Yuhang Guo^{1,2}, and Ping Jian^{1,2}

¹ School of Computer Science and Technology, Beijing Institute of Technology,
Beijing, China

{likenan, hhy63, guoyuhang, pjian}@bit.edu.cn

² Beijing Engineering Research Center of High Volume Language Information
Processing and Cloud Computing Application, 5 South Zhongguancun Street,
Beijing 100081, China

³ Beijing Advanced Innovation Center for Imaging Technology,
Capital Normal University, Beijing 100048, People's Republic of China

Abstract. Mention detection is the first and a key stage in most of coreference resolution systems. Singleton mentions are the ones which appear only once and are not mentioned in the following texts. Singleton mentions always affect the performance of coreference resolution systems. To remove the singleton ones from the automatically predicted mentions, we propose a novel singleton detection method based on multi-window and multi-filter convolutional neural network (MMCNN). The MMCNN model can detect singleton mentions with less use of hand-designed features and more sentence information. Experiments show that our system outperforms all the existing singleton detection systems.

Keywords: Singleton detection · Coreference resolution · Convolutional neural network

1 Introduction

In recent years, we have seen some successful attempts which use coreference information to improve machine translation. Some works [1–3] performed coreference resolution on the source side to extract the coreference information of pronoun, and used the coreference information to improve the pronoun translation. Werlen and Popescu-Belis [4] presented a proof-of-concept of a coreference-aware decoder for document-level machine translation. So, we can see that the effect of coreference resolution is being found in machine translation. But existing coreference resolution systems are not performing very well. In this paper, we focus on the study of coreference resolution.

Coreference resolution that involves identification and clustering of noun phrase mentions that refer to the same real-world entity, and existing coreference systems usually consider a pipelined system, which has two steps: detecting all mentions in a text and clustering mentions into coreference chains. After the mention detection step, coreference resolution is performed on those predicted mentions. The predicted mentions can be split into three categories by their status in the gold standard: singleton

(mentioned just once and isn't included in gold mentions), mention starting a new entity with at least two mentions, and anaphora. Here is an example:

- (1) Opening *[the street gate]₁*, *[they]₂* see *[two soldiers]₃* standing by *[the gate]₄* and *[they]₅* seem to be discussing *[something]₆*.

In example (1), the predicted mentions are *[the street gate]₁*, *[they]₂*, *[two soldiers]₃*, *[the gate]₄*, *[they]₅* and *[something]₆*. *[the street gate]₁* and *[the gate]₄* refer to the same entity. *[the street gate]₁* starts a new entity with at least two mentions. *[the gate]₄* is anaphoric. *[two soldiers]₃* and *[they]₅* refer to the same entity. *[two soldiers]₃* starts a new entity with at least two mentions. *[they]₅* is anaphoric. *[they]₂* and *[something]₆* are singletons. The goal of coreference resolution is to get the coreference chains $\{[the\ street\ gate]_1, [the\ gate]_4\}$ and $\{[two\ soldiers]_3, [they]_5\}$. *[they]₂* and *[something]₆* will be discarded.

We have surveyed the results of some papers [6–9] in terms of gold mentions versus predicted mentions. The results are given in Table 1. As we can see, those systems show a significant decline in performance when running on predicted mentions. These performance gaps are worrisome, because the real goal of NLP systems is to process the raw data with no annotation. Obviously, reducing the difference of gold mentions and predicted mentions can improve coreference resolution performance. The main difference between predicted mentions and gold mentions is that singleton mentions exist only in predicted mentions but not in gold mentions. For instance, we used the rule-based mention detection algorithm from Raghunathan et al. [10] to extract mentions. We got a ratio of singleton mentions to non-singleton mentions of 1.5 to 1. There are a large number of singleton mentions taking part in the clustering process. This makes the clustering more difficult. In example (1), *[they]₅* has a high probability to be linked to singleton mention *[they]₂* in many coreference resolution systems. So, we think filtering out singleton mentions after mention detection step can improve the performance of coreference resolution.

Table 1. Performance gaps between using gold mentions and predicted mentions.

System	Dataset	Gold	Predict	Gap
Chen	CoNLL-12	70.46	59.69	10.77
Illinois-Coref	CoNLL-12	77.22	60.18	17.04
Berkeley	CoNLL-11	76.68	60.42	16.26
Berkeley	CoNLL-12	72.51	61.21	11.30
Prune-and-Score	CoNLL-12	80.26	61.56	18.70

De Marneffe et al. [11] used multiple hand-designed features to detect singleton: the morphosyntactic features, the grammatical role features and semantic environment features. The choice of features is a completely empirical process, mainly based on linguistic intuition, and trial and error. And the feature selection is task dependent, implying additional research for NLP task. Haagsma [12] proposed a singleton detection system based on word embeddings and a multi-layer perceptron (MLP). Haagsma used the mention itself, context words around the mention and other mentions

rather than whole sentence as inputs. This breaks the semantic integrity of sentence, and the MLP has a poor ability of extracting feature from word embeddings. In order to minimize the use of hand-designed features and make full use of the complete semantic information contained in sentence, we explore a novel singleton detection system which uses multi-window and multi-filter convolutional neural network (MMCNN) to extract useful features from sentence. By using multi-window and multi-filter, we can extract multi-granularity level features, and extract enough information for the coarser feature. We transform words of sentence into vectors by looking up word embeddings. Then, sentence level features are learned by using a convolutional approach. We concatenate them with some additional surface features to form the final feature representation. Finally, we judge whether a mention is a singleton by a logistic regression classifier. The experimental result show that our system outperforms the two existing systems [11, 12]. To give a fuller evaluation of the coreference applications of our model, we incorporate our best model into a learning-based coreference resolution system [8].

The remainder of the paper is structured as follows. Section 2 discusses relevant works towards singleton detection. Section 3 presents the main framework and detail description of our model. Section 4 shows the specific data we use and the experimental results of our method and the compared baselines. And the conclusions are given in Sect. 5.

2 Related Work

The first step of coreference resolution system is mention extraction. In order to allow the coreference resolver to see most of the possible mentions, most systems use a high recall, low precision module to extract mention. So, there is a significant gap between gold mentions and predicted mentions. To reduce the gap, many mention filtering approaches have been tried.

Raghunathan et al. [10] used a rule-based method to remove spurious mentions such as numeric entities and pleonastic *it* pronouns. Yuan et al. [13] focused on filtering *it*, and use Decision Tree (C4.5) algorithm to classify whether *it* refers to entity based on the training data. Björkelund and Farkas [14] extracted referential and non-referential examples from the training set and train binary MaxEnt classifiers for the pronouns *it*, *we* and *you*. De Marneffe et al. [11] worked on singleton detection, and use a logistic regression classifier with both discourse-theoretically inspired semantic features and more superficial features (animacy, NE type, POS, etc.) to perform singleton detection. The state-of-the-art in mention filtering is the system described by Haagsma [12]. Haagsma proposed a system based on word embeddings and neural networks, feeding the mention itself, context words around the mention and other mentions in the context into a multi-layer perceptron to identify whether a mention is singleton.

Here, we focus on filtering the singleton mentions, and propose a new convolutional neural (CNN) network to extract sentence level features for singleton detection. In other NLP tasks, many existing works have proved that CNN could extract useful features from the word tokens to improve the performance of NLP task. Some works [15–17] used CNN to classify the relation of two target nouns in a sentence. Kim [18]

reported on a series of experiments with CNN trained on top of pre-trained word vectors for sentence-level classification tasks. Iida et al. [19] used a multi-column Convolutional Neural Network (MCNN) for predicting zero anaphoric relations. An MCNN has several independent columns, each of which has its own convolutional and pooling layers.

3 Method

After mentions are predicted from raw texts, filtering out the singleton mentions can improve the performance of coreference resolution. We propose a novel singleton detection system which uses multi-window and multi-filter convolutional neural network (MMCNN) to extract sentence level features for singleton detection. Figure 1 describes the architecture of our model. First, the sentence which mention is located on is fed into a convolution-pooling layer to extract sentence level features. Then, sentence level features and additional features are directly concatenated to form the final feature representation. Finally, the logistic regression classifier computes a score to identify whether the mention is a singleton.

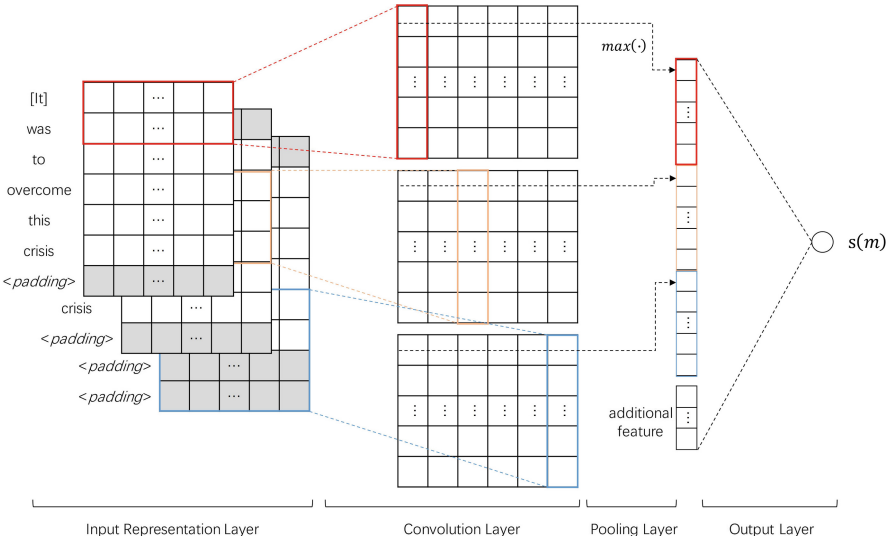


Fig. 1. Schematic overview of our multi-window and multi-filter neural convolutional network.

3.1 Input Representation

The input of the MMCNN is the sentence which mention is located on, and we want to extract context feature of mention in sentence. But the length of mention is variable, a mention can be made up of one or more words. In convolution operation, if the given mention contains many words, it's internal information could affect the context information. Peng et al. [5] proposed a coreference system which focus on mention

heads rather than mentions in coreference step. This work proved that mentions had an intrinsic structure, in which mention heads carried the crucial information. So, mention head is used instead of mention as the input in this work. Given a sentence S with a target mention \mathbf{m} , consists of \mathbf{n} words $S = \{w_1, w_2, \dots, w_n\}$ the replace operation change S to $S^* = \{w_1, \dots, w_{m-}, w_h, w_{m+}, \dots, w_n\}$ where w_{m-} is the preceding word of \mathbf{m} , w_h is the head word of \mathbf{m} , w_{m+} is the following word of \mathbf{m} . In order to feed S^* to the MMCNN, we also need to convert S^* into real-valued vectors.

Word Embeddings. Word representations are encoded by row vectors in an embedding matrix $\mathbf{W}^e \in \mathbb{R}^{|V| \times d^w}$, where V is a fixed-sized vocabulary, $|V|$ is the size of V , d^w is the size of the word embedding. In matrix \mathbf{W}^e , each row $\mathbf{W}_i^e \in \mathbb{R}^{d^w}$ corresponds to the word embedding of the i -th word in the vocabulary. We transform a word w into word embedding by:

$$r^w = \text{look_up}(w) \quad (1)$$

where the operation $\text{look_up}(\cdot)$ gets the index i^w of w in V , then extracts $\mathbf{W}_{i^w}^e$ as the word embedding r^w of w .

Position Embeddings. Zeng et al. [15] proposed the use of word position embeddings (position features) to help the CNN by keeping track of how close words were to the target nouns. In this work, we use the word position embeddings (WPE) to keep track of how close the words are to the target mention \mathbf{m} . The WPE is derived from the relative distances of the current word to the target mention \mathbf{m} . For example, the relative distances of “Opening” and “soldiers” to [the street gate]₁ are -1 and 5. Each relative distance is mapped to a vector $wpe^w \in \mathbb{R}^{d^p}$, which is initialized with random numbers.

Finally, the word embedding and the position embedding of each word are concatenated to form the input for the convolutional layer, a real-valued matrix $\mathbf{S}^e = \{\text{emb}^{w_1}, \text{emb}^{w_2}, \dots, \text{emb}^{w_n}\}$, where $\text{emb}^{w_i} = [r^{w_i}, wpe^{w_i}] \in \mathbb{R}^{d^w + d^p}$.

3.2 Convolution and Pooling

In the next step, the matrix \mathbf{S}^e is fed into the convolutional layer to extract higher level features. We use multi-window and multi-filter convolution with \mathbf{S}^e . Each filter focusing on difference feature can extract a vector from \mathbf{S}^e . In the pooling layer, a max pooling operation is used to create fixed-sized sentence level features.

Convolution. We apply a multi-window and multi-filter convolution operation to \mathbf{S}^e . In the single-window architecture, a convolution operation involves a filter $\mathbf{W}^c \in \mathbb{R}^{1 \times k(d^w + d^p)}$, which is applied to a window of k words to produce a new feature. For example, a feature c_i is generated by:

$$c_i = f(\mathbf{W}^c \cdot \text{emb}^{w_i:w_i+k-1} + b^c) \quad (2)$$

where $b \in \mathbb{R}^1$ is a bias term, f is the non-linear function ReLU and $\text{emb}^{w_i:w_i+k-1} \in \mathbb{R}^{k(d^w + d^p) \times 1}$ refer to the concatenation of words $\text{emb}^{w_i}, \text{emb}^{w_{i+1}}, \dots, \text{emb}^{w_{i+k-1}}$. Then, this filter is applied to each possible window of words in \mathbf{S}^e , and we get a feature map

$c = \{c_1, c_2, \dots, c_n\}$ for S^e . To overcome the issue of referencing words with indices outside of the sentence boundaries, we augment S^e with a special *padding* token replicated $\lfloor \frac{k-1}{2} \rfloor$ times at the beginning and $\lceil \frac{k-1}{2} \rceil$ times at the end. So, the dimension of c is equal to the sentence length. After all filters is applied to S^e , we will get a matrix $C \in \mathbb{R}^{d^c \times n}$, where d^c is the number of filter and n is the length of sentence. In the multi-window architecture, illustrated in Fig. 1. We use multiple sizes of window to focus on multi-granularity level features, and the coarser feature need more filters to extract enough information. In other words, the number d_i^c of the filter which focus on k_i words increases with k_i , and those k_i -size filters can extract a matrix $C_i \in \mathbb{R}^{d_i^c \times n}$ from S^e .

Pooling. After the convolutional layer, the matrix C_i contains local features around each word in the sentence. Then, this pooling layer combines these local features by using a max pooling operation to create a fixed-sized and more abstract higher-level feature vector from matrix C_i . The max aggregating function is popular, as it bears responsibility for identifying the most important or relevant features from the output of the convolutional layer. Concretely, the max pooling operation can be described as:

$$h_j = \max C_i(j, \cdot), \quad 0 \leq j \leq d_i^c \quad (3)$$

where $C_i(j, \cdot)$ denotes the i -th row of matrix C_i . The pooling layer extract a feature vector h_i from matrix C_i , the dimension of h_i is related to the number d_i^c of filter, not the sentence length. All the outputs of the pooling layer are concatenated into the sentence level features $x \in \mathbb{R}^{d^s}$, where d^s equals the number of all multi-window filters.

3.3 Additional Features

We argue some information can't be extracted by the convolution-pooling operation. So, some surface features which are common in coreference resolution are extracted to provide more information for singleton detection. The full set of additional features is as follows:

Type. The type of the mention is divided into 4 categories: "PROPER", "NOMINAL", "PRONOMINAL", and "LIST".

Named entities. Our model also includes named-entity features for all the 18 OntoNotes entity-types.

Number. The number of the mention has 3 categories: "SINGULAR", "PLURAL", "UNKNOWN".

Length. Like the length features proposed by Clark and Manning [20], the mention length is binned into one of the buckets [1, 2, 3, 4, 5, 6–7, 8–10, 11–14, 15–19, 20+] and then encoded into a one-hot vector.

String match. Whether the mention is exact or relax string match with other mentions.

Those surface features are extracted from text and encoded into one-hot vectors, then those vectors will be concatenated with sentence level features to create the final feature representation $z \in \mathbb{R}^{d^o \times 1}$, where d^o equals the dimension d^s of the sentence

level features plus the dimension d^a of the additional features. Then, the scoring function is defined by:

$$s(m) = \text{sigmoid}(\mathbf{W}z + b) \quad (4)$$

where \mathbf{W} is a $1 \times d^o$ weight matrix, the $\text{sigmoid}(\cdot)$ is a standard nonlinear function and makes $s(m) \in (0, 1)$.

3.4 Training Procedure

Now that we have defined our model, we must decide how to train its parameters $\theta = (\mathbf{W}^e, \mathbf{W}^c, b^c, \mathbf{W}, b)$. In our work, the singleton detection is considered as a logistic regression problem. We choose the logarithmic loss function as the object function. Given t training examples of the form (m_k, a_k^*) , the loss function is defined as:

$$L(\theta) = -\frac{1}{t} \sum_{k=1}^t [a_k^* \log s(m_k) + (1 - a_k^*) \log(1 - s(m_k))] + \lambda \|\theta\|_2 \quad (5)$$

where the relation between m_k and a_k^* is described as:

$$a_k^* = \begin{cases} 1 & \text{if } m_k \text{ is a singleton} \\ 0 & \text{if } m_k \text{ is't a singleton} \end{cases} \quad (6)$$

We update θ using RMSProp and apply dropout with a rate of 0.5 to the concatenation layer. L_2 regularization is used to avoid over-fitting problem.

4 Experiments

4.1 Experimental Setup

Dataset. All experiments use the English portion of the CoNLL 2012 Shared Task dataset [21], which is based on the OntoNotes corpus [22]. The data set contains 3,493 documents with 1.6 million words. The documents are from seven different domains: broadcast conversation (20%), broadcast news (13%), magazine (7%), newswire (21%), telephone conversation (13%), weblogs and newsgroups (15%), and pivot text (11%). We use the standard experimental split of the CoNLL 2012 Shared Task with the training set containing 2,802 documents and 156K annotated mentions, the development set containing 343 documents and 19K annotated mentions, and the test set containing 348 documents and 20K annotated mentions.

In OntoNotes corpus, the singleton mentions are not annotated. So, an extra mention extract step is necessary to get singleton mentions from the data set. We used the rule-based mention detection algorithm from Stanford's rule-based coreference system [10], which first extracted pronouns and maximal NP projections as candidate mentions and then filtered this set with rules. Then, we went through all predicted mentions and select the mentions which were not found in the gold mentions as

singleton mentions. In Table 2, we can find that singleton mentions have a large proportion in the total mentions. This proves that the singleton mention is the main difference of predicted mentions and gold mentions.

Table 2. CoNLL-2012 Shared Task data statistics.

Dataset	Docs	Predicted mentions	
		Coreferent	Singleton
Training	2,802	141,509	198,877
Development	343	17,093	25,547
Test	348	18,128	25,553

Settings. There are some hyperparameters in our model, such as the dimension d^w of word embeddings, the dimension d^p of word position embeddings, the dimension d^a of the additional features, the learning rate η , the regularization parameter λ , the window size k and the number d^c of filter. We initialized our word embeddings with 50 dimensional ones produced by word2vec [23] on the Gigaword corpus for English. Like Zeng et al. [15], we chose $d^p = 5$. All the additional features were encoded in a one-hot vector and concatenated into a 39-dimensional vector. We experimentally studied the effects of the three parameters in our proposed method: learning rate η , regularization parameter λ and the window size k . In Fig. 2, we respectively varied the number of hyper parameters $\{\eta, \lambda, k\}$ and compute the F1 on the CoNLL-2012 development set. At the beginning, the performance became larger and larger with the increase of the regularization parameter λ . While λ is greater than 10^{-9} , the performance starts to fall. When learning rate $\eta = 0.001$, our model got the best performance. the multi-window architecture performs better than the single-window architecture. Table 3 reports all the hyperparameters used in our model.



Fig. 2. Effect of hyperparameters.

Table 3. Hyperparameters used in our experiments.

Hyperparameter	d^w	d^p	d^a	η	λ	k	d^c	d^s
Value	50	5	39	0.001	10^{-9}	[2, 3, 4]	[16, 24, 32]	72

4.2 Singleton Detection Results

We compared the results of our system with state-of-the-art singleton detection approaches, the performance measured by F1-score is shown in Table 4. De Marneffe et al. [11] only reported scores on the CoNLL-2012 development set. Haagsma [12] reported scores on the CoNLL-2012 development and test set. To prove the sentence level features extracted from sentence by MMCNN are effective, we built a Surface system which only uses the additional surface features to detect singleton mentions. Our method achieves the best performance among all the compared methods. We also performed a bootstrap resample test ($p < 0.05$) [24], which indicates that our method significantly outperforms the baseline systems.

Table 4. Comparison with the current state-of-the-art approaches on the CoNLL 2012 dataset. Significant differences ($p < 0.05$) from the baseline are marked *.

System	Singleton detection		
	Precision	Recall	F1
Marneffe(dev)	81.1	80.8	80.9
Haagsma(dev)	79.77	85.83	82.69
Haagsma(test)	81.57	83.78	82.66
Surface(dev)	80.22	85.30	82.08
Surface(test)	77.54	84.83	81.02
MMCNN(dev)	81.15	87.27	84.10*
MMCNN(test)	78.35	88.22	83.00*

4.3 Application to Coreference Resolution

To further evaluate the performance of singleton detection, we incorporated our model into the Berkeley Coreference System, and took the Berkeley Coreference System as a baseline system. Simply, our model was used as a pre-filtering step to coreference resolution. The predicted mentions were fed into our model, then the singleton mentions were filtered out. Like Haagsma [12], this baseline system used the ‘FINAL’ feature set, and the dataset used in this experiment is the 2012 development set. In our model, the threshold value of 0.95 was chosen so that the singleton classification had a precision of approximately 90%. Table 5 shows the performance of the Berkeley system, and all the coreference results are obtained using the latest version (v8) of the CoNLL scorer [25]. As we can see, this work gets a significant 2.5 percentage point increase over the baseline. This means the singleton detection is meaningful for the coreference resolution. In contrast, Haagsma [12] reported a non-significant performance increase of 0.3 percentage points for the Berkeley. In Table 5, ‘-’ indicates that we could not find published results for those cases.

Table 5. Performance of the Berkeley system on the 2012 development set. Significant differences ($p < 0.05$) from the baseline are marked *.

System	MUC			B ³			CEAF _e			CoNLL
	P	R	F ₁	P	R	F ₁	P	R	F ₁	F ₁
Berkeley	73.44	67.67	70.44	63.13	55.54	59.09	56.99	54.21	55.57	61.71
Haagsma	-	-	-	-	-	-	-	-	-	62.02
This work	75.78	70.31	72.94	65.42	58.65	61.85	59.28	56.46	57.84	64.21*

5 Conclusion

Aiming at reducing the use of hand-designed features and making full use of the information contained in sentence, we explored a novel singleton detection system which uses multi-window and multi-filter convolutional neural network (MMCNN) to extract useful features from sentence, and combined some additional features to compute a score for the singleton detection. The results show that singleton detection can improve the performance of coreference resolution, and our system outperforms all the existing singleton detection systems. Comparing to Surface system proves that the sentence level features extracted from sentence by MMCNN are effective.

In future, we will focus on the application of coreference resolution in machine translation. Since the existing works mainly focus on using coreference information to improve pronoun translation, we plan to explore the deeper integration of machine translation and coreference resolution, such as using the coreference information to solve ambiguity problem in machine translation.

Acknowledgment. This research is supported by the National Basic Research Program of China (973 Program, No. 2013CB329303), National Natural Science Foundation of China (NSFC, No. 61502035) and Beijing Advanced Innovation Center for Imaging Technology (BAICIT, No. 2016007).

References

1. Hardmeier, C., Fondazione, M.F., Kessler B.: Modelling pronominal anaphora in statistical machine translation. In: Proceedings of the seventh International Workshop on Spoken Language Translation, pp. 283–289 (2010)
2. Luong, N.Q., Werlen, L.M., Popescu-Belis, A.: Pronoun translation and prediction with or without coreference links. In: The Workshop on Discourse in Machine Translation, pp. 94–100 (2015)
3. Luong, N.Q., Popescu-Belis, A.: Improving pronoun translation by modeling coreference uncertainty. In: Conference on Machine Translation, vol. 1, Research Papers (2016)
4. Werlen, L.M., Popescu-Belis, A.: Using Coreference Links to Improve Spanish-to-English Machine Translation. Idiap (2017)
5. Peng, H., Chang, K.W., Dan, R.: A joint framework for coreference resolution and mention head detection. In: Nineteenth Conference on Computational Natural Language Learning (2015)

6. Chang, K.W., Samdani, R., Rozovskaya, A., Sammons, M., Roth, D.: Illinois-Coref: the UI system in the CoNLL-2012 shared task. In: Joint Conference on EMNLP and CoNLL - Shared Task. Association for Computational Linguistics, pp. 113–117 (2012)
7. Chen, C., Ng, V.: Combining the best of two worlds: a hybrid approach to multilingual coreference resolution. In: Joint Conference on EMNLP and CoNLL - Shared Task. Association for Computational Linguistics, pp. 56–63 (2012)
8. Durrett, G., Klein, D.: Easy victories and uphill battles in coreference resolution. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1971–1982. ACL (2013)
9. Ma, C., Doppa, J.R., Orr, J.W., Mannem, P., Fern, X.Z., Dietterich, T.G., Tadepalli, P.: Prune-and-score: learning for greedy coreference resolution. In: Conference on Empirical Methods in Natural Language Processing, pp. 2115–2126 (2014)
10. Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., Manning, C.: A multi-pass sieve for coreference resolution. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 492–501. Association for Computational Linguistics (2010)
11. De Marneffe, M.C., Recasens, M., Potts, C.: Modeling the lifespan of discourse entities with application to coreference resolution. *J. Artif. Intell. Res.* **52**, 445–475 (2015)
12. Haagsma, H.: Singleton detection using word embeddings and neural networks. In: ACL 2016 (2016)
13. Yuan, B., Chen, Q., Xiang, Y., Wang, X., Ge, L., Liu, Z., Si, X.: A mixed deterministic model for coreference resolution. In: Joint Conference on EMNLP and CoNLL - Shared Task. Association for Computational Linguistics (2012)
14. Björkelund, A., Farkas, R.: Data-driven multilingual coreference resolution using resolver stacking. In: Joint Conference on EMNLP and CoNLL - Shared Task (2012)
15. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation classification via convolutional deep neural network. In: COLING (2014)
16. dos Santos, C.N., Xiang, B., Zhou, B.: Classifying relations by ranking with convolutional neural networks. *Computer Science* (2015)
17. Wang, L., Cao, Z., de Melo, G., Liu, Z.: Relation classification via multi-level attention CNNs. In: Meeting of the Association for Computational Linguistics (2016)
18. Kim, Y.: Convolutional neural networks for sentence classification. *Eprint Arxiv* (2014)
19. Iida, R., Torisawa, K., Oh, J.H., Kruengkrai, C., Kloetzer, J.: Intra-sentential subject zero anaphora resolution using multi-column convolutional neural network. In: Proceedings of EMNLP (2016)
20. Clark, K., Manning, C.D.: Improving coreference resolution by learning entity-level distributed representations. In: Meeting of the Association for Computational Linguistics (2016)
21. Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., Zhang, Y.: CoNLL-2012 shared task: modeling multilingual unrestricted coreference in OntoNotes. In: Joint Conference on EMNLP and CoNLL - Shared Task (2012)
22. Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: OntoNotes: the 90% solution. In: Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers, pp. 57–60. Association for Computational Linguistics (2006)
23. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
24. Koehn, P.: Statistical significance tests for machine translation evaluation. In: Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 388–395 (2004)
25. Pradhan, S., Luo, X., Recasens, M., Hovy, E.H., Ng, V., Strube, M.: Scoring coreference partitions of predicted mentions: a reference implementation. In: Meeting of the Association for Computational Linguistics (2014)

Machine Translation

13th China Workshop, CWMT 2017, Dalian, China,
September 27-29, 2017, Revised Selected Papers

Wong, F.D.; Xiong, D. (Eds.)

2017, XI, 125 p. 25 illus., Softcover

ISBN: 978-981-10-7133-1