Empty Category Detection With Joint Context-Label Embeddings

Xun Wang, Katsuhito Sudoh and **Masaaki Nagata**
NTT Communication Science Laboratories
Kyoto 619-0237, Japan
wang.xun,sudoh.katsuhito,nagata.masaaki@lab.ntt.co.jp

Abstract

This paper presents a novel technique for empty category (EC) detection using distributed word representations. A joint model is learned from the labeled data to map both the distributed representations of the contexts of ECs and EC types to a low dimensional space. In the testing phase, the context of possible EC positions will be projected into the same space for empty category detection. Experiments on Chinese Treebank prove the effectiveness of the proposed method. We improve the precision by about 6 points on a subset of Chinese Treebank, which is a new state-of-the-art performance on CTB.

1 Introduction

The empty category (EC) is an important concept in linguistic theories. It is used to describe nominal words that do not have explicit phonological forms (they are also called "covert nouns"). This kind of grammatical phenomenons is usually caused by the omission or dislocation of nouns or pronouns. Empty categories are the "hidden" parts of text and are essential for syntactic parsing (Gabbard et al., 2006; Yang and Xue, 2010). As a basic problem in NLP, the resolution of ECs also has a huge impact on lots of downstream tasks, such as co-reference resolution (Ponzetto and Strube, 2006; Kong and Ng, 2013), long distance dependency relation analysis (Marcus et al., 1993; Xue et al., 2005). Research also uncovers the important role of ECs in machine translation. Some recent work (Chung and Gildea, 2010; Xiang et al., 2013) demonstrates the improvements they manage to obtain through EC detection in Chinese-English translation.

To resolve ECs, we need to decide 1) the position and type of the EC and 2) the content of the EC (to which element the EC is linked to if plausible). Existing research mainly focuses on the first problem which is referred to as EC detection (Cai et al., 2011; Yang and Xue, 2010), and so is this paper. As ECs are words or phrases inferable from their context, previous work mainly designs features mining the contexts of ECs and then trains classification models or parsers using these features (Xue and Yang, 2013; Johnson, 2002; Gabbard et al., 2006; Kong and Zhou, 2010). One problem with these human-developed features are that they are not fully capable of representing the semantics and syntax of contexts. Besides, the feature engineering is also time consuming and labor intensive.

Recently neural network models have proven their superiority in capturing features using low dense vector compared with traditional manually designed features in dozens of NLP tasks (Bengio et al., 2006; Collobert and Weston, 2008; Socher et al., 2010; Collobert et al., 2011; Li and Hovy, 2014; Li et al., 2014).

This paper demonstrates the advantages of distributed representations and neural networks in predicting the locations and types of ECs. We formulate the EC detection as an annotation task, to assign predefined labels (EC types) to given contexts. Recently, Weston et al. (2011) proposed a system taking advantages of the hidden representations of neural networks for image annotation which is to annotate images with a set of textual words. Following the work, we design a novel method for EC detection. We represent possible EC positions using the word embeddings of their contexts and then map them to a low dimension space for EC detection.

Experiments on Chinese Treebank show that the proposed model obtains significant improvements over the previous state-of-the-art methods based on strict evaluation metrics. We also identify the dependency relations between ECs and their heads, which is not reported in previous work. The dependency relations can help us with the resolution of ECs and benefit other tasks, such as full parsing and machine translation in practice.

2 Proposed Method

We represent each EC as a vector by concatenating the word embeddings of its contexts. As is shown in Fig. 1, we learn a map MAP_A from the annotated data, to project the ECs' feature vectors to a low dimension space K. Meanwhile, we also obtain the distributed representations of EC types in the same low dimension space K. In the testing phase, for each possible EC position, we use MAP_A to project its context feature to the same space and further compare it with the representations of EC types for EC detection.

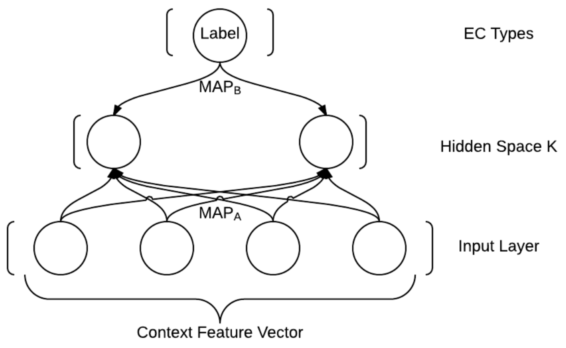


Figure 1: System Architecture

Distributed representations are good at capturing the semantics and syntax of contexts. For example, with word embeddings we are able to tell that "吃/eat" and "喝/drink" have a closer relationship than "吃/eat" and "走/walk" or "喝/drink" and "走/walk". Thus the knowledge we learn from: "EC(你/You)-吃/have-EC(晚饭/supper)-了/past tense marker-么/question marker" could help us to detect ECs in sentences such as "EC(你/You)-饮料/beverage-喝/drink-了/past tense marker-么/question marker", which are similar, though different from the original sentence.

Below is a list of EC types contained in the Chinese Treebank, which are also the types of EC we are to identity in this work.

- pro: small pro, refer to dropped pronouns.
- PRO: big PRO, refer to shared elements in control structures or elements that have generic references.
- OP: null operator, refer to empty relative pronouns.
- T: trace left by A'-movement, e.g., topicalization, relativization.
- RNR: used in right nodes rising.
- *: trace left by passivization, raising.
- Others: other ECs.

According to the reason that one EC is caused, we are able to assign it one of the above categories.

We can formulate EC detection as a combination of a two-class classification problem (is there an EC or not) and a seven-class classification problem (what type the EC is if there is one) following the two-pass method. For one-pass method, EC detection can be formulated as an eight-class (seven EC types listed above plus a dummy "No" type) classification problem. Previous research shows there is no significant differences between their performances (Xue and Yang, 2013). Here we adopt the one-pass method for simplicity.

2.1 System Overview

The proposed system consists of two maps.

MAP_A is from the feature vector of an EC position to a low dimensional space.

$$\begin{aligned} &MAP_A : R^n \rightarrow R^k, k \ll n \\ &f_A(X) \rightarrow W_A X \end{aligned} \quad (1)$$

MAP_A is a linear transformation, and W_A is a $k * n$ matrix.

The other one is from labels to the same low dimensional space.

$$\begin{aligned} &MAP_B : \{Label_1, Label_2, ...\} \in R \rightarrow R^k \\ &f_B(Label_i) \rightarrow W_B^i \end{aligned} \quad (2)$$

MAP_B is also a linear transformation. W_B^i is a k dimensional vector and it is also the distributed representation of $Label_i$ in the low dimensional space.

The two maps are learned from the training data simultaneously. In the testing phase, for any possible EC position to be classified, we extract the corresponding feature vector X, and then map it to the low dimensional space using $f_A(X) = W_A X$. Then we have $g_i(X)$ for each $Label_i$ as follows:

$$g_i(X) = (f_A(X))^T W_B^i \quad (3)$$

For each possible label $Label_i$, $g_i(X)$ is the score that the example having a $Label_i$ and the label predicted for the example is the i that maximizes $g_i(X)$.

Following the method of Weston et al. (2011), we try to minimize a weighted pairwise loss, learned using stochastic gradient descent:

$$\sum_X \sum_{i \neq c} L(rank_c(X)) \max(0, (g_i(X) - g_c(X))) \quad (4)$$

Here c is the correct label for example X, and $rank_c(X)$ is the rank of $Label\ c$ among all possible labels for X. L is a function which reflects our attitude towards errors. A constant function $L = C$ implies we aim to optimize the full ranking list. Here we adopt $L(\alpha) = \sum_{i=1}^{\alpha} 1/i$, which aims to optimize the top 1 in the ranking list, as stated in (Usunier et al., 2009). The learning rate and some other parameters of the stochastic gradient descent algorithm are to be optimized using the development set.

An alternative method is to train a neural network model for multi-class classification directly. It is plausible when the number of classes is not large. One of the advantages of representing ECs and labels in a hidden space is that EC detection usually serves as an intermediate task. Usually we want to know more about the ECs such as their roles and explicit content. Representing labels and ECs as dense vectors will greatly benefit other work such as EC resolution or full parsing. Besides, such a joint embedding framework can scale up to the large set of labels as is shown in the image annotation task (Weston et al., 2011), which makes the identification of dependency types of ECs (which is a large set) possible.

2.2 Context Features Construction

2.2.1 Defining Locations

In a piece of text, possible EC positions can be described with references to tokens, e.g., before the n^{th} token (Yang and Xue, 2010). One problem with such methods is that if there are more than one ECs preceding the n^{th} token, they will occupy the same position and can not be distinguished. One solution is to decide the number of ECs for each position, which complicates the problem. But if we do nothing, some ECs will be ignored.

A compromised solution is to describe positions using parse trees (Xue and Yang, 2013). Adjacent ECs before a certain token usually have different head words, which means they are attached to different nodes (head words) in a parse tree. Therefore it is possible to define positions using “head word, following word” pairs. Thus the problem of EC detection can be formulated as a classification problem: for each “head word, following word” pair, what is the type of the EC? An example is shown in figure 2, in which there are 2 possible EC positions, (吃, 了) and (吃, 。)[1].

[1]Note that there are still problems with the tree based method. As is shown in Fig. 3, the pro and T are attached to the same head word (告别) and share the same

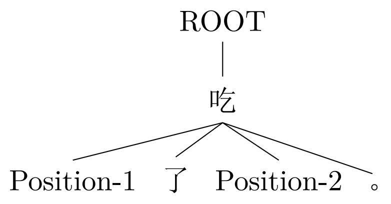


Figure 2: Possible EC Positions in a Dependency Tree

Besides, we keep punctuations in the parse tree so that we can describe all the possible positions using the "head node, following word" pairs, as no elements will appear after a full stop in a sentence.

2.2.2 Feature Extraction

The feature vector is constructed by concatenating the word embeddings of context words that are expected to contribute to the detection of ECs.

1. The head word (except the dummy root node). Suppose words are represented using d dimension vectors, we need d elements to represent this feature. The distributed representations of the head word would be placed at the corresponding positions.

2. The following word in the text. This feature is extracted using the same method with head words.

3. "Nephews", the sons of the following word. We choose the leftmost two.

4. Words in dependency paths. ECs usually have long distance dependencies with words which cannot be fully captured by the above categories. We need a new feature to describe such long distance semantic relations: Dependency Paths. From the training data, we collect all the paths from root nodes to ECs (ECs excluded) together with dependency types. Below we give an example to illustrate the extraction of this kind of features using a complex sentence with a multi-layer hierarchical dependency tree as in Fig. 3. If we have m kinds of such paths with different path types or dependency types, we need md elements to represent this kind of features. The distributed representations of the words would be placed at the corresponding positions in the feature vector and the remaining are set to 0.

following word (德国). But such phenomenas are rare, so here we still adopt the tree based method.

Previous work usually involves lots of syntactic and semantic features. In the work of (Xue and Yang, 2013), 6 kinds of features are used, including those derived from constituency parse trees, dependency parse trees, semantic roles and others. Here we use only the dependency parse trees for the feature extraction. The words in dependency paths we use have proven their potential in representing the meanings of text in frame identification (Hermann et al., 2014).

Take the OP in the sentence shown in Fig. 3 for example. For the OP, its head word is "的", its following word is "告别" and its nephews are "NULL" and "NULL" (ECs are invisible).

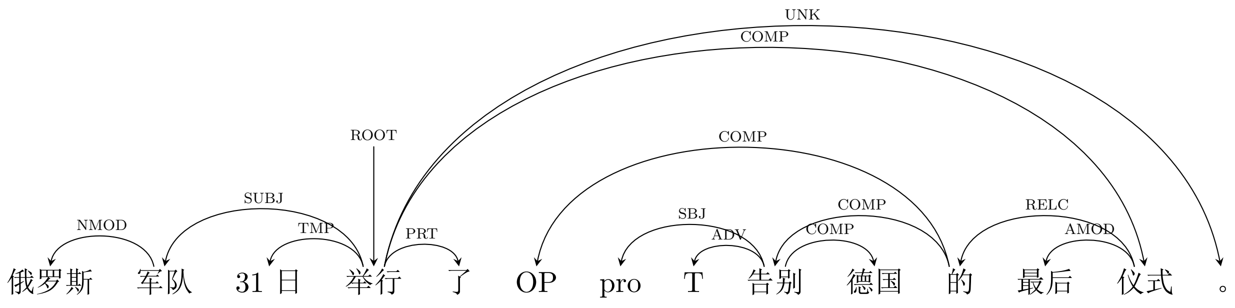
The dependency path from root to OP is: $Root \xrightarrow{ROOT}$ 举行/hold $\xrightarrow{COMP}$ 仪式/ceremony $\xrightarrow{RELC}$ 的/DE $\xrightarrow{COMP}$ OP

For such a path, we have the following subpaths:

$$Root \xrightarrow{ROOT} . \xrightarrow{COMP} . \xrightarrow{RELC} X$$
$$Root \xrightarrow{ROOT} . \xrightarrow{COMP} X$$
$$Root \xrightarrow{ROOT} X$$

For the position of the OP in the given example, the words with corresponding dependency paths are "的", "仪式" and "举行". Similarly, we collects all the paths from other ECs in the training examples to build the feature template.

In the testing phase, for each possible EC position, we place the distributed representations of the right words at the corresponding positions of its feature vector.



俄罗斯/Russian 军队/troops 31 日/31rd 举行/hold 了/past-tense-marker 告别/farewell 德国/Germany 的/DE 最后/final 仪式/ceremony 。

Figure 3: ECs in a Dependency Tree

	Train	Dev	Test
File	81-325, 400-454 500-554, 590-596 600-885, 900	41-80	1-40 901-931
#pro	1023	166	297
#PRO	1089	210	298
#OP	2099	301	575
#T	1981	287	527
#RNR	91	15	32
#*	22	0	19
#Others	0	0	0
Total	6305	979	1748

Table 1: Data Division and EC Distribution

3 Experiments on CTB

3.1 Data

The proposed method can be applied to various kinds of languages as long as annotated corpus are available. In our experiments, we use a subset of Chinese Treebank V7.0.

We split the data set into three parts, training, development and test data. Following the previous research, we use File 1-40 and 901-931 as the test data, File 41-80 as the development data. The training data includes File {81-325, 400-454, 500-554, 590-596, 6000-885, 900}. The development data is used to tune parameters and the final results are reported on the test data. CTB trees are transferred to dependency trees for feature extraction with ECs preserved (Xue, 2007).

The distributed word representation we use is learned using the word2vec toolkit (Mikolov et al., 2013). We train the model on a large Chinese news copora provided by Sogou[2], which contains about 1 billion words after necessary preprocessing. The text is segmented into words using ICTCLAS(Zhang et al., 2003)[3].

3.2 Experiment Settings

Initialization W_A is initialized according to $uniform[-\frac{24}{d_{in}+d_{hidden}}, \frac{24}{d_{in}+d_{hidden}}]$.
And W_B is initialized using $uniform[-\frac{24}{d_{hidden}+d_{out}}, \frac{24}{d_{hidden}+d_{out}}]$.
Here d_{in}, d_{hidden} and d_{out} are the dimensions of the input layer, the hidden space and the label space.

Parameter Tuning To optimize the parameters, firstly, we set the dimension of word vectors to be 80, the dimension of hidden space to be 50. We search for the suitable learning rate in $\{\underline{10^{-1}}, 10^{-2}, 10^{-4}\}$. Then we deal with the dimension of word vectors $\{\underline{80}, 100, 200\}$. Finally we tune the dimension of hidden space in $\{50, 200, \underline{500}\}$ against the F-1 scores. . Those underlined figures are the value of the parameters after optimization. We use the stochastic gradient descent algorithm to optimize the model. The details can be checked here (Weston et al., 2011). The maximum iteration number we used is 10K. In the following experiments,

[2]http://www.sogou.com/labs/dl/cs.html

[3]The word segment standards used by CTB and ICTCLAS are roughly the same with minor differences.

we set the parameters to be learning rate=10^{-1}, word vector dimension=80 and hidden layer dimension=500.

From the experiments for parameter tuning, we find that for the word embeddings in the proposed model, low dimension vectors are better than high dimensions one for low dimension vectors are better in sharing meanings. For the hidden space which represents inputs as uninterpreted vectors, high dimensional vectors are better than low dimensional vectors. The learning rates also have an impact on the performance. If the learning rate is too small, we need more iterations to achieve convergence. If we stop iterations too early, we will suffer under-fitting.

3.3 Results

3.3.1 Metrics and Evaluation

Previous work reports results based on different evaluation metrics. Some work uses linear positions to describe ECs. ECs are judged on a "whether there is an EC of type A before a certain token in the text" basis (Cai et al., 2011). Collapsing ECs before the same token to one, Cai et al. (2011) has 1352 ECs in the test data. Xue and Yang (2013) has stated that some ECs that share adjacent positions have different heads in the parse tree. They judge ECs on a "whether there is an EC of type A with a certain head word and a certain following token in the text" basis. Using this kind of metric, they gets 1765 ECs.

Here we use the same evaluation metric with Xue and Yang (2013). Note that we still cannot describe all the 1838 ECs in the corpora, for on some occasions ECs preceding the same token share the same head word. We also omit some ECs which cause cycles in dependency trees as described in the previous sections. We have 1748 ECs, 95% of all the ECs in the test data, very close to 1765 used by Xue and Yang (2013). The total number of ECs has an impact on the recall. In Table 3, we include results based on each method's own EC count (1748, 1765, 1352 for Ours, Xue's and Cai's respectively) and the real total EC count 1838 (figures in brackets).

Yang and Xue (2010) report an experiment result based on a classification model in a unified parsing frame. We do not include it for it uses different and relativelyThe distributions of ECs in the test data are shown in Table 2.

Type	PRO	pro	T	OP	RNR	*	Others	Total
	297	298	575	527	32	19	0	1748
Xue	305	298	584	527	32	19	0	1765
Cai	299	290	578	134	32	19	0	1352

Table 2: EC Distribution in the Test Data

class	correct	p	r	F1
PRO	162	.479	.545	.510
pro	161	.564	.540	.552
OP	409	.707	.776	.740
T	506	.939	.88	.908
RNR	23	.767	.719	.742
*	0	0	0	0
Overall	1261	.712	.721 (.686)	.717 (.699)
(Xue)	903	.653	.512 (.491)	.574 (.561)
(Cai)	737	.660	.545 (.401)	.586 (.499)

Table 3: Performance on the CTB Test Data

The results are shown in Table 3. We present the results for each kind of EC and compare our results with two previous state-of-the-art methods(Cai et al., 2011; Xue and Yang, 2013).

The proposed method yields the newest state-of-the-art performances on CTB as far as we know. We also identify the dependency types between ECs and their heads. Some ECs, such as pro and PRO, are latent subjects of sentences. They usually serve as SBJ with very few exceptions. While the others may play various roles. There are 31 possible (EC, Dep) pairs. Using the same model, the overall result is $p = 0.701, r = 0.703, f = 0.702$.

3.3.2 Analysis

We compare the effectiveness of different features by eliminating each kind of features described in the previous section. As Table 4 shows, the most important kind is the dependency paths, which cause a huge drop in performance if eliminated. Dependency paths encode words and path pattern information which is proved essential for the detection of ECs. Besides, headwords are also useful. While for the

	-dep	-head	-following	-nephews
F1	.501	.604	.703	.716
	(-.216)	(-.103)	(-.014)	(-.001)

Table 4: Effectiveness of Features

others, we cannot easily make the conclusion that they are of little usage in the identification of ECs. They are not fully explored in the proposed model, but may be vital for EC detection in reality.

Worth to mention is that of the several kinds of ECs, the proposed method shows the best performance on ECs of type T, which represents ECs that are the trace of "A'"-movement, which moves a word to a position where no fixed grammatical function is assigned. Here we give an example:

"[ ] 看起来/seem A 喜欢/like B."
"A 看起来/seem (EC) 喜欢/like B."

A is moved to the head of the sentence as the topic (topicalization) and left a trace which is the EC. To detect this EC, we need information about the action "喜欢/like", the link verb "看起来/seem" and the arguments "A" and "B". ECs of type *T* are very common in Chinese, since Chinese is a topic-prominent language. Using distributed representations, it is easy to encode the context information in our feature vectors for EC detection.

We also have satisfying results and significant improvements for the other types except * (trace of A-movement), which make up about 1% of all the ECs in the test data. Partly because there are too few * examples in the training data. We need to further improve our models to detect such ECs.

4 Discussion

The proposed method is capable of handling large set of labels. Hence it is possible to detect EC types and dependency types simultaneously. Besides, some other NLP tasks can also be formulated as annotation tasks, and therefore can be resolved using the same scheme, such as the frame identification for verbs (Hermann et al., 2014).

This work together with some previous work that uses classification methods (Cai et al., 2011; Xue and Yang, 2013; Xue, 2007), regards ECs in a sentence as independent to each other and even independent to words that do not appear in the feature vectors. Such an assumption makes it easier to design models and features but does not reflect the grammatic constraints of languages. For example, simple sentences in Chinese contain one and only one subject, whether it is an EC or not. If it is decided there is an EC as a subject in a certain place, there should be no more ECs as subjects in the same sentence. But such an important property is not reflected in these classification models. Methods that adopt parsing techniques take the whole parse tree as input and output a parse tree with EC anchored. So we can view the sentence as a whole and deal with ECs with regarding to all the words in the sentence. Iida and Poesio (2011) also take the grammar constraints into consideration by formulating EC detection as an ILP problem. But they usually yield poor performances compared with classification methods partly because the methods they use can not fully explore the syntactic and semantic features.

5 Related Work

Empty category is a complex problem (Li and Hovy, 2015). Existing methods for EC detection mainly explores syntactic and semantic features using classification models or parsing techniques.

Johnson (2002) proposes a simple pattern based algorithm to recover ECs, both the positions and their antecedents in phrase structure trees. Gabbard et al. (2006) presents a two stage parser that uses syntactical features to recover Penn Treebank style syntactic analyses, including the ECs. The first stage, sentences are parse as usual without ECs, and in the second stage, ECs are detected using a learned model with rich text features in the tree structures. Kong and Zhou (2010) reports a tree kernel-based model which takes as input parse trees for EC detection. They also deal with EC resolution, to link ECs to text pieces if possible. They reports their results on Chinese Treebank. Yang and Xue (2010) try to restore ECs from parse trees using a Maximum Entropy model. Iida and Poesio (2011) propose an cross-lingual ILP-based model for zero anaphora detection. Cai et al. (2011) reports a classification model for EC detection. Their method is based on "is there an EC *before* a certain token".

Recently Xue and Yang (2013) further develop the method of Yang and Xue (2010) and explore rich syntactical and semantical features, including paths in parse trees and semantic roles, to train an ME classification model for EC detection and yield the best performance reported using a strict evaluation metric on Chinese Treebank as far as we know.

As we have stated, the traditional features used by above methods are not good at capturing the meanings of contexts. Currently the distributed representations together with deep neural networks have proven their ability not only in representing meaning of words, inferring words from the context, but also in representing structures of text (Socher et al., 2010; Li et al., 2015). Deep neural networks are capable of learning features from corpus, therefore saves the labor of feature engineering and have proven their ability in lots of NLP task (Collobert et al., 2011; Bengio et al., 2006).

The most relevant work to this paper are that of Weston et al. (2011) and that of Hermann et al. (2014). Weston et al. (2011) propose a deep neural network scheme exploring the hidden space for image annotation. They map both the images and labels to the same hidden space and annotate new images according to their representations in the hidden space. Hermann et al. (2014) extend the scheme to frame identification, for which they obtain satisfying results. This paper further uses it for empty category detection with features designed for EC detection.

Compared with previous research, the proposed model simplifies the feature engineering greatly and produces distributed representations for both ECs and EC types which will benefit other tasks.

6 Conclusion

In this paper, we propose a new empty category detection method using distributed word representations. Using the word embeddings of the contexts of ECs as features enables us to employ rich information in the context without much feature engineering. Experiments on CTB have verified the advantages of the proposed method. We successfully beat the existing state-of-the-art methods based on a strict evaluation metric. The proposed method can be further applied to other languages such as Japanese. We will further explore the feasibility of using neural networks to resolve empty categories: to link ECs to their antecedents.

References

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Shu Cai, David Chiang, and Yoav Goldberg. 2011. Language-independent parsing with empty elements. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 212–216. Association for Computational Linguistics.

Tagyoung Chung and Daniel Gildea. 2010. Effects of empty categories on machine translation. In *Proceedings of EMNLP*, pages 636–645. ACL.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Ryan Gabbard, Mitchell Marcus, and Seth Kulick. 2006. Fully parsing the penn treebank. In *Proceedings of the main conference on human language technology conference of the North American chapter of the association of computational linguistics*, pages 184–191. Association for Computational Linguistics.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Proceedings of ACL*.

Ryu Iida and Massimo Poesio. 2011. A cross-lingual ilp solution to zero anaphora resolution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 804–813. Association for Computational Linguistics.

Mark Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 136–143. ACL.

Fang Kong and Hwee Tou Ng. 2013. Exploiting zero pronouns to improve chinese coreference resolution. In *EMNLP*, pages 278–288.

Fang Kong and Guodong Zhou. 2010. A tree kernel-based unified framework for chinese zero anaphora resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 882–891. Association for Computational Linguistics.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 2061–2069.

Jiwei Li and Eduard Hovy. 2015. The nlp engine: A universal turing machine for nlp. *arXiv preprint arXiv:1503.00168*.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 2061–2069.

Jiwei Li, Dan Jurafsky, and Eudard Hovy. 2015. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185*.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS2013*, pages 3111–3119.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 192–199. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Nicolas Usunier, David Buffoni, and Patrick Gallinari. 2009. Ranking with ordered weighted pairwise classification. In *Proceedings of ICML2009*, pages 1057–1064. ACM.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of IJCAI2011*, volume 11, pages 2764–2770.

Bing Xiang, Xiaoqiang Luo, and Bowen Zhou. 2013. Enlisting the ghost: Modeling empty categories for machine translation. In *ACL (1)*, pages 822–831. Citeseer.

Nianwen Xue and Yaqin Yang. 2013. Dependency-based empty category detection via phrase structure trees. In *HLT-NAACL*, pages 1051–1060.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(02):207–238.

Nianwen Xue. 2007. Tapping the implicit information for the ps to ds conversion of the chinese treebank. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistics Theories*.

Yaqin Yang and Nianwen Xue. 2010. Chasing the ghost: recovering empty categories in the chinese treebank. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1382–1390. ACL.

Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. Hhmm-based chinese lexical analyzer ictclas. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 184–187. Association for Computational Linguistics.