Dependency-based empty category detection via phrase structure trees

Nianwen Xue
Brandeis University
Waltham, MA, USA
xuen@brandeis.edu

Yaqin Yang
Brandeis University
Waltham, MA, USA
yaqin@brandeis.edu

Abstract

We describe a novel approach to detecting empty categories (EC) as represented in dependency trees as well as a new metric for measuring EC detection accuracy. The new metric takes into account not only the position and type of an EC, but also the head it is a dependent of in a dependency tree. We also introduce a variety of new features that are more suited for this approach. Tested on a subset of the Chinese Treebank, our system improved significantly over the best previously reported results even when evaluated with this more stringent metric.

1 Introduction

In modern theoretical linguistics, empty categories (ECs) are an important piece of machinery in representing the syntactic structure of a sentence and they are used to represent phonologically null elements such as dropped pronouns and traces of dislocated elements. They have also found their way into large-scale treebanks which have played an important role in advancing the state of the art in syntactic parsing. In phrase-structure treebanks, ECs have been used to indicate long-distance dependencies, discontinuous constituents, and certain dropped elements (Marcus et al., 1993; Xue et al., 2005). Together with labeled brackets and function tags, they make up the full syntactic representation of a sentence.

The use of ECs captures some cross-linguistic commonalities and differences. For example, while both the Penn English TreeBank (PTB) (Marcus et al., 1993) and the Chinese TreeBank (CTB) (Xue et al., 2005) use traces to represent the extraction site of a dislocated element, dropped pronouns (represented as ***pro***s) are much more widespread in the CTB. This is because Chinese is a pro-drop language (Huang, 1984) that allows the subject to be dropped in more contexts than English does. While detecting and resolving traces is important to the interpretation of the syntactic structure of a sentence in both English and Chinese, the prevalence of dropped nouns in Chinese text gives EC detection added significance and urgency. They are not only an important component of the syntactic parse of a sentence, but are also essential to a wide range of NLP applications. For example, any meaningful tracking of entities and events in natural language text would have to include those represented by dropped pronouns. If Chinese is translated into a different language, it is also necessary to render these dropped pronouns explicit if the target language does not allow pro-drop. In fact, Chung and Gildea (2010) reported preliminary work that has shown a positive impact of automatic EC detection on statistical machine translation.

Some ECs can be resolved to an overt element in the same text while others only have a generic reference that cannot be linked to any specific entity. Still others have a plausible antecedent in the text, but are not annotated due to annotation limitations. A common practice is to resolve ECs in two separate stages (Johnson, 2002; Dienes and Dubey, 2003b; Dienes and Dubey, 2003a; Campbell, 2004; Gabbard et al., 2006; Schmid, 2006; Cai et al., 2011). The first stage is *EC detection*, where empty categories are first located and typed. The second stage

Proceedings of NAACL-HLT 2013, pages 1051–1060,
Atlanta, Georgia, 9–14 June 2013.

is *EC resolution*, where empty categories are linked to an overt element if possible.

In this paper we describe a novel approach to detecting empty categories in Chinese, using the CTB as training and test data. More concretely, EC detection involves (i) identifying the position of the EC, relative to some overt word tokens in the same sentence, and (ii) determining the type of EC, e.g., whether it is a dropped pronoun or a trace. We focus on EC detection here because most of the ECs in the Chinese Treebank are either not resolved to an overt element or linked to another EC. For example, dropped pronouns (***pro***) are not resolved, and traces (***T***) in relative clauses are linked to an empty relative pronoun (***OP***).

In previous work, ECs are either represented linearly, where ECs are indexed to the following word (Yang and Xue, 2010) or attached to nodes in a phrase structure tree (Johnson, 2002; Dienes and Dubey, 2003b; Gabbard et al., 2006). In a linear representation where ECs are indexed to the following word, it is difficult to represent consecutive ECs because that will mean more than one EC will be indexed to the same word (making the classification task more complicated). While in English consecutive ECs are relatively rare, in Chinese this is very common. For example, it is often the case that an empty relative pronoun (***OP***) is followed immediately by a trace (***T***). Another issue with the linear representation of ECs is that it leaves unspecified where the EC should be attached, and crucial dependencies between ECs and other elements in the syntactic structure are not represented, thus limiting the utility of this task.

In a phrase structure representation, ECs are attached to a hierarchical structure and the problem of multiple ECs indexed to the same word token can be avoided because linearly consecutive ECs may be attached to different non-terminal nodes in a phrase structure tree. In a phrase structure framework, ECs are evaluated based on their linear position as well as on their contribution to the overall accuracy of the syntactic parse (Cai et al., 2011).

In the present work, we propose to look at EC detection in a dependency structure representation, where we define EC detection as (i) determining its linear position relative to the following word token, (ii) determining its head it is a dependent of, and (iii) determining the type of EC. Framing EC detection this way also requires a new evaluation metric. An EC is considered to be correctly detected if its linear position, its head, and its type are all correctly determined. We report experimental results that show even using this more stringent measure, our EC detection system achieved performance that improved significantly over the state-of-the-art results.

The rest of the paper is organized as follows. In Section 2, we will describe how to represent ECs in a dependency structure in detail and present our approach to EC detection. In Section 3, we describe how linguistic information is encoded as features. In Section 4, we discuss our experimental setup and present our results. In Section 5, we describe related work. Section 6 concludes the paper.

2 Approach

In order to detect ECs anchored in a dependency tree, we first convert the phrase structure trees in the CTB into dependency trees. After the conversion, each word token in a dependency tree, including the ECs, will have one and only one head (or parent). We then train a classifier to predict the position and type of ECs in the dependency tree. Let W be a sequence of word tokens in a sentence, and T is syntactic parse tree for W, our task is to predict whether there is a tuple (h, t, e), such that h and t are word tokens in W, e is an EC, h is the head of e, and t immediately follows e. When EC detection is formulated as a classification task, each classification instance is thus a tuple (h, t). The input to our classifier is T, which can either be a phrase structure tree or a dependency tree. We choose to use a phrase structure tree because phrase structure parsers trained on the Chinese Treebank are readily available, and we also hypothesize that phrase structure trees have a richer hierarchical structure that can be exploited as features for EC detection.

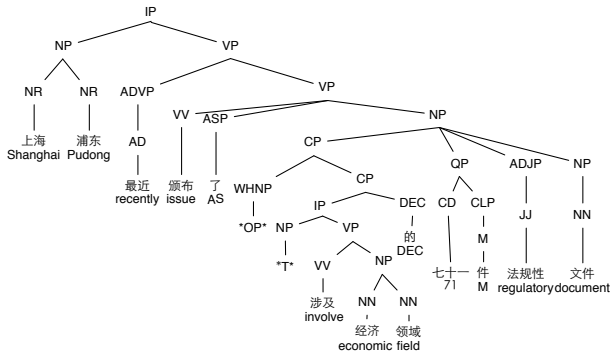
2.1 Empty categories in the Chinese Treebank

According to the CTB bracketing guidelines (Xue and Xia, 2000), there are seven different types of ECs in the CTB. Below is a brief description of the empty categories:

1. ***pro***: small pro, used to represent dropped pronouns.

2. ***PRO***: big PRO, used to represent shared elements in control structures or elements that have generic references.
3. ***OP***: null operator, used to represent empty relative pronouns.
4. ***T***: trace left by movement such as topicalization and relativization.
5. ***RNR***: right node raising.
6. *: trace left by passivization and raising.
7. ***?***: missing elements of unknown category.

An example parse tree with ECs is shown in Figure 1. In the example, there are two ECs, an empty relative pronoun (*OP*) and a trace (*T*), a common syntactic pattern for relative clauses in the CTB.



"Shanghai Pudong recently enacted 71 regulatory documents involving the enconomic field."

Figure 1: Empty categories in a phrase structure tree

2.2 Converting phrase structure to dependency structure

We convert the phrase structure parses in the CTB to dependency trees using the conversion tool that generated the Chinese data sets for the CoNLL 2009 Shared Task on multilingual dependency parsing and semantic role labeling (Hajič et al., 2009)[1]. While the Chinese data of CoNLL 2009 Shared Task does not include ECs, the tool has an option of preserving the ECs in the conversion process. As an example, the dependency tree in Figure 2 is converted from the phrase structure tree in Figure 1, with the ECs preserved.

[1]The tool can be downloaded at http://www.cs.brandeis.edu/ clp/ctb/ctb.html.

In previous work EC detection has been formulated as a classification problem with the target of the classification being word tokens (Yang and Xue, 2010; Chung and Gildea, 2010), or constituents in a parse tree (Gabbard et al., 2006). When word tokens are used as the target of classification, the task is to determine whether there is an EC before each word token, and what type EC it is. One shortcoming with that representation is that more than one EC can precede the same word token, as is the case in the example in Figure 1, where both *OP* and *T* precede 涉及 ("involve"). In fact, (Yang and Xue, 2010) takes the last EC when there is a sequence of ECs and as a result, some ECs will never get the chance to be detected. Notice that this problem can be avoided in a dependency structure representation if we make the target of classification a tuple that consists of the following word token and the head of the EC. From Figure 2, it should be clear that while *OP* and *T* both precede the same word token 涉及 ("involve"), they have different heads, which are 的 (DE) and 涉及 respectively.

Dependency-based EC detection also has other nice properties. For ECs that are arguments of their verbal head, when they are resolved to some overt element, the dependency between the referent of the EC and its head will be naturally established. This can be viewed as an alternative to the approach adopted by Levy and Manning (2004), where phrase structure parses are augmented to recover non-local dependencies. Dependency structures are also easily decomposable into head/dependency pairs and this makes the evaluation more straightforward. Each classification instance can be evaluated independently of other parts of the dependency structure.

2.3 One pass vs two passes

With pairs of tokens (h, t) as the classification target, all possible pairs in a sentence will have to be considered and there will be a large number of (h, t) tuples that are not associated with an EC, leading to a highly imbalanced data set. One can conceive a two-pass scenario where we first make a binary decision of whether there is an empty category associated with the head in the first pass and then determine whether there is an EC associated with the tuple as well as the EC type in the second pass. The alternative is to have a one-pass model in which we

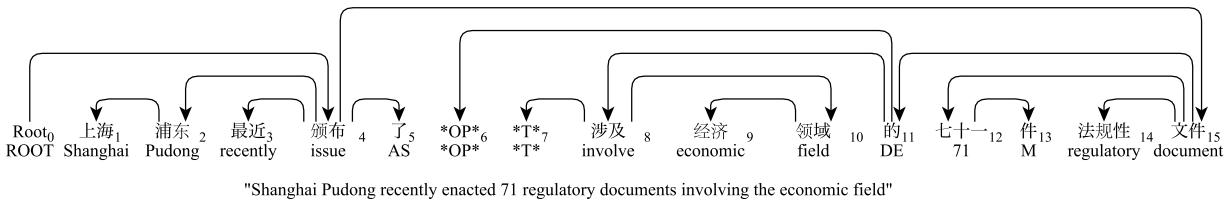


Figure 2: Empty categories in a dependency structure tree

add a NONE category indicating there is no EC associated with the tuple. With the seven EC types presented earlier in this section, this will be an eight-way classification problem. There are reasons for either model: the one-pass model is simpler but in the two-pass model we can bring different sources of information to bear on each sub-problem. Ultimately which model leads to better accuracy is an empirical question. We experimented with both models and it turned out that they led to very similar results. In this paper, we report results from the simpler one-pass model.

3 Features

We explored a wide range of features, all derived from the phrase structure parse tree (T). With each classification instance being a tuple (h, t), the "pivots" for these features are h the head, t the word token following the EC, and p, the word token preceding the EC. The features we tried fall into six broad groups that are all empirically confirmed to have made a positive contribution to our classification task. These are (i) horizontal features, (ii) vertical features, (iii) targeted grammatical constructions, (iv) head information, (v) transitivity features, and (vi) semantic role features. We obviously have looked at features used in previous work on Chinese EC detection, most notably (Yang and Xue, 2010), which has also adopted a classification-based approach, but because we frame our classification task very differently, we have to use very different features. However, there is a subset of features we used here that has at least a partial overlap with their features, and such features are clearly indicated with $*$.

3.1 Horizontal features

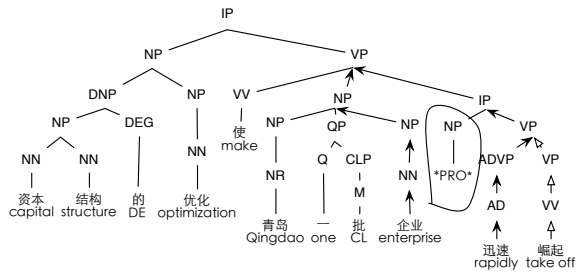
The first group of features we use can be described as horizontal features that exploit lexical context of the head (h), the word token following the EC (t), and the word token before the EC (p) . These include different combinations of h, t and p, as well as their parts-of-speech. They also include various linear distance features between h and t. Below is the full list of lexical features:

1. $*$The token string representation of h, t and p, as well as their part-of-speech tag (POS).
2. $*$The POS combination of h and t, the POS combination of t and p.
3. The normalized word distance between h and t, with the values of this feature being $same$, $immediately\ before$, $immediately\ after$, $near\ before$, and $near\ after$, and $other$.
4. The verb distance between h and t, defined as the number of verbs that occur between h and t.
5. The comma distance between h and t, defined as the number of commas that occur between h and t.

3.2 Vertical features

Vertical features are designed to exploit the hierarchical structure of the syntactic tree. Our hierarchical features are based on the following observations. An empty category is always located between its *left frontier* and *right frontier*, anchored by t and p. Given the lowest common ancestor A of p and t, the right frontier is the path from t to A and the left frontier is the path from the p to A. We also define a path feature from h to t, which constrains the distance between the EC and its head, just as it constrains the distance between a predicate and its argument in the semantic role labeling task (Gildea and Jurafsky, 2002). Given the lowest common ancestor A' of h and t, the path from h to t is the path from h to A' and from A' to t.

In Figure 3, assuming that t is 迅速 ("rapidly") and h is 崛起 ("take off"), the vertical features ex-



"The optimization of the capital structure has led to the rapid take-off of a host of enterprises in Qingdao."

Figure 3: Empty category on the right frontier

tracted include:

1. The string representation of the right frontier, AD↑ADVP↑VP↑IP↑VP
2. The path from the head t to h, AD↑ADVP↑VP↓VP↓VV
3. The path from the head h to A, VV↑VP↑VP↑IP↑VP. Notice there is not always a path from h to A.

The vertical features are really a condensed representation of a certain syntactic configuration that helps to predict the presence or absence of an empty category as well as the empty category type. For example, the right frontier of **PRO** in Figure 3 AD↑ADVP↑VP↑IP↑VP represents a subjectless IP. Had there been an overt subject in the place of the **PRO**, the right frontier would have been AD↑ADVP↑VP↑IP. Therefore, the vertical features are discriminative features that can help detect the presence or absence of an empty category.

3.3 Targeted grammatical constructions

The third group of features target specific, linguistically motivated grammatical constructions. The majority of features in this group hinge on the immediate IP (roughly corresponds to S in the PTB) ancestor of t headed by h. These features are only invoked when t starts (or is on the left edge of) the immediate IP ancestor, and they are designed to capture the context in which the IP ancestor is located. This context can provide discriminative clues that may help identify the types of empty category. For example, both **pro**s and **PRO**s tend to occur in the subject position of an IP, but the larger context of the IP often determines the exact empty category type. In Figure 3, the IP that has a **PRO** subject is the complement of a verb in a canonical object-control construction. An IP can also be a sentential subject, the complement of a preposition or a localizer (also called postposition in the literature), or the complement in a CP (roughly SBAR in the PTB), etc. These different contexts tend to be associated with different types of empty categories. The full list of features that exploit these contexts include:

1. $*$Whether t starts an IP
2. $*$Whether t starts a subjectless IP
3. The left sisters of the immediate IP parent that t starts
4. The right sisters of the immediate IP parent that t starts
5. The string representation of the governing verb of the immediate IP parent that t starts
6. Whether the IP started by t is the complement of a localizer phrase
7. Whether the immediate IP parent that t starts is a sentential subject

3.4 Head information

Most ECs have a verb as its head, but when there is a coordination VP structure where more than one VP share an EC subject, only one such verb can be the head of this EC. The phrase structure to dependency structure conversion tool designates the first verb as the head of the coordinated VP and thus the head of the EC subject in the dependency structure. Other verbs have no chance of being the head. We use a VP head feature to capture this information. It is a binary feature indicating whether a verb can be a head.

3.5 Transitivity features

A transitivity lexicon has been extracted from the Chinese Treebank and it is used to determine the transitivity value of a word. A word can be $transitive$, $intransitive$, or $unknown$ if it is not a verb. Ditransitive verbs are small in number and are folded into transitive verbs. Transitivity features are defined on h and constrained by word distance: it is only used when h immediately precedes t. This feature category is intended to capture transitive verbs that are missing an object.

3.6 Semantic role features

There are apparent connections between semantic role labeling and EC detection. The task of semantic role labeling is typically defined as one of detecting and classifying arguments for verbal or nominal predicates, with more work done so far on verbal than nominal predicates. Although empty categories are annotated as arguments to verbal predicates in linguistic resources such as the English (Palmer et al., 2005) and Chinese (Xue and Palmer, 2009) Propbanks, they are often left out in semantic role labeling systems trained on these resources. This is because the best performing semantic role labeling systems rely on syntactic features extracted from automatic parses (Gildea and Palmer, 2002; Punyakanok et al., 2005) and the parsers that produce them do not generally reproduce empty categories. As a result, current semantic role labeling systems can only recover explicit arguments. However, assuming that all the explicit arguments to a predicate are detected and classified, one can infer the empty arguments of a predicate from its explicit arguments, given a list of $expected$ arguments for the predicate. The list of expected arguments can be found in the "frame files" that are used to guide probank annotation. We defined a semantic role feature category on h when it is a verb and the value of this feature is the semantic role labels for the EC arguments. Like transitivity features, this feature category is also constrained by word distance. It is only used when h immediately precedes t.

To extract semantic role features, we retrained a Chinese semantic role labeling system on the Chinese Propbank. We divided the Chinese Propbank data into 10 different subsets, and automatically assigned semantic roles to each subset with a system trained on the other nine subsets. Using the frame files for the Chinese Propbank, we are able to infer the semantic roles for the missing arguments and use them as features.

4 Experimental Results

4.1 Experimental setup

Our EC detection models are trained and evaluated on a subset of the Chinese TreeBank 6.0. The training/development/test data split in our experiments is recommended in the CTB documentation. The CTB file IDs for training, development and testing are listed in Table 1. The development data is used for feature selection and tuning, and results are reported on the test set.

Train	Dev	Test
81-325, 400-454, 500-554 590-596, 600-885, 900	41-80	1-40 901-931

Table 1: Data set division.

As discussed in Section 2, the gold standard dependency structure parses are converted from the CTB parse trees, with the ECs preserved. From these gold standard parse trees, we extract triples of (e, h, t) where e is the EC type, h is (the position of) the head of the EC, and t is (the position of) the word token following the EC. During the training phrase, features are extracted from automatic phrase structure parses and paired with these triples. The automatic phrase structure parses are produced by the the Berkeley parser[2] with a 10-fold cross-validation, which each fold parsed using a model trained on the other nine folds. Measured by the ParsEval metric (Black et al., 1991), the parsing accuracy on the CTB test set stands at 83.63% (F-score), with a precision of 85.66% and a recall of 81.69%. We chose to train a Maximum Entropy classifier using the Mallet toolkit[3] (McCallum, 2002) to detect ECs.

4.2 Evaluation metric

We use standard metrics of precision, recall and F-measure in our evaluation. In a dependency structure representation, evaluation is very straightforward because individual arcs from the dependency tree can be easily decomposed. An EC is considered to be correctly detected if it is attached to the correct head h, correctly positioned relative to t, and correctly typed. This is a more stringent measure than metrics proposed in previous work, which evaluates EC detection based on its position and type without considering the head it is a dependent of.

4.3 Results

There are 1,838 total EC instances in the test set, and if we follow (Yang and Xue, 2010) and collapse all

[2] http://code.google.com/p/berkeleyparser
[3] http://mallet.cs.umass.edu

consecutive ECs before the same word token to one, we will end up with a total EC count of 1,352, and this is also the EC count used by (Cai et al., 2011) in their evaluation. On the dependency-based representation adopted here, after collapsing all consecutive ECs before the same word token AND attached to the same head to one, we end up with a total EC count of 1,765. The distribution of the ECs in the test set are presented in Table 2, with the EC count per type from (Yang and Xue, 2010) in parenthesis if it is different. The number of *OP*s, in particular, has increased dramatically from 134 to 527, and this is because a null relative pronoun (*OP*) immediately followed by a trace (*T*) in the subject position of a relative clause is a very common pattern in the Chinese Treebank, as illustrated in Figure 2. In (Yang and Xue, 2010), the *OP*-*T* sequences are collapsed into one, and only the *T*s are counted. That leads to the much smaller count of *OP*s.

type	count	type	count
pro	298 (290)	*PRO*	305 (299)
OP	527 (134)	*T*	584 (578)
*	19	*RNR*	32
?	0	total	(1352)/1765/(1838)

Table 2: EC distribution in the CTB test set

Our results are shown in Table 3. These results are achieved by using the full feature set presented in Section 3. The overall accuracy by F1-measure is 0.574 if we assume there can only be one EC associated with a given (h, t) tuple and hence the total EC count in the gold standard is 1,765, or 0.561 if we factor in all the EC instances and use the higher total count of 1,838, which lowers the recall. If instead we use the total EC count of 1,352 that was used in previous work (Yang and Xue, 2010; Cai et al., 2011), then the F1-measure is 0.660 because the lower total count greatly improves the recall. This is a significant improvement over the best previous result reported by Cai et al (2011), which is an F1 measure of 0.586 on the same test set but based on a less stringent metric of just comparing the EC position and type, without considering whether the EC is attached to the correct head.

There are several observations worth noting from these results. One is that our method performs particularly well on null relative pronouns (*OP*) and

class	correct	prec	rec	F1
pro	46	.397	.154	.222
PRO	162	.602	.531	.564
OP	344	.724	.653	.687
T	331	.673	.567	.615
*	0	0	0	0
RNR	20	.714	.625	.667
all	903	.653	.512 (.491) (.668)	.574 (.561) (.660)
CCG		.660	.545	.586

Table 3: EC detection results on the CTB test set and comparison with (Cai et al., 2011) [CCG]

traces (*T*), indicating that our features are effective in capturing information from relative clause constructions. This accounts for most of the gain compared with previous approaches. The *OP* category, in particular, benefits most from the dependency representation because it is collapsed to the immediately following *T* in previous approaches and does not even get a chance to be detected. On the other hand, our model did poorly on dropped pronouns (*pro*). One possible explanation is that **pro**s generally occupy subject positions in a sentence and is attached as an immediate child of an IP, which is the top-level structure of a sentence that an automatic parser tends to get wrong. Unlike **PRO**, it is not constrained to well-defined grammatical constructions such as subject- and object-control structures.

To evaluate the effectiveness of our features, we also did an ablation study on the contribution of different feature groups. The most effective features are the ones when taken out lead to the most drop in accuracy. As should be clear from Table 4, the most effective features are the horizontal features, followed by vertical structures. Features extracted from targeted grammatical constructions and features representing whether h is the head of a coordinated VP lead to modest improvement. Transitivity and semantic role features make virtually no difference at all. We believe it is premature to conclude that they are not useful. Possible explanations for their lack of effectiveness is that they are used in very limited context and the accuracy of the semantic role label-

ing system is not sufficient to make a difference.

class	correct	prec	rec	F1
all	903	.653	.512	.574 (.561)
-Horizontal	827	.627	.469	.536 (.524)
-Vertical	865	.652	.490	.559 (.547)
-Gr Cons	887	.646	.483	.565 (.552)
-V head	891	.651	.505	.569 (.556)
-Trans	899	.654	.509	.573 (.560)
-SRL	900	.657	.510	.574 (.561)

Table 4: Contribution of feature groups

5 Related Work

The work reported here follows a fruitful line of research on EC detection and resolution, mostly in English. Empty categories have initially been left behind in research on syntactic parsing (Collins, 1999; Charniak, 2001) for efficiency reasons, but more recent work has shown that EC detection can be effectively integrated into the parsing process (Schmid, 2006; Cai et al., 2011). In the meantime, both pre-processing and post-processing approaches have been explored in previous work as alternatives. Johnson (2002) has showed that empty categories can be added to the skeletal parses with reasonable accuracy with a simple pattern-matching algorithm in a postprocessing step. Dienes and Dubey (2003b; 2003a) achieved generally superior accuracy using a machine learning framework without having to refer to the syntactic structure in the skeletal parses. They described their approach as a pre-processing step for parsing because they only use as features morpho-syntactic clues (passives, gerunds and to-infinitives) that can be found in certain function words and part-of-speech tags. Even better results, however, were obtained by Campbell (2004) in a postprocessing step that makes use of rules inspired by work in theoretical linguistics. Gabbard et al (2006) reported further improvement largely by recasting the Campbell rules as features to seven different machine learning classifiers.

We adopted a machine-learning based postprocessing approach based on insights gained from prior work in English and on Chinese-specific considerations. All things being equal, we believe that a machine learning approach that can exploit partial information is more likely to succeed than deterministic rules that have to make reference to morpho-syntactic clues such as to-infinitives and gerunds that are largely non-existent in Chinese. Without these clues, we believe a preprocessing approach that does not take advantage of skeletal parses is unlikely to succeed either. The work we report here also builds on emerging work in Chinese EC detection. Yang and Xue (2010) reported work on detecting just the presence and absence of empty categories without further classifying them. Chung and Gildea (2010) reported work on just detecting just a small subset of the empty categories posited in the Chinese TreeBank. Kong and Zhou (2010) worked on Chinese zero anaphora resolution, where empty category detection is a subtask. More recently, Cai et al (2011) has successfully integrated EC detection into phrase-structure based syntactic parsing and reported state-of-the-art results in both English and Chinese.

6 Conclusions and Future Work

We described a novel approach to detecting empty categories (EC) represented in dependency trees and a new metric for measuring EC detection accuracy. The new metric takes into account not only the position and type of an EC, but also the head it is a dependent of in a dependency structure. We also proposed new features that are more suited for this new approach. Tested on a subset of the Chinese Treebank, we show that our system improved significantly over the best previously reported results despite using a more stringent evaluation metric, with most of the gain coming from an improved representation. In the future, we intend to work toward resolving ECs to their antecedents when EC detection can be done with adequate accuracy. We also plan to test our approach on the Penn (English) Treebank, with the first step being converting the Penn Treebank to a dependency representation with the ECs preserved.

Acknowledgments

This work is supported by the National Science Foundation via Grant No. 0910532 entitled "Richer Representations for Machine Translation". All views expressed in this paper are those of the authors and do not necessarily represent the

view of the National Science Foundation.

References

E. Black, S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. A procedure for quantitively comparing the syntactic coverage of English grammars. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 306–311.

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*, pages 212–216, Portland, Oregon, USA, June. Association for Computational Linguistics.

Richard Campbell. 2004. Using linguistic principles to recover empty categories. In *Proceedings of the 42nd Annual Meeting on Association For Computational Linguistics*.

E. Charniak. 2001. Immediate-head Parsing for Language Models. In *ACL-01*.

Tagyoung Chung and Daniel Gildea. 2010. Effects of empty categories on machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 636–645, Cambridge, MA.

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Péter Dienes and Amit Dubey. 2003a. Antecendant Recovery: Experiments with a Trace Tagger. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* , Sapporo, Japan.

Péter Dienes and Amit Dubey. 2003b. Deep syntactic processing by combining shallow methods. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.

Ryan Gabbard, Seth Kulick, and Mitchell Marcus. 2006. Fully parsing the penn treebank. In *Proceedings of HLT-NAACL 2006*, pages 184–191, New York City.

D. Gildea and D. Jurafsky. 2002. Automatic labeling for semantic roles. *Computational Linguistics*, 28(3):245–288.

Dan Gildea and Martha Palmer. 2002. The Necessity of Parsing for Predicate Argument Recognition. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics*, Philadelphia, PA.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

James C.T. Huang. 1984. On the distribution and reference of empty pronouns. *Linguistics Inquiry*, 15:531–574.

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

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*, MIT, Massachusetts.

Roger Levy and Christopher Manning. 2004. Deep dependencies from context-free statistical parsers: correcting the surface dependency approximation. In *Proceedings of the ACL*.

M. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Vasin Punyakanok, Dan Roth, and W. Yih. 2005. The Necessity of Syntactic Parsing for Semantic Role Labeling. In *Proceedings of IJCAI-2005*, pages 1124–1129, Edinburgh, UK.

Helmut Schmid. 2006. Trace prediction and recovery with unlexicalized PCFGs and slash features. In *Proc of ACL*.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Nianwen Xue and Fei Xia. 2000. The Bracketing Guidelines for Penn Chinese Treebank Project. Technical Report IRCS 00-08, University of Pennsylvania.

Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207–238.

Yaqin Yang and Nianwen Xue. 2010. Chasing the Ghost: Recovering Empty Categories in the Chinese Tree-

bank. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, Beijing, China.