Chasing the ghost: recovering empty categories in the Chinese Treebank

Yaqin Yang
Computer Science Department
Brandeis University
yaqin@cs.brandeis.edu

Nianwen Xue
Computer Science Department
Brandeis University
xuen@cs.brandeis.edu

Abstract

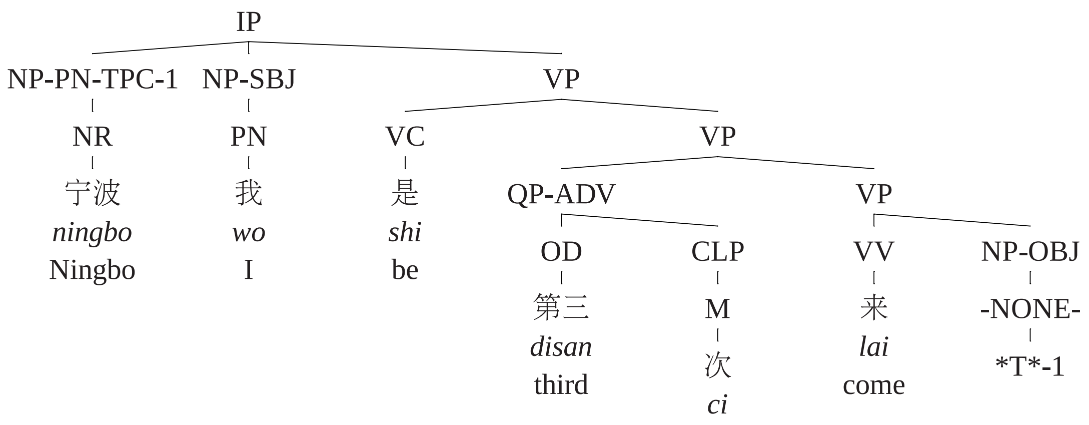
Empty categories represent an important source of information in syntactic parses annotated in the generative linguistic tradition, but empty category recovery has only started to receive serious attention until very recently, after substantial progress in statistical parsing. This paper describes a unified framework in recovering empty categories in the Chinese Treebank. Our results show that given skeletal gold standard parses, the empty categories can be detected with very high accuracy. We report very promising results for empty category recovery for automatic parses as well.

1 Introduction

The use of empty categories to represent the syntactic structure of a sentence is the hallmark of the generative linguistics and they represent an important source of information in treebanks annotated in this linguistic tradition. The use of empty categories in the annotation of treebanks started with the Penn Treebank (Marcus et al., 1993), and this practice is continued in the Chinese Treebank (CTB) (Xue et al., 2005) and the Arabic Treebank, the Penn series of treebanks. Empty categories come in a few different varieties, serving different purposes. One use of empty categories is to mark the extraction site of an dislocated phrase, thus effectively reconstructing the canonical structure of a sentence, allowing easy extraction of its predicate-argument structure. For example, in Figure 1, the empty category ***T*-1** is coindexed with the dislocated topic NP 宁波 ("Ningbo"), indicating that the canonical position of this NP is next to the verb 来 ("come"). The empty category effectively localizes the syntactic dependency between the verb and this NP, making it easier to detect and extract this relation.

Marking the extraction site of a dislocated item is not the only use of empty categories. For languages like Chinese, empty categories are also used to represent dropped pronouns. Chinese is a pro-drop language (Huang, 1989) and subject pronouns are routinely dropped. Recovering these elliptical elements is important to many natural language applications. When translated into another language, for example, these dropped pronouns may have to be made explicit and replaced with overt pronouns or noun phrases if the target language does not allow dropped pronouns.

Although empty categories have been an integral part of the syntactic representation of a sentence ever since the Penn Treebank was first constructed, it is only recently that they are starting to receive the attention they deserve. Works on automatic detection of empty categories started to emerge (Johnson, 2002; Dienes and Dubey, 2003; Campbell, 2004; Gabbard et al., 2006) after substantial progress has been made in statistical syntactic parsing. This progress has been achieved after over a decade of intensive research on syntactic parsing that has essentially left the empty categories behind (Collins, 1999; Charniak, 2000). Empty categories were and still are routinely pruned out in parser evaluations (Black et al., 1991). They have been excluded from the parser development and evaluation cycle not so much because their importance was not understood, but because researchers haven't figured out



"Ningbo, this is the third time I came here."

Figure 1: A CTB tree with empty categories

a way to incorporate the empty category detection in the parsing process. In fact, the detection of empty categories relies heavily on the other components of the syntactic representation, and as a result, empty category recovery is often formulated as postprocessing problem after the skeletal structure of a syntactic parse has been determined. As work on English has demonstrated, empty category detection can be performed with high accuracy given high-quality skeletal syntactic parses as input.

Because Chinese allows dropped pronouns and thus has more varieties of empty categories than languages like English, it can be argued that there is added importance in Chinese empty category detection. However, to our knowledge, there has been little work in this area, and the work we report here represents the first effort in Chinese empty category detection. Our results are promising, but they also show that Chinese empty category detection is a very challenging problem mostly because Chinese syntactic parsing is difficult and still lags significantly behind the state of the art in English parsing. We show that given skeletal gold-standard parses (with empty categories pruned out), the empty detection can be performed with a fairly high accuracy of almost 89%. The performance drops significantly, to 63%, when the output of an automatic parser is used.

The rest of the paper is organized as follows. In Section 2, we formulate the empty category detection as a binary classification problem where each word is labeled as either having a empty category before it or not. This makes it possible to use any standard machine learning technique to solve this problem. The key is to find the appropriate set of features. Section 3 describes the features we use in our experiments. We present our experimental results in Section 4. There are two experimental conditions, one with gold standard treebank parses (stripped of empty categories) as input and the other with automatic parses. Section 5 describes related work and Section 6 conclude our paper.

2 Formulating the empty category detection as a tagging problem

In the CTB, empty categories are marked in a parse tree which represents the hierarchical structure of a sentence, as illustrated in Figure 1. There are eight types of empty categories annotated in the CTB, and they are listed in Table 1. Among them, *pro* and *PRO* are used to represent nominal empty categories, *T* and *NP* are used to represent traces of dislocated items, *OP* is used to represent empty relative pronouns in relative clauses, and *RNR* is used to represent pseudo attachment. The reader is referred to the CTB bracketing manual (Xue and Xia, 2000) for detailed descriptions and examples. As can be seen from Table 1, the distribution of these empty categories is very uneven, and many of these empty categories do not occur very often.

EC Type	count	Description
pro	2024	small pro
PRO	2856	big pro
T	4486	trace for extraction
RNR	217	right node raising
OP	879	operator
*	132	trace for raising

Table 1: Empty categories in CTB.

As a first step of learning an empty category model, we treat all the empty categories as a unified type, and for each word in the sentence, we only try to decide if there is an empty category before it. This amounts to an empty category detection task, and the objective is to first locate the empty categories without attempting to determine the specific empty category type. Instead of predicting the locations of the empty categories in a parse tree and having a separate classifier for each syntactic construction where an empty category is likely to occur, we adopt a linear view of the parse tree and treat empty categories, along with overt word tokens, as leaves in the tree. This allows us to identify the location of the empty categories in relation to overt word tokens in the same sentence, as illustrated in Example (1):

(1) 宁波 我 是 第三 次 来 *T* 。

In this representation, the position of the empty category can be defined either in relation to the previous or the next word, or both. To make this even more amenable to machine learning approaches, we further reformulate the problem as a tagging problem so that each overt word is labeled either with EC, indicating there is an empty category *before* this word, or NEC, indicating there is no empty category. This reformulated representation is illustrated in Example (2):

(2) 宁 波/NEC 我/NEC 是/NEC 第 三/NEC 次/NEC 来/NEC 。/EC

In (2), the EC label attached to the final period indicates that there is an empty category before this punctuation mark. There is a small price to pay with this representation: when there is more than one empty category before a word, it is indistinguishable from cases where there is only one empty category. What we have gained is a simple unified representation for all empty categories that lend itself naturally to machine learning approaches. Another advantage is that for natural language applications that do not need the full parse trees but only need the empty categories, this representation provides an easy-to-use representation for those applications. Since this linearlized representation is still aligned with its parse tree, we still have easy access to the full hierarchical structure of this tree from which useful features can be extracted.

3 Features

Having modeled empty category detection as a machine learning task, feature selection is crucial to successfully finding a solution to this problem. The machine learning algorithm scans the words in a sentence from left to right one by one and determine if there is an empty category before it. When the sentence is paired with its parse tree, the feature space is all the surrounding words of the target word as well as the syntactic parse for the sentence. The machine learning algorithm also has access to the empty category labels (EC or NEC) of all the words before the current word. Figure 2 illustrates the feature space for the last word (a period) in the sentence.

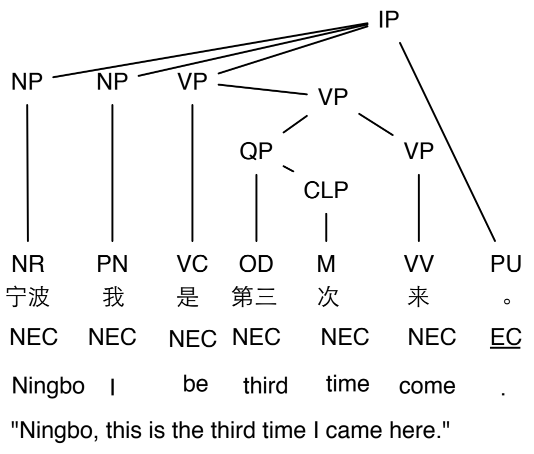

IP
NP
NP
VP
VP
QP
CLP
VP
NR PN VC OD M VV PU
宁波 我 是 第三 次 来 。
NEC NEC NEC NEC NEC NEC EC
Ningbo I be third time come .
"Ningbo, this is the third time I came here."

Figure 2: Feature space of empty category detection

For purposes of presentation, we divide our features into lexical and syntactic features. The lexical features are different combinations of the words and their parts of speech (POS), while syntactic features are the structural information gathered from the nonterminal phrasal labels and their syntactic relations.

3.1 Lexical features

The lexical features are collected from a narrow window of five words and their POS tags. If the target word is a verb, the lexical features also include transitivity information of this verb, which is gathered from the CTB. A transitivity lexicon is induced from the CTB by checking whether a verb has a right NP or IP sibling. Each time a verb is used as a transitive verb (having a right NP or IP sibling), its transitive count is incremented by one. Conversely, each time a verb is used as an intransitive verb (not having a right NP or IP sibling), its intransitive use is incremented by one. The resulting transitivity lexicon after running through the entire Chinese Treebank consists of a list of verbs with frequencies of their transitive and intransitive uses. A verb is considered to be transitive if its intransitive count in this lexicon is zero or if its transitive use is more than three times as frequent as its intransitive use. Similarly, a verb is considered to be intransitive if its transitive count is zero or if its intransitive use is at least three times as frequent as its transitive use. The full list of lexical features is presented in Table 2.

Feature Names	Description
word(0)	Current word
word(-1)	Previous word
pos(0)	POS of current word
pos(-1,0)	POS of previous and current word
pos(0, 1)	POS of current and next word
pos(0, 1, 2)	POS of current & next word, & word 2 after
pos(-2, -1)	POS of previous word & word 2 before
word(-1), pos(0)	Previous word & POS of current word
pos(-1),word(0)	POS of previous word& current word
trans(0)	current word is transitive or intransitive verb
prep(0)	true if POS of current word is a preposition

Table 2: Feature set.

3.2 Syntactic features

Syntactic features are gathered from the CTB parses stripped of function tags and empty categories when the gold standard trees are used as input. The automatic parses used as input to our system are produced by the Berkeley parser. Like most parsers, the Berkeley parser does not reproduce the function tags and empty categories in the original trees in the CTB. Syntactic features capture the syntactic context of the target word, and as we shall show in Section 4, the syntactic features are crucial to the success of empty category detection. The list of syntactic features we use in our system include:

1. **1st-IP-child**: True if the current word is the first word in the lowest IP dominating this word.
2. **1st-word-in-subjectless-IP**: True if the current word starts an IP with no subject. Subject is detected heuristically by looking at left sisters of a VP node. Figure 3 illustrates this feature for the first word in a sentence where the subject is a dropped pronoun.
3. **1st-word-in-subjectless-IP+POS**: POS of the current word if it starts an IP with no subject.
4. **1st-VP-child-after-PU**: True if the current word is the first terminal child of a VP following a punctuation mark.
5. **NT-in-IP**: True if POS of current word is NT, and it heads an NP that does not have a subject NP as its right sister.
6. **verb-in-NP/VP**: True if the current word is a verb in an NP/VP.
7. **parent-label**: Phrasal label of the parent of the current node, with the current node always corresponding to a terminal node in the parse tree.
8. **has-no-object**: True If the previous word is a transitive verb and this verb does not take an object.

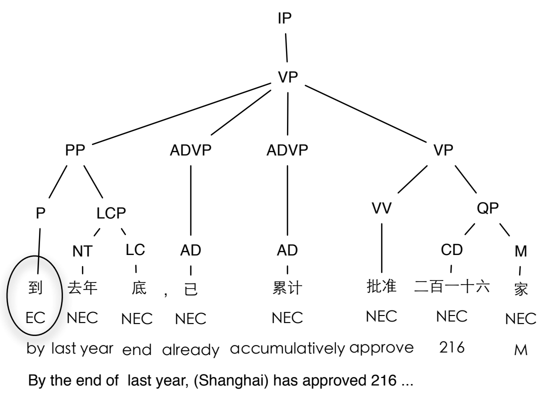


Figure 3: First word in a subject-less IP

Empty categories generally occur in clausal or phrasal boundaries, and most of the features are designed to capture such information. For example, the five feature types, *1st-IP-child*, *1st-word-in-subjectless-IP*, *1st-word-in-subjectless-IP*, *1st-VP-child-after-PU* and *NT-in-IP* all represent the left edge of a clause (IP) with some level of granularity. *parent label* and *verb-in-NP/VP* represent phrases within which empty categories typically occur do not occur. The *has-no-object* feature is intended to capture transitive uses of a verb when the object is missing.

4 Experiments

Given that our approach is independent of specific machine learning techniques, many standard machine learning algorithms can be applied to this task. For our experiment we built a Maximum Entropy classifier with the Mallet toolkit[1].

4.1 Data

In our experiments, we use a subset of the CTB 6.0. This subset is further divided into training (files chtb_0081 thorough chtb_0900), development (files chtb_0041 through chtb_0080) and test sets (files chtb_0001 through chtb_0040, files chtb_0901 through chtb_0931). The reason for not using the entire Chinese Treebank is that the data in the CTB is from a variety of different sources and the automatic parsing accuracy is very uneven across these different sources.

[1]http://mallet.cs.umass.edu

4.2 Experimental conditions

Two different kinds of data sets were used in the evaluation of our method: 1) gold standard parse trees from the CTB; and 2) automatic parses produced by the Berkeley parser[2] .

4.2.1 Gold standard parses

There are two experimental conditions. In our first experiment, we use the gold standard parse trees from the CTB as input to our classifier. The version of the parse tree that we use as input to our classifier is stripped of the empty category information. What our system effectively does is to restore the empty categories given a skeletal syntactic parse. The purpose of this experiment is to establish a topline and see how accurately the empty categories can be restored given a “correct” parse.

4.2.2 Automatic parses

To be used in realistic scenarios, the parse trees need to be produced automatically from raw text using an automatic parser. In our experiments we use the Berkeley Parser as a representative of the state-of-the-art automatic parsers. The input to the Berkeley parser is words that have already been segmented in the CTB. Obviously, to achieve fully automatic parsing, the raw text should be automatically segmented as well. The Berkeley parser comes with a fully trained model, and to make sure that none of our test and development data is included in the training data in the original model, we retrained the parser with our training set and used the resulting model to parse the documents in the development and test sets.

When training our empty category model using automatic parses, it is important that the quality of the parses match between the training and test sets. So the automatic parses in the training set are acquired by first training the parser with 4/5 of the data and using the resulting model to parse the remaining 1/5 of the data that has been held out. Measured by the ParsEval metric (Black et al., 1991), the parser accuracy stands at 80.3% (F-score), with a precision of 81.8% and a recall of 78.8% (recall).

[2]http://code.google.com/p/berkeleyparser

4.3 Evaluation metrics

We use precision, recall and F-measure as our evaluation metrics for empty category detection. Precision is defined as the number of correctly identified Empty Categories (ECs) divided by the total number of ECs that our system produced. Recall is defined as the number of correctly identified ECs divided by the total number of EC labels in the CTB gold standard data. F-measure is defined as the geometric mean of precision and recall.

$$R = \frac{\text{\# of correctly detected EC}}{\text{\# of EC tagged in corpus}} \tag{1}$$

$$P = \frac{\text{\# of correctly detected EC}}{\text{\# of EC reported by the system}} \tag{2}$$

$$F = \frac{2}{1/R + 1/P} \tag{3}$$

4.4 Overall EC detection performance

We report our best result for the gold standard trees and the automatic parses produced by the Berkeley parser in Table 3. These results are achieved by using all lexical and syntactic features presented in Section 3.

Data	Prec.(%)	Rec.(%)	F(%)
Gold	95.9 (75.3)	83.0 (70.5)	89.0 (72.8)
Auto	80.3 (57.9)	52.1 (50.2)	63.2 (53.8)

Table 3: Best results on the gold tree.

As shown in Table 3, our feature set works well for the gold standard trees. Not surprisingly, the accuracy when using the automatic parses is lower, with the performance gap between using the gold standard trees and the Berkeley parser at 25.8% (F-score). When the automatic parser is used, although the precision is 80.3%, the recall is only 52.1%. As there is no similar work in Chinese empty category detection using the same data set, for comparison purposes we established a baseline using a rule-based approach. The rule-based algorithm captures two most frequent locations of empty categories: the subject and the object positions. Our algorithm labels the first word within a VP with EC if the VP does not have a subject NP. Similarly, it assigns the EC label to the word immediately following a transitive verb if it does not have an NP or IP object. Since the missing subjects and objects account for most of the empty categories in Chinese, this baseline covers most of the empty categories. The baseline results are also presented in Table 3 (in brackets). The baseline results using the gold standard trees are 75.3% (precision), 70.5% (recall), and 72.8% (F-score). Using the automatic parses, the results are 57.9% (precision), 50.2% (recall), and 53.8% (F-score) respectively. It is clear from our results that our machine learning model beats the rule-based baseline by a comfortable margin in both experimental conditions. Table 4 breaks down our results by empty category types. Notice that we did not attempt to predict the specific empty category type. This only shows the percentage of empty categories our model is able to recover (recall) for each type. As our model does not predict the specific empty category type, only whether there is an empty category before a particular word, we cannot compute the precision for each empty category type. Nevertheless, this breakdown gives us a sense of which empty category is easier to recover. For both experimental conditions, the empty category that can be recovered with the highest accuracy is ***PRO***, an empty category often used in subject/object control constructions. ***pro*** seems to be the category that is most affected by parsing accuracy. It has the widest gap between the two experimental conditions, at more than 50%.

EC Type	Total	Correct	Recall(%)
pro	290	274/125	94.5/43.1
PRO	299	298/196	99.7/65.6
T	578	466/338	80.6/58.5
RNR	32	22/20	68.8/62.5
OP	134	53/20	40.0/14.9
*	19	9/5	47.4/26.3

Table 4: Results of different types of empty categories.

4.5 Comparison of feature types

To investigate the relative importance of lexical and syntactic features, we experimented with using just the lexical or syntactic features under both experimental conditions. The results are presented in Table 5. Our results show that when using only the lexical features, the drop in accuracy is small when automatic parses are used in place of gold standard trees. However, when using only the syntactic features, the drop in accuracy is much more dramatic. In both experimental conditions, however, syntactic features are more effective than the lexical features, indicating the crucial importance of high-quality parses to successful empty category detection. This makes intuitive sense, given that all empty categories occupy clausal and phrasal boundaries that can only defined in syntactic terms.

Data	Prec.(%)	Rec.(%)	F(%)
Lexical	79.7/77.3	47.6/39.9	59.6/52.7
Syntactic	95.9/78.0	70.0/44.5	81.0/56.7

Table 5: Comparison of lexical and syntactic features.

4.6 Comparison of individual features

Given the importance of syntactic features, we conducted an experiment trying to evaluate the impact of each individual syntactic feature on the overall empty category detection performance. In this experiment, we kept the lexical feature set constant, and switched off the syntactic features one at a time. The performance of the different syntactic features is shown in Table 6. The results here assume that automatic parses are used. The first row is the result of using all features (both syntactic and lexical) while the last row is the result of using only the lexical features. It can be seen that syntactic features contribute more than 10% to the overall accuracy. The results also show that features (e.g., *1st-IP-child*) that capture clause boundary information tend to be more discriminative and they occupy the first few rows of a table that sorted based on feature performance.

Feature Name	Prec.(%)	Rec.(%)	F(%)
all	80.3	52.1	63.2
1st-IP-child	79.8	49.2	60.8
1st-VP-child-after-PU	79.7	50.5	61.8
NT-in-IP	79.4	50.8	61.9
1st-word-in-subjectless-IP+Pos	79.5	51.1	62.2
has-no-object	80.0	51.1	62.4
1st-word-in-subjectless-IP	79.4	51.5	62.5
verb-in-NP/VP	79.9	52.0	63.0
parent-label	79.4	52.4	63.1
only lexical	77.3	39.9	52.7

Table 6: Performance for individual syntactic features with automatic parses.

5 Related work

The problem of empty category detection has been studied both in the context of reference resolution and syntactic parsing. In the reference resolution literature, empty category detection manifests itself in the form of zero anaphora (or zero pronoun) detection and resolution. Zero anaphora resolution has been studied as a computational problem for many different languages. For example, (Ferrández and Peral, 2000) describes an algorithm for detecting and resolving zero pronouns in Spanish texts. (Seki et al., 2002) and (Lida et al., 2007) reported work on zero pronoun detection and resolution in Japanese.

Zero anaphora detection and resolution for Chinese has been studied as well. Converse (2006) studied Chinese pronominal anaphora resolution, including zero anaphora resolution, although there is no attempt to automatically detect the zero anaphors in text. Her work only deals with anaphora resolution, assuming the zero anaphors have already been detected. Chinese zero anaphora identification and resolution have been studied in a machine learning frameworking in (Zhao and Ng, 2007) and (Peng and Araki, 2007).

The present work studies empty category recovery as part of the effort to fully parse natural language text and as such our work is not limited to just recovering zero anaphors. We are also interested in other types of empty categories such as traces. Our work is thus more closely related to the work of (Johnson, 2002), (Dienes and Dubey, 2003), (Campbell, 2004) and (Gabbard et al., 2006).

Johnson (2002) describes a pattern-matching algorithm for recovering empty nodes from phrase structure trees. The idea was to extract minimal connected tree fragments that contain an empty node and its antecedent(s), and to match the extracted fragments against an input tree. He evaluated his approach both on Penn Treebank gold standard trees stripped of the empty categories and on the output of the Charniak parser (Charniak, 2000).

(Dienes and Dubey, 2003) describes an empty detection method that is similar to ours in that it treats empty detection as a tagging problem. The difference is that the tagging is done without access to any syntactic information so that the identified empty categories along with word tokens in the sentence can then be fed into a parser. The success of this approach depends on strong local cues such as infinitive markers and participles, which are non-existent in Chinese. Not surprisingly, our model yields low accuracy if only lexical features are used.

Cambell (2004) proposes an algorithm that uses linguistic principles in empty category recovery. He argues that a rule-based approach might perform well for this problem because the locations of the empty categories, at least in English, are inserted by annotators who follow explicit linguistic principles.

Yuqing(2007) extends (Cahill et al., 2004) 's approach for recovering English non-local dependencies and applies it to Chinese. This paper proposes a method based on the Lexical-Functional Grammar f-structures, which differs from our approach. Based on parser output trees including 610 files from the CTB, the authors of this paper claimed they have achieved 64.71% f-score for trace insertion and 54.71% for antecedent recovery.

(Gabbard et al., 2006) describes a more recent effort to fully parse the Penn Treebank, recovering both the function tags and the empty categories. Their approach is similar to ours in that they treat empty category recovery as a post-processing process and use a machine learning algorithm that has access to the skeletal information in the parse tree. Their approach is different from ours in that they have different classifiers for different types of empty categories.

Although generally higher accuracies are reported in works on English empty category recovery, cross-linguistic comparison is difficult because both the types of empty categories and the linguistic cues that are accessible to machine learning algorithms are different. For example, there are no empty complementizers annotated in the CTB while English does not allow dropped pronouns.

6 Conclusion and future work

We describe a unified framework to recover empty categories for Chinese given skeletal parse trees as input. In this framework, empty detection is formulated as a tagging problem where each word in the sentence receives a tag indicating whether there is an empty category before it. This advantage of this approach is that it is amenable to learning-based approaches and can be addressed with a variety of machine learning algorithms. Our results based on a Maximum Entropy model show that given skeletal gold standard parses, empty categories can be recovered with very high accuracy (close to 90%). We also report promising results (over 63%). when automatic parses produced by an off-the-shelf parser is used as input.

Detecting empty categories is only the first step towards fully reproducing the syntactic representation in the CTB, and the obvious next step is to also classify these empty categories into different types and wherever applicable, link the empty categories to their antecedent. This is the line of research we intend to pursue in our future work.

Acknowledgment

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

Black, E., 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.

Cahill, Aoife, Michael Burke, Ruth O' Donovan, Josef van Genabith, and Andy Way. 2004. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

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

Charniak, E. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of NAACL-2000*, pages 132–139, Seattle, Washington.

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

Converse, Susan. 2006. *Pronominal anaphora resolution for Chinese*. Ph.D. thesis.

Dienes, Péter and Amit Dubey. 2003. Deep syntactic processing by combining shallow methods. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, volume 1.

Ferrández, Antonio and Jesús Peral. 2000. A computational approach to zero-pronouns in Spanish. In *Proceedings of the 38th Annual Meeting on Association For Computational Linguistics*.

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

Guo, Yuqing, Haifeng Wang, and Josef van Genabith. 2007. Recovering Non-Local Dependencies for Chinese. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Huang, James C.-T. 1989. Pro drop in Chinese, a generalized control approach. In O, Jaeggli and K. Safir, editors, *The Null Subject Parameter*. D. Reidel Dordrecht.

Johnson, Mark. 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*.

Lida, Ryu, Kentaro Inui, and Yuji Matsumoto. 2007. Zero-anaphora resolution by learning rich syntactic pattern features. *ACM Transactions on Asian Language Information Processing*, pages 1–22.

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

Peng, Jing and Kenji Araki. 2007. Zero-anaphora resolution in chinese using maximum entropy. *IEICE - Trans. Inf. Syst.*, E90-D(7):1092–1102.

Seki, Kazuhiro, Atsushi Fujii, and Tetsuya Ishikawa. 2002. A probabilistic method for analyzing Japanese anaphora integrating zero pronoun detection and resolution. In *Proceedings of the 19th international Conference on Computational Linguistics*, volume 1.

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

Xue, Nianwen, 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.

Zhao, Shanheng and Hwee Tou Ng. 2007. Identification and Resolution of Chinese Zero Pronouns: A Machine Learning Approach. In *Proceedings of EMNLP-CoNLL Joint Conference*, Prague, Czech Republic.