Enlisting the Ghost: Modeling Empty Categories for Machine Translation

Bing Xiang
IBM T. J. Watson Research Center
1101 Kitchawan Rd
Yorktown Heights, NY 10598
bxiang@us.ibm.com

Xiaoqiang Luo *
Google Inc.
111 8th Ave
New York, NY 10011
xql@google.com

Bowen Zhou
IBM T. J. Watson Research Center
1101 Kitchawan Rd
Yorktown Heights, NY 10598
zhou@us.ibm.com

Abstract

Empty categories (EC) are artificial elements in Penn Treebanks motivated by the government-binding (GB) theory to explain certain language phenomena such as pro-drop. ECs are ubiquitous in languages like Chinese, but they are tacitly ignored in most machine translation (MT) work because of their elusive nature. In this paper we present a comprehensive treatment of ECs by first recovering them with a structured MaxEnt model with a rich set of syntactic and lexical features, and then incorporating the predicted ECs into a Chinese-to-English machine translation task through multiple approaches, including the extraction of EC-specific sparse features. We show that the recovered empty categories not only improve the word alignment quality, but also lead to significant improvements in a large-scale state-of-the-art syntactic MT system.

1 Introduction

One of the key challenges in statistical machine translation (SMT) is to effectively model inherent differences between the source and the target language. Take the Chinese-English SMT as an example: it is non-trivial to produce correct pronouns on the target side when the source-side pronoun is missing. In addition, the pro-drop problem can also degrade the word alignment quality in the training data. A sentence pair observed in the real data is shown in Figure 1 along with the word alignment obtained from an automatic word aligner, where the English subject pronoun "that" is missing on the Chinese side. Consequently, "that" is incorrectly aligned to the second to the last Chinese word "De", due to their high co-occurrence frequency in the training data. If the dropped pronoun were recovered, "that" would have been aligned with the dropped-pro (cf. Figure 3), which is a much more sensible alignment.

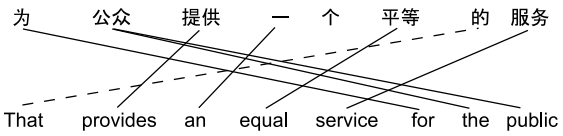

Figure 1: Example of incorrect word alignment due to missing pronouns on the Chinese side.

In order to account for certain language phenomena such as pro-drop and wh-movement, a set of special tokens, called empty categories (EC), are used in Penn Treebanks (Marcus et al., 1993; Bies and Maamouri, 2003; Xue et al., 2005). Since empty categories do not exist in the surface form of a language, they are often deemed elusive and recovering ECs is even figuratively called "chasing the ghost" (Yang and Xue, 2010).

In this work we demonstrate that, with the availability of large-scale EC annotations, it is feasible to predict and recover ECs with high accuracy. More importantly, with various approaches of modeling the recovered ECs in SMT, we are able to achieve significant improvements[1].

The contributions of this paper include the following:

- Propose a novel structured approach to EC prediction, including the exact word-level lo-

* This work was done when the author was with IBM.

[1] Hence "Enlisting the ghost" in the title of this paper.

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 822–831,
Sofia, Bulgaria, August 4-9 2013.

cation and EC labels. Our results are significantly higher in accuracy than that of the state-of-the-art;

- Measure the effect of ECs on automatic word alignment for machine translation after integrating recovered ECs into the MT data;
- Design EC-specific features for phrases and syntactic tree-to-string rules in translation grammar;
- Show significant improvement on top of the state-of-the-art large-scale hierarchical and syntactic machine translation systems.

The rest of the paper is organized as follows. In Section 2, we present a structured approach to EC prediction. In Section 3, we describe the integration of Chinese ECs in MT. The experimental results for both EC prediction and SMT are reported in Section 4. A survey on the related work is conducted in Section 5, and Section 6 summarizes the work and introduces some future work.

2 Chinese Empty Category Prediction

The empty categories in the Chinese Treebank (CTB) include trace markers for A'- and A-movement, dropped pronoun, big PRO etc. A complete list of categories used in CTB is shown in Table 1 along with their intended usages. Readers are referred to the documentation (Xue et al., 2005) of CTB for detailed discussions about the characterization of empty categories.

EC	Meaning
T	trace of A'-movement
*	trace of A-movement
PRO	big PRO in control structures
pro	pro-drop
OP	operator in relative clauses
RNR	for right node raising

Table 1: List of empty categories in the CTB.

In this section, we tackle the problem of recovering Chinese ECs. The problem has been studied before in the literature. For instance, Yang and Xue (2010) attempted to predict the existence of an EC before a word; Luo and Zhao (2011) predicted ECs on parse trees, but the position information of some ECs is partially lost in their representation. Furthermore, Luo and Zhao (2011) conducted experiments on gold parse trees only. In our opinion, recovering ECs from machine parse trees is more meaningful since that is what one would encounter when developing a downstream application such as machine translation. In this paper, we aim to have a more comprehensive treatment of the problem: all EC types along with their locations are predicted, and we will report the results on both human parse trees and machine-generated parse trees.

2.1 Representation of Empty Categories

Our effort of recovering ECs is a two-step process: first, at training time, ECs in the Chinese Treebank are moved and preserved in the portion of the tree structures pertaining to surface words only. Original ECs and their subtrees are then deleted without loss of information; second, a model is trained on transformed trees to predict and recover ECs.

Empty categories heavily depend on syntactic tree structure. For this reason, we choose to project them onto a parse tree node. To facilitate presentation, we first distinguish a *solid* vs. an *empty* non-terminal node. A non-terminal node is *solid* if and only if it contains at least one child node that spans one or more surface words (as opposed to an EC); accordingly, an *empty* node is a non-terminal node that spans only ECs. In the left half of Figure 2, the `NP` node that is the immediate child of `IP` has only one child node spanning an EC – `(-NONE- *pro*)`, and is thus an *empty* node; while all other non-terminal nodes have at least one surface word as their child and are thus all *solid* nodes.

We decide to attach an EC to its lowest *solid* ancestor node. That is, the EC is moved up to the first solid node in the syntactic tree. After ECs are attached, all empty nodes and ECs are deleted from the tree. In order to uniquely recover ECs, we also need to encode the position information. To this end, the relative child index of an EC is affixed to the EC tag. Take the `NP` node spanning the `*pro*` in Figure 2 as an example, the `*pro*` is moved to the lowest solid ancestor, `IP` node, and its position is encoded by `@1` since the deleted `NP` is the second child of the `IP` node (we use 0-based indices). With this transformation, we are able to recover not only the position of an EC, but its type as well. A special tag `NULL` is attached to non-terminal nodes without EC. Since an EC is introduced to express the structure of a sentence, it is a good practice to associate it with the syn-

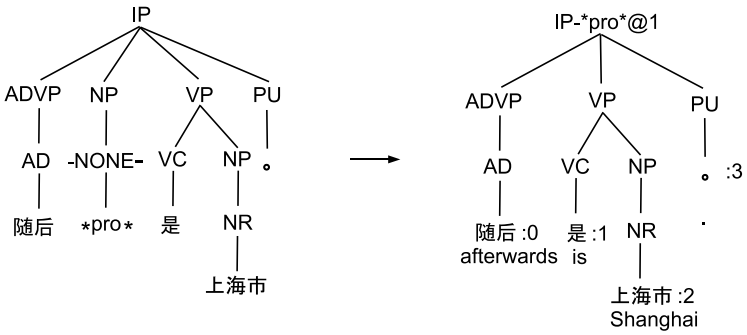

Figure 2: Example of tree transformation on training data to encode an empty category and its position information.

tactic tree, as opposed to simply attaching it to a neighboring word, as was done in (Yang and Xue, 2010). We believe this is one of the reasons why our model has better accuracy than that of (Yang and Xue, 2010) (cf. Table 7).

In summary, a projected tag consists of an EC type (such as `*pro*`) and the EC's position information. The problem of predicting ECs is then cast into predicting an EC tag at each non-terminal node. Notice that the input to such a predictor is a syntactic tree without ECs, e.g., the parse tree on the right hand of Figure 2 without the EC tag `*pro*@1` is such an example.

2.2 A Structured Empty Category Model

We propose a structured MaxEnt model for predicting ECs. Specially, given a syntactic tree, T, whose ECs have been projected onto solid nodes with the procedure described in Section 2.1, we traverse it in post-order (i.e., child nodes are visited recursively first before the current node is visited). Let $T = t_1 t_2 \cdots t_n$ be the sequence of nodes produced by the post-order traversal, and $e_i (i = 1, 2, \cdots, n)$ be the EC tag associated with t_i. The probabilistic model is then:

$$P(e_1^n|T) = \prod_{i=1}^{n} P(e_i|T, e_1^{i-1}) = \prod_{i=1}^{n} \frac{exp\big(\sum_k \lambda_k f_k(e_1^{i-1}, T, e_i)\big)}{Z(e_1^{i-1}, T)} \quad (1)$$

Eq. (1) is the familiar log linear (or MaxEnt) model, where $f_k(e_1^{i-1}, T, e_i)$ is the feature function and $Z(e_1^{i-1}, T) = \sum_{e \in \mathcal{E}} exp\big(\sum_k \lambda_k f_k(e_1^{i-1}, T, e)\big)$ is the normalization factor. $\mathcal{E}$ is the set of ECs to be predicted. In the CTB 7.0 processed by the procedure in Section 2.1, the set consists of 32 EC tags plus a special `NULL` symbol, obtained by modulating the list of ECs in Table 1 with their positions (e.g., `*pro*@1` in Figure 2).

Once the model is chosen, the next step is to decide a set of features $\{f_k(e_1^{i-1}, T, e_i)\}$ to be used in the model. One advantage of having the representation in Section 2.1 is that it is very easy to compute features from tree structures. Indeed, all features used in our system are computed from the syntactic trees, including lexical features.

There are 3 categories of features used in the model: (1) tree label features; (2) lexical features; (3) EC features, and we list them in Table 2. In the feature description column, all node positions (e.g., "left", "right") are relative to the current node being predicted.

Feature 1 to 10 are computed directly from parse trees, and are straightforward. We include up to 2 siblings when computing feature 9 and 10. Feature 11 to 17 are lexical features. Note that we use words at the edge of the current node: feature 11 and 12 are words at the internal boundary of the current node, while feature 13 and 14 are the immediately neighboring word external to the current node. Feature 15 and 17 are from head word information of the current node and the parent node. Feature 18 and 19 are computed from predicted ECs in the past – that's why the model in Eq. (1) conditions on e_1^{i-1}.

Besides the features presented in Table 2, we also use conjunction features between the current node label with the parent node label; the current node label with features computed from child nodes; the current node label with features from left and sibling nodes; the current node label with lexical features.

No.	Tree Label Features
1	current node label
2	parent node label
3	grand-parent node label
4	left-most child label or POS tag
5	right-most child label or POS tag
6	label or POS tag of the head child
7	the number of child nodes
8	one level CFG rule
9	left-sibling label or POS tag
10	right-sibling label or POS tag
	Lexical Features
11	left-most word under the current node
12	right-most word under the current node
13	word immediately left to the span of the current node
14	word immediately right to the span of the current node
15	head word of the current node
16	head word of the parent node
17	is the current node head child of its parent?
	EC Features
18	predicted EC of the left sibling
19	the set of predicted ECs of child nodes

Table 2: List of features.

3 Integrating Empty Categories in Machine Translation

In this section, we explore multiple approaches of utilizing recovered ECs in machine translation.

3.1 Explicit Recovery of ECs in MT

We conducted some initial error analysis on our MT system output and found that most of the errors that are related to ECs are due to the missing `*pro*` and `*PRO*`. This is also consistent with the findings in (Chung and Gildea, 2010). One of the other frequent ECs, `*OP*`, appears in the Chinese relative clauses, which usually have a Chinese word "De" aligned to the target side "that" or "which". And the trace, `*T*`, exists in both Chinese and English sides. For MT we want to focus on the places where there exist mismatches between the source and target languages. A straightforward way of utilizing the recovered `*pro*` and `*PRO*` is to pre-process the MT training and test data by inserting ECs into the original source text (i.e. Chinese in this case). As mentioned in the previous section, the output of our EC predictor is a new parse tree with the labels and positions encoded in the tags. Based on the positional information in the tags, we can move the predicted ECs down to the surface level and insert them between original source words. The same prediction and "pull-down" procedure can be conducted consistently cross the MT training and test data.

3.2 Grammar Extraction on Augmented Data

With the pre-processed MT training corpus, an unsupervised word aligner, such as GIZA++, can be used to generate automatic word alignment, as the first step of a system training pipeline. The effect of inserting ECs is two-fold: first, it can impact the automatic word alignment since now it allows the target-side words, especially the function words, to align to the inserted ECs and fix some errors in the original word alignment; second, new phrases and rules can be extracted from the pre-processed training data. For example, for a hierarchical MT system, some phrase pairs and Hiero (Chiang, 2005) rules can be extracted with recovered `*pro*` and `*PRO*` at the Chinese side.

In this work we also take advantages of the augmented Chinese parse trees (with ECs projected to the surface) and extract tree-to-string grammar (Liu et al., 2006) for a tree-to-string MT system. Due to the recovered ECs in the source parse trees, the tree-to-string grammar extracted from such trees can be more discriminative, with an increased capability of distinguishing different context. An example of an augmented Chinese parse tree aligned to an English string is shown in Figure 3, in which the incorrect alignment in Figure 1 is fixed. A few examples of the extracted Hiero rules and tree-to-string rules are also listed, which we would not have been able to extract from the original incorrect word alignment when the `*pro*` was missing.

3.3 Soft Recovery: EC-Specific Sparse Features

Recovered ECs are often good indicators of what hypothesis should be chosen during decoding. In addition to the augmented syntax-based grammar, we propose sparse features as a soft constraint to boost the performance. For each phrase pair, Hiero rule or tree-to-string rule in the MT system, a binary feature f_k fires if there exists a `*pro*` on the source side and it aligns to one of its most frequently aligned target words found in the training corpus. We also fire another feature if `*pro*`

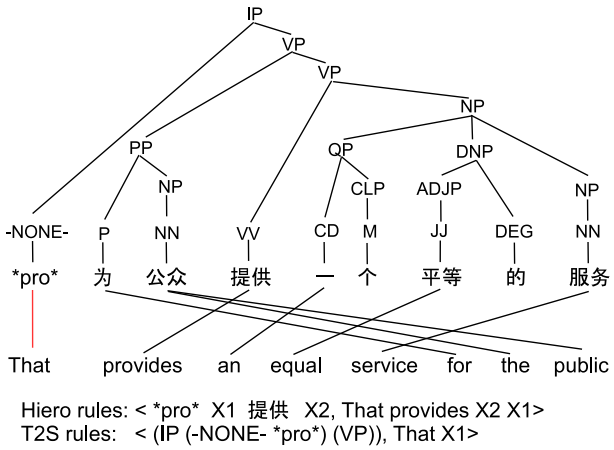

Figure 3: Fixed word alignment and examples of extracted Hiero rules and tree-to-string rules.

aligns to any other target words so the model can choose to penalize them based on a tuning set. Similar features can fire for `*PRO*`. The feature weights can be tuned on a tuning set in a log-linear model along with other usual features/costs, including language model scores, bi-direction translation probabilities, etc. The motivation for such sparse features is to reward those phrase pairs and rules that have highly confident lexical pairs specifically related to ECs, and penalize those who don't have such lexical pairs.

Table 3 listed some of the most frequent English words aligned to `*pro*` or `*PRO*` in a Chinese-English parallel corpus with 2M sentence pairs. Their co-occurrence counts and the lexical translation probabilities are also shown in the table. In total we use 15 sparse features for frequent lexical pairs, including 13 for `*pro*` and 2 for `*PRO*`, and two more features for any other target words that align to `*pro*` or `*PRO*`.

Source	Target	Counts	$P(t\|s)$
pro	the	93100	0.11
pro	to	86965	0.10
pro	it	45423	0.05
pro	in	36129	0.04
pro	we	24509	0.03
pro	which	17259	0.02
PRO	to	195464	0.32
PRO	for	31200	0.05

Table 3: Example of frequent word pairs used for sparse features.

4 Experimental Results

4.1 Empty Category Prediction

We use Chinese Treebank (CTB) v7.0 to train and test the EC prediction model. We partition the data into training, development and test sets. The training set includes 32925 sentences from CTB files 0001-0325, 0400-0454, 0500-0542, 0600-0840, 0590-0596, 1001-1120, 2000-3000, cctv, cnn, msnbc, and phoenix 00-06. The development set has 3033 sentences, from files 0549-0554, 0900-0931, 1136-1151, 3076-3145, and phoenix 10-11. The test set contains 3297 sentences, from files 0543-0548, 0841-0885, 1121-1135, 3001-3075, and phoenix 07-09.

To measure the accuracy of EC prediction, we project the predicted tags from the upper level nodes in the parse trees down to the surface level based on the position information encoded in the tags. The position index for each inserted EC, counted at the surface level, is attached for scoring purpose. The same operation is applied on both the reference and the system output trees. Such projection is necessary, especially when the two trees differ in structure (e.g. gold trees vs. machine-generated trees). We compute the precision, recall and F1 scores for each EC on the test set, and collect their counts in the reference and system output. The results are shown in Table 4, where the LDC gold parse trees are used to extract syntactic features for the model. The first row in the table shows the accuracy for the places where no EC should be inserted. The predictor achieves 99.5% F1 score for this category, with limited number of missing or false positives. The F1 scores for majority of the ECs are above 70%, except for "*", which is relatively rare in the data. For the two categories that are interesting to MT, `*pro*` and `*PRO*`, the predictor achieves 74.3% and 81.5% in F1 scores, respectively.

The results reported above are based on the LDC gold parse trees. To apply the EC prediction to NLP applications, such as MT, it is impossible to always rely on the gold trees due to its limited availability. We parse our test set with a maximum entropy based statistical parser (Ratnaparkhi, 1997) first. The parser accuracy is around 84% on the test set. Then we extract features based on the system-generated parse trees, and decode with the previously trained model. The results are shown in Table 5. Compared to those in Table 4, the F1 scores dropped by different degrees for dif-

Tag	Ref	Sys	P	R	F1
NULL	75159	75508	99.3	99.7	99.5
pro	1692	1442	80.8	68.9	74.3
PRO	1410	1282	85.6	77.8	81.5
T	1851	1845	82.8	82.5	82.7
OP	1721	1853	90.9	97.9	94.2
RNR	51	39	87.2	66.7	75.6
*	156	96	63.5	39.1	48.4

Table 4: Prediction accuracy with gold parse trees, where NULL represents the cases where no ECs should be produced.

ferent types. Such performance drop is expected since the system relies heavily on syntactic structure, and parsing errors create an inherent mismatching condition between the training and testing time. The smallest drop among all types is on NULL, at about 1.6%. The largest drop occurs for *OP*, at 27.1%, largely due to the parsing errors on the CP nodes. The F1 scores for *pro* and *PRO* when using system-generated parse trees are between 50% to 60%.

Tag	Precision	Recall	F1
NULL	97.6	98.2	97.9
pro	51.1	50.1	50.6
PRO	66.4	50.5	57.3
T	68.2	59.9	63.8
OP	66.8	67.3	67.1
RNR	70.0	54.9	61.5
*	60.9	35.9	45.2

Table 5: Prediction accuracy with system-generated parse trees.

To show the effect of ECs other than *pro* and *PRO*, we remove all ECs in the training data except *pro* and *PRO*. So the model only predicts NULL, *pro* or *PRO*. The results on the test set are listed in Table 6. There is 0.8% and 0.5% increase on NULL and *pro*, respectively. The F1 score for *PRO* drops by 0.2% slightly.

As mentioned earlier, for MT we focus on recovering *pro* and *PRO* only. The model generating the results in Table 6 is the one we applied in our MT experiments reported later.

In order to compare to the state-of-the-art models to see where our model stands, we switch our training, development and test data to those used in the work of (Yang and Xue, 2010) and (Cai et al., 2011), for the purpose of a direct comparison. The training set includes CTB files 0081 through 0900. The development set includes files 0041 to 0080, and the test set contains files 0001-0040 and 0901-0931. We merge all empty categories into a single type in the training data before training our EC prediction model. To compare the performance on system-generated parse trees, we also train a Berkeley parser on the same training data and parse the test set. The prediction accuracy for such single type on the test set with gold or system-generated parse trees is shown in Table 7, compared to the numbers reported in (Yang and Xue, 2010) and (Cai et al., 2011). The model we proposed achieves 6% higher F1 score than that in (Yang and Xue, 2010) and 2.6% higher than that in (Cai et al., 2011), which is significant. This shows the effectiveness of our structured approach.

Tag	Precision	Recall	F1
NULL	98.5	98.9	98.7
pro	51.0	51.1	51.1
PRO	66.0	50.4	57.1

Table 6: Prediction accuracy with system-generated parse trees, modeling *pro* and *PRO* only.

Model	T	P	R	F1
(Yang and Xue, 2010)	G	95.9	83.0	89.0
Structured (this work)	G	96.5	93.6	95.0
(Yang and Xue, 2010)	S	80.3	52.1	63.2
(Cai et al., 2011)	S	74.0	61.3	67.0
Structured (this work)	S	74.9	65.1	69.6

Table 7: Comparison with the previous results, using the same training and test data. T: parse trees. G: gold parse trees. S: system-generated parse trees. P: precision. R: recall.

4.2 MT Results

In the Chinese-to-English MT experiments, we test two state-of-the-art MT systems. One is an re-implementation of Hiero (Chiang, 2005), and the other is a hybrid syntax-based tree-to-string system (Zhao and Al-onaizan, 2008), where normal phrase pairs and Hiero rules are used as a backoff for tree-to-string rules.

The MT training data includes 2 million sentence pairs from the parallel corpora released by

LDC over the years, with the data from United Nations and Hong Kong excluded [2]. The Chinese text is segmented with a segmenter trained on the CTB data using conditional random field (CRF), followed by the longest-substring match segmentation in a second pass. Our language model (LM) training data consists of about 10 billion English words, which includes Gigaword and other newswire and web data released by LDC, as well as the English side of the parallel training corpus. We train a 6-gram LM with modified Kneser-Ney smoothing (Chen and Goodman, 1998). Our tuning set for MT contains 1275 sentences from LDC2010E30. We test our system on the NIST MT08 Newswire (691 sentences) and Weblog (666 sentences) sets. Both tuning and test sets have 4 sets of references for each sentence. The MT systems are optimized with pairwise ranking optimization (Hopkins and May, 2011) to maximize BLEU (Papineni et al., 2002).

We first predict `*pro*` and `*PRO*` with our annotation model for all Chinese sentences in the parallel training data, with `*pro*` and `*PRO*` inserted between the original Chinese words. Then we run GIZA++ (Och and Ney, 2000) to generate the word alignment for each direction and apply grow-diagonal-final (Koehn et al., 2003), same as in the baseline. We want to measure the impact on the word alignment, which is an important step for the system building. We append a 300-sentence set, which we have human hand alignment available as reference, to the 2M training sentence pairs before running GIZA++. The alignment accuracy measured on this alignment test set, with or without `*pro*` and `*PRO*` inserted before running GIZA++, is shown in Table 8. To make a fair comparison with the baseline alignment, any target words aligned to ECs are deemed as unaligned during scoring. We observe 1.2% improvement on function word related links, and almost the same accuracy on content words. This is understandable since `*pro*` and `*PRO*` are mostly aligned to the function words at the target side. The precision and recall for function words are shown in Table 9. We can see higher accuracy in both precision and recall when ECs (`*pro*` and `*PRO*`) are recovered in the Chinese side. Especially, the precision is improved by 2% absolute.

[2]The training corpora include LDC2003E07, LDC2003E08, LDC2005T10, LDC2006E26, LDC2006G05, LDC2007E103, LDC2008G05, LDC2009G01, and LDC2009G02.

System	Function	Content	All
Baseline	51.7	69.7	65.4
+EC	52.9	69.6	65.7

Table 8: Word alignment F1 scores with or without `*pro*` and `*PRO*`.

System	Precision	Recall	F1
Baseline	54.1	49.5	51.7
+EC	56.0	50.1	52.9

Table 9: Word alignment accuracy for function words only.

Next we extract phrase pairs, Hiero rules and tree-to-string rules from the original word alignment and the improved word alignment, and tune all the feature weights on the tuning set. The weights include those for usual costs and also the sparse features proposed in this work specifically for ECs. We test all the systems on the MT08 Newswire and Weblog sets.

The BLEU scores from different systems are shown in Table 10 and Table 11, respectively. We measure the incremental effect of prediction (inserting `*pro*` and `*PRO*`) and sparse features. Pre-processing of the data with ECs inserted improves the BLEU scores by about 0.6 for newswire and 0.2 to 0.3 for the weblog data, compared to each baseline separately. On top of that, adding sparse features helps by another 0.3 on newswire and 0.2 to 0.4 on weblog. Overall, the Hiero and tree-to-string systems are improved by about 1 point for newswire and 0.4 to 0.7 for weblog. The smaller gain on the weblog data could be due to the more difficult data to parse, which affects the accuracy of EC prediction. All the results in Table 10 and 11 marked with "*" are statistically significant with $p < 0.05$ using the sign test described in (Collins et al., 2005), compared to the baseline results in each table. Two MT examples are given in Table 12, which show the effectiveness of the recovered ECs in MT.

System	MT08-nw	MT08-wb
Hiero	33.99	25.40
+prediction	34.62*	25.63
+prediction+sparse	34.95*	25.80*

Table 10: BLEU scores in the Hiero system.

Source	*pro* 要 记住 ," 我们 是 幸运 的 一代 " 。
Reference	You should remember, "We are the fortunate generation."
Old	Have to remember that "We are the fortunate generation."
New	You have to remember that "We are the fortunate generation."
Source	塔利班 呼吁 德国 与 韩国 *PRO* 自 阿富汗 撤军 ,
Reference	The Taliban called on Germany and South Korea to withdraw their troops from Afghanistan,
Old	The Taliban called on Germany and South Korea withdraw their troops from Afghanistan,
New	The Taliban called on Germany and South Korea to withdraw their troops from Afghanistan,

Table 12: Examples of baseline (old) and improved system (new) output.

System	MT08-nw	MT08-wb
T2S+Hiero	34.53	25.80
+prediction	35.17*	26.08
+prediction+sparse	35.51*	26.53*

Table 11: BLEU scores in the tree-to-string system with Hiero rules as backoff.

5 Related Work

Empty categories have been studied in recent years for several languages, mostly in the context of reference resolution and syntactic processing for English, such as in (Johnson, 2002; Dienes and Dubey, 2003; Gabbard et al., 2006). More recently, EC recovery for Chinese started emerging in literature. In (Guo et al., 2007), non-local dependencies are migrated from English to Chinese for generating proper predicate-argument-modifier structures from surface context free phrase structure trees. In (Zhao and Ng, 2007), a decision tree learning algorithm is presented to identify and resolve Chinese anaphoric zero pronouns. and achieves a performance comparable to a heuristic rule-based approach. Similar to the work in (Dienes and Dubey, 2003), empty detection is formulated as a tagging problem in (Yang and Xue, 2010), where each word in the sentence receives a tag indicating whether there is an EC before it. A maximum entropy model is utilized to predict the tags, but different types of ECs are not distinguished. In (Cai et al., 2011), a language-independent method was proposed to integrate the recovery of empty elements into syntactic parsing. As shown in the previous section, our model outperforms the model in (Yang and Xue, 2010) and (Cai et al., 2011) significantly using the same training and test data. (Luo and Zhao, 2011) also tries to predict the existence of an EC in Chinese sentences, but the ECs in the middle of a tree constituent are lumped into a single position and are not uniquely recoverable.

There exists only a handful of previous work on applying ECs explicitly to machine translation so far. One of them is the work reported in (Chung and Gildea, 2010), where three approaches are compared, based on either pattern matching, CRF, or parsing. However, there is no comparison between using gold trees and automatic trees. There also exist a few major differences on the MT part between our work and theirs. First, in addition to the pre-processing of training data and inserting recovered empty categories, we implement sparse features to further boost the performance, and tune the feature weights directly towards maximizing the machine translation metric. Second, there is no discussion on the quality of word alignment in (Chung and Gildea, 2010), while we show the alignment improvement on a hand-aligned set. Last, they use a phase-based system trained on only 60K sentences, while we conduct experiments on more advanced Hiero and tree-to-string systems, trained on 2M sentences in a much larger corpus. We directly take advantage of the augmented parse trees in the tree-to-string grammar, which could have larger impact on the MT system performance.

6 Conclusions and Future Work

In this paper, we presented a novel structured approach to EC prediction, which utilizes a maximum entropy model with various syntactic features and shows significantly higher accuracy than the state-of-the-art approaches. We also applied the predicted ECs to a large-scale Chinese-to-English machine translation task and achieved significant improvement over two strong MT base-

lines, i.e. a hierarchical phase-based system and a tree-to-string syntax-based system. More work remain to be done next to further take advantages of ECs. For example, the recovered ECs can be encoded in a forest as the input to the MT decoder and allow the decoder to pick the best MT output based on various features in addition to the sparse features we proposed in this work. Many promising approaches can be explored in the future.

Acknowledgments

We would like to acknowledge the support of DARPA under Grant HR0011-12-C-0015 for funding part of this work. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

Ann Bies and Mohamed Maamouri. 2003. Penn Arabic treebank guidelines. In *http://www.ircs.upenn.edu/arabic/Jan03release/guidelines-TB-1-28-03.pdf*.

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:short papers*, pages 212–216.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. In *Technical Report TR-10-98, Computer Science Group, Harvard University*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, Michigan, June.

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.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 531–540.

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

Ryan Gabbard, Seth Kulick, and Mitchell Marcus. 2006. Fully parsing the penn treebank. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.

Yuqing Guo, 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 (EMNLP-CoNLL)*.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362.

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*.

Philipp Koehn, Franz Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of HLT-NAACL*, pages 48–54.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616.

Xiaoqiang Luo and Bing Zhao. 2011. A statistical tree annotator and its applications. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1230–1238.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. In *Computational Linguistics*, volume 19(2), pages 313–330.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, China, October.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of Second Conference on Empirical*

Methods in Natural Language Processing, pages 1–10.

Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*, volume 11(2), pages 207–238.

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*, pages 1382–1390, Beijing, China, August.

Bing Zhao and Yaser Al-onaizan. 2008. Generalizing local and non-local word-reordering patterns for syntax-based machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 572–581.

Shanheng Zhao and Hwee Tou Ng. 2007. Identification and resolution of Chinese zero pronouns: A machine learning approach. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.