Experiments in Base-NP Chunking and Its Role in Dependency Parsing for Thai

Shisanu Tongchim, Virach Sornlertlamvanich
Thai Computational Linguistics Laboratory
NICT Asia Research Center
112 Paholyothin Road
Klong 1, Klong Luang
Pathumthani 12120, Thailand
{shisanu,virach}@tcllab.org

Hitoshi Isahara
NICT
3-5, Hikari-dai, Seika-cho
Soraku-gun, Kyoto, 619-0289, Japan
isahara@nict.go.jp

Abstract

This paper studies the role of base-NP information in dependency parsing for Thai. The baseline performance reveals that the base-NP chunking task for Thai is much more difficult than those of some languages (like English). The results show that the parsing performance can be improved (from 60.30% to 63.74%) with the use of base-NP chunk information, although the best chunker is still far from perfect ($F_{\beta=1} = 83.06\%$).

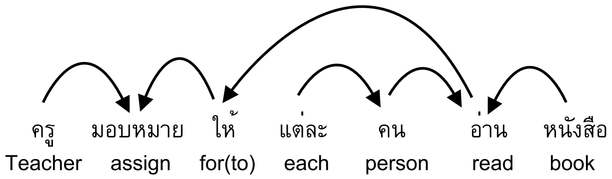
1 Introduction

Many NLP applications require syntactic information and tools for syntactic analysis. However, these linguistic resources are only available for some languages. In case of Thai, the research in developing tools for syntactic analysis and syntactically annotated corpora is still limited. Most research in the past has focused on morphological analysis (i.e. word segmentation, part-of-speech (POS) tagging). This can be viewed as a bottleneck for developing NLP applications that require a deeper understanding of the language.

We have an ongoing project in developing a syntactically annotated corpus. To accelerate the corpus annotation, some syntactic analysis tools can be applied in a preprocessing step before correcting the results by human annotators. In this paper, we use the first portion of completely annotated corpus to examine the dependency parsing and base-NP chunking. The findings will provide some guidelines in selecting a parser and a base-NP chunker for our corpus annotation workflow.

2 Dependency Parsing for Thai

The dependency structure for Thai is more flexible than some languages like Japanese (Sekine et al., 2000), Turkish (Eryigit and Oflazer, 2006), while it is close to Chinese (Cheng et al., 2005) and English (Nivre and Scholz, 2004). An example of a Thai sentence with dependency relations is outlined in Fig. 1. Note that the dependency links are drawn from the dependents to their heads. The dependency relations of Thai are bidirectional in nature and the root node can be found in arbitrary positions. Some languages (e.g. Japanese) have more constrained dependency structures, for example, the dependency relations are only from left to right and the root node is at the rightmost. Due to the lack of structural constraints and larger number of possible candidates, finding the correct dependency structure for Thai is more difficult.



"The teacher assigns each person to read a book"

Figure 1: An example of a Thai sentence with dependency relations.

There are only few studies investigating the dependency parsing for Thai. To our knowledge, the first research regarding dependency analysis was done in (Aroonmanakun, 1989). However, this research is based on a very small corpus (50 sentences). The lack of syntactically annotated cor-

Coling 2008: Companion volume – Posters and Demonstrations, pages 123–126
Manchester, August 2008

pora may be a possible explanation why not much research has been done in this area. Some have been developed, but they are relatively small or not public, for example, a treebank of 400 sentences used in (Satayamas et al., 2005).

To overcome the shortage of corpora, we initiate the development of a syntactically annotated corpus. This corpus will be used as a fundamental linguistic resource for various projects. To improve the annotation workflow, we use the first portion of completely annotated corpus in experimenting with dependency parsing and base-NP chunking. The results will be used to improve the preprocessing step of annotation.

Two dependency parsers are included in our experiments. Both are data-driven.

- *Model 1* : The first model has been widely studied in parsing Japanese text. Some machine learning techniques are used to estimate the probability that word w_i modifies word w_j. Thus, the probability matrix of binary dependency relations can be derived from this estimation. Some search algorithms are then used to find the most probable dependency structure. In this study, we use support vector machines (SVMs) to estimate the probability values and use a beam search algorithm to find the most likely dependency structure.

  In parsing Japanese text, the root position is not an issue. For Thai, however, we have to identify the root position before finding the complete dependency relations. Thus, we incorporate an additional module to identify the root node of the sentence. This root finding module is also based on an SVM.

  The root finding module selects the word with highest probability of being the root node. The following features are used in the root model: 1. POS, 2. position, 3. number of verbs, 4. number of equivalent POS in front of this word, 5. number of equivalent POS after this word, 6. number of equivalent major POS in front of this word and 7. number of equivalent major POS after this word.

  For building the dependency model (e.g. relation between w_i and w_j), the following features are used: 1. POS of w_i and w_j, 2. dependency direction, 3. distance, 4. major category of w_i and w_j, 5. major POS of w_i and w_j and 6. positions of w_i and w_j.

  After identifying the root node and creating the probability matrix, the beam search (beam width=3) is performed.

- *Model 2* : For the second model, we adopt MaltParser 1.0.4 (Nivre et al., 2007) which is a shift-reduce parser. Machine learning algorithms are used to predict the sequence of actions for parsing. In this study, we use the default setting that utilizes an SVM for predicting parsing actions.

  Assuming that $\{i_0, i_1, i_2, i_4\}$ are the first four tokens in the remaining input and $\{s_0, s_1\}$ are the two topmost tokens on the stack, we use the default features including: 1. POS of $\{i_0, i_1, i_2, i_3, s_0, s_1\}$, 2. word form of $\{s_0, i_0, i_1, head(s_0)\}$, 3. dependency type of s_0 and its leftmost and rightmost dependent and the leftmost dependent of i_0.

Table 1: Performance of dependency parsing

	RA	DA	CSA
Model 1†	85.4%	76.0%	44.8%
Model 1‡	**86.2%**	**77.5%**	**47.9%**
Model 2†	89.31%	83.53%	60.30%
Model 2‡	**91.22%**	**86.03%**	**65.27%**

Note: † (without chunk), ‡ (with chunk)

To examine the role of base-NP chunk information in dependency parsing, we include chunk labels in the feature sets of both parsers. Base-NP chunks are represented by using the IOB2 format (Sang and Veenstra, 1999). In the first parsing model, the chunk label of the current word is added as a feature of the root model, while the chunk labels of both considered words are added in the dependency model. We also add a feature showing that both words reside in the same chunk or not to the dependency model. In the second model, we include chunk labels of $\{s_0, s_1, i_0, i_1, i_2, i_3\}$ as its feature set.

We use a section of completely annotated corpus consisting of 2616 sentences to experiment with dependency parsing. The sentence length ranges between 2 words to 20 words with an average of 5.68. These Thai sentences are part of our Thai-Japanese parallel corpus developed for the MT project. Since our MT project aims for the conversation domain, the source sentences are adopted mainly from dialogues and conversation books. A morphological analyzer is applied to these Thai sentences for word segmentation and POS tagging, and the results are revised manually by our annotators. The sentences are then assigned chunk labels with IOB2 representation and syntactic structure respectively.

The corpus is divided into 2355 sentences as the training set and 261 sentences as the test set. The experiment is done with gold-standard POS tags and chunk labels. Three performance metrics are used: 1. *Root accuracy (RA)*: a portion of sentences with correctly identified roots, 2. *Dependency accuracy (DA)*: a ratio of correct dependency relations to all dependency links and 3. *Complete sentence accuracy (CSA)*: a portion of sentences with correct roots and dependency patterns.

Table 1 shows the accuracy of two parsers with and without using chunk information. The results show that chunk information helps in improving the performance of both parsers, especially in the number of completely correct sentences. MaltParser (Model 2) which is a shift-reduce parser performs better in parsing Thai sentences. This conforms with previously published literature that shift-reduce parsers have been widely applied to languages with dependency structure close to Thai (e.g. English and Chinese), while variants of Model 1 are applied to languages with more constraints in dependency structure (like Japanese).

Although the parsing accuracy can be improved by chunk information, the results are based on gold-standard chunk labels. To examine the possibility for deriving chunk labels automatically, we implement and evaluate base-NP chunkers in the next section.

3 Base-NP Chunking

We implement a simplified version of Kudo's chunker (Kudo and Matsumoto, 2001). Kudo's chunker obtained very promising results on standard English chunking tasks (e.g. precision=94.2%, recall=94.3%, $F_{\beta=1} = 94.2\%$). We use forward parsing method and employ an SVM for identifying chunk labels. The original feature set of Kudo's chunker consists of: word form, POS and previous chunk labels. Specifically, the following features are used in identifying the chunk label of the word w_i: word form and POS of $\{w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}\}$, chunk labels of $\{w_{i-2}, w_{i-1}\}$. However, some preliminary results show that the original feature set does not work well with our problem. The obtained model suffers from overfitting and lack of generalization. Thus, we modify the feature set as: POS of $\{w_{i-2}, w_{i-1}, w_i, w_{i+1}\}$, chunk labels of $\{w_{i-2}, w_{i-1}\}$ and the current size of NP chunk in front of w_i. An SVM is trained to estimate the probability of the current word being each of three chunk labels (B, I, O). A beam search strategy is used to find the most probable chunk sequence.

In additional to the SVM-based chunkers, we also examine chunkers based on conditional random fields (CRFs). We use the implementation of CRF++ (Kudo, 2008). CRFs outperform several methods on this task (Sha and Pereira, 2003). Three CRF-based chunkers are included in the experiment: the first one uses word form and POS as its feature set, the second one includes word class (function word, content word) as an additional feature, the third one uses the previous three features and the major POS category.

We use the training set and test set from previous section to experiment with chunking. Table 2 shows the performance of all chunkers. A baseline algorithm selects the chunk label which is most frequently associated with POS of the current word. From the results, all chunkers outperform the baseline algorithm. The best performance can be obtained by one of CRF-based chunkers ($F_{\beta=1} = 83.06\%$). The inclusion of more features for CRF-based chunkers helps in improving the performance. In contrast, SVM-based chunkers tend to suffer from overfitting when adding more features. The results also confirm the findings of (Sha and Pereira, 2003) that CRF-based chunkers can beat any single model. However, the results are still lower than the results found in English experiments. A reason may be that Thai NPs are more ambiguous than English NPs. This is confirmed by a comparison between our baseline result ($F_{\beta=1}$=55.4%) and some baseline results of English base-NP chunking task (e.g. precision=81.9%, recall=78.2%, $F_{\beta=1}$=80.0% (Ramshaw and Marcus, 1995)). Since the baseline algorithms work exactly in the same way, the results imply that the Thai chunking task is more difficult.

We also examine the use of the best chunker as a preprocessing step of dependency parsing. Using the parser Model 2, the results are as follows: RA=90.84%, DA=84.99%, CSA=63.74%. Overall, the accuracy of using predicted chunk labels is

Table 2: Performance of base-NP chunking

	Pr.	R.	$F_{\beta=1}$
Baseline	48.5%	64.5%	55.4%
SVM+beam search			
beam width=1	70.1%	65.5%	67.7%
beam width=3	70.6%	66.6%	68.5%
beam width=5	69.6%	65.5%	67.5%
beam width=10	71.0%	66.9%	68.9%
beam width=20	71.0%	66.9%	68.9%
CRF			
word+POS	84.79%	78.52%	81.54%
word+POS+class	85.34%	79.93%	82.54%
word+POS+class+main POS	**86.04%**	**80.28%**	**83.06%**

lower than the use of gold-standard chunk labels, but still better than without any chunk information. Although the chunking accuracy is not high as in the reported results of English chunking tasks, the results show that the dependency parsing still benefits from the predicted chunk information.

4 Conclusions

The results from the chunking task show that the chunk identification for Thai is not trivial due to ambiguities in Thai NPs. The CRF-based chunkers (best:$F_{\beta=1} = 83.06\%$) are found to be more effective than the SVM-based chunkers (best:$F_{\beta=1} = 68.9\%$).

Using the predicted chunk labels from the best chunker in dependency parsing, the performance of the best dependency parser can be improved from CSA:60.30% to CSA:63.74%. This accuracy may further be improved if the performance of chunker can be increased (as is shown in parsing accuracy when using gold-standard chunk labels).

References

Aroonmanakun, Wirote. 1989. A dependency analysis of thai sentences for a computerized parsing system. Master thesis, Department of Linguistics, Chulalongkorn University.

Cheng, Yuchang, Masayuki Asahara, and Yuji Matsumoto. 2005. Chinese deterministic dependency analyzer: Examining effects of global features and root node finder. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 17–24.

Eryigit, Gülsen and Kemal Oflazer. 2006. Statistical dependency parsing for turkish. In *EACL*. The Association for Computer Linguistics.

Kudo, Taku and Yuji Matsumoto. 2001. Chunking with support vector machines. In *NAACL*.

Kudo, Taku. 2008. CRF++: Yet another CRF toolkit. http://crfpp.sourceforge.net/.

Nivre, Joakim and Mario Scholz. 2004. Deterministic dependency parsing of english text. In *Proceedings of Coling 2004*, pages 64–70, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Stetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering Journal*, 13(2):99–135.

Ramshaw, Lance A. and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94.

Sang, Erik F. Tjong Kim and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 173–179, Morristown, NJ, USA. Association for Computational Linguistics.

Satayamas, Vee, Chalatip Thumkanon, and Asanee Kawtrakul. 2005. Bootstrap cleaning and quality control for Thai tree bank construction. In *The 9th National Computer Science and Engineering Conference*, Bangkok, Thailand, Oct 27–Oct 28. (In Thai).

Sekine, Satoshi, Kiyotaka Uchimoto, and Hitoshi Isahara. 2000. Backward beam search algorithm for dependency analysis of japanese. In *COLING*, pages 754–760. Morgan Kaufmann.

Sha, Fei and Fernando C. N. Pereira. 2003. Shallow parsing with conditional random fields. In *HLT-NAACL*.