Predicate Argument Structure Analysis using Transformation-based Learning

Hirotoshi Taira **Sanae Fujita** **Masaaki Nagata**
NTT Communication Science Laboratories
2-4, Hikaridai, Seika-cho, Souraku-gun, Kyoto 619-0237, Japan
{taira,sanae}@cslab.kecl.ntt.co.jp nagata.masaaki@lab.ntt.co.jp

Abstract

Maintaining high annotation consistency in large corpora is crucial for statistical learning; however, such work is hard, especially for tasks containing semantic elements. This paper describes predicate argument structure analysis using transformation-based learning. An advantage of transformation-based learning is the readability of learned rules. A disadvantage is that the rule extraction procedure is time-consuming. We present incremental-based, transformation-based learning for semantic processing tasks. As an example, we deal with Japanese predicate argument analysis and show some tendencies of annotators for constructing a corpus with our method.

1 Introduction

Automatic predicate argument structure analysis (PAS) provides information of "who did what to whom" and is an important base tool for such various text processing tasks as machine translation information extraction (Hirschman et al., 1999), question answering (Narayanan and Harabagiu, 2004; Shen and Lapata, 2007), and summarization (Melli et al., 2005). Most recent approaches to predicate argument structure analysis are statistical machine learning methods such as support vector machines (SVMs)(Pradhan et al., 2004). For predicate argument structure analysis, we have the following representative large corpora: FrameNet (Fillmore et al., 2001), PropBank (Palmer et al., 2005), and NomBank (Meyers et al., 2004) in English, the Chinese PropBank (Xue, 2008) in Chinese, the GDA Corpus (Hashida, 2005), Kyoto Text Corpus Ver.4.0 (Kawahara et al., 2002), and the NAIST Text Corpus (Iida et al., 2007) in Japanese.

The construction of such large corpora is strenuous and time-consuming. Additionally, maintaining high annotation consistency in such corpora is crucial for statistical learning; however, such work is hard, especially for tasks containing semantic elements. For example, in Japanese corpora, distinguishing true dative (or indirect object) arguments from time-type argument is difficult because the arguments of both types are often accompanied with the 'ni' case marker.

A problem with such statistical learners as SVM is the lack of interpretability; if accuracy is low, we cannot identify the problems in the annotations.

We are focusing on transformation-based learning (TBL). An advantage for such learning methods is that we can easily interpret the learned model. The tasks in most previous research are such simple tagging tasks as part-of-speech tagging, insertion and deletion of parentheses in syntactic parsing, and chunking (Brill, 1995; Brill, 1993; Ramshaw and Marcus, 1995). Here we experiment with a complex task: Japanese PASs. TBL can be slow, so we proposed an incremental training method to speed up the training. We experimented with a Japanese PAS corpus with a graph-based TBL. From the experiments, we interrelated the annotation tendency on the dataset.

The rest of this paper is organized as follows. Section 2 describes Japanese predicate structure, our graph expression of it, and our improved method. The results of experiments using the NAIST Text Corpus, which is our target corpus, are reported in Section 3, and our conclusion is provided in Section 4.

2 Predicate argument structure and graph transformation learning

First, we illustrate the structure of a Japanese sentence in Fig. 1. In Japanese, we can divide a sentence into *bunsetsu* phrases (BP). A BP usually consists of one or more content words and zero,

Proceedings of the ACL 2010 Conference Short Papers, pages 162–167,
Uppsala, Sweden, 11-16 July 2010.

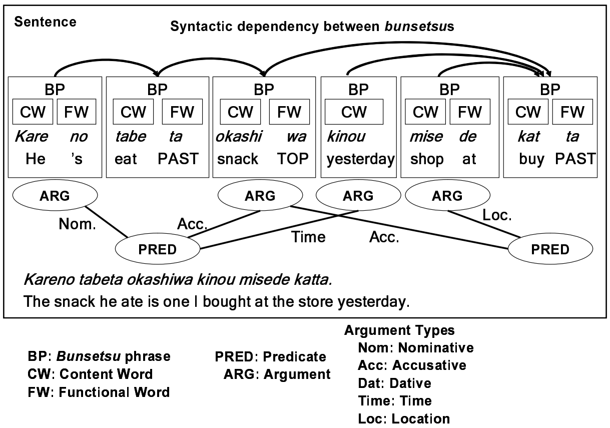

Figure 1: Graph expression for PAS

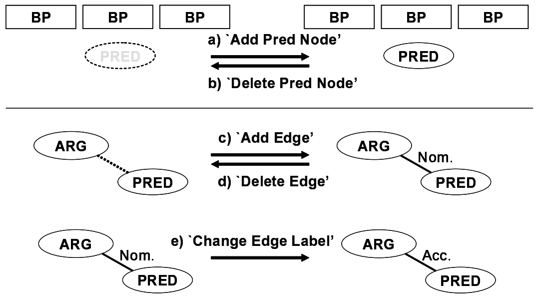

Figure 2: Transform types

one, or more than one functional words. Syntactic dependency between *bunsetsu* phrases can be defined. Japanese dependency parsers such as Cabocha (Kudo and Matsumoto, 2002) can extract BPs and their dependencies with about 90% accuracy.

Since predicates and arguments in Japanese are mainly annotated on the head content word in each BP, we can deal with BPs as candidates of predicates or arguments. In our experiments, we mapped each BP to an argument candidate node of graphs. We also mapped each predicate to a predicate node. Each predicate-argument relation is identified by an edge between a predicate and an argument, and the argument type is mapped to the edge label. In our experiments below, we defined five argument types: nominative (subjective), accusative (direct objective), dative (indirect objective), time, and location. We use five transformation types: a) add or b) delete a predicate node, c) add or d) delete an edge between an predicate and an argument node, e) change a label (= an argument type) to another label (Fig. 2). We explain the existence of an edge between a predicate and an argument labeled t candidate node as that the predicate and the argument have a t type relationship.

Transformation-based learning was proposed by (Brill, 1995). Below we explain our learning strategy when we directly adapt the learning method to our graph expression of PASs. First, unstructured texts from the training data are inputted. After pre-processing, each text is mapped to an initial graph. In our experiments, the initial graph has argument candidate nodes with corresponding BPs and no predicate nodes or edges. Next, comparing the current graphs with the gold standard graph structure in the training data, we find the different statuses of the nodes and edges among the graphs. We extract such transformation rule candidates as ‘add node’ and ‘change edge label’ with constraints, including ‘the corresponding BP includes a verb’ and ‘the argument candidate and the predicate node have a syntactic dependency.’ The extractions are executed based on the rule templates given in advance. Each extracted rule is evaluated for the current graphs, and error reduction is calculated. The best rule for the reduction is selected as a new rule and inserted at the bottom of the current rule list. The new rule is applied to the current graphs, which are transferred to other graph structures. This procedure is iterated until the total errors for the gold standard graphs become zero. When the process is completed, the rule list is the final model. In the test phase, we iteratively transform nodes and edges in the graphs mapped from the test data, based on rules in the model like decision lists. The last graph after all rule adaptations is the system output of the PAS.

In this procedure, the calculation of error reduction is very time-consuming, because we have to check many constraints from the candidate rules for all training samples. The calculation order is $O(MN)$, where M is the number of articles and N is the number of candidate rules. Additionally, an edge rule usually has three types of constraints: ‘pred node constraint,’ ‘argument candidate node constraint,’ and ‘relation constraint.’ The number of combinations and extracted rules are much larger than one of the rules for the node rules. Ramshaw et al. proposed an index-based efficient reduction method for the calculation of error reduction (Ramshaw and Marcus, 1994). However, in PAS tasks, we need to check the exclusiveness of the argument types (for example, a predicate argument structure does not have two nominative arguments), and we cannot directly use the method. Jijkoun et al. only used candidate rules that happen in the current and gold standard graphs and used SVM learning for constraint checks (Jijkoun and de Rijke, 2007). This method is effective for achieving high accuracy; however, it loses the readability of the rules. This is contrary to our aim to extract readable rules.

To reduce the calculations while maintaining readability, we propose an incremental method and describe its procedure below. In this procedure, we first have PAS graphs for only one article. After the total errors among the current and gold standard graphs become zero in the article, we proceed to the next article. For the next article, we first adapt the rules learned from the previous article. After that, we extract new rules from the two articles until the total errors for the articles become zero. We continue these processes until the last article. Additionally, we count the number of rule occurrences and only use the rule candidates that happen more than once, because most such rules harm the accuracy. We save and use these rules again if the occurrence increases.

3 Experiments

3.1 Experimental Settings

We used the articles in the NAIST Text Corpus version 1.4β (Iida et al., 2007) based on the *Mainichi Shinbun* Corpus (Mainichi, 1995), which were taken from news articles published in the Japanese *Mainichi Shinbun* newspaper. We used articles published on January 1st for training examples and on January 3rd for test examples. Three original argument types are defined in the NAIST Text Corpus: nominative (or subjective), accusative (or direct object), and dative (or indirect object). For evaluation of the difficult annotation cases, we also added annotations for 'time' and 'location' types by ourselves. We show the dataset distribution in Table 1. We extracted the BP units and dependencies among these BPs from the dataset using Cabocha, a Japanese dependency parser, as pre-processing. After that, we adapted our incremental learning to the training data. We used two constraint templates in Tables 2 and 3 for predicate nodes and edges when extracting the rule candidates.

Table 1: Data distribution

	Training	Test
# of Articles	95	74
# of Sentences	1,129	687
# of Predicates	3,261	2,038
# of Arguments	3,877	2,468
Nom.	1,717	971
Acc.	1,012	701
Dat.	632	376
Time	371	295
Loc.	145	125

Table 4: Total performances (F1-measure (%))

Type	System	P	R	F1
Pred.	Baseline	89.4	85.1	87.2
	Our system	**91.8**	**85.3**	**88.4**
Arg.	Baseline	79.3	59.5	68.0
	Our system	**81.9**	**62.4**	**70.8**

3.2 Results

Our incremental method takes an hour. In comparison, the original TBL cannot even extract one rule in a day. The results of predicate and argument type predictions are shown in Table 4. Here, 'Baseline' is the baseline system that predicts the BSs that contain verbs, adjectives, and *da* form nouns ('to be' in English) as predicates and predicts argument types for BSs having syntactical dependency with a predicted predicate BS, based on the following rules: 1) BSs containing nominative (*ga*) / accusative (*wo*) / dative (*ni*) case markers are predicted to be nominative, accusative, and dative, respectively. 2) BSs containing a topic case marker (*wa*) are predicted to be nominative. 3) When a word sense category from a Japanese ontology of the head word in BS belongs to a 'time' or 'location' category, the BS is predicted to be a 'time' and 'location' type argument. In all precision, recall, and F1-measure, our system outperformed the baseline system.

Next, we show our system's learning curve in Fig. 3. The number of final rules was 68. This indicates that the first twenty rules are mainly effective rules for the performance. The curve also shows that no overfitting happened. Next, we show the performance for every argument type in Table 5. 'TBL,' which stands for 'transformation-based learning,' is our system. In this table, the performance of the dative and time types improved, even though they are difficult to distinguish. On the other hand, the performance of the location type argument in our system is very low. Our method learns rules as decreasing errors of

Table 2: Predicate node constraint templates

Pred. Node Constraint Template		Rule Example	
Constraint	Description	Pred. Node Constraint	Operation
pos1	noun, verb, adjective, etc.	pos1='ADJECTIVE'	add pred node
pos2	independent, attached word, etc.	pos2='DEPENDENT WORD'	del pred node
pos1 & pos2	above two features combination	pos1='VERB' & pos2='ANCILLARY WORD'	add pred node
'da'	*da* form (copula)	'da form'	add pred node
lemma	word base form	lemma='%'	add pred node

Table 3: Edge constraint templates

Edge Constraint Template			Rule Example	
Arg. Cand. Const.	Pred. Node Const.	Relation Const.	Edge Constraint	Operation
FW (=func. word)	$*$	dep(arg $\rightarrow$ pred)	FW of Arg. ='*wa*(TOP)' & dep(arg $\rightarrow$ pred)	add NOM edge
$*$	FW	dep(arg $\leftarrow$ pred)	FW of Pred. ='*na*(ADNOMINAL)' & dep(arg $\leftarrow$ pred)	add NOM edge
SemCat (=semantic category)	$*$	dep(arg $\rightarrow$ pred)	SemCat of Arg. = 'TIME' & dep(arg $\rightarrow$ pred)	add TIME edge
FW	passive form	dep(arg $\rightarrow$ pred)	FW of Arg. ='*ga*(NOM) & Pred.: passive form	chg edge label NOM $\rightarrow$ ACC
$*$	kform (= type of inflected form)	$*$	kform of Pred. = continuative 'ta' form	add NOM edge
SemCat	Pred. SemCat	$*$	SemCat of Arg. = 'HUMAN' & Pred. SemCat = 'PHYSICAL MOVE'	add NOM edge

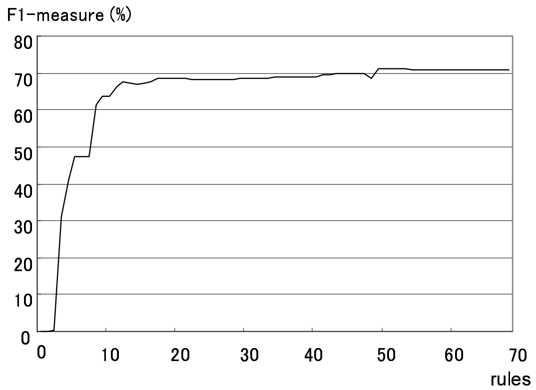

Figure 3: Learning curves: x-axis = number of rules; y-axis: F1-measure (%)

all arguments, and the performance of the location type argument is probably sacrificed for total error reduction because the number of location type arguments is much smaller than the number of other argument types (Table 1), and the improvement of the performance-based learning for location type arguments is relatively low. To confirm this, we performed an experiment in which we gave the rules of the baseline system to our system as initial rules and subsequently performed our incremental learning. 'Base + TBL' shows the experiment. The performance for the location type argument improved drastically. However, the total performance of the arguments was below the original TBL. Moreover, the 'Base + TBL' performance surpassed the baseline system. This indicates that our system learned a reasonable model.

Finally, we show some interesting extracted rules in Fig. 4. The first rule stands for an expression where the sentence ends with the performance of something, which is often seen in Japanese newspaper articles. The second and third rules represent that annotators of this dataset tend to annotate time types for which the semantic category of the argument is time, even if the argument looks like the dat. type, and annotators tend to annotate dat. type for arguments that have an dat.

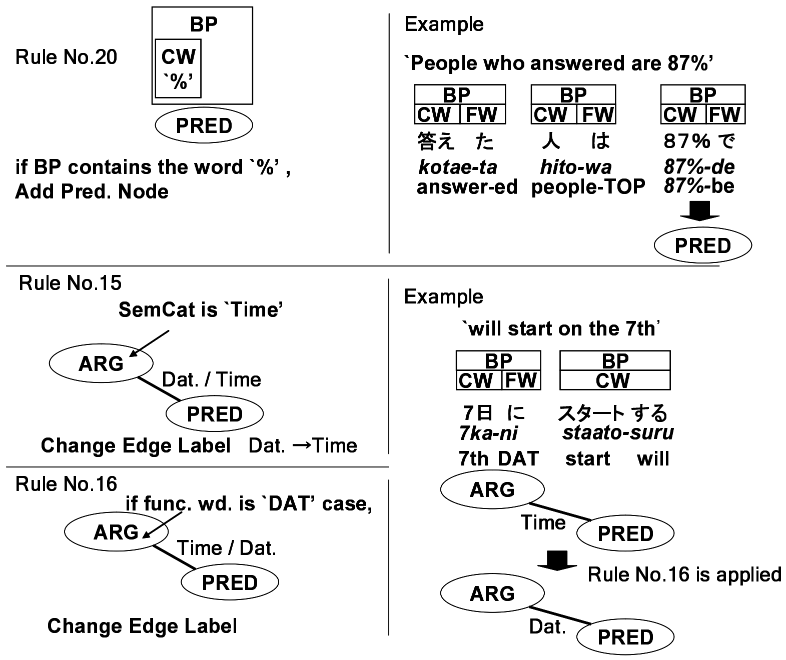

Figure 4: Examples of extracted rules

Table 5: Results for every arg. type (F-measure (%))

System	Args.	Nom.	Acc.	Dat.	Time	Loc.
Base	68.0	**65.8**	79.6	70.5	51.5	**38.0**
TBL	**70.8**	64.9	**86.4**	**74.8**	**59.6**	1.7
Base + TBL	69.5	63.9	85.8	67.8	55.8	37.4

type case marker.

4 Conclusion

We performed experiments for Japanese predicate argument structure analysis using transformation-based learning and extracted rules that indicate the tendencies annotators have. We presented an incremental procedure to speed up rule extraction. The performance of PAS analysis improved, especially, the dative and time types, which are difficult to distinguish. Moreover, when time expressions are attached to the 'ni' case, the learned model showed a tendency to annotate them as dative arguments in the used corpus. Our method has potential for dative predictions and interpreting the tendencies of annotator inconsistencies.

Acknowledgments

We thank Kevin Duh for his valuable comments.

References

Eric Brill. 1993. Transformation-based error-driven parsing. In *Proc. of the Third International Workshop on Parsing Technologies*.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Charles J. Fillmore, Charles Wooters, and Collin F. Baker. 2001. Building a large lexical databank which provides deep semantics. In *Proc. of the Pacific Asian Conference on Language, Information and Computation (PACLING)*.

Kouichi Hashida. 2005. Global document annotation (GDA) manual. http://i-content.org/GDA/.

Lynette Hirschman, Patricia Robinson, Lisa Ferro, Nancy Chinchor, Erica Brown, Ralph Grishman, and Beth Sundheim. 1999. Hub-4 Event'99 general guidelines. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.

Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In *Proc. of ACL 2007 Workshop on Linguistic Annotation*, pages 132–139.

Valentin Jijkoun and Maarten de Rijke. 2007. Learning to transform linguistic graphs. In *Proc. of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing (TextGraphs-2)*, pages 53–60. Association for Computational Linguistics.

Daisuke Kawahara, Sadao Kurohashi, and Koichi Hashida. 2002. Construction of a Japanese relevance-tagged corpus (in Japanese). *Proc. of the 8th Annual Meeting of the Association for Natural Language Processing*, pages 495–498.

Taku Kudo and Yuji Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In *Proc. of the 6th Conference on Natural Language Learning 2002 (CoNLL 2002)*.

Mainichi. 1995. *CD Mainichi Shinbun 94*. Nichigai Associates Co.

Gabor Melli, Yang Wang, Yudong Liu, Mehdi M. Kashani, Zhongmin Shi, Baohua Gu, Anoop Sarkar, and Fred Popowich. 2005. Description of SQUASH, the SFU question answering summary handler for the DUC-2005 summarization task. In *Proc. of DUC 2005*.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proc. of HLT-NAACL 2004 Workshop on Frontiers in Corpus Annotation*.

Srini Narayanan and Sanda Harabagiu. 2004. Question answering based on semantic structures. In *Proc. of the 20th International Conference on Computational Linguistics (COLING)*.

M. Palmer, P. Kingsbury, and D. Gildea. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Sameer Pradhan, Waybe Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proc. of the Human Language Technology Conference/North American Chapter of the Association of Computational Linguistics HLT/NAACL 2004*.

Lance Ramshaw and Mitchell Marcus. 1994. Exploring the statistical derivation of transformational rule sequences for part-of-speech tagging. In *The Balancing Act: Proc. of the ACL Workshop on Combining Symbolic and Statistical Approaches to Language*.

Lance Ramshaw and Mitchell Marcus. 1995. Text chunking using transformation-based learning. In *Proc. of the third workshop on very large corpora*, pages 82–94.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP/CoNLL)*, pages 12–21.

Nianwen Xue. 2008. Labeling Chinese predicates with semantic roles. *Computational Linguistics*, 34(2):224–255.