A Structured SVM Semantic Parser Augmented by Semantic Tagging with Conditional Random Field

Minh Le Nguyen
Graduate Information School
Japan Advanced Institute of
Science and Technology

Address: 1-1, Asahidai, Nomi,
Ishikawa 923-1292 Japan
E-mail: nguyenml@jaist.ac.jp

Akira Shimazu
Graduate Information School
Japan Advanced Institute of
Science and Technology

Address: 1-1, Asahidai, Nomi,
Ishikawa 923-1292 Japan
E-mail: shimazu@jaist.ac.jp

Hieu Xuan Phan
Graduate Information School
Japan Advanced Institute of
Science and Technology

Address: 1-1, Asahidai, Nomi,
Ishikawa 923-1292 Japan
E-mail: hieuxuan@jaist.ac.jp

Abstract

This paper presents a novel method of semantic parsing that maps a natural language (NL) sentence to a logical form. We propose a semantic parsing method by conducting separately two steps as follows;

1) The first step is to predict semantic tags for a given input sentence.

2) The second step is to build a semantic representation structure for the sentence using the sequence of semantic tags.

We formulate the problem of semantic tagging as a sequence learning using a conditional random field models (CRFs). We then represent a tree structure of a given sentence in which syntactic and semantic information are integrated in that tree. The learning problem is to map a given input sentence to a tree structure using a structure support vector model. Experimental results on the CLANG corpus show that the semantic tagging performance achieved a sufficiently high result. In addition, the precision and recall of mapping NL sentences to logical forms i.e. the meaning representation in CLANG show an improvement in comparison with the previous work.

1. Introduction

Semantic parsing is an interesting problem in NLP as it would very likely be part of any interesting NLP applications. For example, the ability to map natural language to a formal query or command language is critical to developing more user-friendly interfaces. Traditional approaches to constructing database interfaces require an expert to hand-craft an appropriate semantic parser (Allen, 95). However, such hand-crafted parsers are time consuming to develop and suffer from problems with robustness and incompleteness, even for domain specific applications. Recent approaches have focused on using machine learning methods on the corpus of sentences and semantic annotations to map natural language sentences (NL) to a complete formal language including cite (Miller, 96)(Zelle, 96)(L.R.Tang, 2003). These machine learning methods are the application of inductive logic programming (Zelle, 96) (L.R. Tang, 2003), the adaptation of transformation based learning (R.J. Kate, et.al. 2005) to semantic parsing, and the modification of a head-driven parsing model (R.Ge and R.J. Mooney 2005).

This paper addresses a novel approach to the problem of semantic parsing by considering it as a structure classification problem with a support vector machine model (SVM). We describe a tree structure in which semantic information and syntactic information are incorporated and the semantic representation of a sentence, such as first order logic or CLANG, can be interpreted by the
tree. With this flexible structure, the problem of semantic parsing is converted to the problem of how we can parse a sentence to a tree.

Furthermore, we can easily incorporate many useful features by mapping a NL sentence to the formal representation under the framework of a structured support vector machine (Tsochantaridis、

et.al., 2004). We also utilize the advantages of kernel methods as well as discriminate learning to the semantic parsing problem. In addition, we propose a semantic tagging problem that maps each word in NL sentence to a semantic category by formulating it in a sequence learning mode. We then exploit a Conditional Random Field (CRF) model with a set of template features for that problem. Our semantic tagging proved to be very accurate showing that it can provide preprocessing in mapping NL sentences to a formal language.

Although the proposed method can be applied to any task of mapping NL sentence to a formal language, in this paper we focus on experimenting on the problem of mapping NL sentence to its representation in Coach Language (CLANG corpus) (Chen, et.al., 2003).

The rest of this paper is organized as follows: Section 2 gives some background in which we briefly present the previous work and the conditional random field model, as well as the structured support vector machine model. Section 3 proposes our method, which uses a conditional random filed for semantic tagging and augment it to the structured support vector machine for transforming a NL sentence to the coach language. Section 4 shows experimental results and Section 5 discusses the advantage of our methods and describes future works.

2. Background

2.1 Related works

Zelle and Mooney initially proposed the empirically based method using a corpus of NL sentences and their formal representation for learning by inductive logic programming (ILP) (Zelle, 1996). Several extensions for mapping a NL sentence to its logical form have been addressed by (L.R. Tang, 2003). The task of transforming a natural language sentence to a logical form was formulated as the task of determining a sequence of actions that would transform the given input sentence to a logical form (L.R. Tang, 2003) (J.R. Mooney, 2004). The main problem is how to learn a set of rules from the corpus using the ILP method. The advantage of the ILP method is that we do not need to design features for learning a set of rules from corpus. The disadvantage is that it is quite complex and slow to acquire parsers for mapping sentences to logical forms. Kate et al. presented a method (R.J.Kate, et.al. 2005) that used transformation rules to transform NL sentences to logical forms. Those transformation rules are learnt using the corpus of sentences and their logical forms. This method is much simpler than the ILP method, while it can achieve comparable result on the CLANG (Coach Language) and Query corpus. The transformation based method has the condition that the formal language should be in the form of LR grammar.

Ge and Mooney also presented a statistical method (R.Ge and J.R. Mooney 2005) by merging syntactic and semantic information. Their method relaxed the condition in (R.J. Kate et.al. 2005) and achieved a state-of the art performance on the CLANG and query database corpus. The recent work proposed by (Zettlemoyer, et.al. 2005) that maps a NL sentence to its logical form by structured classification, using a log-linear model that represents a distribution over syntactic and semantic analyses conditioned on the input sentence. This work is quite similar to our work in considering the structured classification problem. The difference is that we used the kernel based method instead of a log-linear model in order to utilize the advantage of handling a very large number of features by maximizing the margin in the learning process. In addition, the learning process in our method is performed on the corpus of NL sentences and their tree structures, while their method used probabilistic categorical grammar.

2.2 Conditional Random Fields

Let $o = o_1 o_2 ... o_T$ be some observed data sequence. Let *S* be a set of FSM states, each of which is associated with a label, $l \in L$. Let $s = (s_1, s_2, ..., s_T)$ be some state sequence, CRFs (Lafferty et.al., 2001) define the conditional probability of a state sequence given an observation sequence as

$$p_\theta(s \mid o) = \frac{1}{Z(o)} \exp\left[\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(s_{t-1}, s_t, o, t)\right]$$

where $Z(o) = \sum_{s'} \exp(\sum_{t=1}^{T}\sum_{k} \lambda_k f_k(s'_{t-1}, s'_t, o, t)$ is a normalization summing over all label sequences. f_k denotes a feature function in the language of maximum entropy modeling and λ_k is a learned weight associated with f_k feature . Each f_k is either a per--*state or a transition* feature:

$$f_k^{(per-state)}(s_t, o, t) = \delta(s_t, l)\chi_k(o, t)$$
$$f_k^{(transition)}(s_{t-1}, s_t, t) = \delta(s_{t-1}, l')\delta_k(s_t, l$$

where δ denotes the Kronecker--δ. A per-state feature combines the label *l* of current states s_t and a context predicate, i.e. the binary function $\chi_k(o, t)$. To train a Conditional Random Fields model, one can use the quasi-Newton methods, such as L-BFGS (Liu 89). The L-BFGS method has showed that it is more efficient than the others (Sha 2003). To inference a CRF models, a slightly modification of the Viterbi algorithm can be applied. The detail of the training and inference CRFs has been introduced in (Lafferty, et.al., 2001) and (Sha 2003).

2.3 Structured Support Vector Machine

Structured classification is the problem of predicting *y* from *x* in the case where *y* having a meaningful internal structure. Elements $y \in Y$ may be, for instance, sequences, strings, labeled trees, lattices, or graphs. The major problem for the structured support vector model is the modification of multiple classifications to the very large number of labels problem. To solve the problem, Tsochantaridis et al. (Tsochantaridis, et.al., 2004) presented a re-scaling method for the SVM optimization problem and viewed it as discriminative classification by employing several loss function and maximization methods. As the principle of the maximum-margin presented in (Vapnik, 1998), in the structured classification problem (Tsochantaridis, et.al., 2004) proposed several maximum-margin optimization problems $\delta\psi_i(y) \equiv \psi(x_i, y_i) - \psi(x_i, y)$. The hard-margin optimization problem is:

$$\mathrm{SVM}_0 : \underbrace{\min}_{w} \frac{1}{2}\|w\|^2$$
$$\forall i, \forall y \in Y \setminus y_i : \langle w, \delta\psi_i(y)\rangle > 0$$

where $\langle w, \delta\psi_i(y)\rangle$ is the linear combination of features representation for input and output.

The soft-margin criterion was proposed (Tsochantaridis, et.al., 2004) in order to allow errors in the training set, by introducing slack variables.

$$\mathrm{SVM}_1 : \underbrace{\min}_{w} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \text{ s.t. } \forall i, \xi_i \geq 0$$
$$\forall i, \forall y \in Y \setminus y_i : \langle w, \delta\psi_i(y)\rangle > 1 - \xi_i$$

Alternatively, using a quadratic term $\frac{C}{2n}\sum_i \xi_i^2$ to penalize margin violations, we obtained SVM_2. Here C>0 is a constant that control the tradeoff between training error minimization and margin maximization.

To deal with problems in which |Y| is very large, such as semantic parsing (Tsochantaridis:2004) proposed two approaches that generalize the formulation SVM_0 and SVM_1 to the cases of arbitrary loss function. The first approach is to re-scale the slack variables according to the loss incurred in each of the linear constraints.

$$\mathrm{SVM}^{\Delta_s} : \underbrace{\min}_{w} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \text{ s.t. } \forall i, \xi_i \geq 0$$
$$\forall i, \forall y \in Y \setminus y_i : \langle w, \delta\psi_i(y)\rangle \geq \frac{1-\xi_i}{\Delta(y_i, y)}$$

The second approach to include loss function is to re-scale the margin as a special case of the Hamming loss. The margin constraints in this setting take the following form:

$$\forall i, \forall y \in Y \setminus y_i : \langle w, \delta\psi_i(y)\rangle \geq \Delta(y_i, y) - \xi_i$$

This set of constraints yields an optimization problem, namely $\mathrm{SVM}_1^{\Delta_m}$. The algorithm to solve the maximum-margin problem in structured learning problem is presented in detail in (Tsochantaridis, et.al., 2004). In this section we briefly describe the algorithm that exploits the special structure of the maximum-margin problem, in which only a much smaller subset of constraints needs to be examined.
The algorithm aims to find a small set of active constraints that ensures a sufficiently accurate solution. The detailed algorithm, as presented in (Tsochantaridis, et.al., 2004) can be applied to all SVM formulations mentioned above. The only difference is the way the cost function are listed followings

$$\mathrm{SVM}_1^{\Delta_s} : H(y) \equiv (1 - \langle \delta\psi_i(y), w\rangle \Delta(y_i, y)$$
$$\mathrm{SVM}_2^{\Delta_s} : H(y) \equiv (1 - \langle \delta\psi_i(y), w\rangle \sqrt{\Delta(y_i, y)}$$
$$\mathrm{SVM}_1^{\Delta_m} : H(y) \equiv (\Delta(y_i, y) - \langle \delta\psi_i(y), w\rangle)$$
$$\mathrm{SVM}_2^{\Delta_m} : H(y) \equiv (\sqrt{\Delta(y_i, y)} - \langle \delta\psi_i(y), w\rangle)$$

Typically, the way to apply structured SVM is to implement feature mapping $\psi(x, y)$, the loss function $\Delta(y_i, y)$, as well as the maximization algorithm. All of those presented in the algorithm are treated as the black boxes. The modeling of $\psi(x, y)$ and $\Delta(y_i, y)$ is more or less straightforward, but solving the maximization problem for constraints selection requires exploiting the structure of ψ. In the following section, we apply a structured support vector machine to the problem of semantic parsing in which the mapping function, the maximization algorithm, and the loss function are introduced.

3. The Proposed Algorithm

The input of our semantic parsing is a NL sentence and the output is the semantic representation; that is the logical form. The proposed algorithm for semantic parsing consists of two phases:

(1) Predicting a sequence of semantic tags for a given input sentence

(2) Building a semantic representation with the sequence of semantic tags

We describe a method using CRFs and a structure support vector model in the following subsection for the first step and the second step, respectively.

3.1 Semantic tagging with CRFs

Let $w_1 w_2 ... w_N$ be a NL sentence in which N is the number of words and w_i is i^{th} word. Assuming that a POS tag for a word w_i is p_i, the lexical semantic prediction problem is to determine a lexical semantic s_i for each word w_i. Semantic tagging exploits the use of CRFs to the lexical semantic prediction problem in which a feature set for this task is presented. Table 1 shows our designed templates of words and POS tags within a window size of 5. The templates contain conjunction of two or three consecutive words, two and three consecutive POS tags. For example, $w_{-1} \wedge w_0 \wedge w_{+1}$ is a 3-conjunction template of the previous word w_{-1}, the current word w_0, and the next word w_{+1}; $p_{-1} \wedge p_0$ is a 2-conjunction template of the POS tag p_{-1} of the previous word and the POS tag p_0 of the current word.

Table 1. Feature set

Fixed templates (window size=5)
$w_{-2}; w_{-1}; w_0; w_{+1}; w_{+2}$
$w_{-2} \wedge w_{-1}; w_{-1} \wedge w_0 \wedge w_{+1}; w_{+1} \wedge w_{+2}$
$w_{-2} \wedge w_{-1} \wedge w_0; w_{-1} \wedge w_0 \wedge w_{+1}$
$w_0 \wedge w_{+1} \wedge w_{+2}$
$p_{-2}; p_{-1}; p_0; p_{+1}; p_{+2}$
$p_{-2} \wedge p_{-1}; p_{-1} \wedge p_0; p_0 \wedge p_{+1} \wedge w_{+2}$
$p_{-2} \wedge p_{-1} \wedge p_0; p_{-1} \wedge p_0 \wedge p_{+1}$
$p_0 \wedge p_{+1} \wedge p_{+2}$

Note that the task of semantic tagging assumes that a given sentence can be tagged to obtain a sequence of POS tags. This assumption is valid since the English POSs tagging accuracy is currently very high.

3.2 Structured Support Vector Models for Semantic Parsing

3.2.1 Structure representation

In this section we describe a structure representation for semantic parsing problem. A tree structure representation incorporated with semantic and syntactic information is named semantically augmented parse tree (SAPT) (R.Ge and R.J. Mooney, 2005). As defined in their work, in an SAPT, each internal node in the parse tree is annotated with a semantic label. Figure 1 shows the SAPT for a simple sentence in the CLANG domain. The semantic labels which are shown after dashes are concepts in the domain. Some concepts refer to predicates, takes an ordered list of arguments and concepts might not have arguments such as *team* and *unum*. A special semantic label, null, is used for nodes that do not correspond to any concept in the domain.

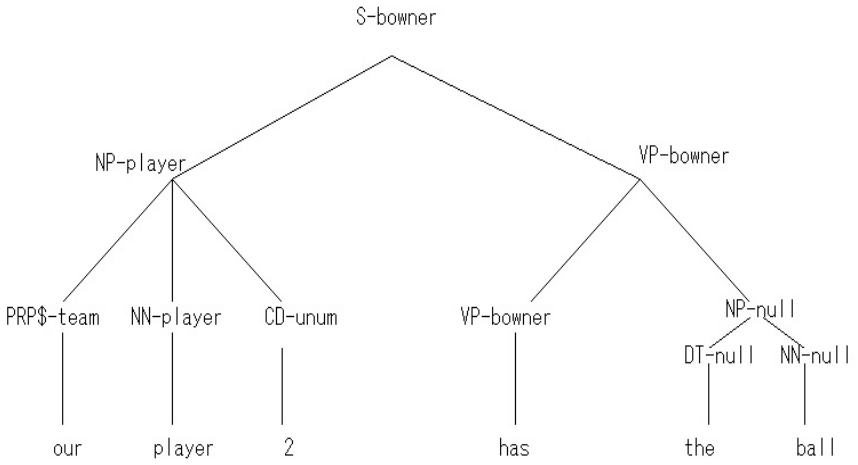

Figure 1. An Example of tree representation in SAPT

The general idea of producing a logical form using a SAPT structure is expressed as Algorithm 1. It generates a logical form based on a knowledge database K for given input node *N*. The GETSEMANTICHEAD determines which of node's children is its semantic head based on their having matching semantic labels. For example in Figure 2 N3 is determined to be the semantic head of the sentence, since it matches N8's semantic label. COMPOSEMR assigns their meaning representation(MR) to fill the arguments in the head's MR to construct the complete MR for the node. Algorithm1 with an input which is the tree shown in Figure 1 yields the result in logical form as shown in Figure 2.

1: C_i =the ith child node of N
2: C_h =GETSEMANTICHEAD(N)
3: C_{hMR} =BUILDMR(C_h,K)
4: for each other child C_i where $i \neq h$ do
5: C_{iMR} = BUILDMR(C_i,K)
6: COMPOSEMR(C_{hMR}, C_{iMR},K)
7: $N_{MR} = C_{hMR}$

Algorithm 1. Computing a logical form from an SAPT

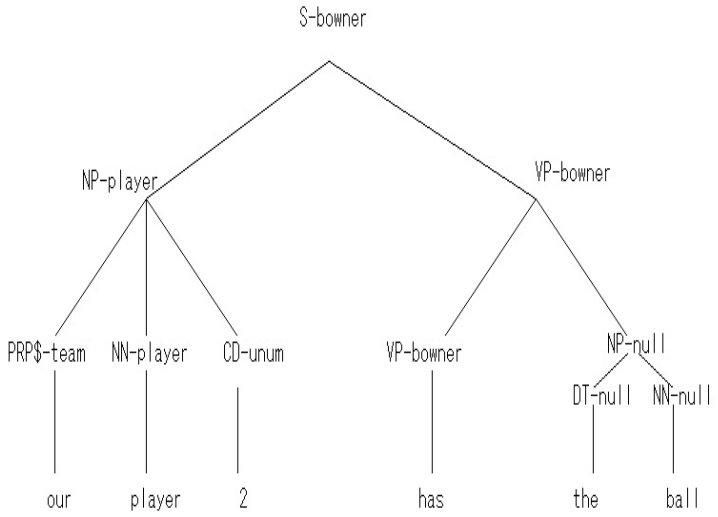

Figure 2. Example of a logical form extracted from a SAPT （R.Ge:2005）

3.2.2 Structured SVM for Semantic Parsing

We obtained a set of sentences and their SAPT representation which are used for semantic parsing problem. As discussed in (Tsochantaridis, et.al., 2004), the major problem to apply the structured SVM is to implement the feature mapping $\psi(x, y)$, the loss function $\Delta(y_i, y)$, as well as the maximization algorithm.

3.2.3 Feature mapping

For semantic parsing, we can choose a mapping function to get a model that is isomorphic to a probabilistic grammar in which each rule within the grammar consists of both a syntactic rule and a semantic rule. Each node in a parse tree *y* for a sentence *x* corresponds to a grammar rule g_j with a score w_j. All valid parse trees *y* for a sentence *x* are scored by the sum of the w_j of their nodes, and the feature mapping $\psi(x, y)$ is a history gram vector counting how often each grammar rule g_j occurs in the tree *y*. The example shown in Figure 3 clearly explains the way features are mapped from an input sentence and a tree structure.

3.2.4 Loss function

In semantic parsing, a parse tree that differs from the correct parse tree in only a few nodes should be treated differently from a parse tree that is radically different. The correctness of the predicated parsed tree is measured by its F_1 score, in which precision and recall are calculated based on the overlap of nodes between trees. We used the F_1 score as well as the zero-one loss for structured support vector learning.

3.2.5 Maximization algorithm

Note that the learning function can be efficiently computed by finding the structure $y \in Y$ that maximizes $F(x; y; w) = \langle w, \delta\psi_i(y) \rangle$ via a maximization algorithm. Typically we used the variant of CYK maximization algorithm which is similar to the one for the syntactic parsing problem (Johnson, 1999).

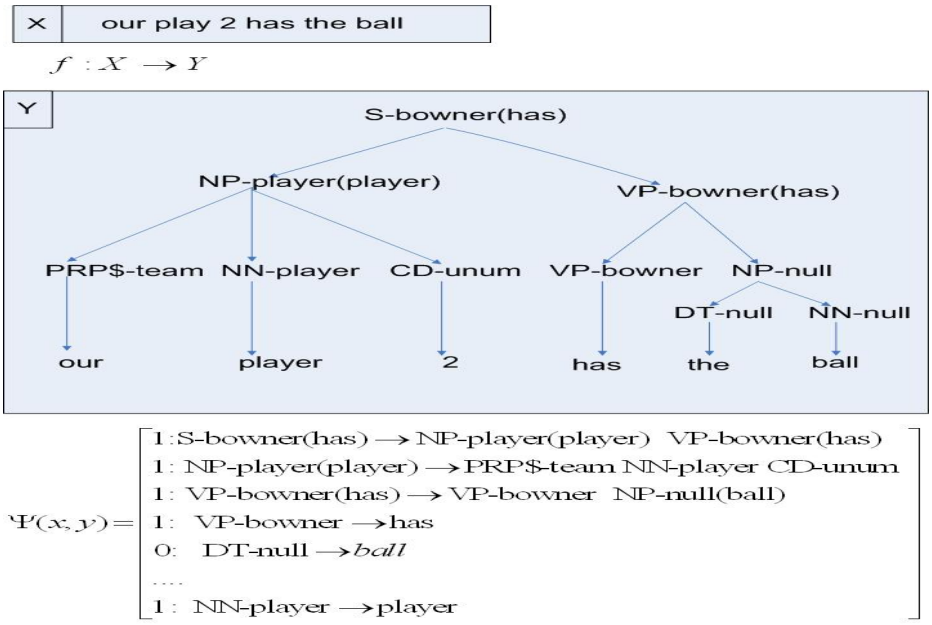

Figure 3. Example of feature mapping using tree representation

4. Experimental Results

We used the CLANG corpus which is the RoboCup Coach Language (www.robocup.org). In the Coach Competition, teams of agents compete on a simulated soccer field and receive advice from a team coach in a formal language. The CLANG consists of 37 non-terminal simples and 133 productions; the corpus for CLANG includes 300 sentences and their structured representation in SAPT (R.J.Kate, et.al., 2005). Below are some sample annotated statements from this corpus:

If player 4 has the ball, it should pass the ball to player 2 or 10.
((bowner our {4}) (do our {4} (pass {2 10}))))
No one pass to the goalie.
((bowner our {0}) (dont our {0} (pass {1}))))
If players 9, 10 or 11 have the ball, they should shoot and should not pass to players 2-8.
((bowner our {9 10 11}) (do our {9 10 11} (shoot)) (dont our {9 10 11} (pass {2 3 4 5 6 7 8})))

We used the standard 10-fold cross validation test for evaluating the method. NL test sentences were first tagged to a sequence of semantic tags, and those sequences were used to build trees in the form of SAPT via a structured support vector machine [1], then the MR's were built from the trees. We evaluated the accuracy of tagging a NL sentence to a sequence of semantic tags by computing the total number of correct semantic tags in comparison with the gold-standard. For this purpose, our CRF model obtained a very high result, with 98.5% accuracy after 24 iterations of training CRF by L-BFGs algorithm[2]. Figure 4 shows accuracy as a function of the number of L-BFGS training iterations. It indicates that the accuracy score increases smoothly with little fluctuation.

[1] http://svmlight.joachims.org/

[2] http://www.jaist.ac.jp/~hieuxuan/flexcrfs/flexcrfs.html

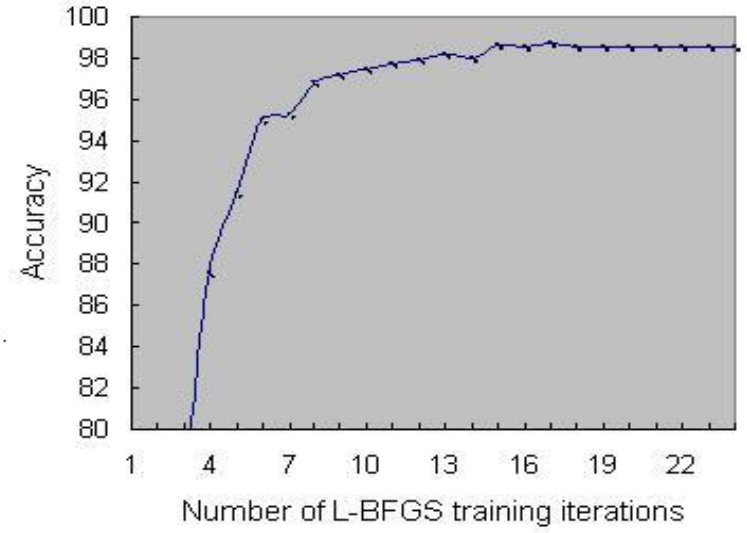

Figure 4. The relation of accuracy and iteration using L-BFGS estimation

To evaluate the method in parsing NL sentences to MR, we measured the number of test sentences that produced complete MR's, and the number of those MR's that were correct. For CLANG, an MR is correct if it exactly matches the correct representation, up to reordering of the arguments of commutative operators like "and". We used the exact the evaluation method presented in (R.J.Kate, et.al., 2005) in which the performance of the parser was then measured in terms of precision and recall as the formula below.

$$\text{Precision} = \frac{\#\text{correct-representation}}{\#\text{completed-representation}}$$

$$\text{Recall} = \frac{\#\text{correct-representation}}{\#\text{completed-representation}}$$

We conducted two experiments. The first was to investigate suitable parameters for structured support vector machines in the semantic parsing problem. The second experiment illustrated the performance of our method using the precision and recall measurements mentioned above. Table 2 shows the training accuracy when estimated on the CLANG corpus using a cross-validation test.

Table 2. Experiment results with CLANG corpus using cross-validation test with 20 iterations of training algorithm

Parameter	**Training Accuracy**	**F1-label brackets**	**Precision**	**Recall**
linear+ F-loss (Δ_S)	83.9	98.7	82.6	72.3
polynomial(d=2)+F-loss (Δ_m)	90.1	98.7	83.6	72.6
polynomial(d=2)+F-loss(Δ_S)	88.8	98.8	**84.9**	**74.3**
polynomial(d=2)+F-loss(Δ_m)	90.2	98.6	83.4	73.0
RBF+ F-loss(Δ_S)	86.3	98.8	83.8	74.3
SCSISSOR	unknown	unknown	89%	72.3
SILT	unknown	unknown	83.9%	51.3

It also shows the training and testing results with various parameters including linear kernel, polynomial kernel, and RBF kernel. Table 2 indicates that our method achieved high accurate

results and the polynomial kernel obtained better results in comparison with the linear kernel. In order to compare our result with previous work, we compared our method with the SCSISSOR system(R.Ge and R.J. Mooney, 2005). For the CLANG corpus, SCSISSOR obtained approximately 89% precision and 72.3% recall while on the same corpus our method achieved a better recall and slightly lower precision. In addition, with the second polynomial kernel, we obtained 84.9% precision and 74.3% recall, respectively.

We also compared our method with the SILT system (R.J. Kate, et.al., 2005). SILT achieved approximately 83.9% precision and 51.3% recall. Note that those figures for precision and recall was described clearly in (R.J. Kate, et.al., 2005) and it showed approximately this precision and recall of their method in this paper. Summarily, our method achieved the best recall result and a high precision on CLANG corpus. We also employ an evaluation method that is mainly used in syntactic parsing (Johnson, 1999) by computing the F1-label brackets. Table 2 shows that our method obtained a sufficiently high performance. This clearly indicates that the output of the proposed system is highly correlated with the gold-standard result. We believe that when the size of training corpus becomes large, our method could beneficially achieve improvement results in comparison with our current system.

5. Conclusions

This paper presents a novel application of a support vector machine to semantic parsing by employing it on the corpus of sentences and their representation in logical form. Experimental results on the CLANG corpus have demonstrated that our method achieves a very good performance in comparison with previous work. We thus can confidently conclude that the structured support vector models are suitable to the problem of semantic parsing. We also provide a semantic tagging tool with a very high accuracy by using a linear CRF model that can be beneficially used as preprocessing for the semantic parsing problem. Future work will be focused on extending our method to a version of SVM semi-supervised learning that can efficiently lean by using labeled and unlabeled data.

6. Acknowledgements

We would like to thank Rohit Kate and Raymond Mooney for their help with obtaining the CLANG data sets. The work on this paper was supported by a Monbukagakusho 21st COE Program and Japanese Telecommunication Advancement foundation.

7. References

Allen, J. F. 1995. Natural Language Understanding (2nd Edition). Mento Park, CA Benjaming/Cumming, 1995.

Mao Chen, Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, JohaKummeneje, Itsuki Noda, Oliver Obst, Patrick Riley, Timo Steffens, Yi Wang, and Xiang Yin. 2003. Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at http://sourceforge.net/projects/sserver/.

M. Johnson (1999). PCFG models of linguistic tree representation. *Computational Linguistics.*

R. Ge and R.J. Mooney (2005). A Statistical Semantic Parser that Integrates Syntax and Semantics, *Proceedings of the Ninth Conference on Computational Natural Language Learning*, Ann Arbor, MI, pp. 9--16, June 2005.

R.J. Kate, Y.W. Wong, R. Ge, and R.J. Mooney (2005). Learning to Transform Natural to Formal Languages, In the AAAI05 proceedings pp. 1062--1068, July 2005.

Zettlemoyer and Michael Collins (2005). Learning to Map Sentences to Logical Form: Structured

Classification with Probabilistic Categorical Grammars, *in the proceedings UAI 05.*

Miller, et al. (1996): A fully statistical approach to natural language interfaces. Proceedings of the *34th Annual Meeting of the Association for Computational Linguistics*, 55-61, 1996.

J.R. Mooney, 2004: Learning semantic Parsers: An Important but Under-Studied problem AAAI *2004 Spring Symposium on Language Learning: An Interdisciplinary Perspective, pp. 39-44, Stanford, CA. March.*

L.R. Tang (2003): Integrating Top-down and Bottom-up Approaches in Inductive Logic Programming: Applications in Natural Language Processing and Relation Data Mining. Ph.D. Dissertation, Department of Computer Sciences, University of Texas, Austin, TX.

J. Lafferty, A. McCallum, and F. Pereira (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. *In proc. ICML, pages 282--289.*

D. Liu and J. Nocedal (1989). On the limited memory BFGS method for large-scale optimization. *Mathematical Programming*, 45:503--528.

F. Sha and F. Pereira (2003). Shallow parsing with conditional random fields. *In Proc. HLT/NAACL*, 2003.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun (2004). Support Vector Machine Learning for Interdependent and Structured Output Spaces, *Proceedings of 21st International Conference on Machine Learning*, Banff, Canada.

V. Vapnik 1998: Statistical Learning theory. Wiley and Sons Inc.

J.M. Zelle and R.J. Mooney (1996): Learning to parse database queries using inductive logic programming. *In Proceedings of the Third teen National Conference on Artificial Intelligence, AAAI-96*, 1050-1055.

L.Rabiner (1989). A tutorial on hidden markov models and selected applications in speech recognition. *In {Proc. the IEEE}*, 77(2):257--286.