A new approach to syntactic annotation

Noguchi Masaki, Ichikawa Hiroshi, Hashimoto Taiichi and Tokunaga Takenobu

Department of Computer Science, Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo, 152-8550, JAPAN
{mnoguchi, ichikawa, taiichi, take}@cl.cs.titech.ac.jp

Abstract

Many systems have been developed for creating syntactically annotated corpora. However, they mainly focus on interface usability and hardly pay attention to knowledge sharing among annotators in the task. In order to incorporate the functionality of knowledge sharing, we emphasized the importance of normalizing the annotation process. As a first step toward knowledge sharing, this paper proposes a method of system initiative annotation in which the system suggests annotators the order of ambiguities to solve. To be more concrete, the system forces annotators to solve ambiguity of constituent structure in a top-down and depth-first manner, and then to solve ambiguity of grammatical category in a bottom-up and breadth-first manner. We implemented the system on top of eBonsai, our annotation tool, and conducted experiments to compare eBonsai and the proposed system in terms of annotation accuracy and efficiency. We found that at least for novice annotators, the proposed system is more efficient while keeping annotation accuracy comparable with eBonsai.

1. Introduction

In the last two decades, statistical methods using large scale corpora have become mainstream in natural language processing research. Especially, syntactically annotated corpora play important roles such as training data for probabilistic parsers and test sets for parsers. Creating syntactically annotated corpora by hand requires a lot of time and human resources. To remedy this problem, there have been many attempts to build support systems to create annotated corpora.

Some of such support systems enable annotators to manipulate syntactic trees directly and freely by providing a graphical user interface. This type of system was used to create Penn Treebank Corpus (Marcus et al., 1994) and Negra Corpus (Skut et al., 1997). The system used to create Titech Corpus (Noro et al., 2005), eBonsai (Ichikawa et al., 2005), does not allow annotators to manipulate syntactic trees directly. Given candidates of syntactic trees as an output of a parser, eBonsai allows annotators to choose a correct alternative at ambiguous tree nodes.

What is common among the previous systems is that their attention was focused on providing a user-friendly interface for manipulating syntactic trees. What is missing in the previous systems is *supporting decision-making in choosing a correct syntactic tree and knowledge sharing among annotators*. These missing features have been compensated for by preparing annotation manuals and assuming that all annotators completely understand them. When shifting our focus on the aspect of cooperative work in corpus annotation, we found it is important to introduce a mechanism to support these missing features to reduce annotators' burden and improve the quality of the corpus.

Related to this issue, there has been much work in the field of computer supported cooperative work. For example, Kadowaki and Shikida proposed a system to share know-hows during a process of a certain task (Kadowaki and Shikida, 2002). In their system, a problem encountered by a user and its solution are recorded in a database together with its context, that is, the time point the problem occurred and the operation history up to that point. When another user comes to a certain context, the system retrieves the similar contexts in the database and suggests probable problems and their solutions to the user.

When we apply their method to corpus annotation, we need to formalize a workflow so as to define contexts easily which are retrieval keys of the database. Unfortunately, the past annotation systems allow annotators too much freedom in operation order to define contexts in a workflow. To normalize the context, the annotation workflow has to be controlled by the system.

In this context, this paper proposes a method of creating syntactically annotated corpora in which the system controls annotators' operation so as to normalize the annotation workflow. We conducted experiments to compare accuracy and efficiency between an existing system and the proposed system. Despite less freedom of operation, annotators' performance with the proposed system was comparable to eBonsai. We expect if we would implement knowledge sharing support on eBonsai, the annotators performance would be improved.

2. eBonsai: A system for creating syntactically annotated corpora

This section describes the existing corpus annotation system eBonsai which was developed for creating syntactically annotated corpora (Ichikawa et al., 2005). We develop a new annotation system on top of eBonsai.

Figure 1 shows the overview of creating syntactically annotated corpora by using eBonsai.

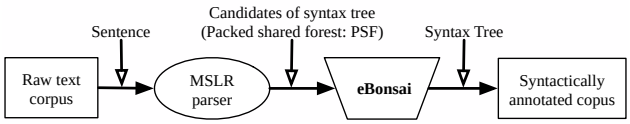

Figure 1: Corpus creating process eBonsai

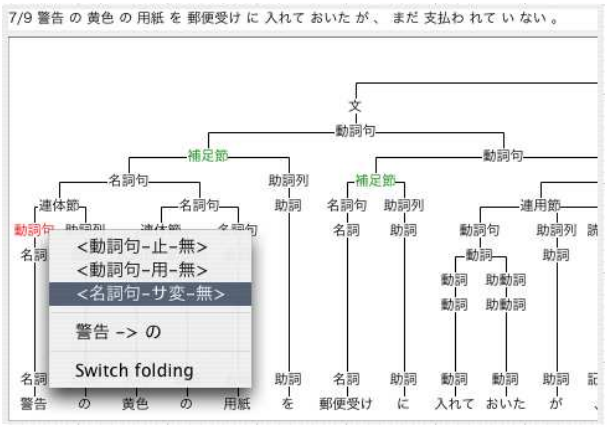

Figure 2: Interface of eBonsai

The flow of creating the corpus is as follows:

1. select a sentence from a raw text corpus
2. analyze the sentence with the MSLR parser
3. choose a correct syntax tree from the output of the parser by using eBonsai
4. store the correct syntax tree in the syntactically annotated corpus

The MSLR parser (Shirai et al., 2000) outputs candidates of syntactic trees which is represented as a packed shared forest (PSF) (Tomita, 1986). Annotators manually choose a correct syntax tree from the PSF.

The current eBonsai interface shows a single syntax tree at a time among the output of the parser. Figure 2 shows a screenshot of the eBonsai interface. Annotators solve ambiguities of nodes to narrow the number of candidate trees down to a correct one. Ambiguous nodes are colored in the interface which give annotators clues to work on. Since we adopt phrase structure grammars, there are two types of ambiguities: ambiguity in constituent structure and ambiguity in grammatical category. These ambiguities are denoted in different colors.

Annotators choose any colored node and select a correct choice in the pop up menu. When an ambiguity is solved, dependent ambiguities are automatically solved as well. There is no restriction on the order of ambiguous nodes to solve. It is entirely up to the annotators.

3. Controlling the annotation process

As we mentioned in section 1., introducing the know-how database can help sharing knowledge of problem solving among participants of a cooperative work. The key of the database is a context of the work process, that is, the time point a problem occurred and the operation history up to that point, and the information to retrieve is a problem description and its solution in that context. By consulting the database, the system can suggest probable problems and their solution automatically to the annotators when they come across the similar context in their work process. In order to increase the chance of retrieving similar cases, the database keys should be normalized as much as possible.

When applying this idea to creating syntactically annotated corpora, normalizing the order of solving ambiguities or building a partial structure becomes crucial. In the past systems supporting corpus annotation, annotators have freedom in the order of annotating information and that makes it difficult to find similar contexts in the annotation process. This fact leads us to the idea of controlling the annotation process by the system. We will incorporate the idea of system initiative annotation into eBonsai which was described in section 2..

3.1. The order of disambiguation

As mentioned in section 2., eBonsai asks annotators to solve ambiguities in the parsing result of a sentence. Since we adopt phrase structure grammars, we have two types of ambiguities: ambiguity in constituent structure and ambiguity in grammatical category. eBonsai highlights these different types of ambiguities in different node colors in a syntactic tree through a graphical user interface. The order of ambiguities to solve is entirely left to the annotators.

In the proposed method, the system forces annotators to solve ambiguities in constituent structure first, then ambiguities in grammatical category. We can reduce the diversity of the annotation process by fixing the order of disambiguation in this way.

Since ambiguities in constituent structure have dependency with each other, solving an ambiguity might implicitly disambiguate other ambiguities. Such implicit disambiguation is performed automatically by the system. Considering the order of solving ambiguities in constituent structure, solving ambiguities in lower nodes in a tree might delete a correct option in upper nodes even though the choice in the lower node is locally correct. We have encountered such cases in our past experience in creating Titech Corpus (Noro et al., 2005). Therefore the system forces annotators to solve ambiguities in constituent structure in top-down and depth-first manner.

On the other hand, ambiguities in grammatical category are solved in bottom-up and breadth-first manner, since lexical information of words provides important clues to decide grammatical categories.

3.2. An example

We explain the proposed method using an example below.

"They are flying planes with radio control."

First, the system constructs a packed shared forest (PSF) without grammatical category label from the output of the parser. Figure 3 shows the initial PSF of the example sentence. A rectangle enclosing multiple nodes represents a packed node denoting an ambiguity in constituent structure.

The order of solving ambiguity in constituent structure is calculated by traversing this PSF. A packed node is

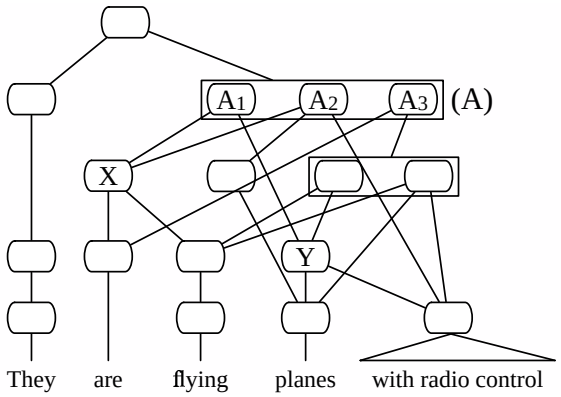

Figure 3: The initial PSF

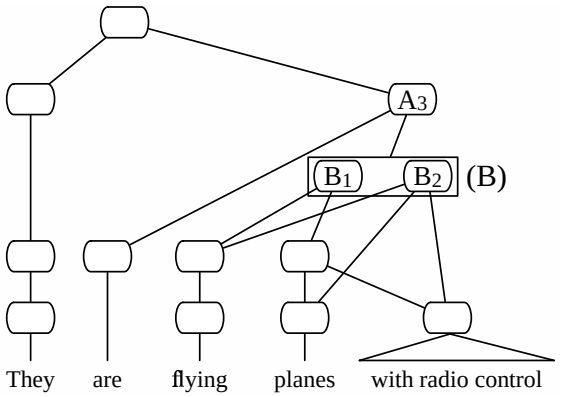

Figure 6: The PSF after choosing choice 3

searched from the root node in top-down and first-order manner. In this example, packed node (A) is found first. The system presents annotators the choices at this packed node and asks them to choose a correct one. When presenting the choices, the system shows segmented sentences instead of complicated syntactic trees as shown in Figure 4. Choice 1, 2 and 3 in Figure 4 correspond to node A_1, A_2 and A_3 in Figure 3 respectively.

1.	are flying planes with radio control
2.	are flying planes with radio control
3.	are flying planes with radio control

Figure 4: Presentation of options at packed node (A)

Figure 5: Interface of the proposed method

In each choice, each segment in a line corresponds to a word sequence which is dominated by the immediate child of a choice node in the packed node. For example, node A_1 has two immediate children X and Y and they dominate the word sequence "are flying" and "planes with radio control" respectively. Thus, this process is similar to the immediate constituent analysis. Since packed node (A) dominates "are flying planes with radio control", the subject "They" is not shown in the choices of Figure 4. In the actual interface (shown in Figure 5), a whole sentence is always shown in the top line and the current target part is highlighted with a colored background.

At this choice point, selecting choice 3 leads to the PSF shown in Figure 6.

Searching for a packed node continues in a top-down and depth-first manner and packed node (B) is found. The choices at node (B) are presented as node (A). This process continues likewise until all the packed nodes are disambiguated as shown in Figure 7.

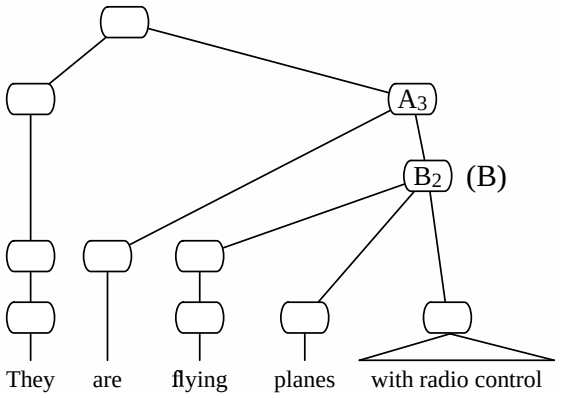

Figure 7: Uniquely determined constituent structure

Once a constituent structure is determined, the system constructs a packed shared forest with grammatical category labels in each node. Figure 8 shows the PSF at this stage. Similar to the disambiguation process in constituent structure, a rectangle enclosing multiple nodes represents a packed node. Note that ambiguity in grammatical category is packed in this case.

In disambiguating grammatical category, the system searches a packed node in bottom-up, breadth-first and left-to-right order. In this example (Figure 8), packed node (C) is found first. Possible grammatical categories are presented to annotators as shown in Figure 9. Each choice consists of several lines. The last line always displays one of possible grammatical categories with its child categories.

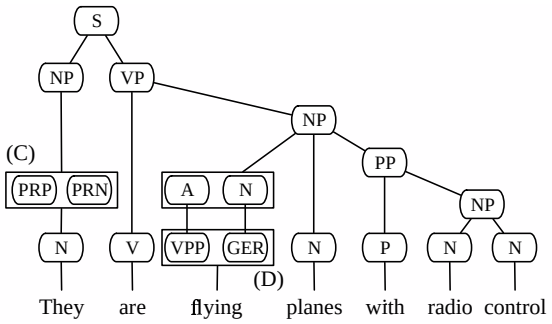

Figure 8: PSF for disambiguation of grammatical category

1.	They	⟨N⟩
		⟨PRP⟩→⟨N⟩
2.	They	⟨N⟩
		⟨PRN⟩→⟨N⟩

Figure 9: Choices at packed node (C)

In this case, both categories have one child category ⟨N⟩. The preceding lines display child categories and its dominating word sequence.

After selecting choice 1, the system traverses the PSF in bottom-up and breadth-first order to find the next packed node (D). This process continues likewise until all packed nodes are disambiguated like Figure 10.

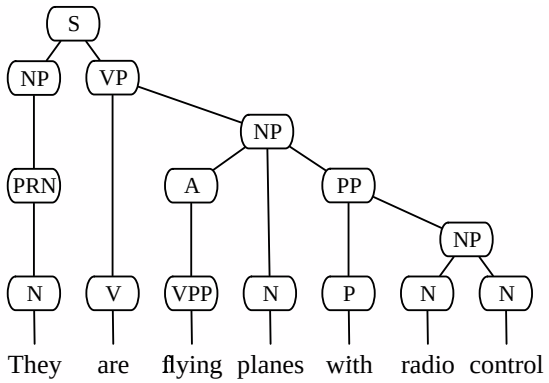

Figure 10: Fully disambiguated PSF

Note that during the entire process of disambiguation (annotation), the system always takes the initiative on deciding which ambiguity to solve next. This is an advantage for both the annotators and the system, because the annotators do not have to worry about choosing the ambiguity to solve next, and it is easy for the system to keep track of the annotators' behavior. The latter will be particularly advantageous when transferring the know-how of skilled annotators to the novice, since the annotators' behavior is normalized to a certain extent.

4. Experiments

We conducted experiments to compare accuracy and efficiency between eBonsai and the proposed system. The subjects are divided in two classes: a novice class and an expert class. The novice class includes 18 subjects who have no experience in corpus annotation. While the expert class includes 2 subjects who have experience in creating Titech corpus with eBonsai.

Table 1: Results of experts

	eBonsai	Proposed system
Accuracy [%]	78.5	74.3
Ave. time [sec]	42.5	35.4

Table 2: Result for novices

	eBonsai	Proposed system
Accuracy [%]	52.9	49.6
Ave. time [sec]	49.6	42.9

We build two test set each of which consists of 200 sentences. They are extracted from Titech Corpus containing about 20,000 syntactically annotated sentences. The extraction is performed basically at random only ensuring that the average numbers of trees of sentences are balanced among the test sets.

Since the expert class subjects have experience with eBonsai, both of them annotated a test set with eBonsai first, and another test set with the proposed system. To see the effect of difference of test set, two subjects used the different system in annotating the same test set.

The novice subjects were divided into two groups in terms of the order of systems, and the combination of the system and the test set. Ahead of the experiment, the novice subjects had a training session with 30 sample sentences which were not included in the test sets. In the training session, instructors first explained the goal of their task, and the subjects actually worked on the sample sentences. During the training, the subjects were allowed to ask the instructors any question. In the experiment, the subjects were instructed to complete the task by themselves.

Table 1 shows the results of experts in terms of the average accuracy and the average time to annotate a sentence.

For Experts using the two systems, eBonsai has a slightly higher accuracy. This might be because they have already had experience with eBonsai and are used to the system. eBonsai shows a lot of information such as labels and structures at one time. The proposed system, however, shows only the word sequences and the segmentations. This difference in the amount of information provided might cause the difference of accuracies. On the other hand, the proposed system is slightly better in the average time to annotate a sentence. The reason for this is that the annotators do not have to select which ambiguous node should be solved next with the proposed system. Additionally, it is a problem that the accuracy for experts is less than 80% regardless of which system is used. Since the reliability of the corpus is dependant on the constituency, at the stage of creating

syntactically annotated corpora the accuracy also has to be high. Since we have only two experts subjects, the result is not conclusive.

The same tendency is observed in the novice case as shown in Table 2. We conducted a t-test on accuracy and average time and found that there is no significant difference in accuracy but some difference in average time at $P = 0.01$.

Comparing the results of novices and experts, we find that accuracy is drastically different. This comes from the difference of knowledge about grammars based on which the subjects annotate the corpus. Our tutorial might not be enough to make the subjects understand the grammars and the annotation criteria.

We could not observe any significant effect of the order of systems to use, and the combination of the system and the test set for the novice subjects.

To summarize, we can mildly claim that at least for novice annotators, the proposed system is more efficient while keeping annotation accuracy comparable with eBonsai.

We evaluated the performance of the probabilistic parsing models (PCFG and PGLR (Inui et al., 2000)) with the same test sets. All sentences in Titech Corpus except for those used in the experiments were used for training the probabilistic models. The syntactic tree with the highest probability were considered as output for each sentence. The result is shown in Table 3.

Table 3: Performance of probabilistic models

	PCFG	PGLR
No. of correct parses	168	196
Accuracy [%]	42.0	49.0

It turned out that the performance of the probabilistic models are comparable with average novice annotators. We suspected several novice subjects whose performance is lower than the probabilistic models.They might not well understand the goal of the task. There were 12 subjects whose performance was worse than the PGLR model, and 7 subjects worse than the PCFG model.

We devided novice subjects to 4 groups those whose accuracies are lower and higher than the probabilistic model with both systems respectively and examined the tendency.

- <**PCFG**: 7 novices whose accuracies are lower than PCFG model.
- >**PCFG**: 11 novices whose accuracies are higher than PCFG model.
- <**PGLR**: 12 novices whose accuracies are lower than PGLR model.
- >**PGLR**: 6 novices whose accuracies are higher than PGLR model.

Table 4 and Table 5 show the accuracy and the average time respectively.

Table 4: Accuracy of each group(%)

	<PCFG	>PCFG	<PGLR	>PGLR
eBonsai	41.2	60.4	47.5	63.7
Proposed system	35.1	58.8	41.8	65.2

Table 5: Ave. time of each group (sec.)

	<PCFG	>PCFG	<PGLR	>PGLR
eBonsai	43.5	53.4	49.7	49.3
Proposed system	35.9	47.4	42.3	44.3

Regardless of the systems, the novices whose accuracies are lower tend to work in short time. Comparing to the experts' work time, the novices whose accuracies are higher work in long time. Counterwise, the novices whose accuracies are lower than the probabilistic models might be less understood because of being scarce of the tutorials. We thought that the time was a one of the keys to decide whether the novices were learned enough to acquire know-hows and able to select a correct syntax tree or not.

5. Conclusion and future work

To incorporate functionality of sharing knowledge among annotators in the environment for creating syntactically annotated corpora, we emphasized the importance of normalizing the annotation process. As a first step, this paper proposed a method of system initiative annotation in which the system suggests annotators the order of solving ambiguities. To be more concrete, the system forces annotators to solve ambiguity of constituent structure in a top-down and depth-first manner, and then to solve ambiguity of grammatical category in a bottom-up and breadth-first manner. We implemented the system and conducted the experiments to compare an existing system and the proposed system in terms of annotation accuracy and efficiency. We found that at least for novice annotators, the proposed system is more efficient while keeping annotation accuracy comparable with eBonsai.

Some novice annotators performed worse than probabilistic parsing models, probably due to a lack of the grammatical knowledge and the annotation criteria. Introducing a knowledge sharing functionality could be helpful not only for this sort of annotators but also for all annotators in terms of accuracy. As a future work, we are planing to integrate knowledge sharing functionality into the proposed system and improve the user interface to increase the accuracy.

6. References

H. Ichikawa, M. Noguchi, T. Hashimoto, T. Tokunaga, and H. Tanaka. 2005. eBonsai: An integrated environment for annotating treebanks. In *The 2nd International Joint Conference on Natural Language Processing.*, Oct.

K. Inui, V. Sornlertlamvanich, H. Tanaka, and T. Tokunaga, 2000. *Probabilistic GLR Parsing.In Advances in Proba-*

bilistic and Other Parsing Technologies, chapter 5 Probabilistic GLR Parsing, pages 85–104. Kluwer Academic Publisher.

C. Kadowaki and M. Shikida. 2002. Effective organizational memory sharing by a process-linking approach. In *Proc. of Int'l Conf on Information and Knowledge Sharing*, pages 39–44.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1994. Building a large annotated corpus of English: The penn treebank. *Computational Linguistics*, 19(2):313–330.

T. Noro, C. Koike, T. Hashimoto, T. Tokunaga, and H. Tanaka. 2005. Evaluation of a japanese cfg derived from a syntactically annotated corpus with respect to dependency measures. In *The 5th Workshop on Asian Language Resources*, pages 9–16, Oct.

K. Shirai, M. Ueki, T. Hashimoto, T. Tokunaga, and H. Tanaka. 2000. Mslr parser tool kit — tools for natural language analysis. *Special Interest Group of Natural Language Processing (IPSJ-SIGNL)*, 7(5):93–112.

W. Skut, B. Krenn, T. Brants, and H. Uszkoreit. 1997. An annotation scheme for free word order languages. In *Proceedings of the Fifth Conference on Applied Natural Language Processing ANLP-97*, Washington, DC.

M. Tomita. 1986. Efficient parsing for natural language.