An Annotated Corpus Management Tool: ChaKi

Yuji Matsumoto*, Masayuki Asahara*, Kiyota Hashimoto†, Yukio Tono‡, Akira Ohtani◇, Toshio Morita§

* Nara Institute of Science and Tachnology
8916-5 Takayama, Ikoma, Nara 630-0192 Japan
{matsu, masayu-a}@is.naist.jp

† Osaka Prefectural University
hash@lc.osakafu-u.ac.jp

‡ Meikai University
y.tono@meikai.ac.jp

◇ Osaka Gakuin University
ohtani@utc.osaka-gu.ac.jp

§ Sowa Giken Corp.
morita@sowa.com

Abstract

Large scale annotated corpora are very important not only in linguistic research but also in practical natural language processing tasks since a number of practical tools such as Part-of-speech (POS) taggers and syntactic parsers are now corpus-based or machine learning-based systems which require some amount of accurately annotated corpora. This article presents an annotated corpus management tool that provides various functions that include flexible search, statistic calculation, and error correction for linguistically annotated corpora. The target of annotation covers POS tags, base phrase chunks and syntactic dependency structures. This tool aims at helping development of consistent construction of lexicon and annotated corpora to be used by researchers both in linguists and language processing communities.

1. Introduction

Recent progress in natural language processing has made it real to implement highly accurate natural language processing tools such as part-of-speech (POS) tagging, phrase and named entity chunking, and syntactic parsing, which are now being used in various natural language processing applications. Large scale annotated corpora are very important not only for linguistic research but also for improving accuracy of practical language processing tools since most of the practical tools are now machine learning-based systems which rely on accurately annotated corpora. On the other hand, it is well-known that even widely used annotated corpora such as the Penn Treebank(Marcus et al., 1993) still include a number of annotation errors. To develop highly accurate annotated corpora, it is indispensable to have a supporting environment to maintenance annotated corpora so as to find and correct errors remaining in manually or automatically annotated corpora.

This paper presents an annotated corpus management tool named, ChaKi, which has been developed for last three years as a project supported by Japan Society of the Promotion of Science. This tool aims at helping users in consistent construction of annotated corpora and lexicons that are to be used by researchers both in linguiscs and language processing communities. The sysmtem will be distributed as free software[1].

2. Functions of ChaKi: Overview

Figure 1 shows the overall configuration of ChaKi. Annotated corpora may be manually or automatically annotated, and are imported into the database together with a dictionary if the user has his/her own dictionary. If the user does not have any specific dictionary, all the words contained in the annotated corpus constitute the dictionary. ChaKi works as an interface to the database (Currently, MySQL[2] is used for the database) and provides various functions such as searching, statistic calculation, and error correction. This section briefly overviews those functions.

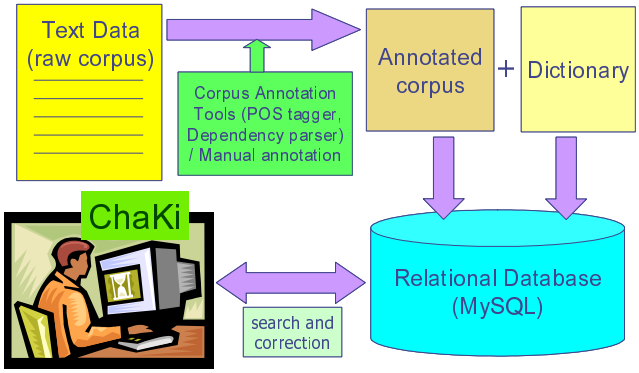

Figure 1: Configuration of ChaKi

1. Coordination between annotated corpora and dictionary: The words in annotated corpora are represented as pointers to the dictionary entries. This help to keep consistency between the corpora and the dictionary in such a way that all the words in the corpus should be defined in the dictionary and correction in the corpus is always done by selecting appropriate entries in the dictionary.
2. Variation of annotation: Part-of-speech tags, base phrase chunks and syntactic dependency structures (dependency between words or chunks) are handled. Multi-word expressions can be defined in the dictionary together with their constituent word information. Search can be performed either on multi-word expressions or on the constituent words. Bibliographic information of

[1] http://chasen.naist.jp/hiki/ChaKi/

[2] www.mysql.com/

document of sentences are to be annotated at the sentence level.

3. Search: Three modes of search are possible. They are string search, word search, and dependency structure search.
4. Browsing: KWIC presentation of retrieved sentences, and tree representation of dependency structure trees are provided.
5. Error correction: Functions of correcting segmentation, word information, and dependency structure errors are provided.
6. Statistic calculation: Basic statistic calculations such as word frequencies and collocation counts in fixed size windows are provided.
7. Multilinguality: The system is designed as language independent, and now handles Japanese, Chinese and English corpora.

3. Detailed Description of Functions

This section describe each of the functions of ChaKi in detail.

3.1. Coordination between corpora and dictionary

When an annotated corpus is imported into the database, words in the corpus are represented as pointers to the dictionary entries. When a word that does not have any entry in the dictionary appears in the corpus, that word is tentatively registered in the dictionary. Therefore, the corpus is usually not allowed to contain any words that are not defined in the dictionary. When such words appear, they are clearly marked as exceptional. Exisiting annotated corpora often includes impossible annotations. For example, the most frequently used Penn Treebank includes some occurrences of "have" tagged as "VBD" (past tense of a verb) or VBN (past paticiple). Such errors are easily detected when importing the corpus into the database. Moreover, other information described in the dictionary is automatically added to the corpus when the corpus is read in the database. For example, our current English dictionary describes the base forms for all the inflected forms of words, so that all the words in the corpus automatically receives its base forms even in case the original corpus doesn't have any such information.

3.2. Variation of annotation

The types of annotation can be divided into two: One is the annotation within sentences, and the other is those at the sentence level.

3.2.1. Annotation within sentences

The current system allows the following annotation within sentences: Word (morpheme), base phrase, and dependency structure. Word information includes the form of appearance, pronunciation[3], POS tag, base form, inflection type and inflection form. This is the maximum set of information for a word, and each corpus can specify which of them are included in the database. The dictionary can describe the constituent words for a multi-word expression, whose definition is done by pointers in a recursive manner. For example, the multi-word expression "in respect of" is defined as a preposition, and its constituents are also defined as a preposition, a common noun and a preposition by pointing them within the dictionary.

When the base phrase chunk and the syntactic dependency relation between chunks are annotated, those are also registered in the database. As for those annotation in Japanese corpora, Japanese morphological analyser ChaSen(Matsumoto, et al., 2003) and Japanese dependency parser CaboCha(Kudo and Matsumoto, 2002) are used. For English, we assume the Penn Treebank or British National Corpus format.

3.2.2. Annotation at sentences: Bibliographic information

Bibliographic information of the corpus (the name of the corpus, the authors' name(s), etc) and attribute information of sentences (speaker, contextual information, etc) are annotated at the sentence level. They have no structural format and are in a simple character string form, which are retrieved by string level partial matching. Moreover, Users may maintain their corpora not by a single file but by a set of files within a heirarchy of folders classifying them so that the folder structure represents bibliographcal information related to those files. All of those information, the sequence of the folder names, is allocated at the sentence level as well.

While they are not considered in the current system, there should be more sentence level or extra-sentence information: Discourse relaton between sentences is one of such information. Also, links between reference expression such as anapora and their antecidents are another example of extra-sentence information, which the current system does not cover.

3.3. Search functions

In the search component, the unit for search is a sentence in all the following modes of search functions.

3.3.1. Surface character strings

All occurrences of character strings are searched in this mode. Regular expressions can be used and the results are displayed in the KWIC format.

3.3.2. Word sequences

When the sentences are POS tagged, any lexical information is used to describe patterns of a word sequence. The lexical information includes surface form, POS tag, base form of a word, and in the case of Japanese, other information such as pronunciation, inflection type and inflection form can also be specified. Regular expressions can be used in specifying any of the word information. Figure 2 shows an example of word sequence search, in which a sequence consistng of three words are specified. The figure shows a Japanese word sequence search pattern. Each box corresponds to a word, in which the top row specifies the surface form and the fourth row specifies the POS tag.

[3]For Japanese, other than pronunciation, there is a slot for "*reading*" as well, which stands for the entry of the word in ordinary dicitonaries

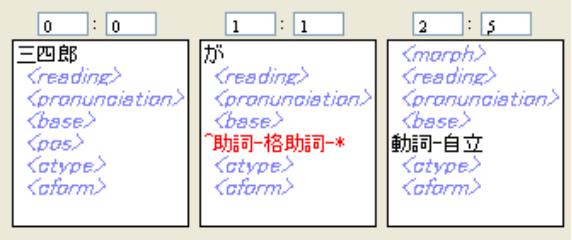

Figure 2: Example of Word Sequence Search

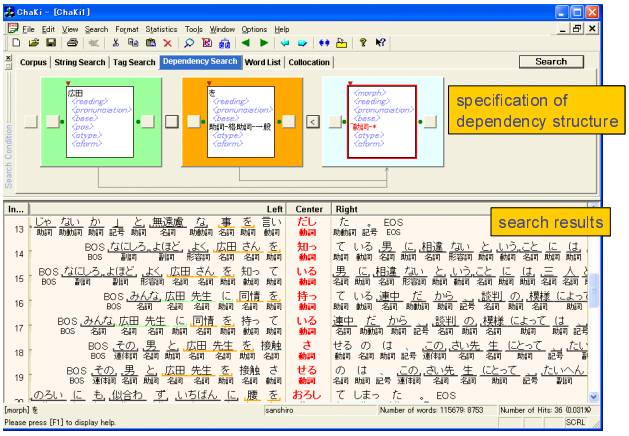

Figure 3: Dependency Structure Search in ChaKi

3.3.3. Word Dependency structure

When the senteces are parsed in dependency structure, they are searched for by specifying a partial dependency structure. Any sentences that include the specified dependency structure are retrieved. Figure **??** shows a snapshot of dependency structure search. The query is depicted in the upper part of the interface. Each shaded box shows a base phrase chunk and a small white box in a chunk specifies a word, in which various information of the word can be described just like the word sequence search. Arrows between chunk boxes describe the syntactic dependency relation between them. This example shows a Japanese case, and all the sentences that include the specified dependency structure are retrieved and shown in the KWIC format as shown in the lower part of the Figure.

In the case of English, the system supposes that the syntactic information of a sentence is represented as a word or base phrase dependency structure where the dependency relation holds between a word/phrase and its modifying word/phrase. In the current system, phrase structure trees of Penn Treebank are transformed into dependency trees using the head rules that specify the head constituent in phrase structure rules(Collins, 1996).

Multiword expressions can be retrieved as their whole form (with/without their POS tags) or by their constituents. For example, if ”in short” is defined as an idiom and is annotated as an adverb, it is retrieved as a single adverb. On the other hand, if its constituents are defined as ”in/IN, short/JJ” (IN stands for a preposition and JJ stands for an

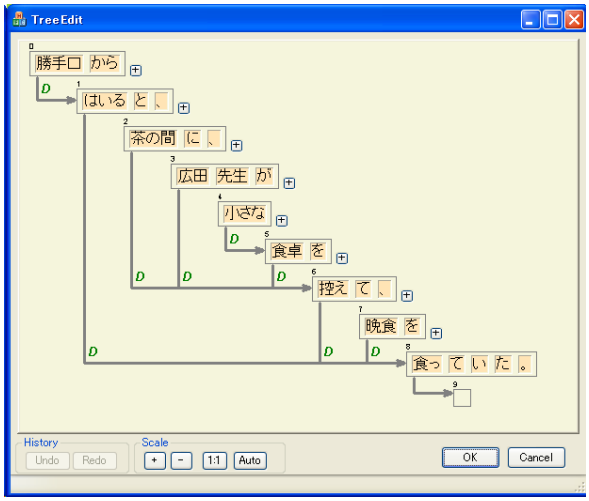

Figure 4: Dependency Tree

adjective in the Penn Treebank POS tag set), this expression can also be retrieved by those constituent words as well.

3.4. Browsing of search results

Since the search is done on sentence-wise, the search results are shown on sentence-wise, too. As seen in Figure 3, matched sentences are shown in the KWIC format. In the search query such as Figure 2 and Figure 3, only one word is specified as the focus word and is presented as the center word in the KWIC in all modes of the search functions. In word sequence and dependency structure searches, any specified information can be associated with a word as shown in Figure 3, where the POS tag is shown below its corresponding word.

In case of dependency tree search, the user can pick up one of the obtained sentences from the KWIC, and trigger the TreeEditor, which presents the dependency tree as shown in Figure 4. Each row corresponds to a base phrase chunk, in which a sequence of words are presented in small boxes. Arrows between chunks show syntactic dependency relations.

As explained in the preceding section, the corpus can be retrieved by multi-word expressions as well as their constituent words. The browsing function has two modes, showing the sequence of words either in multi-word forms or in constituent words. In the multi-word mode, the constituents of multi-word expressions can be displayed in a separate window just similarly as dependency trees.

3.5. Statistic calculation

Some basic statistics can be calculated by the system. For example, if the user is interested in only the types and frequencies of the centered word in the word sequence search, he/she can select “Word Search” mode instead of “Tag Search” mode. Then, all the retrieved center words and their frequencies are presented in another table.

Another statistics provided by the system are collocations between the centered word and the surrounding words

within a specified context length window. Collocation may be simple frequencies, mutual information, or frequent word N-grams based on any information on words, such as surface word forms, base forms, POS tags, etc.

3.6. Error correction functin

Error correction function is one of the most important functions for developing accurately annotated corpora. Annotated corpora should be corrected when annotation errors are detected. Once an annotation error is found, it is often the case that errors of the same type retain in other parts of the same corpus. The search functions of the system are effectively used for detecting similar types of errors. And once the instances of the same error type are collected, the error correction module helps to issue a transformation rule so as to correct all the erroneous instances by one operation (currently this does not apply to dependency error correction).

There are two modes of error correction corresponding to the types of seach, the word sequence search and the dependency structure search. In the former, word segmentation errors and word related erros such as POS tag or inflection form errors are corrected. Correction of the word information is guided by a special interface, in which reselection of words is basically done for error correction. This is bacause the corpus is represented as a sequence of pointers to the dictionary entries. In the latter, base phrase chunking errors and dependency relation errors are corrected. For example, in Figure 4, any adjacent chunks can be merged into one, and one chunk can be divided into two, which enable any modification of segmentation. Furthermore, the arrows representing dependency relation can be modified, by switching the head of an arrow from one chunk to any other.

3.7. Other functions

3.7.1. Multilinguality

The system is designed as language independent and can be used with any language that is only accepted by MySQL. The current system is tested with Japanese, Chinese and English corpora.

3.7.2. Distribution

The system is and will be distributed as free software. For the database system we use MySQL(version 5.0), which is also free software. The database runs on various platform, but ChaKi's interface runs only on Windows.

3.7.3. Interface to language analysis tools

Any corpora annotated with POS or dependency structure can be imported to the system. At the moment, the corpus is annotated as either in the standard output formats of ChaSen and CaboCha. An interface to apply ChaSen and CaboCha to raw corpora is provided. Also, another interface is provided to import annotated corpora in ChaSen or CaboCha format into the database. For English corpora, we prepared a transformation program for corpora annotated in Penn Treebank or British National Corpus formats.

4. Conclusions

In this paper we presented our annotated corpus management system, ChaKi. The aim of the system is to develop an environment that helps users to construct, use and manage accurately annotated corpora. It is important not only to develop high performance natural language analysis tools but also to provide tools that search for erroneous patterns and correct such errors in annotated corpora. ChaKi provide various functions for these purposes.

In any languages, there are idiomatic or collocational expressions that are formed by multiple words. Especially in Japanese and Chinese, it is hard to define proper segmentation of sentences uniquely since definition of words may vary according to the grammar system or to applications. We decided to describe multi-word expressions and their constituent words in the dictionary and define the corpus as a sequence of pointers to the dictionary. By doing this, it becomes possible to search patterns either in multi-word expression or in constituent words. Also, we could provides two modes of displaying the search results accordingly.

Currently we work on three languages, Japanese, Chinese and English. We assume that extension of the target languages is easy since most of the tools are designed to be language independent.

The system will be distributed as a free and open source software. ChaKi and the interface to the database are developed in C, C++, Ruby and VisualC++.

5. Acknowledgements

We would like to thank our colleagues who helped and inspired us in developing the corpus management tool. We are very grateful to the former and current members of Computational Linguistics Lab at NAIST, especially to Kentaro Inui, Kazuma Takaoka, and Taku Kudo. Development of ChaKi is supported by JSPS Grants-in-Aid for Scientific Research (B) No.15300046.

6. References

M.J. Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. *ACL-96*, 184-191.

T. Kudo, Y. Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. *6th Conference on Natural Language Learning*, 63–69.

M. Marcus, B. Santorini and M. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Y. Matsumoto, et al. 2003. *Morphological Analysis System ChaSen 2.3.3 Users Manual*. NAIST Technical Report.