NAIST-IS-DT9761211

Doctor's Thesis

A Machine Learning Approach to Natural Language Processing

Masahiko Haruno

July 27, 1998

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

Doctor's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
DOCTOR of ENGINEERING

Masahiko Haruno

A Machine Learning Approach to Natural Language Processing*

Masahiko Haruno

Abstract

Over the past 50 years, researchers have made concerted efforts to construct practical natural language systems such as automatic machine translators. Although these systems are now commercially available and used, those with high performance are limited to simple processors (part-of-speech taggers and kana-kanji converters) or systems tuned for specialized domains. Why is it so difficult to construct practical systems for broader use? The most crucial barrier has been identified as the variety of *knowledge* required for language processing.

This thesis explores the machine learning approach to natural language processing in which the *knowledge* is automatically acquired by using a corpus currently available. In this thesis, I would not claim that machine learning is superior to and should replace conventional methods (nearly all of them done by human learning). Instead, my intention is to clarify what is appropriate and reasonable for machine learning and what is appropriate for human learning. In the following chapters, my ultimate objective is to promote a new approach to language engineering by gaining a deep understanding of the machine learning techniques.

Conceptually, the knowledge problem can be processed by the following three steps.

1. Enumerate linguistic features that potentially influence the target language task.

2. Select features and combine them into a form of rules.

*Doctor's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT9761211, July 27, 1998.

3. Determine preference parameters of rules.

Although these steps in fact are deeply interleaved, let us consider them sequentially from the machine learning viewpoint. For the first step, it is basically difficult for current computer programs to automatically enumerate feature types and their coding scheme from scratch. The machine learning approach to extracting features has been limited to the co-occurrence statistics of words and phrases. Although the task is very simple, it is useful for various types of language processing. The main decision in co-occurrence statistics is what range of expressions should be considered by the statistics. The more complex the expressions are, the more difficult it is for the system to induce them.

If we assume the existence of features to represent the target task, the second step is to construct rules by combining the features. The machine learning approach can produce a large-scale simple structure such as a decision tree and a hidden Markov model (HMM). Its great advantage is to produce rules by using global data distribution, while a human linguist depends on intuition and may resort to near-sighted rules. Thus, the most important factor in applying machine leaning techniques is to apply the necessarily and sufficiently simplest representation for the task.

The third step is to set the preference parameters. The machine learning approach has fundamental advantages because the optimization of preference parameters should be performed from a global point of view. If a human linguist sets the parameters only from a near-sighted perspective, the system would suffer from side-effects. Worse still, even if the linguist makes the greatest efforts to maintain the coherence of preference parameters, it gradually becomes difficult as the number of rules increases. The machine learning approach determines the preference parameters, for example, by using maximum likelihood estimation (MLE), which selects the parameters to maximize the likelihood of generating given data. The more data are available, the more effective the parameters are.

The only obstacle faced by the approach for the second and third steps is the limited amount of data available. Natural language is full of infrequent colloquial or exceptional expressions that cannot be handled by general rules, which deteriorates the system's overall performance. An effective learning scheme to cope with these expressions is needed.

Subsequent chapters describe concrete examples of machine learning techniques for natural language processing. Following the introduction in Chapter 1, Chapters 2 and 3 deal with unsupervised learning from bilingual (Japanese and English) corpora. Chapter 2 describes the sentence alignment task to find sentence correspondences in bilingual texts[1]. The most prominent features (the first step) for sentence alignment is word correspondences between two languages. The learning algorithm (the second and third steps) is based on the intuition that the more word correspondences a sentence pair contains, the more reliable the sentence correspondence is. The proposed alignment method employs word correspondences from both statistics and electronic dictionaries. By combining the two kinds of word correspondences, sentence alignment was performed with a higher precision than conventional methods. In addition, to cope with errors left to be modified by humans, we also introduce a sentence alignment environment, called BACCS (Bilingual Aligned Corpus Construction System), devised for human supervisors.

In Chapter 3, the sentence aligned corpora are then used to extract bilingual collocations by a novel word-level sorting method. The extracted collocations are important clues for machine translation and multi-lingual information retrieval systems. The feature used in the method is only part-of-speech tags. The learning method is simple occurrence statistics. According to the difficulty of learning, we differentiate two kinds of collocations: fixed and flexible collocations. Fixed collocations comprise a continuous sequence of words, while flexible collocations consist of more than two separate sequences of words. Although conventional methods focus on either noun phrases or flexible collocations, I would like to emphasize that the fixed collocation is an important class due for the following three reasons.

- Fixed collocations can be efficiently and precisely extracted by our word-level sorting method.

- Fixed collocations contain a broad range of expressions rather than noun

[1]Sentence correspondence is not one-to-one.

phrases.

- The construction for flexible collocations is efficient because a two-step combination of fixed collocations can reduce redundant computation.

Note that the three steps in Chapters 2 and 3 are very simple, reflecting that the problem settings are unsupervised.

By using the word-level sorting method, fixed collocations were extracted with a precision better than 80%. On the other hand, Performance for flexible collocations remained 36.8%. These results confirm that the two classes of collocations are completely different in terms of the learning feasibility.

Chapters 4 and 5 address the most important tasks of Japanese language processing: morphological analysis and dependency analysis. Morphological analysis segments an input sentence into a sequence of words annotated with part-of-speech tags. Because there are no explicit delimiters between words in the Japanese language, morphological analysis is the first step for almost all Japanese text processing. Dependency analysis determines the syntactic structure (modification relations of bunsetsu) of input sentences. In these two chapters, I would like to emphasize the importance of appropriate representations (hierarchical context trees and decision trees) and mistake-driven learning algorithm (boosting) that is sensitive to data distribution. By introducing these methods, the resulting morphological analyzer and dependency parser significantly outperformed conventional systems.

In addition, I will show in Chapter 5 that the combination of feature extraction by humans (the first step) and the machine learning approach (the second and third steps) is quite promising for future natural language engineering.

Chapter 6 describes a concrete application of corpus-based natural language processing. We will introduce an adaptive dictionary environment, called AIDA (A daptive and Integrated Dictionary Agent), in which several dictionaries (Japanese-English, English-Japanese, English-English), several corpora (bilingual and monolingual) and collocations (extracted by the method in Chapter 3) are integrated through a single graphical interface. When a user wants to write or read texts, particularly in foreign languages, his or her requirements are often not clear.

AIDA provides the user with an adaptive and flexible search mechanism; the system selects the expressions syntactically and semantically similar to the user's input. The similarity measure itself is learned through the interactions. Finally, Chapter 7 concludes the thesis and mentions future directions.

These chapters confirm that machine learning approach achieves good performance particularly when a target language task has a simple representation. Both in supervised and unsupervised settings, arbitrariness of the conventional rule compilation is removed by machine learning techniques. The objectiveness of the approach should play a more essential role in applying to larger-scale practical systems.

Keywords:

machine learning, bilingual corpora, sentence alignment, collocation, morphological analysis, dependency analysis

Acknowledgments

I would like to express my sincere appreciation to my supervisor, Professor Yuji Matsumoto of Nara Institute of Science and Technology, for his continuous encouragement and guidance. Since I was an undergraduate student at Kyoto University, he has guided me in adopting the basic attitude of a researcher and appreciating the joys of research, as well as the fundamentals of natural language processing. If I had not met him, I would have never pursued a research career.

I would also like to express my gratitude to the members of my dissertation committee: Professor Kiyohiro Shikano, Professor Toyoaki Nishida and Professor Yasuharu Den of Nara Institute of Technology for their valuable suggestions and helpful comments.

I am deeply indebted to Professor Makoto Nagao of Kyoto University for supervising my master thesis and for continuous encouragement. He guided me not only in the skills of conducting research but also in what is most important for research and for a researcher. Although I could not realize it at the time, I now consider working with him as a hilight of my research career.

Special thanks are due to my colleagues of the Nagao Labs. for their advice and discussions. In particular, I am indebted to Professor Yasuharu Den of NAIST for instructing me in the basic skills of conducting research, Professor Satoshi Sato of the Japan Advance Institute of Science and Technology, Professor Yuichi Nakamura of Tsukuba University, and Dr. Takehito Utsuro for their valuable comments. I am also indebted to Hitoshi Shimbo for his discussions in diverse domains and furthermore for his warm friendship.

The research reported in this thesis was conducted at the Communication Science Laboratories, NTT Corporation. I would like to express my sincere gratitude to Mr. Koichi Matsuda, the former director of the Communication Science Laboratories, for providing me an the opportunity to conduct this study and for his insightful comments.

My sincere appreciation also goes to Mr. Toshifumi Yamaki, Dr. Norihiro Hagita, Professor Tsukasa Kawaoka of Doshisha University and Professor Shigeo Kaneda of Doshisha University. Without their generosity and support, I would not have been able to complete the work.

Some works reported in this thesis were performed with members of the Communication Science Laboratories. I am further deeply indebted to Professor Satoru Ikehara of Tottori University, Mr. Yoshifumi Ooyama, Mr. Satoshi Shirai and Takefumi Yamazaki for their valuable cooperations and stimulating discussions.

I have completed this thesis at ATR Human Information Processing Laboratories. I am deeply appreciative of the encouragement and generosity given by Dr. Mitsuo Kawato, Head of Department 3. His enlightening and stimulating discussions always intrigued me.

Special thanks are also due to all of the people who gave me valuable comments and continuous encouragement. They include Pascale Fung, Ido Dagan, Fernando Pereira, Raymond Mooney, Hang Li, Francis Bond, Takafumi Inoue, Yutaka Sasaki, Takafumi Mukouchi, Yoshinori Sagisaka, Genichiro Kikui, Masaaki Nagata, Tsuneko Nakazawa.

I greatly acknowledge the many people who kindly helped me over the course of this work. Although I cannot list all of their names, I would like to express my thanks to all of them.

Finally, I also would like to thank my parents for their continuous encouragement and support and for their allowing me to pursue the work I love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past 50 years, researchers have made concerted efforts to construct practical natural language systems such as automatic machine translators. Although these systems are now commercially available and used, those with high performance are limited to simple processors (part-of-speech taggers and kana-kanji converters) or systems tuned for specialized domains. Why is it so difficult to construct practical systems for broader use? The most crucial barrier has been identified as the variety of *knowledge* required for language processing.

This thesis explores the machine learning approach to natural language processing in which the *knowledge* is automatically acquired by using a corpus currently available. Although the proposed techniques yield significant results in both supervised and unsupervised settings, I would not simple-mindedly claim that machine learning is superior to and should replace all conventional methods (nearly all of them done by human learning). Rather, my intention is to clarify what is appropriate for the current machine learning and what is appropriate for the conventional methods. My ultimate objective is to promote a new style of natural language engineering by gaining a deep understanding of the pros and cons of machine learning techniques.

Before going into the details of actual learning techniques, this chapter introduces three types of knowledge and then discusses them from the machine learning point of view. Finally, the tasks and techniques discussed in the subsequent chapters are briefly overviewed.

Let us start with the two examples below. The first example is taken from a Japanese stock market bulletin and its English translation. The two sentences have colloquial styles very specific to the stock market domain; although the Japanese sentence may be translated as 'Tokyo Gold Future market ended trading for the day' human translators never produce such a sentence because they adhere to the rigid translation pattern inherent to the domain. To output such a natural translation, machine translation systems must have domain-specific rules.

- 東京貴金属大引け
  Tokyo Gold futures Cls.

- 花子と京都へ行った (I went to Kyoto with Hanako.)
  大阪と京都へ行った (I went to Kyoto and Osaka.)

The second example concerns the knowledge to disambiguate syntactic structures of Japanese sentences. Although the two sentences have only one word difference (花子 (Hanako) and 大阪 (Osaka)), there is great dissimilarity in their syntactic structures. The postpositional phrase 花子と modifies the verb phrase 行った in the first sentence, while the postpositional phrase 大阪と modifies the postpositional phrase 京都へ in the second sentence. To address such a difference, we need rules governing, for example, what kind of nouns (semantic class) the verb 行く tends to take as its objects and what semantic class the nouns 花子 (Hanako) and 京都 (Kyoto) belong to. It is further interesting to consider the following sentence in which 花子と modifies 太郎が but not 行った. Although we introduced a rule that says 行く tends to take 花子と as an object, it is not valid for the new sentence. Because most language rules deal with only a part or a generalized form of a sentence, they cannot be applied deterministically. In other words, each rule has to be augmented with its preference parameters to determine the most plausible structure of input sentences.

- 花子と太郎が京都へ行った (Hanako and Taro went to Kyoto.)

The close investigation of the above examples shows that the knowledge for language processing can be constructed from the following three steps[1].

1. Enumerating linguistic features that have potential effects on target tasks.

2. Selecting features and combining them into a form of rules.

3. Setting preference parameters for the rules.

In the translation example, bilingual word correspondences (the first step) might be the linguistic features used. The domain specific translation rules (the second step) are constructed by using these features. The third (preference) step can be omitted because there is no ambiguity in applying the domain-specific (sentence-level) rule. In the second syntactic example, on the other hand, the preference step is needed because parsing systems are applied to a broad range of texts. Thus this example involves all three steps; the thesaurus as a feature, the rules of modification and the preferences for modifications.

In conventional systems, all three steps have been accomplished by human linguists. They classify language phenomena and extract linguistic features in order to compile rules. Each rule is then constructed with its preference parameter. All these processes are iterated in a trial-and-error fashion.

Let us now discuss the three steps from the machine learning viewpoint. For the first step, machine learning has limited use. It is generally difficult for current computer programs to automatically enumerate feature types and their coding scheme from scratch. Going back to the second example, it is the insight and experience of a human linguist that enumerate noun categories to be significant features of dependency analysis and that determine how each feature is encoded (say, in a noun thesaurus). It is important to remember that the machine learning approach to enumerating features is limited to the co-occurrence statistics of words and phrases; thus machine learning cannot solve everything. The research on probabilistic context free grammars (PCFG) [9] indicates the importance of the first step. From the late 80s to the present, parameter estimation of PCFG

[1]Note that these steps in fact are deeply interleaved. The separation is not clear-cut but made for explanation.

has been popular in the research community. These grammars are described only with non-terminal symbols (features). This is clearly poor information for disambiguating real-world natural language texts. Parsing accuracy with this approach has remained around 60% with any learning algorithm.

If we assume the existence of features to represent the target task, the second step is to construct rules by combining the features. The machine learning approach is good at producing large scale and comparatively simple structures such as a decision trees [3] or a hidden Markov model (HMM) [67], while human linguist is good at producing a small number of complex rules like typed feature structures [8]. The machine learning approach constructs the rules that explain the global data distribution, while a human linguist depends on intuition and may resort to near-sighted rules. The key point in using machine leaning techniques is to select the necessarily and sufficiently simplest representation that contains enough features for the task [46].

For the third step, the machine learning approach has a significant advantage because the optimization of parameters should be performed from a global point of view. The approach determines the parameters, for example, by maximum likelihood estimation (MLE), which selects the parameters that maximize the likelihood of generating given data. If a human linguist sets the parameters from a near-sighted perspective, the total system would suffer from side-effects. Worse still, even if the linguist makes his utmost efforts to maintain coherence, it gradually becomes difficult as the number of rules increases.

In summary, machine learning approach has the following characteristics related to the above three steps.

1. In enumerating features, current machine learning techniques have limited use (co-occurrence statistics).

2. In constructing rules, the machine learning approach is particularly powerful for a large scale and simple structures such as a decision tree and a HMM.

3. The machine learning approach is well suited to evaluating preference parameters because the problem requires global optimization.

Let us now move on to the concrete applications and techniques discussed in the subsequent chapters. These techniques clearly reflect the characteristics of the machine learning approach.

The first part of the thesis focuses on the bilingual corpus. Japanese articles and its English translations are mainly considered. A Japanese sentence is not always translated into a single English sentence. It may be split into two or three sentences and vice versa. Chapter 2 deals with the sentence alignment problem, whose goal is to find sentence correspondences. The most crucial feature for the task is the word correspondences between two languages. The learning method is based on the intuition that the more word correspondences a sentence pair contains, the more reliable the correspondence is. The proposed method employs two kinds of word correspondences: statistically acquired word correspondences and dictionary word correspondences. Each of the correspondences has its advantages and disadvantages as summarized below.

Statistics Advantage Statistics is robust in the sense that it can extract context-dependent usage of words and that it works well even if word segmentation[2] is not correct.

Statistics Disadvantage The amount of word correspondences acquired by statistics is not enough for complete alignment.

Dictionaries Advantage They can contain information about words that appear only once in the corpus.

Dictionaries Disadvantage They cannot capture context-dependent keywords in the corpus and are weak against incorrect word segmentation. Entries in the dictionaries differ from author to author and are not always the same as those in the corpus.

By combining the two correspondences, sentence alignment is attained with high precision and recall. When some errors are left to be modified by a human supervisor, we will also introduce a sentence alignment environment, called BACCS (Bilingual Aligned Corpus Construction System), designed for human supervisors.

[2]In Japanese, there are no explicit delimiters between words. The first task for alignment is, therefore, to divide the text stream into words.

Chapter 3 describes the word-level sorting method to extract bilingual collocations from the sentence aligned corpora. The bilingual collocations are important features for machine translation systems and mutilingual information retrieval systems. According to the difficulty of learning, the proposed method differentiates two types of collocations: fixed and flexible collocations. Fixed collocations comprise a continuous sequence of words, while flexible collocations consist of more than two separate sequences of words. The translation pair of the first example is an instance of fixed collocations acquired by the proposed method.

The word-level sorting method can efficiently detect continuous sequences of words that occur frequently in a corpus because sequences with the same prefix words are placed next to by the word-level sorting procedure. The fixed collocations are then extracted by reducing the redundant sequences such as 'Bank of' against 'Bank of Japan'. Finally, flexible collocations are extracted by combining the fixed collocations. This two-step strategy for fixed and flexible collocations has the following advantages, although conventional methods have focused on either noun phrases or flexible collocations.

- Fixed collocations can be efficiently and precisely extracted by our word-level sorting method.
- Fixed collocations contain a broad range of expressions, as shown by the above example, rather than noun phrases.
- The construction for flexible collocations is efficient because a two-step combination of fixed collocations reduces redundant computation.

It is worth noting that the three steps in Chapters 2 and 3 are very simple because the problem settings are unsupervised.

The second part of the thesis addresses two fundamental processes in the Japanese language: morphological analysis and dependency analysis. Morphological analysis segments an input sentence into words that are annotated with part-of-speech tags. Because there are no explicit delimiters between words in the Japanese language, morphological analysis is the first step for almost all Japanese text processing. The dependency analysis determines the syntactic structure of the input sentences. More specifically, an input sentence is first segmented into a sequence of bunsetsu (minimal phrases). The modifications between bunsetsu are then determined.

Because these two processes have to deal with general genres of texts, infrequent expressions and their preferences are extremely important. To address this issue, we adopt the mistake-driven mixture approach. The following is the intuition behind this approach. Let us consider the process in which linguists incorporate infrequent expressions into hand-crafted rules: they first construct coarse rules that seem to cover a broad range of data. They then try to analyze data by using the rules and extract exceptions that the rules cannot handle. Next, they generalize the exceptions and refine the previous rules. The following two steps abstract the human algorithm for incorporating infrequent expressions.

1. construct temporary rules that seem to successfully generalize given data.

2. try to analyze the data by using the constructed rules and extract the exceptions that cannot be correctly handled, then return to the first step and focus on the exceptions.

In the same general approach, the mistake-driven learning algorithm (Adaboost) [27] attaches a weight to each example and iteratively performs the following two procedures in the training phase:

1. constructing a model based on the current data distribution (weight vector)

2. updating the distribution (weight vector) by focusing on the data not well predicted by the constructed model. More precisely, the algorithm reduces the weight of the examples that are correctly handled.

For the prediction phase, the algorithm outputs a final model by mixing all the constructed models according to their performance. By using the algorithm, a series of models gradually changes from general to specific. In other words, the algorithm incorporates not only frequent expressions but also infrequent ones that are often considered to be exceptional.

In fact, the Adaboost algorithm has a theoretical basis [72] that explains why it can successfully handle difficult instances without over-fitting the data. In statistical learning theory, the difficulty in classifying an instance is determined by *margin* [79]. The margin intuitively represents the distance to the classification boundaries. The larger the margin of an instance is, the easier it is to classify. [72] proves that the Adaboost algorithm continuously increases the margin of instances and that the generalization error of the final aggregated classifier is bounded by the sum of the error for the training instances and a decreasing function of the margins. The Adaboost algorithm thus can deal with the originally small-margin instances that correspond to the infrequent and difficult expressions from the viewpoint of natural language processing. The margin theory of Adaboost will be briefly discussed in Chapter 4.

Besides the learning algorithm, there remains the problem how tasks are represented. We introduce efficient tree structures: hierarchical context trees for morphological analysis and decision trees for dependency analysis. Chapter 4 constructs a morphological analyzing system by boosting hierarchical context trees. Compared to the commonly used n-gram model, the hierarchical context tree has the following two variabilities in modeling previous contexts (preceding symbols). These variabilities in fact are used intuitively in hand-crafted morphological analyzing systems [60]. The combination of these variabilities and the Adaboost algorithm generates a high performance system.

1. The context length for predicting a symbol is variably determined by an information theoretic criteria according to the distribution of the symbol.

2. The elements representing contexts are automatically selected from a hierarchical structure: word, detailed part-of-speech and coarse part-of-speech.

In the same way, Chapter 5 introduces a Japanese dependency parser that boosts decision trees. The decision tree is used to select linguistic features and to compute the probability that one bunsetsu modifies the other. In fact, the hierarchical context tree used in Chapter 4 is a limited form of the decision tree. The dependency structure of an entire sentence is determined by maximizing the product of the probabilities. Because the feature selection and probability estimation are automatically performed by the learning algorithm, human linguists can focus on the enumeration of linguistic features that have potential effect on parsing performance.

The last part of the thesis describes a concrete application of corpus-based techniques. Chapter 6 introduces an adaptive dictionary environment, called AIDA (Adaptive and Integrated Dictionary Agent), in which several dictionaries (Japanese-English, English-Japanese, English-English), several types of corpora (bilingual and monolingual) and automatically extracted collocations are integrated through a single graphical interface in which users can access any resources bilaterally. When a user wants to write or read texts in foreign languages, his or her requirements are in many cases unclear. AIDA provides a flexible search mechanism for the corpus and dictionary; it finds expressions syntactically and semantically similar to user's input. The similarity parameters in this mechanism are also learned from interaction with users. Finally, Chapter 7 concludes the thesis and mentions future directions.

Chapter 2

Bilingual Sentence Alignment Using Statistical and Dictionary Information

Corpus-based approaches based on bilingual texts are promising for various applications (i.e., lexical knowledge extraction [50, 59, 75, 20, 49], machine translation [7, 71, 41] and information retrieval [70]). Most of these works assume voluminous aligned corpora.

Many methods have been proposed to align bilingual corpora. One of the major approaches is based on the statistics of simple features such as sentence length in words [6] or in characters [31] and character sequences themselves [14]. These techniques are widely used because they can be implemented in an efficient and simple way through dynamic programming. However, their main targets are rigid translations (i.e., Hansard corpus) that are nearly literal translations. In addition, the aligned texts have been structurally similar European languages (i.e., English-French, English-German).

The simple-feature based approaches don't work in flexible translations for structurally different languages such as Japanese and English, mainly for the following two reasons. One is the difference in the character systems of the two languages. Japanese has three types of characters (Hiragana, Katakana and Kanji), each of which carries different amounts of information. In contrast, English has only one type of characters. The other is the grammatical difference between the two languages. In particular, the systems of functional (closed) words are quite different from language to language. Thus, it is impossible in general to apply the simple-feature based methods to Japanese-English translations.

One alternative alignment method is the lexicon-based approach, which makes use of the word-correspondence knowledge of the two languages. [45] proposed a statistical relaxation method to iteratively align bilingual texts using the word correspondences acquired during the alignment process. Although the lexicon-based method works well among European languages, the method does not work in aligning structurally different languages. In Japanese-English translations, the method does not capture enough word correspondences to permit alignment; it can align only portions of the two texts. This is mainly because the number of confident statistical word correspondences is not enough for complete alignment. This problem cannot be addressed as long as the method relies only on statistics. Other methods in the lexicon-based approach embed lexical knowledge into stochastic models [12, 82], but these methods were tested using just rigid translations (i.e., Hansard corpus).

To tackle the problem, we describe in this chapter a text alignment system that uses both statistics and bilingual dictionaries at the same time. Bilingual dictionaries are now widely available on-line due to advances in CD-ROM technologies. For example, English-Spanish, English-French, English-German, English-Japanese, Japanese-French, Japanese-Chinese and other dictionaries are now commercially available. It is reasonable to make use of these dictionaries in bilingual text alignment. The pros and cons of statistics and online dictionaries are discussed below. They show that statistics and on-line dictionaries are complementary in making bilingual text alignment.

Statistics Advantage Statistics is robust in the sense that it can extract context-dependent usage of words and that it works well even if word segmentation is not correct.

Statistics Disadvantage The amount of word correspondences acquired by statistics is not sufficient for complete alignment.

Dictionaries Advantage They can contain information about words that appear only once in the corpus.

Dictionaries Disadvantage They cannot capture context-dependent keywords in the corpus and are weak against incorrect word segmentation. Entries in the dictionaries differ from author to author and are not always the same as those in the corpus.

Our system iteratively aligns sentences by using statistical and on-line dictionary word correspondences. The characteristics of the system are as follows.

- The system performs well and is robust for various lengths (especially short) and various genres of texts.
- The system is very economical because it assumes only online-dictionaries of general use and doesn't require the labor-intensive construction of domain-specific dictionaries.
- The system is extendable by registering statistically acquired word correspondences into user dictionaries.

We will discuss hereafter Japanese-English translation, although the proposed method is language independent.

The proposed alignment method is incorporated in a graphical alignment environment called BACCS (B̲iligngual A̲ligned C̲orpus C̲onstruction S̲ystem). BACCS offers a user interface that enables the user to easily confirm and modify alignment results and register word correspondences created by the alignment program into a user dictionary.

The organization of the chapter is as follows. First, Section 2.1 offers an overview of our alignment system. Section 2.2 describes the entire alignment algorithm in detail. Section 2.3 reports experimental results for various kinds of Japanese-English texts including newspaper editorials, scientific papers and critiques on economics. The evaluation is performed from two points of view: precision-recall of alignment and word correspondences acquired during alignment. Section 2.4 describes our BACCS graphical alignment environment, in which the user can see the alignment results and then easily modify the results and the user dictionary. Section 2.5 describes related works and Section 2.6 summarizes the chapter.

2.1. System Overview

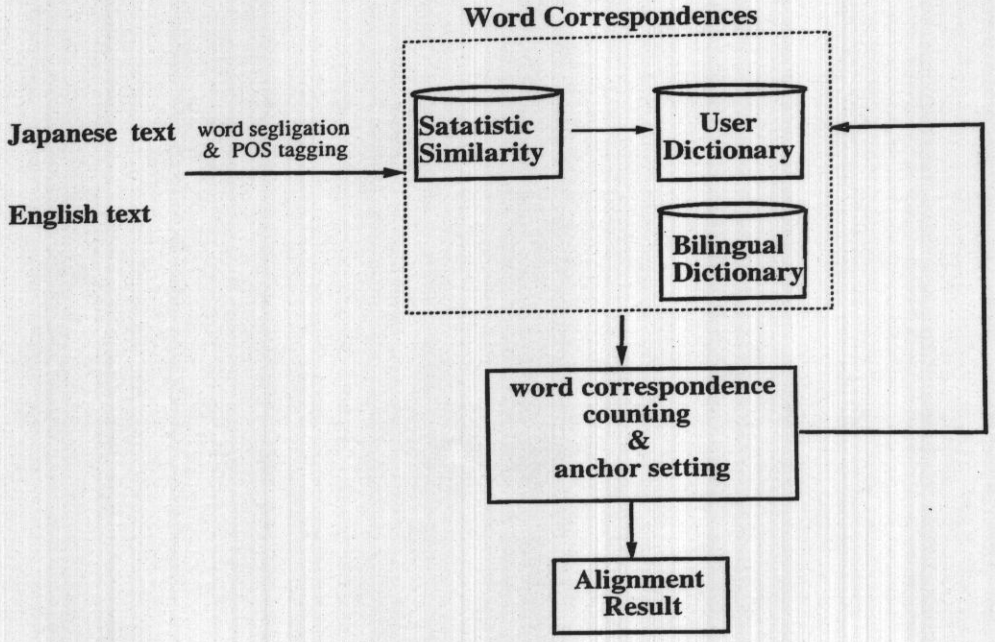


Figure 2.1. Overview of Alignment System BACCS

Figure 2.1 overviews our alignment system called BACCS. The input to the system is a pair of Japanese and English texts, one the translation of the other. First, sentence boundaries are found in both texts using finite state transducers. The texts are then part-of-speech (POS) tagged and separated into original form words[1]. Original forms of English words are determined by 80 rules using the POS information. From the word sequences, we extract only nouns, adjectives, adverbs, verbs and unknown words (only in Japanese) because Japanese and En-

[1]We use the JUMAN morphological analyzing system [52] in this phase for tagging Japanese texts and Brill's transformation-based tagger [4, 5] for tagging English texts. We would like to thank all people concerned for providing us with the tools.

glish closed words are different and impede text alignment. These pre-processing operations can be easily implemented with regular expressions.

The initial state of the algorithm is a set of already known anchors (sentence pairs). These are determined by article boundaries, section boundaries and paragraph boundaries. In the most typical case, initial anchors are simply the first and final sentence pair of each text (Fig. 2.2). Possible sentence correspondences are determined from the anchors. Intuitively, the number of possible correspondences for a sentence is small near anchors, while large between the anchors. In this phase, the most important point is that each set of possible sentence correspondences should include the correct correspondence.

The main task of the system is to find anchors from the possible sentence correspondences by using two kinds of word correspondences: statistical word correspondence and word correspondence from a bilingual dictionary[2]. By using both correspondences, the sentence pair whose correspondences exceed a predefined threshold is considered an anchor. These newly found anchors make word correspondences more precise in the subsequent session. By repeating this anchor setting process with threshold reduction, sentence correspondences are gradually determined from confident pairs to non-confident pairs. The gradualism of the algorithm makes it robust because anchor-setting errors in the final stage of the algorithm have little effect on overall performance. The output of the algorithm is the alignment result (a sequence of anchors) and word correspondences as by-products.

By using BACCS, the user can view and modify sentence correspondences graphically and easily extend the system by registering word correspondences into user dictionaries.

[2]In addition to a general-use bilingual dictionary, users can reuse their own dictionaries created in previous sessions.

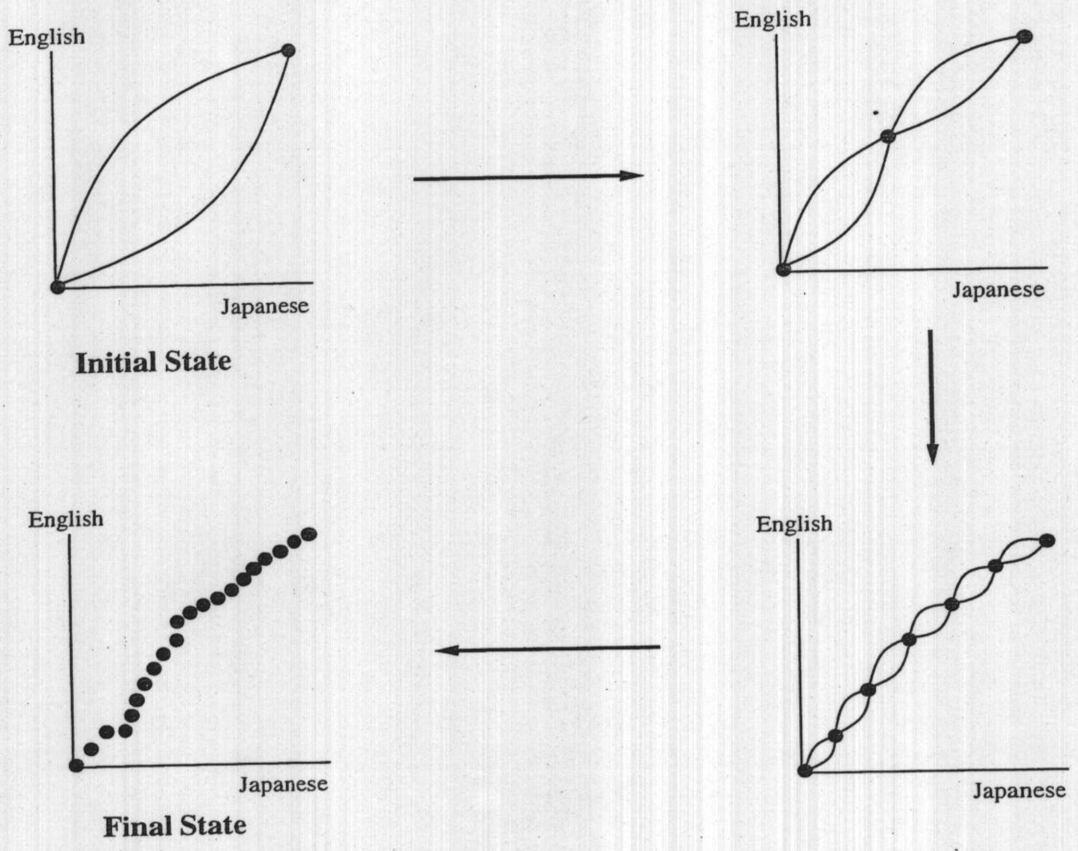


Figure 2.2. Alignment Process

2.2. Algorithms

2.2.1 Statistics Used

In this section, we describe the statistics used to decide word correspondences. From the many similarity metrics applicable to the task, we chose mutual information and *t-score* because the relaxation of parameters can be controlled in a sophisticated manner. Mutual information represents the similarity between occurrence distributions and *t-score* represents the confidence of the similarity. These two parameters permit more effective relaxation than the single parameter used in conventional methods [45].

Our basic data structure is the alignable sentence matrix (ASM) and the anchor matrix (AM). ASM represents possible sentence correspondences and consists of ones and zeros. A one in ASM indicates the intersection of the column, and a row constitutes a possible sentence correspondence. On the other hand, AM is introduced to represent how a sentence pair is supported by word correspondences. The *i*-*j* Element of AM indicates how many times the corresponding words appear in the *i*-*j* sentence pair. As alignment proceeds, the number of ones in ASM decreases, while the elements of AM increase.

Let p_i be a sentence set comprising the *i*th Japanese sentence and its possible English correspondences as depicted in Figure 2.3. For example, p_2 is the set comprising $Jsentence_2$, $Esentence_2$ and $Esentence_3$, which means $Jsentence_2$ has the possibility of aligning with $Esentence_2$ and $Esentence_3$. The p_is can be directly derived from ASM.

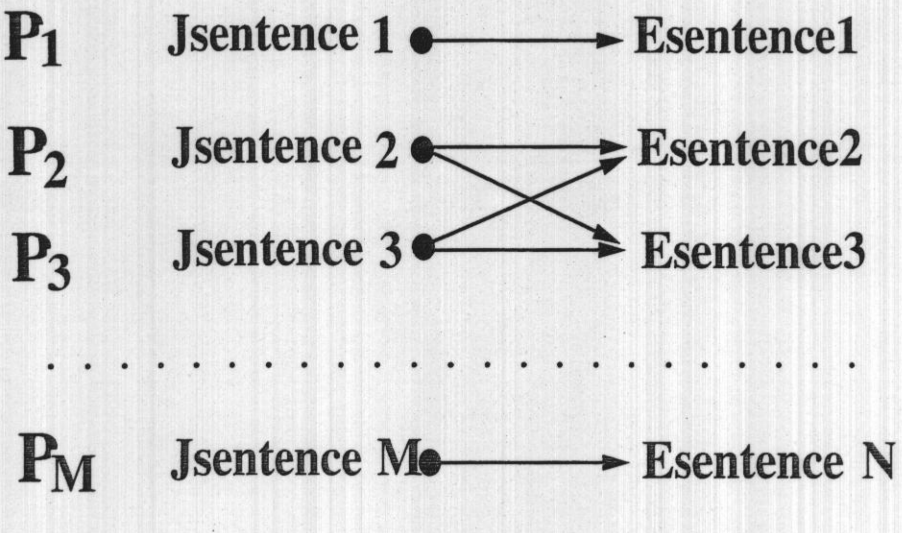


Figure 2.3. Possible Sentence Correspondences

We introduce a contingency matrix [30] to evaluate the similarity of word occurrences. Consider the contingency matrix shown in Table 2.1 between the Japanese word w_{jpn} and the English word w_{eng}. The contingency matrix shows: (a) the number of p_is in which both w_{jpn} and w_{eng} were found, (b) the number of p_is in which just w_{eng} was found, (c) the number of p_is in which just w_{jpn} was found, (d) the number of p_is in which neither word was found. Note that p_is overlap each other and w_{eng} may be counted twice in the contingency matrix. We count each w_{eng} only once, even if it occurs more than twice in p_is.

	w_{jpn}	
w_{eng}	a	b
	c	d

Table 2.1. Contingency Matrix

If w_{jpn} and w_{eng} are good translations of one another, a should be large, and b and c should be small. In contrast, if the two are not good translations of each other, a should be small, and b and c should be large. To make this argument more precise, we introduce mutual information:

$$\log \frac{prob(w_{jpn}, w_{eng})}{prob(w_{jpn})prob(w_{eng})}$$

The probabilities are:

$$prob(w_{jpn}) = \frac{a+c}{a+b+c+d} = \frac{a+c}{M}$$

$$prob(w_{eng}) = \frac{a+b}{a+b+c+d} = \frac{a+b}{M}$$

$$prob(w_{jpn}, w_{eng}) = \frac{a}{a+b+c+d} = \frac{a}{M}$$

Unfortunately, mutual information is not reliable when the number of occurrences is small. Many words occur just once, which weakens the statistics approach. In order to avoid this, we employ *t-score*, defined below, where M is the number of Japanese sentences. Insignificant mutual information values are filtered out by thresholding the *t-score*. For example, *t-score*s above 1.65 are significant at the $p > 0.95$ confidence level.

$$t \approx \frac{prob(w_{jpn}, w_{eng}) - prob(w_{jpn})prob(w_{eng})}{\sqrt{\frac{1}{M}prob(w_{jpn}, w_{eng})}}$$

2.2.2 Basic Alignment Algorithm

Our basic algorithm is an iterative adjustment of the Anchor Matrix (AM) using the Alignable Sentence Matrix (ASM). Given an ASM, mutual information and *t-score* are computed for all word pairs in possible sentence correspondences. A word combination exceeding a predefined threshold is judged to be a word correspondence. In order to find new anchors, we combine these statistical word correspondences with the word correspondences in a bilingual dictionary. Each element of AM, which represents a sentence pair, is updated by adding the number of word correspondences in the sentence pair. A sentence pair containing more than a predefined number of corresponding words is determined to be a new anchor. The detailed algorithm is described below.

2.2.3 Constructing Initial ASM

This step constructs the initial ASM. If the texts contain M and N sentences respectively, the ASM is an $M \times N$ matrix. First, we decide a set of anchors using article boundaries, section boundaries and so on. In the most typical case, initial anchors are the first and last sentences of both texts as depicted in Figure 2.2. Next, possible sentence correspondences are generated. Intuitively, true correspondences are close to the diagonal linking of the two anchors. We construct the initial ASM by using a function that pairs sentences near the middle of the two anchors with as many as $O(\sqrt[2]{L})$ (L is the number of sentences existing between two anchors) sentences in the other text because the maximum deviation can be stochastically modeled as $O(\sqrt[2]{L})$ [45]. The initial ASM has little effect on the alignment performance so long as it contains all correct sentence correspondences.

2.2.4 Constructing AM

This step constructs an AM when given an ASM and a bilingual dictionary. Let t_{high}, t_{low}, I_{high} and I_{low} be two thresholds for *t-score* and two thresholds for mutual information, respectively. Let ANC be the minimal number of corresponding words for a sentence pair to be judged an anchor.

First, mutual information and *t-score* are computed for all word pairs appearing in a possible sentence correspondence in ASM. We use hereafter the word correspondences whose mutual information exceeds I_{low} and whose *t-score* exceeds t_{low}. For all possible sentence correspondences $Jsentence_i$ and $Esentence_j$ (any pair in ASM), the following operations are applied in order.

1. If the following three conditions hold, add 3 to the *i-j* element of AM. (1) $Jsentence_i$ and $Esentence_j$ contain a bilingual dictionary word correspondence (w_{jpn} and w_{eng}). (2) w_{eng} does not occur in any other English sentence that is a possible translation of $Jsentence_i$. (3) $Jsentence_i$ and $Esentence_j$ do not cross any sentence pair that has more than ANC word correspondences.

2. If the following three conditions hold, add 3 to the *i-j* element of AM. (1) $Jsentence_i$ and $Esentence_j$ contain a stochastic word correspondence (w_{jpn} and w_{eng}) that has mutual information I_{high} and whose *t-score* exceeds t_{high}. (2) w_{eng} does not occur in any other English sentence that is a possible translation of $Jsentence_i$. (3) $Jsentence_i$ and $Esentence_j$ do not cross any sentence pair that has more than ANC word correspondences.

3. If the following three conditions hold, add 1 to the *i-j* element of AM. (1) $Jsentence_i$ and $Esentence_j$ contain a stochastic word correspondence (w_{jpn} and w_{eng}) that has mutual information I_{low} and whose *t-score* exceeds t_{low}. (2) w_{eng} does not occur in any other English sentence that is a possible translation of $Jsentence_i$. (3) $Jsentence_i$ and $Esentence_j$ do not cross any sentence pair that has more than ANC word correspondences.

The first operation deals with word correspondences in the bilingual dictionary. The second operation deals with stochastic word correspondences that are highly confident and in many cases involve domain specific keywords. These word correspondences are given the value of 3. The third operation is introduced because the number of highly confident corresponding words are too small to align all sentences. Although word correspondences acquired by this step are sometimes false translations of each other, these words appear in corresponding

No.	Text Name	Jpn	Eng	1-1	1-2	2-1	3-1
1	*Root out guns at all costs*	26	28	24	2	0	0
2	*Economy facing last hurdle*	36	41	25	7	2	0
3	*Pacific Asia in the Post-Cold-War World*	134	124	114	0	10	0
4	*Visualizing the Mind*	225	214	186	6	15	1

Table 2.2. Test Texts

sentences with high probability. Thus, they can play a crucial role mainly in the final iteration phase. They are given one point.

2.2.5 Adjusting ASM

This step adjusts ASM using the AM constructed by the above operations. The sentence pairs that have at least ANC word correspondences are determined to be new anchors. By using the new set of anchors, a new ASM is constructed using the same method as that used for initial ASM construction.

Our algorithm implements a kind of relaxation by gradually reducing t_{low}, I_{low} and ANC, which enables us to find confident sentence correspondences first. As a result, our method is more robust than dynamic programming techniques against the shortage of word-correspondence knowledge.

2.3. Experimental Results

In this section, we report the results of experiments on aligning sentences in bilingual texts and on statistically acquired word correspondences. The texts for the experiment varied in length and genres as summarized in Table 2.2. Texts 1 and 2 are editorials taken from 'Yomiuri Shinbun' and its English version 'Daily Yomiuri'. This data was distributed electronically via a WWW server[3]. The first two texts clarify the system's performance on shorter texts. Text 3 is an essay on economics taken from a quarterly publication of The International House of Japan. Text 4 is a scientific survey on brain science taken from 'Scientific

[3]We would like to thank Yomiuri Shinbun Co. for permitting us to use the data.

Japanese	English	Mutual Information
記録	recording	3.58
リアルタイム	real	3.51
ニューロン	neuron	3.51
フィルム	film	3.51
グルコース	glucose	3.51
増加	increase	3.51
磁	MEG	3.51
解像度	resolution	3.43
電気	electrical	3.43
グループ	group	3.39
電気	recording	3.39
記録	electrical	3.39
言う	generate	3.33
提供	provide	3.33
図	MEG	3.33
NMR	NMR	3.17
ファンクショナル	functional	3.17
機器	equipment	3.17
臓器	organ	3.15
注射	compound	3.10
水	water	3.10
標識	radioactive	3.10
ＰＥＴ	PET	3.10
解像度	spatial	3.10
そのようだ	such	3.10
代謝	metabolism	3.06
言う	verb	3.04
科学者	scientist	2.95
地図	mapping	2.92
大学	university	2.92
思考	thought	2.90
化合物	compound	2.82
標識	label	2.82
視覚	visual	2.77
信号	signal	2.77
リアルタイム	time	2.69
オートラジオ	autoradiography	2.67

Table 2.3. Statistically Acquired Keywords

American' and its Japanese version 'Nikkei Science'. **Jpn** and **Eng** in Table2.2 represent the number of sentences in the Japanese and English texts respectively. The remaining table entries show categories of matches by manual alignment and indicate the difficulty of the task.

We briefly report here the computation time of our method. Let us consider Text 4 as an example. After 15 seconds for full preprocessing, the first iteration took 25 seconds with $t_{low} = 1.55$ and $I_{low} = 1.8$. The rest of the algorithm took 20 seconds in all. This experiment was performed on a SPARC Station 20. From the result, we may safely say that our method can be applied to voluminous corpora.

2.3.1 Sentence Alignment

	Proposed		**Statistics**		**Dictionary**	
Text	PRECISION	RECALL	PRECISION	RECALL	PRECISION	RECALL
1	96.4%	96.3%	65.0%	48.5%	89.3%	88.9%
2	95.3%	93.1%	61.3%	49.6%	87.2%	75.1%
3	96.5%	97.1%	87.3%	85.1%	86.3%	88.2%
4	91.6%	93.8%	82.2%	79.3%	74.3%	63.8%

Table 2.4. Results of Sentence Alignment

Table 2.4 shows the performance of sentence alignment for the texts in Table 2.2. **Proposed, Statistics** and **Dictionary** represent the methods using both statistics and dictionary, only statistics and only dictionary, respectively. Both **Proposed** and **Dictionary** use a CD-ROM version of a Japanese-English dictionary containing 40,000 entries. **Statistics** repeats the iteration by using statistically corresponding words only. This is identical to Kay's method [45] except for the statistics used. **Dictionary** performs the first iteration of the algorithm by using corresponding words of the bilingual dictionary. This delineates the coverage of the dictionary. The parameter setting used for each method was optimal as determined by empirical tests.

In Table 2.4, PRECISION indicates how many of the aligned pairs are correct and RECALL indicates how many of the manual alignments we included in system output. Unlike conventional sentence-chunk based evaluations, our result is measured on a sentence-sentence basis. Let us consider a 3-1 matching (sentence chink). Conventional evaluations can make at most one error from the chunk because they count errors based on the number of chunks. On the other hand, three errors may arise by our sentence-sentence based evaluation. Note that our evaluation is more strict than the conventional one, especially for difficult texts because they contain more complex matches.

For Text 1 and Text 2, both the proposed method and the dictionary method perform much better than the statistical method. This is obviously because statistics cannot capture word-correspondences in the case of short texts.

Text 3 is easy to align in terms of both complexity of alignment and vocabulary used. All methods performed well on this text.

For Text 4, **Proposed** and **Statistics** perform much better than **Dictionary**. The reason for this is that Text 4 concerns brain science and the bilingual dictionaries in general use did not contain domain-specific keywords. On the other hand, the proposed and statistical methods well captured the keywords, as described in the next section. Note here that **Proposed** also performs better than **Statistics** in the case of longer texts. There is clearly a limitation in the amount of word correspondences that can be captured by statistics. The precision of the proposed method for Text 4 is not as high as for the other texts due to the following reasons. First, Text 4 contains more technical terms than others because it concerns brain science. Bilingual dictionaries in general use cannot capture these technical terms. Second, Text 4 contains more multiple alignments (such as 1-2 and 2-1). Multiple alignments degrade alignment performance.

In summary, **Proposed** has a better performance than either that of **Statistics** or **Dictionary** for all texts, regardless of text length and domain.

2.3.2 Word Correspondence

In this section, we will demonstrate how well the proposed method captured domain-specific word correspondences by using Text 4 as an example. Table 2.3 shows the word correspondences that have high mutual information. These are typical keywords involved in reporting the non-invasive approach to human brain analysis. For example, NMR, MEG, PET, and functional MRI are devices for measuring brain activity from outside the head. These technical terms are the subjects of the text and are essential for alignment. However, none of them have their own entry in the bilingual dictionary, which would severely obstruct the dictionary method.

It is interesting to note that the correct Japanese translation of 'MEG' is '脳磁図'. The Japanese morphological analyzer we used does not contain an entry for '脳磁図' and so splits it into a sequence of three characters '脳','磁' and '図'. Our system skillfully combined '磁' and '図' with 'MEG' as a result of statistical acquisition. These word correspondences greatly improved the performance for Text 4. Thus, the statistical method well captures the domain-specific keywords that are not included in general-use bilingual dictionaries. The dictionary method would yield false alignments if statistically acquired word correspondences were not used.

2.4. Graphical Alignment Environment BACCS

This section describes a graphical alignment environment called BACCS[4] (Bilingual Aligned Corpus Construction System) that incorporates our alignment program. The experimental results in the previous section showed that our alignment method attained high accuracy; however, this method can not always achieve 100% correct alignment. Thus it is necessary for humans to modify and confirm the alignment results given by our alignment program to build an accurately aligned corpus. Such a corpus is a useful resource for extracting various kinds of expressions and their translations in the next chapter.

BACCS offers a user interface that enables the user to easily confirm and modify alignment results and register word correspondences created by the alignment program into a user dictionary. Figure 2.4 shows a screenshot image of BACCS. The rear window shows alignment results, while the front window shows the word correspondence candidates.

The process of making a correctly aligned corpus with BACCS is as follows.

[4]BACCS was implemented in Tcl/Tk on X Window.

1. Select the target bilingual texts.
2. Run the alignment program and display the alignment results.
3. Confirm and modify these results by mouse actions.
4. Choose the proper word correspondences from among the candidates created by the alignment program and register these into the user dictionary[5].

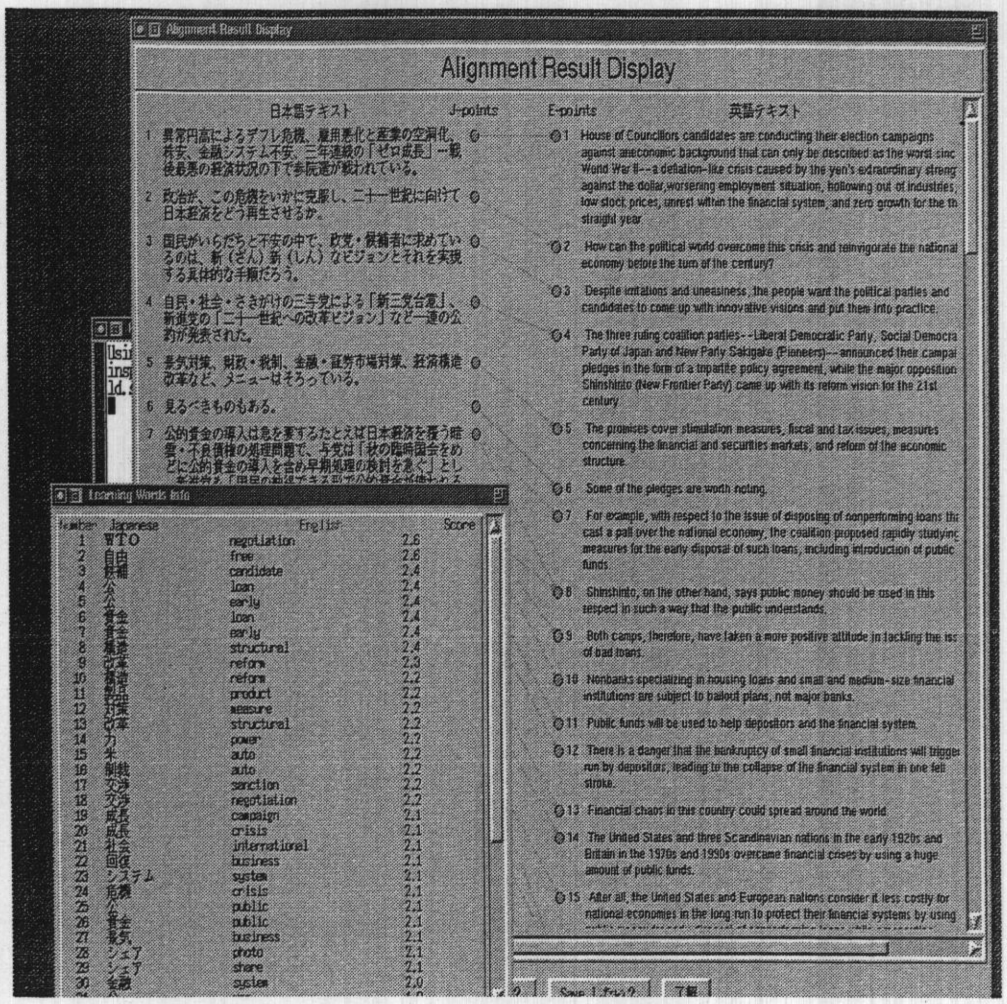
BACCS not only finds sentence alignment but also extracts bilingual collocations from aligned sentences by using the word-level sorting method[33] described in the next chapter.

The annotated bilingual corpus is used in Chapter 6 for our translation aid system AIDA, which integrates bilingual corpus, mono-lingual corpus and hand-compiled dictionaries.

2.5. Related Work

Sentence alignment between Japanese and English was first explored by Sato and Murao [61]. They found (character or word) length-based approaches were not appropriate due to the structural differences between the two languages . They devised a dynamic programming method based on the number of corresponding words in a hand-crafted bilingual dictionary. Their approach is similar to a bilingual phrase extracting method [21]. Although some results were promising, the method's performance strongly depended on the domain of the texts and the dictionary entries. [78] introduced a statistical post-processing step to tackle this problem. He first applied Sato's method and extracted statistical word correspondences from the result of the first path. Sato's method was then reiterated using both the acquired word correspondences and the hand-crafted dictionary. His method involves the same problem as Sato's method. That is, unless the hand-crafted dictionary contains domain-specific keywords, the first path yields false alignment, which in turn leads to false statistical correspondences. Because it is basically impossible to cover keywords from all possible domains, it is inevitable

[5]In the experiments reported in the previous section, no user dictionaries were used.



Alignment Result Display

日本語テキスト　J-points

1 異常円高によるデフレ危機、雇用悪化と産業の空洞化、株安、金融システム不安、三年連続の「ゼロ成長」―戦後最悪の経済状況の下で参院選が戦われている。

2 政治が、この危機をいかに克服し、二十一世紀に向けて日本経済をどう再生させるか。

3 国民がいらだちと不安の中で、政党・候補者に求めているのは、新（ざん）新（しん）なビジョンとそれを実現する具体的な手順だろう。

4 自民・社会・さきがけの三与党による「新三党合意」、新進党の「二十一世紀への改革ビジョン」など一連の公約が発表された。

5 景気対策、財政・税制、金融・証券市場対策、経済構造改革など、メニューはそろっている。

6 見るべきものもある。

7 公的資金の導入は急を要するたとえば日本経済を覆う暗雲・不良債権の処理問題で、与党は「秋の臨時国会をめどに公的資金の導入を含め早期処理の検討を急ぐ」とし

E-points　英語テキスト

1 House of Councillors candidates are conducting their election campaigns against aneconomic background that can only be described as the worst sinc World War II--a deflation-like crisis caused by the yen's extraordinary streng against the dollar,worsening employment situation, hollowing out of industries, low stock prices, unrest within the financial system, and zero growth for the th straight year.

2 How can the political world overcome this crisis and reinvigorate the national economy before the turn of the century?

3 Despite irritations and uneasiness, the people want the political parties and candidates to come up with innovative visions and put them into practice.

4 The three ruling coalition parties--Liberal Democratic Party, Social Democra Party of Japan and New Party Sakigake (Pioneers)--announced their campai pledges in the form of a tripartite policy agreement, while the major opposition Shinshinto (New Frontier Party) came up with its reform vision for the 21st century.

5 The promises cover stimulation measures, fiscal and tax issues, measures concerning the financial and securities markets, and reform of the economic structure.

6 Some of the pledges are worth noting.

7 For example, with respect to the issue of disposing of nonperforming loans thi cast a pall over the national economy, the coalition proposed rapidly studyinc measures for the early disposal of such loans, including introduction of public funds.

8 Shinshinto, on the other hand, says public money should be used in this respect in such a way that the public understands.

9 Both camps, therefore, have taken a more positive attitude in tackling the iss of bad loans.

10 Nonbanks specializing in housing loans and small and medium-size financial institutions are subject to bailout plans, not major banks.

11 Public funds will be used to help depositors and the financial system.

12 There is a danger that the bankruptcy of small financial institutions will trigger run by depositors, leading to the collapse of the financial system in one fell stroke.

13 Financial chaos in this country could spread around the world.

14 The United States and three Scandinavian nations in the early 1920s and Britain in the 1970s and 1990s overcame financial crises by using a huge amount of public funds.

15 After all, the United States and European nations consider it less costly for national economies in the long run to protect their financial systems by using

Number	Japanese	English	Score
1	WTO	negotiation	2.6
2	自由	free	2.6
3	候補	candidate	2.4
4	公	loan	2.4
5	公	early	2.4
6	資金	loan	2.4
7	資金	early	2.4
8	構造	structural	2.4
9	改革	reform	2.3
10	構造	reform	2.2
11	製品	product	2.2
12	対策	measure	2.2
13	改革	structural	2.2
14	力	power	2.2
15	米	auto	2.2
16	制裁	auto	2.2
17	交渉	sanction	2.2
18	交渉	negotiation	2.2
19	成長	campaign	2.1
20	成長	crisis	2.1
21	社会	international	2.1
22	回復	business	2.1
23	システム	system	2.1
24	危機	crisis	2.1
25	公	public	2.1
26	資金	public	2.1
27	景気	business	2.1
28	シェア	photo	2.1
29	シェア	share	2.1
30	金融	system	2.0

Figure 2.4. BACCS

that statistics and hand-crafted bilingual dictionaries be used simultaneously. The proposed method involves iterative alignment that simultaneously uses both statistics and a bilingual dictionary.

[30, 29] proposed methods to find Chinese-English word correspondences without aligning parallel texts. Their motivation was based on the fact that structurally different languages such as Chinese-English and Japanese-English are generally difficult to align. Their methods abandoned aligning sentences and only acquired word correspondences. Although their approaches are robust and do not require any information source, aligned sentences are indispensable for higher ap-

plications such as translation template acquisition [59, 41], example-based translation [71] and translation aide systems such as AIDA. Our method performs accurate alignment for such use by combining the detailed word correspondences of statistically acquired word correspondences with those from a bilingual dictionary in general use.

2.6. Summary

We have described a bilingual text alignment system for structurally different languages. The proposed method uses two kinds of word correspondences at the same time: word correspondences acquired by statistics and those of a bilingual dictionary. By combining these two types of word correspondences, the method covers both domain-specific keywords not included in the dictionary and infrequent words not detected by statistics. As a result, the proposed method outperforms conventional methods for texts of different lengths and different domains.

Chapter 3

Learning Bilingual Collocations by Word-level Sorting

In the field of translation, there is a growing interest in corpus-based approaches [71, 20, 59, 49, 75]. The main motivation behind this is to well handle domain specific expressions. Each application domain has various kinds of collocations ranging from word-level to sentence-level. The correct use of these collocations greatly influences the quality of texts. Because such detailed collocations are difficult to hand-compile, an automatic extraction of bilingual collocations is needed.

[75] proposed a general method to extract a broader range of collocations. This method first extracts English collocations using the Xtract system [74] and then looks for French counterparts. Their search strategy is an iterative combination of two elements. This is based on the intuitive idea that "if a set of words constitutes a collocation, its subset will also be correlated". Although this idea is correct, the iterative combination strategy generates a number of useless expressions. In fact, Xtract employs a robust English parser to filter out the wrong collocations that form more than half of the candidates. In other languages, parser-based pruning cannot be used. Another drawback of their approach is that only the longest n-gram is adopted. When 'Japan-US auto trade talks' is adopted as a collocation, 'Japan-US' cannot be recognized as a collocation though it is frequently used independently.

In this chapter, we propose an alternative method based on word-level sorting.

1. 東京外為１７時・円、小反発－－２６銭高８４円２１－２４銭。
   Tokyo Forex 5 PM: Dollar at 84.21-84.24 yen

2. 前週末比２６銭円高ドル安の１ドル＝８４円２１－２４銭で大方の取引を終えた。
   The dollar stood 0.26 yen lower at 84.21-84.24 at 5 p.m.

3. ドル安マルク高を受けて小口の円買いが先行したが、日米自動車問題での橋本通産相とカンター米通商代表部代表との会談を日本時間２７日未明に控えて模様眺めムードが強まり、円は８４円前半で小動きに推移した。
   Forex market trading was extremely quiet ahead of further auto talks between Japan and the U.S., slated for early dawn Tuesday.

4. ドルは対マルクで反落し１ドル＝１．３８６３－６６マルクでほぼ取引を終えた。
   The U.S. currency was quoted at 1.361-1.3863 German marks at 5:15 p.m.

Table 3.1. Sample of Target Texts

Our method comprises two steps: (1) extracting useful word chunks (n-grams) in each language by using word-level sorting and (2) constructing bilingual collocations by combining the word-chunks acquired at stage (1). Given sentence-aligned texts in two languages [36], the first step detects useful word chunks by sorting and counting all uninterrupted word sequences in sentences. In this phase, we developed a new technique for extracting only useful chunks. The second step of the method evaluates the statistical similarity of the word chunks appearing in the corresponding sentences. Most of the fixed (uninterrupted) collocations are directly extracted from the word chunks. More flexible (interrupted) collocations are acquired level-by-level by iteratively combining the chunks. The proposed method, which uses effective word-level sorting, not only extracts fixed collocations with high precision but also avoids the combinatorial explosion involved in searching flexible collocations. In addition, our method is robust and suitable for real-world applications; even if the part-of-speech taggers make errors in word segmentation, most of the errors can be recovered in the word chunk extraction stage because this stage can recombine the false segments.

The organization of the chapter is as follows. In Section 3.1, we demonstrate sample Japanese-English collocations and classify them into two types (**fixed** and **flexible**) according to the difficulty of statistical learning. After discussing how useful word chunks are extracted by the word-level sorting in Section 3.2, we will move on to the extraction of bilingual collocations. In Section 3.3, we discuss our experimental result from using two different kinds of data. One is scientific articles that consist of comparatively rigid translations. The other is more challenging: a Japanese stock market bulletin and its English abstract. The different characteristics of the data will shed light on the ability of statistical learning methods. Section 3.4 discusses related works and Section 3.5 gives the summary.

3.1. Two Types of Japanese-English Collocations

In this section, we briefly classify the types of Japanese-English collocations by using the material in Table 3.1 as an example. These texts were derived from a stock market bulletin written in Japanese and its abstract written in English.

In Table 3.1, (東京外為 / *Tokyo Forex*), (日米自動車問題 / *auto talks between Japan and the U.S.*) and (控えて / *ahead of*) are Japanese-English collocations whose elements constitute uninterrupted word sequences. Hereafter, we call this type of collocation **fixed collocation**. Although the fixed collocation seems trivial, more than half of all useful collocations belong to this class. Thus, it is important to extract fixed collocations with high precision. In contrast, (ドルは〜で取り引きを終えた / *The U.S. currency was quoted at* ∼) and (ドルは〜で取り引きを終えた / *The dollar stood* ∼)[1] are constructed from interrupted word sequences. We will call this type of collocation **flexible collocation**. From the viewpoint of machine learning, flexible collocations are much more difficult to learn because they involve the combination of elements. The point to consider

[1] ∼ represents any sequence of words.

when extracting flexible collocations is how the number of combinations (candidates) can be reduced.

Our learning method is twofold according to the collocation types. First, useful uninterrupted word chunks are extracted by the word-level sorting method. We then evaluate stochastic similarity of the chunks to find out fixed collocations. Finally, we iteratively and hierarchically combine the chunks to extract flexible collocations.

3.2. Extracting Useful Chunks by Word-level Sorting

3.2.1 Previous Research

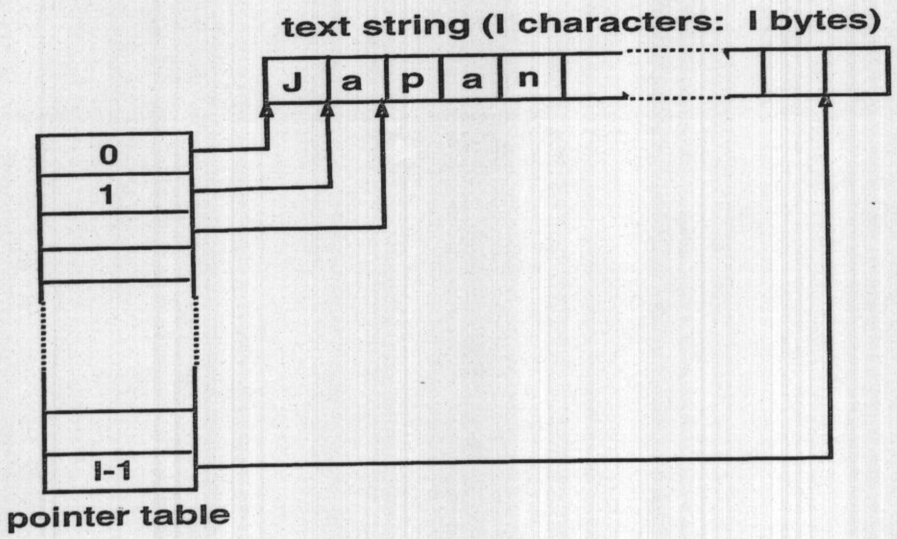


Figure 3.1. Character-level Sorting Approach

With the availability of large corpora and memory devices, there is once again growing interest in extracting n-grams with large values of n. [62] introduced an efficient method for calculating an arbitrary number of n-grams from large corpora. When the length of a text is l bytes, it occupies l consecutive bytes in memory (Fig. 3.1). First, another table of size l is prepared, each field of which represents a pointer to a substring. A substring pointed to by the $(i-1)$th entry of the table constitutes a string existing from the ith character to the end of the text string. Next, to extract common substrings, the pointer table is sorted in alphabetic order. Two adjacent words in the pointer table are compared and the lengths of coincident prefix parts are counted [32].

For example, when *'auto talks between Japan and the U.S.'* and *'auto talks between Japan and China'* are two adjacent words, the number of coincidences is 29, i.e., *'auto talks between Japan and '*. The n-gram frequency table is constructed by counting the number of pointers that represent the same prefix parts. Although the method is efficient for large corpora, it involves a large volume of fractional and unnecessary expressions. The reason for this is that the method does not consider the inter-relationships between the extracted strings. That is, the method generates redundant substrings that are subsumed by longer strings.

To settle this problem, [38] proposed a method to extract only useful strings. Basically, the method is based on the *longest-match* principle. When the method extracts a longest n-gram as a chunk, strings subsumed by the chunk are derived only if the shorter string often appears independently of the longest chunk. If *'auto talks between Japan and the U.S.'* is extracted as a chunk, *'Japan and the U.S.'* is also extracted because *'Japan and the U.S.'* is so often used independently as in *'Japan and the U.S. agreed ···'*. However, *'Japan and the'* is not extracted because it always appears in the context of *'Japan and the U.S.'*. The method strongly suppresses fractional and unnecessary expressions. More than 75% of the strings extracted by Nagao's method are removed by the new method.

3.2.2 Word-level Sorting Method

The research described in the previous section deals with character-based n-grams, which generate excessive numbers of expressions and require large memory for the pointer table. Thus, from a practical point of view, word-based n-grams are preferable in order to further suppress fractional expressions and pointer table use. We extend Ikehara's method [38] to handle word-based n-grams. First, both Japanese and English texts are part-of-speech (POS) tagged[2] and stored in

[2]We use the JUMAN morphological analyzing system [52] for tagging Japanese texts and Brill's transformation-based tagger [5] for tagging English texts. We would like to thank all

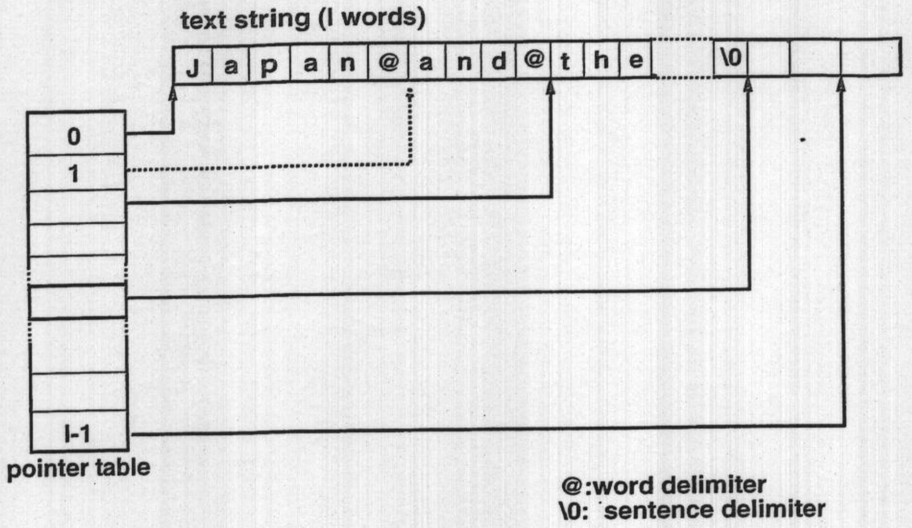


Figure 3.2. Word-level Sorting Approach

memory (Fig. 3.2).

POS tagging is required for two main reasons: (1) There are no explicit word delimiters in Japanese and (2) By using POS information, useless expressions can be removed[3]

In Figure 3.2, '@' and '\0' represent the explicit word delimiter and the explicit sentence delimiter, respectively. The dotted arrow indicates a useless pointer because phrases starting from a conjunct never constitute collocations. This pointer is not registered in the pointer table. Compared to previous research, this data structure has the following advantages.

1. Only heads of each candidate word are recorded in the pointer table. As shown in Figure 3.2, this significantly reduces memory use because the pointer table also contains other string characteristics as Figure 3.3.

2. As shown in Figure 3.2, only expressions within a sentence are considered by introducing the explicit sentence delimiter '\0'.

people concerned for providing us with the tools.

[3]To filter out useless expressions, 50 POS rules (for each language) in regular expressions are applied in both the pointer generation process (Figure 3.2) and the useful chunk detection process (Figure 3.3).

3. Only word-level coincidences are extracted by introducing the explicit word delimiter '@'. This removes strings arising from a partial match of different words. For example, the coincident string between *'Japan and China'* and *'Japan and Costa Rica'* is *'Japan and'* in our method, while it is *'Japan and C'* in previous methods.

sent no.	adopt	coinci dence	string
			· · · · · · · · ·
24		10	Japan@and@China@
105		10	Japan@and@Costa Rica
1064		16	Japan@and@the@US
3		16	Japan@and@the@US
2104		16	Japan@and@the@US
1702		16	Japan@and@the@US
1104		16	Japan@and@the@US
104		16	Japan@and@the@US
			· · · · · · · · ·

Figure 3.3. Sorted Pointer Table

Next, the pointer table is sorted in alphabetic order as shown in Figure 3.3. In this table, **sentno.** and **coincidence** represent which sentence the string appeared in and how many characters are shared by the two adjacent strings, respectively. That is, **coincidence** totals the candidates for useful expressions. Note here that the coincidence between *Japan@and@China·*· · and *Japan@and@Costa Rica·*· · is 10 as mentioned above. After counting **coincidence**, the POS rules are applied again to filter out the syntactically useless expressions. The above *'Japan and'* is then removed from the candidates.

Next, in order to remove useless subsumed strings, the pointer table is sorted according to **sentno.**. In this stage, **adopt** is filled with '1' or '0' , each of which represents if or not if a string is subsumed by longer word chunks, respectively. Sorting by **sentno.** makes it much easier to check the subsumption of word chunks. If both *'Japan and the U.S.'* and *'Japan and the'* arise from a sentence[4], the latter is removed because the former subsumes the latter.

Finally, to determine which word-chunks to extract, the pointer table is sorted once again in alphabetic order. In this stage, we count how many times a string whose **adopt** is 1 appears in the corpus. By thresholding the frequency, only useful word chunks are extracted.

3.2.3 Extracting Bilingual Collocations

In this section, we will explain how fixed and flexible collocations are constructed from word chunks extracted in the previous stage. First, we introduce the metric to evaluate the similarity of word-chunk occurrences in two languages. We use the contingency matrix shown in Table 3.2 for Japanese word chunk c_{jpn} and English word chunk c_{eng}. The contingency matrix shows: (a) the number of Japanese-English corresponding sentence pairs in which both c_{jpn} and c_{eng} were found, (b) the number of Japanese-English corresponding sentence pairs in which just c_{eng} was found, (c) the number of Japanese-English corresponding sentence pairs in which just c_{jpn} was found, (d) the number of Japanese-English corresponding sentence pairs in which neither chunk was found.

	c_{jpn}	
c_{eng}	a	b
	c	d

Table 3.2. Contingency Matrix

If c_{jpn} and c_{eng} are good translations of one another, a should be large, and b and c should be small. In contrast, if the two are not good translations of each other, a should be small, and b and c should be large. To make this argument more precise, our system implements two similarity criteria: (1) mutual information and (2) Dice's coefficient [45].

[4]In fact, *'Japan and the'* should be eliminated by POS rules

$$(1) \log \frac{prob(c_{jpn}, c_{eng})}{prob(c_{jpn})prob(c_{eng})} = \log \frac{a(a+b+c+d)}{(a+b)(a+c)}$$

$$(2) \frac{2a}{(2a+b+c)}$$

Because these two criteria do not make great distinctions empirically, we will hereafter consider the combination of mutual information and co-occurrence (a in Table 3.2) as our similarity measure.

```
Construct two (Japanese and English) word chunk index
while(agenda file ($F$,$M$) is not null)
  for each sentence pair $i$
    if a word chunk pair in $i$ exceeds $F$ and $M$
      keep it as a candidate
  select collocation pairs whose elements constitute the maximally similar
  translation of each other
  remove the collocations from word chunk index
```

Table 3.3. Algorithm for Fixed Collocations

Table 3.3 summarizes how fixed collocations are extracted by using the word chunks induced in the previous section. The algorithm is an iterative method [45, 36, 47] that gradually extracts from highly confident to less confident collocations. The iteration schedule is given to the algorithm by an agenda file that constitutes a sequence of co-occurrence (F) and mutual information (M) thresholds.

For all bilingual word chunks in each sentence pair, co-occurrence frequency and mutual information are computed. If both values exceed thresholds (F) and (M), the bilingual word chunks are kept as collocation candidates. The candidates whose elements constitute the maximally similar translation of each other are extracted as bilingual collocations. These collocations are removed from word chunk index and the algorithm continues until the agenda file comes to the end.

Next, we summarize how flexible collocations are extracted. The following is a series of procedures to extract flexible collocations.

1. For any pair of chunks in a Japanese sentence, compute mutual information. Combine the two chunks of highest mutual information. Iteratively repeat this procedure and construct a tree level-by-level (maximum connection number is 3).

2. For any pair of chunks in an English sentence, repeat the operations done in the Japanese sentence.

3. Perform node matching between trees of both languages by using mutual information of Japanese and English word chunks.

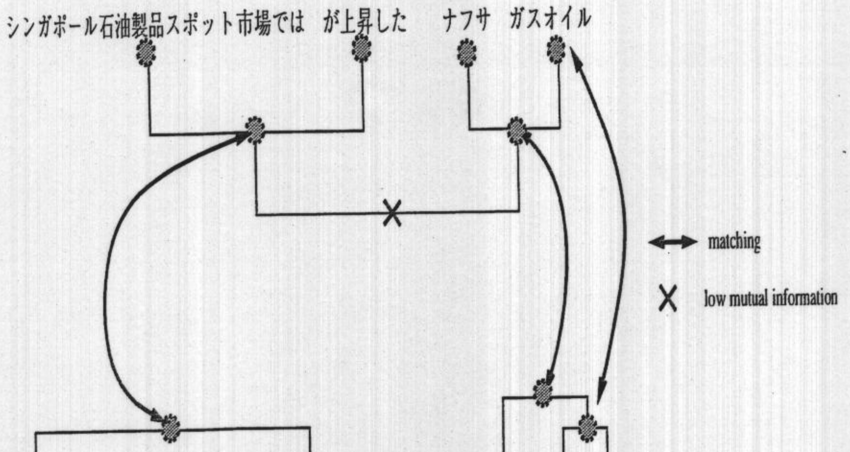

シンガポール石油製品スポット市場では
が上昇した
ナフサ
ガスオイル
matching
low mutual information
rose
on the oil products spot marcket in Singapore
Naphtha
gas
oil

Figure 3.4. Constructing Flexible Collocations

The first two steps construct monolingual similarity trees of word chunks in sentences. The third step iteratively evaluates the bilingual similarity of word chunk combinations by using the above trees. Because word chunks in two languages are registered in index trees, we can also efficiently compute the co-occurrence and mutual information between a Japanese group of chunks and an

English group of chunks. The node matching process continues while mutual information of nodes increases.

Consider the example below, in which the underlined word chunks construct a flexible collocation (シンガポール石油製品スポット市場では～が上昇した / ～ rose ～ on the oil products spot market in Singapore). First, two similarity trees are constructed as shown in Figure 3.4. Node matching is then iteratively attempted by computing mutual information for groups of word chunks. In the present implementation, the system combines three word chunks at most. The technique we use is similar to the parsing-based methods for extracting bilingual collocation [59]. Our method replaces the parse trees with similarity trees and thus avoids the combinatorial explosion inherent to parsing-based methods.

Example:

シンガポール石油製品スポット市場では ナフサとガスオイル が上昇した

Naphtha and gas oil rose on the oil products spot market in Singapore

3.3. Experimental Results

No.	Contents	Sentence Pairs
1	*Stock Market Bulletin*	1114
2	*Scientific Review (Brain Science)*	1766
3	*Scientific Review (Various Areas)*	75000

Table 3.4. Test Data

We performed an evaluation of the proposed method by using quite different data sets. One is scientific articles taken from 'Scientific American' and its Japanese version 'Nikkei Science'[5]. Although these data contain about 10% of

[5] We would like to thank Nikkei Science Co. for permitting us to use the data.

multiple sentence alignment (i.e., 1-2, 2-1, 1-3, etc.), almost all contents are literally translated. The other text is very challenging: Japanese stock market bulletins and their English abstracts. The sample data displayed in Table 3.1 show that much of the Japanese content is skipped in the English abstract. Table 3.4 summarizes the data used in the experiment. Text 1 and Text 2 were used to evaluate the induced collocations and Text 3 to estimate computational efficiency of the proposed method.

As for Text 3, the total time required for extracting fixed and flexible collocations was 2 hours and 14 minutes on a SPARC Station 20 Model HS21. From this result, we may conclude that the proposed method is very efficient and can be applied to voluminous corpora.

3.3.1 Fixed Collocations

No.	Extracted Collocations	Precision
1	397	72.3 %
2	929	88.5 %

Table 3.5. Results for Fixed Collocations

We extracted fixed collocations from Text 1 and Text 2 by using the same agenda file[6].

Table 3.5 displays the results. Text 1 yielded 397 collocations, 72.3% of which were correct. Text 2 yielded 929 collocations, 88.5% of which were correct. Unlike Precision, it is generally difficult to evaluate Recall; we can get a rough idea by comparing the number of extracted collocations and that of given sentence pairs. Text 1 generates a collocation every third sentence, while Text 2 does it every second sentence. The difference between two texts would seem to originate from the rigidness of translations. That is, the more literal translation are, the more precision and recall can be attained.

[6]The actual agenda sequence is (5,6.0), (5,5.5), (4,6.5), (4,6.0), (4,5.5), (4,5.0), (3,6.0), (3,5.5), (3,5.0), (2,6.5), (2,6.0), (2,5.5), (2,5.0), (2,4.5), (2,4.0).

	Text 1		Text 2	
Word Length	Jpn	Eng	Jpn	Eng
1	144	156	480	580
2	126	117	299	253
3	62	64	102	53
4	31	38	37	19
5	21	7	10	11
6	9	2	0	9
7	3	6	1	2
8	0	2	0	0
9	0	2	0	1
10	1	0	0	0
11	0	0	0	1
12	0	0	0	0
13	0	0	0	0
14	0	1	0	0
15	0	2	0	0

Table 3.6. Word Length of Fixed Collocations

Table 3.6 demonstrates the word length of extracted collocations. Outputs from Text 1 tend to be longer than those from Text 2. This is because stock market collocations contain a variety of constructions: noun phrases, verb phrases, prepositional phrases and sentence-level collocations. On the other hand, noun phrase collocations are the majority in scientific articles. Table 3.7 and Table 3.8 exemplify the most confident collocations induced from Text 1 and Text 2, respectively.

Many domain-specific jargon words are found in both Tables. In particular, it is interesting to notice the variety of fixed collocations from Text 1. For example, No 9 in Table 3.7 means 'Tokyo Gold Future market ended trading for the day', but was never written as such. As well as No. 9, a number of sentence-level collocations were also extracted from Text 1. No. 9, No. 18, No. 23, No. 26, No. 35, No. 56 and No. 67 are typical heads of the stock market report. These expressions appear everyday in stock market reports. These collocations are useful for translators, but greatly differ from domain to domain. Thus, it is generally difficult to hand-compile a dictionary that contains these kinds of collocations. Because our method automatically extracts these collocations, it will be of significant use in compiling domain-specific dictionaries. In contrast, domain-specific noun phrases are extracted from Text 2 with very high precision. In the field of science and technology, these technical terms play a central role in the translating process.

Although conventional methods focus on noun phrases or try to encompass all kinds of collocations at the same time, fixed collocation is an important class of collocation, as shown above. It is useful to intensively study fixed collocations because the collocation of more complex structures is difficult to learn regardless of the method used.

3.3.2 Flexible Collocations

We extracted flexible collocations only from Text 1 because scientific articles do not seem to contain many flexible collocations. The experiment yielded 87 flexible collocations, 36.8% of which were correct. Table 3.9 exemplifies the collocations acquired from Text 1. No. 1 to No. 4 are typical expressions in stock market reports. These collocations are extremely useful for template-based machine translation systems. No. 5 is an example of a useless collocation. Both Omron and Sumitomo Forestry are company names that co-occur frequently in stock market reports, but these two companies have no direct relation. In fact, more than half of all flexible collocations acquired were like No. 5. To remove useless collocations, we need a robust parser to find out the direct modifier/modificant relations. We are planing to apply the dependency parser [35] described in Chapter 5.

No.	Japanese	English	No.	Japanese	English
1	東京外為	Tokyo Forex	38	債券相場	bonds and bond futures
2	控えて	ahead of	39	公的資金	public funds
3	マルク	German mark	40	機関投資家	institutional investors
4	ＪＡＦＣＯ	Japan Associated Finance	41	中心限月	benchmark
5	半面	in contrast	42	半導体関連株	semiconductor-related stocks
6	模様眺め気分が強い	remained sidelined watching	43	外国人投資家	foreign investors
7	警戒感	fear	44	ハイテク株	high-tech stocks
8	手控えられている	awaiting	45	売買高	turnover
9	東京貴金属大引け	Tokyo Gold futures Cls:	46	小口の売り	small-lot selling
10	小動き	slow	47	更新した	record high
11	見送り気分	wait-and-see mood	48	指標銘柄	benchmark
12	アジアのディーラー間ドル建て相場	Loco-London gold	49	見送り気分が強く	low
13	対マルクで	against mark	50	東証２部	Tokyo Stocks 2nd Sec
14	ＣＢＱ平均	Convertible bonds	51	軟調	were weak
15	証券会社の自己売買部門	dealers	52	個人投資家	individual investors
16	売買高	trading volume	53	経常増益	pretax profit
17	利回り銘柄	high-yielders	54	第１部相場	The first section of TSE
18	日経３００先物・後場寄り	Nikkei 300 futures Aft-opg:	55	日経平均株価	the Nikkei stock average
19	前引け	mng-cls:	56	東証ＣＢ寄り付き・	Tokyo CBs Opg:
20	取り引きを終えた	contract ended	57	長期国債	long-term government bonds
21	緊急経済対策	economic stimulus package	58	で取引されている	were traded at
22	終り値は	closed at	59	輸入企業	importers
23	先物・大引け	futures cls:	60	は堅調	advanced
24	債券相場	bond market	61	買い戻し	covering
25	相場	convertible bonds	62	昭電工	Showa Denko
26	先物・後場寄り	Nikkei futures aft-opg:	63	売買高は	volume was
27	嫌気し	disheartened by	64	年初来高値を更新し	hit a new year's high
28	への期待から	on expectation of	65	与党３党	ruling coalition
29	もしっかり	edged up	66	住特金	Sumitomo Special Metals
30	ハイテク株	high-tech shares	67	日経３００先物・前引け	Nikkei 300 futures Mng-cls:
31	様子見気分	wait-and-see mood	68	＜大証＞	OSE
32	住友林	Sumitomo Forestry	69	年初来安値	year's low
33	日米自動車交渉	U.S.-Japan auto talks	70	日経国際商品指数概況・	Nikkei World Commodities:
34	物色	speculative buying of	71	先物・寄り付き	Nikkei futures opg:
35	東証外国部・前引け	Tokyo foreign stocks mng:	72	前引け	morning close
36	金利	interest rates	73	小動きに終始した	inched up
37	日米自動車交渉	the Japan-U.S. auto dispute			

Table 3.7. Fixed Collocations Induced from Text 1

3.4. Related Work

A number of studies have attempted to extract bilingual collocations from parallel corpora. These studies can be classified into two general approaches. One is based on the full parsing techniques. [59] proposed a method to find out phrase-level correspondences while resolving syntactic ambiguities at the same time. Their methods determine phrase correspondences by using the phrase structures of the two languages and existing bilingual dictionaries. Unfortunately, these approaches are promising only for the comparatively short sentences that can be analyzed by a simple Chart parser [44]. It will be interesting to use practical parsers, as mentioned in Chapter 5, for the parser-based method.

No.	Japanese	English	No.	Japanese	English
1	アデニル酸シクラーゼ	adenylyl cyclase	38	アドレナリンレセプター	adrenoceptors
2	シナプス前促通	presynaptic facilitation	39	オートラジオグラフィー	autoradiography
3	1次 視覚 野	primary visual cortex	40	カリフォルニア大学ロサンゼルス校	University of California at Los Angeles
4	古典的 条件 付け	classical conditioning	41	ヘッブ型シナプス	Hebb synapses
5	長期 記憶	long-term memory	42	受容野	receptive fields
6	調節 ニューロン	modulatory neuron	43	一卵性双生児	identical twins
7	脳 梁	corpus callosum	44	異性愛男性	heterosexual men
8	テトロドトキシン	tetrodotoxin	45	機能詞	functors
9	プロテインキナーゼ	protein kinase	46	顔細胞	face cells
10	フリーラジカル	free radicals	47	失語症患者	aphasic patients
11	神経 原 線 維 変化	neurofibrillary tangles	48	潜在的学習	implicit learning
12	接続詞	conjunctions	49	潜在的再現像	implicit representations
13	辺縁系	limbic system	50	脊椎動物	vertebrates
14	カテコールアミン	catecholamines	51	遅延反応課題	delayed-response tasks
15	注視する	gaze	52	カリフォルニア大学サンディエゴ校	University of California at San Diego
16	コロンビア大学	Columbia University	53	脳活動	brain activity
17	パーキンソン病	Parikinson's disease	54	連想記憶	associative memory
18	モノアミン	monoamines	55	老人斑	senile plaques
19	外套 膜	mantle shelf	56	ポスドク	postdoctoral fellow
20	皮質下	subcortical	57	樹状突起	dendrites
21	アミロイドタンパク質	amyloidprotein	58	側枝	side branches
22	ノルアドレナリン	norepinephrine	59	米国立精神衛生研究所	National Institute of Mental Health
23	シナプス前終末	presynaptic terminals	60	盲点	blind spot
24	眼優位性カラム	ocular dominance columns	61	幻覚	hallucinations
25	解剖学的	anatomic	62	運動ニューロン	motor neuron
26	抗うつ薬	antidepressants	63	NMDA 受容体	NMDA receptors
27	神経伝達物質	neurotransmitters	64	媒介系	mediation systems
28	主溝	principal sulcus	65	高次精神機能	higher mental functions
29	条件付け	conditioning	66	両側	bilateral
30	新皮質	neocortex	67	グルタミン酸受容体	glutamate receptor
31	臨界期	critical period	68	シンガー	Singer
32	染色体	chromosome	69	ジヒドロテストステロン	dihydrotestosterone
33	痴呆	dementia	70	ホルモン環境	hormonal environment
34	短期記憶	short-term memory	71	一夫多妻	polygynous
35	長期増強	long-term potentiation	72	一酸化炭素中毒	carbon monoxide poisoning
36	無条件刺激	unconditioned stimulus	73	音刺激	auditory stimuli
37	クロサピン	clozapine	74	遅発性ジスキネシア	tardive dyskinesia

Table 3.8. Fixed Collocations Induced from Text 2

The other approach to extracting bilingual collocations involves simple statistics. [29] acquired bilingual word correspondences without sentence alignment. Although these methods are robust and assume no information source, their outputs are just word-word correspondences. [50] and [49] extracted noun phrase (NP) correspondences from aligned parallel corpora. In [50], NPs in English and French texts are first extracted by a NP recognizer. Their correspondence probabilities are then gradually refined by using an EM-like iteration algorithm. [49] first extracted Japanese NPs in the same way, and then combined statistics with a bilingual dictionary for MT to find NP correspondences. Although these approaches attained high accuracy for the task considered, the most crucial knowledge for MT is more complex correspondences such as NP-VP correspondences

No.	Japanese	English
1	東京外為 ～ 円 ～	Tokyo Forex ～ Dollar at ～ yen
2	ドルは ～ で取り引きを終えた	The U.S. currency was quoted at ～
3	～ が売られ ～ も安い	～ were sold ～ dropped as well
4	日銀が ～ の供給を通知した	Bank of Japan injected ～
5	オムロン ～ 住友林 ～	Omron ～ Sumitomo Forestry ～

Table 3.9. Samples of Flexible Collocations

and sentence-level correspondences. It seems difficult to extend these statistical methods to a broader range of collocations because they are specialized to NPs or single words.

3.5. Summary

This chapter has described a new method for learning bilingual collocations from parallel corpora. Our method consists of two steps: (1) extracting useful word chunks by the word-level sorting technique and (2) constructing bilingual collocations by combining these chunks. This architecture reflects the fact that fixed collocations play a more crucial role than accepted in previous research. Our method not only extracts fixed collocations with high precision but also reduces the combinatorial explosion that would be otherwise considered inescapable in extracting flexible collocations.

Chapter 4

Mistake-Driven Mixture Approach to Morphological Analysis

This chapter deals with the stochastic morphological analysis system (part-of-speech (POS) tagger) [13, 50, 11, 4, 63]. The stochastic approach generally attains 94 to 96% accuracy and has the potential to replace the labor-intensive compilation of linguistics rules by using an automated learning algorithm. However, a practical tagger requires more accuracy because morphological analysis is an inevitable pre-processing step for almost all practical systems.

To derive a new stochastic tagger, we have two options since stochastic taggers generally comprise two components: *word model* and *tag model*. The word model is a set of probabilities that a word occurs with a tag (part-of-speech) when given the preceding words and their tags in a sentence. On the other hand, the tag model is a set of probabilities that a tag appears after the preceding words and their tags.

The first option is to construct more sophisticated word models. [11] reports that their model considers the roots and suffixes of words to greatly improve tagging accuracy for English corpora. However, the word model approach has the following shortcomings:

- For agglutinative languages such as Japanese and Chinese, the simple Bayes transfer rule is inapplicable because the word length of a sentence is not fixed in all possible segmentations[1]. We can only use simpler word models in these languages.

- Sophisticated word models largely depend on the target language. It is time-consuming to compile fine-grained word models for each language.

The second option is to devise a new tag model. [73] have introduced a variable-memory-length tag model. Unlike conventional bi-gram and tri-gram models, the method selects the optimal length by using the context tree [68] originally introduced for use in data compression [19]. Although the variable-memory length approach significantly reduces the number of parameters, tagging accuracy is only as good as conventional methods. Why didn't the method have higher accuracy? The crucial problem for current tag models is the set of colloquial sequences of words that cannot be captured by their tags only. Because the maximal likelihood estimator (MLE) emphasizes the most frequent connections, an exceptional connection is placed in the same class as a frequent connection.

To tackle this problem, we introduce a new tag model based on the *mistake-driven mixture* (boosting) of hierarchical tag context trees. Compared to Schütze and Singer's context tree [73], the hierarchical tag context tree is extended in that the context is represented by a hierarchical tag set (i.e., NTT $<$ proper noun $<$ noun). This is extremely useful in capturing exceptional connections that can be detected only at the word level.

To make the best use of the hierarchical context tree, the boosting method imitates the process in which linguists incorporate exceptional connections into hand-crafted rules: They first construct coarse rules that seem to cover a broad range of data. They then try to analyze data by using the rules and extract exceptions that the rules cannot handle. Next they generalize the exceptions and refine the previous rules. The following two steps abstract the human algorithm for incorporating exceptional connections.

1. Construct temporary rules that seem to well generalize the given data.

[1]In $P(w_i|t_i) = \frac{P(w_i)P(t_i|w_i)}{P(t_i)}$, $P(w_i)$ cannot be considered identical for all segmentations.

2. Try to analyze data with the constructed rules and extract the exceptions that cannot be correctly handled, then return to the first step and focus on the exceptions.

To put the above idea into our learning algorithm, the Adaboost algorithm attaches a weight vector to each example and iteratively performs the following two procedures in the training phase:

1. Constructing a context tree based on the current data distribution (weight vector)

2. Updating the distribution (weight vector) by focusing on data not well predicted by the constructed tree. More precisely, the algorithm reduces the weight of examples that are correctly handled.

For the prediction phase, the algorithm then outputs a final tag model by mixing all of the constructed models according to their performance. By using a hierarchical tag context tree, the constituents of a series of tag models gradually change from broad coverage tags (e.g., noun) to specific exceptional words that cannot be captured by general tags. In other words, the method incorporates not only frequent connections but also infrequent ones that are often considered to be exceptional.

The organization of the chapter is as follows. Section 4.1 describes the stochastic POS tagging scheme and hierarchical tag setting. Section 4.2 presents a new probability estimator that uses a hierarchical tag context tree, and Section 4.3 explains the Adaboost algorithm. Section 4.5 reports our evaluation using Japanese newspaper articles. We tested several tag models by keeping all other conditions (i.e., dictionary and word model) constant. The experimental results show that the proposed method outperforms both hand-crafted and conventional statistical methods. Section 4.6 describes related works and Section 4.7 gives the summary.

4.1. Preliminaries

4.1.1 Basic Equation

In this section, we will briefly review the basic equations for part-of-speech tagging and introduce hierarchical-tag setting.

The tagging problem is formally defined as finding a sequence of tags $t_{1,n}$ that maximize the probability of input string L.

$$argmax_{t_{1,n}} P(w_{1,n}, t_{1,n}|L) = argmax_{t_{1,n}} \frac{P(w_{1,n}, t_{1,n}, L)}{P(L)}$$
$$\Leftrightarrow argmax_{t_{1,n}, w_{1,n} \in L} P(t_{1,n}, w_{1,n})$$

We break out $P(t_{1,n}, w_{1,n})$ as a sequence of the products of tag probability and word probability.

$$P(t_{1,n}, w_{1,n}) = P(w_{1,n})P(t_1|w_1)P(w_2|t_1, w_1)P(t_2|t_1, w_{1,2})$$

$$\cdots P(t_n|t_{1,n-1}, w_{1,n})P(w_n|t_{1,n-1}, w_{1,n-1})$$

$$= \prod_{i=1}^{n} P(w_i|t_{1,i-1}, w_{1,i-1})P(t_i|t_{1,i-1}, w_{1,i})$$

By approximating word probability as constrained only by its tag, we obtain equation (4.1). Equation (4.1) yields various types of stochastic taggers. For example, bi-gram and tri-gram models approximate their tag probability as $P(t_i|t_{i-1})$ and $P(t_i|t_{i-1}, t_{i-2})$, respectively. In the rest of the chapter, we assume all tagging methods share the word model $P(w_i|t_i)$ and differ only in the tag model $P(t_i|t_{1,i-1}, w_{1,i})$.

$$argmax_{t_{1,n}, w_{1,n} \in L} \prod_{i=1}^{n} P(t_i|t_{1,i-1}, w_{1,i})P(w_i|t_i) \tag{4.1}$$

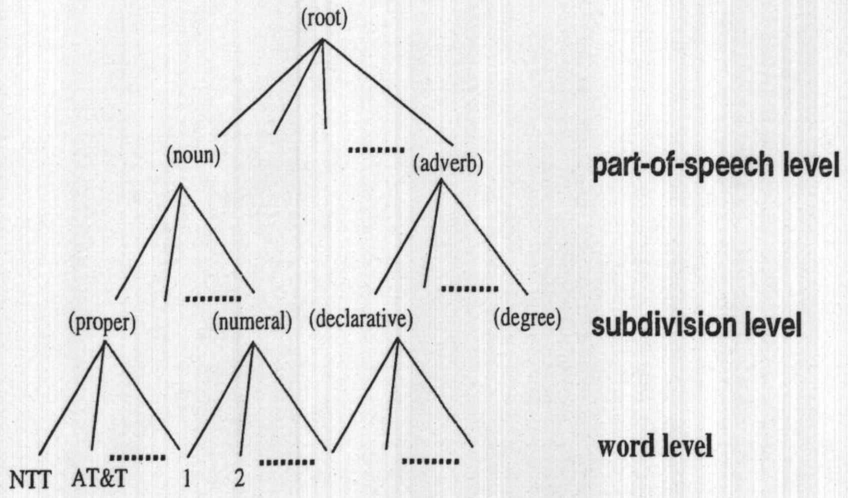


Figure 4.1. Hierarchical Tag Set

4.1.2 Hierarchical Tag Set

To construct a tag model that captures exceptional connections, we have to consider word-level context as well as tag-level context. In a more general form, we introduce a tag set that has a hierarchical structure. Our tag set has a three-level structure (Fig. 4.1). The topmost and the second levels of the hierarchy are the *part-of-speech* level and the part-of-speech *subdivision* level, respectively. Although stochastic taggers usually make use of the *subdivision* level, the *part-of-speech* level is remarkably robust against data sparseness. The bottom level is the *word* level and is indispensable in coping with exceptional and colloquial sequences of words. Our objective is to construct a tag model that precisely evaluates $P(t_i|t_{1,i-1}, w_{1,i})$ (in equation (4.1)) by using the three-level tag set.

To construct this model, we have to answer the following questions.

1. Which level is appropriate for t_i ?

2. Which length is to be considered for $t_{1,i-1}$ and $w_{1,i}$?

3. Which level is appropriate for $t_{1,i-1}$ and $w_{1,i}$?

To resolve the first question, we fix t_i at the *subdivision* level, as is done in other tag models. The second and third questions are resolved by introducing *hierar-*

chical tag context trees and a boosting method that are respectively described in Section 4.2 and Section 4.3.

Before moving to the next section, let us define the *basic tag set.* If all words are considered context candidates, the search space will be enormous. Thus, it is useful for the tagger to constrain the candidates to frequent open class words and closed class words. The *basic tag set* is the set of the most detailed context elements that comprises the *words* selected above and the part-of-speech *subdivision* level.

4.2. Hierarchical Tag Context Tree

A hierarchical tag context tree is constructed by a two-step methodology. The first step produces a context tree by using the basic tag set. The second step then produces the hierarchical tag context tree. It generalizes the basic tag context tree and avoids over-fitting the data by replacing excessively specific context in the tree with more general tags. Finally, the generated tree is transformed into a finite automata to improve tagging efficiency [69].

4.2.1 Constructing a Basic Tag Context Tree

In this section, we construct a basic tag context tree. Before going into the details of the algorithm, we will briefly explain the context tree by using a simple binary case. The context tree was originally introduced in the field of data compression [68, 81, 19] to represent how many times and in what context each symbol appeared in a sequence of symbols. Figure 4.2 exemplifies two context trees comprising binary symbols *'a'* and *'b'*. T(4) is constructed from the sequence *'baab'* and T(6) from *'baabab'*. The root node of T(4) explains that both *'a'* and *'b'* appeared twice in *'baab'* when no consideration is taken of previous symbols. The nodes of depth 1 represent an order 1 (bi-gram) model. The left node of T(4) indicates that both *'a'* and *'b'* appeared only once after symbol *'a'*, while the right node of T(4) indicates that only *'a'* occurred once after *'b'*. In the same way, the node of depth 2 in T(6) represents an order 2 (tri-gram) context model.

This binary tree can be directly extended to a basic tag context tree. In

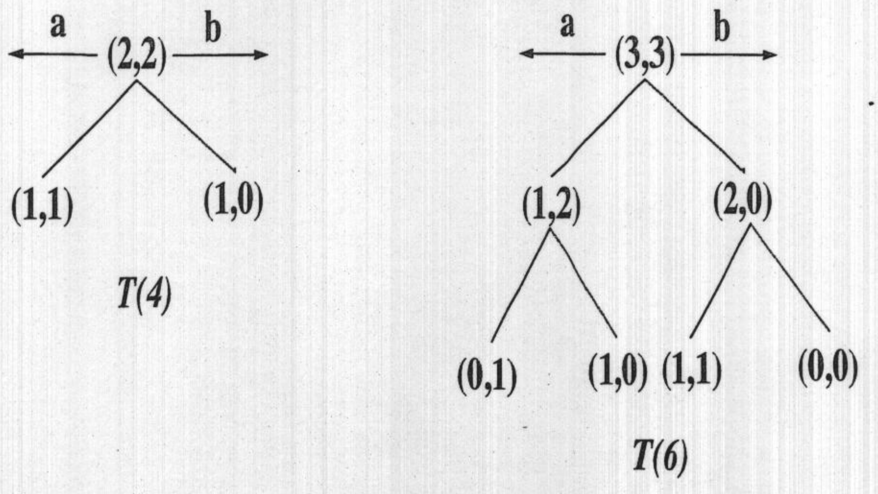


Figure 4.2. Context Trees for *'baab'* and *'baabab'*

this case, context symbols *'a'* and *'b'* are replaced by an element of the basic tag set, and the frequency table of each node then consists of the part-of-speech *subdivision* set.

Table 4.1 shows the algorithm *construct-btree*, which constructs a basic tag context tree. Let a set of *subdivision* tags be $s_1, \cdots, s_n$. Let $weight[t]$ be a weight vector attached to the tth example *x(t)*. Initial values of $weight[t]$ are set to 1 for all t.

4.2.2 Constructing a Hierarchical Tag Context Tree

This section describes how a hierarchical tag context tree is constructed from a basic tag context tree. Before describing the algorithm, we prepare some definitions and notations.

Let A be a part-of-speech *subdivision* set. As described in the previous section, frequency tables of each node consist of the set A. At any node s of a context tree, let $n(a|s)$ and $\hat{P}(a|s)$ be the count of element a and its probability, respectively.

$$\hat{P}(a|s) = \frac{n(a|s)}{\sum_{b \subseteq A} n(b|s)}$$

We introduce an information-theoretical criteria $\Delta(sb)$ [80] to evaluate the gain of expanding a node s by its daughter sb.

1. the only node, the root, is marked with the count table (c(s_1,λ),$\cdots$, c(s_n,λ) = (0,$\cdots$,0)).

2. Apply the following recursively. Let T(t-1) be the last constructed tree with counts of nodes z, (c(s_1,z),$\cdots$,c(s_n,z)). After the next symbol with *subdivision x(t)* is observed, generate the next tree T(t) as follows. Follow T(t-1), starting at the root and taking the branch indicated by each successive symbol in the past sequence by using basic tag level. For each node z visited, increment the component count c($x(t)$,z) by *weight*[*t*]. Continue until node w is a leaf node.

3. If w is a leaf, extend the tree by creating new leaves: c($x(t)$,ws_1)=$\cdots$=c($x(t)$,ws_n) = *weight*[*t*], c($\overline{x(t)}$,ws_1)=$\cdots$= c($\overline{x(t)}$,ws_n)=0. Define the resulting tree to be T(t).

Table 4.1. Algorithm *construct-btree*

$$\Delta(sb) = \sum_{a \subseteq A} n(a|sb) log \frac{\hat{P}(a|sb)}{\hat{P}(a|s)} \tag{4.2}$$

$\Delta(sb)$ is the difference in optimal code lengths when symbols at node sb are compressed by using probability distribution $\hat{P}(\cdot|s)$ at node s and $\hat{P}(\cdot|sb)$ at node sb. Thus, the larger $\Delta(sb)$ is, the more meaningful it is to expand a node by sb.

Now, we go back to the hierarchical tag context tree construction. As illustrated in Figure 4.3, the construction process amounts to the iterative selection of b out of *word level*, *subdivision*, *part-of-speech* and *null* (no expansion). Let us look at the procedure from the information-theoretical viewpoint. Breaking out equation (4.2) as equation (4.2.2), $\Delta(sb)$ is represented as the product of the frequencies of all subdivision symbols at node sb and Kullback-Leibler (KL) divergence.

$$\Delta(sb) = n(sb) \sum_{a \subseteq A} \frac{n(a|sb)}{n(sb)} log \frac{\hat{P}(a|sb)}{\hat{P}(a|s)} =$$

n(sb) $\sum_{a \subseteq A} \hat{P}(a|sb) log \frac{\hat{P}(a|sb)}{\hat{P}(a|s)} = n(sb) D_{KL}(\hat{P}(\cdot|sb), \hat{P}(\cdot|s))$(4.3)

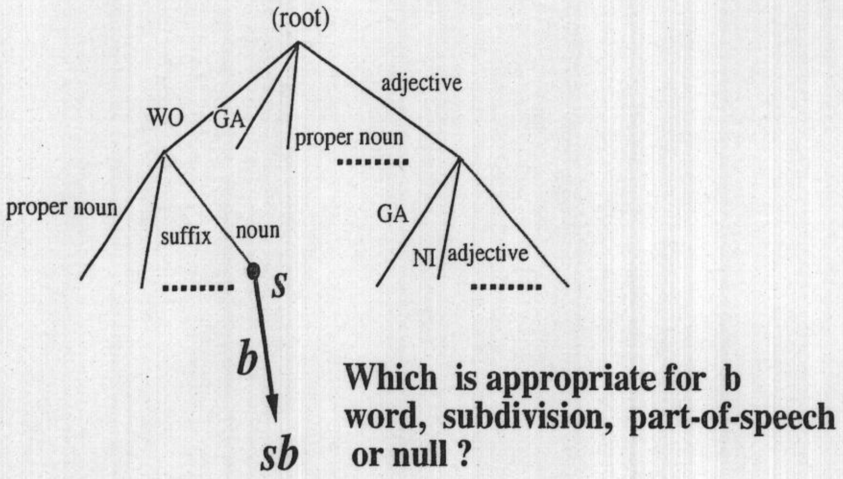


Figure 4.3. Constructing Hierarchical Tag Context Tree

Because the KL divergence defines a distance measure between probability distributions, $\hat{P}(\cdot|sb)$ and $\hat{P}(\cdot|s)$, there is the following trade-off between the two terms of equation (4.2.2).

- The more general b is, the more *subdivision* symbols appear at node sb.
- The more specific b is, the more $\hat{P}(\cdot|s)$ and $\hat{P}(\cdot|sb)$ differ.

By using the trade-off, the optimal level of b is selected.

Table 4.2 summarizes the algorithm *construct-htree* that constructs the hierarchical tag context tree. First, *construct-htree* generates a basic tag context tree by calling *construct-btree*. Assume that the training examples consist of a sequence of triples, $< p_t, s_t, w_t >$, in which p_t, s_t and w_t represent part-of-speech, subdivision and word, respectively. Each time the algorithm reads an example, it first reaches current leaf node s by following the past sequence, computes $\Delta(sb)$, and then selects the optimal b. The initially constructed basic tag context tree is used to compute $\Delta(sb)$s.

```
1. Initialize weight[j] = 1 for all examples $j$
   $t = 1$

2. call construct-btree

3. do
      Read $t$th example $x_t(< p_t, d_t, w_t >)$
      Follow $x_{t-1}$,$x_{t-2}$, $\cdots$, $x_{t-(i-1)}$ and Reach leaf node $s$
      $low = sw_{t-i}, \quad high = sd_{t-i}$
      while($max(\Delta(low), \Delta(high)) \geq Threshold$) {
        if($\Delta(low) \geq \Delta(high)$)
          Expand the tree by the node $low$
        else if(high==$sp_{t-i}$)
          Expand the tree by the node $high$
        else low = $sd_{t-i}$, high = $sp_{t-i}$
      }
      $t = t + 1$
   while($x_t$ is not empty)
```

Table 4.2. Algorithm *construct-htree*

4.3. Mistake-Driven Mixture of Hierarchical Tag Context Trees

Up to this section, we introduced a new tag model that uses a single hierarchical tag context tree to cope with the exceptional connections that cannot be captured by just the part-of-speech level. However, this approach has a clear limitation: the exceptional connections that do not occur so often cannot be detected by the single tree model. In such a case, the first term $n(sb)$ in equation (4.2.2) is enormous for general b and the tree is expanded by using more general symbols.

To overcome this limitation, we introduced the *mistake-driven mixture* approach that uses the Adaboost algorithm summarized in Table 4.3. The algorithm constructs T context trees and outputs a final tag model. It sets the weights to 1 for all examples and repeats the following procedures T times. The algorithm first construct a hierarchical context tree by using the current weight vector. Example data are then tagged by the tree and the weights of correctly handled examples are reduced by (A). Finally, the final tag model is constructed by mixing T trees according to equation (B).

```
1. Input: sequence of $N$ examples $< p_1, d_1, w_1 >, \ldots, < p_N, d_N, w_N >$
   in which $p_i$, $d_i$ and $w_i$ represent part-of-speech, subdivision and word, respectively.
2. Initialize the weight vector weight[i] =1 for $i = 1, \ldots, N$
3. Do for $t = 1, 2, \ldots, T$
      Call construct-htree providing it with the weight vector weight and
      Construct a part-of-speech tagger $h_t$
      Let $Error$ be a set of examples that are not identified by $h_t$
      Compute the error rate of $h_t$: $\epsilon_t = \sum_{i \subset Error} weight[i] / \sum_{i=1}^{N} weight[i]$
      $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
      For examples correctly predicted by $h_t$, update the weights vector to be
            weight[i] = weight[i]$\beta_t$    (A)
4. Output a final tag model $h_f = \sum_{t=1}^{T} (log \frac{1}{\beta_t}) h_t / \sum_{t=1}^{T} (log \frac{1}{\beta_t})$    (B)
```

Table 4.3. Algorithm *Adaboost*

By using the mistake-driven mixture approach, the constituents of a series of hierarchical tag context trees gradually change from broad coverage tags (e.g., noun) to specific exceptional words that cannot be captured by *part-of-speech* and *subdivisions.* The method, by mixing different levels of trees, incorporates not only frequent connections but also infrequent ones that are often considered colloquial without over-fitting the data.

4.4. Margin Theory and Adaboost Algorithm

This section briefly overviews the two theorems related to the generalization ability of the Adaboost algorithm [72] in classification. The first theorem is relevant for any voting method such as boosting and bagging [2]. It explains the generalization error of any aggregating algorithm in terms of the margin [79, 18] for the training examples and the complexity of the base hypothesis (no dependence on the number of iterations). The second theorem gives the probability bound of the small margin in terms of the training errors of the base hypotheses.

The margin intuitively represents the distance to a classification boundary. Figure 4.4 illustrates a simple two-class case, in which black and white circles are training examples from two distinctive classes. The examples with a line are small-margin examples. Dotted lines represent a small positive margin because they are classified correctly by the boundary. On the other hand, solid lines represent a negative-valued margin because they are misclassified. As shown in Figure 4.4, the larger the margin of an instance is, the easier it is to classify. The following two theorems prove that the generalization error of a aggregated classifier is bounded by the sum of the error for the training examples and a decreasing function of the margins, and that the Adaboost algorithm continuously increases the margin for training examples, respectively.

Let H denote the space from which each base hypothesis is chosen; for example, a context tree or a decision tree of an appropriate size. A base hypothesis $h \in H$ is a mapping from an instance space X to class $\{-1,+1\}$[2]. We assume that examples are generated independently at random according to some fixed but unknown distribution D over $X \times \{-1,+1\}$. The training set is a list of m pairs $S =< (x_1, y_1), \cdots, (x_m, y_m) >$ chosen according to D. We use $P_{(x,y)}[A]$ to denote the probability of the event A when the example (x, y) is chosen according to D and $P_{(x,y)}[A]$ to denote probability with respect to choosing an example uniformly at random from the training set. We abbreviate these by $P_D[A]$ and $P_S[A]$.

We define the convex hull C of H as the set of mappings that can be generated

[2]The following theorems hold for larger finite sets of labels.

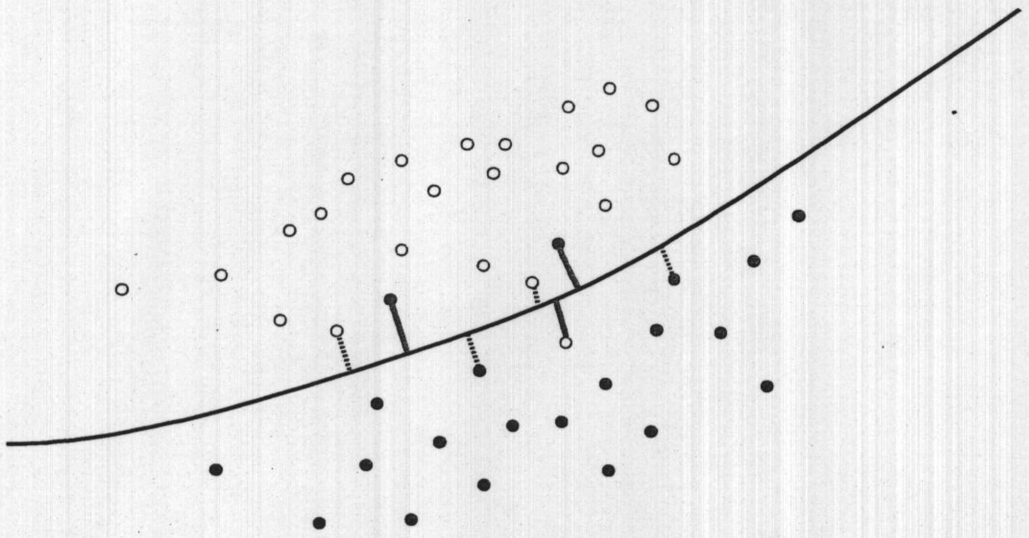


Figure 4.4. Margin of Examples

by taking a weighted average of hypotheses from H:

$$C \doteq \{f : x \mapsto \sum_{h \in H} a_h h(x) \mid a_h \geq 0; \sum_h a_h = 1\}.$$

The majority vote rule that is associated with f gives a wrong prediction for the example (x, y) only if $yf(x) \leq 0$. Also, the margin (feasibility of classification) of an example (x, y) in this case is simply $yf(x)$.

The following **Theorem 4.4.1** implies that with high probability, the generalization error of any majority vote hypothesis can be bounded in terms of the number of training examples with margin below a threshold θ, plus an additional term that depends on the number of training examples, VC-dimension [79, 46] of H and the threshold θ. Note that the theorem applies to every majority vote hypothesis, regardless of how it is computed.

Theorem 4.4.1 *[1, 72] Let D be a distribution over $X \times \{-1, +1\}$, and let S be a sample of m examples chosen independently at random according to D. Suppose the base hypothesis space H has VC-dimension d, and let $\delta > 0$. Assume that $m \geq d \geq 1$. Then with probability at least $1 - \delta$ over the random choice of the* *training set S, every weighted average function $f \in C$ satisfies the following bound for all $\theta > 0$:*

$$P_D[yf(x) \leq 0] \leq P_S[yf(x) \leq \theta] + O(\frac{1}{\sqrt{m}}(\frac{d\log^2(m/d)}{\theta^2} + \log(\frac{1}{\delta}))^{1/2})$$

The following **Theorem 4.4.2** implies that the fraction of training examples for which $yf(x) \leq \theta$ decreases to zero exponentially with the number of base hypotheses (boosting iteration T).

Theorem 4.4.2 *[72] Suppose the base learning algorithm (say, a context tree or a decision tree algorithm), when called by Adaboost, generates hypotheses with weighted training errors $\epsilon_1, \epsilon_2, \cdots, \epsilon_T$. Then for any θ, the following inequality holds.*

$$P_{(x,y)}[yf(x) \leq \theta] \leq \prod_{t=1}^{T} \sqrt{4\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}$$

From the above two theorems, the Adaboost algorithm is found to produce a larger margin hypothesis on the training example set (**Theorem 4.4.2**) and then to yield smaller generalization error (**Theorem 4.4.1**). It can thus deal with the originally small-margin instances that correspond to the infrequent and difficult expressions from the viewpoint of natural language processing.

4.5. Experimental Results

We performed a preliminary evaluation using the first 8,939 Japanese sentences in a year's volume of newspaper articles [57]. We first automatically segmented and tagged these sentences and then revised them by hand. The total number of words in the hand-revised corpus was 226,162. We trained our tag models on the corpora with every tenth sentence removed (starting with the first sentence) and then tested the removed sentences. There were 22,937 words in the test corpus.

As the first indicator of performance, we tested a hand-crafted tag model of JUMAN [52], the most widely used Japanese part-of-speech tagger. The tagging accuracy of JUMAN for the test corpus was only 92.0%. This shows that our corpus is difficult to tag because the corpus contains various genres of texts, ranging from obituaries to editorials.

Next, we compared the mixture of bi-grams and the mixture of hierarchical tag context trees. In this experiment, only post-positional particles and auxiliaries were *word-level* elements of *basic tags* and all other elements were in the *subdivision* level. In contrast, a bi-gram model was constructed by using the *subdivision* level. We set the iteration number T to 5. The results of our experiments are summarized in Figure 4.5.

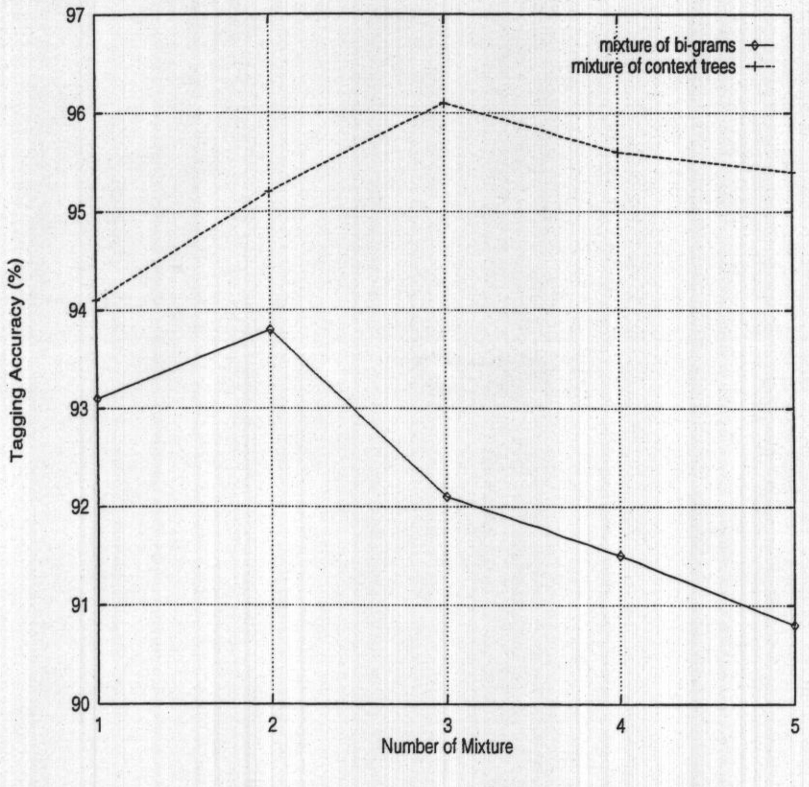


Figure 4.5. Context Tree Mixture vs. Bi-gram Mixture

As a single tree estimator (Number of Mixture = 1), the hierarchical tag context tree attained 94.1% accuracy, while bi-gram yielded 93.1%. A hierarchical tag context tree offers a slight improvement, but not a great deal. This conclusion agrees with Schütze and Singer's experiments that used a context tree of usual part-of-speech.

When we turn to the mixture estimator, a great difference is seen between hierarchical tag context trees and bi-grams. The hierarchical tag context trees produced by the *mistake-driven mixture* approach greatly improved the accuracy, and over-fitting data was not serious. The best and worst performances were 96.1% (Number of Mixture = 3) and 94.1% (Number of Mixture = 1), respectively. On the other hand, the performance of the bi-gram mixture was not satisfactory. The best and worst performances were 93.8% (Number of Mixture = 2) and 90.8% (Number of Mixture = 5), respectively.

From these results, we may say that exceptional connections are well captured by hierarchical context trees but not by bi-grams. Bi-grams of *subdivision* are too general to selectively detect exceptions.

4.6. Related Work

Although statistical natural language processing has mainly focused on Maximum Likelihood Estimators, [65] proposed a mixture approach to predict next words by using the Context Tree Weighting (CTW) method [81]. The CTW method computes probability by mixing subtrees in a single context tree in Bayesian fashion. Although the method is very efficient, it cannot be used to construct hierarchical tag context trees.

Various kinds of *re-sampling* techniques have been studied in statistics [24, 25] and machine learning [2, 37, 27]. The Adaboost was designed to construct a high-performance predictor by iteratively calling a weak learning algorithm (which is slightly better than a random guess). An empirical work reports that the method greatly improved the performance of decision-tree, k-nearest-neighbor, and other learning methods given relatively simple and small data [26]. We used the algorithm to detect exceptional connections and first proved that such a re-sampling method is also effective for a practical application using a large amount of data. The next step is to fill the gap between theory and practice. Most theoretical work on re-sampling assumes *i.i.d.* (identically, independently distributed) samples. This is not a realistic assumption in part-of-speech tagging and other NL applications because they usually involve optimization that uses dynamic programming. An interesting future research direction is to construct a theory that handles the Markov processes.

4.7. Summary

This chapter has described a new tag model that uses a *mistake-driven mixture* approach to produce *hierarchical tag context trees* that can deal with exceptional connections whose detection is not possible at the simple part-of-speech level. Our experimental results show that combining *hierarchical tag context trees* with the *mistake-driven mixture* approach is effective for 1. incorporating exceptional connections and 2. avoiding data over-fitting. Although we have focused on part-of-speech tagging in this chapter, the *mistake-driven mixture* method is useful for other applications. The next chapter describes a Japanese dependency parser that boosts decision trees.

Chapter 5

Mistake-Driven Mixture Approach to Dependency Analysis

With the recent availability of large annotated corpora, there is growing interest in stochastic parsing methods to replace labor-intensive rule compilation by automated learning algorithms. Conventional parsers with practical levels of performance require a number of sophisticated rules that have to be hand-crafted by human linguists. It is time-consuming and cumbersome to maintain these rules for the following two reasons.

- The rules are specific to the application domain.
- Specific rules for handling infrequent expressions create side effects. Such rules often deteriorate the overall performance of the parser. It is generally difficult to decide whether to include a specific rule, particularly when the number of rules becomes large.

The stochastic approach, on the other hand, has the potential to overcome these difficulties. Because it induces stochastic rules to maximize overall performance against training data, it not only adapts to any application domain but also avoids over-fitting the data. Now that machine learning techniques are mature enough to deal with real-world applications, it is promising to construct practical parsers by using machine learning methods.

In the late 80s and early 90s, the induction and parameter estimation of probabilistic context free grammars (PCFGs) from corpora were intensively studied. Because these grammars comprise only nonterminal and part-of-speech tag symbols, their performances were not good enough to be used in practical applications [9]. A broader range of information, in particular lexical information, was found to be essential in disambiguating the syntactic structures of real-world sentences. SPATTER parser [56] replaced the pure PCFG with transformation (extension and labeling) rules that are augmented with a number of lexical attributes. The parser controlled applications of each rule by using the lexical constraints resulting from using a decision tree algorithm [3]. The SPATTER parser attained 86% accuracy and first made stochastic parsers a practical choice. The other type of high-precision parser, which is based on dependency analysis, was introduced by Collins [16]. Dependency analysis first segments a sentence into syntactically meaningful sequences of words and then considers the modification of each segment. Collins' parser computes the likelihood that each segment modifies the other (2 term relation) by using large corpora. These modification probabilities are conditioned by headwords of two segments, the distance between the two segments and other syntactic features. Although these two parsers have shown similar performance, the keys to their successes are slightly different. SPATTER parser performance greatly depends on the feature selection capability of the decision tree algorithm rather than its linguistic representation. On the other hand, dependency analysis plays an essential role in Collins' parser for efficiently extracting information from corpora.

This chapter describes a practical Japanese dependency parser that uses decision trees. In the Japanese language, dependency analysis has been shown to be powerful because the segment (bunsetsu) order in a sentence is relatively unrestricted compared to European languages.

Japanese dependency parsers generally proceed in three steps.

1. Segment a sentence into a sequence of bunsetsu.

2. Prepare a modification matrix, each value of which represents how one bunsetsu is likely to modify another.

3. Find optimal modifications in a sentence by dynamic programming.

The most difficult step is the second: how to construct a sophisticated modification matrix. With conventional Japanese parsers, the linguist must classify the bunsetsu and select appropriate features to compute modification values. The parsers also suffer from the diversity of application domains and the side effects of specific rules.

Stochastic dependency parsers like Collins', on the other hand, define a set of attributes and condition the modification probabilities by all of the attributes regardless of the bunsetsu type. These methods can encompass only a small number of features if the probabilities are to be precisely evaluated from a finite number of data. Our method constructs a more sophisticated modification matrix by using decision trees. It automatically selects a sufficient number of significant attributes according to the bunsetsu type. We can thus use an arbitrary number of attributes that potentially increase parsing accuracy.

Natural languages are full of exceptional and colloquial expressions, and it is difficult for machine learning algorithms, as well as human linguists, to judge whether a specific rule is relevant in terms of overall performance. Because the maximal likelihood estimator (MLE) emphasizes the most frequent phenomena, an exceptional expression is placed in the same class as a frequent one. To tackle this problem, we investigate the mixture of sequentially generated decision trees. Specifically, we use the Adaboost algorithm [27] as in Chapter 4. The algorithm iteratively performs two procedures:

1. construct a decision tree based on the current data distribution

2. update the distribution by focusing on data that are not well explained by the constructed tree

The final modification probabilities are computed by mixing all of the decision trees according to their performance. The sequential decision trees gradually change from broad coverage to specific exceptional trees that cannot be captured by a general tree. In other words, the method incorporates not only general expressions but also infrequent specific ones.

The rest of this chapter is organized as follows. Section 5.1 summarizes the dependency analysis for the Japanese language. Section 5.2 introduces our feature setting for learning and then explains decision tree models that compute modification probabilities. Section 5.3 presents experimental results obtained by using the EDR Japanese annotated corpora. Section 5.4 relates our parser to other research from both natural language processing and machine learning viewpoints. Finally, Section 5.5 gives the summary.

5.1. Dependency Analysis in Japanese Language

This section overviews the dependency analysis in the Japanese language. The parser generally performs the following three steps.

1. Segment a sentence into a sequence of bunsetsu.
2. Prepare a modification matrix, each value of which represents how one bunsetsu is likely to modify the other.
3. Find optimal modifications in a sentence by a dynamic programming technique.

Because there are no explicit delimiters between words in Japanese, input sentences are first word segmented, part-of-speech tagged and then chunked into a sequence of segments (bunsetsu). Bunsetsu basically consists of a set of non-function words + function words, although its definition greatly depends on the user and usage. In our system, word segmentation and part-of-speech tagging are performed by a Japanese morphological analyzing program called Chasen [60]. Then the output from the tagger is passed to an automatic bunsetsu segmenter developed by Fujio [28]. The bunsetsu segmenter is implemented in Perl and contains 1,311 rules in the form of regular expressions to determine the bunsetsu boundaries. The first step yields, for the following example, the sequence of bunsetsu displayed below. The parenthesis in the Japanese expressions represent the internal structures of the bunsetsu (word segmentations).

Example: 昨日の夕方に近所の子どもがワインを飲んだ

((昨日)(の))	((夕方)(に))	((近所)(の))	((子ども)(が))	((ワイン)(を))
kinou-no	*yuugata-ni*	*kinjo-no*	*kodomo-ga*	*wain-wo*
yesterday-NO	*evening*-NI	*neighbor*-NO	*children*-GA	*wine*-WO
((飲ん)(だ)				
nomu+ta				
drink+PAST				

The second step of parsing is to construct a modification matrix whose values represent the likelihood that one bunsetsu modifies another in a sentence.

In the Japanese language, almost all practical parsers assume the following two constraints [83]. Although there are a small number of exceptions (i.e., inversion) for these two constraints, the majority of them can be avoided by devising corpus annotations and bunsetsu definitions. These two constraints with careful bunsetsu definitions can improve the parsing performance by cutting off excessive modification candidates.

1. Every bunsetsu except the last one modifies only one posterior bunsetsu.

2. No modification crosses to other modifications in a sentence.

Table 5.1 illustrates a modification matrix for the example sentence. In the matrix, columns and rows represent anterior and posterior bunsetsus, respectively. For example, the first bunsetsu *'kinou- no'* modifies the second *'yuugata-ni'* with score 0.70 and the third *'kinjo-no'* with score 0.07. The aim of this chapter is to generate a modification matrix by using decision trees.

The final step of parsing optimizes the entire dependency structure by using the values in the modification matrix. A chart parsing algorithm [44] is adopted to determine the optimal dependencies (best parse).

Before going into the details of the proposed method, we introduce here the notations that will be used in this chapter. Let S be the input sentence. S generally comprises a bunsetsu set B of length m ($\{<b_1, f_1>, \cdots, <b_m, f_m>\}$), in which b_i and f_i represent the ith bunsetsu and its features. We define D as a modification set; $D = \{mod(1), \cdots, mod(m-1)\}$, in which $mod(i)$ is the bunsetsu modified by the ith bunsetsu. Because of the first assumption above, the length

	kinou-no				
yuugata-ni	0.70	*yuugata-ni*			
kinjo-no	0.07	0.10	*kinjo-no*		
kodomo-ga	0.10	0.10	0.70	*kodomo-ga*	
wain-wo	0.10	0.10	0.20	0.05	*wain-wo*
nomu-ta	0.03	0.70	0.10	0.95	1.00

Table 5.1. Modification Matrix for Sample Sentence

of D is always $m - 1$. Using these notations, the result of the third step for the example sentence is $D_best = \{2, 6, 4, 6, 6\}$ as shown in Figure 5.1.

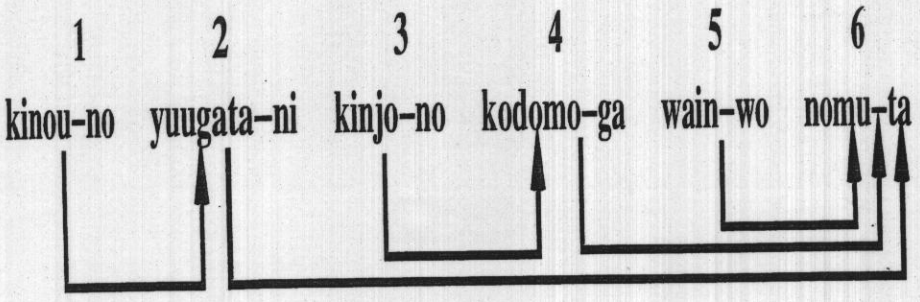


Figure 5.1. Modification Set for Sample Sentence

5.2. Decision Trees for Dependency Analysis

5.2.1 Linguistic Feature Types Used for Learning

This section explains the concrete feature set used for learning. Out of $f_1, \cdots, f_m$, we construct the feature set f_{ij} for the two bunsetsu b_i and b_j, which form each piece of the modification data. We use thirteen features for a f_{ij}, ten directly from the two bunsetsu under consideration and three for the other bunsetsu information summarized in Table 5.2.

Each bunsetsu (anterior and posterior) has the five features shown as No. 1 to No. 5 in Table 5.2. Features No. 6 to No. 8 are related to bunsetsu pairs. Both

No.	Two Bunsetsu	No.	Others
1	lexical information of headword	6	distance between two bunsetsu
2	part-of-speech of headword	7	particle ‘wa’ between two bunsetsu
3	type of bunsetsu	8	punctuation between two bunsetsu
4	punctuation		
5	parentheses		

Table 5.2. Linguistic Feature Types Used for Learning

Feature	Values
2	サ変名詞, 感動詞, 記号, 形式名詞, 形容詞, 固有名詞, 時制相副詞, 時相名詞, 人名, 数詞, 接続詞, 地名, 陳述副詞, 程度副詞, 動詞, 動詞性接尾辞, 発言副詞, 評価副詞, 頻度副詞, 普通名詞, 副詞, 副詞形態指示詞, 副詞的名詞, 名詞形態指示詞, 名詞性名詞助数辞, 名詞性名詞接尾辞, 名詞接続助詞, 名詞接頭辞, 様態副詞, 量副詞, 連体詞, 形態指示詞
3	きり, くらい, けど, けれど, けれども, こそ, こと, さ, さえ, し, しか, じゃ, すなわち, すら, て, で, でも, ぜ, そして, それに, ぞ, ため, だけ, だって, だの,っけ,ったら,って, つつ, と, とか, とも, ども, な, なあ, ない, ないし, ないしは, ながら, など, なら, ならびに, なり, なんか, なんて, に, ね, の, のみ, は, ばかり, へ, ほど, また, または, まで, も, もしくは, もの, ものの, や, やら, よ, よう, より, る, わ, を, サ変名詞, 意志, 括弧開, 感動詞, 基本, 記号, 及び, 共, 形式名詞, 形容詞, 形容詞性述語接尾辞, 固有名詞, 語幹, 時制相副詞, 時相名詞, 条件, 人名, 推量, 数詞, 接続詞, 地名, 中, 陳述副詞, 程度副詞, 動詞, 発言副詞, 評価副詞, 頻度副詞, 普通名詞, 副詞, 形態指示詞, 副詞的名詞, 並びに, 又は, 未然, 名詞形態指示詞, 名詞性述語接尾辞, 名詞性名詞助数辞, 名詞性名詞接尾辞, 命令, 様態副詞, 量副詞, 連体, 連体詞, 形態指示詞, 連用
4	no, 読点, 句点
5	no, ‘, (, “, 「, 『, 【, 〈, [, ’ , ” ,) , 〉, 」, 』, 】,]
6	A(0), B(1 ~ 4), C(≥5)
7	yes, no
8	yes, no

Table 5.3. Possible Values for Each Feature

No. 1 and No. 2 concern the headword of the bunsetsu. No. 1 takes values of frequent words or thesaurus categories [40]. No. 2, on the other hand, takes values of part-of-speech tags. No. 3 deals with bunsetsu types consisting of functional word chunks or part-of-speech tags that dominate the bunsetsu's syntactic characteristics. No. 4 and No. 5 are binary features and correspond to punctuation and parentheses, respectively. No. 6 represents how many bunsetsu exist between the two bunsetsu. Possible values in our setting are A (0), B ($0 \sim 4$) and C (≥ 5). No. 7 deals with the post-positional particle 'wa' which greatly influences the long-distance dependency of subject-verb modifications. Finally, No.8 addresses the punctuation between the two bunsetsu. The detailed values of each feature type are summarized in Table 5.3.

The data for the decision tree learning comprise any unordered combination of two bunsetsu in a sentence. The features used for learning are from the linguistic information shown in Table 5.2. Table 5.4 illustrates the sample data generated from the example sentence discussed in Section 5.1. t-noun and c-noun in the table represent the temporal noun and the common noun, respectively. Note also that *no* in italic and no in Roman mean the Japanese post positional particle '*no*' and a binary value 'no', respectively.

Each of the first 13 rows corresponds to the features shown in Table 5.2. The first 5 and the second 5 rows represent No. 1 to No. 5 features for both anterior and posterior bunsetsu. The remaining 3 features handle No. 6 to No. 8 features in this order. The last row takes the binary class values (yes and no) that delineate whether the data (the two bunsetsu) have a modification relation or not. In this setting, the decision tree algorithm automatically and consecutively selects the significant features for discriminating modify/non-modify (yes/no) relations.

5.2.2 Stochastic Model and Decision Trees

The stochastic dependency parser assigns the most plausible modification set D_{best} to a sentence in terms of the training data distribution as equation (5.1).

$$D_{best} = argmax_D P(D|S) = argmax_D P(D|B) \tag{5.1}$$

Although modifications in a sentence are in fact dependent on each other, as il-

Anterior					Posterior					Others			Class
1	2	3	4	5	1	2	3	4	5	6	7	8	
kinou	t-noun	*no*	no	no	yuugata	c-noun	*ni*	no	no	A	no	no	yes
kinou	t-noun	*no*	no	no	kinjo	c-noun	*no*	no	no	B	no	no	no
kinou	t-noun	*no*	no	no	kodomo	c-noun	*ga*	no	no	B	no	no	no
kinou	t-noun	*no*	no	no	wain	c-noun	*wo*	no	no	B	no	no	no
kinou	t-noun	*no*	no	no	nomu	verb	verb	no	no	B	no	no	no
yuugata	c-noun	*ni*	no	no	kinjo	c-noun	*no*	no	no	A	no	no	no
yuugata	c-noun	*ni*	no	no	kodomo	c-noun	*ga*	no	no	B	no	no	no
yuugata	c-noun	*ni*	no	no	wain	c-noun	*wo*	no	no	B	no	no	no
yuugata	c-noun	*ni*	no	no	nomu	verb	verb	no	no	B	no	no	yes
kinjo	c-noun	*no*	no	no	kodomo	c-noun	*ga*	no	no	A	no	no	yes
kinjo	c-noun	*no*	no	no	wain	c-noun	*wo*	no	no	B	no	no	no
kinjo	c-noun	*no*	no	no	nomu	verb	verb	no	no	B	no	no	no
kodomo	c-noun	*ga*	no	no	wain	c-noun	*wo*	no	no	A	no	no	no
kodomo	c-noun	*ga*	no	no	nomu	verb	verb	no	no	B	no	no	yes
wain	c-noun	*wo*	no	no	nomu	verb	verb	no	no	A	no	no	yes

Table 5.4. Training Data Generated from Example Sentence

lustrated by the two constraints in Section 5.1, it is generally difficult to efficiently determine what range of modification should be considered in the online computation of modification probability of two bunsetsus. In addition, the learning of such dependencies would require large amounts of training data. We thus adopt the usual independence assumption for efficient learning from a limited amount of training data. By assuming the independence of modifications, $P(D|B)$ can be transformed as equation (5.2). $P(yes|b_i, b_j, f_{ij})$ means the probability that a pair of bunsetsu b_i and b_j have a modification relationship given the features set f_{ij}. Note that f_{ij} contains the information of other bunsetsu despite the assumption of independence. In other words, features from other bunsetsus are also assumed to decide whether the two bunsetsus have a modification relationship. We use decision trees to dynamically select appropriate features for each combination of bunsetsus (Table 5.4).

$$P(D|B) = \prod_{i=1}^{m} P(yes|b_i, b_j, f_{ij}) \tag{5.2}$$

Let us first consider the single tree case. We slightly changed C4.5 [66] programs to be able to extract class frequencies at every node of the decision tree because our task is regression rather than classification. The following will explain how a decision tree is constructed in C4.5. The algorithm is given samples of feature-value vectors associated with their class labels. Table 5.4 shows the samples generated from the example sentence. The first task of the algorithm is to construct a decision tree that completely classifies these samples. The C4.5 program recursively splits the samples by selecting a feature that maximizes a heuristic *gain_ratio* criteria.

Let S and C_i be a set of samples and the ith (total number $= k$) class, respectively. $freq(C_i, S)$ represents the number of samples in S that are classified C_i. $|S|$ and m represent the number of samples in S and the number of splits. Consider a random sample from S classified C_j. The probability of the event is

$$\frac{freq(C_j, S)}{|S|}.$$

The information of the event can be computed as

$$-log_2 \frac{freq(C_j, S)}{|S|} \quad bits.$$

By taking the expectation over S,

$$info(S) = -\sum_{j=1}^{k} \frac{freq(C_j, S)}{|S|} \times log_2 \frac{freq(C_j, S)}{|S|} \quad bits$$

is the entropy of S.

For training samples T, $info(T)$ represents the average information to classify a sample in T. Similarly, we define $info_f(T)$ which is average information when we split T by using a feature f.

$$info_f(T) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} \times info(T_i)$$

The split of the data by using a feature f is useful only when the prediction of the classes becomes easier. In other words, the class distribution should be more concentrated. To test the usefulness of f, we can utilize $gain(f)$, which is the difference between $info(T)$ and $info_f(T)$.

$$gain(f) = info(T) - info_f(T)$$

Because $gain(f)$ tends to bias the feature with many splits, we introduce $gain\_ratio(f)$ by regularizing the $gain(f)$ over the following $split\_info(f)$.

$$split\_info(f) = -\sum_{i=1}^{m} \frac{|T_i|}{|T|} \times log_2 \frac{|T_i|}{|T|}$$

$$gain\_ratio(f) = \frac{gain(f)}{split\_info(f)}$$

Although the constructed decision tree can completely classify the training samples, it does not achieve very high performance against outside (unknown) samples. To avoid over-fitting training samples, the second task is pruning the constructed tree. The C4.5 program employs a pruning technique based on the statistical test in which the confidence level varies from 0% to 100%. The smaller the confidence level is, the more the decision tree is pruned.

Figure 5.2 illustrates the simplified version of an unpruned decision tree. The tree is generated from the training data shown in Table 5.4. Each node and edge in the tree is labeled with a feature name and its value, respectively. This means that the data are split by the feature according to its values. These features and values are selected in order to maximally separate class labels of the data. The finally separated class values are attached to each leaf node of the tree. In the current example, the decision tree first selects the feature Distance at the top node. If its value is A and the value of the next feature anterior type is *no*, the class of data is then determined as yes. The rest of the data can be correctly classified in the same way. Note here that we can extract the class frequencies from each node. For example, the circled node of depth 1 stores 5 examples of Table 5.4 whose distance values are A. The class frequency at this node is (yes, no) = (3, 2). We can thus compute at this node the probability of yes as 3/(3+2) = 0.6.

In the case of the pruned decision tree, we can also compute the probability $P_{DT}(yes|b_i, b_j, f_{ij})$, which is the Laplace estimate of the empirical likelihood that

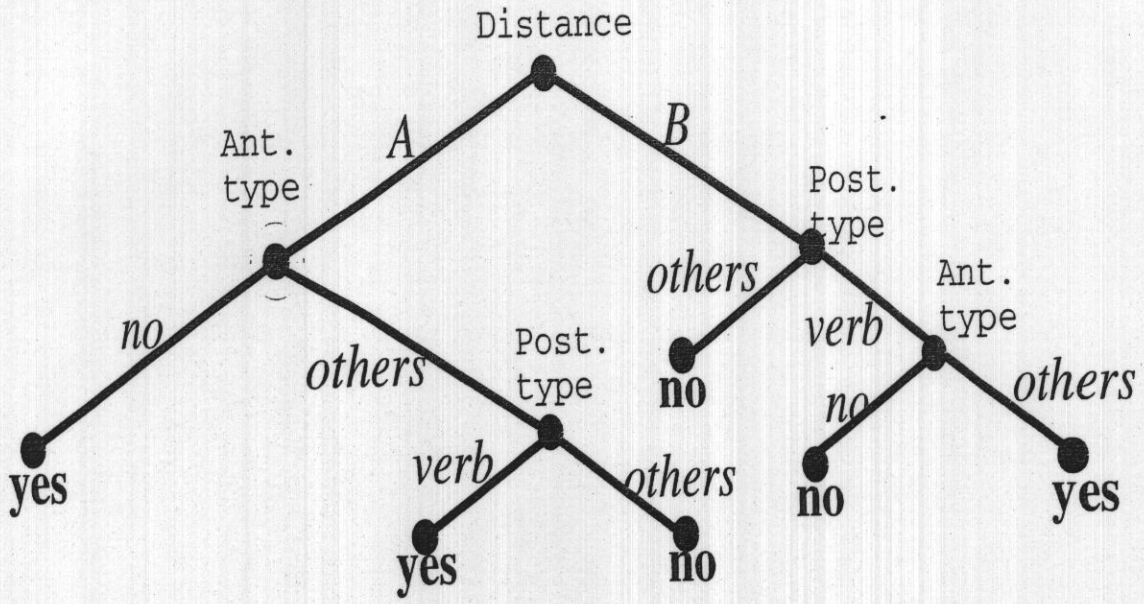


Figure 5.2. Simple Decision Tree Generated from Example Data

b_i modifies b_j at the leaf of the constructed decision tree DT. Note that it is necessary to normalize $P_{DT}(yes|b_i, b_j, f_{ij})$ to approximate $P(yes|b_i, b_j, f_{ij})$. By considering all candidates posterior to b_i, $P(yes|b_i, b_j, f_{ij})$ is computed using a heuristic rule (5.3). Such a normalization technique is also utilized by Collins [16]. It is of course reasonable to normalize class frequencies instead of the probability $P_{DT}(yes|b_i, b_j, f_{ij})$. Equation (5.3) tends to emphasize long-distance dependencies more than in the case of frequency-based normalization.

$$P(yes|b_i, b_j, f_1, \cdots, f_m) \simeq \frac{P_{DT}(yes|b_i, b_j, f_{ij})}{\sum_{k>i}^{m} P_{DT}(yes|b_i, b_j, f_{ij})} \tag{5.3}$$

Let us extend the above to use a set of decision trees. As briefly mentioned at the beginning of the chapter, a number of infrequent and exceptional expressions appear in any natural language phenomena; they deteriorate the overall performance of application systems. It is also difficult for automated learning systems to detect and handle these expressions because exceptional expressions are placed in the same class as frequent ones. To tackle this difficulty, we generate a set of decision trees by introducing the Adaboost [27] algorithm illustrated in

Table 5.5. The algorithm first sets the weights to 1 for all (training) examples (2 in Table 5.5) and repeats the following two procedures T times (3 in Table 5.5).

1. A decision tree is constructed by using the current weight vector ((a) in Table 5.5)
2. The same training data are then parsed by using the tree, and the weights of correctly handled data are reduced ((b) and (c) in Table 5.5)

```
1. Input: sequence of $N$ training examples $< e_1, w_1 >, \ldots, < e_N, w_N >$ in which
   $e_i$ and $w_i$ represent an example and its weight, respectively.
2. Initialize the weight vector $w_i = 1$ for $i = 1, \ldots, N$
3. Do for $t = 1, 2, \ldots, T$
   (a) Call C4.5 providing it with the weight vector $w_i$s and Construct a modification probability set $h_t$
   (b) Let Error be a set of training examples that are not identified by $h_t$
       Compute the pseudo error rate of $h_t$:
       $\epsilon_t = \sum_{i \subset Error} w_i / \sum_{i=1}^{N} w_i$
       if $\epsilon_t \geq \frac{1}{2}$, then abort loop
       $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
   (c) For training examples correctly predicted by $h_t$, update the weights vector
       to be $w_i = w_i \beta_t$
4. Output a final probability set:
   $$h_f = \sum_{t=1}^{T} (log \frac{1}{\beta_t}) h_t / \sum_{t=1}^{T} (log \frac{1}{\beta_t})$$
```

Table 5.5. Combining Decision Trees by Adaboost Algorithm

The final probability set h_f is then computed by mixing T trees according to their performance (4 in Table 5.5). Using h_f instead of $P_{DT}(yes|b_i, b_j, f_{ij})$ in equation (5.3) generates a boosted version of the dependency parser. The sequential decision trees gradually change from broad coverage to specific exceptional trees that cannot be captured by a general tree. In other words, the method incorporates not only general expressions but also infrequent specific ones.

5.3. Experimental Results

The proposed parser was evaluated by using the EDR Japanese annotated corpus [23]. The experiment consisted of two parts. One evaluated the single-tree parser and the other the boosted counterpart. In the rest of this section, parsing accuracy refers only to precision, that is, how many of the system's outputs are correct in terms of the annotated corpus. We do not show recall because we assume every bunsetsu modifies only one posterior bunsetsu. The features used for learning were non-headword features, (i.e., types 2 to 8 in Table 5.2). Section 5.9 investigates the lexical information of headwords such as frequent words and thesaurus categories. Before going into the details of the experimental results, we summarize here how training and test data were constructed.

1. After all sentences in the EDR corpus were word-segmented and part-of-speech tagged, they were then chunked into a sequence of bunsetsu.

2. All bunsetsu pairs were compared with EDR bracketing annotation (bunsetsu segmentations and modifications). If a sentence contained a bunsetsu pair inconsistent with the EDR annotation, the sentence was removed from the data.

3. All data examined (total number of sentences: 207,802, total number of bunsetsu: 1,790,920) were divided into 20 files. Training and test data were completely differentiated. The training data was 50,000 sentences, which were the first 2,500 sentences of the 20 files. Test data (10,000 sentences) were the 2,501th to 3,000th sentences of each file.

Because there are no explicit delimiters between words in Japanese, the primary task is to word segment and part-of-speech tag the input sentences. The part-of-speech information contained in the EDR corpus is too coarse to be used in parsing. We thus employed only bracketing information from the corpus. For the POS tagging, we used a Japanese morphological analyzing program called Chasen [60] to make use of the well-grained tag information to address the syntactic ambiguities. After part-of-speech tagging, a sequence of bunsetsus was automatically generated by utilizing the automatic bunsetsu segmenter developed by Fujio [28]. The bunsetsu segmenter is implemented in Perl and contains 1311 segmentation rules.

The second step detects the modification relationships existing in the EDR corpus. Due to the difference between our definition of bunsetsu and that used in the EDR corpus, some discrepancies of bunsetsu boundary occurred. We removed from the data those sentences that contained these discrepancies. Note that the removal has no effect on the use of the resulting parser because all bunsetsu segmentations in use are performed according to our bunsetsu definition.

The third step is to construct completely differentiated training and test data. For both data, we took the same number of sentences from 20 different files in order to avoid imbalances in the genres of texts.

In the single-tree experiments, we evaluated the following four properties of the new dependency parser.

- Tree pruning and parsing accuracy
- Number of training sentences and parsing accuracy
- Significance of features other than Headword Lexical Information
- Significance of Headword Lexical Information

5.3.1 Pruning and Parsing Accuracy

Table 5.6 summarizes the parsing accuracy with various confidence levels of pruning. The number of training sentences was 10,000.

In C4.5 programs, a larger value of confidence means weaker pruning; 25% is commonly used in various domains [66]. Our experimental results show that 75% pruning attains the best performance, i.e. weaker pruning than usual. In the remaining single-tree experiments, we used the 75% confidence level. Although strong pruning handles infrequent data as noise, parsing involves many

Confidence Level	25%	50%	75%	95%
Parsing Accuracy	82.01%	83.43%	83.52%	83.35%

Table 5.6. Pruning Confidence Level vs.Parsing Accuracy

exceptional and infrequent modifications as mentioned before. Our results indicate that information included in only a small number of samples is useful for disambiguating the syntactic structure of sentences.

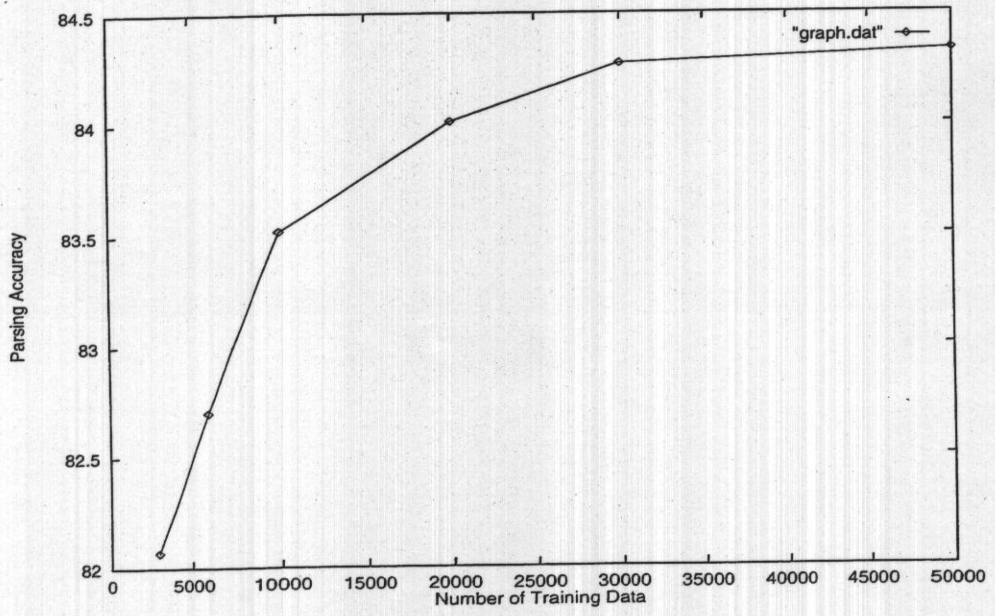


Figure 5.3. Learning Curve of Single Decision Tree

5.3.2 Amount of Training Data and Parsing Accuracy

Figure 5.3 and Table 5.7 show how the number of training sentences influences the parsing accuracy for the same 10,000 test sentences. They illustrate the following two characteristics of the learning result.

1. Parsing accuracy rapidly rises up to 30,000 sentences and is almost flat at around 50,000 sentences.

2. The maximum parsing accuracy is 84.33% at 50,000 training sentences.

Number of Training Sentences	3000	6000	10000	20000	30000	50000
Parsing Accuracy	82.07%	82.70%	83.52%	84.07%	84.27%	84.33%

Table 5.7. Number of Training Sentences vs. Parsing Accuracy

We will discuss the maximum accuracy of 84.33%. Compared to recent stochastic English parsers that yield 86 to 87% accuracy [16, 56], our result is not so impressive at a glance. The main reason lies in the difference between the two corpora used: Penn Treebank [58] and the EDR corpus [23]. Penn Treebank is also used to induce part-of-speech (POS) taggers because the corpus contains very detailed POS information as well as bracket annotations. In addition, English parsers incorporate the syntactic tags that are contained in the corpus. The EDR corpus, on the other hand, contains only coarse POS tags. We had to utilize another Japanese POS tagger [60] to make use of well-grained information for disambiguating syntactic structures. Only the bracket information in the EDR corpus was considered. We conjecture that the difference between the parsing accuracies is due to the difference of the corpus information available. Other research seems to support this conjecture. [28] constructed another EDR-based dependency parser by using a similar method to Collins' [16]. The performance of the parser was 80.48 % precision with the same evaluation method as ours. Furthermore, the features they used were exactly the same as ours although possible values of each feature were slightly different. They also employed the same part-of-speech tagger and bunsetsu segmenter. Although we do not exactly know what portion of the EDR corpus was used as their 200,000 training and 10,000 test sentences, a comparison of the two parsers would give some information on the characteristics of the EDR corpus because of the great similarity in our settings. The two parsers difference in performance probably arises mainly from the feature selection capability of the decision tree algorithm.

5.3.3 Significance of Non-headword Features

We will now summarize the significance of each non-headword feature introduced in Section 5.2. The influence of the lexical information of headword will be discussed in the next section. Table 5.8 illustrates how the parsing accuracy was reduced when each feature was removed. The number of training sentences was 10,000. In the table, ant and post represent the anterior and the posterior bunsetsu, respectively.

Feature	**Accuracy Decrease**
ant POS of head	-0.07%
ant bunsetsu type	+9.34%
ant punctuation	+1.15%
ant parentheses	±0.00%
post POS of head	+2.13%
post bunsetsu type	+0.52%
post punctuation	+1.62%
post parentheses	±0.00%
distance between two bunsetsu	+5.21%
punctuation between two bunsetsu	+0.01%
'wa' between two bunsetsu	+1.79%

Table 5.8. Decrease of Parsing Accuracy When Each Attribute Removed

Table 5.8 clearly demonstrates that the most significant features are anterior bunsetsu type and distance between the two bunsetsu. This result may partially support an often used heuristic requiring that bunsetsu modification distance be as short-range as possible, provided the modification is syntactically possible. Thus, we need to concentrate on the types of bunsetsu to attain a higher level of accuracy. Most features contribute, to some extent, to parsing performance. In our experiment, information on parentheses has no effect on performance. The reason may be that EDR contains only a small number of parentheses. One exception in our features is anterior POS of head. We currently hypothesize that this drop in accuracy is due to the following two reasons.

- In many cases, the POS of a headword can be determined from the bunsetsu type.
- Our POS tagger sometimes assigns verbs for verb-derived nouns.

5.3.4 Significance of Headword Lexical Information

We focused on the headword feature by testing the following four lexical sources with the 10,000 training sentences. The first and the second are the 100 and 200 most frequently occurring words. The third and the fourth are derived from a broadly used Japanese thesaurus, Word List by Semantic Principles [40], in which Level 1 and Level 2 classify words into 15 and 67 categories, respectively.

1. 100 most Frequent words
2. 200 most Frequent words
3. Word List Level 1
4. Word List Level 2

Headword Information Added	100 words	200 words	Level 1	Level 2
Parsing Accuracy	83.34%	82.68%	82.51%	81.67%

Table 5.9. Headword Information vs. Parsing Accuracy

Table 5.9 displays the parsing accuracy when each of the above four types of headword lexical information was used in addition to the previous features. In all cases, the performance was worse than 83.52%, which was the level attained without them. More surprisingly, more headword information yielded worse performance. Why does lexical information not improve the parsing accuracy even though lexical information is reported to be very effective for parsing European languages? In the Japanese language, the functional words such as post positional particles offer stronger clues for dependency structures than is true in European languages. Note that these most influential word forms are utilized in another feature, type of bunsetsu (3 in Table 5.2). Other lexical information may have to be more elaborate to further improve the parsing accuracy of Japanese texts. It may be helpful to incorporate the more structured lexical information (case frame information) as in [17]. We will briefly discuss problems involved in word statistics and thesaurus information we used in turn.

For frequently occurring words, there remains the possibility that the performance was worse because we considered directly a limited number (100 and 200) of words. Other research [28] has reported that considering all content words with the same smoothing technique as Collins slightly improves the parsing accuracy.

The preprocessing of the head word co-occurrence might be helpful even for our decision tree learning method. For example, the feature used for learning may be a quantized co-occurrence probability between two head words. Anyway, these two results indicate that word statistics does not have as strong an impact in parsing the EDR corpus as it does in European language corpora. The reason for this difference may be that the EDR corpus comprises more diverse genres of texts than Penn Treebank (Wall Street Journal articles). We need further research to test this possibility.

The result from Word List by Semantic Principles involves more sensitive and complicated problems. First, the categories of the thesaurus may be too coarse to be used in parsing. Second, the entries of the thesaurus contain only 30,000 words. These two shortcomings of the thesaurus may deteriorate the overall performance. Thus, further investigation of other thesaurus [39] and clustering [10] techniques might improve the parsing performance.

In summary, it may now be safely said, at least for the Japanese corpus, that we cannot expect lexical information of content words to consistently make significant improvement in performance.

5.3.5 Boosting Experiments

This section reports experimental results on the boosted version of our parser. In all experiments, pruning confidence levels were set to 55%. Table 5.10 and

Figure 5.4 show the parsing accuracy when the number of training examples was increased. Because the number of iterations in each data set changed between 5 and 8, we will show the accuracy by combining the first 5 decision trees. In Figure 5.4, the dotted line plots the learning of the single-tree case (identical to Figure 5.3) for the reader's reference. The characteristics of the boosted version can be summarized as follows in comparison with the single-tree version.

- The learning curve rises more rapidly with a small number of examples. It is surprising that the boosted version with 10,000 sentences performs better than the single-tree version with 50,000 sentences.
- The boosted version significantly outperforms the single-tree counterpart for any number of sentences, although they use the same features for learning.

Number of Training Sentences	3000	6000	10000	20000	30000	50000
Parsing Accuracy	83.10%	84.03%	84.44%	84.74%	84.91%	85.03%

Table 5.10. Number of Training Sentences vs. Parsing Accuracy

Next, we discuss how the number of iterations influences the parsing accuracy. Table 5.11 shows the parsing accuracy for various iteration numbers when 50,000 sentences were used as training data. The results have two characteristics.

- Parsing accuracy rose rapidly at the second iteration.
- No over-fitting of data was observed. The performance of each generated tree fell to around 30% at the final stage of iteration, showing that the trees become more specialized.

5.4. Related Work

We first relate our work to other research in the parsing community. Recent sophisticated English parsers [56, 10] replaced PCFG with transformation (extension and labeling) rules that are augmented with several attributes, in particular

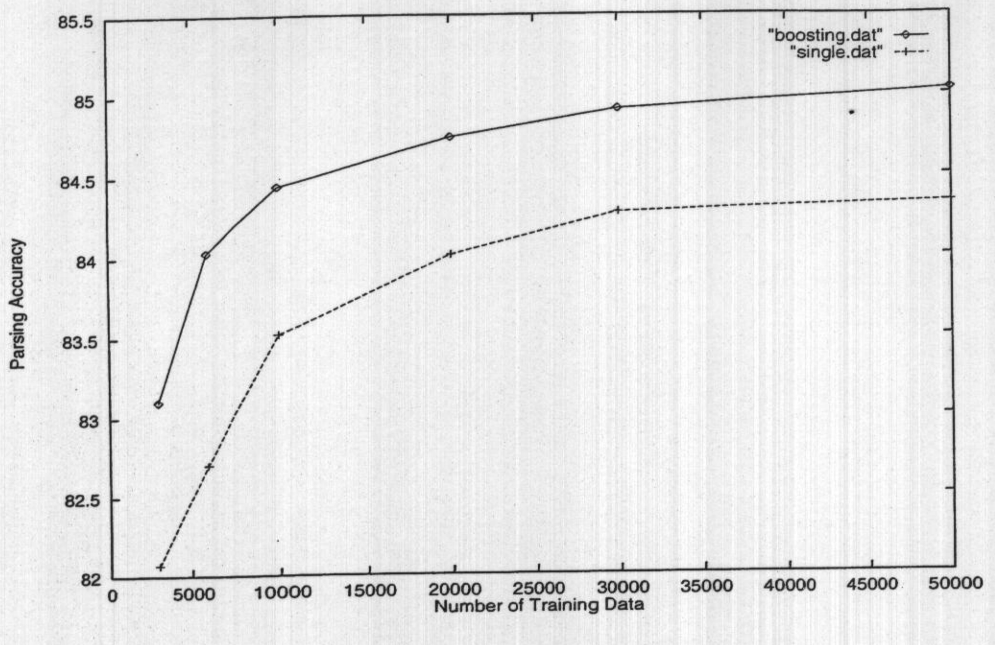


Figure 5.4. Learning Curve of Boosting Parser

Number of Iteration	1	2	3	4	5	6
Parsing Accuracy	84.32%	84.93%	84.89%	84.86%	85.03%	85.01%

Table 5.11. Number of Iteration vs. Parsing Accuracy

lexical information. However, we adopt dependency analysis because word order in Japanese, particularly bunsetsu order, is relatively unrestricted compared to European languages. Our approach is, in spirit, similar to Collins' dependency-based English parser. The main difference lies in the feature selection process. A Collins-type parser [16, 28] predefines a set of features and conditions the modification probabilities by all of the attributes, regardless of the bunsetsu type. The method can incorporate only a small number of features to precisely evaluate parameters from a finite amount of data. Our decision tree method offers more sophisticated feature selection to automatically detect the most significant features according to the bunsetsu type. Thus, we can use an arbitrary number of features that may contribute to parsing accuracy. These potential features for learning have been enumerated in many papers on conventional hand-compiled parsers. For example, [51, 42] consider many features (types of characters, similarity of bunsetsu sequence, etc.). Collins incorporates subcategorization frames in his extended generative model [17] and reports 2.3% improvement of performance over his dependency model [16]. Such structured lexical information should be useful in our decision tree based parsers.

There are also some interesting points from the machine learning viewpoint. Various kinds of *re-sampling* techniques have been studied in statistics [24, 25] and machine learning [2, 37, 27]. The Adaboost method was designed to construct a high-performance predictor by iteratively calling a weak learning algorithm (which is slightly better than random guessing). An empirical work reports that the method greatly improved the performance of decision-tree, k-nearest-neighbor, and other learning methods [26, 22]. In the context of natural language processing, Chapter 4 reports that the Adaboost improves the performance of a Japanese part-of-speech tagger that uses hierarchical context trees [68, 34]. However, most of these experiments were done with a small number of examples, and no asymptotic results were reported. We tested the proposed algorithm with one million pieces of training data (50,000 sentences) in a real-world parsing task. Our results confirm the usefulness of the Adaboost algorithm in both the rapidity of learning curve improvement and the improved performance after sufficient data are given to the parser.

5.5. Summary

This chapter has described a new Japanese dependency parser that uses decision trees. First, we introduced a single-tree parser to clarify the basic characteristics of our parser. The experimental results show that it achieves an accuracy of 84% outperforming conventional stochastic parsers. Next, the boosted version of our parser was introduced. The promising results of the boosted parser can be summarized as follows.

- An accuracy of 85%, outperforming the single-tree counterpart regardless of the amount of training data.

- No data over-fitting was observed when the number of iterations changed.

This research should be continued in two directions. One is to make the parser available to a broad range of researchers and to use their feedback to revise the features for learning. Second, the method should be tested with other languages such as English. Although we have focused on the Japanese language, our parser could be easily modified to work with other languages.

Chapter 6

AIDA: An Adaptive and Integrated Dictionary Agent

6.1. Need for Integrating Various Language Resources

This chapter reports an adaptive dictionary environment called AIDA (Adaptive and Integrated Dictionary Agent) as a concrete application of corpus-based natural language processing techniques. The following competitive requirements for foreign language dictionaries motivated the development of the system.

brevity and context dependence Although brief descriptions for an entry are desirable, they should contain explanations and examples that indicate the correct usage of words.

generality and specificity of word usage There are several kinds of words from domain-specific words such as technical terms to general words such as frequently used verbs. The former mainly requires word-to-word translation, while the latter also demands subtle nuance and cultural background.

A single dictionary cannot satisfy these requirements because a larger dictionary does not always subsume smaller ones. Users have to use several dictionaries according to a given purpose [77]. Even if users try to utilize different dictionaries simultaneously, there are the following drawbacks.

- Each dictionary is a separate piece of work. It is not assumed by developers that dictionaries are mutually linked for the flexible use.

- Dictionaries are static; they do not offer adaptation to new domains or personal customization for users.

To address these issues, AIDA is designed to be an adaptive dictionary environment that can be changed on-line according to a user's preferences. By using corpus-based language processing techniques, this system provides us with a cross reference of various dictionaries and corpora in a single graphical user interface. More specifically, AIDA offers the following functions to users.

1. **Cross reference for dictionaries and corpora** Users can access any language source in their favorite direction, say from a Japanese-English dictionary to an English monolingual corpus or from a bilingual corpus to an English-English dictionary. This cross referencing is realized by sharing a common index structure among all language sources.

2. **Flexible expression retrieval from corpora**
AIDA outputs corpus sentences in the order of similarity with an input expression. Similarity is computed as the sum of active feature weights. The features are syntactic and semantic characteristics of a sequence of words. They are extracted from input by morphological analysis. The weights for each feature can be learned through interactions between the system and users.

3. **Automatic extraction of bilingual and monolingual collocations**
The statistical methods reported in Chapter 3 can extract bilingual and monolingual collocations from corpora. These collocations are linked with other language resources in AIDA. They are very effective for adapting AIDA to the application domain of each user.

4. **Integrated user interface implemented on Netscape**
The functions summarized above can be accessed through a single graphical interface implemented on Netscape. The interface enables users to share and access various language resources via the Internet.

The organization of the chapter is as follows. Section 6.2 describes the implemented functions of AIDA. Section 6.3 discusses the flexible retrieval of corpora. The retrieval algorithm, the features used and the learning mechanism of feature weights are explained in this order. Finally, Section 6.4 relates AIDA with other research, and Section 6.5 summarizes the chapter.

6.2. Overview of AIDA

The input to the system is any kind of dictionary annotated with SGML, including sentence aligned bilingual corpora and monolingual corpora. Table 6.1 illustrates the language resources we implemented in AIDA[1]. In the initialization step, the input data is part-of-speech tagged [60, 4] and then word-indexed by using Patricia trees [32]. The collocation extraction is then performed for both the bilingual and monolingual corpora.

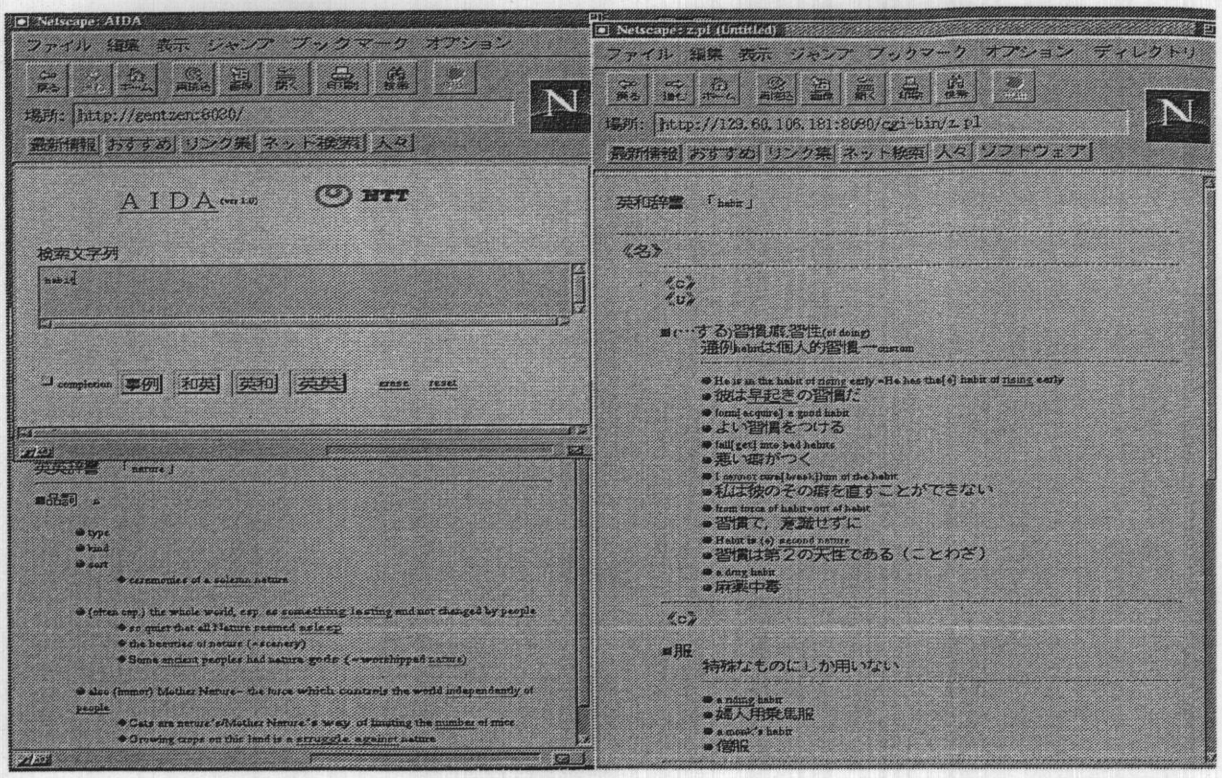
Data Type	**Name (Volume)**
Dictionary	Anchor Eng.-Japanese, Anchor Japanese-Eng., Concise Eng.-Eng.
Bilingual Corpus	Nikkei Science (65000 pair), Yomiuri Editorial (7000 pair)
English Corpus	Wall Street Journal (1987-1989)

Table 6.1. Data Incorporated in AIDA

Figure 6.1 is a screenshot image of a user looking up the English word '*habit*' in the English-Japanese dictionary. The underlined words in the image signify that they have entries in other dictionaries. The left window displays the entry for '*nature*' of the English-English dictionary accessed from the right window.

Figure 6.2 shows the screenshot image when a user consults the bilingual collocation dictionary linked from the Nikkei Science. The right window shows the list of collocations extracted from the data. The left window displays example sentences including the collocation 'アルツハイマー病患者 -*Alzheimer patients*'.

[1]We would like to thank Nikkei Science Publishing Co. and Yomiuri Shinbun Co. for permitting us to use the data.


Netscape: AIDA
ファイル 編集 表示 ジャンプ ブックマーク オプション
戻る 進む ホーム 再読込 画像 開く 印刷 検索
最新情報 おすすめ リンク集 ネット検索 人々
Netscape: z.pl (Untitled)
ファイル 編集 表示 ジャンプ ブックマーク オプション ディレクトリ
最新情報 おすすめ リンク集 ネット検索 人々 ソフトウェア

場所: http://gentzen:8020/

AIDA (ver 1.0) NTT

検索文字列

habit

completion 事例 和英 英和 英英 erase reset

英英辞書 「nature」

■品詞 n

● type
● kind
● sort
◆ ceremonies of a solemn nature

● (often cap.) the whole world, esp. as something lasting and not changed by people
◆ so quiet that all Nature seemed asleep
◆ the beauties of nature (=scenery)
◆ Some ancient peoples had nature gods (=worshipped nature)

● also (humor) Mother Nature– the force which controls the world independently of people
◆ Cats are nature's/Mother Nature's way of limiting the number of mice
◆ Growing crops on this land is a struggle against nature

場所: http://129.60.106.181:8080/cgi-bin/z.pl

英和辞書 「habit」

《名》

《C》
《U》

■(…する)習慣,癖,習性(of doing)
通例habitは個人的習慣→custom

● He is in the habit of rising early =He has the[a] habit of rising early
● 彼は早起きの習慣だ
● form[acquire] a good habit
● よい習慣をつける
● fall[get] into bad habits
● 悪い癖がつく
● I cannot cure[break]him of the habit
● 私は彼のその癖を直すことができない
● from force of habit=out of habit
● 習慣で, 意識せずに
● Habit is (a) second nature
● 習慣は第2の天性である（ことわざ）
● a drug habit
● 麻薬中毒

《C》

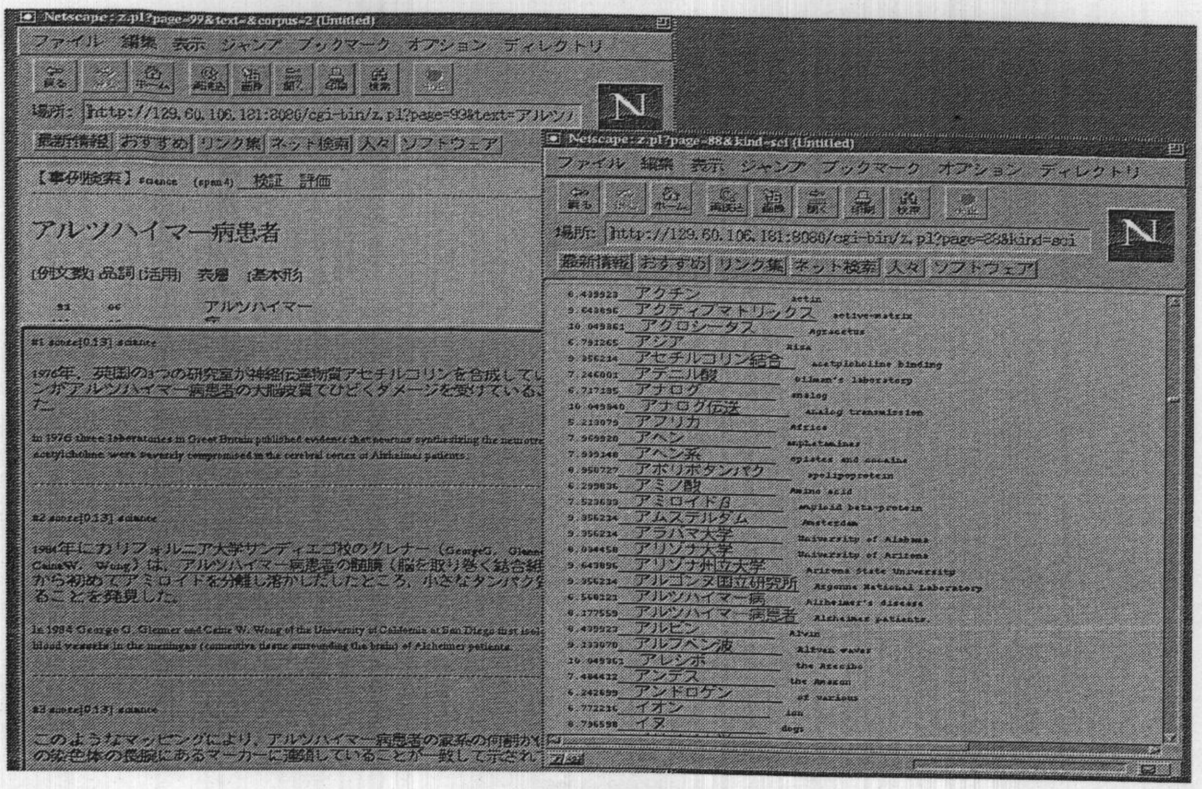
■服
特殊なものにしか用いない

● a riding habit
● 婦人用乗馬服
● a monk's habit
● 僧服

Figure 6.1. Looking Up Dictionaries

Let us move on to the flexible retrieval of corpora. When a user inputs an expression, the AIDA system outputs similar expressions by using a dynamic programming technique. The user then interacts with the system and evaluates the outputs. When an appropriate expression is provided with low similarity or an inappropriate expression with high similarity, the system can be supplied error feedback from the user. The system adapts to the user's preferences by changing the feature weights according to the feedback. Figure 6.3 is a screenshot image when a user inputs the Japanese expression '用地の取得を急ぐ必要がある' to retrieve bilingual sentences. The left window displays the resulting sentence pairs. The right window is used to evaluate (feedback) the output.

Although we have so far discussed the dictionaries, the collocations and the retrieval of expressions separately, the actual use of the AIDA system is complex and integrates processes according to text domains and the user's ability in


Netscape: z.pl?page=99&text=&corpus=2 (Untitled)
ファイル 編集 表示 ジャンプ ブックマーク オプション ディレクトリ
場所: http://129.60.106.181:8080/cgi-bin/z.pl?page=93&text=アルツ/
最新情報 おすすめ リンク集 ネット検索 人々 ソフトウェア
Netscape: z.pl?page=88&kind=sci (Untitled)
ファイル 編集 表示 ジャンプ ブックマーク オプション ディレクトリ
場所: http://129.60.106.181:8080/cgi-bin/z.pl?page=88&kind=sci
最新情報 おすすめ リンク集 ネット検索 人々 ソフトウェア

【事例検索】science (span4) 検証 評価

アルツハイマー病患者

[例文数] 品詞 [活用] 表層 [基本形]

51 06 アルツハイマー

#1 score[0.13] science

1976年、英国の3つの研究室が神経伝達物質アセチルコリンを合成してい ンがアルツハイマー病患者の大脳皮質でひどくダメージを受けている た。

In 1976 three laboratories in Great Britain published evidence that neurons synthesizing the neurotr acetylcholine were severely compromised in the cerebral cortex of Alzheimer patients.

#2 score[0.13] science

1984年にカリフォルニア大学サンディエゴ校のグレナー (GeorgeG. Glenn CaineW. Wong) は、アルツハイマー病患者の髄膜 (脳を取り巻く結合組 から初めてアミロイドを分離し溶かしだしたところ、小さなタンパク ることを発見した。

In 1984 George G. Glenner and Caine W. Wong of the University of California at San Diego first isol blood vessels in the meninges (connective tissue surrounding the brain) of Alzheimer patients.

#3 score[0.13] science

このようなマッピングにより、アルツハイマー病患者の家系の何割かは の染色体の長腕にあるマーカーに連鎖していることが一致して示され

6.439923	アクチン	actin
9.643896	アクティブマトリックス	active-matrix
10.049861	アクロシータス	Agracetus
6.791265	アジア	Asia
9.356214	アセチルコリン結合	acetylcholine binding
7.246001	アデニル酸	Gilman's laboratory
6.717135	アナログ	analog
10.049840	アナログ伝送	analog transmission
5.213079	アフリカ	Africa
7.969920	アヘン	amphetamines
7.939148	アヘン系	opiates and cocaine
8.950727	アポリポタンパク	apolipoprotein
6.299836	アミノ酸	Amino acid
7.523639	アミロイドβ	amyloid beta-protein
9.356214	アムステルダム	Amsterdam
9.356214	アラバマ大学	University of Alabama
8.034458	アリゾナ大学	University of Arizona
9.643896	アリゾナ州立大学	Arizona State University
9.356214	アルゴンヌ国立研究所	Argonne National Laboratory
6.560121	アルツハイマー病	Alzheimer's disease
8.177559	アルツハイマー病患者	Alzheimer patients.
8.439923	アルビン	Alvin
9.133070	アルフベン波	Alfven waves
10.049361	アレシボ	the Arecibo
7.484432	アンデス	the Amazon
6.242699	アンドロゲン	of various
6.772216	イオン	ion
8.796598	イヌ	dogs

Figure 6.2. Looking Up Bilingual Collocations

English.

6.3. Retrieving Corpora

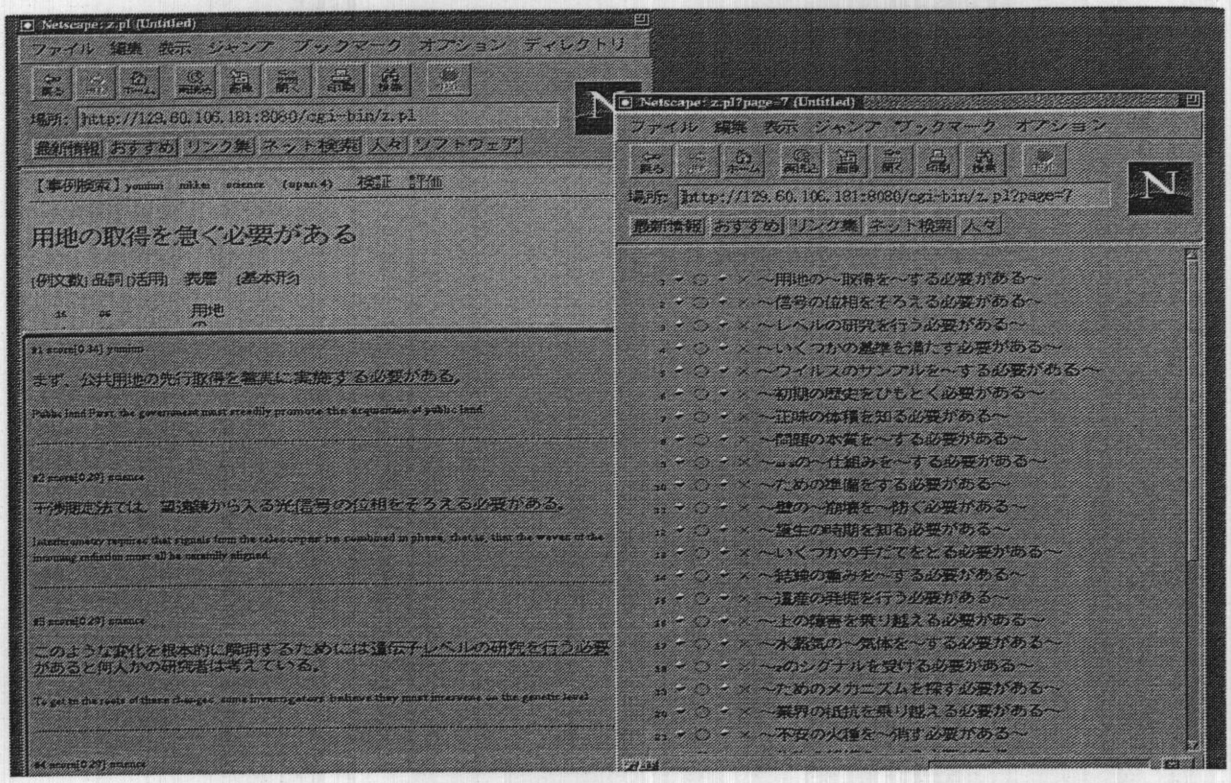
6.3.1 DP Search Algorithm at Word-POS level

In retrieving corpora, the following three stages of preprocessing are done for the raw corpus.

1. Nothing is done; the character level search is performed.

2. Only morphological analysis is done; the word and local structure level search is performed.

3. Dependency analysis is done; the global structure level search is performed.


Netscape: z.pl (Untitled)
ファイル 編集 表示 ジャンプ ブックマーク オプション ディレクトリ
場所: http://129.60.106.181:8080/cgi-bin/z.pl
最新情報 おすすめ リンク集 ネット検索 人々 ソフトウェア
Netscape: z.pl?page=7 (Untitled)
ファイル 編集 表示 ジャンプ ブックマーク オプション
場所: http://129.60.106.181:8080/cgi-bin/z.pl?page=7
最新情報 おすすめ リンク集 ネット検索 人々

【事例検索】yomiuri nikkei science (span 4) 検証 評価

用地の取得を急ぐ必要がある

[例文数] 品詞 [活用] 表層 [基本形]

#1 score[0.34] yomiuri

まず、公共用地の先行取得を着実に実施する必要がある。

Public land First, the government must steadily promote the acquisition of public land.

#2 score[0.29] science

干渉測定法では、望遠鏡から入る光信号の位相をそろえる必要がある。

Interferometry requires that signals from the telescopes be combined in phase, that is, that the waves of the incoming radiation must all be carefully aligned.

#3 score[0.29] science

このような変化を根本的に解明するためには遺伝子レベルの研究を行う必要があると何人かの研究者は考えている。

To get to the roots of these changes, some investigators believe they must intervene on the genetic level.

#4 score[0.29] science

1 ○ × ~用地の~取得を~する必要がある~
2 ○ × ~信号の位相をそろえる必要がある~
3 ○ × ~レベルの研究を行う必要がある~
4 ○ × ~いくつかの基準を満たす必要がある~
5 ○ × ~ウイルスのサンプルを~する必要がある~
6 ○ × ~初期の歴史をひもとく必要がある~
7 ○ × ~正味の体積を知る必要がある~
8 ○ × ~問題の本質を~する必要がある~
9 ○ × ~[illegible]の~仕組みを~する必要がある~
10 ○ × ~ための準備をする必要がある~
11 ○ × ~壁の~崩壊を~防ぐ必要がある~
12 ○ × ~誕生の時期を知る必要がある~
13 ○ × ~いくつかの手だてをとる必要がある~
14 ○ × ~結論の重みを~する必要がある~
15 ○ × ~遺産の発掘を行う必要がある~
16 ○ × ~上の障害を乗り越える必要がある~
17 ○ × ~水蒸気の~気体を~する必要がある~
18 ○ × ~[illegible]のシグナルを受ける必要がある~
19 ○ × ~ためのメカニズムを探す必要がある~
20 ○ × ~業界の抵抗を乗り越える必要がある~
21 ○ × ~不安の火種を~消す必要がある~

Figure 6.3. Flexible Expression Retrieval and Evaluation Image

Because kanji characters carry semantic information, the character-level search in Japanese has an advantage in utilizing the limited semantics. However, the method cannot capture the syntactic structures of a sequence of words. The global level search with parsing, on the other hand, incorporates the long-distance dependency in a sequence. It depends on the user's demand whether the long-distance dependency is needed or not. Because our main target is phrase search within middle-size bilingual corpora, the current version of AIDA uses a local structure level search. The following summarizes the main reasons. Of course, a more global search (say, example-based translation) requires long-distance dependencies. An important future direction is to incorporate the parsing technique introduced in Chapter 5.

- Most user demands are for short expressions that are well handled by morphological analysis alone.

- A long-distance dependency is effective only when the volume of available data is large.
- The accuracy of parsing techniques is not as stable as morphological analysis.

Let $sim(m,n)$ be a similarity measure between a morphologically analyzed input $s(s_1, s_2, \cdots, s_m)$ and a corpus sentence $t(\ t_1, t_2, \cdots, t_n)$. The computation of $sim(m,n)$ proceeds as follows by a dynamic programming method. In the equation, $m(i,j)$ represents the contribution of words s_i and t_j to the total similarity $sim(m,n)$. $m(i,j)$ is computed by summing the weight of active features. The actual features used will be explained in the next section. Dynamic programming (search width=4) with the morphological analysis can efficiently consider the local phrase structures as well as the word level concordances.

$$sim(i,j) = \begin{cases} 0 & if(i=0 \vee j=0) \\ max & \begin{pmatrix} sim(i-1,j-1)+m(i,j) \\ sim(i-1,j) \\ sim(i,j-1) \end{pmatrix} \end{cases}$$

6.3.2 Features Used for Similarity Estimation

Table 6.2 illustrates the features for Japanese that produce the similarity $m(i,j)$ between two words s_i and t_j. The right column shows the number of parameters. $m(i,j)$ is computed by summing the weights of active features that are satisfied by s_i and t_j. Figure 6.3 clearly indicates that these simple features well capture semantic and syntactic characteristics of the input sequence. A future direction is to consider other features such as the categories of a thesaurus and the character level information of kanji.

6.3.3 On-line Learning of Feature Weights

This section describes how the weight learning is performed. Because a user's consultation and feedback occur interactively, we have to employ an on-line learning algorithm rather than a batch algorithm in which induction is performed by using

Type of feature	**Number**
Word surface	1
Part-of-speech	POS symbols
Conjugation	conjugation type
Word surface of s_{i-1} and t_{j-1}	POS symbols^2
Last word surface	1
Last part-of-speech	POS symbols
First word surface	1
First part-of-speech	POS symbols

Table 6.2. Features Used

the stored samples at one time. We introduce a simple on-line algorithm called WINNOW [53, 54]. WINNOW learns weights of linear separable models. Theoretical analysis shows that the algorithm is very tolerant to irrelevant features. More specifically, the upper bound of the number of mistakes is in the linear order of relevant features and in the logarithmic order of total number of features. This is the main reason WINNOW is used in applications with large dimensionality such as the text categorization task. In our setting, WINNOW updates the feature weights every time AIDA outputs an inappropriate result labeled by the user feedback through the graphical interface in Figure 6.3.

Let θ, α (>1) be a threshold and a learning parameter. WINNOW performs the following procedures in Table 6.3 to obtain the feature weights. WINNOW enables AIDA to adapt to cope with a user's preferences by updating feature weights.

1. Initialize weight vectors $w = (w_1, w_2, ..., w_n)$.
2. When feedback is given, the two strategies are iterated until all the given examples converge.
   (a) When AIDA's similarity is too small, update the weights of active features by multiplying α, $(w \leftarrow \alpha \cdot w)$.
   (b) When AIDA's similarity is too large, update the weights of active features by multiplying $1/\alpha$, $(w \leftarrow 1/\alpha \cdot w)$.

Table 6.3. Weight Update by WINNOW

6.4. Related Work

This section relates AIDA to other previous research. [76] and [70] concern the methods for flexible retrieval of a bilingual corpus. [76] reduces useless expressions by incorporating a case frame structure of verbs. The generalization using case frames is effective only when a large amount of data is available. In addition, the accuracy of current parsing techniques may become an obstacle to the approach. [70] proposes a character level search to make the best use of the kanji information. For example, the method detects the similarity between '観察' and '観測'. However, more frequent phrase retrieval requires generalization with part-of-speech tags. We consider the local level structures by performing morphological analysis. Although there were about 5% errors in the results of the morphological analysis, no serious problems were found because the errors occurred consistently both in corpus and input word sequences. Note that our method can easily incorporate the character level information of Kanji and the semantic categories of a thesaurus as proposed in the above research.

Although [48, 64] combined an online dictionary and corpus, their objectives are quite different from that of AIDA. [48] proposes a method to augment an existing bilingual dictionary with a new usage and its frequency information taken from corpora. The method was tested only with a small number of verbs. [64] on the other hand aims at foreign language support. Users can look up words from a French-Dutch dictionary and a French corpus through a graphical interface.

AIDA applied corpus-based natural language techniques to various language sources and combined them. In this example-driven spirit, it was greatly inspired by the COBUILD English dictionary [15]. The difference lies in the adaptive functions such as the language source selection, the collocation extraction and the learning of user preferences. In addition, AIDA emphasizes the use of bilingual corpora.

Further functions are needed to augment AIDA. For example, word sense disambiguation is important to reduce useless outputs. The stochastic approach [55] can also be applied to couple a dictionary and a corpus by determining which entry of the dictionary a word usage in the corpus belongs to.

6.5. Summary

This chapter has described an adaptive and integrated dictionary agent AIDA. By combining morphological analysis and simple machine learning methods, we can tightly couple dictionaries and corpora. Further research will be needed to provide the system with rich functions by investigating the actual use of AIDA in various communication roles.

Chapter 7

Conclusions

7.1. Summary

This thesis has described the machine learning approach to natural language processing in order to tackle the *knowledge* problem. The construction of knowledge for NLP has been conceptually classified into three steps.

1. Enumerate linguistic features that potentially influence the target language task.

2. Select features and combine them into a form of rules.

3. Determine preference parameters of rules.

Throughout the thesis, I have tried to emphasize two points.

1. The first step cannot be easily solved by the current machine learning techniques, which instead help human linguists by only extracting significant co-occurrence of words and phrases.

2. The second and third steps can be robustly performed by state-of-the-art machine learning techniques, particularly when the representation of the task has a simple structure such as a context tree or decision tree.

The machine learning approach (probably other approaches, too) succeeds when good features, good and simple representations, and a good learning algorithm are all prepared. Although not all of the NLP tasks have these properties, the problems described in the thesis satisfy these requirements:

1. **Bilingual Sentence Alignment Using Statistical and Dictionary Information** The only linguistic feature used is the word-correspondence of dictionaries. The representation is a non-crossing, weighted and undirected graph. The learning algorithm corresponds to iterative refinements of the weights. Although sentence alignment is essentially an unsupervised learning task, the performance of the method is good because each of the three steps has very a simple structure.

2. **Learning Bilingual Collocations by Word-level Sorting** The linguistic features used are word forms and part-of-speech tags. The representation for fixed collocations is the pointer table, and the learning algorithm is very simple statistics. Although the task is unsupervised, fixed collocations are accurately extracted. Flexible collocations aren't on the other hand. This may be because the two linguistic features are too weak for the task.

3. **Mistake-Driven Mixture Approach to Morphological Analysis** The linguistic features are part-of-speech tags and frequent word forms. The representation is hierarchical context trees, and boosting is adopted as the learning algorithm. Because these three components complement each other, the proposed method attains good performance.

4. **Mistake-Driven Mixture Approach to Dependency Analysis** Several features are devised by human linguists. The representation is a simple decision tree that can utilize the features effectively. Boosting is adopted as the learning algorithm and found to be very powerful.

5. **AIDA: Adaptive and Integrated Dictionary Agent** Several features are prepared by a human linguist. The representation is in the form of a feature-weight vector. The learning algorithm is the well-known WINNOW method.

7.2. Future Directions

When we decide to further develop the machine learning approach for language processing, what kind of research will be needed in the future? In the rest of the chapter, I will discuss the directions that seem to be important fro extending the research conducted in the thesis.

1. The most important and challenging research direction concerns how linguistic features are constructed (the first step), a question which has significant influence on the target language task. In this thesis, I have only shown applications whose features are simple or can be empirically handled by experienced language engineers. However, the success of more complicated tasks requires automatic detection of the language features. Feature detection is one of the greatest bottlenecks of symbol processing because learning them is tightly coupled with sensory signals. We humans acquire and use languages in association with vision, motion and audition of the real world. Therefore, the computational neuroscience approach [43] to cognitive activities might be promising to the basic research in this direction.

2. More practical research should pursue learning algorithms suitable for language processing (the second and the third steps). In unsupervised learning, the most interesting direction is to extract full translation patterns from a bilingual corpus. The sentence alignment algorithm, the fixed collocation extraction method and the robust dependency analysis method reported in this thesis should provides strong clues for the task. Furthermore, a new extraction algorithm should consider not only translation patterns but also the overall performance of the translation system. In other words, the learning algorithm needs to decide how the extracted rules are used (the third step) as well as what are the rules (the second step).

   Now let us move on to supervised learning. The Adaboost algorithm [27] effectively deals with infrequent and colloquial expressions of natural language, as shown in the thesis. Other margin-based [79] leaning algorithms such as Support Vector Machine [18, 79] might be helpful to handle natural language corpus data. It would be interesting to investigate values of margin for various types of language tasks.

3. An alternative promising direction concerns the cooperation between computational linguists and machine learning researchers. As I have emphasized in this thesis, the most practical machine learning approach at present is for human linguists to find effective features by using machine learning techniques. The current machine learning techniques and high-spec computers have made the second and third steps feasible. The preference parameters are automatically set when a machine learning algorithm quantitatively selects the significant features. This cooperative strategy will reduces the human burden involved in the conventional trial-and-error rule compilation. I believe this cooperative approach will be a new shape of natural language engineering in the near future. The following summarizes several issues to be resolved in this direction.

    - Machine learning techniques for language processing are now mature enough to cooperate with computational linguists who have engaged in the development of practical systems. The computational linguists know which features have an effect on the performance of the system, but cannot optimize the preference parameters because the size of rules is too large. The power of the machine learning approach should be proved in such a situation. I think the most promising domain for such cooperation is dependency analysis because the features for the task are already known to some extent and their combinations and preferences are difficult to set.
    - I proposed a new learning-driven language engineering in which a computational linguist tests the potential features in his mind by a learning algorithm. If a feature is judged to be significant, rules and preferences are automatically determined by the algorithm. Such an interaction between human and machine requires a sophisticated interface. The computational linguist should know the effect of each feature and understand the output rules of the algorithm to further improve the fea-

ture set. A graphical environment for this purpose is needed to advance the leaning-driven language engineering.

Bibliography

[1] P. Bartlett. The Sample Complexity of Pattern Classification with Neural Networks: The size of the Weights is More Important than the size of Network. *IEEE Transaction on Information Theory*, 44(2):525–536, March 1998.

[2] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996.

[3] L. Breiman, J. Friedman, R. Olshen, and J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[4] E. Brill. A simple rule-based part of speech tagger. In *Proc. Third Conference on Applied Natural Language Processing*, pages 152–155, 1992.

[5] E. Brill. Some advances in transformation-based part of speech tagging. In *Proc. 12th National Conference on Artificial Intelligence*, pages 722–727, 1994.

[6] P. Brown et al. Aligning sentences in parallel corpora. In *the 29th Annual Meeting of the Association for Computational Linguistics*, pages 169–176, 1991.

[7] P. Brown et al. The mathematics of statistical machine translation. *Computational Linguistics*, 19(2):263–311, June 1993.

[8] B. Carpenter. *The logic of Typed Feature Structures.* Cambridge University Press, 1992.

[9] E. Charniak. *Statistical Language Learning.* The MIT Press, 1993.

[10] E. Charniak. Statistical Parsing with a Context-free Grammar and Word Statistics. In *Proc. 15th National Conference on Artificial Intelligence*, pages 598–603, 1997.

[11] E. Charniak, C. Hendrickson, N. Jacobson, and M. Perkowits. Equations for Part-of-Speech Tagging. In *Proc. 11th National Conference on Artificial Intelligence*, pages 784–789, 1993.

[12] S.F. Chen. Aligning sentences in bilingual corpora using lexical information. In *the 31th Annual Meeting of the Association for Computational Linguistics*, pages 9–16, 1993.

[13] K. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. ACL 2nd Conference on Applied Natural Language Processing*, pages 126–143, 1988.

[14] K. Church. Char_align: A program for aligning parallel texts at the character level. In *the 31th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, 1993.

[15] COBUILD. *COBUILD English Dictionary.* Collins, 1995.

[16] M. Collins. A New Statistical Parser based on bigram lexical dependencies. In *Proc. 34th Annual Meeting of Association for Computational Linguistics*, pages 184–191, 1996.

[17] M. Collins. Three Generative, Lexicalised Models for Statistical Parsing. In *Proc. 35th Annual Meeting of Association for Computational Linguistics*, pages 16–23, 1997.

[18] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[19] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

[20] I. Dagan and K. Church. *Termight:* identifying and translating technical terminology. In *Proc. Fourth Conference on Applied Natural Language Processing*, pages 34–40, 1994.

[21] F. Debili and E. Sammouda. Appariement des phrases de textes bilingues. In *Proc. 14th International Conference on Computational Linguistics*, pages 518–524, 1992.

[22] H. Drucker and C. Cortes. Boosting decision trees. In *Advances in Neural Information Processing Systems 8*, pages 479–485, 1996.

[23] Japan Electronic Dictionary Research Institute Ltd. EDR. *The EDR Electronic Dictionary Technical Guide*, 1995.

[24] B. Efron. Bootstrap: another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

[25] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, 1993.

[26] Y. Freund and R. Schapire. Experiments with a New Boosting algorithm. In *Proc. 13rd International Conference on Machine Learning*, pages 148–156, 1996.

[27] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[28] M. Fujio and Y. Matsumoto. Japanese dependency structure analysis based on statistics. In *SIGNL NL117-12*, pages 83–90, 1997. (in Japanese).

[29] P. Fung. A pattern matching method for finding noun and proper nouns translations from noisy parallel corpora. In *Proc. 33rd Annual Meeting of the Association for Computational Linguistics*, pages 236–243, 1995.

[30] P. Fung and K. Church. K-vec: A new approach for aligning parallel texts. In *Proc. 15th International Conference on Computational Linguistics*, pages 1096–1102, 1994.

[31] W. Gale and K. Church. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102, March 1993.

[32] G.H. Gonnet, R.A. Baeza-Yates, and T. Snider. New Indices for Text: PAT Trees and PAT Arrays. In W. Franks, editor, *Information Retrieval*, chapter 5, pages 66–82. Prentice-Hall, 1992.

[33] M. Haruno, S. Ikehara, and T. Yamazaki. Learning Bilingual Collocations by Word-level Sorting. In *Proc. 16th International Conference on Computational Linguistics*, pages 525–530, 1996.

[34] M. Haruno and Y. Matsumoto. Mistake-driven Mixture of Hierarchical Tag Context Trees. In *Proc. 35th Annual Meeting of the Association for Computational Linguistics*, pages 230–237, 1997.

[35] M. Haruno, S. Shirai, and Y. Ooyama. Using decision trees to construct a practical parser. *Machine Learning*, 1998. (in press).

[36] M. Haruno and T. Yamazaki. High-Performance Bilingual Text Alignment Using Statistical and Dictionary Information. In *Proc. 34th Annual Meeting of Association for Computational Linguistics*, 1996.

[37] D. Hull, J. Pedersen, and H. Schütze. Method combination for document filtering. In *Proc. ACM SIGIR 96*, pages 279–287, 1996.

[38] S. Ikehara, S. Shirai, and H. Uchino. A statistical method for extracting uninterrupted and interrupted collocations from very large corpora. In *Proc. 16th International Conference on Computational Linguistics*, pages 574–579, 1996.

[39] S. Ikehara, A. Yokoo, and M. Miyazaki. Semantic analysis dictionaries for machine translation (in Japanese). In *NLC 91-19*. IEICE, 1991.

[40] NLRI (National Language Research Institute). *Word List by Semantic Principles.* Syuei Syuppan, 1964. (in Japanese).

[41] H. Kaji, Y. Kida, and Y. Morimoto. Learning translation templates from bilingual text. In *Proc. 14th International Conference on Computational Linguistics*, pages 672–678, 1992.

[42] M. Kameda. A portable & quick japanese parser: Q_jp. In *Proc. 16th International Conference on Computational Linguistics*, pages 616–621, 1996.

[43] M. Kawato. *Computational Theory of Brain.* Sangyo-Thôsho, 1996. in Japanese.

[44] M. Kay. Algorithm schemata and data structure in syntactic processing. Technical Report CLS-80-12, Xerox PARC, 1980.

[45] M. Kay and M. Roscheisen. Text-translation alignment. *Computational Linguistics*, 19(1):121–142, March 1993.

[46] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory.* The MIT Press, 1994.

[47] M. Kitamura and Y. Matsumoto. Automatic Extraction of Word Sequence Correspondences in Parallel Corpora. In *Proc. Fourth Workshop on Very Large Corpora*, pages 79–87, 1996.

[48] J. Klavans and E. Tzoukermann. The BICORD System. In *Proc. 13rd International Conference on Computational Linguistics*, pages 174–179, 1990.

[49] A. Kumano and H. Hirakawa. Building an MT dictionary from parallel texts based on linguistic and statistical information. In *Proc. 15th International Conference on Computational Linguistics*, pages 76–81, 1994.

[50] J. Kupiec. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6:225–242, 1992.

[51] S. Kurohashi and M. Nagao. A syntactic analysis method of long japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534, 1994.

[52] S. Kurohashi, T. Nakamura, Y. Matsumoto, and M. Nagao. Improvements of Japanese morphological analyzer juman. In *Proc. International Workshop on Sharable Natural Language Resources*, pages 22–28, 1994.

[53] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[54] N. Littlestone. Comparing several linear-threshold learning algorithm on tasks involving superfluous attributes. In *Proc. 12nd International Conference on Machine Learning*, pages 353–361, 1995.

[55] A. Luk. Statistical Sense Disambiguation with Relatively Small Corpora Using Dictionary Definitions. In *Proc. 33rd Annual Meeting of the Association for Computational Linguistics*, 1995.

[56] D. Magerman. Statistical Decision-Tree Models for Parsing. In *Proc.33rd Annual Meeting of Association for Computational Linguistics*, pages 276–283, 1995.

[57] Mainichi. *CD Mainichi Shinbun.* Nichigai Associates Co, 1993.

[58] M. Marcus, B. Santorini, and M. Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, June 1993.

[59] Y. Matsumoto, H. Ishimoto, and T. Utsuro. Structural matching of parallel texts. In *the 31th Annual Meeting of the Association for Computational Linguistics*, pages 23–30, 1993.

[60] Y. Matsumoto, A Kitauchi, T. Yamashita, Y. Hirano, O. Imaichi, and T. Imamura. *Japanese Morphological Analysis System Chasen Manual*, 1997. NAIST Technical Report NAIST-IS-TR97007.

[61] H. Murao. Studies on bilingual text alignment. Bachelor Thesis, Kyoto University (in Japanese), 1991.

[62] M. Nagao and S. Mori. A new method of n-gram statistics for large number of n and automatic extraction of words and phrases from large text data of Japanese. In *Proc. 15th International Conference on Computational Linguistics*, pages 611–615, 1994.

[63] M. Nagata. A Stochastic Japanese Morphological Analyzer Using Forward-DP Backward-A^* N-Best Search Algorithm. In *Proc. 15th International Conference on Computational Linguistics*, pages 201–207, 1994.

[64] J. Nerbonne and P. Smit. GLOSSER-RuG: in Support of Reading. In *Proc. 16th International Conference on Computational Linguistics*, pages 830–835, 1996.

[65] F. Pereira, Y. Singer, and N. Tishby. Beyond Word N-Grams. In *Proc. Third Workshop on Very Large Corpora*, pages 95–106, 1995.

[66] J.R. Quinlan. *C4.5 Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.

[67] L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, 1986.

[68] J. Rissanen. A universal data compression system. *IEEE Transaction on Information Theory*, 29(5):656–664, September 1983.

[69] D. Ron, Y. Singer, and N. Tishby. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25(2/3):117–149, November/December 1996.

[70] S. Sato. CTM: An Example-Based Translation Aid System. In *Proc. 14th International Conference on Computational Linguistics*, pages 1259–1263, 1992.

[71] S. Sato and M. Nagao. Toward memory-based translation. In *Proc. 13th International Conference on Computational Linguistics*, pages 247–252, 1990.

[72] R. Schaipre, Y. Freund, P. Bartlett, and W.S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proc. 14th International Conference on Machine Learning*, pages 338–347, 1997.

[73] H. Schütze and Y. Singer. Part-of-speech tagging using a variable markov model. In *the 32th Annual Meeting of the Association for Computational Linguistics*, pages 181–187, 1994.

[74] F. Smadja. Retrieving collocations from text: Xtract. *Computational Linguistics*, 19(1):143–177, March 1993.

[75] F. Smadja and K. McKeown. Translating collocations for use in bilingual lexicons. In *ARPA Human Language Technology Workshop 94*, pages 152–156, 1994.

[76] E. Sumita and Y. Tsutsumi. A practical method of retrieving similar examples for translation aid (in Japanese). *Transaction of IEICE*, J-74(D-II)(10):1437–1447, 1991.

[77] S. Tobita. *How to find useful Dictionaries.* Babel Press, 1995. (in Japanese).

[78] T. Utsuro, H. Ikeda, M. Yamane, Y. Matsumoto, and M. Nagao. Bilingual text matching using bilingual dictionary and statistics. In *Proc. 15th International Conference on Computational Linguistics*, pages 1076–1082, 1994.

[79] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.

[80] M J. Weinberger, J J. Rissanen, and M. Feder. A universal finite memory source. *IEEE Transaction on Information Theory*, 41(3):643–652, May 1995.

[81] F M J. Willems, Y M. Shtarkov, and T J. Tjalkens. The context-tree weighting method: Basic properties. *IEEE Transaction on Information Theory*, 41(3):653–664, May 1995.

[82] D. Wu. Aligning a parallel English-Chinese corpus statistically with lexical criteria. In *the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 80–87, 1994.

[83] S. Yoshida. Japanese Syntactic analysis based on the modification between two bunsetsu. *Transactions of the IEICE*, 55-D(4), 1972. (in Japanese).

List of Publications

List of Major Publications

[1] Masahiko Haruno, Yasuharu Den, Yuji Matsumoto and Makoto Nagao, Bidirectional Chart Generation of Natural Language Texts. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pp.350-356, 1993.

[2] Masahiko Haruno and Takefumi Yamazaki, High-Performance Bilingual Text Alignment Using Statistical and Dictionary Information, In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp131-138, 1996.

[3] Masahiko Haruno, Satoru Ikehara and Takefumi Yamazaki, Learning Bilingual Collocations by Word-level Sorting, In *Proceedings of the 16th International Conference on Computational Linguistics*, pp525-530, 1996.

[4] Masahiko Haruno, Yasuharu Den and Yuji Matsumoto, A Chart-based Semantic Head Driven Generation Algorithm. In Giovanni Adorni and Michael Zock (eds.) *Trends in Natural Language Generation*, pp.300-313, Springer, 1996.

[5] Masahiko Haruno, Bilingual Text Alignment Using Statistical and Dictionary Information (in Japanese), *Transactions of Information Processing Society of Japan, 38(4)*, 719-726, 1997.

[6] Masahiko Haruno and Yuji Matsumoto, Mistake-Driven Mixture of Hierarchical-Tag Context Trees, In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pp230-237, 1997.

[7] Masahiko Haruno and Takefumi Yamazaki, High-Performance bilingual text alignment using statistical and dictionary information, *Journal of Natural Language Engineering, 3(1)*, 1-14, Cambridge University Press, 1997.

[8] Masahiko Haruno and Satoru Ikehara, Two-step Extraction of Bilingual Collocations by Word-Level Sorting, *IEICE Transactions on Information and Systems*, (in press), 1998.

[9] Masahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, Using a Decision Tree to Construct a Practical Parser (in Japanese), *Transactions of Information Processing Society of Japan* , (to appear), 1998.

[10] Masahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, Using Decision Trees to Construct a Practical Parser, In *Proceedings of 36th Annual Meeting of the Association for Computational Linguistics*, pp505-511, 1998.

[11] Masahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, Using Decision Trees to Construct a Practical Parser, *Machine Learning Journal*, Kluwer Academic Press, (in press), 1998.

[12] Masahiko Haruno, AIDA: An Adaptive and Integrated Dictionary Agent, submitted to *Journal of Natural Language Engineering.*

List of Other Publications

[1] Masahiko Haruno, Yuji Matsumoto and Makoto Nagao, Handling gaps in wh-constructions with a concurrent parsing system (in Japanese), In *Proceedings of the 42nd Convention of IPSJ*, 1991.

[2] Masahiko Haruno, Yasuharu Den and Yuji Matsumoto, A Parallel Generation Mechanism based on the Bottom-up Chart Algorithm (in Japanese), *IPSJ-SIGNL 92-NL-88-13*, 1992.

[3] Masahiko Haruno, Yuji Matsumoto and Makoto Nagao, An Extended Chart Parsing Algorithm for Gap Handling, *IPSJ-SIGNL 92-NL-89-5*, 1992.

[4] Masahiko Haruno, Yuji Matsumoto and Makoto Nagao, A Grammar Driven Approach to Cooperative Sentence Generation (in Japanese), In *Proceedings of the 9th convention of JSSST*, pp145-148, 1992.

[5] Masahiko Haruno, Yuji Matsumoto and Makoto Nagao, A Grammar-Driven Approach to Natural Language Text Generation (in Japanese), *Symposium on Fundamental Issues in Natural Language Processing*, JSSST & IEICE, pp88-96, 1992.

[6] Masahiko Haruno, Yasuharu Den and Yuji Matsumoto, Bidirectional Chart Generation Algorithm. In *Proceedings of the 4th European Workshop on Natural Language Generation*, pp.31-42, 1993.

[7] Masahiko Haruno, Japanese Verbal Case Frame Acquisition by Least Generalization (in Japanese), In *Proceedings of the 11th convention of JSSST*, pp145-148, 1994.

[8] Masahiko Haruno, Japanese Verbal Case Frame Acquisition by Least Generalization (in Japanese), *Symposium on Learning in Natural Language Processing*, JSSST & IEICE, pp9-16, 1994.

[9] Masahiko Haruno, A Case Frame Learning Method for Japanese Polysemous Verbs, In *Proceedings of the AAAI Spring Symposium, Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity and Generativity*, pp45-50, 1995.

[10] Masahiko Haruno, Verbal Case Frame Acquisition as Data Compression, In *Proceedings of the 5th Workshop on Natural Language Understanding and Logic Programming*, pp97-105, 1995.

[11] Masahiko Haruno, Verbal Case Frame Acquisition by Least Generalization (in Japanese), *IPSJ-SIGNL 95-NL-108-5*, 1995.

[12] Masahiko Haruno and Takefumi Yamazaki, Bilingual Text Alignment Using Statistical and Dictionary Information (in Japanese), *IPSJ-SIGNL 96-NL-112-4*, 1996.

[13] Masahiko Haruno and Yuji Matsumoto, Japanese Morphological Analyzer Using Context Tree (in Japanese), *IPSJ-SIGNL 96-NL-112-5*, 1996.

[14] Masahiko Haruno and Takefumi Yamazaki, High-Performance Bilingual Text Alignment Using Statistical and Dictionary Information (in Japanese), In *Proceedings of the 2nd Annual Meeting of ANLP*, pp131-138, 1996.

[15] Masahiko Haruno, AIDA: An Adaptive and Integrated Dictionary Agent (in Japanese), *IPSJ-SIGNL 97-NL-120-18*, 1997.

[16] Yutaka Sasaki and Masahiko Haruno, RHB^+: Type-Oriented ILP System Learning from Positive Data. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, pp.894-899, 1997.

[17] Masahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, A Japanese Dependency Parser Using a Decision Tree (in Japanese), *Symposium on Practical Systems in Natural Language Processing*, http://www.csl.sony.co.jp/person/nagao/nlsym97/, 1997.

[18] Masahiko Haruno, Satoshi Shirai and Yoshifumi Ooyama, Japanese Dependency Analysis based on decision tree mixture (in Japanese), In *Proceedings of the 4th Annual Meeting of ANLP*, pp.217-220, 1998.

[19] Takefumi Yamazaki and Masahiko Haruno, BACCS: Graphical Sentence-Level Aligned Corpora Construction Environment (in Japanese), *Journal of JSAI, 13(3)*, pp.144-151, 1998.

[20] Masahiko Haruno, Japanese Part-of-speech Tagger Using Context Trees (in Japanese), In *Proceedings of 1998 Workshop on Information- Based Induction Sciences (IBIS'98)*, pp.103-110, 1998.

Abbreviations

ANLP Association for Natural Language Processing

IEICE Institute of Electronics, Information and Communication Engineers

IPSJ Information Processing Society of Japan

- **SIGNL** Special Interest Group on Natural Language Processing

JSAI Japan Society for Artificial Intelligence

JSSST Japan Society for Software Science and Technology