Feature-Rich Information Extraction for the Technical Trend-Map Creation

Risa Nishiyama, Yuta Tsuboi, Yuya Unno, and Hironori Takeuchi
IBM Research - Tokyo
1623-14 Shimo-tsuruma, Yamato-shi
Kanagawa 242-8502, Japan
{lisa, yutat, yunno, hironori}@jp.ibm.com

ABSTRACT

The authors used a word sequence labeling method for technical effects and base-technology extraction in the Technical Trend Map Creation Subtask of the NTCIR-8 Patent Mining Task. The method labels each word based on CRF (Conditional Random Field) trained with labeled data. The word features employed in the labeling are obtained by using explicit/implicit document structures, technology fields assigned to the document, effect context phrases, phrase dependency structures and a domain adaptation technique. Results of the formal run showed that the explicit document structure feature and the phrase dependency structure feature are effective in annotating patent data. The implicit document structure feature and the domain adaptation feature are also effective for annotating paper data.

Categories and Subject Descriptors

H.3.1 [**Information Systems**]: Information Storage and Retrieval-Content Analysis and Indexing; I.2.7 [**Computing Methodologies**]: Artificial IntelligenceNatural Language Processing

General Terms

Algorithms, Design, and Experimentation

Keywords

word sequence labeling, Conditional Random Fields, explicit/implicit document structures, technology fields, effect contexts, phrase dependency structures, domain adaptation

1. INTRODUCTION

Technical effects and base technologies are key information in technical documents such as patent disclosures and research papers. In the Trend Map Creation Subtask of the NTCIR-8 Patent Mining Task, systems to annotate phrases mentioning those technical effects and base technologies are proposed and compared [4].

In this paper, several approaches used for this task are explained. We considered this information extraction task as a sequential labeling problem, which is an often-applied assumption for the named entity recognition task. This paper first introduces the system overview and the word labeling method using Conditional Random Field (CRF). Next, the features used to represent each word in the word sequence labeling is explained. Then, the effectiveness of the features is compared in the formal run results by applying the proposed method against Japanese patent and paper data.

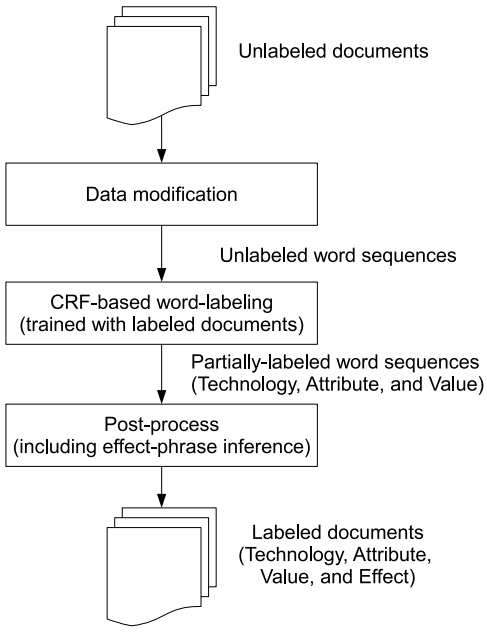

Figure 1: System overview

2. ANNOTATION METHOD

2.1 System Overview

Figure 1 shows an overview of the annotation system. First, the data modification component converts unlabeled documents into word sequences. The text in the unlabeled documents is splitted into sentences, and each of the sentences is segmented by a Japanese morphological analyzer. A sentence in an unlabeled document is converted into a word sequence, and each word is represented as a feature vector, whose elements are explained in Section 2.3. Next, the word labeling component trained with the labeled documents accepts each word sequence and infers the label of each word in the word sequence as explained in Section 2.2. The labeler is normally trained with labeled patent data when labeling patents, and with labeled research paper data when labeling papers. Finally, the post-processing annotates the input text based on the labels inferred by the word labeling component.

2.2 Word labeling using CRFs

In this work, we deal with the information extraction task as a sequence labeling problem and solve it with a machine learning tech-

$\boldsymbol{x}$	O	O	B-V	O	B-A	I-A
$\boldsymbol{y}$	The	system	minimizes	the	power	loss

Figure 2: An example of labeled words.

nique. The phrases of the base technologies, effect attributes, and effect values are encoded in the IOB2 representation [6] as shown in Figure 2. Note that the range of an effect phrase is determined in the post processing, since each consists of a set of attributes and values. To represent a tagged phrase, the first word of a phrase is labeled with the "B" label and the following words of that phrase are labeled with "I" labels. These labels are combined with class labels: "T" label for technologies, "A" label for attributes, and "V" label for values. In Figure 2, for example, the phrase "power loss" is an attribute phrase. The first word "power" is given a "B-A" label, and the second word "loss" is given a "I-A" label. In addition, the dummy label "O" is used to label words that are not in the phrases of any class. Therefore, the number of distinct labels is 7: B-T, I-T, B-A, I-A, B-V, I-V, and O.

Then this task can be considered as the prediction of a corresponding label sequence $\boldsymbol{y} = (y_1, y_2, \cdots, y_n)$ for a given sentence $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ where $\boldsymbol{x}$ represents the word sequence of a input sentence.

As a sequence labeler, a discriminative log-linear model is used. Let $\boldsymbol{X}$ be the set of all of the input sequences and $\mathcal{Y}(\boldsymbol{x})$ be the set of all of the label configurations. Let $\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{X} \times \mathcal{Y}(\boldsymbol{x}) \rightarrow \mathbb{R}^\ell$ denote a map function from a pair of $\boldsymbol{x}$ and $\boldsymbol{y}$ to an arbitrary vector of ℓ dimensions and $\boldsymbol{w} \in \mathbb{R}^\ell$ denote the vector of the model parameters. The conditional probability of a label sequence $\boldsymbol{y}$ given an observed sequence $\boldsymbol{x}$ is modeled as

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z} \exp(\boldsymbol{w} \cdot \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})), \tag{1}$$

where $\cdot$ denotes the inner product of the vectors. Note that the denominator is the partition function $Z = \sum_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \exp(\boldsymbol{w} \cdot \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}))$. The parameter $\boldsymbol{w}$ can be estimated by using regularized log-likelihood maximization[1] using the labeled data $E = \{(\boldsymbol{x}, \boldsymbol{y})\}$.

To represent the correlation of consecutive labels based on linear-chain CRFs [2], we design the map function $\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$ as the sum of decomposed function

$$\boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} \boldsymbol{\phi}(\boldsymbol{f}(\boldsymbol{x}, i), y_{i-1}, y_i), \tag{2}$$

where $\boldsymbol{f}(\boldsymbol{x}, i) : \boldsymbol{X} \times \mathbb{Z} \rightarrow \mathbb{R}^m$ maps the i-th word into a vector representation of m dimensions. The vector of $\boldsymbol{f}(x, i)$ is the feature vector for the i-th word and the elements of the feature vector are described in the following section. Note that in Section 2.3.5, we will use tree-type CRFs for a dependency parsing in which label structure $\boldsymbol{y}$ represents the modification relationship of *Bunsetsu* phrases. We applied CRFs to projective dependency parsing and inside-outside algorithm [3].

2.3 Features

Each word is represented as a feature vector as shown in Figure 3 to be processed in the CRF-based word labeler. The feature vector consists of features of the represented word itself and features of the words surrounding the represented word within the fixed size window, since these surrounding words also have some common characteristics to determine whether or not the word is a part of the phrase to be extracted [2]. Size of the feature vector using all of the proposed features is approximately 100K - 200K.

[1] In experiments, $\boldsymbol{w}$ was optimized by the combination of a stochastic gradient method and a variant of the Newton methods. The regularization parameter of CRFs was determined by 5-fold cross-validation.

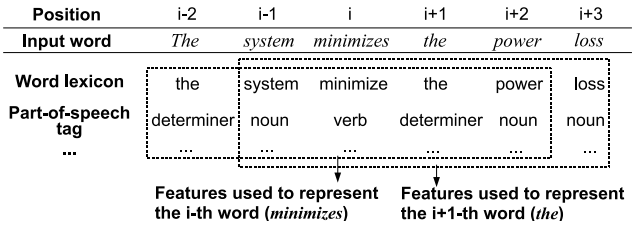

Figure 3: Feature example (window size = 2)

The features used to represent a word and the surrounding words are explained in the later sections. Note that although all experiments are done by using Japanese text data, examples of feature values are given by taking an similar expressions in English to share the characteristics of these features with English speakers.

2.3.1 Baseline features

Here are the baseline features:

- Word lexicon
- Part-of-speech tag
- Character type
- Word prefix type
- Word suffix type

Note that all of these features are binary features, and for the word lexicon feature, each numeric value contained in a word is converted into "#" for generalization.

Distinguishing among the character types of words should be effective for detecting technology, attribute, and value phrases. Each word is classified into one or more of those classes defined in Table 1 by using regular expression patterns. Since many technology names, especially chemical names, are described in *katakana* characters in Japanese text, words consisting of *katakana* are likely to be in a technology phrase. Similarly, words consisting numeric values and units such as "sec" and "Hz" are likely to appear in a value phrase.

Besides the character type feature, certain types of prefixes and suffixes can also be exploited. Many value phrases contain words starting with prefixes indicating an amount or degree of a value such as "high" or "much" and prefixes indicating state change such as the "en-" of "enlarge" or the "in-" of "inhibit." The state changes are also expressed in suffixes such as the "-ize" of "minimize" or the "-ate" of "carbonate." The prefix and suffix feature values are true if the word starts with one of the prefixes or ends with one of the suffixes defined in Table 2

2.3.2 Document structure features (sec & pos)

Technical documents often have pre-defined structures. For example, a patent disclosure is composed of a set of specific sections such as "Background" and "Summary of Invention." Places where some types of phrases are likely to appear are expected to be correlated with the document structures. In the example of a patent disclosure, the "Summary of Invention" section often contains effect phrases to explain the technical advantages of the invention. The

[2] The window size was set to 2 in the experiment.

Table 1: Character types

Type	Pattern (in regular expressions)
Hiragana	.*[あ-ん一]+.*
Katakana	.*[ア-ケ一]+.*
English characters	.*[A-Za-z]+.*
Numerals	.*[0-9]+.*
Units	[nm μ M]*(G |sec |A |m |T |K |s |W |l |dB |Hg |cm |mm |km |wt |Pa |mol |Hz |min |φ |℃ |% |Ω |°)
Signs	.*[/／^:〜+-×]+.*

Table 2: High-low prefixes and state-changing suffixes

Prefix	高 (high), 低 (low), 上 (up), 下 (down), 活 (en-), 抑 (in-), 大 (large), 小 (small)
Suffix	化 (-ize), 化する (-ate), 倍 (-times)

sectioning can be considered as an example of an *explicit document structure*. As well as the explicit document structure, *implicit document structure* such as "Introduction-Body-Conclusion," which is often referred as an English technical writing style, is also expected to be useful for finding technology and effect phrases.

To represent the explicit and implicit document structures, two features are used: section (explicit) and relative position (implicit).

The section feature (sec) represents a section where the word appears. This is a binary feature and takes a value representing one of these values: title, abstract, patent-problem, patent-mean, patent-effect [3].

The relative position feature (pos) represents the position of the word in a section. The relative position p of the ith word in a section is given by $p = i/(\#\text{ of words in the section})$. The value of p is classified into one of 4 classes: the 1st quartile ($p < 0.25$), the 2nd quartile ($0.25 \leq p < 0.5$), the 3rd quartile ($0.5 \leq p < 0.75$) and the 4th quartile part ($0.75 \leq p \leq 1.0$) and the relative position feature takes a binary value representing one of these classes.

2.3.3 Technology field feature (ipc)

Technology, attribute, and value phrases often have different characteristics in different technology fields. For example, patents and papers in the chemistry field often contain chemical names, which are often described in *katakana*, as the technologies. By implementing technology field features, these technology field-specific characteristics can contribute to word labeling.

The IPC codes given to each patent and paper are used to determine the technology field of the document. Although the IPC codes are not available in the original papers, we can infer the technology fields of papers by using proceedings or journal information where the paper appeared or by using paper classification methods proposed in the Patent Classification Subtask [4].

The most general classifications of the IPC codes are used for the technical field feature. Several IPC codes are given to a patent or paper, and each IPC code belongs to one of the IPC classes defined as Table 3 [8]. The technical field feature is also designed as a binary feature. If the word is in a document of the IPC class A, then the feature value representing IPC class A is true. Since one patent or paper can belong to two or more IPC classes, multiple technical field feature values can be true.

[3] Although the English patent data provided for this task contains only title and abstract sections, the Japanese patent data contains more fine-grained sections such as title, patent-problem, patent-mean and patent-effect in the distributed training and test datasets [4].

Table 3: IPC classes

A	Human necessities
B	Performing operations, transporting
C	Chemistry, metallurgy
D	Textiles, paper
E	Fixed constructions
F	Mechanical engineering, lightning, heating, weapons, blasting
G	Physics
H	Electricity

2.3.4 Effect context features (ec_p & ec_w)

The number of annotated phrases in the training data is limited compared with the variation of the actual phrases to be annotated in the test data. Regarding this issue, it is natural to use massive amounts of unlabeled documents to search for phrases that are not present in the annotated data.

Nishiyama has proposed an unsupervised phrase extraction method using contexts, i.e. surrounding phrases of the phrases to be extracted [5]. By enhancing the extraction method proposed in [5], effect phrases are extracted from unlabeled technical documents [4]. In this method, phrases that are likely to depend on pre-defined context phrases are extracted as effect phrases. The pre-defined context phrase, "ことができる" (be able to) was used in the experiment. A phrase p, which consists of a dependency structure of (noun)-(adposition)-(verb or adjective), is evaluated with a reliablity score, $\text{score}_p(p)$, given by the equation

$$\text{score}_p(p) = \frac{\text{freq}(p, c)}{\text{freq}(p)}$$

where c represents the context phrase, $\text{freq}(p, c)$ represents the frequency of occurrences where the phrase p depended on the context phrase c, and $\text{freq}(p)$ represents the frequency of occurrences where the phrase p appeared in the unlabeled documents. $\text{score}_p(p)$ takes a large value for phrases that are likely to appear in the effect context. After eliminating the phrases whose $\text{freq}(p)$ is lower than 6, the top 50% of the phrases based on their reliability scores were extracted in the experiment.

Words appearing in the effect phrases extracted by this method are likely to be attribute or value phrases. To exploit this intuition, two types of unlabeled document features are defined. The first type of feature value, **ec_p**, is true if the word appeared in those extracted effect phrases. On the other hand, the second type of feature value, **ec_w**, takes a real value representing an average score given to a word w in the phrase p. The value $\text{score}_w(w)$ is calculated by the equation

$$\text{score}_w(w) = \frac{\sum_{p \in P_w} \text{score}_p(p)\text{freq}(p)}{|P_w|}$$

where w represents the word represented by the feature, P_w represents the extracted effect phrases containing the word w, and $|P_w|$ represents the size of P_w. $\text{score}_w(w)$ takes a large value for words that are likely to appear in the phrases of large $\text{score}_p(p)$.

[4] In the experiment, 127,028 patent disclosures issued in 2003-2006 were used.

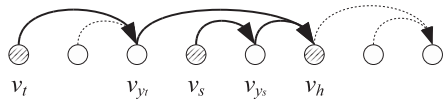

Figure 4: An example of a dependency tree and a path between two bunsetsus.

2.3.5 Dependency structure features (dep)

Dependency structures are helpful when extracting terms. In this paper, we deal with dependency structures that represents relationships among *bunsetsu* (Japanese phrasal unit). Terms can be extracted by using terms near to them in a dependency tree. The distance in the trees is used as a *dependency structure feature*.

The most straightforward way to implement the distance as a word feature is using a distance determined by a syntactic analysis, but such a method is sensitive to errors of in the syntactic analysis. In particular, long sentences, which often appear in patent documents, are error-prone for most of the conventional syntactic analysis methods. Because of this problem, we use the distribution of a probabilistic dependency parser instead of a one-best dependency structure.

Figure 4 shows an example of a dependency tree and a path between the t-th bunsetsu and the s-th bunsetsu. In this figure, the vertex of the k-th bunsetsu is represented as v_k. In a tree structure, there is always only one path between any two bunsetsu. The path between v_t and v_s is represented as a sequence of bold arrows in the figure, and a common ancestor bunsetsu, denoted as v_h, always exists in this path.

The relevance function, r, is defined as a product of edge weights between two bunsetsu on a dependency tree as

$$r(t,s) = \prod_{(i,j)\in \mathrm{PATH}(t,h)} \theta(i,j) \prod_{(i,j)\in \mathrm{PATH}(s,h)} \theta(i,j), \qquad (3)$$

where $\theta (0 < \theta < 1)$ is a weight parameter and $\mathrm{PATH}(t,s)$ is a set of index pairs representing each of the paths between the t-th and the s-th bunsetsu. This function exponentially decays as a bunsetsu becomes farther away, because $\theta < 1$. When t and s are near in a tree, the value of this function is high, and vice versa.

The value of the relevance function depends on the parsing result. In order to make the distance function robust against parsing errors, Unno and Tsuboi [7] proposed an efficient method to calculate the expected value of the distance function between two bunsetsu in a dependency trees. We extend this method to calculate the expected value of the relevance function defined in Equation (3).

As in [7], the probability of each dependency tree is approximated as a product of the dependency marginal probabilities to handle exponential number of candidates. Let the input bunsetsu list be $\boldsymbol{b}$, the length of the list $\boldsymbol{b}$ be $|\boldsymbol{b}|$, the output dependency structure be $\boldsymbol{d} = \{d_1, \ldots, d_{|\boldsymbol{b}|}\}$, the index of the bunsetsu modified by the k-th bunsetsu in the structure $\boldsymbol{d}$ be d_k, and the set of candidate dependency structures for $\boldsymbol{b}$ be $\mathcal{D}(\boldsymbol{b})$. Consider t and s as indexes of an arbitrary pair of bunsetsu, then the expected value of the relevance function for the approximated dependency distribution, $R_e(t,s) = \sum_{\boldsymbol{d}\in\mathcal{D}(\boldsymbol{b})} \mathrm{P}(\boldsymbol{d}|\boldsymbol{b}) r(t,s)$, can be calculated using a recursive function as in [7]. The relevance function is defined as the recursive function

$$r(t,s) = \begin{cases} \theta(t,d_t) r(d_t,s) & (t<s) \\ 1 & (t=s) \\ \theta(s,d_s) r(t,d_s) & (t>s). \end{cases}$$

The definition of $R_e(t,s)$ is modified by using this equation. For $t < s$, the definition of $R_e(t,s)$ with $\mathrm{P}(d_k|\boldsymbol{b})$, which is abbreviated as p_k for simplicity in this equation, is

$$\begin{aligned} R_e(t,s) &= \sum_{\boldsymbol{d}\in\mathcal{D}(\boldsymbol{b})} \left(\prod_{k=1}^{|\boldsymbol{b}|} p_k \right) r(t,s) \\ &= \sum_{d_1} p_1 \cdots \sum_{d_k} p_k \cdots \sum_{d_{|\boldsymbol{b}|}} p_{|\boldsymbol{b}|} \{\theta(t,d_t) r(d_t,s)\} \\ &= \sum_{d_1} p_1 \cdots \sum_{d_t} p_t \theta(t,d_t) \sum_{d_{t+1}} p_{t+1} \cdots \sum_{d_{|\boldsymbol{b}|}} p_{|\boldsymbol{b}|} r(d_t,s). \end{aligned}$$

Since only backward dependencies are permitted in Japanese sentences, $R_e(t,s)$ is independent of $d_1 \ldots d_{t-1}$, and $R_e(d_t,s) = \sum_{d_{t+1}} p_{t+1} \cdots \sum_{d_{|\boldsymbol{b}|}} p_n r(d_t,s)$. This means R_e can be represented as

$$R_e(t,s) = \sum_{d_t} p_t \theta(t,d_t) R_e(d_t,s).$$

By transforming the original function in the same way for the case of $t > s$, the recursive equations become

$$R_e(t,s) = \begin{cases} \sum_{d_t} \mathrm{P}(d_t|\boldsymbol{b}) \theta(t,d_t) R_e(d_t,s) & (t<s) \\ 1 & (t=s) \\ \sum_{d_s} \mathrm{P}(d_s|\boldsymbol{b}) \theta(s,d_s) R_e(t,d_s) & (t>s). \end{cases}$$

In our experiments, θ always returns 0.5.

We use the value of $R_e(t,s)$ as the feature value for words in a bunsetsu $\boldsymbol{b}_t$ conditioned by the words in a bunsetsu $\boldsymbol{b}_s$. Let ω be the lexical form of any word in the sentence, $\mathrm{BID}(\omega)$ be the index set of bunsetsu containing ω, and β_k be the index of the bunsetsu containing the k-th word. For the k-th word, we define two types of features conditioned by ω and the relative positions of the k-th word and ω:

$$f_{<,\omega}(\boldsymbol{x},k) = \max_{p\in \mathrm{BID}(\omega) \text{ s.t. } \beta_k < p} R_e(\beta_k, p)$$

and

$$f_{>,\omega}(\boldsymbol{x},k) = \max_{p\in \mathrm{BID}(\omega) \text{ s.t. } \beta_k > p} R_e(\beta_k, p).$$

Note that, if ω appears more than once in a sentence, then this feature function returns the maximum value of $R_e(\beta_k, p)$.

To evaluate the effectiveness of using the expected value of the relevance function, we compare the above feature with a direct and deterministic dependency feature. The one-best dependency feature (dep_onebest) represents whether a bunsetsu modified by a given bunsetsu contains a given word in the one-best dependency tree. It is defined as

$$f_{\text{onebest},\omega}(\boldsymbol{x},k) = \begin{cases} 1 & (\text{the bunsetsu modified by } k \text{ contains } \omega) \\ 0 & (\text{otherwise}). \end{cases}$$

This feature is used as a baseline dependency structure feature.

2.3.6 Domain adaptation features (feda)

Since word distributions and writing styles are different between patents and papers, these can be regarded as different domains. In general, domain adaptation (DA) techniques leverage the performance in the target domain by the labeled and unlabeled data in a similar source domain. Since the numbers of annotated sentences are limited for both the patent and paper domains, we used a DA technique, FEDA [1]. FEDA is a feature augmentation technique that simply adds features for the source and target domains into the

original features. The augmented feature vector for the paper domain is $\boldsymbol{f}_{\text{paper}}(\boldsymbol{x}) = \langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}), \mathbf{0} \rangle$ and that for the patent domain is $\boldsymbol{f}_{\text{patent}}(\boldsymbol{x}) = \langle \boldsymbol{f}(\boldsymbol{x}), \mathbf{0}, \boldsymbol{f}(\boldsymbol{x}) \rangle$ where $\mathbf{0} = \langle 0, 0, \cdots, 0 \rangle \in \Re^m$ is the zero vector. Then this augmented data in both domains is used for predictive modeling, and therefore the weights of the shared features are estimated using the training data from both domains. Even if the prospective prediction rules are different in the patent and paper domains, the weights of these augmented features will be learned correctly for each domain.

2.4 Post-process

In the post-processing, all of the inferred word labels (B-T, I-T, B-A, I-A, B-V, I-V, and O) are applied to annotations of the input text.

Since word labels representing effect phrases have not been inferred in the preceding processes, the effect phrase is inferred by combining the attribute and value phrases that have already been inferred. The effect annotation is done as in Figure 5, where the phrases enclosed in dashed lines represent attribute phrases and the phrases enclosed in dotted lines represent value phrases. From the leaves of a dependency tree, an attribute phrase and a value phrase on the same dependency path are combined with the other as phrases enclosed in solid lines in the figure. Note that attributes and values that are not able to form a pair with the other, such as "容易に" (easily) in Figure 5, will be orphan. It should also be noted that an effect phrase is considered as a pair of an attribute and a value phrase in this approach. However, some effect phrases contain two or more attributes or effects such as the effect phrase "wide dynamic range from 0.1 mG (milligauss) to 50 G" where "dynamic range" represents an attribute and "wide" and "0.1 mG (milligauss) to 50 G" represent values. An enhanced method will be developed in future work to deal with this type of effect phrases.

In Japanese text, some nominals such as "向上" (improvement) or "容易" (ease) often form verbs (e.g. "向上する" (improve)) and adjectives (e.g. "容易だ" (easy)) by combining these nominals with inflections such as "する", "だ." Since those nominals themselves can form value phrases as well as the verbs and adjectives derived from them, the nominal verbs and adjectives are often tagged only on their nominal parts (e.g. "向上," "容易") in the training data. By considering this characteristic of the training data when applying all of the word labels against the plain text, the labels of nominal verbs such as "向上する" (improve) and nominal adjectives such as "容易だ" (easy) are actually applied only to strings representing nominal parts (e.g. "向上," "容易") without their inflections (e.g. " する," "だ").

3. EVALUATIONS

The regularization parameters of the CRF-based word labeler was estimated with 9 sets of features using the training data (Table 4). Since we found and fixed a bug in the dependency structure features after the formal-run, the revised values obtained after the bug-fix are also reported with asterisks (*) as well as the original values obtained in the formal-run. Values of the regularization parameters were determined for each of those feature sets to maximize the average F-values of 5-fold cross-validation using the training data. Since the word labeler does not infer labels representing effect phrases, the F-values are calculated by excepting effect phrases. The cross-validation result showed that the proposed dependency structure feature (dep) outperforms the baseline dependency structure feature (dep_onebest). Based on this result, we employed the proposed dependency structure feature (dep) in the formal-run.

In the formal-run, the 8 sets of features were used for both of

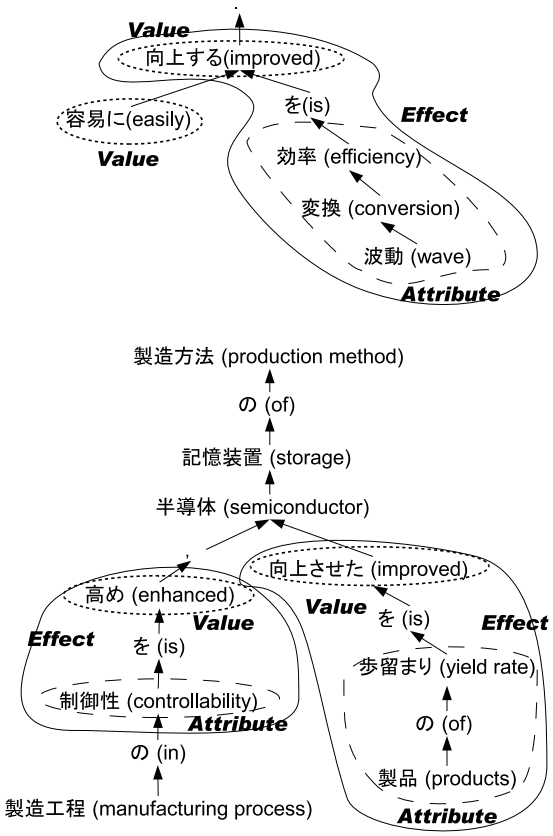

Figure 5: Overview of the effect annotation: Phrases enclosed in dashed lines represent attributes, phrases enclosed in dotted lines represent values, and phrases enclosed in solid lines represent effects.

the patent and paper data. Table 5 shows the recalls, precisions and F-values for each tag type and the averages. Again, the values with asterisks are revised results obtained after fixing a bug in the dependency structure features.

None of the feature sets were able to annotate the technology phrases in the patent titles. This may be because no technology phrases appear in the patent titles in the training data. Six out of the 9 technology phrases in the patent titles appeared at the beginning of the titles (that is, at the beginning of their sentences), which rarely occurs for the technology phrases in the training data. Therefore, the annotation systems were unable to learn the characteristics of those phrases.

The section feature (sec) contributed strongly to the annotations on the patent data. By adding this feature to the baseline feature set, the technology annotation performance for abstracts was improved by 4%, the attribute annotation performance was improved by 8%, the value annotation performance was improved by 8%, and the average performance was improved 6% for the F-values. These results showed that the section information is useful to find effect phrases in the patent data.

The dependency structure feature (dep) was also effective for the annotation of attributes (4% improvement in the F-value) and values (2% improvement in the F-value) for the patent data. This may be because the sentences in the patent data tend to be much longer than the sentences in the paper data, and so there should be many

Table 4: 9 feature sets and the average F-values of 5-fold cross-validation using the training data: Note that the F-value of ID=7 is evaluated by using the mixed data of both patent and paper data. (Values with asterisks are the revised results after the formal-run)

ID	Feature set	F. (patent)	F. (paper)
1	baseline	0.363	0.244
2	1 + sec	0.414	0.244
3	2 + pos	0.428	0.272
4	3 + ipc	0.426	0.276
5	4 + ec_p	0.424	0.282
6	5 + dep	0.439 **0.448**∗	0.281 **0.298**∗
6'	5 + dep_onebest	0.417 0.428∗	0.279 0.286∗
7	6 + feda	0.492 0.496∗	
8	4 + ec_w	0.427	0.279

parse errors when analyzing the dependency structures in the patent text. Since this feature uses the expected phrase distances in the dependency trees instead of the deterministic distances, the feature should be robust against the parse errors.

In contrast, the relative position feature (pos) contributed for the annotation of the paper data, particularly for the attribute annotations (4% improvement in the F-values) and the value annotations (4% improvement in the F-value). These results showed that the relative position feature reflects the characteristics that the effect phrases are likely to appear in the last parts of the abstracts.

The domain adaptation feature (feda) also contributed for the annotations for the paper data. Since the texts in the paper data are relatively shorter than the texts in the patent data, the features learned from the patent data will handle features that do not appear in the paper data.

The technical field feature (ipc) using the IPC codes given to the documents contributed towards slightly improving the precision of the technology annotations for the patent abstracts. The result showed the possibility of improving the technology annotations by implementing the technical field feature, but more detailed investigation is needed to understand why the feature did not contribute towards the technology annotations for the paper data.

Unfortunately, the effect context features (ec_p and ec_w) were not able to show its effectiveness in the formal-run results, even they showed slight improvement in the cross-validation using the training data (Table 4). Further improvement would be needed for the reliability function and the way of determining feature values based on the extracted effect phrases from unlabeled documents.

4. CONCLUSIONS

In this report, approaches taken in the Trend Map Creation Subtask in the NTCIR-8 Patent Mining Task were described. The information extraction task was regarded as a word sequence labeling problem, and approached by using CRF model to infer the IOB labels for each word. After applying several types of features, the formal-run result showed that the section feature (sec) and the dependency structure feature (dep) were effective for technology and effect phrase extraction from the patent documents, and the relative position feature (pos) and the domain adaptation feature (feda) were effective for the papers.

Future work will include investigations of the usage of unlabeled documents and the utilization of technical field information given in the original documents.

Acknowledgement

The authors thank Hiroshi Kanayama for many helpful suggestions and valuable comments.

5. REFERENCES

[1] H. Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 256–263, 2007.

[2] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, 2001.

[3] S. Lee and K. S. Choi. Reestimation and best-first parsing algorithm for probabilistic dependency grammar. In *Proceedings of the Fifth Workshop on Very Large Corpora*, pages 41–55, 1997.

[4] H. Nanba, A. Fujii, M. Iwayama, and T. Hashimoto. Overview of the patent mining task at the ntcir-8 workshop. In *Proceedings of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, 2010.

[5] R. Nishiyama. Technical issue extraction from patent disclosure. In *Proceedings of the 16th Annual Meeting of the Association of Natural Language Processing (in Japanese)*, pages 82–85, 2010.

[6] E. F. Tjong Kim Sang and J. Veenstra. Representing Text Chunks. In *Proceedings of the 9th Conference on European Chapter of the Association for Computational Linguistics*, pages 173–179, 1999.

[7] Y. Unno and Y. Tsuboi. Distance metric based on dependency marginal probabilities. In *Proceedings of the 16th Annual Meeting of the Association of Natural Language Processing (in Japanese)*, pages 23–26, 2010.

[8] World Intellectual Property Organization (WIPO). International patent classification (IPC). http://www.wipo.int/classifications/ipc/en/ (as of: 2010/03/17).

Table 5: Results in the formal run (Japanese patent and research paper data; Values with asterisks are the revised results after the formal-run.)

		Technology (Title)			Technology (Abst.)		
System ID	Feature set	Rec.	Prec.	F.	Rec.	Prec.	F.
patent_TRL1_2	baseline	0.000(0/9)	0.000(0/7)	0.000	0.335(248/740)	0.410(248/605)	0.369
patent_TRL2_2	1+sec	0.000(0/9)	0.000(0/3)	0.000	0.388(287/740)	0.432(287/664)	0.409
patent_TRL3_2	2+pos	0.000(0/9)	0.000(0/1)	0.000	0.378(280/740)	0.438(280/639)	0.406
patent_TRL4_2	3+ipc	0.000(0/9)	0.000(0/1)	0.000	0.378(280/740)	0.445(280/629)	0.409
patent_TRL5_2	4+ec_p	0.000(0/9)	0.000(0/1)	0.000	0.374(277/740)	0.445(277/622)	0.407
patent_TRL6_2	5+dep	0.000(0/9)	0.000(0/2)	0.000	0.399(295/740)	0.407(295/725)	0.403
		0.000(0/9)*	0.000(0/1)*	0.000*	**0.418 (309/740)***	**0.487 (309/725)***	**0.449***
patent_TRL7_2	6+feda	0.000(0/9)	0.000(0/2)	0.000	0.400(296/740)	0.402(296/736)	0.401
		0.000(0/9)*	0.000(0/2)*	0.000*	0.396(293/740)*	0.423(293/693)*	0.409
patent_TRL8_2	4+ec_w	0.000(0/9)	0.000(0/1)	0.000	0.374(277/740)	0.449(277/617)	0.408
paper_TRL1_1	baseline	0.290(27/93)	0.871(27/31)	0.435	0.188(68/362)	0.602(68/113)	0.286
paper_TRL2_1	1+sec	0.290(27/93)	0.871(27/31)	0.435	0.188(68/362)	0.607(68/112)	0.287
paper_TRL3_1	2+pos	0.301(28/93)	0.875(28/32)	0.448	0.191(69/362)	0.570(69/121)	0.286
paper_TRL4_1	3+ipc	0.301(28/93)	**0.903 (28/31)**	0.452	0.191(69/362)	0.590(69/117)	0.288
paper_TRL5_1	4+ec_p	0.280(26/93)	0.867(26/30)	0.423	0.193(70/362)	0.574(70/122)	0.289
paper_TRL6_1	5+dep	**0.323 (30/93)**	0.769(30/39)	0.455	0.196(71/362)	0.617(71/115)	0.298
		0.323 (30/93)*	0.811(30/37)*	0.462*	0.224(81/362)*	0.659(81/123)*	0.334*
paper_TRL7_1	6+feda	**0.323 (30/93)**	0.811(30/37)	0.462	0.207(75/362)	0.605(75/124)	0.309
		0.323 (30/93)*	0.857(30/35)*	**0.469***	**0.232 (84/362)***	**0.700 (84/120)***	**0.349***
paper_TRL8_1	4+ec_w	0.301(28/93)	**0.903 (28/31)**	0.452	0.182(66/362)	0.532(66/124)	0.272

		Attribute (Abst.)			Value (Abst.)		
System ID	Feature set	Rec.	Prec.	F.	Rec.	Prec.	F.
patent_TRL1_2	baseline	0.229(116/506)	0.472(116/246)	0.309	0.407(193/474)	0.654(193/295)	0.502
patent_TRL2_2	1+sec	0.330(167/506)	0.490(167/341)	0.394	0.517(245/474)	0.664(245/369)	0.581
patent_TRL3_2	2+pos	0.332(168/506)	0.532(168/316)	0.409	0.504(239/474)	0.701(239/341)	0.587
patent_TRL4_2	3+ipc	0.330(167/506)	0.535(167/312)	0.408	0.502(238/346)	0.688(238/346)	0.580
patent_TRL5_2	4+ec_p	0.322(163/506)	0.553(163/295)	0.407	0.494(234/474)	0.705(234/332)	0.581
patent_TRL6_2	5+dep	0.403(204/506)	0.523(204/390)	0.455	0.540(256/376)	0.681(256/376)	0.602
		0.381(193/506)*	**0.594 (193/325)***	**0.465***	0.494(234/474)*	**0.752 (234/311)***	0.596*
patent_TRL7_2	6+feda	**0.405 (205/506)**	0.519(205/395)	0.455	0.542(257/474)	0.676(257/380)	0.602
		0.403(204/506)*	0.533(204/383)*	0.459*	**0.559 (265/474)***	0.705(265/376)*	**0.624***
patent_TRL8_2	4+ec_w	0.332(168/506)	0.545(168/308)	0.413	0.502(238/474)	0.686(238/347)	0.580
paper_TRL1_1	baseline	0.111(33/296)	0.367(33/296)	0.171	0.153(45/294)	0.372(45/121)	0.217
paper_TRL2_1	1+sec	0.108(32/296)	0.376(32/85)	0.168	0.146(43/294)	0.364(43/118)	0.209
paper_TRL3_1	2+pos	**0.135 (40/296)**	0.404(40/99)	**0.203**	0.184(54/294)	0.394(54/137)	0.251
paper_TRL4_1	3+ipc	0.115(34/296)	0.425(34/80)	0.181	0.184(54/294)	0.446(54/121)	0.260
paper_TRL5_1	4+ec_p	0.115(34/296)	0.436(34/78)	0.182	0.170(50/294)	0.467(50/107)	0.249
paper_TRL6_1	5+dep	0.108(32/296)	**0.471 (32/68)**	0.176	0.160(47/294)	**0.618 (47/76)**	0.254
		0.101(30/296)*	0.395(30/76)*	0.161*	0.187(55/294)*	0.539(55/102)*	0.278*
paper_TRL7_1	6+feda	0.122(36/296)	0.450(36/80)	0.191	0.163(48/294)	0.539(48/89)	0.251
		0.118(35/296)*	0.443(35/79)*	0.187*	**0.194 (57/294)***	0.600(57/95)*	**0.293***
paper_TRL8_1	4+ec_w	0.118(35/296)	0.376(35/93)	0.180	0.190(56/294)	0.364(56/154)	0.250

		Effect (Abst.)			Avg. (Tech., Att. and Val.)		
System ID	Feature set	Rec.	Prec.	F.	Rec.	Prec.	F.
patent_TRL1_2	baseline	0.137(67/489)	0.493(67/136)	0.214	0.322(557/1729)	0.483(557/1153)	0.387
patent_TRL2_2	1+sec	0.209(102/489)	0.477(102/214)	0.290	0.404(699/1729)	0.508(699/1377)	0.450
patent_TRL3_2	2+pos	0.198(97/489)	0.487(97/199)	0.282	0.397(687/1729)	0.530(687/1297)	0.454
patent_TRL4_2	3+ipc	0.196(96/489)	0.500(96/192)	0.282	0.396(685/1729)	0.532(685/1288)	0.454
patent_TRL5_2	4+ec_p	0.202(99/489)	0.553(99/179)	0.296	0.390(674/1729)	0.539(674/1250)	0.453
patent_TRL6_2	5+dep	0.256(125/489)	0.517(125/242)	0.342	0.437(755/1729)	0.506(755/1493)	0.469
		0.237(116/489)*	**0.595 (116/195)***	0.339*	0.426(736/1729)*	**0.579 (736/1272)***	**0.491***
patent_TRL7_2	6+feda	**0.264 (129/489)**	0.531(129/243)	0.352	0.438(758/1729)	0.501(758/1513)	0.468
		0.260(127/489)*	0.550(127/231)*	**0.353***	**0.441 (762/1729)***	0.524(762/1454)*	0.479*
patent_TRL8_2	4+ec_w	0.190(93/489)	0.489(93/190)	0.274	0.395(683/1729)	0.537(683/1273)	0.455
paper_TRL1_1	baseline	0.048(14/293)	0.483(14/29)	0.087	0.166(173/1045)	0.487(173/355)	0.247
paper_TRL2_1	1+sec	0.044(13/293)	0.481(13/27)	0.081	0.163(170/1045)	0.491(170/346)	0.244
paper_TRL3_1	2+pos	0.055(16/293)	0.500(16/32)	0.098	0.183(191/1045)	0.491(191/389)	0.266
paper_TRL4_1	3+ipc	0.044(13/293)	0.464(13/28)	0.081	0.177(185/1045)	0.530(185/349)	0.265
paper_TRL5_1	4+ec_p	0.041(12/293)	0.429(12/28)	0.075	0.172(180/1045)	0.534(180/337)	0.260
paper_TRL6_1	5+dep	**0.061 (18/293)**	**0.600 (18/30)**	**0.111**	0.172(180/1045)	0.604(180/298)	0.268
		0.0510(15/293)*	0.441(15/34)*	0.092*	0.188(196/1045)*	0.580(196/338)*	0.283*
paper_TRL7_1	6+feda	0.051(15/293)	0.500(15/30)	0.093	0.181(189/1045)	0.573(189/330)	0.275
		0.055(16/293)*	0.457(16/35)*	0.098*	**0.197 (206/1045)***	**0.626 (206/329)***	**0.300***
paper_TRL8_1	4+ec_w	0.044(13/293)	0.406(13/32)	0.080	0.177(185/1045)	0.460(185/402)	0.256