A Thesis Presented to Graduate School of Engineering
at the University of Tokyo for the Degree of Master in 2003

Text Classification with a Polysemy Considered Feature Set

Department of Electrical Engineering
Graduate School of Engineering
the University of Tokyo

Jun Araki

Supervisor D. Eng. Masaya Nakayama

Table of Contents

List of Figures

List of Tables

Abstract

As we store and distribute a large amount of computerized text, we have an important issue about how we extract useful data effectively from the text data. For this reason, the techniques for classifying text automatically with computers have attached attention.

Generally, in a field of text classification, we use a model called Vector Space Model(VSM), in which we map a document into a point in a vector space with multiple dimension that has axes based on feature sets of keywords to characterize categories. In the past, lots of different attempts to extract words with highly evaluated values based on some measures, such as mutual information between categories and words, have been made for selection of feature words which characterize categories in text classification. However, some words are polysemous ones which have not a single meaning but multiple meanings, and therefore in the case of those polysemous words, there are documents which belong to different categories from the one to be intended, which causes problems for classification.

In our research, we consider polysemous words as features with a risk factor for classification, and propose a method that we determine whether each feature word is the risk factor or not, using mutual information as a measure for feature selection, and disambiguate feature sets by removing features judged as risk factors. We compare classifying results with our method to the ones with an existing method, and evaluate its efficiency by using the Reuters-21578 corpus as the target data for classifying.

Chapter 1

Introduction

1.1 A Problem about Polysemy in Text Classification

As we store and distribute a large amount of computerized text, we have an important issue about how we extract useful data effectively from the text data. It is possible that one way to solve the issue is to classify text data into some categories decided in advance. This is because we can search essential data from them more easily if we group a lot of data into some categories given in advance, and therefore, we can say that categorizing data is one of ways to help us access the data.

Generally, in the field of text classification, we use a model called Vector Space Model(VSM), in which we map a document into a point in a multidimensional vector space with its axes based on feature sets of keywords to characterize categories. It is preferable that we use as many features as possible in order to classify an enormous amount of documents with high accuracy, but at the same time we need to reduce the number of the features, taking into account disadvantages of over-learning and calculation time in learning. In the past, because of this necessity for the reduction of features, different attempts to extract words with highly evaluated values based on some measures, such as mutual information between categories and words, have been made for selection of feature words which characterize categories in text classification.

However, some words are polysemous ones which have not a single meaning but multiple meanings. These words can have high mutual information not only with the categories to be intended but also with the ones not to be intended, so they are likely to be selected as features. If we select polysemous words as features, we could get only low classification accuracy because the classifiers often misunderstand documents including those words due to their other meanings related to other categories not to be intended, and that causes increase of incorrect classification.

1.2 Related Research

In text classification, some methods to solve the problem about polysemy of words have been already suggested. Yuasa et al. suggested a method in which they defined frequencies of nouns which co-occurred in a certain article as a noun co-occurrence distribution, and made standardized vectors of each category based on the noun co-occurrence distribution vectors, and classified articles by comparing unknown ones with the vectors[27]. They conducted experiments in which they classified articles into five categories by using article data of Asahi Shimbun in 1987, which is one

of major Japanese newspapers, and reported that they obtained accuracy rates(recall) of more than 80 percents on average.

Meanwhile, a method to utilize thesauri organizing words in a view of their meanings has been also suggested[26]. In this method, they made up feature spaces of words and their sense categories provided by the thesauri. They determined meanings of polysemous words and disambiguated by using relevance ratio between feature words and categories, which they calculated from an error-driven algorithm, called WINNOW, and the value of χ^2. They conducted experiments in which they classified articles into 13 categories, and reported that the disambiguation based on the value of χ^2 had an effect of improving classification accuracy which had been deteriorated by use of the thesauri.

In addition to them, there is also a method called Latent Semantic Indexing(LSI), which deploys documents in a conceptual space by using the multivariate analysis technique called Singular Value Decomposition(SVD)[5]. While we make up a space with words its axes in VSM, we make up a semantic space in LSI with less axes merged from the axes in VSM by the SVD techniques, and are able to measure similarities between documents. By using the SVD method, documents which share sets of frequently co-occurring words come to have analogous relationships with one another in the latent semantic space.

1.3 Composition of the Thesis

In our research, we consider polysemous words as features with a risk factor for classification, and propose a method that we determine whether each feature word is the risk factor or not, utilizing mutual information as a measure for feature selection, and disambiguate feature sets by removing features judged as risk factors. By using the Reuters-21578 corpus as the target data for classifying, we compare the classification results with our method to the ones with an existing method, and evaluate its efficiency.

The composition of this paper is the following. In next chapter, Chapter 2 , we explain the background of text classification research, the definition of text classification, and some feature selection methods including a measure for evaluation of mutual information. We also explain a machine learning algorithm, Support Vector Machine(SVM), which we utilized as a classifier in our research, and several ways for evaluating the classification accuracy. In Chapter 3 , we go into detail about our proposal method for reducing ambiguity. We describe the experimental environment and the way to conduct experiments in this work in Chapter 4 , and report the experimental results in Chapter 5 . We consider what the result indicates, and evaluate our proposal method in Chapter 6 . We describe future works in Chapter 7 , and conclude this paper in Chapter 8 .

Chapter 2

Text Classification

2.1 Techniques for Text Classification

We throw out needless information. The essence of the information management which contemporary individuals or organizations are facing is in our works to classify things into a category of "necessary" or of "unnecessary." Techniques for classifying texts in an automatic way play a very significant role on the today's Internet where enormous and diverse contents are distributed, as it is often expressed as "flood information."

For instance, the web content filters, which prevent children from accessing objectionable web sites such as one with violence or pornography, classify web sites into "harmful" and "harmless." And spam filters, which shut out unwanted e-mail messages sent to general public unilaterally for advertising, soliciting and so forth, classify e-mail messages into "usual" and "spam."

The content filtering function of the web and e-mail has been the one that should be provided normally in both of the Internet service providers(ISPs) and client software such as browsers and mailers, in response to social requirements. In many cases, these content filtering is implemented in a combination of techniques for managing lists of unwanted information sources, called black lists, and techniques for filtering based on the contents, called text classification techniques.

The Internet search engines with directory structures represented by Yahoo![34] have an advantage that people are able to find a number of web pages through categories about the fields they interested in, because editors register the pages into the categories which they judged as the most suitable for the pages. However, these search engines have also a disadvantage that they have only a little absolute amount of information with them, because web pages on the Internet has been increasing so enormously now that they have their own limits of registering web pages manually[35]. Therefore, if text classification techniques were able to classify data with few practical problems, they would save labor of editors registering web pages into categories, and we apply them to classification for enormous amount of web pages, even if the techniques cannot achieve the finesse of the editors.

Going back to a historical transition of text classification techniques, a knowledge engineering approach, that is, a method to write rules for classification manually was the mainstream until the late 1980s. In the 1990s, however, a large amount of text data became available and computers grew in performance prominently, and consequently, it changed into the mainstream to use a machine learning approach, the method to make classifiers automatically from text information

given category labels manually. This is because it has more advantages in terms of cost and maintenance. In addition, researchers have recently applied the cutting edge algorithms in machine learning, such as Support Vector Machine(SVM), AdaBoost and so forth, to text classification problems one after another, and as the result, some common benchmarks for classification is now available for us to weigh effectiveness of different learning theories.

2.2 Text Classification Methods

Various methods are proposed in text classification according to ways to express documents and build up classifiers. There are such techniques as TF-IDF, information gain, mutual information, chi-square tests in terms of the difference according to the ways to express documents, that is, selection of feature words as elements of document vectors, weighting in document vectors, and reduction of dimensions. As for the rest, there is also a method called Latent Semantic Indexing(LSI) that maps document vectors into a completely different space by using the multivariate analysis technique called Singular Value Decomposition(SVD)[5], as mentioned in Section 1.2 .

On the other hand, building up classifiers is a very classic supervised inductive learning problem, and researchers have examined most of main machine learning algorithms such as Naive Bayes, decision trees, decision lists, the k-nearest neighbor algorithm, online algorithms, maximum entropy modeling, SVM, boosting and so on. Among these machine learning algorithms, SVM and AdaBoost show the best performance at the present time.

We put those text classification methods all together in Tab. 2.1, based on the difference according to the ways to express documents and the ways to build up documents.

Tab. 2.1 Text Classification Methods

<table>
<tr><td rowspan="5">(By the difference according to the ways to express documents)</td><td rowspan="4">(By the difference according to the ways for feature selection and weighting)</td><td>TF-IDF</td></tr>
<tr><td>Information Gain</td></tr>
<tr><td>Mutual Information</td></tr>
<tr><td>Chi-square test</td></tr>
<tr><td colspan="2">LSI: Latent Semantic Indexing</td></tr>
<tr><td rowspan="7">(By the difference according to the ways to build up documents)</td><td colspan="2">Naive Bayes</td></tr>
<tr><td colspan="2">Decision Trees, Decision List</td></tr>
<tr><td colspan="2">k-Nearest Neighbor Algorithm</td></tr>
<tr><td colspan="2">Online Algorithm</td></tr>
<tr><td colspan="2">Maximum Entropy Modeling</td></tr>
<tr><td colspan="2">Support Vector Machines</td></tr>
<tr><td colspan="2">Boosting</td></tr>
</table>

2.3 Workflow in Text Classification Processing

We show an outline drawing about a whole flow of general text classification process in Fig. 2.1. A procedure of general text classification is as indicated below.

1. Data Selection

We extract training data and test data necessary for classification process from original data.

2. Preprocessing

In general, since training data and test data which we just extract include noises, we usually remove them. Especially in text classification, we carry out preprocessing for the data, which includes a removal of disabled words, a conversion from a plural form to a singular form, and a conversion derivative words into ones with an original form.

3. Feature Selection

We carry out feature selection from preprocessed training data. In this process, there are some methods for feature selection due to difference of ways to express documents in Tab. 2.1.

4. Making Vectors

We convert all the test data into document vectors based on selected features.

5. Text Classification

We apply text classification algorithms to the document-vectorized test data, and classify documents into categories. In this process, there are some methods for text classification due to difference of ways to build up documents in Tab. 2.1, and different machine learning algorithms are proposed.

6. Evaluation

We determine classification accuracy by using some measure for evaluation.

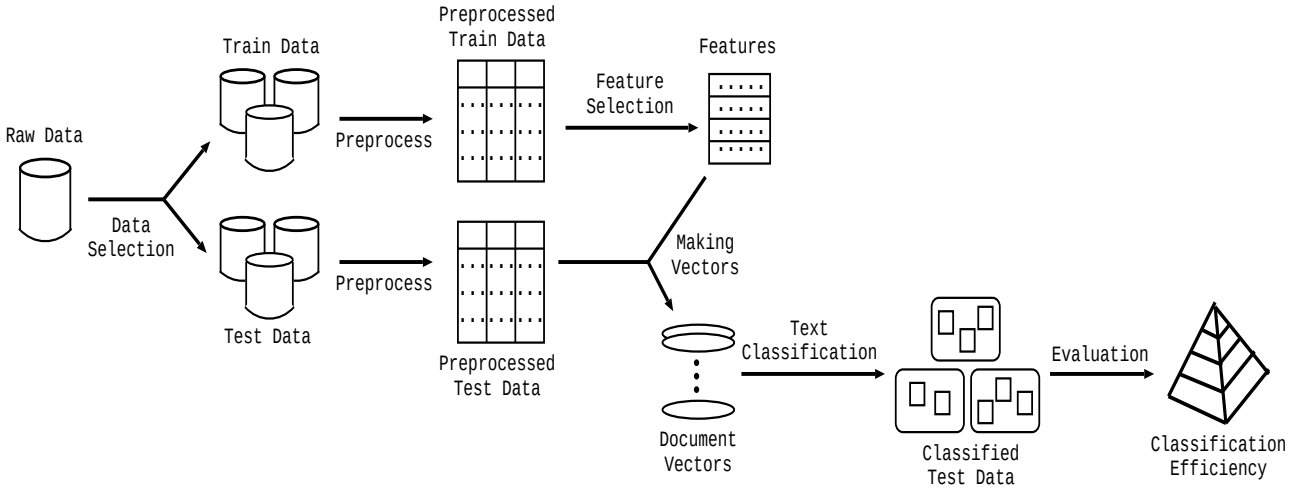


Fig. 2.1 Flow diagram of text classification process

2.4 Definition of Text Classification

In our research, we define text classification as classifying documents into more than two categories given in advance. Generally in text classification, we describe a document with a multi-dimensional vector as follows.

$$\mathbf{x} = (x_1, x_2, \ldots, x_d) \tag{2.1}$$

In this equation, Eq.(2.1), d indicates the dimension number of the vector. Each element in the document vector can be a boolean value that shows whether the word occurs in the document or not, or can be a real value that weighs based on an appropriate means such as TF-IDF* and so on. We use the former; each element is 1 in the case the word occurs, and 0 in the other case.

For instance, when we describe the following two documents using vectors with existence of four words, "love", "reverse", "Parliament"and "home-run," we can get the document vectors, $\mathbf{x}_1$ and $\mathbf{x}_2$, shown in Fig. 2.2.

Doc.1: "The home-run burst out and reversed the game in the last round."

Doc.2: "The power of influence of the governing and opposition parties were reversed in Parliament."

	"love"	"reverse"	"Parliament"	"home-run"
Doc.1($\mathbf{x}_1$)	0	1	0	1
Doc.2($\mathbf{x}_2$)	0	1	1	0

Fig. 2.2 Description of document vectors

In the example noted above, we describe the feature of the documents with four-dimensional vectors. Actually, it is preferable to use as many features as possible, for example several tens of thousands of features, in order to classify a huge variety of documents with a high degree of accuracy. However, it is necessary to reduce features to the extent from several hundreds to several thousands because of problems such as over-training and computation time[21]. Therefore, some methods have been proposed for feature selection using various evaluation criteria such as word frequencies, document frequencies, mutual information and so on[15]. We will enter into more details about feature selection at next section.

We give each document a label y for the category which the document belongs to. The number of categories can be two, which is the easiest case and means whether the document belongs to one category or not, and can be more than that. And with respect to the label, there are two cases; one case is that one document has only singular label (belongs to only one category) and the other case is that one document has plural labels (belongs to multiple categories). In the case that one

*See Section 2.5.1 .

document has plural labels, we often solve a multi-label classification problem by combinations of plural binary classifiers, in general.

Under the settings mentioned above, given a collection of training data,

$$S = (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_l, y_l) \tag{2.2}$$

we can formulate a problem of making classifiers for text classification into finding the function $f(\mathbf{x})$ that minimizes the counts which predicted labels differ from correct labels;

$$\sum_{i=1}^{l} P(f(\mathbf{x}_i) \neq y_i) \tag{2.3}$$

In those equations, we define l as the number of training documents, and $P(X)$ as a function that returns an integer value according to X in the following Eq.(2.4).

$$P(X) = \begin{cases} 1 & \text{if X is true} \\ 0 & \text{if X is false} \end{cases} \tag{2.4}$$

In addition, in case of classification into two categories, we set category labels as $y \in \{-1, +1\}$.

2.5 Feature Selection

When we use a large amount of features which are keywords of each category in order to achieve high classification accuracy, the classification is subject to over-training and enlargement of computation time. Therefore, we use feature selection methods to select only the features that accentuate the classification in the aim of reducing dimensionality of feature spaces. There are document frequency, mutual information, information gain, chi-square tests as evaluation criteria used in feature selection. A paper[15] reports that they obtained almost the same accuracy among document frequency, information gain and chi-square tests, but lower accuracy about mutual information as they reduce dimensionality, and therefore document frequency was superior because it enabled them to obtain relatively high accuracy with a little amount of calculation.

In our research, we used mutual information between categories and words as a feature selection method.

2.5.1 Document Frequency

Document Frequency(DF) is a number of documents that include one specific word. We eliminate from a feature space the words whose document frequencies are lower than a threshold value which we decided beforehand. The basic idea is that rare words are not useful for predicting their categories, and not have much effect on the whole classification.

In recent years, the TF-IDF method has been the mainstream as well as Term Frequency(TF). Term Frequency is the number of words within a document. This concept is based on the idea that if a word T is seen at a document D in a high frequency, T characterizes D more precisely. However, it is obvious that for example, a conjunction, “and,” is seen at any documents in a high frequency,

so the word does not characterize particular documents. Therefore, the words appearing in a small number of documents among a collection of documents that is a collection of search targets are likely to be adequate for characterising particular documents. We define the paucity of document frequency as an inverse number of DF, IDF. The inverse number of document frequency as to T, IDF, can be normalized by the total number of documents and shown in the following equation.

$$IDF(T) = \log \frac{N}{n_T} \tag{2.5}$$

In this equation, n_T is the number of documents where the word T occurs. The value of *TF-IDF* is defined as a multiplication of TF and IDF.

$$TFIDF(T, D) = f_{TD} \times \log \frac{N}{n_T} \tag{2.6}$$

In this equation, f_{TD} is the number of times that the word T occurs in the document D.

2.5.2 Information Gain

Information Gain(IG) is to measure information for category prediction by knowing that a word occurs in a document or not. When we express a set of categories as $\{c_i\}_{i=1}^m$, we can define information gain of a word T as the following expression.

$$\begin{aligned} IG(T) &= -\sum_{i=1}^{m} P_r(c_i)\log P_r(c_i) \\ &\quad + P_r(T)\sum_{i=1}^{m} P_r(c_i|T)\log P_r(c_i|T) \\ &\quad + P_r(T)\sum_{i=1}^{m} P_r(c_i|\bar{T})\log P_r(c_i|\bar{T}) \end{aligned} \tag{2.7}$$

In this expression, $\bar{T}$ is a word that is not T.

If we use this measure for text classification, we calculate information gain of each word in given training data. Based on the result of the calculation, we remove the words that exceed a certain threshold value decided beforehand. If we describe the number of training documents as N and the average length of them as V, we need time with the order of $O(N)$ and capacity of memory with the order of $O(VN)$ for calculating conditional probabilities of categories when words are given. Thus, we need time with order of $O(Vm)$ for calculating the entropy.

2.5.3 Mutual Information

Mutual Information is an index value that shows relevance between categories and words. We define Mutual Information MI between a word T and a category C in a collection of documents as the following equation;

$$MI(T, C) = \sum_{t\in\{T,\bar{T}\}} \sum_{c\in\{C,\bar{C}\}} P(t, c)\ \log \frac{P(t, c)}{P(t)P(c)} \tag{2.8}$$

In this equation, $\bar{T}$ is a set of words that does not include T, and $\bar{C}$ a set of categories that does not include C. We define other values as follows;

$P(t)$: a rate of articles that include a set of words t in all of the articles

$P(c)$: a rate of articles that belong to a set of categories c in all of the articles

$P(t, c)$: a rate of articles that include a set of words t and belong to a set of categories c in all of the articles

We show a table about the number of co-occurrence of a word T and a category C in Tab. 2.2. For instance, a shows the number of documents which include a word T and belong to a category C. When we give a total of documents $N = a + b + c + d$, we can redefine MI as the following equation;

Tab. 2.2 Number of co-occurrence of a word T and a category C

	C occurs	C does not occur
T occurs	a	b
T does not occur	c	d

$$\begin{aligned} MI(t,c) &= \sum_{t\in T}\sum_{c\in C} P(t,c)\,\log\frac{P(t,c)}{P(t)P(c)} + \sum_{t\in T}\sum_{c\in \bar{C}} P(t,c)\,\log\frac{P(t,c)}{P(t)P(c)} \\ &\quad + \sum_{t\in \bar{T}}\sum_{c\in C} P(t,c)\,\log\frac{P(t,c)}{P(t)P(c)} + \sum_{t\in \bar{T}}\sum_{c\in \bar{C}} P(t,c)\,\log\frac{P(t,c)}{P(t)P(c)} \\ &= \frac{a}{N}\log\frac{aN}{(a+b)(a+c)} + \frac{b}{N}\log\frac{bN}{(a+b)(b+d)} + \frac{c}{N}\log\frac{cN}{(c+d)(a+c)} + \frac{d}{N}\log\frac{dN}{(c+d)(b+d)} \end{aligned} \tag{2.9}$$

Mutual Information shown in Eq.(2.9) is normalized to run from 0 through 1. Mutual Information becomes a small value when a frequency of a word T appearance is unbiased among a category C and other categories. It gets 0 when the word T and the category C are independent of each other. What we mean by being independent here is that we have equality of $a = b$ and $c = d$, or $a = c$ and $b = d$ in Tab. 2.2. Conversely, it gets 1 when the word T and the category C are dependent on each other. What we mean by being dependent here is that we have equality of $a = d$ and $b = c = 0$, or $a = d = 0$ and $b = c$.

We need computation time of $O(Vm)$ order for calculating MI as well as Information Gain, IG. The weakness of Mutual Information is that it is more likely to be affected by insignificant words. This is because the values $P(T|C)$ of these infrequent words in the following equation have tendency to be more larger than ones of general words.

$$MI(T, C) = \log P(T|C) - \log P(T) \tag{2.10}$$

2.5.4 Chi-square Tests

In chi-square tests, we measure independency between words and categories. Using Tab. 2.2 we described at the section of Mutual Information, we define the criteria as follows:

$$\chi^2(t,c) = \frac{N \times (AD - BC)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)} \tag{2.11}$$

In this equation, N stands for an amount of documents as we mentioned above. $\chi^2(t,c)$ also gets 0 when the word t and the category c are independent. Computation time in chi-square tests has the same order as ones in MI and IG.

It is known that chi-square tests as to infrequent words result in lack of reliability because we cannot compare their distribution with one of chi-square distribution precisely.

2.6 Support Vector Machine

Support Vector Machine(SVM) derives from a statistical pattern recognition method, "Optimal Separating Hyperplane," proposed by Vapnik, et al. in 1960s. It became evident that SVMs has a superior generalization capability in a number of classification problems that require input attributes with high dimension[2, 4, 7]. It was also shown that it has a competitive classification capability in text classification tasks[6, 8].

In this section, we state the way to determine separating hyperplanes in linear SVMs, non-linear SVMs using kernel functions, and explain a concept of the Structural Risk Minimization. And we describe features of SVMs and its current trend at the last.

2.6.1 Decision of Separating Hyperplanes

SVMs are binary classifiers to classify data with the hyperplanes that maximize margins between positive and negative examples, which we separate training data into.

We show a conceptual diagram of SVMs in Fig. 2.3. In this figure, we assume a binary classification problem, and define input data as document vectors $\mathbf{x}$ and output data as class labels of positive and negative examples, respectively $y = 1$ or $y = -1$. As shown in Fig. 2.3, data of positive and negative examples are divided by the following two hyperplanes:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x_i} \geq 1 & \text{if } y_i = 1 \\ \mathbf{w} \cdot \mathbf{x_i} \leq -1 & \text{if } y_i = -1 \end{cases} \tag{2.12}$$

We apply the word a margin to the distance the boundary surface of positive examples closest to negative ones, $\mathbf{w} \cdot \mathbf{x} + b = 1$, and the other boundary surface of negative examples closest to positive ones, $\mathbf{w} \cdot \mathbf{x} + b = -1$. In this case, the separating hyperplane that we would like to set up is $\mathbf{w} \cdot \mathbf{x} + b = 0$. We describe training data on $\mathbf{w} \cdot \mathbf{x} + b = \pm 1$ as support vectors.

The distance d from the hyperplane to the point $\mathbf{x}$ is shown as follows:

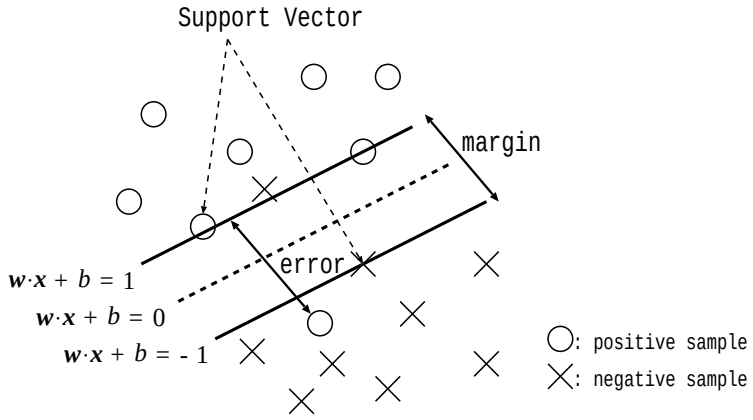


Fig. 2.3 Conceptual diagram of SVMs

$$d(\mathbf{w}, b; \mathbf{x}) = \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{||\mathbf{w}||} \tag{2.13}$$

So the margin ρ is described as follows:

$$\rho = \min_{\mathbf{x}_i; y_i=1} \frac{|\mathbf{w} \cdot \mathbf{x}_i + b|}{||\mathbf{w}||} + \min_{\mathbf{x}_i; y_i=-1} \frac{|\mathbf{w} \cdot \mathbf{x}_i + b|}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||} \tag{2.14}$$

In this equation, y_i is a category label of a document vector $\mathbf{x}_i$, and can take the value of 1 in the case where $\mathbf{x}_i$ is a positive example, and -1 in the case where $\mathbf{x}_i$ is a negative one. Consequently, in order to maximize the margin ρ, it is only necessary to minimize $||\mathbf{w}||$. This means that we solve for the minimum value of Lagrangian about $\mathbf{w}$ and b under the condition of Lagrange multiplier $\alpha_i \geq 0$ by applying Lagrange's method of undetermined multipliers.

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{l} \alpha_i y_i [(\mathbf{w} \cdot \mathbf{x}_i) + b] - 1 \tag{2.15}$$

We can state the following at the minimum value:

$$\frac{\partial L}{\partial \mathbf{w}} = 0, \quad \frac{\partial L}{\partial b} = 0 \tag{2.16}$$

Therefore, we can derive the following:

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i \mathbf{x}_i y_i \tag{2.17}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2.18}$$

From the discussion above, the problem of maximizing the margin boils down to a quadratic programming problem of minimizing the objective function $W(\alpha)$ in Eq.(2.19) under the constrained condition in Eq.(2.20).

$$W(\alpha) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{2.19}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1, \ldots, l \tag{2.20}$$

When we define the solution for this quadratic programming problem by numerical calculations as $\bar{\alpha_i}$, a support vector of positive examples as $\mathbf{x}_p$, and a support vector of negative examples as $\mathbf{x}_n$, we are able to decide $\bar{\mathbf{w}}$ and $\bar{b}$ as follows:

$$\bar{\mathbf{w}} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \tag{2.21}$$

$$\bar{b} = -\frac{1}{2}(\mathbf{w} \cdot \mathbf{x}_p + \mathbf{w} \cdot \mathbf{x}_n) \tag{2.22}$$

Support vectors, $\mathbf{x}_p$ and $\mathbf{x}_n$, meet the following equations, respectively.

$$\bar{\alpha_p}, \bar{\alpha_n} \geq 0, \ y_p = 1, \ y_n = -1 \tag{2.23}$$

Finally, the function distinguishing between positive and negative examples is given as follows:

$$f(\mathbf{x}) = sgn(\bar{\mathbf{w}} \cdot \mathbf{x} + \bar{b}) \tag{2.24}$$

In this equation, $sgn(z)$ is a function showing a sign of an argument z, and concretely the following:

$$sgn(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases} \tag{2.25}$$

2.6.2 Soft Margin

We extend the method as discussed above so as to conduct linear separation toward linearly-inseparable training data by a method called soft margin. Using new non-negative variables ξ_i ($i = 1, 2, \ldots, l, \ \xi_i \geq 0$), we introduce the following equations instead of Eq.(2.26), (2.27).

$$\mathbf{w} \cdot \mathbf{x_i} \geq 1 - \xi_i \quad if \ y_i = 1 \tag{2.26}$$

$$\mathbf{w} \cdot \mathbf{x_i} \leq -1 + \xi_i \quad if \ y_i = -1 \tag{2.27}$$

In this case, we consider minimization of Φ in the following equation instead of one of $||\mathbf{w}||$.

$$\Phi = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{l} \xi_i \tag{2.28}$$

The first item in the right-hand side is one about the margin, and the second is the error one showing how the linearly-inseparable training data is apart from the two hyperplanes $\mathbf{w} \cdot \mathbf{x} + b = \pm 1$.

C is a positive value parameter which balances weight of the first item and the second one. When C is a large value, the classifier puts importance on the errors from the training data to the hyperplane relatively, and when case C is small, it gives priority to the size of the margin relatively.

We apply Lagrange's method of undetermined multipliers to Eq.(2.28), and obtain the following equation with α_i and β_i Lagrange multipliers.

$$L(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{l}\{\alpha_i y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] - 1 + \xi_i\} - \sum_{i=1}^{l}\beta_i\xi_i \tag{2.29}$$

We can state the following:

$$\frac{\partial L}{\partial \mathbf{w}} = 0, \quad \frac{\partial L}{\partial b} = 0 \tag{2.30}$$

Therefore, we can derive the following:

$$\mathbf{w} = \sum_{i=1}^{l}\alpha_i \mathbf{x}_i y_i \tag{2.31}$$

$$\sum_{i=1}^{l}\alpha_i y_i = 0 \tag{2.32}$$

$$\alpha_i + \beta_i = C \tag{2.33}$$

From the above discussion, the problem of maximizing the margin boils down to a quadratic programming problem of minimizing the objective function $W(\alpha)$ at Eq.(2.34) under the constrained condition at Eq.(2.35).

$$W(\alpha) = -\sum_{i=1}^{l}\alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{2.34}$$

$$\sum_{i=1}^{l}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \ldots, l \tag{2.35}$$

The difference from the normal linear separation which we discussed at the preceding section is that the equation just gives an upper limit to α_i in Eq.(2.20). Hereinafter, in the same way as described above, when we define the solution for this quadratic programming problem by numerical calculations as $\bar{\alpha}_i$, a support vector of positive examples as $\mathbf{x}_p$, and a support vector of positive examples as $\mathbf{x}_n$, we are able to decide $\bar{\mathbf{w}}$ and $\bar{b}$ as follows:

$$\bar{\mathbf{w}} = \sum_{i=1}^{l}\alpha_i y_i \mathbf{x}_i \tag{2.36}$$

$$\bar{b} = -\frac{1}{2}(\mathbf{w} \cdot \mathbf{x}_p + \mathbf{w} \cdot \mathbf{x}_n) \tag{2.37}$$

We are able to decide $\bar{\mathbf{w}}$ and $\bar{b}$ as follows:

$$\bar{\mathbf{w}} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \tag{2.38}$$

$$\bar{b} = -\frac{1}{2}(\mathbf{w} \cdot \mathbf{x}_p + \mathbf{w} \cdot \mathbf{x}_n) \tag{2.39}$$

Support vectors, $\mathbf{x}_p$ and $\mathbf{x}_n$, meet the following equations, respectively.

$$\bar{\alpha_p}, \bar{\alpha_n} \geq 0,\ y_p = 1,\ y_n = -1 \tag{2.40}$$

Finally, the function distinguishing between positive examples and negative ones is given as follows:

$$f(\mathbf{x}) = sgn(\bar{\mathbf{w}} \cdot \mathbf{x} + \bar{b}) \tag{2.41}$$

2.6.3 Dealing with Non-linear Problems Using Kernel Functions

Using SVM, we are able to tackle a nonlinear problem even if it includes nonlinear boundary surfaces, in the way we map the problem into a higher-dimensional space by applying kernel functions, and solve the nonlinear problem by solving the linear separation problem in the space.

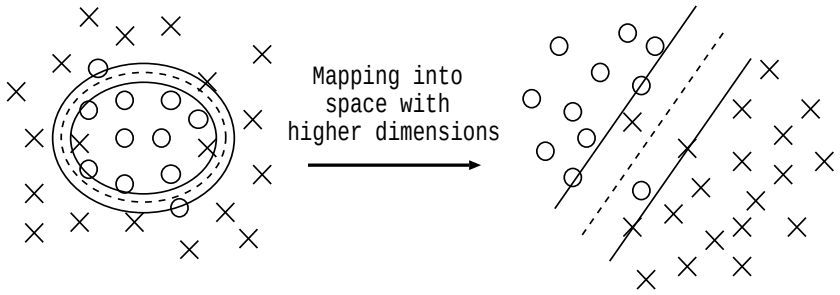


Fig. 2.4 Mapping into space with higher dimensions by kernel functions

When we express mapping into space with high dimension as $\vec{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_{d'}(\mathbf{x}))$ using d' nonlinear functions $\phi_k(\mathbf{x})$, the kernel function meets the following equation:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \vec{\phi}(\mathbf{x}_i)^t \vec{\phi}(\mathbf{x}_j) = \sum_{k=1}^{d'} \phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j) \tag{2.42}$$

In this case, the minimization problem that we should solve turns out to be the problem of minimizing the objective function $W'(\alpha)$ in Eq.(2.43) under the constrained condition in Eq.(2.44).

$$W'(\alpha) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{2.43}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad \forall i : \alpha_i \geq 0 \tag{2.44}$$

Consequently, $W'(\alpha)$ does not contain $\vec{\phi}$ explicitly, and we have only to define the inner product of $\vec{\phi}$. This is called the kernel trick. The following kernel functions are known as ones that meet this condition.

$$Polynomial\ type: \quad K_p(\mathbf{x}_i, \mathbf{x}_j) = (s\ \mathbf{x}_i \cdot \mathbf{x}_j + c)^d \tag{2.45}$$

$$Gaussian\ type: \quad K_g(\mathbf{x}_i, \mathbf{x}_j) = \exp\Big(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\Big) \tag{2.46}$$

$$Sigmoid\ type: \quad K_s(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa\ \mathbf{x}_i \cdot \mathbf{x}_j + \theta) \tag{2.47}$$

2.6.4 Applying SVM to Multiclass Classification

As mentioned in Section 2.4 , since the classifier such as SVMs is a binary classifier, if each document can belong to multiple categories, we need to extend it into multiple classification. Due to this, we generally deal with multiple classification by combining more than one binary classifiers, and typical methods for this combination are one vs. rest and pairwise method[25]. We explain one vs. rest and pairwise method.

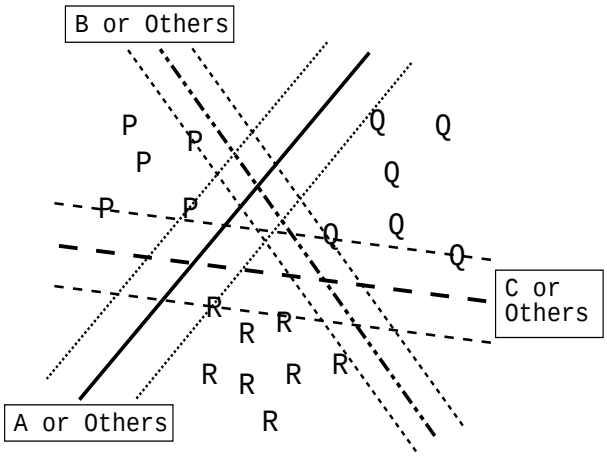


Fig. 2.5 One vs. rest method

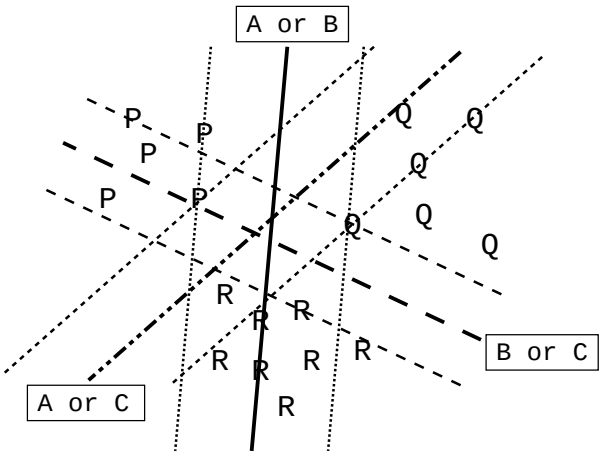


Fig. 2.6 Pairwise method

One vs. rest method

The one vs. rest method is a technique in which we build up k binary classifiers $f_c(\mathbf{x})$ to classify into a category or not about each k category. If we use SVMs, we classify an unknown case $\mathbf{x}'$ into the category that corresponds to the classifier where the value of $f_c(\mathbf{x}')$ is maximized. We show an example of one vs. rest method in Fig. 2.5. We build up three binary classifiers of $f_P(\mathbf{x})$, $f_Q(\mathbf{x})$ and $f_R(\mathbf{x})$ so as to classify three categories of P, Q and R.

Pairwise method

The pairwise method is a technique in which we build up ${}_kC_2$ binary classifiers to classify into arbitrary two categories selected from k categories. In this method, we introduce the binary classifier $f_{st}(\mathbf{x})$ to classify a category s and t. The classifier $f_{st}(\mathbf{x})$ determines that the example $\mathbf{x}$ is included in the category s when $f_{st}(\mathbf{x}) \geq 0$, and that it is included in the

category t when $f_{st}(\mathbf{x}) < 0$. We define a polling number V_c of a category c as the number of classifiers which determine that the example is included in the category c, out of the ${}_kC_2$ binary classifiers. We decide the definitive category in pairwise method as the one whose V_c has the largest number. We show an example of pairwise method in Fig. 2.6. We build up three binary classifiers of $f_{PQ}(\mathbf{x})$, $f_{PR}(\mathbf{x})$ and $f_{QR}(\mathbf{x})$ in order to classify three categories of P, Q and R.

In our research, we used one vs. rest method.

2.6.5 Avoidance of Over-learning by Structural Risk Minimization

Typically, when we apply machine learning to adjustment of parameters about classifiers, we gain the classifier that is so complex and expressive as to have lower frequency of errors toward a set of training data, whereas the classifier adapts to the training data so excessively that it causes a phenomenon called over-training, in which it has lower classification accuracy toward a set of test data. In response to this, when we use SVMs, we define complexity of classifiers as a VC dimension(Vapnik-Chervonenkis dimension) mathematically, we carry out a strategy to minimize the sum of the complexity and the frequency of classification errors. This notion is called structural risk minimization(SRM). We explain this notion.

We assume that the training data and test data are independent on each other, and were generated with the same probability distribution, $P(\mathbf{x}, y)$. We consider a classifier f which we gain by learning l training data, and describe the frequency of classification errors in the training data and in the whole of data as $R_{emp}(f)$ and $R(f)$, respectively. In this case, it is known that they are expressed in Eq.(2.48) and Eq.(2.49), and they meet the relationship of Eq.(2.50) with probability of $1 - \eta$[14].

$$R_{emp}(f) = \frac{1}{l}\sum_{i=1}^{l}\frac{1}{2}|f(\mathbf{x}_i) - y_i| \tag{2.48}$$

$$R(f) = \int \frac{1}{2}|f(\mathbf{x}) - y|\, dP(\mathbf{x}, y) \tag{2.49}$$

$$R \leq R_{emp} + \sqrt{\frac{h(\ln\frac{2l}{h} + 1) - \ln\frac{\eta}{4}}{l}} \tag{2.50}$$

In Eq.(2.50), h is the VC dimension. The second item in the right-hand side of this equation is called VC confidence, and the whole of the right-hand side called structural risk.

When we define the minimum diameter of hypersphere encompassing all of the data as D, the margin as ρ and the number of dimensions of feature space as d, the VC dimension is limited by a ceiling as the following equation:

$$h \leq min(\frac{D^2}{\rho^2}, d) + 1 \tag{2.51}$$

Due to the equation, when we make the margin ρ larger, we obtain a smaller VC dimension h. Because of this and Eq.(2.50), when we make the margin ρ larger, we obtain a lower structural

risk, while we get a larger frequency of classification errors, R_{emp}. In contrast, when we make ρ smaller, we obtain a lower R_{emp} but a larger structural risk[†]. As just described, the first item, R_{emp}, in the right-hand side of Eq.(2.50) and the second one, structural risk, have a relationship of trade-off with each other.

In addition, when we fully increase the dimension number of feature space, d, the minimization of h does not depend on d, so SVMs can be said to have generalization capability regardless of a dimension number of a feature space.

2.6.6 Characteristics of SVM and its Current Trend

We put the characteristics of SVMs shown above together in the following. Their advantages are the following:

- We are able to avoid over-learning by structural risk minimization.
- If we deal with higher-dimensional data, we do not need much more calculation amount and memory capacity in learning.

On the other hand, their disadvantages are the following:

- We are not able to draw classification rules explicitly.
- We need enough amount of training data to develop classifiers with high classification accuracy (generally in inductive learning including SVMs).

When researchers build up classifiers by machine learning methods, they have conducted inductive learning in the past, in which they have classified unknown test data with the classifiers that they gained by learning a certain number of training data. But they actually try to classify text data such as online information, they often cannot obtain enough amount of training data to build up a classifier with high accuracy. This is because it costs so much to classify training data and give labels of categories to it manually. Due to this, they have recently proposed some methods to generate classifiers with high accuracy by combining training data and test data, even if they have a small amount of training data.

As one of those methods, Joachims applied a transductive way to SVMs, and achieved high accuracy in classification results[10]. Moreover, Taira et al. reported improvement of classification accuracy by using a transductive boosting method[22]. While usual inductive learning is a method to minimize frequency of classification errors toward distribution of training data, the transductive boosting method is a learning method to focus on distribution of test data and minimize frequency of classification errors about test data.

[†]This is a state of over-training.

2.7 Evaluation of Text Classification Accuracy

In general search systems, we usually evaluate accuracy of text classification not analytically but empirically. This is because in order to evaluate a system analytically as being, for instance, reasonable and complete about its functionality, it is necessary to give the system problems formally which it tries to solve, such as about how its reasonability and completeness are defined, but this part depends much on our subjective judgement, so we cannot essentially formalize the problem as an objective one.

This is why empirical evaluation of classifiers means measuring "effectiveness of classification," that is, how it can make right decisions on classification from a subjective standpoint.

2.7.1 Precision and Recall

Precision and recall are old-established measures for evaluation of classification systems in a domain of information retrieval. Precision is a measure showing how many documents meet search conditions in all of the ones that the system has retrieved, and recall is a measure showing how many documents the system has retrieved in all of the ones that meet search conditions. Applying these to text classification, we define the following variable numbers based on correct answers and judgment of classification systems:

A: the number of documents which are positive examples and which the classification system also determines as positive ones.

B: the number of documents which are negative examples and which the classification system determines as positive ones.

C: the number of documents which are positive examples and which the classification system determines as negative ones.

D: the number of documents which are negative examples and which the classification system also determines as negative ones.

Using this definition, we can express precision P and recall R of the classification system as the following Eq.(2.52) and Eq.(2.53), respectively.

$$P = \begin{cases} \dfrac{A}{A+B} & \text{if A + B} > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.52}$$

$$R = \begin{cases} \dfrac{A}{A+C} & \text{if A + C} > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.53}$$

Generally, precision and recall have a trade-off relationship. We can also use error and fallout as measures focusing on the number of documents which are retrieved fallaciously by the system. We can express error e and fallout f as the following Eq.(2.54) and Eq.(2.55), respectively.

$$e = \frac{B+C}{A+B+C+D} \tag{2.54}$$

$$f = \begin{cases} \dfrac{B}{B+D} & \text{if B + D} > 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.55}$$

Fig. 2.7 is a view showing a frame format of the relationship between correct documents and retrieved documents (ones that the system determines as positive examples) in all of the documents. In this figure, we label omission as what we subtract the number of documents that suit to a search condition from the total number of correct documents, and noise as what we subtract the number of the documents that suit to the search condition from the retrieved documents. In the definition above, omission corresponds to C and noise to B.

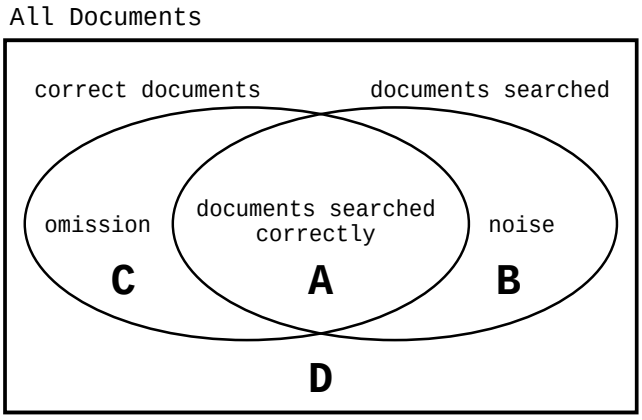


Fig. 2.7 Congruity of search systems

We can express precision, recall, error and fallout in the following words respectively.

Precision: a rate of correct documents in all of the retrieved documents, and this shows with how little omission the system can retrieve documents.

Recall: a rate of retrieved documents in all of the correct documents, and this shows with how little noise the system can retrieve documents.

Error: a rate of the sum of omission and noise in all of the documents, and this shows with how many errors (omission and noise) the system retrieves documents.

Fallout: a rate of retrieved documents in all of the incorrect documents, and this shows how many documents the system retrieves in all of the documents that we do not want to retrieve.

2.7.2 F-measure

F-measure[13, 17] is one of measures for evaluation of classification systems which combines precision and recall mentioned in the previous section, and was first advocated by Rijsbergen in

1979. F-measure is a one-dimensional numeric value meaning that when it is higher, the classification accuracy is better. We explain the definition of F-measure.

F-measure showing classification accuracy is defined as the following equation with a parameter β.

$$F_\beta = \frac{1+\beta^2}{\frac{1}{P}+\beta^2\frac{1}{R}} \tag{2.56}$$

F_β corresponds to precision P when $\beta = 0$, and corresponds to recall R when $\beta = +\infty$. In text classification, we usually use $\beta = 1$. This means that we equate the weight of precision with the one of recall. In this case, we express F-measure as follows by Eq.(2.56).

$$F_{\beta=1} = \frac{2PR}{P+R} = \frac{2A}{2A+B+C} \tag{2.57}$$

We use $F_{\beta=1}$ shown in Eq.(2.57) to evaluate classification experiments in our research.

Chapter 3

Polysemy Considered Method

3.1 A Problem of Polysemy Words in Text Classification

There are still many difficulties in natural language understanding by symbol processing approaches mentioned in Section 2.4 . One of them is a problem on how we disambiguate word senses. In text classification, some methods have been considered, such as ones to focus on co-occurrence relations between words and map into a completely different space called a conceptual space.

For instance, let us explain about a word "plant." The word "plant" has high mutual information between both a category "*grain*" and a category "*oilseed*." This reason is ambiguity of "plant" meaning vegetation and factories. If polysemous words like this are selected as features, they can exert a deleterious effect for classification accuracy because they might be contained in lots of negative documents, especially in cases where their ambiguity makes them have strong relationships with other categories by their different senses from the intended one. In other words, every feature should be fundamentally selected so as to characterize one category by its single meaning, but if they have ambiguous senses, their ambiguity makes a case where the features also characterize categories which we do not intend at all. Thus, here we decide that features expected to have ambiguous meanings are risk factors for text classification.

We will exemplify the impact with which polysemous words mentioned above have on classification accuracy if they are selected as features. We will describe details about categories and experimental conditions later, and here explain a simple simulation for text classification where we used five words that had the highest mutual information under the category "*acq*"* and the experimental condition (3)†.

Five words that had the highest mutual information under the category and the experimental condition mentioned above were "acquire", "inc", "acquisition", "stake" and "company." We denote the number of features as f. We define an initial state with no features as $f = 0$, or the state in which all positive and negative examples become mixed with zero dimension. Next, when we select one feature from the five words (in a state of $f = 1$, or one dimension) in turn, we show in Fig. 3.1 how positive and negative examples are separated. In this figure, circular marks stand for positive examples, and cross marks for negative ones. In those one-dimensional feature spaces with one word its feature, the number of positive or negative examples which exist at the point 1

*See Section 4.2 .
†See Section 4.3 .

or the number of those which exist at the point 0 correspond respectively to a, b, c, or d in Tab. 2.2.

Fig. 3.1 tells us that as for "inc" with the second highest mutual information and "company" with the fifth highest one, hundreds of documents include these words and belong to the category, but at the same time hundreds of documents include these words and belong to categories except for the category. The number of the former documents corresponds to a, and the number of the latter ones corresponds to b. Thus, many documents including "inc" or "company" also belong to other categories, and thus we can tell that "inc" and "company" are relatively ambiguous features.

In addition, we show in Fig. 3.2 distributions with 3,299 test documents in three-dimensional feature spaces by using three words out of the five words described above. The number of positive documents are 719, and the number of negative ones are 2,580. As we remarked in Section 2.4 , in our research, we set each element of document vectors into a binary value showing whether a word occurs in the document or not (we denote 1 when the word occurs, and 0 when it does not), so each document is mapped into any of vertices of the cube with length of a side 1 in the three-dimensional feature space. Fig. 3.2(a) is a result of a conventional approach in which we select three words in the order of descending mutual information into a feature set, and the combination of the features in this case is ("acquire", "inc", "acquisition"). We remove "inc" with the second mutual information from the feature set because of its ambiguity, and bring the third "acquisition" up to the second and the fourth "stake" up to the third. With that, we get Fig. 3.2(b) which shows a distribution based on a feature set ("acquire", "acquisition", "stake").

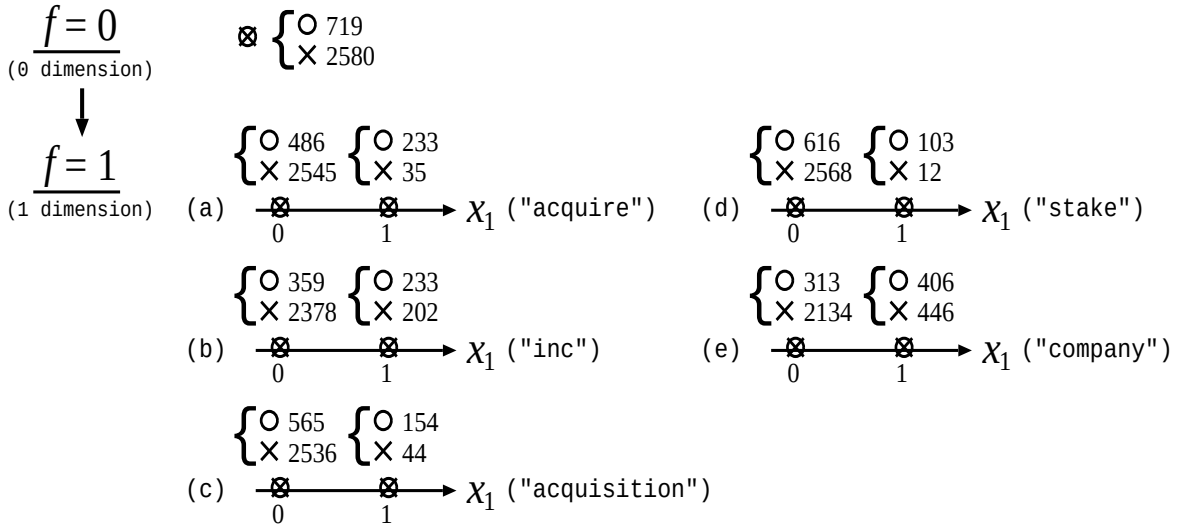
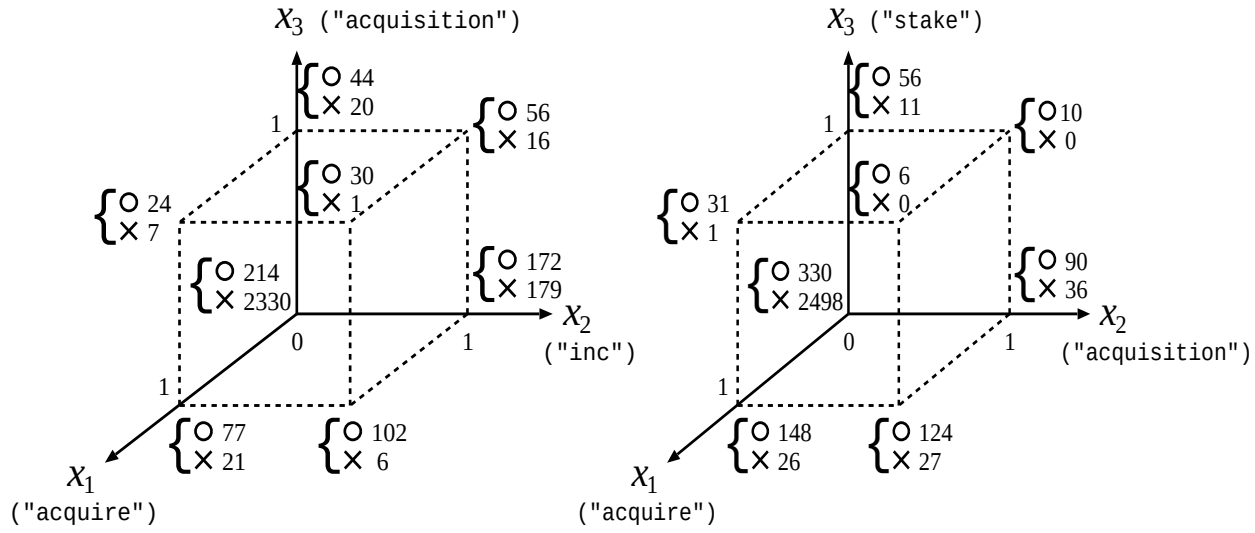


Fig. 3.1 Separation of positive and negative examples by using features of five words

In both cases of Fig. 3.2(a) and Fig. 3.2(b), the documents containing all of the three features, that is, the documents vectorized into the point (1, 1, 1) in a feature space compose a center of a positive cluster. Comparing these figures, we find out that "inc" with the second mutual information let more negative documents in a positive cluster whose center is the point (1, 1, 1). This simulation for classification is a result about test documents, and we can obtain the similar



(a) before removing "inc" (b) after removing "inc"

Fig. 3.2 Distributions of positive and negative examples in three-dimensional feature spaces

result about training documents. For this reason, if we select words viewed as polysemous ones, such as "inc", into features, that is likely to increase noises in training documents and making classification accuracy worse.

In addition, as can be seen by thinking of actual senses of words, the fifth word "company" is an polysemous word which has meanings of "corporation" and "associate," but "inc" is an univocal word which has only one meaning of "corporation." Although, in a strict sense, some words are not polysemous, have only one meaning, and have relevance to multiple categories with the one meaning as the word "inc" does, we regard these words as polysemous ones in a broad sense and risk factors for text classification in our research.

3.2 The Bilateral Character of Features

How can we extract polysemous words, such as "inc," mentioned in the previous section? We will describe a process for this in the next section. In this section, we explain a bilateral character of features as an advance preparation for the section.

Features characterize categories, and some features characterize them to positive but the others to negative. Features which characterize categories to positive has a property that documents including the features belong to the categories or that documents not including the features do not belong to the categories. On the other hand, features which characterize categories to negative has a property that documents including the features do not belong to the categories or that documents not including the features belong to the categories. Both features which characterize to positive and to negative are useful in the sense that they characterize categories.

We denote U as a whole of a set of documents, K_i as a set of documents belonging to target categories for classification, $\overline{K_i}$ as a set of documents belonging to non-target categories, T_w as a set of documents including a term w, $\overline{T_w}$ as a set of documents not including the term. We

present a simple frame format Fig. 3.3 showing how T_w and $\overline{T_w}$ tend to be dispersed over K_i and $\overline{K_i}$ in each case where w is a feature characterizing categories to positive and where it is a feature characterizing to negative.

There is a tendency that T_w exists in K_i in large numbers and $\overline{T_w}$ exists in $\overline{K_i}$ in cases where w characterizes categories to positive(Fig. 3.2 (a)). By contrast, there is a tendency that T_w exists in $\overline{K_i}$ in large numbers and $\overline{T_w}$ exists in K_i in cases where w characterizes categories to negative(Fig. 3.2 (b)).

Using the number of co-occurrence of a category and a word described in Section 2.5.3 , it is likely that in the case of features characterizing categories to positive, the larger the value of a and d at Tab. 2.2 gets, the more strongly the features characterizes to positive. So, figuring out a sum of these, it is safe to say that in the case of features characterizing to positive, the value of $a + d$ is large in a one-dimensional way. Likewise, it is also safe to say that in the case of features characterizing to negative, the value of $b + c$ is large.

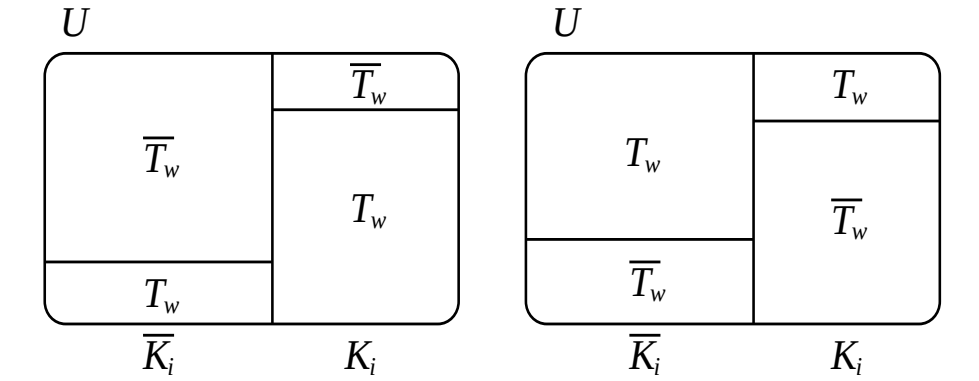


Fig. 3.3 A set of documents including features characterizing to positive and to negative

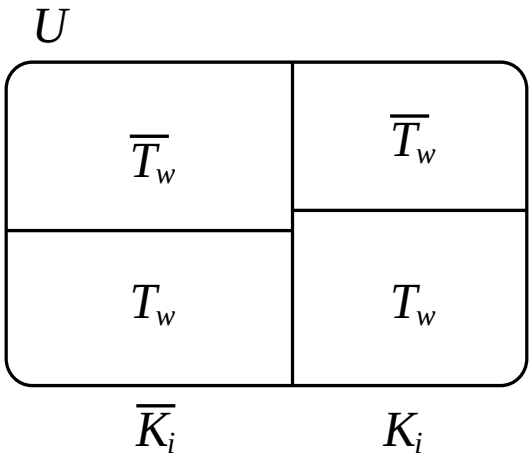


Fig. 3.4 A set of documents including polysemous features

3.3 A Proposal of the Polysemy Considered Method

Taking into account a bilateral character of features characterizing to positive and to negative, it is likely that polysemous features cannot characterize neither to positive nor to negative because of their ambiguity. Thus, it is possible that if w is a polysemous feature, T_w and $\overline{T_w}$ exist in both of categories K_i and $\overline{K_i}$, so more polysemous features make a distribution of words closer to the one shown in Fig. 3.4. Therefore, a polysemous feature w has a property that it makes T_w and $\overline{T_w}$ exist in both categories of K_i and $\overline{K_i}$.

Based on the discussion above, when we select words with such a property as features, the values of $a + d$ and $b + c$ are not likely to be disproportionately large. Consequently, considering a constrained condition, it is conceivable that a feature gets the most polysemous in the case of $a + d = b + c = \frac{N}{2}$. On the basis of this point, we formulate the following quadratic function in order to determine polysemy of those words quantitatively.

Definition (A Condition for Determining Polysemy)

We denote categories into which we classify documents as K_i and the numbers of co-occurrence between a category K_i and a word w as the following table shows. In this case, we determinate that it is the case for w to be a polysemous feature that a polysemous degree s satisfies the following conditional expression with a threshold ϵ.

$$s = \sqrt{\frac{a_i + d_i}{N} \ \frac{b_i + c_i}{N}} > \epsilon \tag{3.1}$$

In this regard, $N = a_i + b_i + c_i + d_i,\ 0 \leq \epsilon \leq \frac{1}{2}$

	K_i occurs	K_i does not occur
w occurs	a_i	b_i
w does not occur	c_i	d_i

This s is an indicator that is able to evaluate polysemy of words quantitatively, and we describe it as a polysemous degree. In the definition mentioned above, the ranges of $a_i + d_i,\ b_i + c_i$ are $0 \leq a_i + d_i \leq N,\ 0 \leq b_i + c_i \leq N$. We can change the conditional expression as follows.

$$\begin{aligned} s &= \frac{1}{N}\sqrt{(a_i + d_i)(N - (a_i + d_i))} \\ &= \frac{1}{N}\sqrt{\frac{N^2}{4} - \left(\frac{N}{2} - (a_i + d_i)\right)^2} \end{aligned} \tag{3.2}$$

Thus, s reaches the minimum $s_{min} = 0$ at the case of $a_i + d_i = 0$ or $a_i + d_i = N$, and reaches the maximum $s_{max} = \frac{1}{2}$ at the case of $a_i + d_i = b_i + c_i = \frac{N}{2}$. Consequently, the range of the polysemous degree is $0 \leq s \leq \frac{1}{2}$.

In addition, we can draw s as a function of $a_i + d_i$ into a graph similar to the shape of a semiellipse shown in Fig. 3.5. So, for example, in the case of $\epsilon = \epsilon_1$, t_1 and t_2 in Fig. 3.5 are expressed in the following equations:

$$t_1 = \frac{N}{2} - \sqrt{\frac{N^2}{4} - \epsilon_1^2} \tag{3.3}$$

$$t_2 = \frac{N}{2} + \sqrt{\frac{N^2}{4} - \epsilon_1^2} \tag{3.4}$$

In this case, we can determine that words satisfying $t_1 \leq a_i + d_i \leq t_2$ have polysemy.

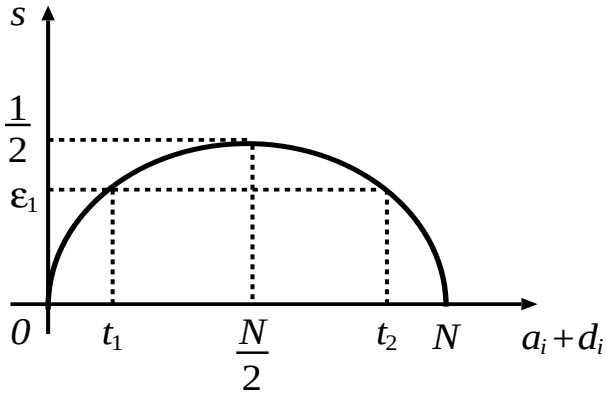


Fig. 3.5 Polysemous degree s as a function of $a_i + d_i$

3.4 Positioning of the Proposal Method in Text Classification Processing

Summarizing a method for polysemy reduction mentioned so far, this method is to eliminate words expected to have polysemy from feature sets, which are selected by a feature selection method using mutual information. In the whole flow of text classification process described in Section 2.3 , we do improve a process of feature selection. And in Fig. 2.1 we can show the positioning of our proposal method for reducing polysemy in the whole flow of text classification process.

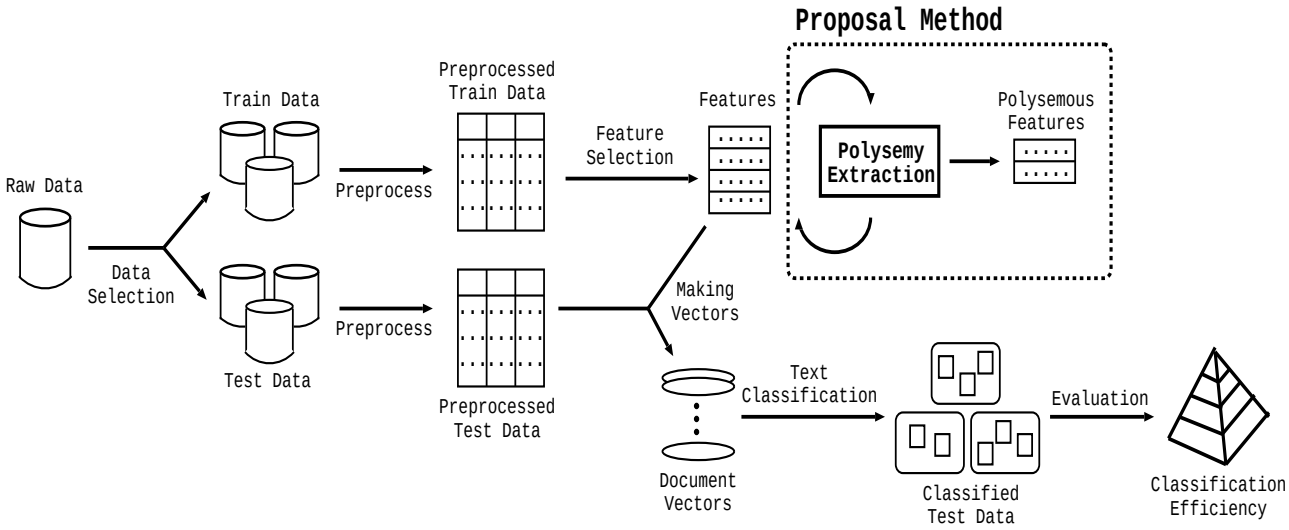


Fig. 3.6 Positioning of our proposal approach in the whole flow of text classification process

Chapter 4

Experimental Setting

4.1 Experimental Environment

In our experiments, we exploited two personal computers with 2.53 GHz CPU and 1.02 gigabyte of memory. And we used RedHat Linux 8.0 with its kernel the version of 2.4.18-14 as an operating system on the computers in our experiments. The software that we utilized in our experiments was SVM^{light} Ver5.00*.

4.2 Text Corpus

We used Reuters-21578† for a text corpus. We split it into training and test articles by following "The Modified Apte(ModApte) Split"(See Appendix A .). By using this way, we split it into 9603 training articles and 3299 test ones. We used these as training and test data respectively in our experiments.

Categories which we used in the classification process of our experiments were the following ten: "*acq*", "*corn*", "*crude*", "*earn*", "*grain*", "*interest*", "*money-fx*", "*ship*", "*trade*" and "*wheat.*" We show in Tab. 4.1 the numbers of training and test articles which belong to these categories.

4.3 Preprocessing

As we mentioned in Section 2.3 , since training and test articles that we just extracted include noises typified by unnecessary words such as "and", "or" and so forth, we usually have to remove these. But in our experiments, we also carried out experiments in the case of leaving unnecessary words deliberately so as to examine the effect of the noises.

To be more precise, we conducted the following three different processes as preprocessing, and we call these experimental conditions.

(1) No preprocessing

(2) Removing disabled words and coping with plural forms and word inflection

*See Appendix C .
†See Appendix A .

Tab. 4.1 Number of training and test articles which belong to the categories used in the classification process

Name of categories	Number of training articles	Number of test articles
acq	1650	719
corn	182	56
crude	389	189
earn	2877	1087
grain	433	149
interest	347	131
money-fx	538	179
ship	197	89
trade	369	118
wheat	212	71

(3) Removing disabled words, coping with plural forms and word inflection, and removing numerals

In these processes of (2) and (3), we conducted part-of-speech analysis of the corpus by using a tool for part-of-speech tagging, TreeTagger[‡] to remove disabled words, cope with plural forms and word infection, and remove numerals based on the rule shown in Tab. 4.2. In the right side of the table, "→ remove" means removing disabled words, and "→ (abbreviated symbols)" means coping with plural forms and word inflection. Moreover, as far as removing numerals, we removed words tagged as a part of speech showing "CD cardinal number" in Tab. 4.2.

We summarize the three different processes into Tab. 4.3. Descriptions of an experimental condition (1), (2), or (3) after this table in this paper stand for the conditions shown in Tab. 4.3.

[‡]See Appendix B .

Tab. 4.2 Removing disabled words, coping with plural forms and word inflection, and removing numerals

#	Abbreviation	Part of Speech	Preprocessing
1.	CC	Coordinating conjunction	→ remove
2.	CD	Cardinal number	(→ remove)
3.	DT	Determiner	→ remove
4.	EX	Existential *there*	→ remove
5.	FW	Foreign word	
6.	IN	Preposition or subordinating conjunction	→ remove
7.	JJ	Adjective	
8.	JJR	Adjective, comparative	→ JJ
9.	JJS	Adjective, superlative	→ JJ
10.	LS	List item marker	→ remove
11.	MD	Modal	
12.	NN	Noun, singular or mass	
13.	NNS	Noun, plural	→ NN
14.	NP	Proper noun, singular	
15.	NPS	Proper noun, plural	→ NP
16.	PDT	Predeterminer	
17.	POS	Possessive ending	→ remove
18.	PP	Personal pronoun	→ remove
19.	PP$	Possessive pronoun	→ remove
20.	RB	Adverb	
21.	RBR	Adverb, comparative	→ RB
22.	RBS	Adverb, superlative	→ RB
23.	RP	Particle	
24.	SYM	Symbol	
25.	TO	*to*	→ remove
26.	UH	Interjection	
27.	VB (VH, VV)	Verb, base form	
28.	VBD (VHD, VVD)	Verb, past tense	→ VB (VH, VV)
29.	VBG (VHG, VVG)	Verb, gerund or present participle	→ VB (VH, VV)
30.	VBN (VHN, VVN)	Verb, past participle	→ VB (VH, VV)
31.	VBP (VHP, VVP)	Verb, non-3rd person singular present	→ VB (VH, VV)
32.	VBZ (VHZ, VVZ)	Verb, 3rd person singular present	→ VB (VH, VV)
33.	WDT	Wh-determiner	→ remove
34.	WP	Wh-pronoun	→ remove
35.	WP$	Possessive wh-pronoun	→ remove
36.	WRB	Wh-adverb	→ remove
37.	#	*#* symbol	→ remove
38.	$	*$* symbol	→ remove
39.	SENT	period	→ remove
40.	,	comma	→ remove
41.	:	colon	→ remove
42.	(	left parenthesis	→ remove
43.	)	right parenthesis	→ remove
44.	”	two single quotation marks, right	→ remove
45.	“	two single quotation marks, left	→ remove

Tab. 4.3 Experimental conditions by the difference of preprocessing

Experimental condition	Removing disabled words and coping with plural forms and word inflection	Removing numerals
(1)	not done	not done
(2)	done	not done
(3)	done	done

Chapter 5

Experimental Results

5.1 Preprocessing

The number of words in training articles under each experimental condition is shown in Tab. 5.1. As shown in the table, the corpus had 13,415 words which were disable words, words of plural forms and inflected words such as comparatives, and 14,617 numerals.

Tab. 5.1 Number of words after removing disabled words, coping with plural forms and word inflection, and removing numerals

Experimental condition	Number of words	Removing disabled words and coping with plural forms and word inflection	Removing numerals
(1)	45,558	Not done	Not done
(2)	32,143	Done	Not done
(3)	17,526	Done	Done

5.2 Pre-experimentation

We conducted the following two experiments as pre-experimentation of classification processing of test articles.

1. Measuring distributions of mutual information between words and categories
2. Classifying training articles with a classification system

The purposes of these experiments are the following; the experiment 1. is to investigate potentiality of features by knowing how many effective features exist and how specifically they characterize each category; the experiment 2. is to take aim about how much classification accuracy we will obstain from the classifier in the case of classifying test articles. We describe the results of these pre-experimentation in Section 5.2.1 and Section 5.2.2 .

5.2.1 Measuring Distributions of Mutual Information

First of all, we examined the number of words with mutual information (the number of candidate words for features) in each of ten categories. We shows words in Tab. 5.3 which have first, fifth, tenth, fiftieth, one hundredth, two hundredth, five hundredth, one thousandth, and two thousandth highest mutual information in the categories. We can tell that top words have strong relevance with the category in any experimental conditions. Particularly, nouns tend to be selected predominately in top 100 words.

Tab. 5.2 Number of words with mutual information (the number of candidate words for features) in the categories used for classification processing.

Category name	Experimental condition(1)	Experimental condition(2)	Experimental condition(3)
acq	14114	9544	7745
corn	4501	3426	2511
crude	7916	5708	4539
earn	18324	15205	5929
grain	7459	5504	4043
interest	5039	3761	2959
money-fx	7136	5246	4178
ship	4967	3602	3067
trade	7202	5315	4439
wheat	4841	3675	2740

Next, we show the distributions of mutual information between categories and words in the figures from Fig. 5.1 to Fig. 5.3. In these figures, figures in the left side represent the numbers of words with mutual information in a form of histogram. Each figure has a horizontal axis measuring mutual information, and a vertical axis measuring the number of words with mutual information logarithmically, and each interval in it is 0.005. Figures in the right side show the numbers of words with mutual information at the figures in the left side as cumulative ratios, and represent them from high mutual information to low one. Each figure has a horizontal axis measuring mutual information linearly, and a vertical axis measuring the cumulative ratio with the maximum 100[%].

We will show these distributions of mutual information about the categories of "*acq*", "*earn*" and "*trade*." As will be noted from these figures on the left side, the word with highest mutual information had more than 0.4 (about 0.43), which was only in the category of "*earn*." And even the word with sixth highest had more than 0.1. On the other hand, in the categories of "*acq*" and "*trade*," the words with highest had only mutual information of 0.1 around. Moreover, as will be noted from the figures on the right side, when we examine the number of words with mutual information from a standpoint of cumulative ratios, we found that more than 99% of words had mutual information from 0 to 0.05 in common to any categories and experimental conditions.

Next, we will compare among the experimental conditions from (1) to (3). When we first compare (1) with (2), the distribution of (2) shows lower values all around than the one of (1). This result makes us recognize that disable words and inflected words, such as plural forms, comparatives and so on, were distributed over all around mutual information. When we compare (2) with (3),

Tab. 5.3 Words with high mutual information

(a) Experimental condition(1)

Category name	Word							
	first	fifth	tenth	fiftieth	one hundredth	five hundredth	one thousandth	two thousandth
acq	inc	acquisition	merger	assets	april	cable	soriano	312.8
corn	corn	tonnes	u.s	cotton	10,572,402	avg	centre-west	rudy
crude	oil	bpd	exploration	fields	on	usx	<slc	snith
earn	vs	said	loss	includes	no	met	place	leader
grain	wheat	corn	barley	subsidized	durum	from	operating	muni-bonds
interest	rate	market	fed	exchequer	nigel	strength	steepening	investor
money-fx	dollar	central	shortage	repurchase	underlying	&#3	inferences	translating
ship	ships	vessel	ship	the	terminal	jurisdictions	231,722	doing
trade	trade	imports	goods	on	ministry	launched	trip's	invoking
wheat	wheat	export	department	company	than	searching	onic's	traded

(b) Experimental condition(2)

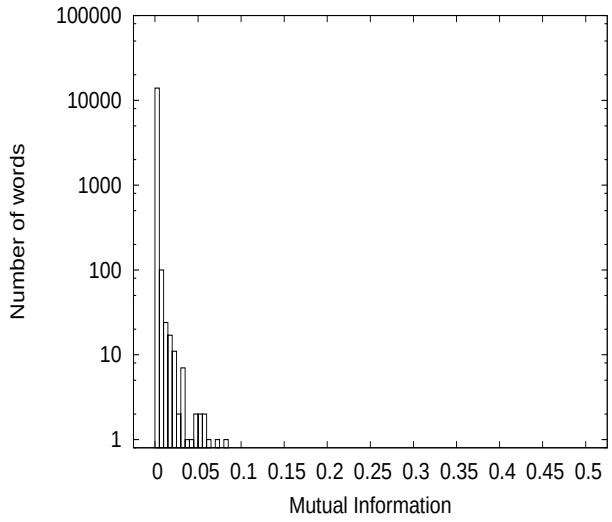
Category name	Word							
	first	fifth	tenth	fiftieth	one hundredth	five hundredth	one thousandth	two thousandth
acq	acquire	company	say	co	systems	consent	accord	mexican
corn	corn	tonne	bushel	marketing	sen	4.500	uphold	provide
crude	oil	energy	day	would	exporting	occupy	6.60	428
earn	vs	be	shrs	exclude	meet	canadian	previous	angeles
grain	wheat	tonne	barley	soviets	loading	fe	shipping	single
interest	rate	england	today	drain	security	7.10	5.99	tune
money-fx	currency	central	dealer	britain	trading	revalue	affairs	60-90
ship	ship	tanker	seaman	sail	andreas	montreal	going	lodge
trade	trade	japan	tariff	barrier	informal	compatible	58.6	939.3
wheat	wheat	export	soviet	1985/86	hectare	impair	546,423	fishery

(c) Experimental condition(3)

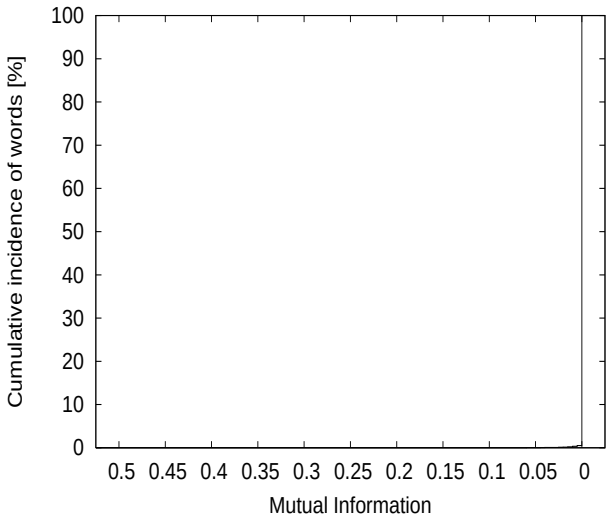
Category name	Word							
	first	fifth	tenth	fiftieth	one hundredth	five hundredth	one thousandth	two thousandth
acq	acquire	company	say	co	usair	hanson	enlarged	escrow
corn	corn	tonne	bushel	share	moscow	reexport	ii	seller
crude	oil	energy	day	mile	pump	meet	commission	angle
earn	vs	be	shrs	today	add	reduce	former	household
grain	wheat	tonne	barley	subsidized	land	iwc	harmonious	tian
interest	rate	england	discount	yesterday	half	debenture	symposium	socialist
money-fx	currency	central	dealer	iraq	circulation	alert	explosive	competitor
ship	ship	tanker	seaman	aegean	turkish	cruzado	continent	arrange
trade	trade	japan	tariff	product	reduce	ambassador	ally	judgement
wheat	wheat	export	soviet	planting	algeria	harkin	cultivation	colorado

the distribution of (3) shows lower values than the one of (2) in a part of low mutual information, but has little difference between them in a part of high mutual information. This indicates that numerals have relatively low mutual information. This indication agree with the results in Tab. 5.3 that there were few numerals in words with high mutual information.

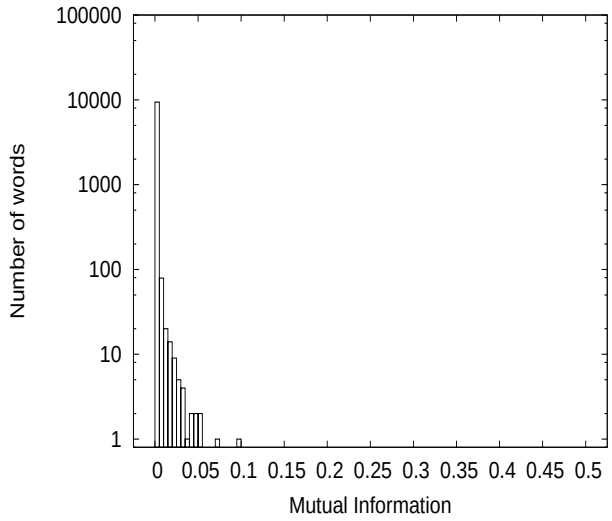
We also found out that other seven categories not shown in the figures from Fig. 5.1 to Fig. 5.3 have the similar characteristics among "*acq*" and "*trade*," and that "*earn*" has a particular distribution out of all the ten categories.



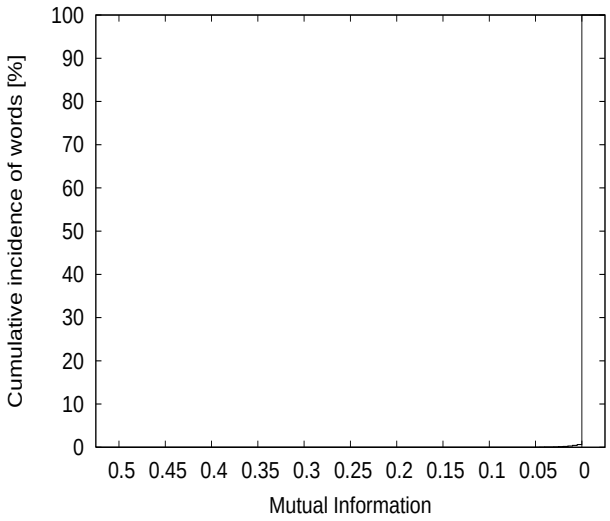
(a) Category "*acq*", experimental condition (1)



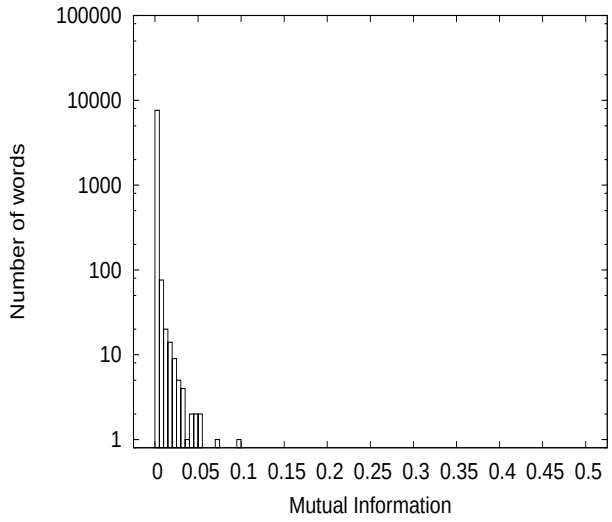
(a′) Category "*acq*", experimental condition (1)



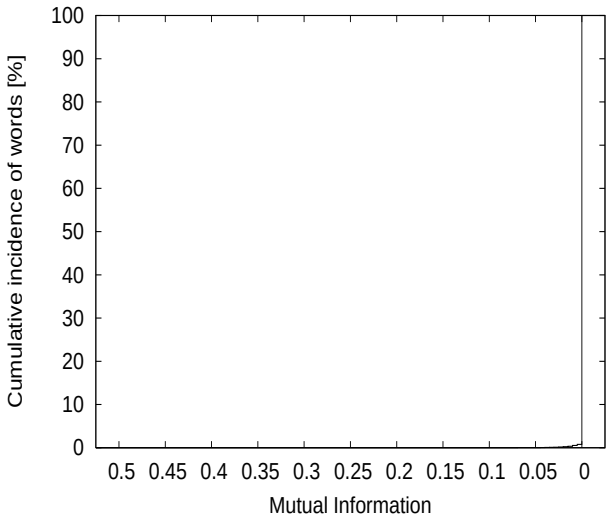
(b) Category "*acq*", experimental condition (2)



(b′) Category "*acq*", experimental condition (2)

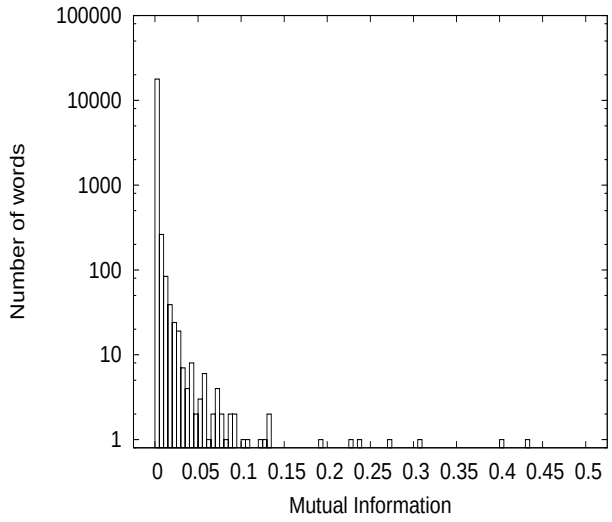


(c) Category "*acq*", experimental condition (3)

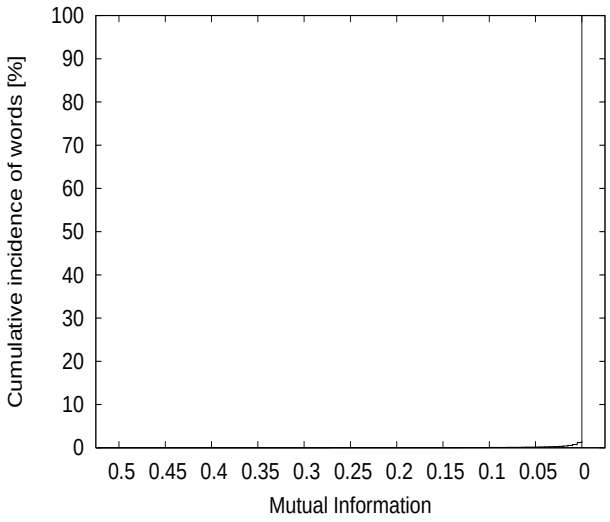


(c′) Category "*acq*", experimental condition (3)

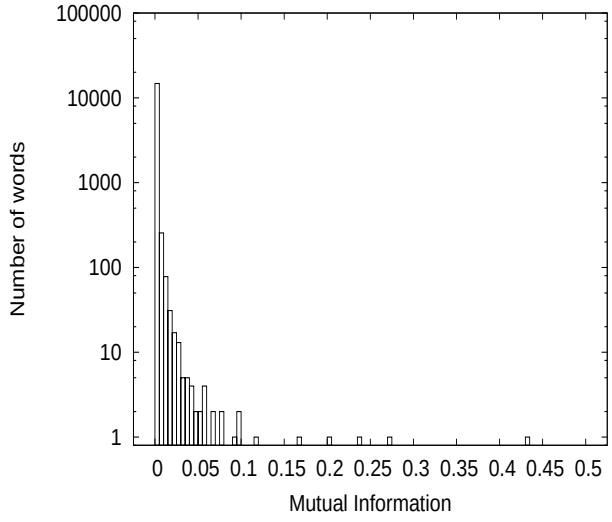
Fig. 5.1 Distribution of words with mutual information (category "*acq*")



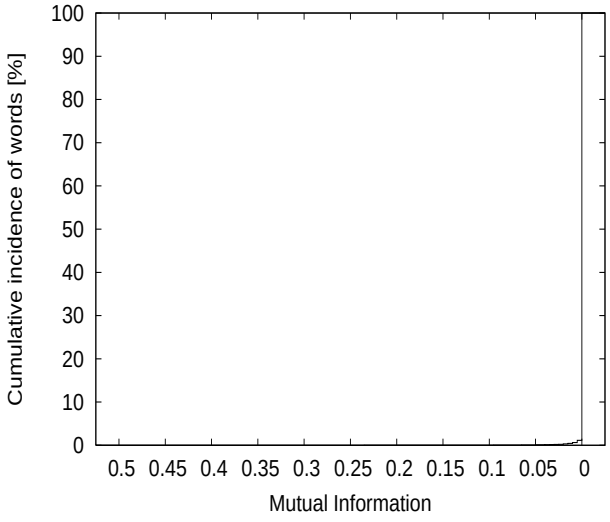
(a) Category "*earn*", experimental condition (1)



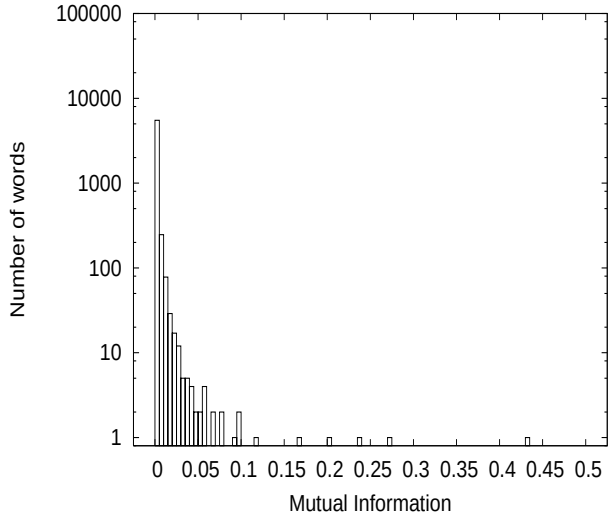
(a′) Category "*earn*", experimental condition (1)



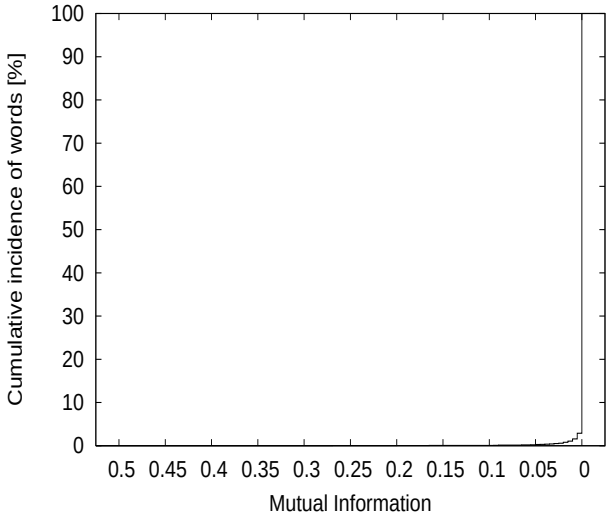
(b) Category "*earn*", experimental condition (2)



(2′) Category "*earn*", experimental condition (2)

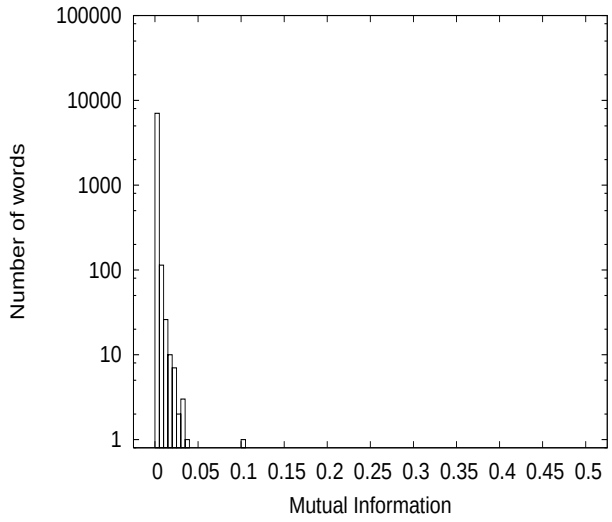


(3) Category "*earn*", experimental condition (3)

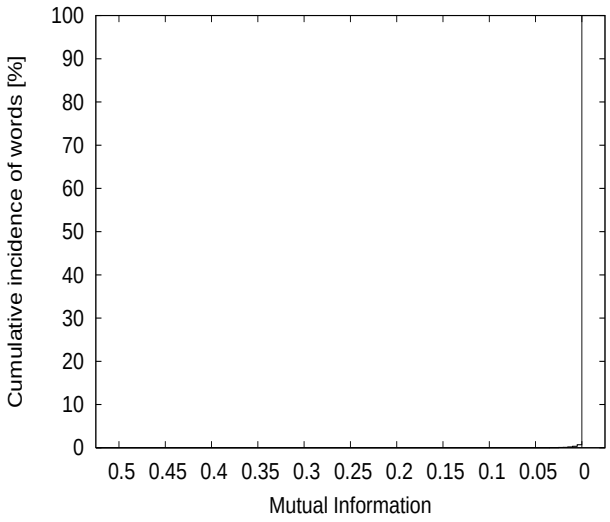


(3′) Category "*earn*", experimental condition (3)

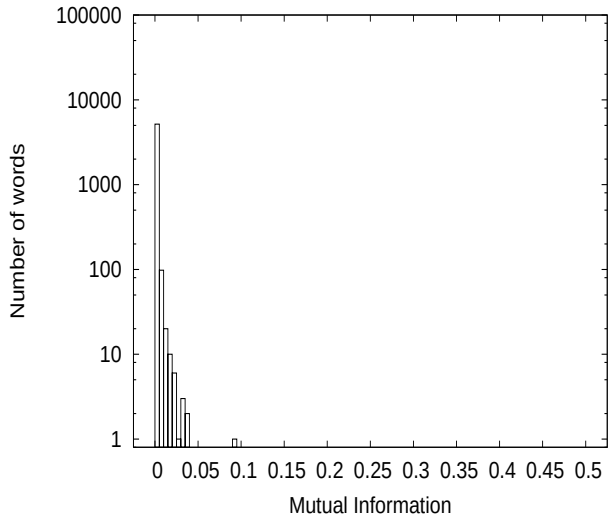
Fig. 5.2 Distribution of words with mutual information (category "*earn*")



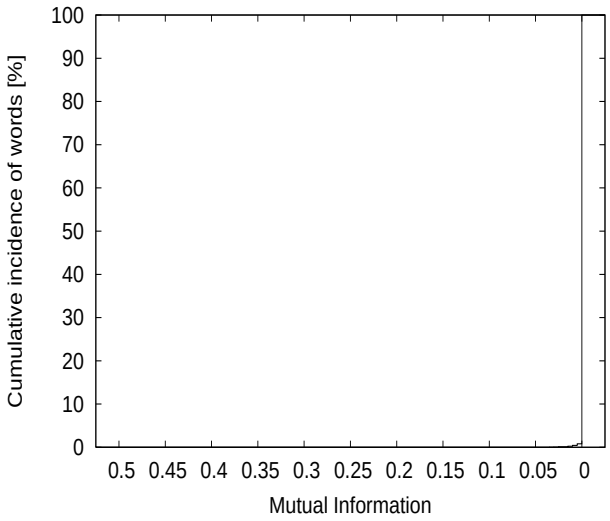
(a) Category "*trade*", experimental condition (1)



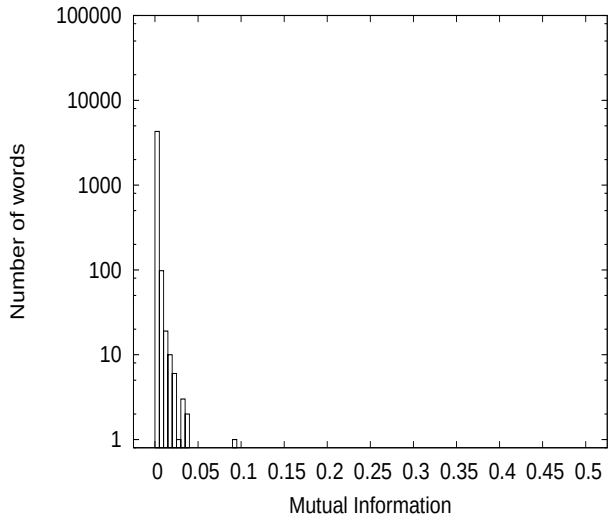
(a′) Category "*trade*", experimental condition (1)



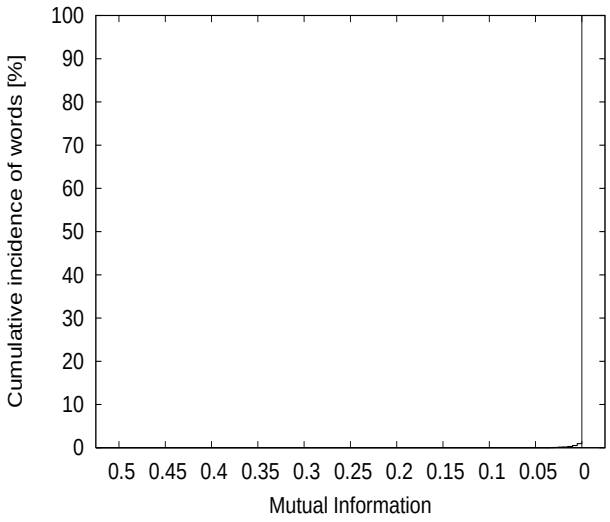
(b) Category "*trade*", experimental condition (2)



(2′) Category "*trade*", experimental condition (2)



(3) Category "*trade*", experimental condition (3)

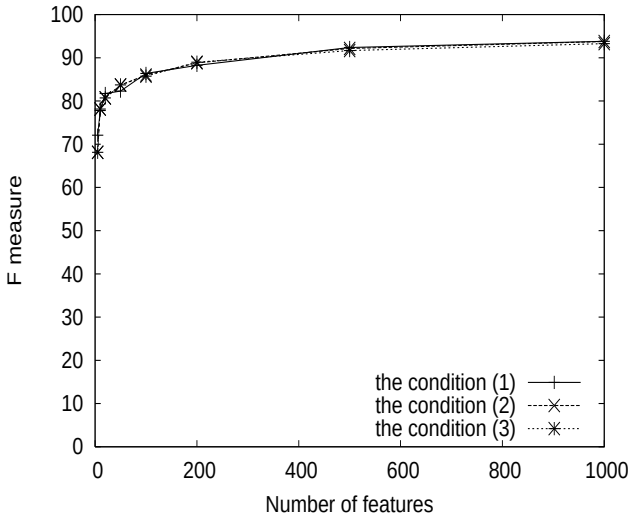


(3′) Category "*trade*", experimental condition (3)

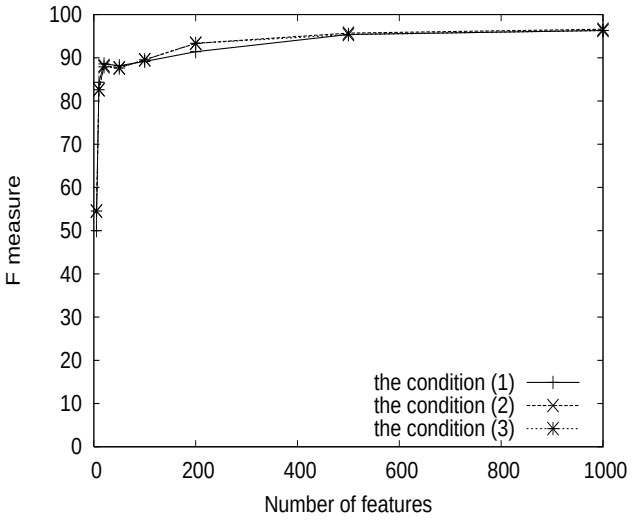
Fig. 5.3 Distribution of words with mutual information (category "*trade*")

5.2.2 Classifying Training Articles

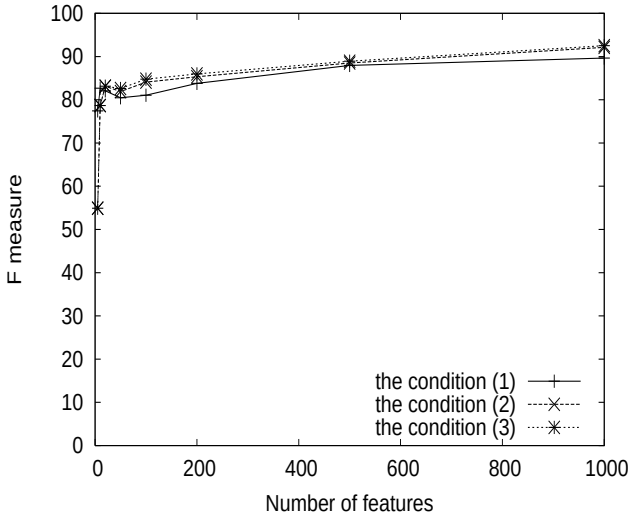
We created a classification system by making it learn training articles with SVM, and measured classification accuracy on each category in the case where we made it classify the learned training articles before classifying test articles. In this experiment, we carried out the classification by exploiting the linear SVM with SVM^{light} described in Section 4.1 . The condition of the linear SVM is the same in classification experiments which we state after this. We changed the number of features into 5, 10, 20, 50, 100, 200, 500 and 1000. We show the experimental results under the experimental conditions from (1) to (3) as to the ten categories in Fig. 5.4(a) to Fig. 5.4(j), respectively.



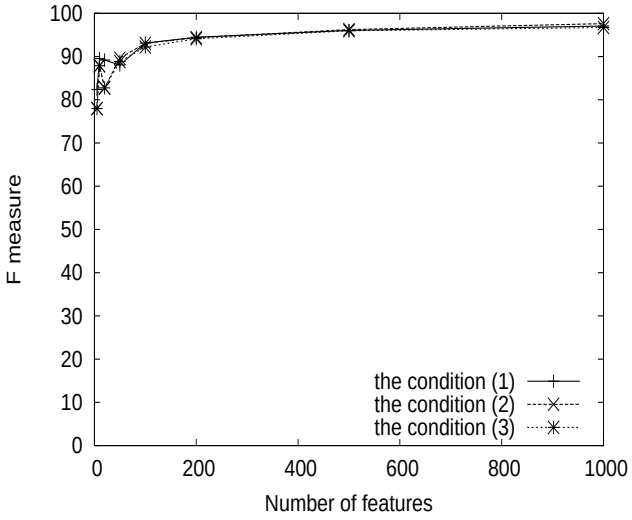
(a) Category "*acq*"



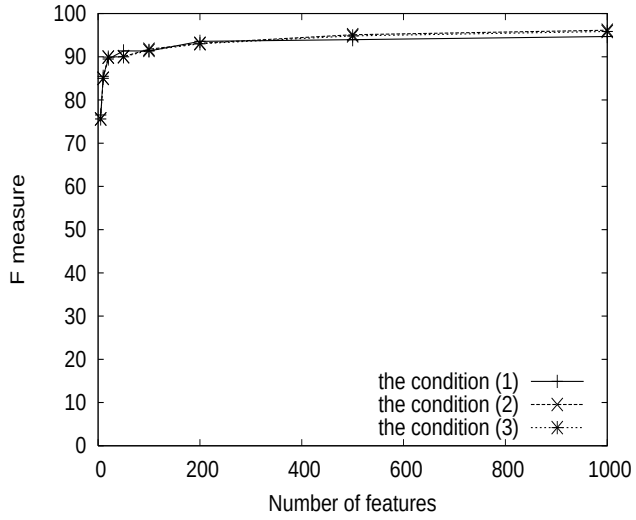
(b) Category "*corn*"



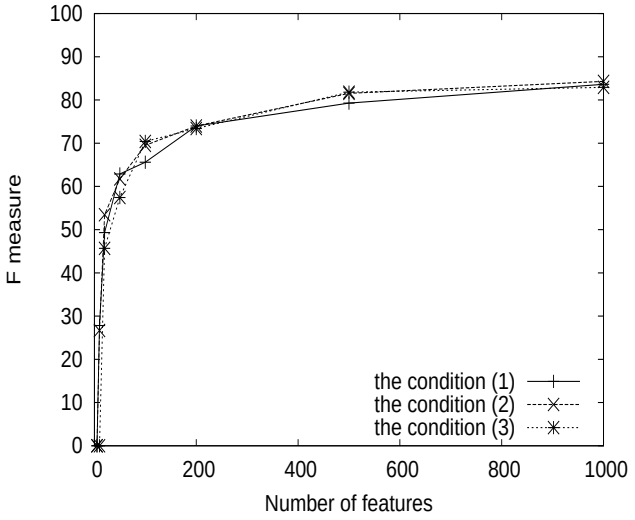
(c) Category "*crude*"



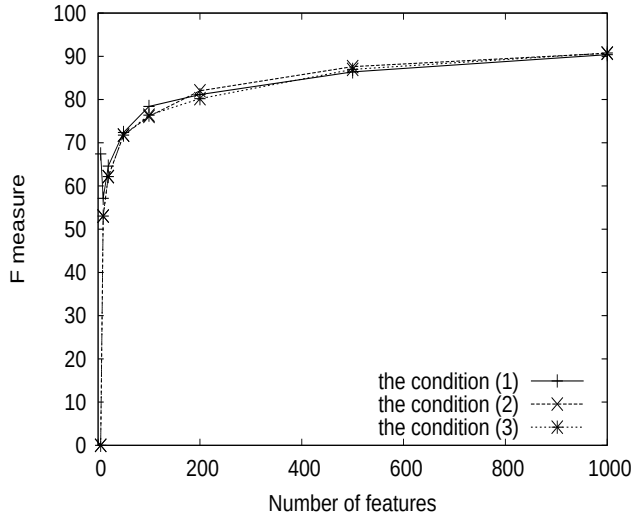
(d) Category "*earn*"



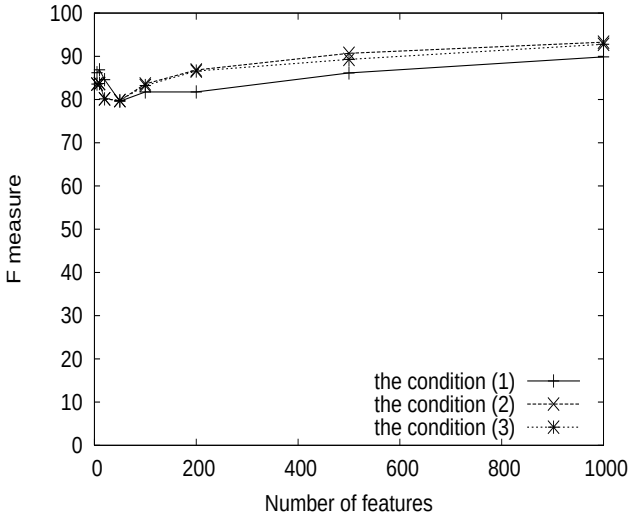
(e) Category "*grain*"



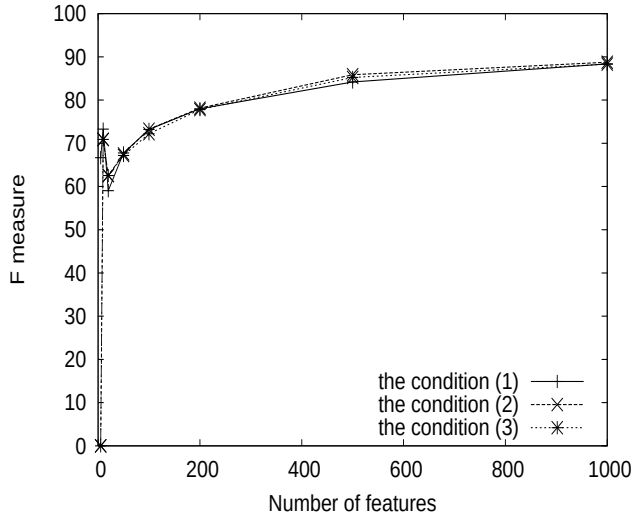
(f) Category "*interest*"



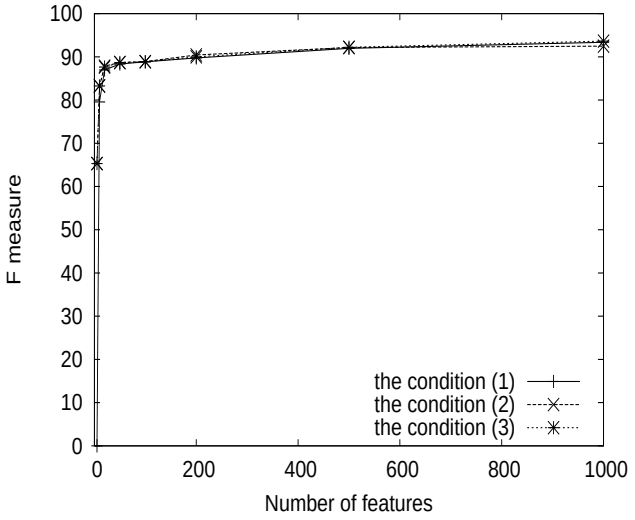
(g) Category "*money-fx*"



(h) Category "*ship*"



(i) Category "*trade*"



(j) Category "*wheat*"

Fig. 5.4 Results of classifying training articles

As will be noted from Fig. 5.4, we obtained the results common to categories, that is, the classifiers achieved only low F-measure when we used a small number of features, and on the other hand, achieved high F-measure when we used a large number of features as a whole, although some of them showed locally temporal diminution in the F-measure in the latter case. When we used more than a certain number of features, they had a peak of a constant F-measure.

Moreover, from comparison of experimental conditions from (1) to (3) in each category, we found improvement of a few points of the F-measure under experimental conditions (2) and (3) in the category of "*ship*," but found little difference in F-measures under the three experimental conditions in the other categories. Then, we put classification accuracy of all the categories under the experimental condition (3) into a single graph in order to look at the difference in those categories. We show it in Fig. 5.5.

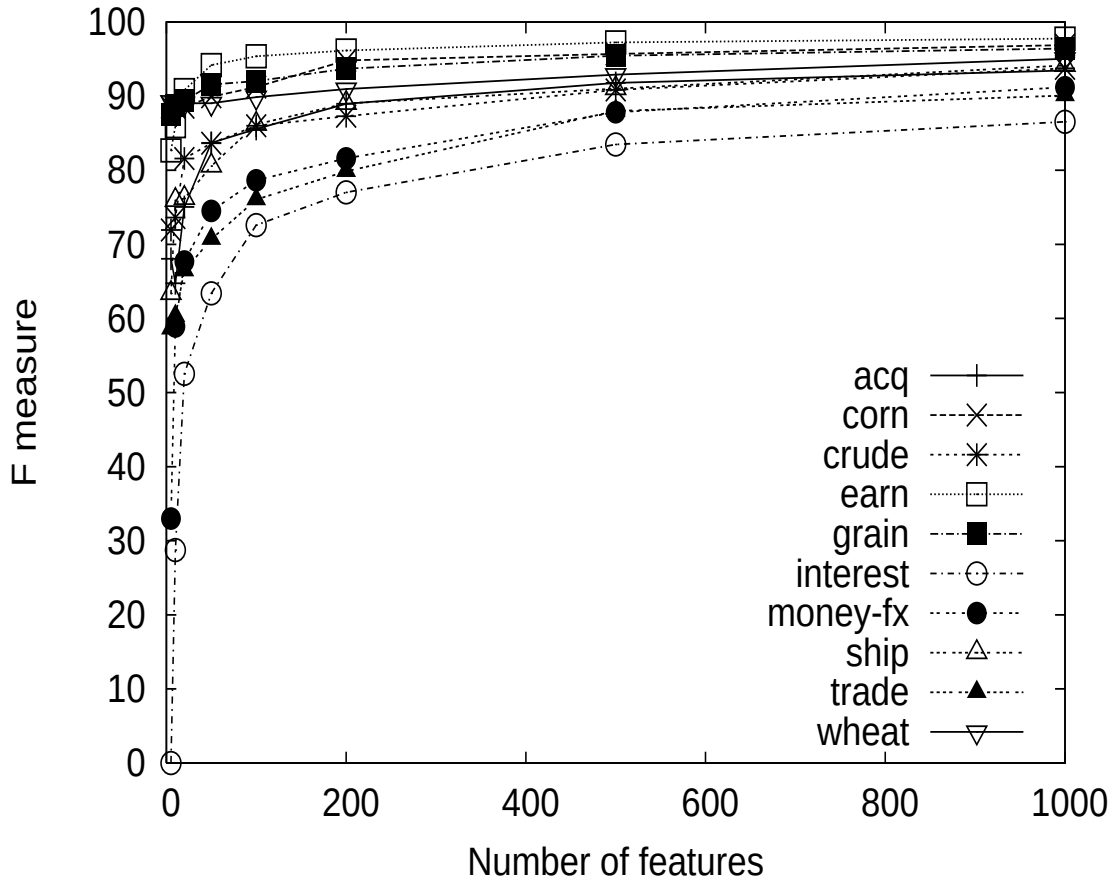


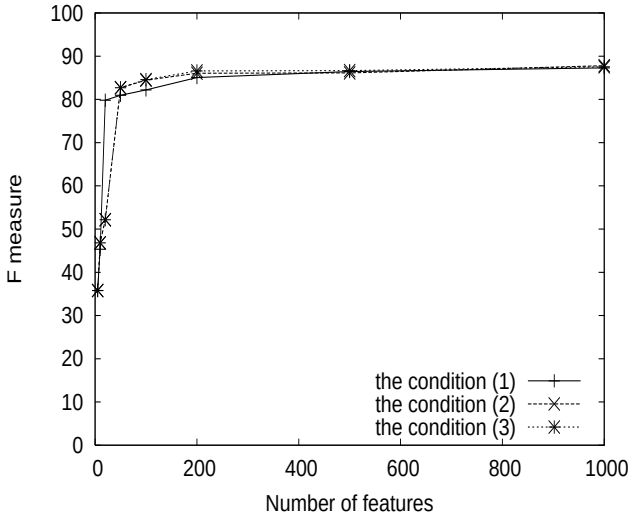
Fig. 5.5 Results of classifying training articles under the experimental condition (3)

From this figure, we found that the classifiers achieved considerably high classification accuracy of training articles, which is from 80 to late 90s as the F-measure. We show particularly high and low measures below. Parenthetic numbers are the highest F-measure within the range in features from 5 to 1,000.

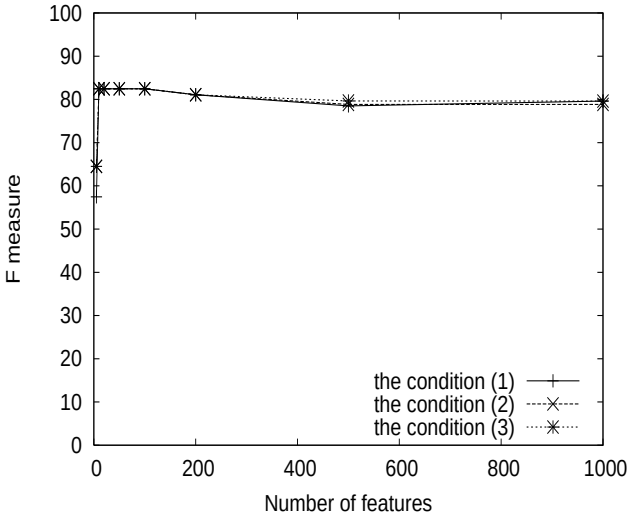
- Categories with a high F-measure: "*earn*"(96.72), "*corn*"(96.30), "*grain*"(95.84)
- Categories with a low F-measure: "*interest*"(82.92), "*trade*"(88.31)

5.3 Classification Results with the Conventional Method

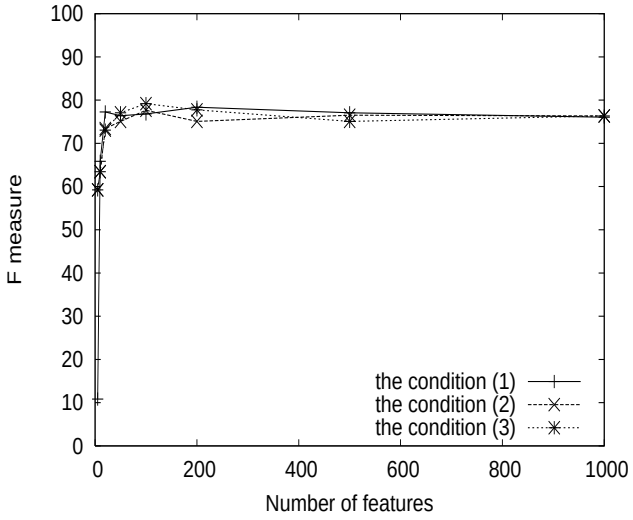
As we described in Section 5.1 , we carried out the three different preprocessing of (1) no preprocessing, (2) removing disabled words and coping with plural forms and word inflection, and (3) removing disabled words and coping with plural forms and word inflection, and removing numerals. We conducted classification experiments of test articles under these three experimental conditions, and show the results in Fig. 5.6(a) to Fig. 5.6(j). We changed the number of features into 5, 10, 20, 50, 100, 200, 500 and 1000, as well as in the case of classifying training articles.



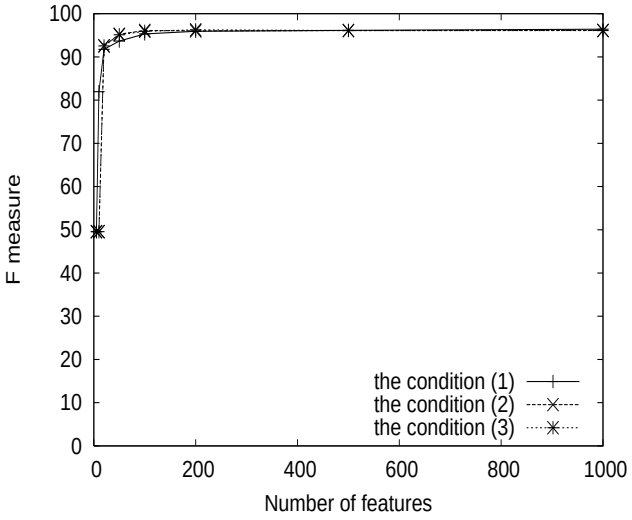
(a) Category "*acq*"



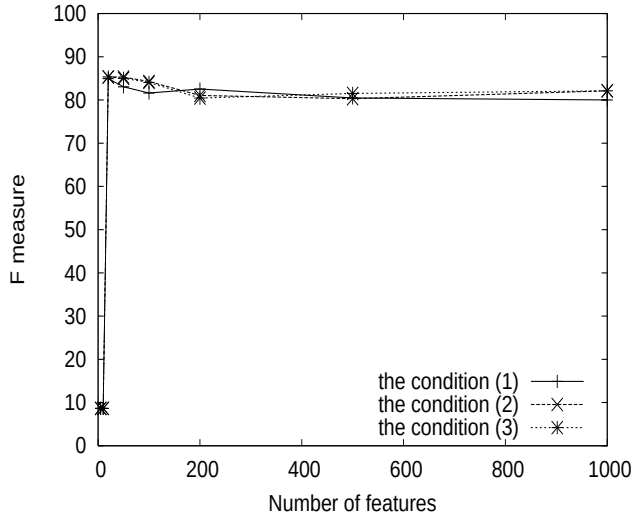
(b) Category "*corn*"



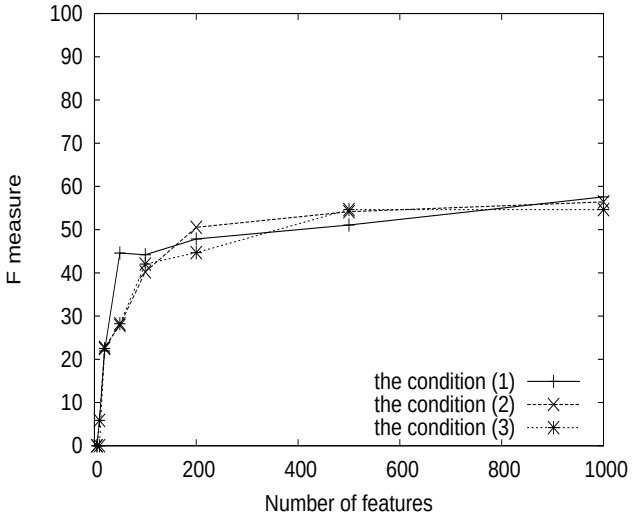
(c) Category "*crude*"



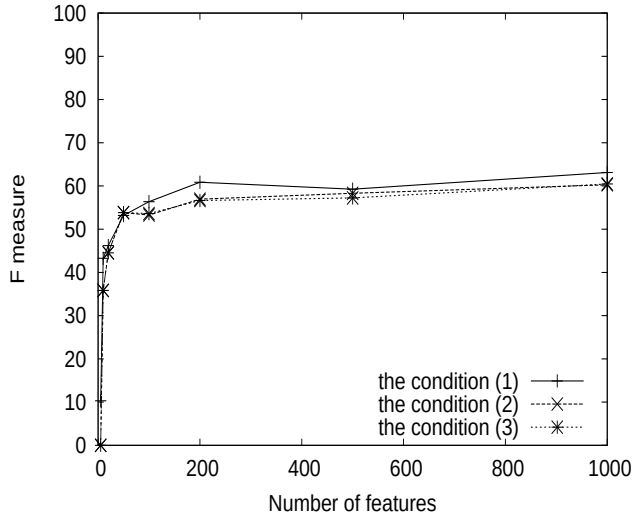
(d) Category "*earn*"



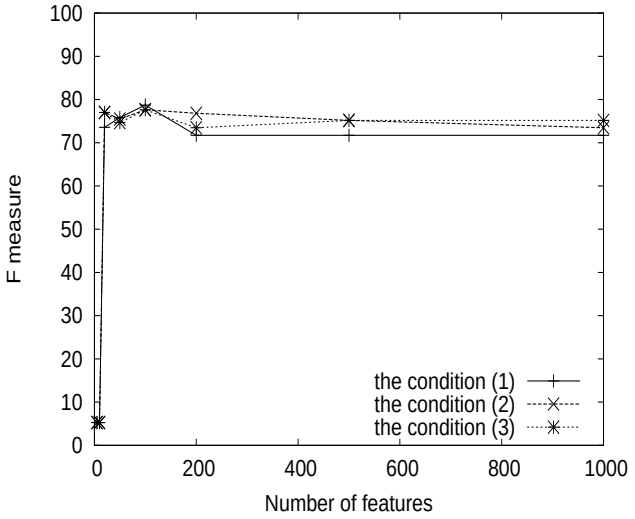
(e) Category "*grain*"



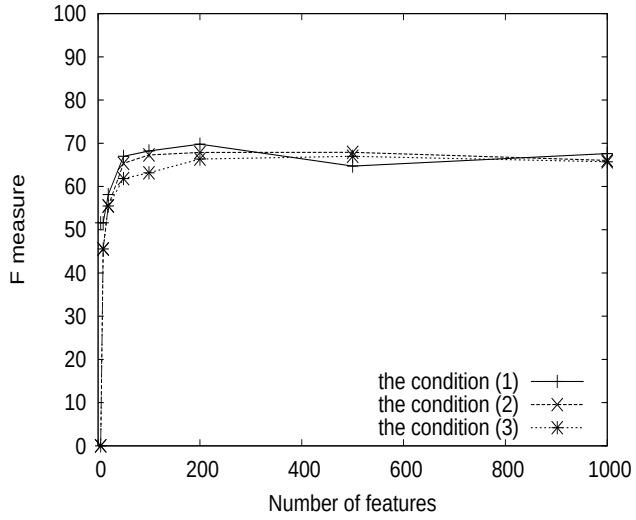
(f) Category "*interest*"



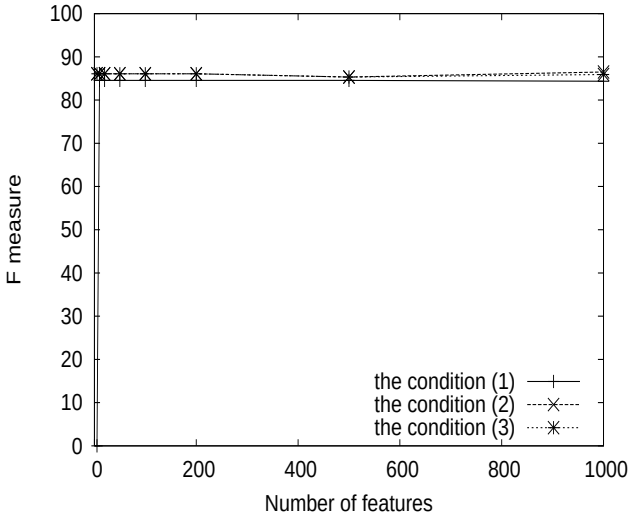
(g) Category "*money-fx*"



(h) Category "*ship*"



(i) Category "*trade*"



(j) Category "*wheat*"

Fig. 5.6 Results of classifying test articles

From comparison of these results with ones of classifying training articles(Fig. 5.4), there are differences in the highest F-measure among categories. We can also observe a characteristic trait observed in the results of classifying training articles in some categories such as "*acq*", "*interest*" and "*money-fx*" indicates , but not observed in other categories such as "*grain*" and "*ship*," which is that the classifiers achieved only low F-measure when we used a small number of features, and achieved high F-measure when we used a large number of features as a whole, although some of them showed locally temporal diminution in F-measure in the latter case, and they had a peak of a constant F-measure when we used more than a certain number of features. We found that in the categories which we could not observe the trait, the classifiers had a peak of the F-measure in a stage of relatively low features and indicated a trait of the F-measure getting unchanged or slightly lower degraded even when we used more features.

Moreover, from comparison of experimental conditions from (1) to (3) in each category, we found a little improvement of the F-measure under experimental conditions (2) and (3) in the category of "*interest*", "*money-fx*", "*ship*" and "*trade*," but found little difference in F-measures under the three experimental conditions in the other categories. Then, we put classification accuracy of all the categories under the experimental condition (3) into a single graph in order to look at the difference in those categories. We show it in Fig. 5.7.

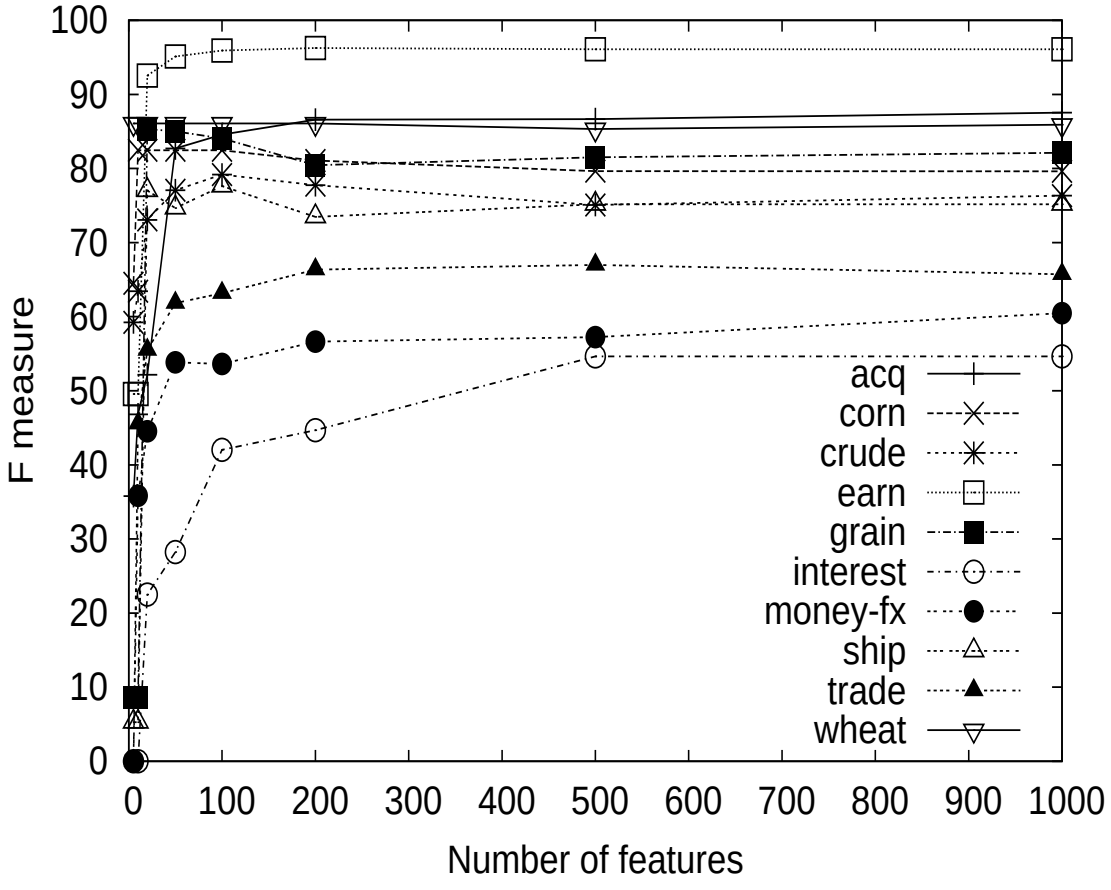


Fig. 5.7 Results of classifying test articles under the experimental condition (3)

From this figure, we found that there were differences in measurements of F-measures among the categories. Although the classifiers achieved considerably high classification accuracy of training articles, which is from 80 to late 90s, they achieved lower F-measures in most of the categories as the results of test articles. We show particularly high and low measures below. Parenthetic numbers are the highest F-measure within the range in features from 5 to 1,000.

- Categories with a high F-measure: "*earn*"(96.08), "*acq*"(87.53)
- Categories with a low F-measure: "*interest*"(54.64), "*money-fx*"(60.48)

5.4 Classification Results with the Polysemy Considered Method

In our method for polysemy reduction which we propose in this study, we have a parameter of a threshold ϵ at the condition equation for determining polysemy(Eq.(3.1)). In order to evaluate our proposal approach, we need to compare F-measure characteristics measured with the approach in fluctuating ϵ in some heuristic way to one measured with the conventional method. However, we did not know what value we should have set for ϵ in each category, and therefore we first made a range of ϵ fixed arbitrarily and conducted an experiment to compare F-measure characteristics to the ones measured with the conventional method. We report its result in the next section Section 5.4.1 .

In Section 5.4.2 , we explain an approach to cut down useless ranges of ϵ in each category, and describe its results. In addition, we took aim at the range of ϵ by using the approach described in Section 5.4.2 , and conducted experiments with ϵ fluctuated minutely to compare with the conventional approach, which we state its result in Section 5.4.3 .

5.4.1 Before Determining a Range of the Threshold

We conducted a comparison experiment between the conventional method and our method for polysemy reduction which we mentioned in Section 3.3 . We set the range of the threshold ϵ to the one from 0.300 to 0.500, switched it at 0.040 intervals within this range, conducted the experiments for classifying test articles with each ϵ, and examined relationships between the number of features and F-measures in these cases. We changed the number of features into 5, 10, 20, 50, 100, 200 and 500.

We show the comparison of the results of classifying with the conventional method and with our proposal method (the one for polysemy reduction) in figures from Fig. 5.8 to Fig. 5.12.

These figures show the experimental results in the category of "*acq*", "*grain*", "*interest*", "*money-fx*" and "*ship*," respectively, and each figure shows the F-measure characteristics in $\epsilon = 0.300, 0.340, 0.380, 0.420 and 0.460$, comparing to the one with the conventional method.

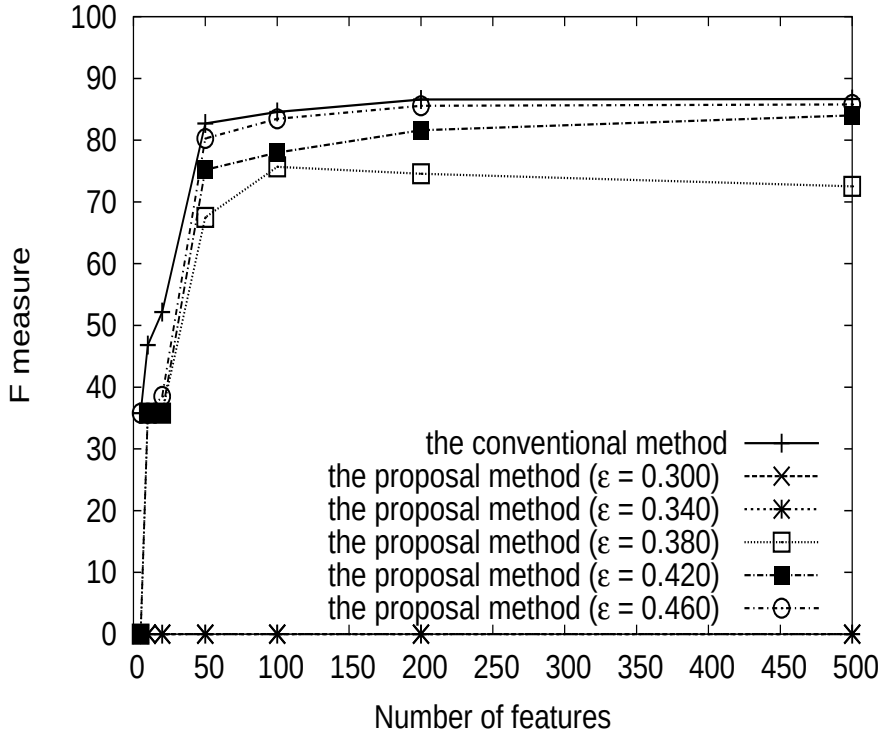


Fig. 5.8 Comparison between our proposal method and the conventional method (category "*acq*") under experimental condition (3) before determining a range of ϵ

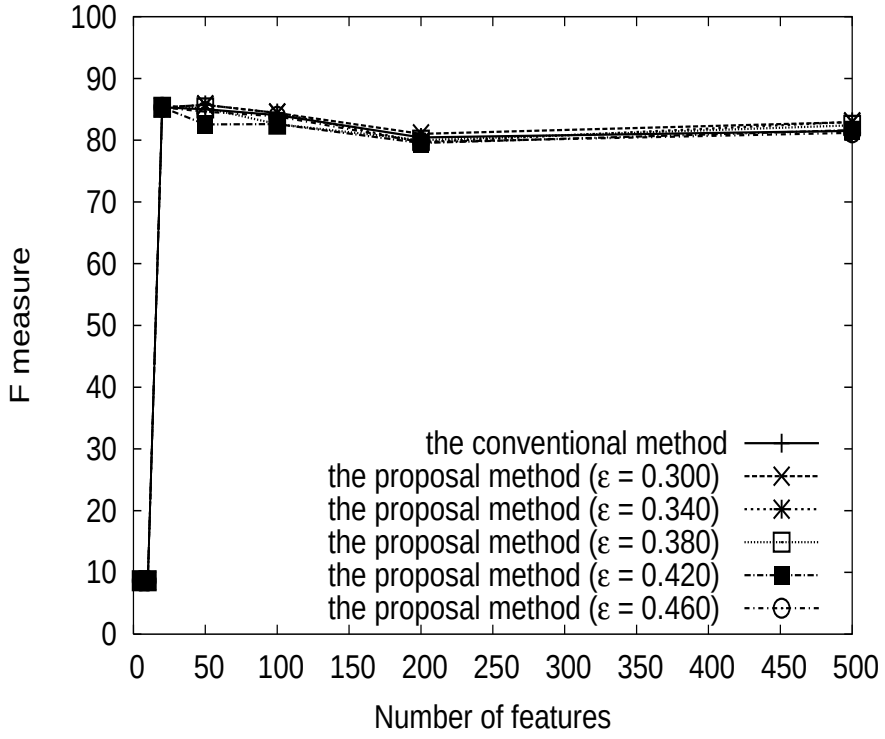


Fig. 5.9 Comparison between our proposal method and the conventional method (category "*grain*") under experimental condition (3) before determining a range of ϵ

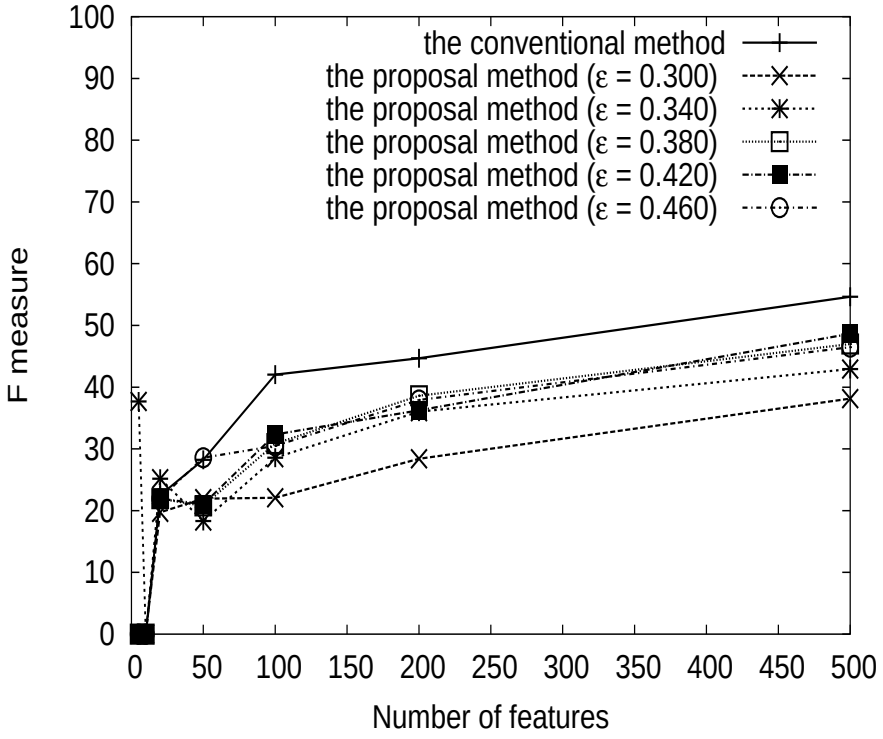


Fig. 5.10 Comparison between our proposal method and the conventional method (category "*interest*") under experimental condition (3) before determining a range of ϵ

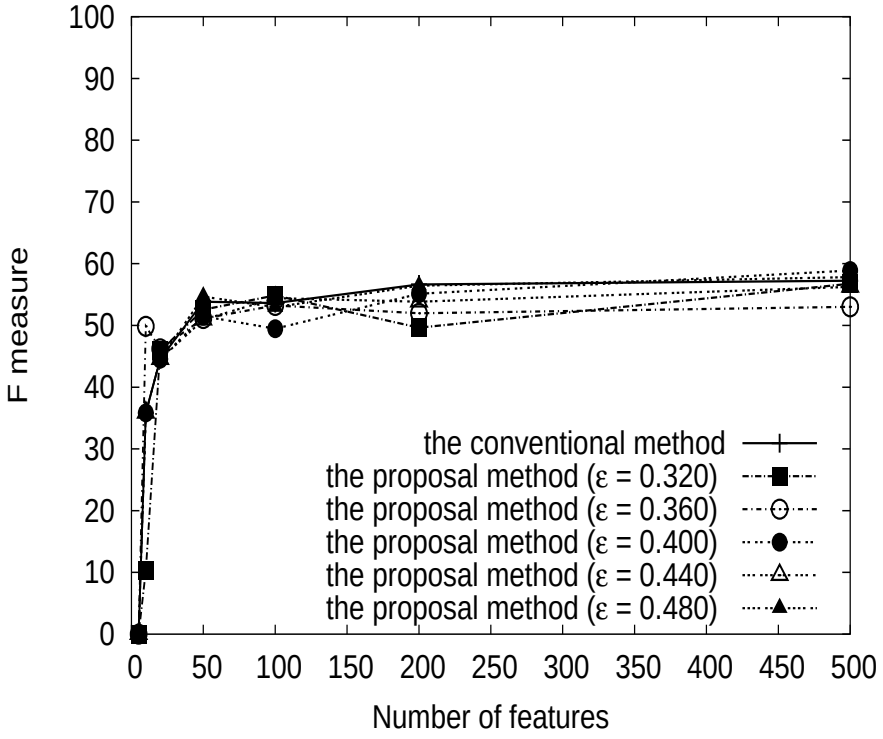


Fig. 5.11 Comparison between our proposal method and the conventional method (category "*money-fx*") under experimental condition (3) before determining a range of ϵ

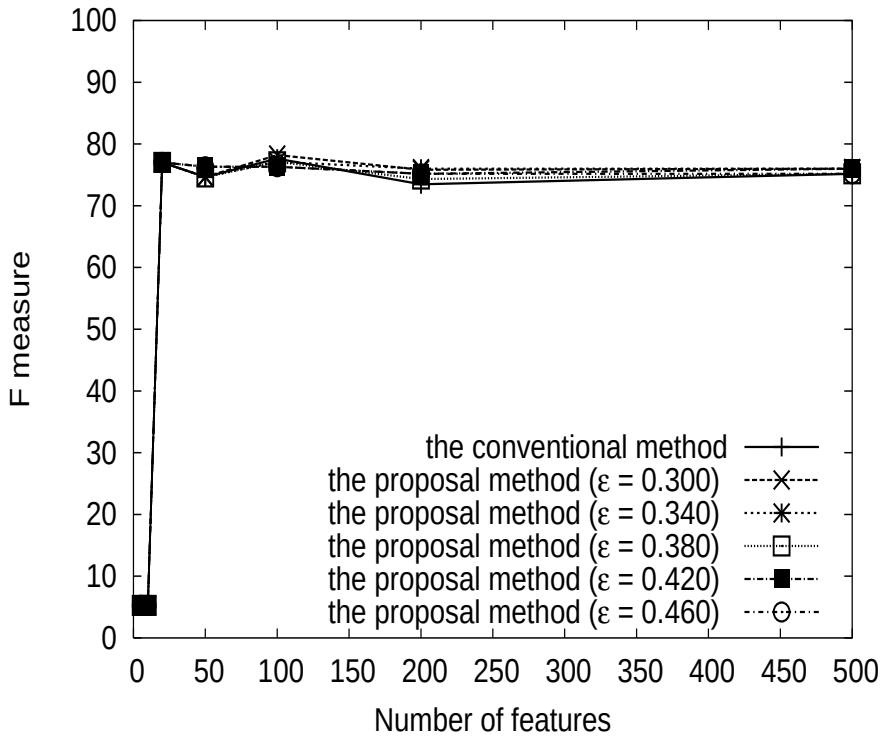


Fig. 5.12 Comparison between our proposal method and the conventional method (category "*ship*") under experimental condition (3) before determining a range of ϵ

F-measures decay as we set a lower ϵ at "*acq*" in Fig. 5.8 and at "*interest*" in Fig. 5.10. Particularly, we had no valid F-measure under the condition of ϵ below 0.340 at "*acq*". On the other hand, we obtained a certain amount of F-measure constantly regardless of the number of features without any effects of ϵ fluctuating at "*grain*" in Fig. 5.9 and at "*ship*" in Fig. 5.12. The situation is different at "*money-fx*" in Fig. 5.11; we found both of increase and decrease of F-measure as we changed the value of ϵ.

The only one case in which we can say slight improvement of F-measure is at "*trade*" in Fig. 5.13. We found F-measure reducing according some numbers of feature, but we confirmed the improvement in the measure by a few points in the case with the number of features 100 ($\epsilon = 0.420$).

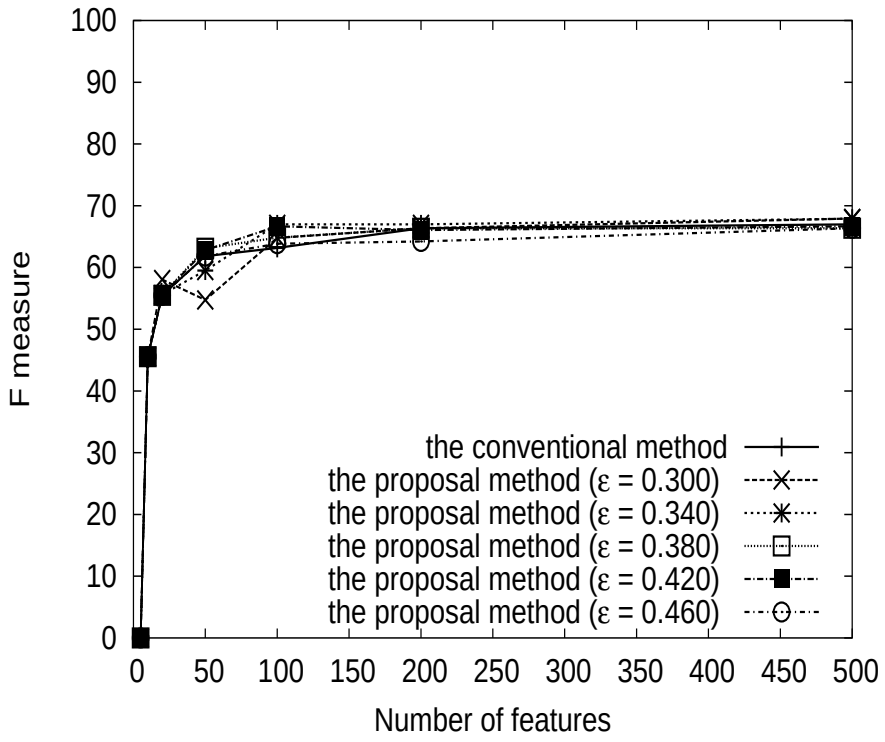


Fig. 5.13 Comparison between our proposal method and the conventional method (category "*trade*") under experimental condition (3) before determining a range of ϵ

5.4.2 Approach for Determining a Range of the Threshold

A method to make a decision on the range of the threshold ϵ which is the condition for determining polysemy is to avoid it being meaningless for each category at least and diminish into the meaningful range.

First, we explain the range of ϵ meaningless for each category. As seen in the previous section, the reason why there was difference in the extent how much the F-measure characteristics changed, even if the fluctuation range of the threshold ϵ is the same among the categories, is that there was difference in polysemy which each feature word has among the categories, even though it has the same rank as to mutual information in each category. For instance, most of words with high mutual information have a polysemous degree of more than 0.340 in "*acq*" in Fig. 5.8, and so the method regards most of high-ranking words as polysemous ones in the range of $\epsilon \leq 0.340$ and removes them from feature sets, which means impossible to classify articles. In contrast, most of words with high mutual information have a polysemous degree less than 0.300 in "*grain*" in Fig. 5.9, and the method does not regard most of high-ranking words as polysemous ones in the range of $\epsilon \geq 0.300$, which causes feature sets to be almost the same as the ones in the conventional method.

Consequently, we can determine the lower limit and upper limit of the range in each category as follows.

- define ϵ_{min} as the ϵ with which all the features are removed for the first time in decreasing the threshold ϵ

- define ϵ_{max} as the ϵ with which at least one feature is removed for the first time in increasing the threshold ϵ

With this definition, the fluctuation range is limited to the range of $\epsilon_{min} \leq \epsilon \leq \epsilon_{max}$.

We show the relationships between ϵ and the number of removed features in every category in Fig. 5.14(a) to Fig. 5.14(f). In these figures, feature sets before removing consists of the 500 words with high mutual information used in the experiment in the previous section. The ranges described by two-headed arrows mean the fluctuation ranges of ϵ mentioned above.

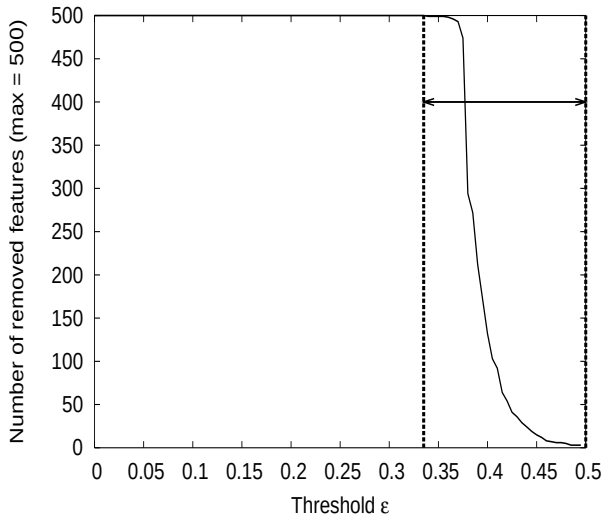
5.4.3 After Determining a Range of the Threshold

We were able to narrow the range of the threshold ϵ by the method to make a decision on the range, described in the previous section. Based on the result, we conducted the comparison experiment between the conventional method and our proposal method about two categories of "*interest*" and "*trade*" by fluctuating ϵ more slightly than doing in the experiment Section 5.4.1 we conducted in the Section 5.4.1 . From the result of Fig. 5.14(c) and (f), we set the fluctuation range of ϵ about "*interest*" and "*trade*" into the range from 0.175 to 0.490 alike, and fluctuated the threshold by 0.005 to obtain the F-measure characteristics within the range. We set the number of features into 5, 10, 20, 50, 100, 200 and 500 as we did in the experiment in Section 5.4.1 .

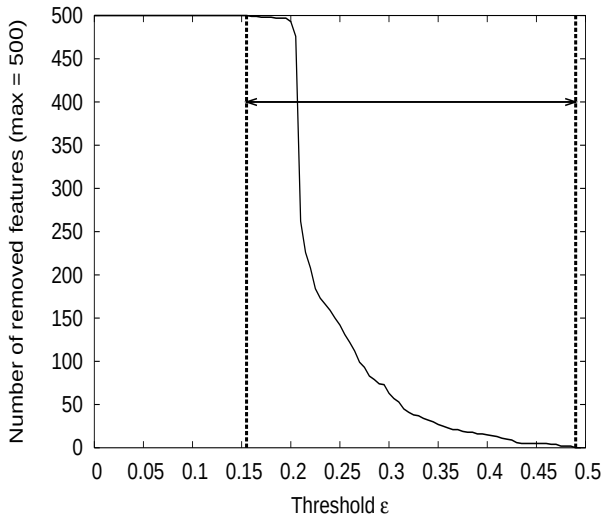
We show the classifying results of this experiment in Fig. 5.15 and Fig. 5.16. As for "*interest*" in Fig. 5.15, we show the case in which we changed ϵ within the fluctuation range. With respect to "*trade*" in Fig. 5.16, we observed an improvement in the F-measure when we changed within the fluctuation range. We particularly observed the improvement in the cases of $\epsilon = 0.330$ and $\epsilon = 0.420$. In order to clarify the difference between our method and the conventional method, we show only the characteristics of these two cases in Fig. 5.16.

The result as to "*interest*" in Fig. 5.15 indicated a high F-measure in a range of very small number of the features, but we did not obtain the F-measures that exceeds those we obtained by the conventional method in the range of more than 50 features. In contrast, the result about "*trade*" in Fig. 5.16 showed the improvement of several points in the F-measure. The F-measure in the case of $\epsilon = 0.330$ and the number of the features is 50 was lower than the one in the conventional method, but in the case where the number of the features is 100, we obtained as much improvement as the one in the case of $\epsilon = 0.420$.

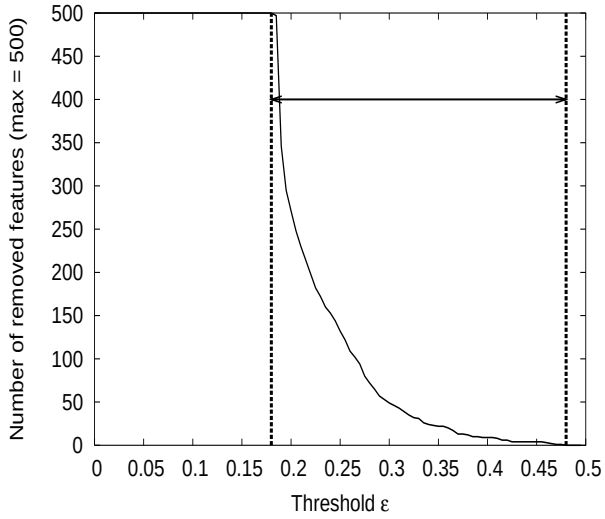
In other categories, for example, we observed a similar tendency in "*acq*" that the F-measure diminished when the number of the features increased as the F-measure about "*interest*" decreased when we observed in Section 5.4.1 . In the categories where the result seemed to show a constant characteristic regardless of the fluctuation of ϵ, in fact, we simply set the lower limit of ϵ into so high that we just observed so under the high ϵ all around, and we were able to observe the tendency of the F-measure characteristics to decrease in the range of lower ϵ than the ones which we set as to "*acq*" and "*interest*."



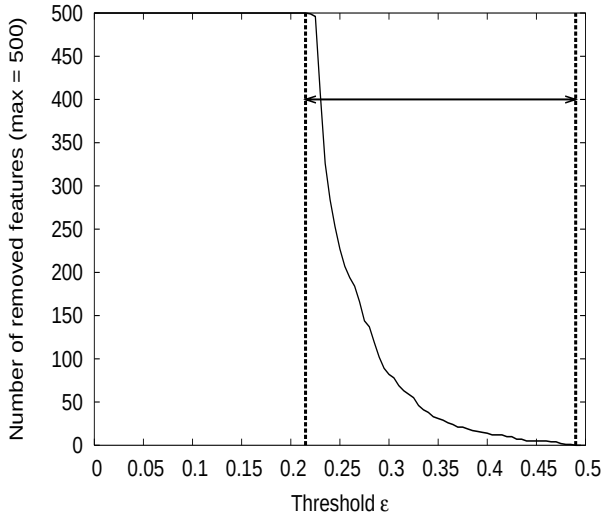
(a) Category "*acq*", experimental condition (3)



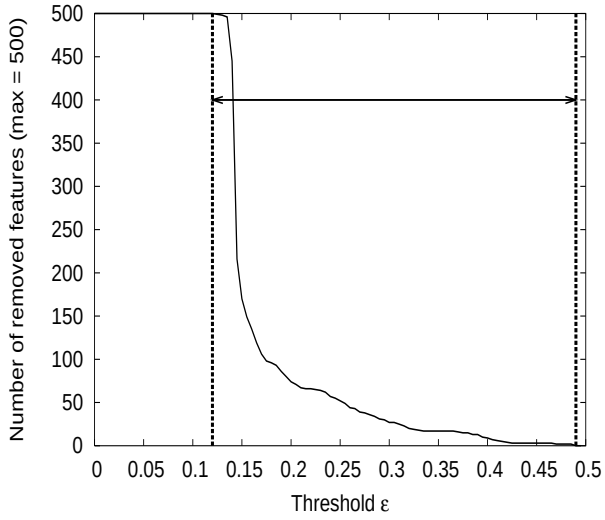
(b) Category "*grain*", experimental condition (3)



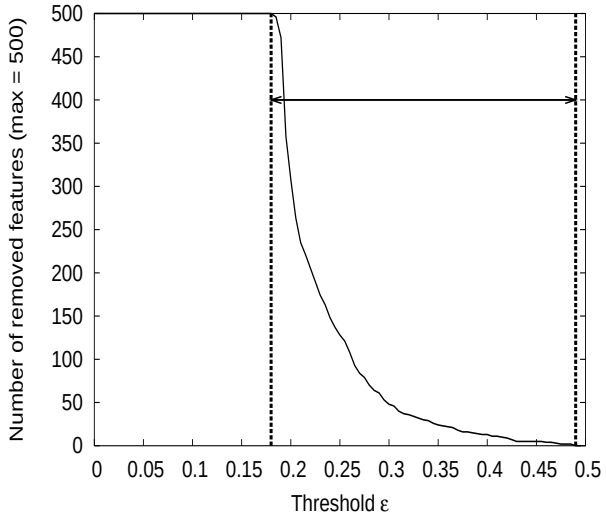
(c) Category "*interest*", experimental condition (3)



(d) Category "*money-fx*", experimental condition (3)



(e) Category "*ship*", experimental condition (3)



(f) Category "*trade*", experimental condition (3)

Fig. 5.14 Relationships between the threshold ϵ and the number of removed features under the experimental condition (3)

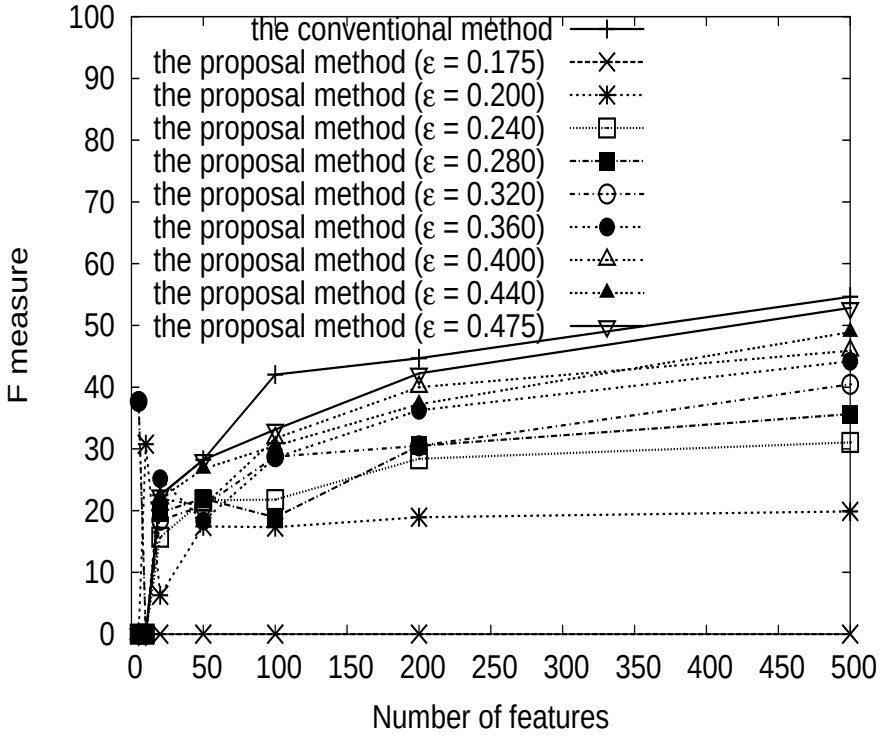


Fig. 5.15 Comparison between our proposal method and the conventional method (category "*interest*") under experimental condition (3) after determining a range of ϵ

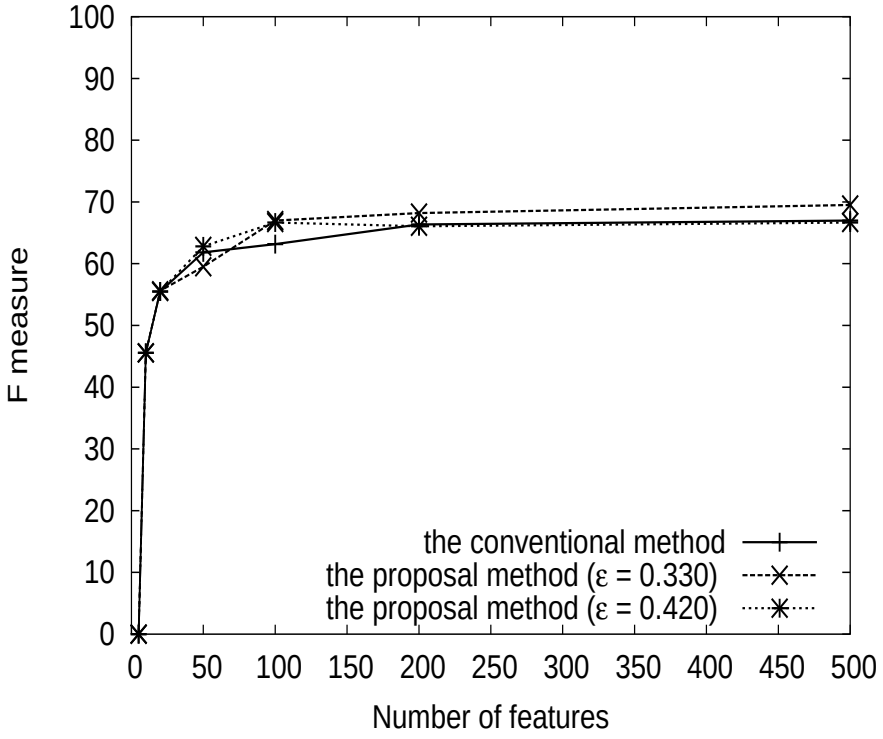


Fig. 5.16 Comparison between our proposal method and the conventional method (category "*trade*") under experimental condition (3) after determining a range of ϵ

Chapter 6

Discussion

6.1 Differences According to Preprocessing

As can be found in Fig. 5.4 and Fig. 5.6, there are little differences in classification accuracy among the experimental conditions (1), (2) and (3) in each category. With respect to this results, we consider two comparisons between the experimental condition (1) and (2), and between (2) and (3).

Comparison between the experimental condition (1) and (2)

The difference between the experimental condition (1) and (2) is whether disable words, word inflection and plurals forms exist or not. First, as for disable words, words such as "and" and "or" exist in any articles in all of the category, and are therefore less likely to be high-ranking words by feature selection using mutual information. We could confirm this respect by looking up those high-ranking words with high mutual information (See Tab. 5.3). Thus, it is likely that the little difference between the results of the experimental condition (1) and (2) supports this filtering effect by feature selection using mutual information.

As far as word inflection and plural forms concerned, there are two cases; a feature word and other words in inflected forms of the word and plural forms have similar mutual information or one-sided mutual information. In the case of one-sided mutual information, feature words with low mutual information are less likely to be high-ranking words, not influencing the classification accuracy. In the case of similar mutual information, it is possible that the feature words exert an influence on the classification accuracy, but we may say that the number of those words are so small, compared to the total number of feature words, that its effect is also small.

Comparison between the experimental condition (2) and (3)

The difference between the experimental condition (2) and (3) is whether numerals exist or not. In general, numerals have more mutual information than disable words, but only a few numerals are included in the high-ranking words. This tendency is most obvious on the categories with a large number of words in training articles (candidate words for features). For instance, in the category of "*earn*" which have the largest number of candidate words for features, there are just 16 numerals in the top 500 words and mere 2 ones in the 100. The categories with which we carried out the classifying processing in the experiments have

a larger number of training articles than other categories and a large number of candidate words for features, and as a result, few numerals were selected as features. Therefore, it seems that numerals has little effect on the results of classifying.

6.2 Classification Results with the Conventional Method

As seen in Section 5.4 , the classifying results of test articles with the conventional method did not show a significant difference among the experimental conditions, as well as the classifying results of training articles. Now, therefore, we consider what Fig. 5.7 tells us.

First of all, we can observe differences in the max value of the F-measure among categories in Fig. 5.7. Particularly, the F-measure characteristics in the category "*earn*" showed enormously higher value than those in other categories. We can explain this by the fact that the category has more feature words with higher mutual infromation, as we can see in Fig. 5.2. For instance, the word "vs" with the most highest mutual information in the category has mutual information of about 0.43. The numeric values in the table of the number of co-occurrence in Tab. 2.2 about the word is the following; $a = 1912, b = 16, c = 965, d = 6710$. The category "*earn*" also has many words with high mutual information other than the word, and it is quite likely that each feature has stronger power to characterize the category than ones in other categories, which results in relatively high classification accuracy.

When we consider it in a feature space, the thing which each feature has stronger power to characterize a category means that the feature has more perceptible tendency regarding whether each document has the vector element of positive and negative examples on the each representative axis or not. Taking all of the features into account comprehensively, we can speculate that the tendency causes the situation in which two clusters of positive and negative examples exist separately in a feature space. This situation is, of course, favorable for some classifiers such as support vector machines, which leads to the improvement of the F-measure as the result.

Next, we would like to discuss the F-measure characteristics in each category. Although F-measures in a few categories reach a peak at a small number of features as we mentioned in Section 5.4 , F-measures in most of categories increase as we set the number of features into small one, and eventually come to stay in a certain value as we set the number of features into large one to some extent. We can explain this as follows in terms of a distribution shape of positive and negative examples in feature spaces.

If we denote the number of features as f, the dimension number of the feature space is also f. In our research, since we set each element of document vectors to a binary value showing whether each feature is included in a document or not, each document is mapped into any of vertices of a hypercube in a feature space. The number of test articles is 3299 as we stated in Section 4.2 , and we need at least 12 features to separate all of the test articles by the following;

$$2^{11} = 2048 < 3299 < 2^{12} = 4096 \tag{6.1}$$

Thus, when we have less than 12 features, we cannot specify each of the 3299 documents separately by the combination of the existence or nonexistence of the features, and therefore positive

and negative examples exist at a vertex in a state of mixture of positive ones, negative ones or the both. The state in which there are vertices where positive and negative examples are located in mixture means that we are not able to separate those examples at the point and improve classification accuracy any more. We can get this state even at more than 12 features, if the number of features are relatively small. When we have substantial features, however, we can specify all of the documents separately by the combination of the existence or nonexistence of the features, and avoid the situation in which we are not able to classify the documents with any classifiers because of the mixture of positive and negative examples in a vertex*, and therefore it is quite likely that we improve classification accuracy in that case.

On the other hand, the reason why the F-measures come to stay in a certain value when we increase the number of features is regarded as follows. When we have features enough to avoid the state of mixture of positive and negative examples, further increase of features does not change any longer the distribution of the positive cluster and the negative one in a significant way, even if it changes slightly. Consequently, it is considered that classification accuracy does not change to a large extent in classifying the documents with classifiers.

6.3 Classification Results with the Polysemy Considered Method

Looking back on the experimental results, it was just about a category "*trade*" that we could observe the improvement of the F-measure by searching a parameter of the threshold ϵ in the experiment before determining a range of the threshold, and the F-measures about other categories changed little or decreased in the experiment. It was difficult for us to observe the alterations of the F-measure characteristics in this experiment because we did not choose the fluctuation range of ϵ in an appropriate way. We therefore introduced a method mentioned in Section 5.4.2 to be able to observe the F-measure characteristics in an appropriate range of ϵ. However, we found out that the F-measures about many categories were likely to decrease by fluctuating ϵ. Even about the only category, "*trade*", which we observed the improvement of the F-measure, the degree of the improvement was just 3 points in the F-measure score.

Let us look into the improvement of the F-measure about "*trade*" in detail when the number of features is 100. The values of A, B, C at the equation to calculate the F-measure(Eq.(2.57)) in the conventional method are the following; $A = 66, B = 26, C = 51$. Meanwhile, the values in the proposal method are the following; $A = 71, B = 24, C = 46$ in the case of $\epsilon = 0.330$, $A = 72, B = 27, C = 45$ in the case of $\epsilon = 0.420$. We expected a decrease of B by reducing polysemy of features, but actually the increase of A and the decrease of C occurred. In this case, feature words removed from the feature set were the following in order of mutual information with "*trade*"; "vs"(the 31st)、"net"(the 61st), "not"(the 64th), "year"(the 73rd), "be"(the 81st), "share"(the 97th). The result indicates that many of the extracted feature words are abstract words rather than polysemous words.

On the other hand, in "*interest*" where we observed the decrease of the F-measure, feature words removed at the point where the number of features is 100 are the following in order of mutual

*See Appendix C .

information; "pct"(the 6th), "company"(the 11th), "share"(the 21th), "net"(the 35th), "year"(the 84th). It is found from the result that our proposal methods removed similar words in "*trade*" and "*interest*." In addition, as far as the values of the equation to calculate the F-measure concerned, the decrease of A is conspicuous.

The reason why the F-measure decreased although similar feature words were removed is generally stated as follows. Feature words include two kind of words which have potentially much or a little relevance between other feature words and themselves, regardless of their polysemy. As mentioned in Section 3.1 , polysemous feature words surely have a bad effect that they take inherent negative examples to a positive cluster in a feature space. However, if they have much relevance to other feature words, it is highly likely that they co-occur with other feature words in documents, and it is possible to separate inherent negative examples including polysemous words from positive ones if co-occurring feature words are univocal. Thus, it is likely that one of the reasons why the F-measure decreased by removing polysemous words in our proposal method is that polysemous words are included in lots of negative documents, but at the same time, some of them characterize lots of positive ones, and we can expect theirs good effects for classifying by the combination with other feature others, and by removing them, the whole of features are less competent to characterize positive documents, which leads to decrease the value of A and the F-measure.

Chapter 7

Future Work

7.1 Utilization of Co-occurrence Among Feature Words

What is first thought of as a future work is a extension of our proposal method based on the consideration in Section 6.3 .

We described a risk of polysemous words in Section 3.1 , referring to a distribution of positive and negative examples in a feature space. In terms of clustering, although one polysemous word has a undesirable effect to take negative documents to a positive cluster, if the word co-occurs with other words frequently in documents which belong to a relevant category, the "partner" words on co-occurrence have effect to keep negative documents far away from a positive cluster. Therefore, we should not remove those polysemous words from feature sets because we can sufficiently expect their functions as features to characterize categories. The features that we should remove for classification are polysemous words which have a little relevance with any other feature words about each category in question.

Thus, we can consider an extended method to add a condition about the co-occurrence relations among words to a condition for determining polysemy in our proposal method, as follows. That is to count the number of co-occurrence among the feature words which we judge as polysemous by applying our proposal method and all of other feature words in the category in question. We would set a threshold about co-occurrence in advance, and we would not regard feature words as the risk factors if the number of co-occurrence about them exceeds the threshold, while we would regard them as ones to be removed if it does not happen.

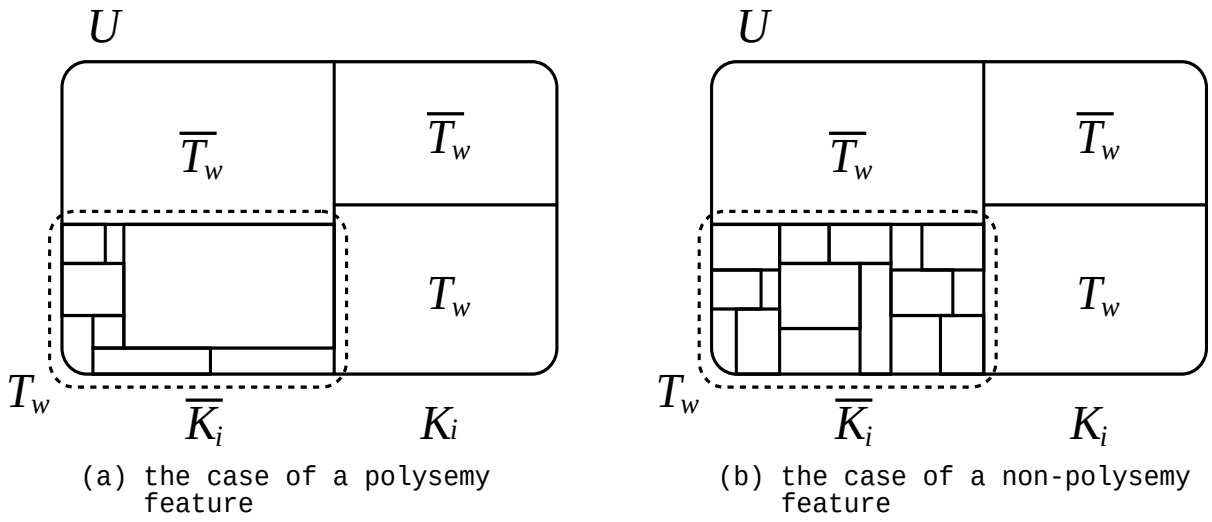
As seen above, it is quite likely that we can clarify questionable polysemous words for classification by the approach to utilize co-occurrence relations among feature words.

7.2 Consideration of Breakdown of Categories

In our research, we determined polysemous words by considering the number of co-occurrence between categories and words, but it is proper to extend our proposal method by adding a condition to consider the breakdown about how many categories, other than the category in question, do documents including polysemous features belong to?

A polysemous word has intrinsically multiple meanings which are entirely different from each other, as a word "plant" has meanings of vegetation and factories. Thus, since polysemous words

have much relevance to other categories by different meanings from the original meaning by which they have much relevance to a category in question, it is likely that there are many documents which include those words and belong to the categories other than the category in question. In contrast, in the case of univocal words, documents including them can belong to multiple categories in small amounts, though there should be a little amounts even if the documents belong to those categories. We show these situations related to polysemous words and non-polysemous words in Fig. 7.1.



(a) the case of a polysemy feature

(b) the case of a non-polysemy feature

Fig. 7.1 Document sets in considering the breakdown of categories which documents belong to

When we define a condition for determining polysemy in our proposal method, we considered only the "number" of documents which include words as to a category in question and other categories, and so we dealt with two situations mentioned above as the same. Thus, it seems effective that in addition to our proposal method, we would make the condition for determining polysemy more strict, considering more detailed information such as the number of categories, other than a category in question, which the documents including "polysemous" words belong to, and the number of documents which belong to those categories.

Chapter 8

Conclusion

In the past, the attempts to select words with highly evaluated values based on some measures, such as mutual information between categories and words, have been made at selection of feature words which characterize categories in text classification. In our research, we proposed a method to remove polysemous words from feature sets by focusing attention on the number of co-occurrence among categories and words on the problem about polysemy of feature words. We observed an improvement of classification accuracy in a particular condition compared with the conventional method. We strongly hope that we would contribute to solution of disambiguation problems by further amelioration being carried out based on our proposal method and more general improvement of classification accuracy being putting into practice.

Acknowledgements

I would like to thank sincerely Associate Prof. Masaya Nakayama in the Information Technology Center at the University of Tokyo, who has always given me painstaking and considerate instruction from both of a panoramic view and a localized view.

I would like to thank deeply Prof. Yasushi Wakahara in the Information Technology Center at the University of Tokyo, who has given me various valuable advice and instruction about not only my research content but also my attitude toward research, mainly in our laboratory meetings. In addition, I also would like to thank Dr. Fumitaka Nakamura in the Information Technology Center at the University of Tokyo, who has given me different valuable advice about our proposal method, essential comprehension of the SVM algorithm and improving the efficacy in implementation of programs.

Prof. Masaru Kitsuregawa and Prof. Hiroshi Nakagawa gave me professional suggestions, advice and instruction in considering a direction of our research. I sincerely appreciate them.

I wish to acknowledge the following colleagues who have discussed this work and provided useful comments and suggestions about how to write a master thesis and life for research: Mr. Tomofumi Matsuzawa at the Ph.D. program in the School of Engineering at the University of Tokyo, Mr. Kunitake Kaneko at the Master's program in the school, Mr. Kenichiro Ishijima, Mr. Toshiaki Ogawa, Mr. Hiroshi Nagatake, Mr. Hitohiko Sato, Mr. Kohei Sugiyama and Mr. Kengo Takahashi at the Master's program in the Graduate School of Frontier Sciences at the University of Tokyo. Mr. Takeshi Ueda, Mr. Yuji Kamite and Mr. Kentaro Saito who graduated from the Master's program in the School of Engineering at the University of Tokyo last year also discussed this work and gave me useful advice about life for research. I wish to appreciate them, too.

Additionally, thanks a lot for secretaries in our laboratory helping paperwork for purchasing materials, attending academic conferences and so forth.

Finally, I appreciate my family for supporting my student life.

January 31, 2003

References

[1] Kjersti Aas, Line Eikvil, "Text Categorisation: A Survey", Report No.941, ISBN 82-539-042-B, 1999

[2] Claus Bahlmann, Bernard Haasdonk, Hans Burkhardt, "On-line Handwriting Recognition with Support Vector Machines — A Kernel Approach", *Proc. 8th IWFHR*, pp.49-54, 2002

[3] Corinna Cortes, Vladimir Vapnik, "Support-Vector Networks", *Machine Learning*, Vol.20, No.3, pp.273-297, 1995

[4] Oliver Chapelle, Patrick Haffner, Vladimir Vapnik, "SVMs for Histogram-Based Image Classification", *IEEE Trans. on Neural Networks*, 1999

[5] Scott Deer, Susan T. Dumais, George W. Furnas, Thomas K, Laudauer, Richard Harshman, "Indexing by Latent Semantic Analysis", *Journal of the American Society for Information Science*, Vol.41, No.6, pp.391-407, 1990

[6] Susan Dumais, John Platt, David Heckerman, "Inductive Learning Algorithm and Representations for Text Categorization", *Proc. 7th International Conference on Information and Knowledge Management*, 1998

[7] A. Ganapathiraju, J. Hamaker, J. Picone, "Hybrid SVM/HMM Architectures for Speech Recognition", *Proc. ICSLP*, Vol.4, 2000

[8] Thorsten Joachims, "Text Categorization with Support Vector Machine: Learning with Many Relevant Features", *Proc. 10th European Conference on Machine Learning*, pp.137-142, 1998

[9] Thorsten Joachims, 11 in: "Making Large-Scale SVM Learning Practical", Advances in Kernel Methods - Support Vector Learning, MIT Press, 1999

[10] Thorsten Joachims, "Transductive Inference for Text Classification using Support Vector Machines", *Proc. 16th International Conference on Machine Learning*, pp.200-209, 1999

[11] James Tin-Yau Kwok, "Automated Text Categorization Using Support Vector Machine", *Proc. the International Conference on Neural Information Processing*, pp.347-351, 1999

[12] Hwee Tou Ng, Hian Beng Lee, "Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach", *Proc. 34th Annual Meeting of the Association for Computational Linguistics*, pp.40-47, 1996

[13] Fabrizio Sebastiani, "A Tutorial on Automated Text Categorisation", *Proc. 1st Argentinian Symposium on Artificial Intelligence*, pp.7-35, 1999

[14] Vladmir N. Vapnik, "The Nature of Statistical Learning Theory – Second Edition", Springer-Verlag New York, Inc, 2000

[15] Yiming Yang, Jan O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization", *Proc. 14th International Conference on Machine Learning*, pp.412-420, 1997

[16] Yiming Yang, "An Evaluation of Statistical Approaches to Text Categolization", CMU-CS Report, CMU-CS-97-127, 1997

[17] Yiming Yang, Xin Liu, "A Re-examination of Text Categorization Methods", *Proc. SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*(Berkeley, US), pp.42-49, 1999

[18] David Yarowsky, "Word-Sense Disambiguation using Statistical Models of Roget's Categories Trained on Large Corpora", *Proc. 14th International Conference on Computational Linguistics*, pp.454-460, 1992

[19] David Yarowsky, "Unsupervised Word Sense Disambiguation Rivaling Supervised Methods", *Proc. 33th Annual Meeting of the Association for Computational Linguistics*, pp.189-196, 1995

[20] Fumiyo Fukumoto, "Toward Optimal Feature Selection for Word Sense Disambiguation (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 2001-NL-141-12, pp.69-76, 2001

[21] Masaaki Nagata, Hirotoshi Taira, "Text Classification : Showcase of Learning Theories(¡Special Features¿Information-based Induction Sciences) (in Japanese)", *Information Processing Society of Japan*, Vol.42, No.1, pp.32-37, 2001

[22] Hirotoshi Taira, Masahiko Haruno, "Text Categorization Using a Transductive Boosting Method (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 2000-NL-139-10, pp.69-76, 2000

[23] Hirotoshi Taira, Takafumi Mukouchi, Masahiko Haruno, "Text Categorization Using Support Vector Machines (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 1998-NL-128-24, pp.173-180, 1998

[24] Hiroyasu Yamada, Taku Kudo, Yuji Matsumoto, "Japanese Named Entity Extraction using Support Vector Machines (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 2001-NL-142-17, pp.121-128, 2001

[25] Hiroyasu Yamada, Yuji Matsumoto, "Applying Support Vector Machine to Multi-Class Classification Problems (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 2001-NL-146-6, pp.33-38, 2001

[26] Takefumi Yamazaki, Ido Dagan, "Mistake-driven learning with thesaurus for text categorization (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 1997-NL-120-4, pp.89-96, 1997

[27] Natsuki Yuasa, Toru Ueda, Fumio Togawa, "Classifying Articles Using Lexical Co-occurrences in Large Document Databases (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 1993-NL-98-4, pp.81-88, 1993

[28] Kenji Kita, "Probabilistic Language Model (in Japanese)", University of Tokyo Press, 1999

[29] Takenobu Tokunaga, "Information Retrieval and Language Processing (in Japanese)", University of Tokyo Press, 1999

[30] "Reuters-21578 Text Categorization Test Collection", http://www.daviddlewis.com/resources/testcollections/reuters21578/

[31] "Support Vector Machine", http://vision.ai.uiuc.edu/mhyang/svm.html

[32] "SVM-Light Support Vector Machine", http://cs.cornell.edu/People/tj/svm_light/

[33] "TreeTagger", http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

[34] "Yahoo!", http://www.yahoo.com/

[35] "Search Engines with Robots and Directories (in Japanese)", http://www.seojapan.com/kensakuengine/search-engine.html

[36] "Researchers' Views: Consideration of Information Retrieval Focusing Retrieval Precision (in Japanese)", http://www.nikkei-r.co.jp/report/9904/kensaku.htm

Presented Papers

[1] Jun Araki, Fumitaka Nakamura, and Masaya Nakayama, "Automated Categorization of Newspaper Articles using Sectorial Dictionary with Relevant Terms (in Japanese)", *In Proceedings of the Forum on Information Technology 2002 (FIT2002)*, E-11, pp.103-104, 2002

[2] Jun Araki, Fumitaka Nakamura, and Masaya Nakayama, "Text Classification with a Polysemy Considered Feature Set (in Japanese)", *Information Processing Society of Japan, Special Interest Group on Natural Language Processing (IPSJ-SIGNL)*, 2003 (to appear)

Appendix A

Reuters-21578 Corpus

The Reuters-21578 corpus is a text corpus classified into categories by tagging articles from Reuters, and able to be utilized by downloading* on the Internet. This corpus consists of 22 files, and each of 21 files (from reut2-000.sgm to reut2-020.sgm) include 1,000 articles and the last file (reut2-021.sgm) includes 578 ones. The aggregation and classification was done by the workers at Reuters. Reuters-21578 is an improved version of Reuters-22173 corpus available in January 1993.

A.1 File Format

Each of the 22 files begins with the following sentence of a Document Type Declaration(DTD).

```
<!DOCTYPE lewis SYSTEM ‘‘lewis.dtd’’>
```

Each article begins with the following a start tag.

```
<REUTERS TOPICS=?? LEWISSPLIT=?? CGISPLIT=?? OLDID=?? NEWID=??>
```

A mark of “??” is assigned each attribute value. The article which begins with the start tag ends with the following a end tag.

```
</REUTERS>
```

Each REUTERS tag has five attributes of TOPICS, LEWISSPLIT, CGISPLIT, OLDID and NEWID. Values and their meanings about each attribute are shown in Tab. A.1.

In our research, we split the corpus into training data and test data by the most-used way for splitting, called “ModApte”. This way for splitting is based on the selection of parameters shown in Tab. A.2.

A.2 Internal Tags in the Articles

As <REUTERS >and </REUTERS > determine the range of an article in each file, other tags are used to determine each element about the article. We show that in Tab. A.3. Each element tagged with these tags can include multiple values with a delimiter of <D >.

In our research, we only focused attention on <TOPICS >and <TEXT > out of these.

*The URL for downloading is the following; http://www.daviddlewis.com/resources/testcollections/reuters21578/

Tab. A.1 Attribute values and their meanings of Reuters tags

Attribute Name	Attribute Value	Meaning
TOPICS	**YES**	belong to at least one category
	NO	does not belong to any categories
	BYPASS	labeled as "bypass" (or its anomaly) in original data before
LEWISSPLIT	**TRAIN**	training data
	TEST	test data
	NOT-USED	not used data
CGISPLIT	**TRAINING-SET**	training data
	PUBLISHED-TESTSET	test data
OLDID	(Old Article ID)	Article ID in the period of Reuters-22173
NEWID	(New Article ID)	Article ID (in chronological order)

Tab. A.2 Splitting into training data and test data by "ModApte"

Type of articles	The number of articles	Conditions of attributes	
Train articles	9603	**LEWISSPLIT="TRAIN"**	**TOPICS="YES"**
Test articles	3299	**LEWISSPLIT="TEST"**	**TOPICS="YES"**
Not used articles	8676	**LEWISSPLIT="NOT-USED"**	**TOPICS="YES"**
		LEWISSPLIT="NO"	
		LEWISSPLIT="BYPASS"	

Tab. A.3 Internal tags except <REUTER >

Tag name	Meaning of tags
DATE	Date of the article
MKNOTE	Memo in revising
TOPICS	Category which the article belongs to
PLACES	Place which the article belongs to
PEOPLE	Person which the article belongs to
ORGS	Organization which the article belongs to
EXCHANGES	Exchange which the article belongs to
COMPANIES	Company which the article belongs to
UNKNOWN	Noise or nonsensical part in the article
TEXT	Body text of the article

A.3 Example of the Articles

The initial part of Reuters-21578 is as shown below.

```
<!DOCTYPE lewis SYSTEM "lewis.dtd">
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID
="5544" NEWID="1">
<DATE>26-FEB-1987 15:01:01.79</DATE>
<TOPICS><D>cocoa</D></TOPICS>
<PLACES><D>el-salvador</D><D>usa</D><D>uruguay</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
\&\#5;\&\#5;\&\#5;C T
\&\#22;\&\#22;\&\#1;f0704\&\#31;reute
u f BC-BAHIA-COCOA-REVIEW   02-26 0105</UNKNOWN>
<TEXT> \&\#2;
<TITLE>BAHIA COCOA REVIEW</TITLE>
<DATELINE>    SALVADOR, Feb 26 - </DATELINE> <BODY> Showers continued
throughout the week in the Bahia cocoa zone, alleviating the drought
since early January and improving prospects for the coming temporao,
although normal humidity levels have not been restored, Comissaria
Smith said in its weekly review.
     The dry period means the temporao will be late this year.
.
.
     Final figures for the period to February 28 are expected to be
published by the Brazilian Cocoa Trade Commission after carnival which
 ends midday on February 27.
\ Reuter
\&\#3;</BODY> </TEXT>
</REUTERS>
<REUTERS TOPICS="NO" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID
="5545" NEWID="2">
.
.
```

Appendix B

TreeTagger

TreeTagger is a tool for annotating text with part-of-speech, developed by Dr. Helmet Schmidt at the University of Stuttgart. It is adaptable in five languages of German, English, French, Italian and Russian. We can utilize it by downloading* on the Internet.

B.1 Function

The function of TreeTagger is to analyze input sentences with the parts of speech shown in Section B.2 and tag the parts of speech. For example, we input the following sentence;

```
The TreeTagger is easy to use.
```

Then, TreeTagger outputs as follows.

```
The          DT    the
TreeTagger   NP    TreeTagger
is           VBZ   be
easy         JJ    easy
to           TO    to
use          VB    use
.            SENT  .
```

B.2 List of Parts of Speech

The list of parts of speech tagged by TreeTagger in English version is the following shown in Tab. B.1.

*The URL for downloading is the following;
http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

Tab. B.1 List of parts of speech tagged by TreeTagger

1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass
13.	NNS	Noun, plural
14.	NP	Proper noun, singular
15.	NPS	Proper noun, plural
16.	PDT	Predeterminer
17.	POS	Possessive ending
18.	PP	Personal pronoun
19.	PP$	Possessive pronoun
20.	RB	Adverb
21.	RBR	Adverb, comparative
22.	RBS	Adverb, superlative
23.	RP	Particle
24.	SYM	Symbol
25.	TO	*to*
26.	UH	Interjection
27.	VB (VH, VV)	Verb, base form
28.	VBD (VHD, VVD)	Verb, past tense
29.	VBG (VHG, VVG)	Verb, gerund or present participle
30.	VBN (VHN, VVN)	Verb, past participle
31.	VBP (VHP, VVP)	Verb, non-3rd person singular present
32.	VBZ (VHZ, VVZ)	Verb, 3rd person singular present
33.	WDT	Wh-determiner
34.	WP	Wh-pronoun
35.	WP$	Possessive wh-pronoun
36.	WRB	Wh-adverb
37.	#	*#* symbol
38.	$	*$* symbol
39.	SENT	period
40.	,	comma
41.	:	colon
42.	(	left parenthesis
43.	)	right parenthesis
44.	”	two single quotation marks, right
45.	“	two single quotation marks, left

Appendix C

SVM^{light}

SVM^{light} is the C program for Support Vector Machine, developed by Thorsten Joachims, et al. at Cornell University. We can utilize it by downloading* on the Internet.

C.1 Overview

The main features of SVM^{light} are the following.

- fast optimization algorithm
  - working set selection based on steepest feasible descent
  - "shrinking" heuristic
  - caching of kernel evaluations
  - use of folding in the linear case
- solves classification, regression and ranking problems.
- computes estimates of the error rate $\xi - \alpha$, the precision, and the recall
- efficiently computes Leave-One-Out estimates of the error rate, the precision, and the recall
- includes algorithm for approximately training large transductive SVMs (TSVMs)
- can train SVMs with cost models and example dependent costs
- handles many thousands of support vectors
- handles several hundred-thousands of training examples
- supports standard kernel functions and lets you define your own
- uses sparse vector representation

*The URL for downloading is the following: http://cs.cornell.edu/People/tj/svm_light/

C.2 How to Use

C.2.1 Making Document Vector Files

SVM^{light} consists of a learning module (svm_learn) and a classification module (svm_classify). The classification module can be used to apply the learned model to new examples. We have to make vector files of training data and test data for the learning module and the classification module. We have to write the vector files by following the description rule below.

The first lines may contain comments and are ignored if they start with #. Each of the following lines represents one training example and is of the following format:

```
<class> .=. +1 | -1 | 0
<feature> .=. integer
<value> .=. real
<line> .=. <class> <feature>:<value> <feature>:<value> .. <feature>:<
value>
```

The target value and each of the feature/value pairs are separated by a space character. Feature/value pairs must be ordered by increasing feature number. Features with value zero can be skipped. +1 as the target value marks a positive example, -1 a negative example respectively. So, for example, the line

```
-1 1:0.43 3:0.12 9284:0.2
```

specifies a negative example for which feature number 1 has the value 0.43, feature number 3 has the value 0.12, feature number 9284 has the value 0.2, and all the other features have value 0.

C.2.2 Execution Commands and Options

As mentioned in C.2.1, there are a learning module (svm_learn) and a classification module (svm_classify) in SVM^{light}. We explain the commands and options of svm_learn and svm_classify. We execute the command of svm_learn as follows:

```
$ svm_learn [options] example_file model_file
```

Available options with svm_learn are the following:

```
General options:
-?           - this help
-v [0..3]    - verbosity level (default 1)
Learning options:
-z {c,r,p}   - select between classification (c), regression (r), and
               preference ranking (p) (default classification)
-c float     - C: trade-off between training error
               and margin (default [avg. x*x] \textasciicircum -1)
-w [0..]     - epsilon width of tube for regression (default 0.1)
-j float     - Cost: cost-factor, by which training errors on
               positive examples outweight errors on negative examples
               (default 1)
-b [0,1]     - use biased hyperplane (i.e. x*w+b0) instead
               of unbiased hyperplane (i.e. x*w0) (default 1)
```

```
-i [0,1]    - remove inconsistent training examples and retrain
              (default 0)
Performance estimation options:
-x [0,1]    - compute leave-one-out estimates (default 0)
-o [0..2]   - value of rho for XiAlpha-estimator and for pruning
              leave-one-out computation (default 1.0)
-k [0..100] - search depth for extended XiAlpha-estimator (default
              0)
Transduction options:
-p [0..1]   - fraction of unlabeled examples to be classified into the
              positive class (default is the ratio of positive and
              negative examples in the training data)
Kernel options:
 -t int     - type of kernel function:
               0: linear (default)
               1: polynomial (s a*b+c)\textasciicircum d
               2: radial basis function exp(-gamma ||a-b|| ^ 2)
               3: sigmoid tanh(s a*b + c)
               4: user defined kernel from kernel.h
 -d int     - parameter d in polynomial kernel
 -g float   - parameter gamma in rbf kernel
 -s float   - parameter s in sigmoid/poly kernel
 -r float   - parameter c in sigmoid/poly kernel
 -u string  - parameter of user defined kernel
Optimization options:
 -q [2..]   - maximum size of QP-subproblems (default 10)
 -n [2..q]  - number of new variables entering the working set
              in each iteration (default n = q). Set n<q to prevent
              zig-zagging.
 -m [5..]   - size of cache for kernel evaluations in MB (default 40)
              The larger the faster...
 -e float   - eps: Allow that error for termination criterion
              [y [w*x+b] - 1] = eps (default 0.001)
 -h [5..]   - number of iterations a variable needs to be
              optimal before considered for shrinking (default 100)
 -f [0,1]   - do final optimality check for variables removed by
              shrinking. Although this test is usually positive,
              there is no guarantee that the optimum was found if
              the test is omitted. (default 1)
Output options:
 -l char    - file to write predicted labels of unlabeled examples
              into after transductive learning
 -a char    - write all alphas to this file after learning (in the
              same order as in the training set)
```

In all modes, the result of svm_learn is the model which is learned from the training data in example_file of the first argument. This learning model is written to model_file. To make predictions on test examples, svm_classify reads this file. We execute the command of svm_classify as follows:

```
$ svm_classify [options] example_file model_file output_file
```

Available options with svm_classify are the following:

```
-h         Help.
-v [0..3]  Verbosity level (default 2).
-f [0,1]   0: old output format of V1.0
           1: output the value of decision function (default)
```

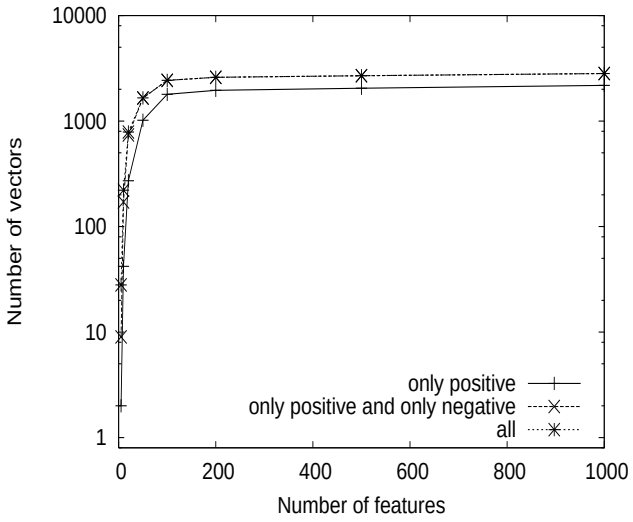
Appendix D

Distribution of Document Vectors in Feature Spaces

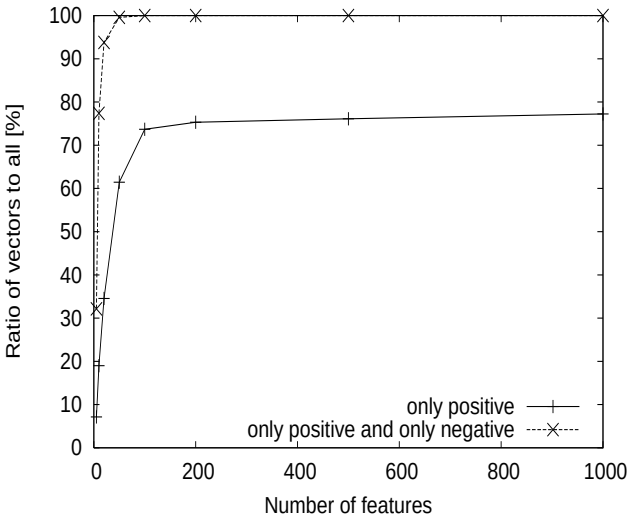
In order to know whether the mixture of positive and negative examples in feature space remains or not when we increase the number of features, we conducted an experiment to count the number of points occupied by positive and negative examples in test articles out of all the possible grid points in a feature space. We show the result about the categories "*acq*", "*earn*" and "*trade*" in Fig. D.1, Fig. D.2 and Fig. D.3, respectively.

In the left figures of those, "only positive" stands for the number of the points where only positive examples exist, "only positive and only negative" for the sum of the number of the points where only positive examples exist and the number of the points where only negative ones exist, and "all" for the number of all of the points where documents are mapped. In the right figures, "only positive" represents the ratio of the points where only positive examples exist to the number of all of the points where documents are mapped, and "only positive and only negative" represents the ratio of the sum of the number of the points where only positive examples exist and the number of the points where only negative ones exist to the number of all of the points where documents are mapped. We set the number of all of the points where documents are mapped to 100[%] in the right figures.

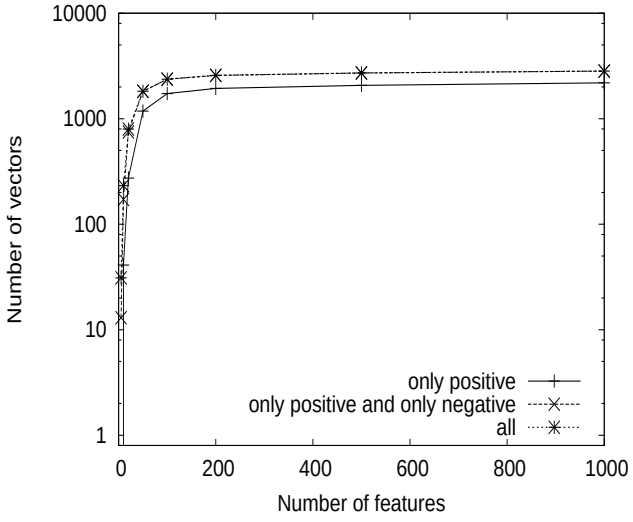
As found from these right figures, the ratio as to "only positive and only negative" reaches 100 in every category when the number of feature is between 50 and 100, and at that time, positive documents and negative ones are completely separated in vertices on a hypercube in a feature space. We observed the same result in the categories other than three categories show in the figures from Fig. D.1 to Fig. D.3.



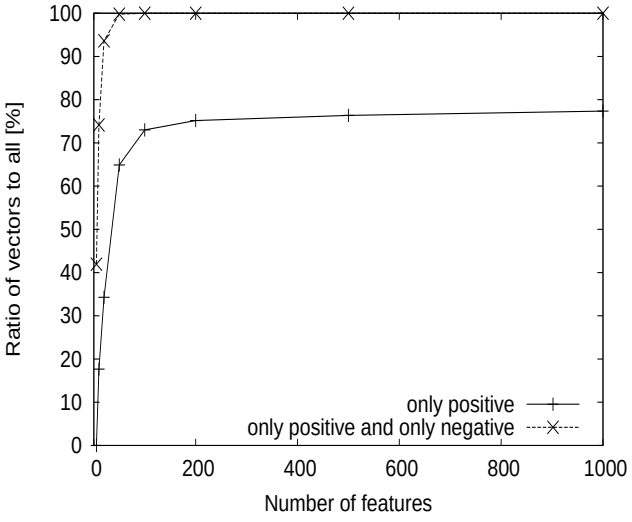
(a) Category "*acq*", experimental condition (1)



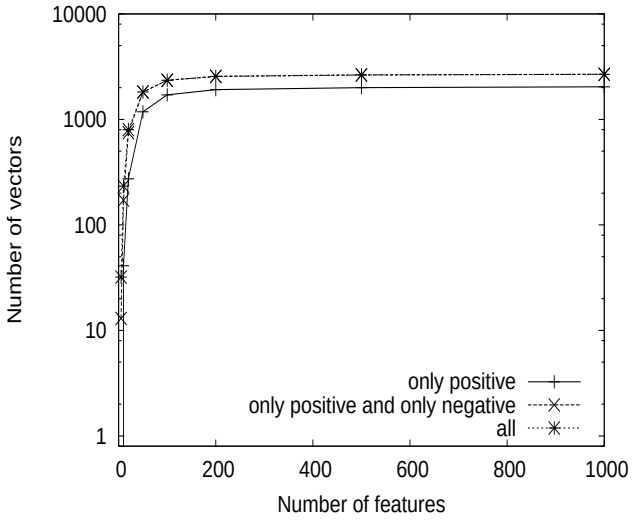
(a′) Category "*acq*", experimental condition (1)



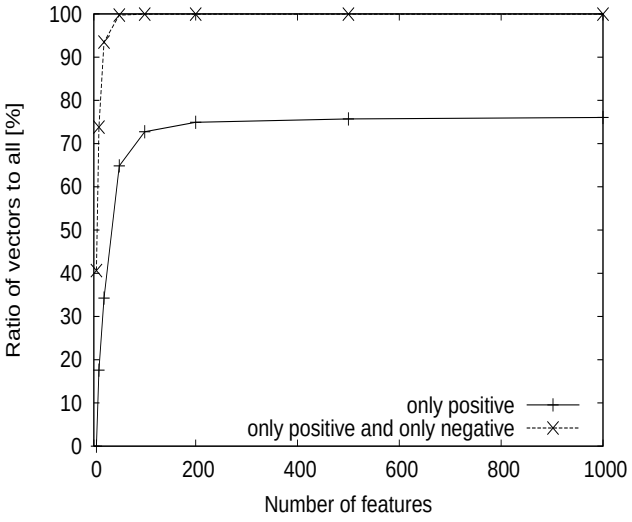
(b) Category "*acq*", experimental condition (2)



(b′) Category "*acq*", experimental condition (2)

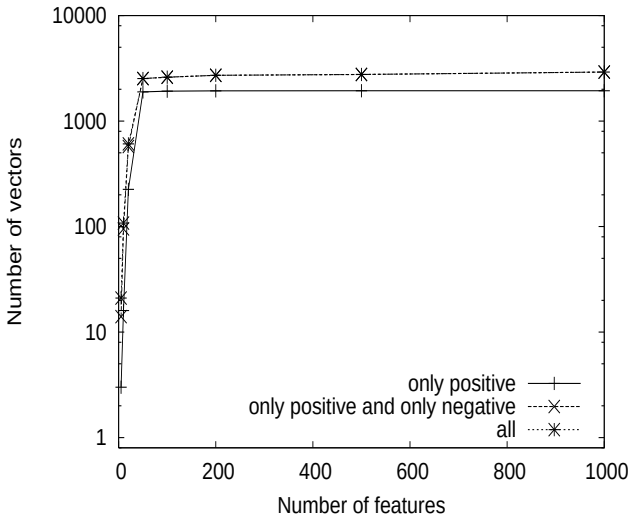


(c) Category "*acq*", experimental condition (3)

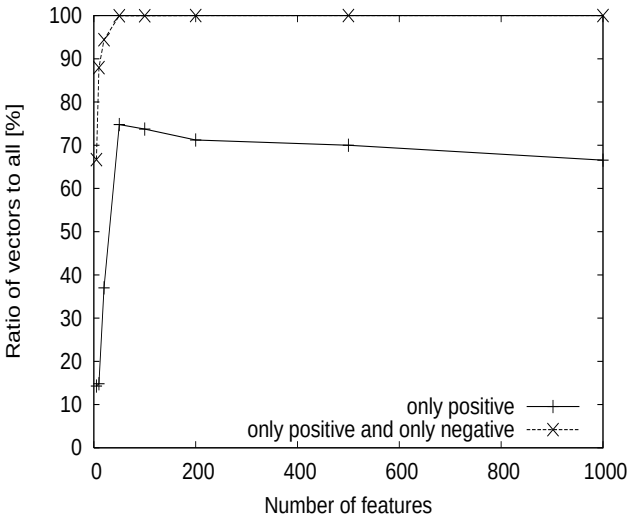


(c′) Category "*acq*", experimental condition (3)

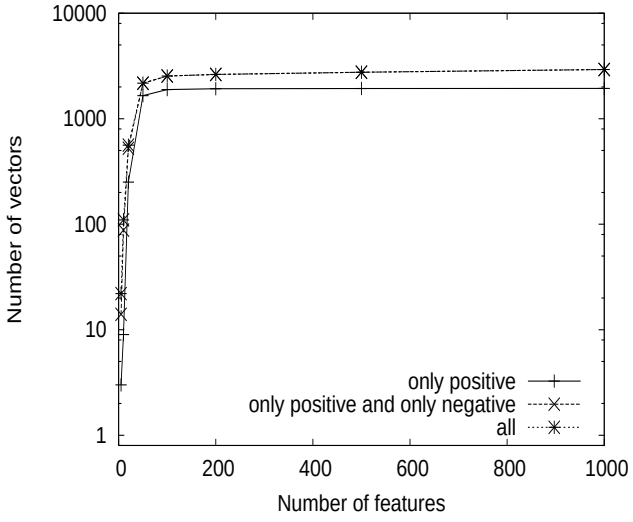
Fig. D.1 Distribution of document vectors in a feature space (category "*acq*")



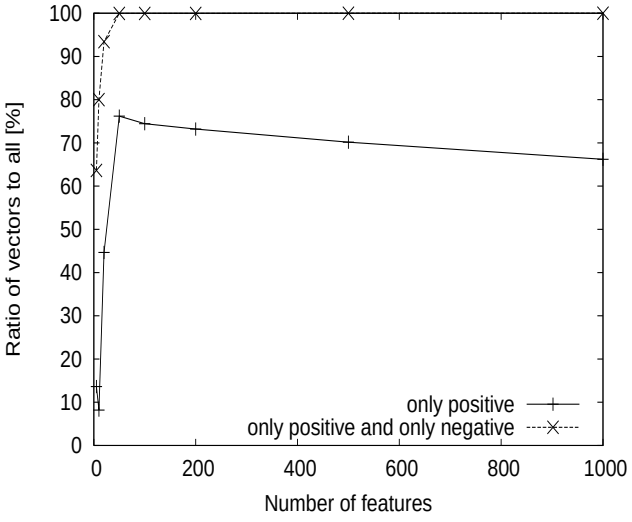
(a) Category "*earn*", experimental condition (1)



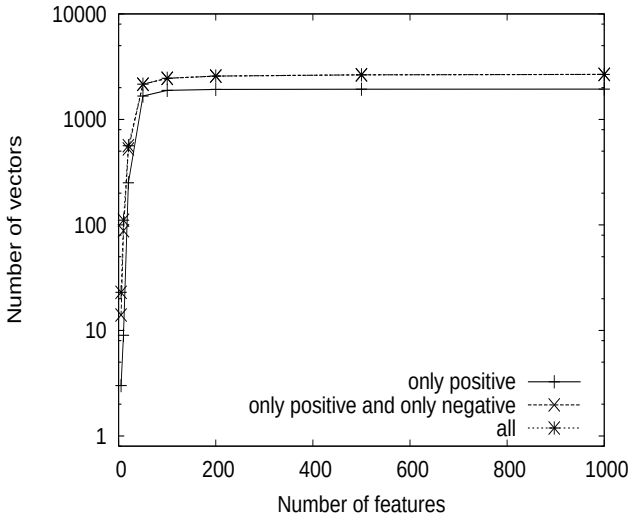
(a′) Category "*earn*", experimental condition (1)



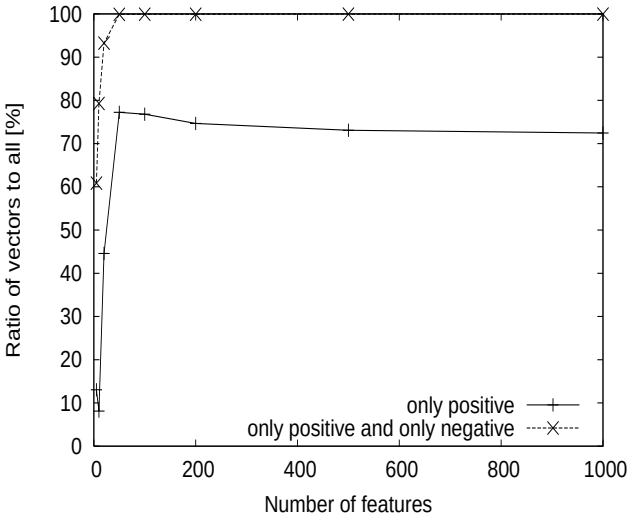
(b) Category "*earn*", experimental condition (2)



(b′) Category "*earn*", experimental condition (2)

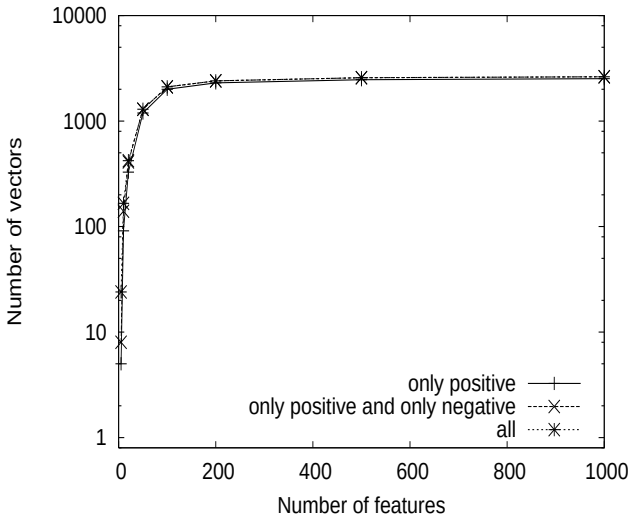


(c) Category "*earn*", experimental condition (3)

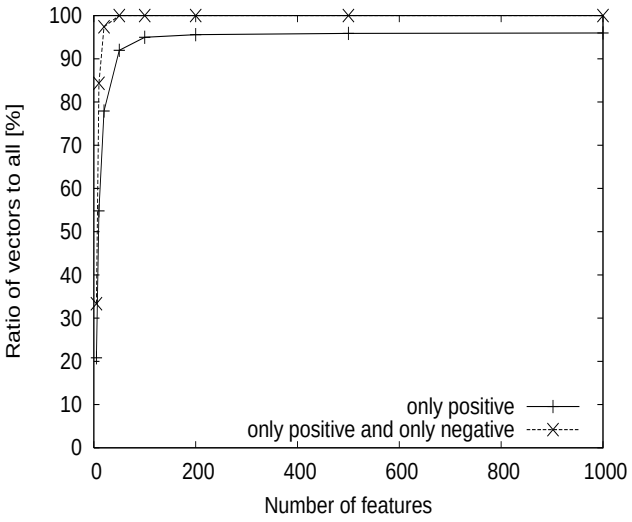


(c′) Category "*earn*", experimental condition (3)

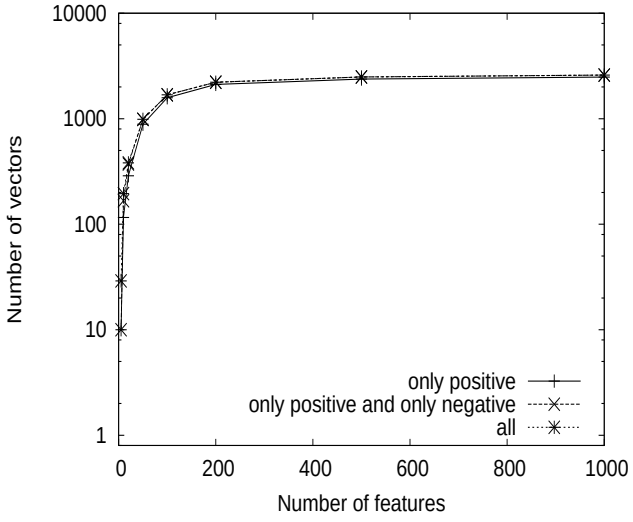
Fig. D.2 Distribution of document vectors in a feature space (category "*earn*")



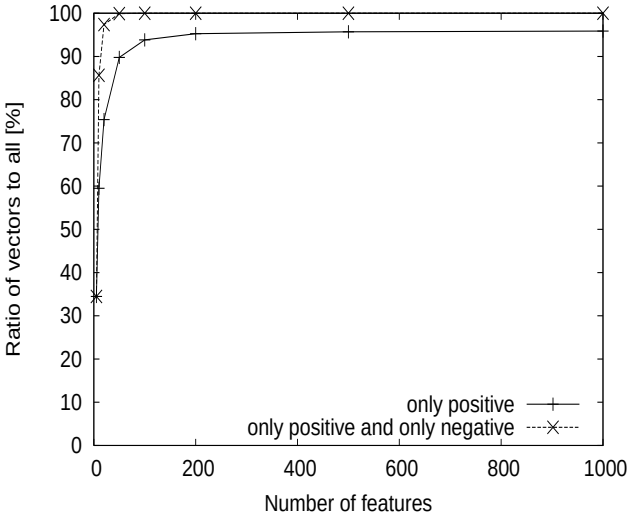
(a) Category "*trade*", experimental condition (1)



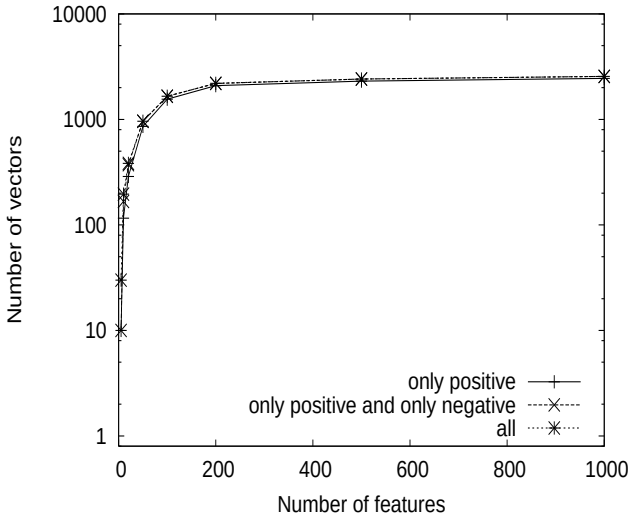
(a′) Category "*trade*", experimental condition (1)



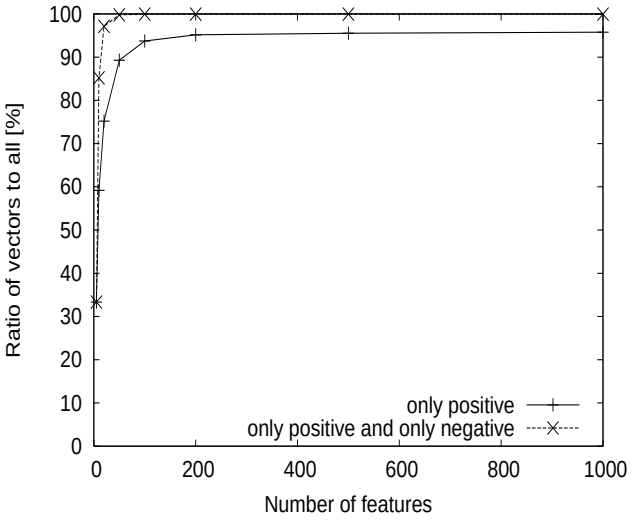
(b) Category "*trade*", experimental condition (2)



(b′) Category "*trade*", experimental condition (2)



(c) Category "*trade*", experimental condition (3)



(c′) Category "*trade*", experimental condition (3)

Fig. D.3 Distribution of document vectors in a feature space (category "*trade*")