DOI:10.3745/JIPS.2007.3.2.073

Optimization of Domain-Independent Classification Framework for Mood Classification

Sung-Pil Choi*,, Yuchul Jung*, and Sung-Hyon Myaeng***

Abstract: In this paper, we introduce a domain-independent classification framework based on both k-nearest neighbor and Naïve Bayesian classification algorithms. The architecture of our system is simple and modularized in that each sub-module of the system could be changed or improved efficiently. Moreover, it provides various feature selection mechanisms to be applied to optimize the general-purpose classifiers for a specific domain. As for the enhanced classification performance, our system provides conditional probability boosting (CPB) mechanism which could be used in various domains. In the mood classification domain, our optimized framework using the CPB algorithm showed 1% of improvement in precision and 2% in recall compared with the baseline.

Keywords: *Text Classification, Mood Categorization, Information Retrieval, Feature Selection, Text Classification Application*

1. Introduction

Detecting the mood or feelings of a specific document has become an interesting topic in the text mining domain. In particular, texts posted in private blog websites are considered as a favorable target of this task because they contain more elements of human feelings and emotions. By analyzing the emotional status of each text, we could detect or extract useful information about the authors from a set of blog websites.

In this paper, we introduce a novel approach for constructing a mood classification system. Rather than just making a domain-dependent classifier likewise other previous approaches, we initially implement a domain-independent text classification system, which is then optimized for the mood classification. One of the most beneficial aspects of the approach is the modularity and reusability of the entire framework. Given that our text classification framework provides various functionalities related to the optimization tasks, we are able to adapt it to any domain for a specific purpose, which is the mood classification of blog texts in this paper.

As the first step of achieving our final goal, we were involved in developing and combining two general-purpose text classification algorithms, Naïve Bayesian and k-nearest neighbors (kNN). Except for the feature manipulation process, almost all the text classification tasks are similar in nature. For the good quality of the result, therefore, it was critical for us to have high-qualified text classification engines. Also, we eagerly tried to make an efficient and practically accessible text classification framework with respect to speed, scalability, and usability. Based on the two reliable text classification models, our entire system manages to provide various options which can be used in tuning the classifier in a certain domain.

Manuscript received October 30, 2007; accepted December 26, 2007.
Corresponding Author: Sung-Pil Choi
* School of Engineering, Information and Communications University, 119, Munjiro, Yuseong-gu, Daejeon, 305-732, South Korea
** Information Technology Development Division, Korea Institute of Science and Technology Information, South Korea
{spchoi, enthusia77, myaeng}@icu.ac.kr

2. Related Work

In the domain of implementing and improving general-purpose text classification systems, there have been many approaches so far [1, 2, 3, 4]. Recently, machine learning based approaches have been popular in the document classification domain [1]. Especially, in [14], they proved that the most effective inductive approach should be Support Vector Machines (SVM) [1, 14]. Although so many ideas and algorithms to improve the existing limitations exist, there are very few researches related to implement the efficient and extendable text classifiers. Moreover, in relation to the optimization of the text classifier by using various feature selection methods and alternative classification approaches (kNN, Naïve Bayes), we could not find any preceding research work.

In the mood classification domain, to our knowledge, there is no similar approach with ours. In [7], they performed an in-depth empirical study on the mood classification using very small set of blog texts. Initially, some psychologists and cognitive scientists constructed a set of normative emotional ratings for a large number of English words [5]. Those words were classified by three different emotional dimensions: pleasure, arousal and dominance. Based on the list of words, some researches were performed for the mood classification of general documents [8, 9]. In this paper, we only used resources which are easy to gather from the web unlike other approaches that needed well-organized lexical resources

such as ANEW, WordNet and so forth [6, 7, 14].

3. Text Classification Framework

Figure 1 shows the conceptual architecture of our text classification framework implemented in this paper. Instead of developing a domain-dependent classifier which is restricted only to the mood classification, we implemented a general-purpose text classification framework so that it can be applied to any problem domain related to document classification. In this respect, the salient characteristics of the system is that it contains more than one document classification algorithms and also that the feature generation module is separated from those algorithms. This structure can be justified by the fact that almost all the classification algorithms take advantage of the similar feature information.

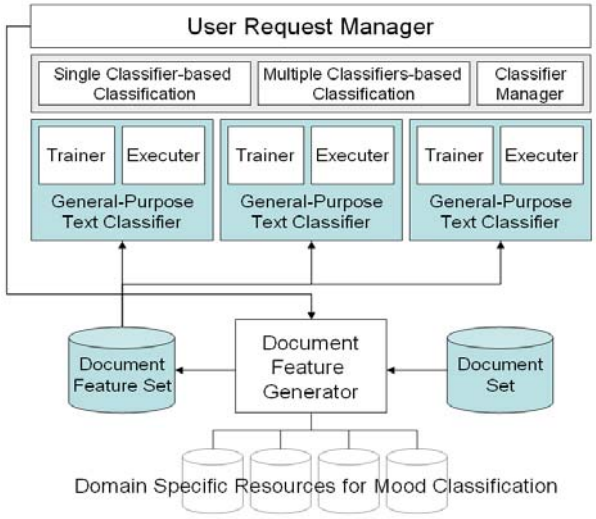

Fig. 1. Architecture of the Proposed Framework

For the fast classification process, we adapted the inverted file structure widely used in information retrieval systems. Unlike conventional inverted file structures, however, our architecture is adequately optimized to the text classification tasks, which includes memory-based binary tree structure for massive terms retrieval, efficient structure for saving and searching class information and so forth.

3.1 Feature Storage Structure

Once each document has been analyzed (e.g. tokenization, lemmatization, and POS tagging), a stream of terms in each document is generated. Fig. 2 shows our storage structure for saving term information (TISS: Term Information Storage Structure). It exploits the general architecture and process of inverted index structure used in IR. However, we expanded the concept so that TISS can handle the additional information related to classification tasks such as class identifiers, class frequencies, and class conditional probabilities, $P(w|c)$, where w is a term and c is a category.

As almost all the classification tasks assume an in-memory processing, which means that every analysis process is executed in main memory, we adapted binary tree structure for saving and searching each term of all the target data rather than B-tree or else. In the posting information part of each term, TISS maintains three kinds of information including total term frequency, general posting information which is the mapping information between a term and document and conditional probability information in terms of each target class.

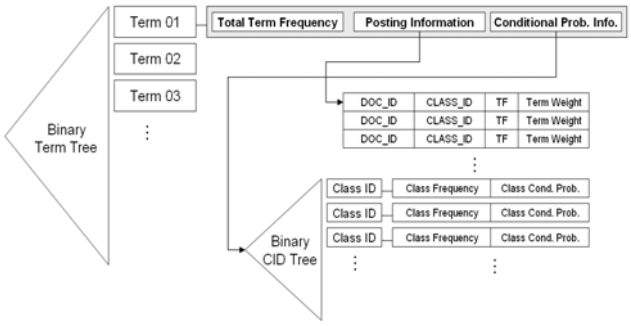

Fig. 2. Term Information Storage Structure

Total term frequency is the number of all occurrences of a term in the target collection. It will be used in calculating the probability of a term in both the learning process and feature selection process. Posting information contains not only every document identifier in which a term appears but additional information such as the class identifier of each document, term frequency and term weight. As you might see, this posting information relates to the similarity-based classification algorithms such as k-Nearest Neighbors and so forth. Finally, conditional probability information includes exactly C nodes if the number of target class is C. Each node has statistics about class and term distribution ($P(w|c)$).

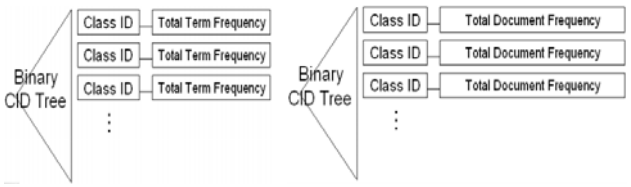

Fig. 3. Class Distribution Information

Fig. 3 denotes additional information for estimating the probabilities of all classes. While right side of the figure relates to saving the number of documents belonging to each class, left figure displays the storage structure for saving total term frequency of each class.

Given the refined feature information after the learning process, the next step is to serialize or write the information into file for the later use. The feature information consists of three separate files each of which plays an important role in the classification process. The first file is the set of the entire posting information of each

term which we mentioned before. Additionally, the second file is a set of the conditional probability of each term given a class, which can be used by Naïve Bayesian classifier. The last file contains the class probabilities. Users should specify the names of all the three files in the configuration file.

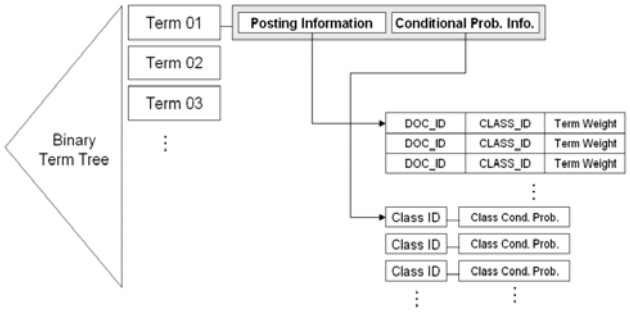

Fig. 4. Simplified Term Information Structure

On the other hand, in order to classify new documents based on the learned features, all the feature information should be loaded into the main memory. Once the information is loaded, classification engines can use it repeatedly, which may be the reasonable reason why our classification process should be configured as a server (daemon). As the loading process is quite time-consuming owing to the size of the feature information, this configuration will be beneficial in operating on-demand classification functionalities. While the term information storage structure in the learning process is quite complicated, in the classification (execution) process, our system uses very simplified structure to maintain the feature information. Fig. 4 shows the structure of term information in the classification phase. All the information is the same as in the *TISS*. Based on the structure, classification engines will be able to get every clue for classifying the target documents

3.2 Feature Selection Mechanism

Once the document loading process is completed, TISS contains all the information needed to classify new documents by using Naïve Bayesian or k-Nearest Neighbors. However, the information may include rather useless features spoiling the performance of the classification tasks. In order to filter these malicious features, our framework supports four kinds of feature estimation functions as described below.

3.2.1 Document Frequency.

Document frequency of a term is the number of documents containing the term. A term with very small document frequency might be less important in the text classification framework in that the term has less discriminative power among the class set due to the infrequent occurrences. If the *document frequency* of a particular term is less than *threshold*, our system will discard the term.

3.2.2 Information Gain

Information gain of a term in the classification framework is the measure of decreasing the uncertainty of the probability distribution of the set of classes given that we are aware of the term. The original equation for computing the information gain of a specific term (t) is:

$$\begin{aligned} IG(t) &= -\sum_{i=1}^{m} P(c_i)\log P(c_i) \\ &+ P(t)\sum_{i=1}^{m} P(c_i \mid t)\log P(c_i \mid t) \\ &+ P(\bar{t})\sum_{i=1}^{m} P(c_i \mid \bar{t})\log P(c_i \mid \bar{t}) \end{aligned} \quad (1)$$

where $IG(t)$ is the information gain value of a term t, $P(c_i)$ is the probability of a specific class c_i, and $P(c_i|t)$ denotes the conditional probability of c_i given a term t. In equation (1), it is not trivial to estimate $P(c_i|t)$ directly. By using the Bayes' rule, we are able to convert this equation into equation (2).

$$\begin{aligned} IG_{new}(t) &= -\sum_{i=1}^{m} P(c_i)\log P(c_i) \\ &+ P(t)\sum_{i=1}^{m} \frac{P(t \mid c_i)P(c)}{P(t)}\log\frac{P(t \mid c_i)P(c)}{P(t)} \\ &+ (1-P(t))\sum_{i=1}^{m} \frac{(1-P(t \mid c_i))P(c)}{P(t)}\log\frac{(1-P(t \mid c_i))P(c)}{P(t)} \end{aligned} \quad (2)$$

3.2.3 Point-wise Mutual Information

Point-wise mutual information is the degree to which both a term and a class occur in the document set. The original equation for computing point-wise mutual information is:

$$MI(t,c) = \log_2 \frac{P(t,c)}{P(t)\times P(c)} \quad (3)$$

With less information about the joint probability, $P(t,c)$, we should derive new equation from (3), which is possible by using Bayes' rule.

$$MI_{new}(t,c) = \log_2 \frac{P(t \mid c)\times P(c)}{P(t)\times P(c)} \quad (4)$$

Using (4), we can evaluate the average mutual information of a term in terms of all the classes by using

$$MI_{avg}(t) = \sum_{i=1}^{m} P(c_i) MI_{new}(t,c_i) \quad (5)$$

3.2.4 Chi-Square Estimation

Chi-Square estimation is a method for inspecting the lack of independence between two probabilistic variables. Given the concept of chi-square estimation, we can estimate the degree of dependency of a term and a class.

- ***A**: the number of documents belonging to a class c*
- ***B**: the number of documents belonging to a class c and not including a term t*
- ***C**: the number of documents not belonging to a class c and including a term t*

- ***D****: the number of documents not belonging to a class c and not including a term t*

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad (6)$$

$$\chi^2_{avg}(t) = \sum_{i=1}^{m} P(c_i)\chi^2(t,c_i) \quad (7)$$

where m is the number of classes to be learned and N is the number of documents in the training set.

As mentioned before, because we have much useful information after loading the entire documents, each estimation value is easily computed by means of converting the original equations into the identical ones using Bayes' rule. For instance, we need to know the conditional probability of each class given a term ($P(c_i|t)$) to calculate the information gain value of each term. But our structure does maintain only $P(t|c_i)$. By modifying the original information gain equation into another form, we can easily calculate the target value.

3.3 Two Classification Algorithms

Our system now supports two famous classification settings which are the most efficient and promising, Naïve Bayesian and k-Nearest Neighbor. If we define a function $f(\cdot)$ to be a text classifier which returns a relevant class given an input document, the Naïve Bayesian classifier which is the first algorithm is:

$$f(D) = \arg\max_{c_i} P(c_i)\prod_j P(x_j \mid c_i) = \arg\max_{c_i}(\log P(c_i) + \sum_j \log P(x_j \mid c_i)) \quad (8)$$

where a document D is composed of and also can be represented by a set of terms (x_j), and there are multiple target classes c_i. In order to estimate the conditional probability of a term given a class, $P(x_j|c_i)$, we used *multinomial estimation method*.

$$P(x_j \mid c_i) = \frac{\sum_{d_k \in c_i} tf_{jk} + 1}{\sum_{w_t \in V}\sum_{d_k \in c_i} tf_{tk} + |V|} \quad (9)$$

where V is a set of all the terms in the entire collection, tf_{ij} is the term frequency of a term t_i in a document d_j.

The second classification algorithm is k-nearest neighbor which is similar with the normal retrieval process based on vector space model. A newly appeared document will be classified by using the class information of the most similar documents which are already loaded in the document vector space. Based on the most similar k documents, we used the majority vote method which favors the most frequent class in the nearest neighbors.

$$f(D) = \arg\max_{c_i} Freq(kNN(D), c_i), \quad kNN(D) = \{d_i \in U \mid SimilarityRank(d_i, D) \le k\} \quad (10)$$

where D is a newly appeared document, c_i is a specific class and $SimilarityRank(\cdot,\cdot)$ returns the ranking information based on the similarity between the two documents. Our system provides two kinds of similarity measurement approaches. With the assumption of the vector space model, we can express a document d as a vector, (w_1, w_2, w_3, …, w_k), where w_i is the term weight value of a term t_i in the document and k is the number of terms in the entire collection.

$$Sim_{\cos}(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 \cdot \vec{d}_2}{|\vec{d}_1| \times |\vec{d}_2|} \quad (11)$$

$$Sim_{dst}(\vec{d}_1, \vec{d}_2) = \sqrt{\sum_{i=1}^{k}(w_i^1 - w_i^2)^2}$$

Sim_{cos} is the cosine value of the angle between d_1 and d_2 while Sim_{dst} is the Euclidian distance between two document vectors. Users may select one of the two similarity methods according to their target domains.

3.4 Classification Environment Settings

If you want to use each of the functions of our system, you should specify it in the configuration file. The configuration file contains various field names such as, TCL_FS_METHOD for denoting the method of feature estimation and one, TCL_FS_THRESH for specifying the threshold value with respect to the method. Also there are various options to be used in the classification process.

Table 1. Sample Classification Settings

```
# Directory of Lemmatization Dictionary...
LEMMA_DIC_DIR=/data2/liter/spchoi/KIE/lemmatizer
# TRUE : extract lemmas as terms
# FALSE: extract tokens as terms
LEMMA_AS_TERM=TRUE
# Classification Method Setting
#   - NB   : Naive Bayes Classification
#   - KNN : k-Nearest Neighbor Classification
TCL_METHOD=KNN
# The File Names of the RESULT of TCL_LEARNER
# TCL_CP_FILE : List of all Conditional Probabilities (P(w|c)) of the entire terms set.
# TCL_TW_FILE : List of all Posting Information (in IR) containing Term Weights.
# TCL_CI_FILE : List of all Classes and their Probabilities(P(c)).
TCL_CP_FILE=TCL.cp
TCL_TW_FILE=TCL.tw
TCL_CI_FILE=TCL.ci
# K value in kNN Classification
TCL_KNN_K=5
# TCL_FS_METHOD: Feature Selection Methods
#   - NN : Nothing
#   - DF : Document Frequency
#          ------> If DF(TERM) < TCL_FS_THRES then DISCARD TERM
#   - IG : Information Gain
#          ------> If IG(TERM) < TCL_FS_THRES then DISCARD TERM
#   - MI : Mutual Information
#          ------> If MI(TERM) < TCL_FS_THRES then DISCARD TERM
#   - CS : Chi-Square
#          ------> If CS(TERM) < TCL_FS_THRES then DISCARD TERM
TCL_FS_METHOD=DF
TCL_FS_THRESH=5
```

Table 1 shows the configuration example of our system. Detailed explanations are supported before or after each section for the convenience. By means of the dynamic configuration function, users can find a way to optimize their own text classifiers to certain domains. During the

optimization process, they can generate this configuration file with different settings while training and classifying candidate classifiers to find a set of classification environments showing the best results.

4. Conditional Probability Boosting

Although Naïve Bayesian Classifier shows competitiveness in both its effectiveness and efficiency in general, the main drawback of Naïve Bayesian Classifier is that its accuracy is susceptible to the sparseness of the training data. For our target domain, mood classification, what is worse, we have been given very small set of training data (only about 2,000 documents). For the purpose of the enhanced effectiveness, additional approaches would be necessary.

First of all, we define two kinds of external lexical resources to be used to improve the performance of Naïve Bayesian Classification:

- **General Lexical Resource**: Sufficient set of terminologies in a certain domain
- **Classified Lexical Resource**: Very small set of terminologies classified with respect to the detailed categories in the domain

While general lexical resource is a set of almost all the terminologies in a certain domain without any information about classes of individual terminologies, classified lexical resource denotes a very small set of clarified terminologies which have been considered as representative or topical and classified with respect to the sub-categories of the domain. The main reason why two kinds of lexical resources exist is that it is not trivial to construct the complete lexical resource with detailed information in a certain domain. Instead, many people tend to construct tiny lexical sets which consist of pivotal lexicons having their detailed information, such as part-of-speeches, semantic categories and so forth.

Table 2. General/Classified Lexical Resource for the domain of mood classification

Name	Category	Descriptions
Feeling Words	General Lexical Resource	http://eqi.org/fw.htm Incomplete set of all the words expressing human feelings (2,514 words)
List of Feeling Words	Classified Lexical Resource	http://www.psychpage.com/learning/library/assess/feelings.html Classified list of feeling words by 16 classes
Feeling Words	Classified Lexical Resource	http://www.wwme.org/feelings.html Classified list of feeling words by 9 classes
Feeling Words	Classified Lexical Resource	http://www.advocatesforyouth.org/youth/health/relationships/feelingwords.htm Classified list of feeling words by 12 classes

In order to improve the accuracy of our Naïve Bayesian Classifier in the domain of mood classification, we gathered those two kinds of lexical resources from the web, which is shown in Table 2. A complete classified lexical resource used in our system is made by combining the three classified resources mentioned in the table and removing redundancies. Although there are much more sense classes in each resource, only 4 classes (angry, fear, happy, and sad) are chosen in this paper. After preprocessing the combined set, we could have 227 words of classified lexical resource. For the general lexical resource, we use the entire set of 2,514 human feeling words provided by the corresponding web site.

Now we should determine how these two resources are applied to Naïve Bayesian Classification. There may be several approaches in doing this, which may include raising the frequencies of the terms that are in the lexical resource. However, this approach is problematic in that we cannot make a decision about the degree to which the frequency of a term increases. We devise another approach called Conditional Probability Boosting (CPB) method, which increases or decreases the conditional probability of a term given a class, $P(t|c)$. In the Naïve Bayesian settings, given a term extracted from the set of documents to be dealt with, we should know the conditional probabilities, $P(t|c)$ to calculate $P(d|c)$, a probability that a document d belongs to the class c. After learning the entire document set, therefore, we can come up with the set of conditional probabilities, $P(t|c_1)$, $P(t|c_2)$, … , $P(t|c_k)$ of a specific term, with k classes.

We can assume that if a term exists in either the general lexical resource or classified lexical resource, it is likely that the conditional probabilities of the term should be boosted or increased. All the conditional probabilities have been calculated only by considering the statistic information of the current document set (learning set). Therefore, it is obvious that those probabilities suffer from overfitting or lack of statistics. By using the lexical knowledge of a certain domain, we can enhance the quality of those conditional probabilities.

$$P_{unbounded}(t \mid c_i) = P_{old}(t \mid c_i) \times \frac{BR}{(Rank(P_{old}(t \mid c_i)))^p} \quad (12)$$

Boosting process of a conditional probability can be performed by the above equation. In this formula, BR denotes a boosting ratio which specifies the degree to which the current conditional probability increases. The rank of a conditional probability represents the rank of the probability value in the current set of conditional probabilities of a specific term. If the number of the target classes is k, then there should be k conditional probabilities of a specific term. In this list, we can rank the probabilities in terms of their values. The rank of the maximum value is 1 and of the minimum is k. However, a problem is that the resulting probabilities can be larger than 1.0, which is not appropriate for the characteristics of the probability. With the resulting values larger than 1.0, the following equation can be used to adjust them.

$$P_{boosted}(t \mid c_i) = (1 + C_1 \times \exp(-C_2 \times P_{unbounded}(t \mid c_i)))^{-1}$$
$$= (1 + C_1 \times \exp(-C_2 \times P_{old}(t \mid c_i) \times \frac{C_1}{(Rank(P_{old}(t \mid c_i)))^p}))^{-1} \quad (13)$$

This formula is a logistic function that has two parameters. One of the benefits of the logistic function is that it can be limited by a specified value which is 1.0 in our system. Based on the boosting criterion, conditional probability boosting procedure is performed as follows.

1. *Check whether the current term is in the general resource. If it exists, all the conditional probabilities of the term are boosted based on the order of the size.*
2. *Check whether the current term is in the classified resource. If it exists, only the probability of the class of the current term is boosted.*

The first step has an intension to grow the gap between the two probabilities. Actually, only the first rank (the maximum) becomes larger, while others become smaller. In the second step, as we know the class information of the current term, the boosting is performed only in the correspond probability of the class.

5. Experimental Results

For inspecting the performance of our system, we make use of a set of documents classified by four different classes (angry, fear, happy, sad). This test collection has been constructed by collecting huge size of postings (texts) from "LiveJournal.com," which is a famous web blog service site. In order to improve the quality of the collection, manual filtering and verification processes have been performed. Table 3 shows the details of the data.

Table 3. Target Data

classes	angry	fear	happy	sad
# docs	534	537	576	503
Total	2,150			

We used 1,754 documents for training our two classifiers, kNN and Naïve Bayesian while remaining ones were used for the evaluation of the performance of each classifier. Our method of performance measurement is the macro-averaging in which computes performance for each class and then averages those results.

Firstly, to find the best feature selection setting that shows optimal performance, we performed large-scaled experiments changing both the feature selection method and its threshold value. Fig. 5 shows the performance of our Naïve Bayesian classifier with respect to the various settings of feature selection mechanisms. We already mentioned that our system provides four fundamental feature selection methods. As you can see in this figure, the precision and recall are maximized when point-wise mutual information is used as the feature selection method and its best threshold is 0.6.

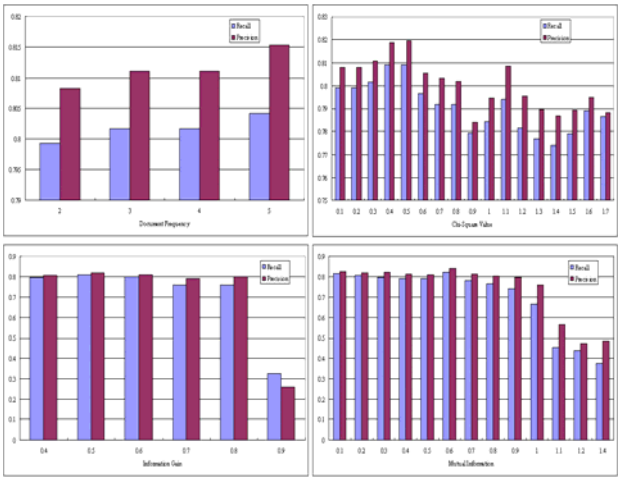

Fig. 5. Performance of Naïve Bayesian w.r.t. Feature Selection

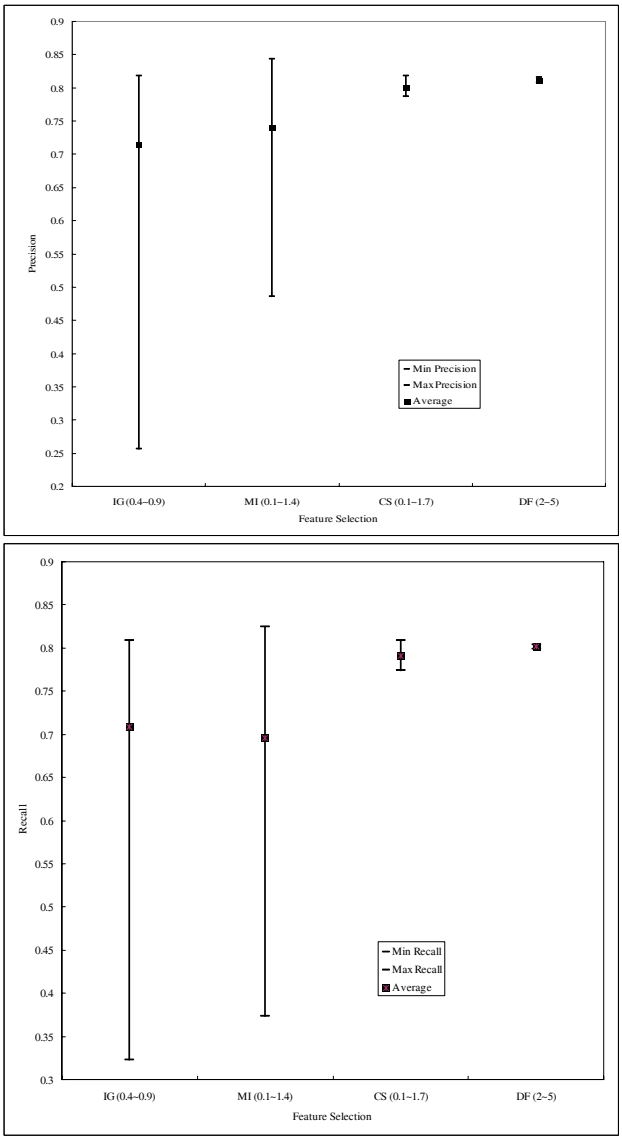

Fig. 6. Precision & Recall Distribution w.r.t. Feature Selection Methods

Fig. 6 shows the distribution and average value of precision and recall when different feature selection methods are applied by using various thresholds. On average, the best feature selection method is to use document frequency as the criteria of cutting off terms. Point-wise mutual information and information gain have very wide ranges of performance, which means that their reliability might not be good in general domains. However, our ultimate goal is to optimize our classifier into the specific domain and we should pay attention to the maximum performances among all the methods. In this respect, we choose mutual information as our main feature selection method.

Although we does not give the result of kNN classifier, in our experiments, its performance is very poor compared to Naïve Bayesian. Therefore, we consider Naïve Bayesian classification as our base-line system and compare it with our enhanced mechanism. The table shown below provides the confusion matrix of the optimal feature selection setting. Its macro-average recall and precision is **0.824419** and **0.843468** respectively.

Table 4. Confusion Matrix of Optimal Setting

Result / Answer	angry	fear	happy	sad
angry	98	1	1	0
fear	9	76	9	5
happy	4	0	96	0
sad	19	7	15	59
P/R	**0.75/0.98**	**0.90/0.76**	**0.79/0.96**	**0.92/0.59**

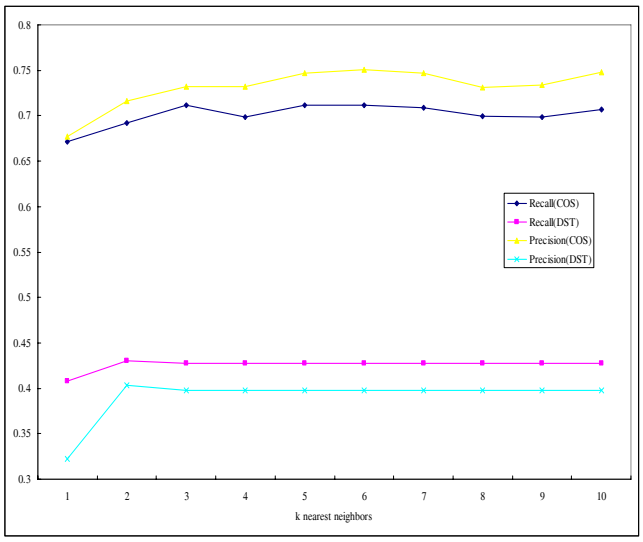

Fig. 7. Performance of kNN classifier

Before performing the comparative experiments, we did some fundamental experiments about the two classifiers. Fig. 7 shows the performance of kNN classifier by changing the value k (the number of nearest neighbors considered).

When it comes to k-nearest neighbor algorithm, we can think of the methods of computing the similarity between two documents. In our system, as mentioned before, we provide two ways to computing the similarity, which include both Euclidian distance-based and cosine-based methods. It is surprising that the performance of using the distance-based method is very poor compared to the cosine-based method. We could not find out the exact reason but it seems to be related to the formation of the distribution of classified document vectors. This could be another research topic in which we should pursue.

Fig. 8 compares the performance of both best-performed kNN and Naïve Bayesian using point-wise mutual information as the feature selection criterion. As we said previously, Naïve Bayesian classifier is superior to kNN in terms of both precision and recall.

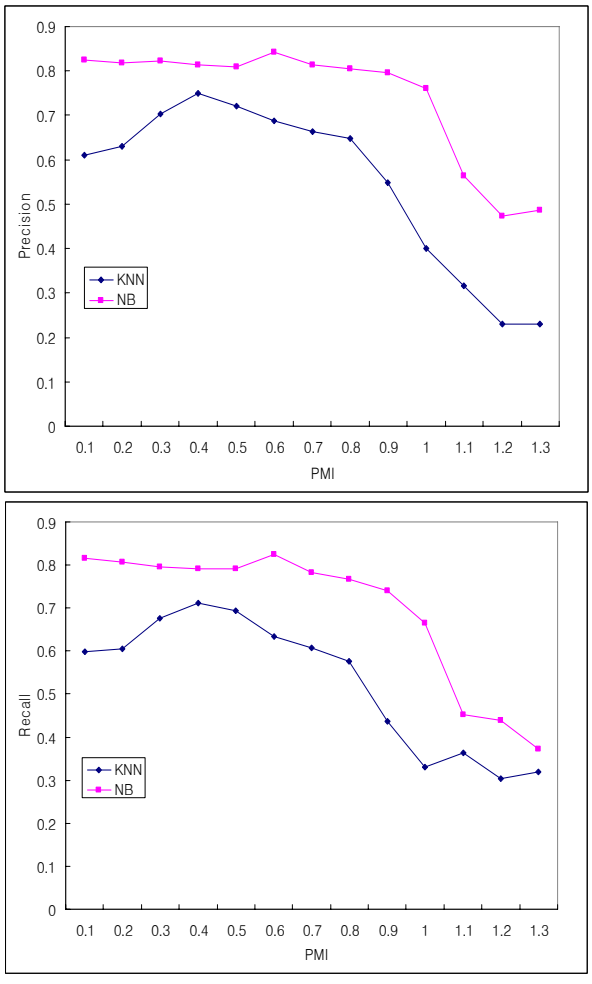

Fig. 8. Performance Comparison between kNN and NB

Finally, we compare our baseline system with the enhanced system using conditional probability boosting method. To fully inspect the effect of the method, the experiment was performed in the four different systems, NA, CPB_GR, CPB_CR, BOTH. The detailed explanation about them is given in the following table. The experiment was performed by changing the threshold of point-wise mutual information.

Table 5. Four Classification Systems either using or not using CPB

System	Details
NA	**No CPB** is applied.
CPB_GR	**Only General Lexical Resource** is applied.
CPB_CR	**Only Classified Lexical Resource** is applied.
BOTH	**Both General and Classified Lexical Resource** is applied.

Fig. 9 shows the result of comparing the four systems. Although NA is showing competitive performance in the small PMI values, at the optimal points (0.5 ~ 0.7) in which represent the best-performance, CPB_CR and BOTH methods shows better precision and recall. Particularly, at the small PMI points, CPB_CR is best performing, which can be explained by the strong effect of domain lexicons boosted by CPB algorithm.

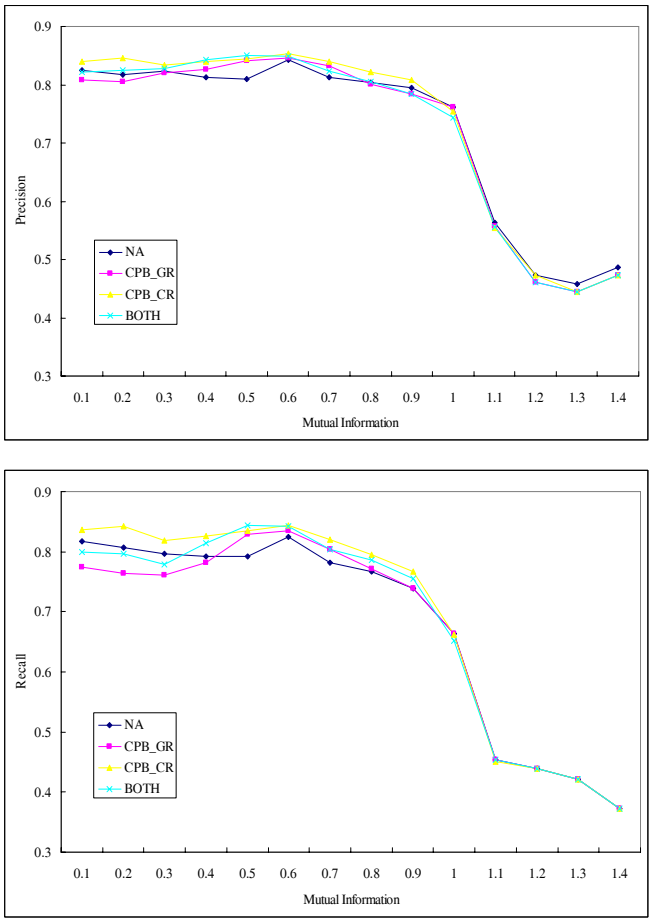

Fig. 9. Performance Comparison between using and not using CPB

The Table 6 shows the comparison between the baseline system and the optimally enhanced system (CPB_CR). The macro-average recall of the enhanced system is **0.844545** which shows the enhancement of about 2%. Also, the macro-average precision of it is **0.852916**. In the table, we can see that it is relatively difficult to correctly classify the documents of fear and sad class. By using CPB algorithm mentioned above, it was possible to improve both the precision and recall of the system particularly in those two classes.

Note the improvements of both precision in the "fear" and "sad" classes in this result. Intuitively, it seems to be more difficult to discriminate the documents of these two classes than others in that their lexical usages are very similar. Although we did not investigate in detail how the two external resources can improve the overall performance of the mood classification, it is likely that accurately classified lexical resource should have played an important role in classifying somewhat ambiguous documents in the "fear" and "sad" classes. Therefore, we could conclude that by suitably applying independently built general and classified resources, we could improve the overall performance of the text classification system in any domain.

Table 6. Confusion Matrix of both baseline and CPB-based system

Baseline System				
	angry	fear	happy	sad
angry	**98**	1	1	0
fear	9	**76**	9	5
happy	4	0	**96**	0
sad	19	7	15	**59**
P/R	**0.75/0.98**	**0.90/0.76**	**0.79/0.96**	**0.92/0.59**
Optimally Enhanced System (CPB_CR)				
Angry	**96**	1	1	2
Fear	7	**81**	5	6
Happy	5	1	**94**	0
Sad	15	7	12	**65**
P/R	**0.78/0.96**	**0.90/0.81**	**0.83/0.94**	**0.89/0.66**

6. Concluding Remarks

In this paper, we developed a multi-purpose classification framework based on k-nearest neighbor and Naïve Bayesian classification algorithms. Although the experiments about the efficiency of our system are not given in this paper, we could notice the fact that the system uses very small memory and its speed is very competitive. Furthermore, the architecture of our system is simple and modularized in that each module could be changed or improved efficiently. Also, it provides various feature selection mechanisms to be applied to optimize the general-purpose classifiers for a specific domain.

As for the enhanced classification performance, our system provides conditional probability boosting (CPB) mechanism which could be used in various domains. In the mood classification domain, the CPB algorithm showed 1% of improvement in precision and 2% in recall.

It is necessary to make the system robust to deal with massive document set in various domains. In addition, the general performance of CPB algorithm should be verified by using other domain-dependent knowledge (general and classified lexical resource). Finally, we should provide GUI (graphic use interface) tools for users to be able to

exploit many benefits of our system easily and effectively while optimizing the system.

Reference

[1] Fabrizio Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, 34(1):1-47, 2002

[2] Andrew McCallum and Kamal Nigam, "A Comparison of Event Models for Naïve Bayes Text Classification," *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pp.41-48

[3] Yiming Yang and J. Pedersen, "A Comparative Study on Feature Selection in Text Categorization," *Proc. of the 14th International Conference on Machine Learning, ICML '97*, pp.412-420

[4] David Lewis, "Evaluating and Optimizing Autonomous Text Classification Systems," *Proc. of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995

[5] Margaret M. Bradley and Peter J. Lang, "Affective Norms for English Words (ANEW): Instruction Manual and Affective Ratings," *Technical Report C-1*, The Center for Research in Psychophysiology, University of Florida.

[6] Hugo Liu, Henry Lieberman, Ted Selker, "A model of textual sensing using real-world knowledge," *Proceedings of the 8th international conference on Intelligent user interfaces*, PP.125-132, 2003

[7] G. Mishne, "Experiments with Mood Classification in Blog Posts," *Style 2005 at SIGIR 2005*, 2005

[8] Owsley, S., Sood, S., and Hammond, K. Domain Specific Affective Classification of Documents. *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs*, 2006.

[9] Ruberry, M., Owsley, S., and Shamma D.A. Affective Behaviors for Theatrical Agents. *Proc .of Intelligent User Interfaces Workshop on Affective Interactions: the Computer in the Affective Loop*, 2005.

[10] S. T. Dumais, J. Platt, D. Heckerman and M. Sahami, "Inductive learning algorithms and representations for text categorization," *Proc. of CIKM '98*, pp.148-155

[11] Y. B. Lee and S. H. Myaeng, "Text Genre Classification with Genre-Revealing and Subject-Revealing Features," *Proc. of the 25th ACM SIGIR Conference, Tempere, Finland*, 2002

[12] Isabelle Guyon, Andr´e Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, 3, pp. 1157-1182, 2003

[13] Rosso, Paolo and Edgardo Ferretti and Daniel Jimenez and Vicente Vidal, "Text Categorization and Information Retrieval Using WordNet Senses," *Proceedings of the Second Global WordNet Conference , pp. 299-304, Brno, Czech Republic*, January 20-23, 2004

[14] Thorsten Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 1998

Sung-Pil Choi is a senior researcher at Korea Institute of Science and Technology Information (KISTI). And also he is a Ph.D. student in the School of Engineering at Information and Communications University (ICU), Korea. He received his MS from the Pusan National University (PNU). His current research interests include Text Mining, Machine Learning, Natural Language Processing, Information Retrieval and other related fields.

Yuchul Jung is a Ph.D. candidate in the School of Engineering at Information and Communications University (ICU), Korea. He received his MS in Digital Media from the same university. His current research interests include Contextual Advertising, Commonsense Knowledge Extraction, Cognitive Robotics, Commonsense Reasoning, Information Retrieval, Mood Classification, Semantic Web Search, and various types of Data Mining.

Sung Hyon Myaeng is currently a professor at Information and Communications University (ICU), Korea. Prior to this appointment, he was a faculty at Chungnam National University, Korea, and Syracuse University, USA, where he was granted tenure. He has served on program committees of many international conferences in the areas of information retrieval, natural language processing, and digital libraries, including his role as a chair for ACM SIGIR, 2002 and 2008, and for AIRS, 2004. He is on editorial boards of several international journals, including ACM Transactions on Asian Information Processing as an associate editor, Information Processing and Management, Journal of Natural Language Processing, and Journal of Computer Processing of Oriental Languages. Domestically, he was the chair of SIG-HLT (Human Language Technology), Korea Information Science Society, between 2005 and 2007. He has published numerous technical articles on conceptual graph-based IR, cross-language IR, automatic summarization, text categorization, topic detection and tracking, and distributed IR in the context of digital libraries.