Information Processing and Management 53 (2017) 1255–1268

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

INFORMATION PROCESSING & MANAGEMENT

An International Journal

How to use negative class information for Naive Bayes classification

Youngjoong Ko

Computer Engineering, Dong-A University, Busan, 604-714 Republic of Korea

ARTICLE INFO

Article history:
Received 30 January 2015
Revised 1 January 2017
Accepted 17 July 2017
Available online 27 July 2017

Keywords:
Naive Bayes classifier
Negative class information
Odds of class probabilities
Text classification

ABSTRACT

The Naive Bayes (NB) classifier is a popular classifier for text classification problems due to its simple, flexible framework and its reasonable performance. In this paper, we present how to effectively utilize negative class information to improve NB classification. As opposed to information retrieval, supervised learning based text classification already obtains class information, a negative class as well as a positive class, from a labeled training dataset. Since the negative class can also provide significant information to improve the NB classifier, the negative class information is applied to the NB classifier through two phases of indexing and class prediction tasks. As a result, the new classifier using the negative class information consistently performs better than the traditional multinomial NB classifier.

1. Introduction

Text Classification (TC) is the task of assigning predefined classes to free texts (Sebastiani, 2002). All of these tasks require text classifiers to determine which class is more relevant to a given document. A number of learning methods have been applied to these tasks such as Naive Bayes (NB) (McCallum and Nigam, 1998), nearest neighbor (Yang & Chute, 1994), support vector machine (SVM) (Joachims, 1998), etc. The NB classifier has been used as one of the popular classifiers for text classification because of its simplicity and reasonable performance. Traditionally, the NB classifier is a family of simple probabilistic classifiers based on applying Bayes' theorem with a strong independent assumption between the features. This is the so-called “Naive Bayes assumption.” Because of the independence assumption, the parameters for each feature can be separately estimated and this fact can simplify feature estimation or learning, especially with a large number of features. Text classification is one such example, where the features for the texts are individual words from a large vocabulary. That is a reason why the NB classifier often performs classification well, despite being based on a clearly false assumption in most real-world applications.

With this background, the NB classifiers have been extensively studied by many researchers (Lewis, 1998; McCallum & Nigam, 1998a; McCallum & Nigam, 1998b). There are two different models for the NB classifier: the multivariate Bernoulli and the multinomial NB models. Since the multivariate Bernoulli model is not equipped to use term frequencies in texts, the multinomial model is usually regarded as the standard NB text classification model (Dumais, Plat, Heckerman, & Sahami, 1998; Han, Ko, & Seo, 2007; Ko & Seo, 2009; Yang & Liu, 1999). Thus we also make use of the multinomial model for the NB classifier and focus on how to utilize negative class information to improve the multinomial NB classifier.

E-mail addresses: youngjoong.ko@gmail.com, yjko@dau.ac.kr

http://dx.doi.org/10.1016/j.ipm.2017.07.005
0306-4573/© 2017 Elsevier Ltd. All rights reserved.

As mentioned earlier, although the NB classifier is very efficient and easily implemented when compared to other text classifiers, its performance is not as good as the state-of-the-art text classifiers such as the SVM (Joachims, 1998) classifier. Therefore, it is worthwhile to improve the performance of the NB classifier for various practical applications such as spam filtering or news article/blog classification. In the multinomial model, the feature probabilities for a class are estimated by calculating the likelihood in only positive training documents. However, the feature probabilities in negative training documents could also be informative for text classification. According to the probabilistic ranking principle of information retrieval, the feature probabilities in the negative class should be used for determining the relevance or non-relevance of ranked documents (Manning, Raghavan, & Schütze, 2008).

This paper proposes a novel approach to effectively utilize negative class information in NB text classification. The negative class information is applied to two parts of text classification by using the log-odds ratio. The first part is the indexing process and the second one is to calculate probabilities of a class given a document for class prediction. We performed experiments comparing two kinds of modified NB classifiers that take advantage of negative class information and showed that they have better performance than the traditional multinomial NB classifier.

The remainder of this paper is organized as follows. Section 2 introduces traditional multinomial NB text classification. Section 3 describes the proposed approach using negative class information in detail. Section 4 presents the experimental design and results. Finally, Section 5 provides some concluding remarks regarding this research.

2. Multinomial Naive Bayes text classification

The multinomial NB model has been widely used as a probabilistic learning model for text classification. This is a supervised learning model and the probability of a document d being in class c is calculated via Bayes' theorem and an independence assumption as follows:

$$\mathrm{P(c|d)} = \frac{P(\mathrm{c})P(\mathrm{d|c})}{P(\mathrm{d})} \propto P(\mathrm{c})P(\mathrm{d|c}) \propto P(\mathrm{c})\prod_{i=1}^{n_d} P(w_i|\mathrm{c})^{tf_{id}}, \tag{1}$$

where $P(w_i|c)$ is the conditional probability of word w_i occurring in document d of class c and $P(c)$ is the prior probability of a document occurring in class c. $\langle w_1, w_2, \ldots, w_{n_d}\rangle$ are the unique words in d and they are a part of the vocabulary that we use for classification. tf_{id} represents the number of word w_i occurrences in document d and n_d is the number of unique words in d.

Contrary to the multivariate Bernoulli model, the multinomial model treats a document as an ordered sequence of word occurrences with each word occurrence as an independent trial. That is, a document is drawn from a multinomial distribution of words. Thus, in the multinomial model, $P(d|c)$ is calculated as follows:

$$\mathrm{P}(d|c) = \mathrm{P}(|d|)|d|!\prod_{i=1}^{n_d}\frac{P(w_i|c)^{tf_{id}}}{tf_{id}!} \propto \prod_{i=1}^{n_d} P(w_i|c)^{tf_{id}} \tag{2}$$

In Eq. (2), the parameter $P(w_i|c)$ is often estimated with Laplace or add-one smoothing (Manning et al., 2008):

$$\hat{\mathrm{P}}(w_i|\mathrm{c}) = \frac{1 + \sum_{k=1}^{|D_c|} tf_{id_k}}{|V| + \sum_{j=1}^{|V|}\sum_{k=1}^{|D_c|} tf_{jd_k}}, \tag{3}$$

where $|V|$ is the number of unique words in labeled training documents, $|D_c|$ is the number of labeled training documents in the given class label c and tf_{id_k}represents the number of word w_i occurrences in document d_k in D_c. This smoothing prevents probabilities of zero for infrequently occurring words.

In text classification, the goal is to find the best class for a document. The best class in NB classification is the most likely or maximum a posteriori (MAP) class *predicted_class* (Manning et al., 2008):

$$\text{predicted\_class}_d = \underset{c\in C}{\operatorname{argmax}}\,\hat{P}(c|d) = \underset{c\in C}{\operatorname{argmax}}\,\hat{P}(c)\prod_{i=1}^{n_d}\hat{P}(w_i|c)^{tf_{id}}, \tag{4}$$

where C is the set of classes in the training set and $\widehat{P}$ is used instead of P because they are not true values of parameters P(c) and P(w_i|c), but they are estimated from the training set as we will see in a moment.

In Eq. (4), because many conditional probabilities are multiplied, this can result in a floating point underflow. Therefore, it is better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. The final maximization by logarithms is as follows:

$$\text{predicted\_class}_d = \underset{c\in C}{\operatorname{argmax}}[\log\hat{P}(c) + \sum_{i=1}^{n_d} tf_{id}\log\hat{P}(w_i|c)] \tag{5}$$

It is the traditional NB text classification, which is denoted by TRA_NB. Each conditional parameter $\log\hat{P}(w_i|c)$ is a weight that indicates how good an indicator w_k is for c and the prior $\log\hat{P}(c)$ is a weight that indicates the relative frequency of c.

For the prior probabilities, more frequent classes are more likely to be the correct class than infrequent classes. The sum of log prior and word weights is then a measure of how much evidence there is for the document being in the class.

This model discards information about the order of the words, but takes the term frequency information of each word in a document (McCallum & Nigam, 1998b).

3. Application of negative class information in two phases

In this section, we first introduce the probability ranking principle that is the starting point for the proposed approach and explain how to utilize the negative class information in the indexing and class prediction phases of text classification in detail.

3.1. Probability Ranking Principle (PRP) in information retrieval

In information retrieval, the obvious order for presenting documents to a user is to rank them by their estimated probability of relevance with respect to the information need using a probabilistic model: $P(R=1|d,q)$. This is the basis of the Probability Ranking Principle (PRP) (Gordon & Lenk, 1991, 1992; van Rijsbergen, 1979), in which all documents are simply ranked in decreasing order of $P(R=1|d,q)$. Rather than estimating this probability directly, we work with some other quantities that are easier to compute and give the same document ordering because we are interested in only how to rank documents. In particular, we can rank documents based on their odds of relevance because the odds of relevance are monotonic with the probability of relevance (Manning et al., 2008). This makes things easier because we can ignore a common denominator as follows:

$$\mathrm{O}(\mathrm{R}|\mathrm{d},\mathrm{q}) = \frac{P(\mathrm{R}=1|\mathrm{d},\mathrm{q})}{P(\mathrm{R}=0|\mathrm{d},\mathrm{q})} = \frac{\frac{P(R=1|\mathrm{q})P(\mathrm{d}|R=1,\mathrm{q})}{P(\mathrm{d}\ \ |\mathrm{q})}}{\frac{P(R=0|\mathrm{q})P(\mathrm{d}|R=0,\mathrm{q})}{P(\mathrm{d}\ \ |\mathrm{q})}} = \frac{P(R=1|\mathrm{q})}{P(R=0|\mathrm{q})} \cdot \frac{P(\mathrm{d}|R=1,\mathrm{q})}{P(\mathrm{d}|R=0,\mathrm{q})} \tag{6}$$

The left-hand term, $\frac{P(R=1|\mathrm{q})}{P(R=0|\mathrm{q})}$, which is the rightmost expression of Eq. (6), is a constant for a given query, and does not need to be estimated when ranking documents. After the NB conditional independence assumption is applied to the right-hand term, $\frac{P(\mathrm{d}|R=1,\mathrm{q})}{P(\mathrm{d}|R=0,\mathrm{q})}$, we can obtain the following Eq. (7):

$$\frac{P(\mathrm{d}|R=1,\mathrm{q})}{P(\mathrm{d}|R=0,\mathrm{q})} = \prod_{i}^{n_d} \frac{P(w_i|R=1,\mathrm{q})}{P(w_i|R=0,\mathrm{q})}, \tag{7}$$

where $d=[w_1,\ldots,w_{n_d}]$ and n_d is the number of words in document d.

We can now define the Relevance Status Value (RSV) of document d and class c as *RSV(d,c)* for text classification using PRP and Eq. (7). In *RSV(d,c)* for text classification, a query q is replaced by a class label c. That is, we want to estimate how relevant a document is to a given class to determine whether the document should be assigned to that class as follows:

$$RSV(d,c) = log\left(\frac{P(\mathrm{d}|R=1,c)}{P(\mathrm{d}|R=0,c)}\right) = log\prod_{i}^{n_d} \frac{P(w_i|R=1,c)}{P(w_i|R=0,c)} \tag{8}$$

In multiclass classification, a single text classifier is generally trained per class to distinguish that class (positive class) from all other classes (negative class); this strategy is called one-vs-all (OvA) or one-vs-rest (OvR) (Galar, Fernández, Barrenechea, Bustince, & Herrera, 2011; Han et al., 2007). In this case, a positive class is denoted by c, and a negative class is by $\bar{c}$. Using this notation, Eq. (8) can be reconstructed as follows:

$$RSV(d,c) = log\left(\frac{P(\mathrm{d}|c)}{P(\mathrm{d}|\bar{\mathrm{c}})}\right) = log\prod_{i}^{n_d} \frac{P(w_i|c)}{P(w_i|\bar{\mathrm{c}})} = \sum_{i}^{n_d} \log \frac{P(w_i|c)}{P(w_i|\bar{c})} \ . \tag{9}$$

3.2. Applying the negative class information into the indexing and class prediction phases

The rightmost expression of Eq. (9), $\log\frac{P(w_i|c)}{P(w_i|\bar{c})}$, for a single word w_i, is called the Term Relevance Ratio (TRR) and it is regarded as the important score of w_i in a class c Ko, 2012). $P(w_i|c)$ and $P(w_i|\bar{c})$ can be estimated through the following Eqs. (10) and ((11). They are based on Laplace smoothing just like Eq. (3) and they are denoted by $\widehat{P_{tf}}(w_i|c)$ and $\widehat{P_{tf}}(w_i|\bar{c})$ for marking the estimation with traditional term frequency *tf*.

$$\widehat{P_{tf}}(w_i|\mathrm{c}) = \frac{1+\ \sum_{k=1}^{|D_c|} tf_{id_k}}{|V| + \sum_{j=1}^{|V|}\sum_{k=1}^{|D_c|} tf_{jd_k}} \tag{10}$$

$$\widehat{P_{tf}}(w_i|\bar{\mathrm{c}}) = \frac{1+\ \sum_{k=1}^{|D_{\bar{c}}|} tf_{id_k}}{|V| + \sum_{j=1}^{|V|}\sum_{k=1}^{|D_{\bar{c}}|} tf_{jd_k}}, \tag{11}$$

Table 1
Summary of a baseline NB classifier and the proposed NB classifiers.

Text Classification	Indexing	Class Prediction
TRA-NB (baseline) **RSV-NN**	original term frequency (*tf*), refer to Eqs. (10) and (11)	by Eq. (17) by Eq. (18)
TRR-TRA-NN **TRR-RSV-NN**	reformed term frequency (*rtf*) with TRR, refer to Eqs. (14) and (15)	by Eq. (19) by Eq. (20)

where $|V|$ is the number of unique words in labeled training documents, $|D_c|$ and $|D_{\bar{c}}|$ are the numbers of the labeled training documents in a given positive class c and its negative class $\bar{c}$, respectively, and tf_{id_k}represents the number of word w_i occurrences in document d_k in D_c.

We want to use TRR for representing a word occurrence instead of a binary weight. TRR is reformulated by adding a constant value α that is the base of the logarithmic operation. Eventually, TRR always becomes a positive value as follows:

$$TRR = \log\left(\frac{\widehat{P_{tf}}(w_i|c)}{\widehat{P_{tf}}(w_i|\bar{c})} + \alpha\right) \tag{12}$$

Therefore, the reformed term frequency rtf_{id} of w_i in document d is calculated by Eq. (13).

$$rtf_{id} = tf_{id} \cdot TRR \tag{13}$$

In the indexing phase of NB text classification, the reformed term frequency *rtf* using TRR is exploited instead of traditional term frequency *tf*. Actually, it is now no longer a probabilistically motivated generative model because a multinomial NB model cannot, by definition, generate documents with fractional term counts. Thus this new model is denoted by NN (Naive Bayes with Negative class information) to distinguish it and traditional NB models. Consequently, it is called TRR-indexed NN text classification, denoted by TRR-NN.

In TRR-NN, $P(w_i|c)$ and $P(w_i|\bar{c})$ are re-estimated by Eqs. (14) and (15) and they can be replaced instead of Eqs. (10) and (11) in the NB text classification. They are also denoted by $\widehat{P_{rtf}}(w_i|c)$ and $\widehat{P_{rtf}}(w_i|\bar{c})$, which mark that they are based on *rtf*.

$$\widehat{\mathrm{P}_{rtf}}(w_i|\mathrm{c}) = \frac{1 + \sum_{k=1}^{|D_c|} rtf_{id_k}}{|V| + \sum_{j=1}^{|V|} \sum_{k=1}^{|D_c|} rtf_{jd_k}} \tag{14}$$

$$\widehat{\mathrm{P}_{rtf}}(w_i|\bar{\mathrm{c}}) = \frac{1 + \sum_{k=1}^{|D_{\bar{c}}|} rtf_{id_k}}{|V| + \sum_{j=1}^{|V|} \sum_{k=1}^{|D_{\bar{c}}|} rtf_{jd_k}} \tag{15}$$

In addition, the RSV of document d and class c as *RSV(d,c)* by Eq. (9) can be directly applied to the traditional NB text classification (TRA-NB) by Eq. (5). It is conducted on simple replacement of $\hat{P}(w_i|c)$ with $\frac{\hat{P}(w_i|c)}{\hat{P}(w_i|\bar{c})}$ as the following equation.

$$\text{predicted\_class}_d = \underset{c\in C}{\text{argmax}}\left[\log\hat{P}(c) + \sum_{i=1}^{n_d} tf_{id}\log\left(\frac{\hat{P}(w_i|c)}{\hat{P}(w_i|\bar{c})}\right)\right] \tag{16}$$

It is called the RSV-based NN text classification, denoted by RSV-NN. As the RSV value for a term is a score, not a probability, it is also not a generative model so this model is also denoted by NN.

Consequentially, they lead to four different kinds of text classification models because TRR-NN with *rtf* is a new indexing approach and it can be applied to both of TRA-NB and RSV-NN. Table 1 summarizes these four kinds of text classification models and the following equations express each text classification models. TRA-NB is considered as the baseline model of this whole study.

$$\textbf{TRA}-\textbf{NB}:\ \text{predicted\_class}_d = \underset{c\in C}{\text{argmax}}\left[\log\hat{P}(c) + \sum_{i=1}^{n_d} tf_{id}\log\widehat{P_{tf}}(w_i|c)\right] \tag{17}$$

$$\textbf{RSV}-\textbf{NN}:\ \text{predicted\_class}_d = \underset{c\in C}{\text{argmax}}\left[\log\hat{P}(c) + \sum_{i=1}^{n_d} tf_{id}\log\left(\frac{\widehat{P_{tf}}(w_i|c)}{\widehat{P_{tf}}(w_i|\bar{c})}\right)\right] \tag{18}$$

$$\textbf{TRR}-\textbf{TRA}-\textbf{NN}:\ \text{predicted\_class}_d = \text{argmax}_{c\in C}\left[\log\hat{P}(c) + \sum_{i=1}^{n_d} rtf_{id}\log\widehat{P_{rtf}}(w_i|c)\right] \tag{19}$$

$$\textbf{TRR}-\textbf{RSV}-\textbf{NN}:\ \text{predicted\_class}_d = \text{argmax}_{c\in C}\left[\log\hat{P}(c) + \sum_{i=1}^{n_d} rtf_{id}\log\left(\frac{\widehat{P_{rtf}}(w_i|c)}{\widehat{P_{rtf}}(w_i|\bar{c})}\right)\right] \tag{20}$$

Table 2
Constitution of Korean newsgroup dataset (*KNG*).

Class	Training data	Test data	Total
han.arts.music	315	136	451
han.comp.database	198	86	284
han.comp.devtools	404	174	578
han.comp.lang	1387	595	1982
han.comp.os.linux	1175	504	1679
han.comp.os.window	517	222	739
han.comp.sys	304	131	435
han.politics	1469	630	2099
han.rec.cars	291	126	417
han.rec.games	261	112	373
han.rec.movie	202	88	290
han.rec.sports	130	56	186
han.rec.travel	102	45	147
han.sci	333	143	476
han.soc.religion	136	59	195
Total	7224	3107	10,331

4. Experiments

We tested the proposed NN text classification on two widely used datasets, Reuters 21,578 and 20 Newsgroups, and on the Korean UseNet dataset. In this section, we present the results from several experiments demonstrating the effectiveness of the proposed NN text classification and summarize its results.

4.1. Test collections

To test the proposed term weighting scheme, we made use of two newsgroup datasets, 20 Newsgroups and the Korean UseNet dataset, written in two different languages, English and Korean, and the Reuters 21,578 dataset.

First, two widely used datasets were exploited as the benchmark.

The Reuters 21578 Distribution 1.0 dataset (*Reuters*) consists of 12,902 articles and 90 topic categories from the Reuters newswire (Debole & Sebastiani, 2003; Gliozzo, Strapparava, & Dagan, 2005; Ke, 2012; Rehman, Javed, Babri, & Saeed, 2015; Sun, Lim, & Liu, 2009; Sun, Lim, & Ng, 2003; Yang & Liu, 1999; Yu, Zhai, & Han, 2003). Following other studies by Nigam (2001) and Joachims (1998), we built binary classifiers for each class to identify the news topic. Since the documents in this dataset can have multiple class labels, each class is traditionally evaluated using a binary classifier. To split the training/test data, we follow a standard *ModApte* split (Lewis, 1997). The standard *ModApte* training/test split divides the articles by time, and the top-ten largest classes were used in the experiments. All words inside the title and body were used, along with a stopword list[1] and no stemming.

The Newsgroups dataset (*NG*), collected by Ken Lang, contains about 20,000 documents evenly divided among 20 UseNet discussion groups (Banerjee & Basu, 2007; Gliozzo et al., 2005; Ke, 2012; McCallum & Nigam, 1998b; Rehman et al., 2015; Sun et al., 2009; Yoon, Lee, & Lee, 2006). For a fair evaluation, we evaluated our scheme using the five-fold cross-validation method. All results of the experiments using this dataset are the averages of five runs. After removing words that occur only once or are on a stopword list, the vocabulary from the training data included 51,018 words (with no stemming).

Secondly, the dataset (*KNG*) was gathered from the Korean UseNet group (Ko, Park, & Seo, 2004). This dataset contains a total of 10,331 documents and 15 categories. In total, 3107 documents (30%) were used for test data, and the remaining 7224 documents (70%) were used for the training data. The resulting vocabulary from the training data has 69,793 words. The standard *ModApte* training/test split divides the articles by time, and the top-ten largest classes were used in the experiments. This dataset is uneven and has a fixed training/test split, as shown in Table 2.

The development data were randomly sampled from the training data of each dataset. They consist of 20% documents of training data and they were used for the optimization of the Dirichlet prior parameter.

4.2. Experimental settings

These datasets have very different characteristics. *Reuters* has a skewed class distribution and many of the documents have two or more class labels, whereas *NG* has a uniform class distribution and its documents have only one class label. On the other hand, *KNG* has a non-uniform class distribution and its documents have only one class label written in another language, Korean. Thus two different measures were used to evaluate various term weighting schemes on three different

[1] This stopword list was built by Gerard Salton and Chris Buckley for the experimental SMART information retrieval system at Cornell University and any stopword list was not used for KNG written in Korean because features were extracted as only contents words (noun, verbs, foreign words) after morphological analysis.

Table 3
Comparison of the *RSV-NN* and *RSV-NNK* text classifiers and the baseline systems.

	Reuters		NG		KNG	
	micro-averaging BEP	macro-averaging BEP	micro-averaging F_1	macro-averaging F_1	micro-averaging F_1	macro-averaging F_1
TRA-NB$_{\text{baseline}}$	91.71	82.97	82.65	82.16	81.68	80.85
RSV-NN	**91.78**	**83.35**	**83.51***	**83.00***	**83.25***	**82.28***
TRA-NBK$_{\text{baseline}}$	91.92	83.21	82.29	81.89	78.11	77.24
RSV-NNK	**91.96**	**83.49**	**83.09**$^{\#}$	**82.56**$^{\#}$	**82.19***	**80.83***

*statistically significant ($p \leq 0.01$) and # statistically significant ($p \leq 0.05$)

Table 4
Comparison of the *TRR-TRA-NN* and *TRR-TRA-NNK* text classifiers and the baseline systems.

	Reuters		NG		KNG	
	micro-averaging BEP	macro-averaging BEP	micro-averaging F_1	macro-averaging F_1	micro-averaging F_1	macro-averaging F_1
TRA-NB$_{\text{baseline}}$	91.71	82.97	82.65	82.16	81.68	80.85
TRR-TRA-NN	**93.00**$^{\#}$	**85.91**$^{\#}$	**85.21***	**84.87***	**84.22***	**82.74***
TRA-NBK$_{\text{baseline}}$	91.92	83.21	82.29	81.89	78.11	77.24
TRR-TRA-NNK	**93.07**$^{\#}$	**86.14**$^{\#}$	**84.98***	**84.63***	**83.86***	**82.73***

*statistically significant ($p \leq 0.01$) and # statistically significant ($p \leq 0.05$)

Table 5
Comparison of the *TRR-RSV-NN* and *TRR-RSV-NNK* text classifiers and the baseline systems.

	Reuters		NG		KNG	
	micro-averaging BEP	macro-averaging BEP	micro-averaging F_1	macro-averaging F_1	micro-averaging F_1	macro-averaging F_1
TRA-NB$_{\text{baseline}}$	91.71	82.97	82.65	82.16	81.68	80.85
TRR-RSV-NN	**93.09**$^{\#}$	**85.77**$^{\#}$	**85.46***	**85.09***	**84.10***	**82.42***
TRA-NBK$_{\text{baseline}}$	91.92	83.21	82.29	81.89	78.11	77.24
TRR-RSV-NNK	**93.51**$^{\#}$	**86.89**$^{\#}$	**85.09***	**84.71***	**84.41***	**83.33***

*statistically significant ($p \leq 0.01$) and # statistically significant ($p \leq 0.05$)

datasets for our experiments. For *Reuters*, we used the Break Even Point (*BEP*) measure, which is a standard information retrieval measure for binary classification, and for *NG* and *KNG*, the performance is reported using an F_1 measure. Herein, we followed the standard definition of recall *(r)*, precision *(p)*, and F_1 measure $(2rp/(r+p))$. To evaluate the average performance across categories, we used the *micro-averaging* and *macro-averaging* methods (Ko & Seo., 2009; Yang, 1999). The recall, precision, and F_1 measure can first be computed for individual categories, and then averaged over the categories as a global measure of the average performance; this method of averaging is called *macro-averaging*. An alternative method, *micro-averaging*, is used to count the decisions for all categories in a joint pool and compute the global recall, precision, and F_1 values for that global pool. Herein, we report the statistical significance for the observed differences in BEP and F_1 values of category pairs for *Reuters* and *KNG* and fold pairs for *NG* when the *p* value is sufficiently small ($p < 0.01$ or $p < 0.05$) based on a one-sided paired *t*-test[2] (Smucker, Allan, & Carterette, 2007; Yang & Liu, 1999).

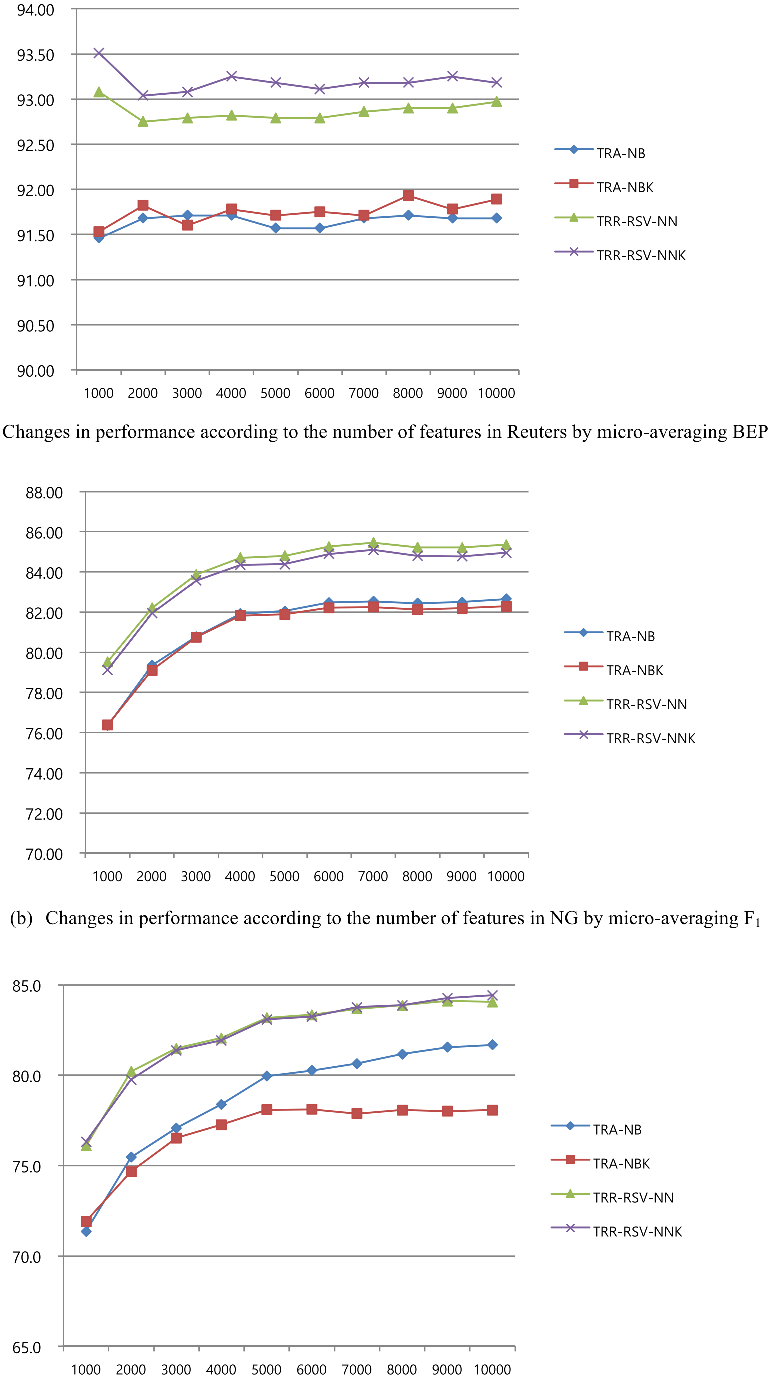
We used χ^2 statistics for the statistical feature selection (Aggarwal & Zhai, 2012; Yang et al., 1997). The χ^2 statistics values are calculated through the following equation.

$$\chi^2(w_i, c) = \frac{[P(w_i, c)P(\bar{w}_i, \bar{c}) - P(\bar{w}_i, c)P(w_i, \bar{c})]^2}{P(w_i)P(\bar{w}_i)P(c)P(\bar{c})}, \tag{21}$$

where $\mathrm{P}(\bar{w}_i, \mathrm{c})$ indicates the probability that, for a random document *d*, word w_i does not occur in *d*, which belongs to class *c*, and is estimated based on the maximum likelihood. As a globalization function, $f_{\max}(w_i) = max_{j=1}^{|C|}\chi^2(w_i, c_j)$ is selected for the feature selection.

Because we know nothing about the class of a test document, it is difficult to calculate *rtf* of each word in the test document: rtf_{id} in Eq. (19) for TRR-TRA-NN and rtf_{id} in Eq. (20) for TRR-RSV-NN. To solve this problem, we just followed our previous research results (Ko, 2012). The *rtf* of each word in a test document is first calculated into |*C*| different values by supposing that the class of the test document is one of |*C*| classes, where |*C*| is the number of classes. Then two methods are used for choosing the final *rtf* values. The first one is that the sum of *rtf* values of all words in the test document is calculated for each class and the *rtf* values of a class with the maximum sum value are then selected as the final *rtf* values for words of the test document. This is applied for the *NG* and *KNG* datasets. Since many documents of the *Reuters*

[2] For *Reuters* and *KNG*, we tested statistical significance on differences in F_1 values of category pairs by Macro *t*-test (Yang & Liu 1999). Because the evaluation on *NG* is based on the five-fold cross validation, we did it on differences in F_1 values of fold pairs.



(a) Changes in performance according to the number of features in Reuters by micro-averaging BEP

(b) Changes in performance according to the number of features in NG by micro-averaging F_1

(c) Changes in performance according to the number of features in KNG by micro-averaging F_1

Fig. 1. Comparison of *TRR-RSV-NN* and *TRA-NB$_{baseline}$* on changes in performance according to the number of features.

dataset have two or more class labels, the *rtf* values of their words have to be calculated by a different method for more effectively resolving their ambiguities. Indeed, after the sum of *rtf* values of all words in a test document is calculated for each class, two classes with the highest and second highest sum values are selected and the final *rtf* value of a word in the test document is chosen by a higher *rtf* value between the *rtf* values of the word in two selected classes.

In the experiments, a new NB classifier with minor modifications based on Kullback-Leibler Divergence (Craven et al., 2000; Ko et al. 2000) is employed and its usage supports the verification of the proposed method's excellency because it

Table 6
Performance comparison on the Reuters dataset.

Category (# of training documents)	*TRA-NB$_{baseline}$*	*TRR-RSV-NN*	*TRA-NBK$_{baseline}$*	*TRR-RSV-NNK*
Acq (1650)	97.50	97.08	96.94	97.08
Corn (181)	62.50	76.79	62.50	78.57
Crude (389)	89.95	89.95	89.42	91.01
Earn (2877)	97.15	98.25	97.24	98.25
Grain (433)	92.62	93.29	92.62	93.96
Interest (347)	76.34	80.15	77.86	80.92
Money-fx (538)	77.65	81.56	79.89	82.12
Ship (197)	80.90	84.27	82.02	84.27
Trade (369)	79.49	84.62	84.62	88.03
Wheat (212)	73.24	71.83	69.01	74.65
micro-avg. BEP	91.71	**93.09 (+1.50)**	91.92	**93.51 (+1.73)**
macro-avg. BEP	82.97	**85.77 (+3.37)**	83.21	**86.89 (+4.42)**

can prove that the proposed method works well in another framework of the NB classifier.

$$\text{predicted\_class}_d = \underset{c \in C}{\operatorname{argmax}} \left[\frac{\log \hat{P}(c)}{n} + \sum_{i=1}^{n_d} \hat{P}(w_i|d) \log \left(\frac{\hat{P}(w_i|c)}{\hat{P}(w_i|d)} \right) \right], \tag{22}$$

where n is the total number of words in d.

We can denote four text classification models in Table 1 using Eq. (22) instead of Eq. (5) by TRA-NBK, RSV-NNK, TRR-TRA-NNK and TRR-RSV-NNK.

4.3. Verification of the RSV-NN text classification

In this subsection, the RSV-NN and RSV-NNK text classifiers are verified by comparing with the baseline systems. As can be seen in Table 3, they achieved better performance than their baseline systems on all the datasets individually. Therefore, we believe that the usage of the negative class information is more effective in class prediction than the traditional NB classification. The statistical significance tests indicate that the differences in the NG and KNG datasets are all statistically significant ($p < 0.01$) except for the RSV-NNK text classification in the NG dataset; its difference is statistically significant ($p < 0.05$). Unfortunately, the RSV-NN and RSV-NNK text classification on the Reuters dataset failed in statistical significance tests on *t*-test due to their very slight improvement.

4.4. Verification of the TRR-TRA-NN text classification

Next, we try to verify the TRR-TRA-NN and TRR-TRA-NNK text classification, in which the negative information is exploited for the indexing phase as *rtf*. Table 4 shows that they obtained better performance than their baseline systems individually. As a result, we could achieve more improvement than performance when the RSV-NN text classification is applied at Section 4.3. The statistical significance tests indicate that the differences in the NG and KNG datasets are all statistically significant ($p < 0.01$) and the differences in the Reuter dataset is statistically significant ($p < 0.05$).

4.5. Verification of the TRR-RSV-NN text classification

In this subsection, we try to verify the TRR-RSV-NN and TRR-RSV-NNK text classification, in which the negative information is exploited for the class prediction phase as well as the indexing phase as *rtf*. Table 5 shows that they obtained better performance than their baseline systems individually and the TRR-RSV-NN text classification showed the best performance among four NB and NN classification approaches (TRA-NB, RSV-NN, TRR-TRA-NN and TRR-RSV-NN) except for TRR-TRA-NN in KNG. The statistical significance tests indicate that the differences in the NG and KNG datasets are all statistically significant ($p < 0.01$) and the differences in the Reuter dataset is statistically significant ($p < 0.05$), similar to results of TRR-TRA-NN.

Fig. 1 shows the performance changes based on the number of features, from 1000 to 10,000, for comparing the baseline model and TRR-RSV-NN. The TRR-RSV-NN classification achieved the better performance than the baseline systems on each interval in both of the NB classification frameworks, original (TRA-NB) and KL (TRA-NBK). These three graphs have something in common with the grouping of performance scores. That is, the shapes of performance changes in the TRR-RSV-NN classifiers are similarly grouped and ones of the baseline models are also grouped, and the similar differences between these two groups are observed in all the intervals. The best performance in the Reuters dataset is achieved on 1000 features whereas the best one in the NG and KNG datasets is on 10,000 features and the performance is converged as more features are used.

Tables 6, 7 and 8 show the results about a comparison of the baseline model and TRR-RSV-NN with respect to performance on each class and the overall performance after micro-averaging and macro-averaging. The TRR-RSV-NN text classification achieved a better performance than the baseline systems in most classes of all three datasets. In particular, the

Table 7
Performance comparison on the Newsgroup dataset.

Category (# of training documents)	*TRA-NB*$_{baseline}$	*TRR-RSV-NN*	*TRA-NBK*$_{baseline}$	*TRR-RSV-NNK*
Atheism (800)	76.12	77.54	76.15	76.62
Graphics (800)	74.94	80.14	74.77	77.21
Windows.misc (800)	57.83	73.91	53.83	64.62
Pc.hardware (800)	68.86	74.12	67.78	70.98
Mac.hardware (800)	82.09	86.69	80.89	83.81
Windows.x (800)	84.35	87.46	84.71	85.86
Forsale (800)	78.89	78.62	79.08	78.85
Autos (800)	90.69	92.79	90.02	91.45
Motorcycles (800)	94.90	96.31	94.63	95.49
Baseball (800)	95.87	96.60	95.77	96.27
Hockey (800)	96.38	96.81	96.47	96.67
Crypt (800)	92.70	95.35	92.66	93.86
Electronics (800)	78.18	82.18	77.40	79.87
Med (800)	91.36	92.97	91.00	91.96
Space (797)	91.49	94.27	91.26	92.62
Christian (800)	90.87	92.02	91.09	91.30
Guns (800)	83.10	85.32	82.54	83.78
Midest (800)	91.54	93.03	91.38	92.15
Politics.misc (800)	71.02	73.92	71.01	72.51
Religion.misc (800)	51.32	51.74	51.71	51.13
micro-avg. BEP	82.65	**85.46 (+3.40)**	82.29	**85.09 (+3.40)**
macro-avg. BEP	82.16	**85.09 (+3.57)**	81.89	**84.71 (+3.44)**

Table 8
Performance comparison on the Korean Newsgroup dataset.

Category (# of training documents)	*TRA-NB*$_{baseline}$	*TRR-RSV-NN*	*TRA-NBK*$_{baseline}$	*TRR-RSV-NNK*
han.arts.music (315)	85.61	87.12	84.06	86.59
han.comp.database (198)	69.94	76.92	67.34	76.47
han.comp.devtools (404)	68.47	72.13	63.85	71.66
han.comp.lang (1387)	82.82	85.86	79.54	85.19
han.comp.os.linux (1175)	72.65	79.63	68.20	80.75
han.comp.os.window (517)	67.90	72.77	63.08	73.85
han.comp.sys (304)	66.67	68.44	61.72	68.64
han.politics (1469)	95.95	93.87	95.15	94.12
han.rec.cars (291)	94.40	95.55	93.93	95.94
han.rec.games (261)	76.36	75.35	68.64	73.45
han.rec.movie (202)	82.28	83.02	83.91	87.27
han.rec.sports (130)	87.04	87.38	80.67	90.38
han.rec.travel (102)	85.37	82.05	66.15	88.64
han.sci (333)	85.93	86.36	84.25	86.79
han.soc.religion (136)	90.77	89.47	91.94	90.27
micro-avg. BEP	81.68	**84.10 (+2.96)**	78.11	**84.41 (+8.07)**
macro-avg. BEP	80.85	**82.42 (+1.94)**	77.24	**83.33 (+7.88)**

proposed TRR-RSV-NN text classification outperformed the baseline systems in both of binary classification for the Reuters dataset and multi-class classification for the NG and KNG datasets. In the Reuters and KNG datasets with skewed class distribution, it also achieved better performance in most categories with small number of documents.

4.6. Summary of experiment results for the proposed methods

Table 9 summarizes the final results of all the proposed NN text classification approaches used in this experiment. The results show that the TRR-RSV-NN text classification, applying negative class information in both of indexing and class prediction, achieved the best performance over all of the datasets. There is only one exception at original NB on KNG that TRR-TRA-NN obtained slightly better performance than TRR-RSV-NN. Note that the scores inside parentheses in TRR-RSV-NN means relative improvements from TRA-NB to TRR-RSV-NN. Moreover, Fig. 2 illustrates the performance changes from TRA-NB to TRR-RSV-NN on each dataset.

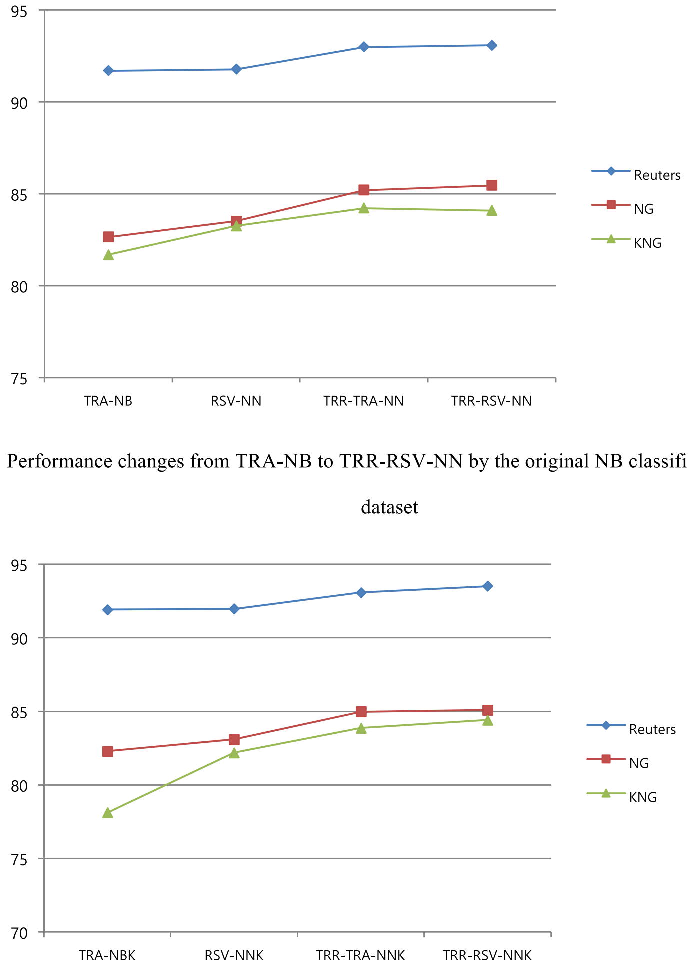
4.7. Applying Dirichlet smoothing to the proposed NN text classification

The parameters $P(w_i|c)$ in all the proposed NN models as well as the baseline NB model are estimated with Laplace smoothing because it is the most common smoothing method for the NB text classification and the purpose of this research is how to utilize the negative class information, not smoothing. However, since more sophisticated smoothing methods such

Table 9
Summary of experiment results from TRA-NB to TRR-RSV-NN.

			TRA-NB	*RSV-NN*	*TRR-TRA-NN*	*TRR-RSV-NN*
Reuters (BEP)	***Original NB***	***micro-averaging***	91.71	91.78	93.00	**93.09 (+1.50)**
		macro-averaging	82.97	83.21	85.91	**85.77 (+3.37)**
	KL based NB	***micro-averaging***	91.92	91.96	93.07	**93.51 (+1.73)**
		macro-averaging	83.21	83.49	86.14	**86.89 (+4.42)**
NG (F_1)	***Original NB***	***micro-averaging***	82.65	83.51	85.21	**85.46 (+3.40)**
		macro-averaging	82.16	83.00	84.87	**85.09 (+3.57)**
	KL based NB	***micro-averaging***	82.29	83.09	84.98	**85.09 (+3.40)**
		macro-averaging	81.89	82.56	84.63	**84.71 (+3.44)**
KNG (F_1)	***Original NB***	***micro-averaging***	81.68	83.25	84.22	**84.10 (+2.96)**
		macro-averaging	80.85	82.28	82.74	**82.42 (+1.94)**
	KL based NB	***micro-averaging***	78.11	82.19	83.86	**84.41 (+8.07)**
		macro-averaging	77.24	80.83	82.73	**83.33 (+7.88)**

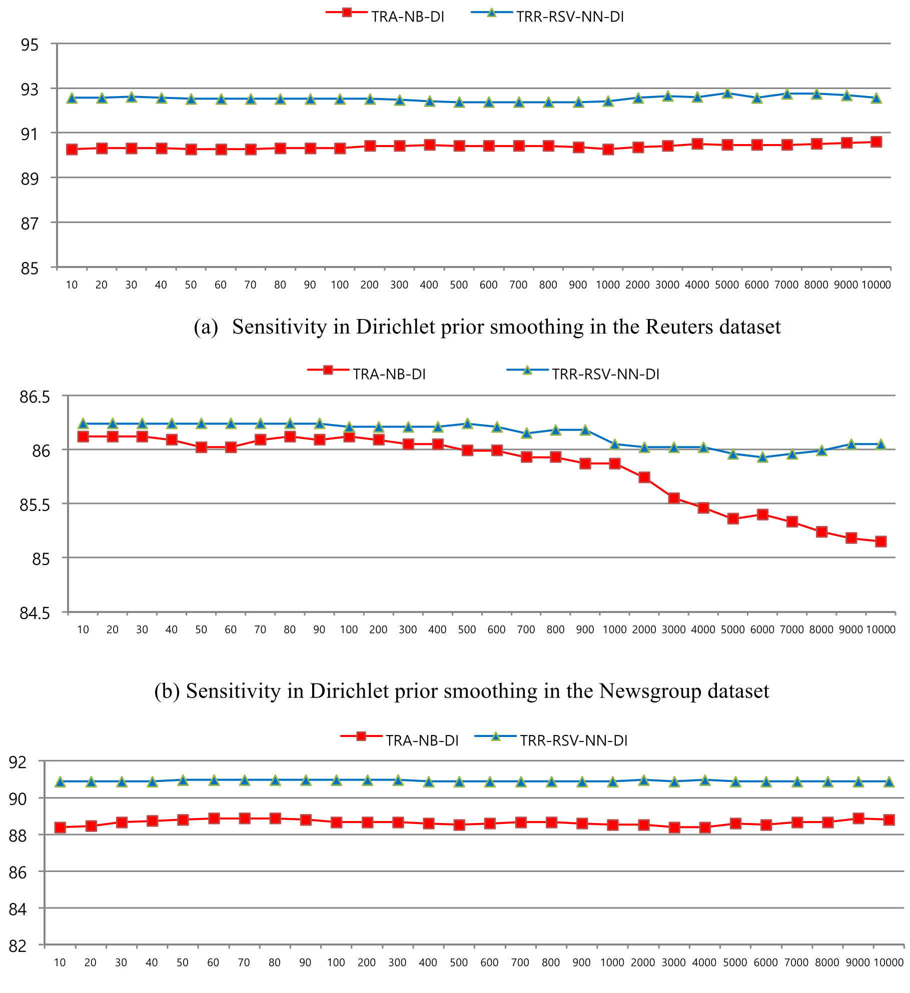
*Original NB describes the original NB based models, NB and NN, and KL based NB does the Kullback-Leibler Divergence based NB models, NBK and NNK.



(a) Performance changes from TRA-NB to TRR-RSV-NN by the original NB classifier on each dataset

(b) Performance changes from TRA-NBK to TRR-RSV-NNK by the KL based NB classifier on each dataset

Fig. 2. Comparison of changes in performance from TRA-NB to TRR-RSV-NN.



(a) Sensitivity in Dirichlet prior smoothing in the Reuters dataset

(b) Sensitivity in Dirichlet prior smoothing in the Newsgroup dataset

(c) Sensitivity in Dirichlet prior smoothing in the Korean Newsgroup dataset

Fig. 3. Sensitivity of BEP in Reuters and F1 in NG and KNG in for Dirichlet prior smoothing.

as Dirichlet smoothing improved performance in text classification as well as information retrieval (Bai & Nie, 2004; Zhai & Lafferty, 2004), we apply the Dirichlet smoothing method to the baseline model and the proposed model by the following Eqs. (23) and (24).

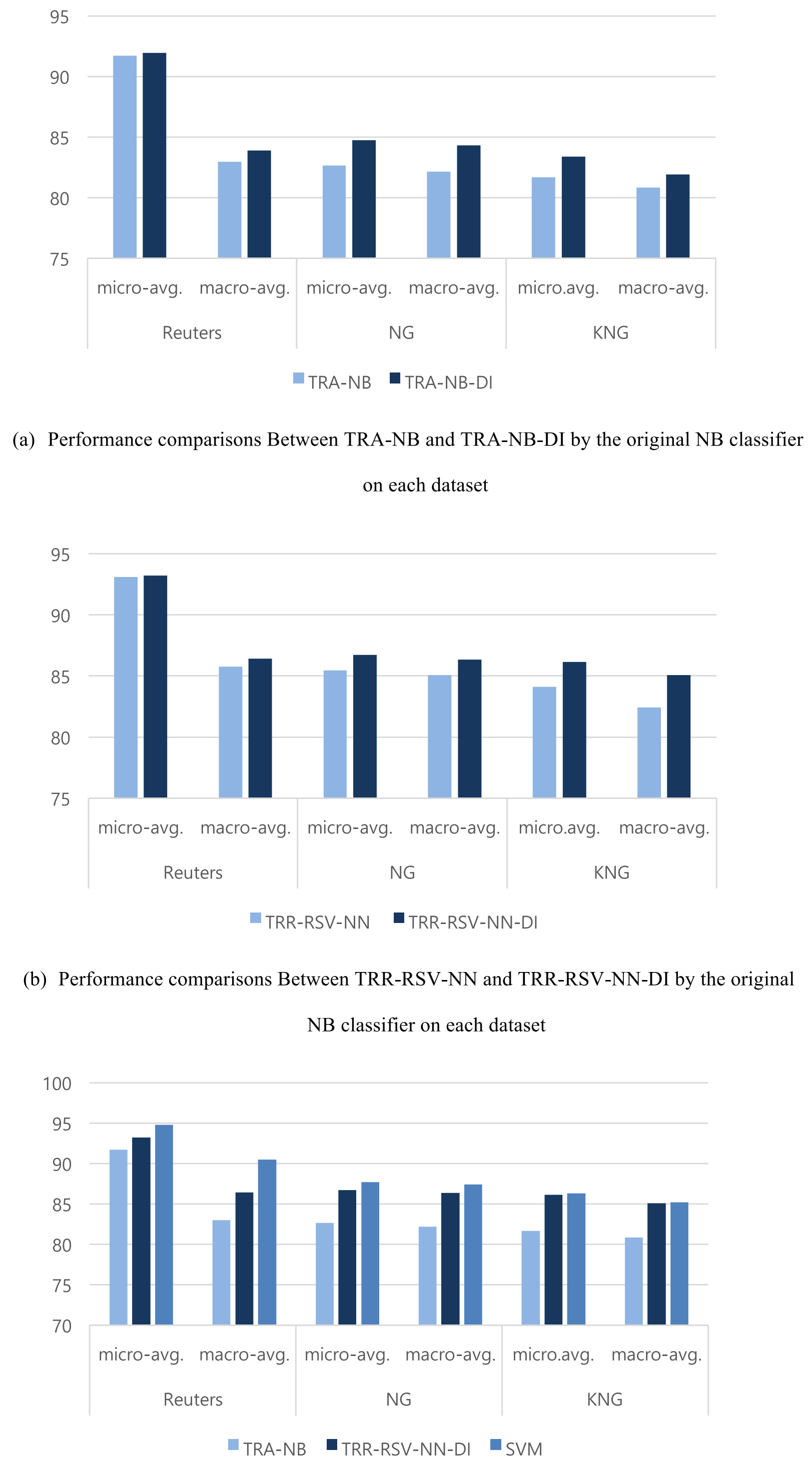
$$\widehat{P_{tf}}(w_i|\text{c}) = \frac{\mu P(w_i|Cor) + \ \sum_{k=1}^{|D_c|} tf_{id_k}}{\mu + \sum_{j=1}^{|V|} \sum_{k=1}^{|D_c|} tf_{jd_k}} \tag{23}$$

$$\widehat{P_{tf}}(w_i|\bar{\text{c}}) = \frac{\mu P(w_i|Cor) + \ \sum_{k=1}^{|D_{\bar{c}}|} tf_{id_k}}{\mu + \sum_{j=1}^{|V|} \sum_{k=1}^{|D_{\bar{c}}|} tf_{jd_k}}, \tag{24}$$

where *Cor* is a whole corpus and μ is the Dirichlet prior parameter. These are equations for Dirichlet smoothing in the baseline model, TRA, modified by Eq. (10) and (11), and ones in the proposed model, TRR, are also modified in the same manner from Eq. (13) and (14).

For Dirichlet smoothing, we varied the value of the smoothing parameter and recorded the classification performance at each parameter value. The results are plotted in Fig. 3. From these figures, we easily see that the performance changes in TRR-RSV-NN-DI behave similarly, as do those in TRA-NB-DI on all three datasets, and there are little performance changes in whole parameter values on the Reuters and KNG datasets. As a result, the Dirichlet prior parameter (μ) for each dataset was set as follows (Table 10):

Table 11 reports the performance of the NB and NN text classifiers with the Dirichlet smoothing, TRA-NB-DI and TRR-RSV-NN-DI, and Fig. 4 illustrates the comparisons between them and the original text classifiers. Consequently, we achieved significant improvements in TRR-RSV-NN-DI as well as TRA-NB-DI. Moreover, we added the results of SVM because we want



(a) Performance comparisons Between TRA-NB and TRA-NB-DI by the original NB classifier on each dataset

(b) Performance comparisons Between TRR-RSV-NN and TRR-RSV-NN-DI by the original NB classifier on each dataset

(c) Performance comparisons among TRA-NB, TRR-RSV-NN-DI and SVM by the original NB classifier on each dataset

Fig. 4. Comparison of NB and NN with Dirichlet smoothing and others (TRA-NB, TRR-RSV-NN and SVM).

Table 10
Optimal Dirichlet prior parameters for TRA-NB-DI and TRR-RSV-NN-DI in each dataset.

	Reuters (BEP)	NG (F_1)	KNG (F_1)
TRA-NB-DI	$\mu = 10{,}000$	$\mu = 100$	$\mu = 70$
TRR-RSV-NN-DI	$\mu = 7000$	$\mu = 90$	$\mu = 70$

Table 11
Comparison of NB and NN with Dirichlet smoothing and others (TRA-NB, TRR-RSV-NN and SVM).

		TRA-NB	TRA-NB-DI	TRR-RSV-NN	TRR-RSV-NN-DI	SVM
Reuters(BEP)	***micro-averaging***	**91.71**	**91.96 (+0.27)**	**93.09**	**93.21 (+0.13)**	**94.76 (+1.66.)**
	macro-averaging	**82.97**	**83.91 (+1.13)**	**85.77**	**86.43 (+0.77)**	**90.46 (+4.66)**
NG(F_1)	***macro-averaging***	**82.65**	**84.74 (+2.53)**	**85.46**	**86.72 (+1.47)**	**87.69 (+1.12)**
	macro-averaging	**82.16**	**84.33 (+2.64)**	**85.09**	**86.34 (+1.47)**	**87.40 (+1.23)**
KNG(F_1)	***micro-averaging***	**81.68**	**83.38 (+ 2.08.)**	**84.10**	**86.15 (+ 2.44)**	**86.31 (+ 0.19)**
	macro-averaging	**80.85**	**81.93 (+ 1.34)**	**82.42**	**85.09 (+ 3.24)**	**85.21 (+ 0.14)**

to compare the proposed classifier with the state-of-the-art text classifier. Note that the scores inside parentheses in TRA-NB-DI means relative improvements from TRA-NB to TRA-NB-DI, ones in TRR-RSV-NN-DI means relative improvements from TRR-RSV-NN to TRR-RSV-NN-DI and ones in SVM denotes relative improvements from TRR-RSV-NN-DI to SVM. Consequently, SVM still showed better than the proposed model, TRR-RSV-NN-DI, but the proposed model can reduce the performance differences about 75% between SVM and TRA-NB on average over three datasets.

Even though SVM showed the better performance than NB and NN in the experiments, I want to focus on that the proposed NN classifiers outperformed the traditional NB classifiers and their performance scores became closer to those of SVM. In addition, the NB classifier has several strong points related to its simplicity and demand for small amount of training data. NB is one of simplest techniques that construct classifiers based on the basic and strong probability theory. Despite its naive design and assumption, NB classifiers have worked quite well in many complex real-world situations. In particular, NB has been used as a week learner for boosting algorithms such as AdaBoost, and Bayesian spam filtering is a popular statistical technique of e-mail filtering. In these application areas, the advantage of NB is caused by the fact that it requires a small amount of training data. One of the main advantages of Bayesian spam filtering is that it can be well trained on a per-user basis. It becomes an important strong point for spam filtering because the spam that a user receives is often related to the online user's activities and its amount is commonly small. For the boosting algorithms, NB is one of classifiers frequently used as a weak learner. By these reasons, getting higher performance of NB is still important in many applications even though its performance does not outperform SVM.

5. Conclusions and future work

In this paper, we presented how to utilize negative class information for the NB text classification. The negative class information is applied to the indexing and class prediction phases of text classification. As a result, the proposed NN text classification performed consistently well on two different NB frameworks as well as on the three benchmark datasets used.

In the future, we would like to apply the proposed NN text classification to other applications such as question classification in cQA systems. Since a question consists of a very short number of words (or features) and the proposed NN technique provides better the feature weight scheme to them, the question classification could be a challengeable application to apply the proposed NB technique. In addition, since the NB text classification has several strong points of easy implementation and understanding, we expect that it can be used in various text-mining tasks such as a sentiment analysis and text summarization.

Acknowledgment

This work was supported by the Dong-A University research fund.

References

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. *Chapter 6. Mining text data*.

Bai, J., & Nie, J.-Y. (2004). Using language models for text classification. In *Proceedings of Asia information retrieval symposium conference*.

Banerjee, A., & Basu, S. (2007). Topic models over text streams: A study of batch and online unsupervised learning. In *Proceedings of conference on data mining* (pp. 431–436).

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., et al. (2000). Learning to construct knowledge bases from the world wide web. *Artificial Intelligence, 118*(1-2), 69–113.

Debole, F., & Sebastiani, F. (2003). Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM symposium on applied computing* (pp. 784–788). 2003.

Dumais, S., Plat, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representation for text catgorization. In *Proceedings of CIKM-98, seventh ACM international conference information and knowledge management* (pp. 148–155).

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition, Elsevier Science, 44*(8), 1761–1776.

Gliozzo, A., Strapparava, C., & Dagan, I. (2005). Investigating unsupervised learning for text categorization bootstrapping. In *Proceedings of the conference on human language technology and empirical methods in natural language processing* (pp. 129–136).

Gordon, M. D., & Lenk, P. (1991). A utility theoretic examination of the probability ranking principle in information retrieval. *Journal of the American Society for Information Science and Technology, 42*(10), 703–714.

Gordon, M. D., & Lenk, P. (1992). When is the probability ranking principle suboptimal. *Journal of the American Society for Information Science and Technology, 43*(1), 1–14.

Han, H., Ko, Y., & Seo, J. (2007). Using the revised EM algorithm to remove noisy data for improving the one-against-the-rest method in binary text classification. *Information Processing and Management, Pergamon-Elsevier Science, 43*(5), 1281–1293.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML-98, 10th European conference on machine learning* (pp. 137–142).

Ke, W. (2012). Least information document representation for automated text classification. In *Proceedings of the American society for information science and technology (ASIST 2012): 49* (pp. 1–10).

Ko, Y. (2012). A study of term weighting scheme using class information for text classification. In *Proceedings of the 35th annual international ACM SIGIR conference (SIGIR 2012)* (pp. 1029–1030).

Ko, Y., Park, J., & Seo, J. (2004). Improving text categorization using the importance of sentences. *Information Processing and Management, Pergamon-Elsevier Science, 40*(1), 65–79.

Ko, Y., & Seo, J. (2009). Text classification from unlabeled documents with bootstrapping and feature projection techniques. *Information Processing and Management, Pergamon-Elsevier Science, 45*(1), 70–83.

Ko, Y., & Seo, J. (2000). Automatic text categorization by unsupervised learning. In *Proceedings of the 18th international conference on computational linguistics (COLING)* (pp. 435–459).

Lewis, D. D. (1997). Reuters-21578 text categorization test collection. *ReadMe file of Distribution 1.0*.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European conference machine learning* (pp. 4–15).

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

McCallum, A. K., & Nigam, K. (1998a). A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI-98 workshop on learning for text categorization* (pp. 41–48).

McCallum, A. K., & Nigam, K. (1998b). Employing EM in pool-based active learning for text classification. In *Proceedings of ICML-98, 15th international conference machine learning* (pp. 350–358).

Nigam, K. P. (2001). *Using unlabeled data to improve text classification, the dissertation for the degree of Dcotor of Philosophy*.

Rehman, A., Javed, K., Babri, H. A., & Saeed, M. (2015). Relative discrimination criterion-a novel feature ranking method for text data. *Expert Systems with Applications, 42*(7), 3670–3681.

Sebastiani, F. (2002). Machine learning in automatic text retrieval. *ACM Computing Surveys (CSUR), 34*(1), 1–47.

Smucker, M., Allan, J., & Carterette, B. (2007). A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the ACM sixteenth conference on information and knowledge management (CIKM 2007)* (pp. 623–632).

Sun, A., Lim, E.-P., & Liu, Y. (2009). On strategies for imbalanced text classification using SVM: A comparative study. *Decision Support Systems, 48*(1), 191–201.

Sun, A., Lim, E.-P., & Ng, W.-K. (2003). Performance measurement framework for hierarchical text classification. *Journal of the American Society for Information Science and Technology, 54*(11), 1014–1028.

Van Rijsbergen, C. J. (1979). *Information retrieval* (2nd edition). Butterworths.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval, 1*(1/2), 67–88.

Yang, Y., & Chute, X. (1994). An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems, 12*(3), 252–277.

Yang, Y., & Pedersen, J. P. (1997). Feature selection in statistical learning of text categorization. In *Proceedings of the fourteenth international conference on machine learning* (pp. 412–420).

Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM international conference research and development in information retrieval* (pp. 42–49).

Yoon, Y., Lee, C., & Lee, G. G. (2006). An effective procedure for constructing a hierarchical text classification system. *Journal of the American Society for Information Science and Technology, 57*(3), 431–442.

Yu, H., Zhai, C., & Han, J. (2003). Text classification from positive and unlabeled documents. In *Proceedings of the 13th ACM international conference on information and knowledge management* (pp. 232–239).

Zhai, C., & Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems., 22*(2), 179–214.