Expert Systems With Applications 94 (2018) 218–227

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Opinion mining using ensemble text hidden Markov models for text classification

Mangi Kang, Jaelim Ahn, Kichun Lee*

Department of Industrial Engineering, Hanyang University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 21 March 2017
Revised 13 July 2017
Accepted 14 July 2017
Available online 19 July 2017

Keywords:
Opinion mining
Sentiment analysis
Hidden Markov models
Ensemble
Boosting
Clustering

ABSTRACT

With the rapid growth of social media, text mining is extensively utilized in practical fields, and opinion mining, also known as sentiment analysis, plays an important role in analyzing opinion and sentiment in texts. Methods in opinion mining generally depend on a sentiment lexicon, which is a set of predefined key words that express sentiment. Opinion mining requires proper sentiment words to be extracted in advance and has difficulty classifying sentences that imply an opinion without using any sentiment key words. This paper presents a new sentiment analysis method, based on text-based hidden Markov models (TextHMMs), for text classification that uses a sequence of words in training texts instead of a predefined sentiment lexicon. We sought to learn text patterns representing sentiment through ensemble TextHMMs. Our method defines hidden variables in TextHMMs by semantic cluster information in consideration of the co-occurrence of words, and thus calculates the sentiment orientation of sentences by fitted TextHMMs. To reflect diverse patterns, we applied an ensemble of TextHMM-based classifiers. In the experiments with a benchmark data set, we show that this method is superior to some existing methods and particularly has potential to classify implicit opinions. We also demonstrate the practicality of the proposed method in a real-life data set of online market reviews.

1. Introduction

Opinion mining, also known as sentiment analysis, is the study and application of text-related computational methods, such as natural language processing and text mining, to identify people's opinions about products, movies, news, etc. In practical life, the need to analyze sentiment and explore opinion has been quite essential (Ravi & Ravi, 2015). When people make a decision, they refer to others' opinions. For instance, a person inquires of others what he or she should do before purchasing a product. Most companies have conducted surveys and market research to catch consumers' opinion about their products. The need for sentiment analysis grows substantially in the era of the internet. With the rapid growth of e-commerce, the online purchase of goods is dominant, and the purchasers often write reviews (Liu, 2012). Such online reviews play an important role in others' decision making, especially with active social media. Recently, more people have taken a close look at such reviews and comments on goods on the web before they buy a product. Due to this phenomenon, not only local stores but also enterprises want to maintain positive reviews of their products and services.

The research in sentiment analysis has recently become active because massive volumes of sentimental data on the web are available (Liu, 2012). The research can be largely divided into two categories (Ravi & Ravi, 2015). First, lexicon-based approaches depend on a sentiment lexicon, which represents positive or negative sentiment words that are precompiled and predetermined (Ding, Liu, & Yu, 2008; Hu & Liu, 2004; Turney, 2002). This approach is usually based on dictionaries or corpuses. Second, a machine learning approach uses machine learning algorithms and text mining with syntactic or linguistic features (Meena & Prabhakar, 2007; Pang, Lee, & Vaithyanathan, 2002), for example, part of speech, sentiment words, and negation, in supervised and/or unsupervised ways. As an extension, a hybrid approach combines the lexicon-based and machine learning approaches. The majority of methods in the hybrid approach use the features and sentiment lexicons extracted from both approaches as a key factor. Most of the current methods have focused on explicit opinions because explicit ones are easier to classify than implicit ones, and implicit opinions have received little attention (Liu, 2012). In particular, major obstacles to the methods in attacking implicit opinions can be summarized as follows. First, a sentence that does not contain sentiment words can also express any sentiment. For example, the sentence "This

* Corresponding author.
E-mail addresses: vkeh127@nate.com (M. Kang), miso1004me@gmail.com (J. Ahn), skylee@hanyang.ac.kr, skylee1020@gmail.com (K. Lee).

http://dx.doi.org/10.1016/j.eswa.2017.07.019
0957-4174/© 2017 Elsevier Ltd. All rights reserved.

smart phone battery frequently requires charging." implies a negative opinion of the battery without using a negative word. Second, a sentence containing sentiment words may not express any sentiment: for example, "I always hope to the movie that I will watch is funny and dramatic" Third, ironic or sarcastic sentences are hard to analyze: for example, "Thanks to the acting of the leading actress, I can fall to sleep." Finally, some sentences (such as interrogative, conditional, and comparative) are hard to analyze: for example, "Do you think this movie is funny?" "Oh, if I could, I would choose another movies." The two sentences imply a negative opinion, but the presence or absence of sentiment words is not useful for determining the sentiment.

In consideration of the above challenges, when many different ways to express various opinions are possible, one of the critical ways for people to grasp intuition is to understand the use of word order. In view of this, we think that sentiment analysis will improve if word orders and syntactic relations between words are incorporated. Thus, this paper focuses on sequences of words to solve the challenges in previous sentiment analyses. Our proposed method does not need the extraction process of sentiment lexicons, so that the pre-process of extracting opinion words and determining the sentiment from them may be avoided. The contribution of this paper is to propose a new sentiment analysis method that uses word orders without the need of sentiment lexicons. For that purpose we propose an ensemble of text-based hidden Markov models using boosting and clusters of words produced by latent semantic analysis. Also, we take advantage of the ample availability and easy preparation of labeled texts. We notice the availability of such training data enables the method to learn diverse patterns of sentiment sentences. In reality, the collection task, which is relatively easy, can be streamlined because there are diverse channels and practical ways, such as data crawling, to systemically collect texts and reviews on the web.

This paper is organized as follows: Section 2 reviews the previous works on sentiment analysis and shows the difference between previous works and the proposed method. Section 3 presents the proposed method, denoted by Ensemble-TextHMM, which uses sequences of words in a sentence to learn the patterns for understanding the sentiment of a sentence. Section 4 reports and discusses the experiment results on movie reviews. Finally, Section 5 concludes the paper and outlines future research directions.

2. Previous works

This section provides a brief description of sentiment analysis and the difference between previous works and the proposed method. Sentiment analysis, also known as opinion mining, mainly focuses on classifying each review (text) as positive or negative. In sentiment analysis, three main classification levels exist: document, sentence, and aspect levels. Document-level sentiment analysis aims to classify the opinion of a document as expressing a positive or negative sentiment, and considers the whole document as a basic information unit. Sentence-level sentiment analysis aims to identify the sentiment expressed in each sentence. The first step is to identify whether the sentence is subjective or objective. If the sentence is subjective, sentence-level analysis will determine whether the sentence expresses a positive or negative opinion. There is no fundamental difference between the document and the sentence level classifications because sentences are just short documents (Ravi & Ravi, 2015). On the other hand, aspect level analysis provides a more detailed analysis of the aspects, for example, whether or not a part is opinion-oriented, and then it classifies the input document as positive or negative (Liu, 2012). While some differences are found among these approaches, a common method for classifying sentiment is the selection of features in texts. In this process, one of the most critical ingredients is the construction of a sentiment lexicon to represent, for example, 'good' and 'bad' sentiment.

In fact, a number of researchers in sentiment analysis have developed feature selection methods. In their early method, Hu and Liu (2004) manually set the initial adjectives and then expanded the set using synonym and antonym relations in WordNet and predicted the orientations of texts. Turney (2002) extracted two-word phrases containing adverbs or adjectives as features in which the phrases are selected by some pre-defined parts of speech (POS) patterns. He used pointwise mutual information (PMI) to calculate the semantic orientation of the extracted phrases using search results. In particular, they issued queries of positive and negative reference words (for example, "excellent" and "poor") to a search engine and captured the associated the phrases that were within a certain boundary in the results for the reference words. The adjective and adverb used in the POS patterns were found to be good predictors, and Meena and Prabhakar (2007) also found that nouns, verbs, and conjunctions helped in classifying subjective sentences.

Pang et al. (2002) first applied some machine learning approaches, such as naive Bayes, maximum entropy, and support vector machines, with unigram (bag-of-words) as features for binary sentiment classification of movie reviews. In the subsequent research, other features, specifically the frequency of terms, POS, sentiment words, syntactic dependency, rules of patterns and negation, were applied. The dependency tree-based classification method was proposed to consider word orders and syntactic relations between words (Matsumoto, Takamura, & Okumura, 2005; Meena & Prabhakar, 2007; Nakagawa, Inui, & Kurohashi, 2010). Matsumoto et al. (2005) extracted frequent-word subsequences and dependency subtrees from sentences and then pruned them to create subtrees for classification. Meena and Prabhakar (2007) used POS-tagging and dependency trees and analyzed the effects of conjunctions. Nakagawa et al. (2010) proposed a dependency tree-based method using conditional random fields with hidden states that are unobservable in the training data. They considered situations in which the sentiment could be reversed and showed that the method performs better than other methods based on a bag-of-features method.

To effectively include word orders in sentiment analysis, several researchers also used HMMs. Duric and Song (2012) and Jin, Ho, and Srihari (2009) used HMMs to select sentiment lexicon and to reflect content and syntax models. Duric and Song (2012) suggested hidden Markov model-latent Dirichlet allocation (HMM-LDA) to get syntactic classes of features. They showed that the method outperformed the feature selection method based on POS. Jin et al. (2009) proposed a novel machine learning framework using lexicalized HMMs. They integrated linguistic features such as POS and lexical patterns into HMMs and used a basic tag set and a pattern tag set. Their method is closest to our method, but substantial differences are found in using HMMs: our method does do not need the extra process of explicitly tagging words, as opposed to their method. Instead, we estimate the similarity between the pattern of input text and that of sentences expressing either a positive or negative sentiment. In addition, we adopted an ensemble boosting method to enhance the learning of diverse patterns in sentiment.

Some research has applied ensemble methods for sentiment classification. Vinodhini and Chandrasekaran (2014) tried several combinations of unigrams, bi-grams and tri-grams as features in the use of support vector machines. Xia, Zong, and Li (2011) suggested an ensemble of feature sets and classification algorithms, naive Bayes, maximum entropy and support vector machines. The results of their methods showed improvements in accuracy compared to other methods using only a single feature. In view of the benefit of the ensemble approach, to boost the prediction accuracy,

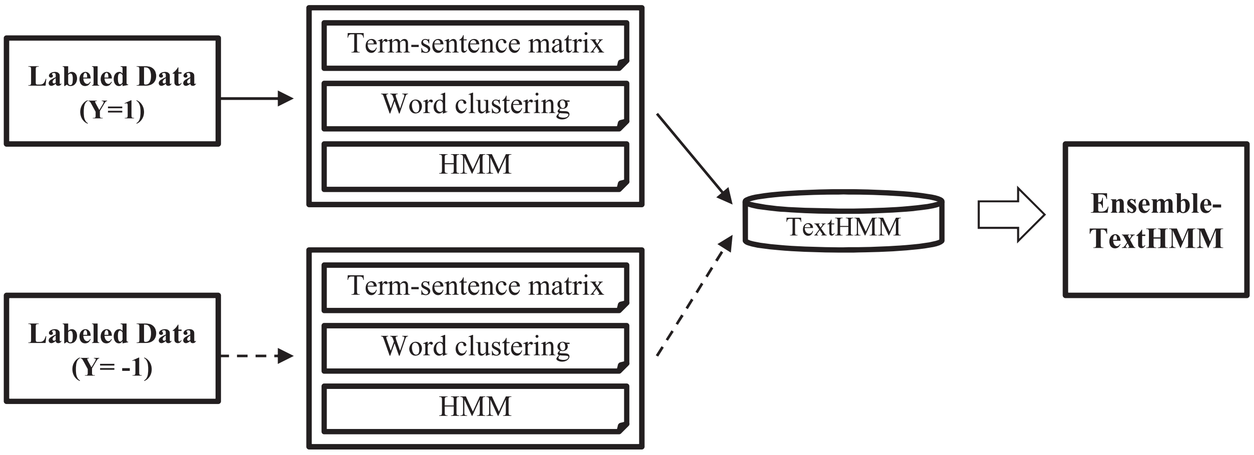

Fig. 1. Overview of the proposed method, an ensemble of HMMs by text clustering (Ensemble-TextHMM).

we apply an ensemble idea in the aggregation of several HMM classifiers that vary the training data and HMM parameters.

In light of the existing methods in sentiment analysis, the contribution of this research is summarized in the following two ways. First, we eliminate the extraction of features and sentiment lexicon, using frequency and sequence of terms. Second, the sentiment orientation is estimated by pattern probability based on an ensemble of hidden Markov models, of which hidden states are assigned by text clustering to consider semantic POS.

3. The proposed method

We build an ensemble of text-based hidden Markov models (HMMs) by clustering texts to classify the sentiment of an input text, denoted by Ensemble-TextHMM. Fig. 1 shows an overview of the proposed method. By using the labeled input data and clustering the words in the texts, we build text-based hidden Markov models (TextHMM) and aggregate such models to obtain a final classifier.

We construct a supervised learning method to classify sentences by training labeled sentence data. We start with input sentences, $s_1, s_2, \ldots, s_{m_1}, s_{m_1+1}, \ldots, s_{m_1+m_2}$, where s_i, $1 \le i \le m_1$, is a sentence that belongs to the attribute label $Y = 1$ in $\text{Dataset}_{Y=1}$ and s_i, $m_1 + 1 \le i \le m_1 + m_2$, to the label $Y = -1$ in $\text{Dataset}_{Y=-1}$. Each sentence s_i is represented as a vector of words: $s_i = (o_{i,1}, o_{i,2}, \ldots, o_{i,L_i})$, where L_i is the length of s_i and $o_{i,j}$, $1 \le j \le L_i$, is the j-th word in s_i. For example, a plain sentence, “this movie is very fantastic”, has $o_{i,1} = \textit{this}$, $o_{i,2} = \textit{movie}$, and so forth, with $L = 5$. The attribute of label Y depends on the application: for example, the training dataset can consist of positive $(Y = 1)$ and negative $(Y = -1)$ sentences or positive $(Y = 1)$ and non-positive $(Y = -1)$ sentences. We calculate the positive orientation of sentence s, defined as $f_{Y=1}(s)$, by a model built from the positively labeled data. We know that unobserved, underlying and usually low-dimensional states, in association with observed words exist, so we tag hidden states for words in the sentence s. Then, we multiply the transition probability of the hidden states between connected words in the sentence s and the observation probability, which represents the frequency of observed words from the hidden state. To obtain the normalized probability, we divide the multiplied probability by the length of the sentence, L, because it decreases as the sentence gets longer without normalization. Similarly, we calculate the negative orientation $f_{Y=-1}(s)$ from the negatively labeled data. After computing the orientation of s for each label, we compare the positive orientation with the negative orientation. If the positive orientation is bigger than the negative orientation, we consider the sentence s to expresses a positive opinion, otherwise it is considered to be negative:

$$\hat{Y} = \arg\max\{f_{Y=1}(s), f_{Y=-1}(s)\}, \tag{1}$$

We denote the above process as TextHMM. Next, we repeat the above process with several different TextHMMs and form an ensemble of TextHMMs to reflect multiple and diverse patterns in the training data. To form several TextHMMs, we extract bootstrap-based subsets of the training data and construct TextHMMs from each bootstrap subset of the training data. In short, the proposed method consists of the following three main steps: (1) determining the hidden states of words through word clustering based on latent semantic analysis; (2) constructing hidden Markov models; and (3) forming an ensemble of TextHMMs by boosting. Now we describe each step in detail.

3.1. Clustering words in texts

To build a hidden Markov model (HMM), we first determine the hidden states based on a latent semantic analysis that uses the co-occurrence of words in sentence (Bellegarda, Butzberger, Chow, Coccaro, & Naik, 1996). Basically, we regard a cluster of words as a hidden state. The reason we assign hidden states to words is that the number of words in a text is vast and we view the words sharing similar roles in sentences as being apt for grouping. If we do not group words but instead treat each word as an individual cluster, the calculated transition probability from one word to another word from the training data will be so small that the observation probability of a sentence pattern will be quite close to zero and practically useless. To avoid this problem, we cluster words through latent semantic analysis and then use the transition probability between clusters instead of words. If some words tend to co-occur with certain words and play a similar role, they will be in the cluster. For example, the following words meaning “tomorrow” need to be clustered because they play the same role in a sentence: “2moro”, “TMR”, “2m”, “2mro”, “twomo”. Especially, many people use characterful words on the internet, not following exact grammar. Thus, to deal with these words as well, we applied the clustering of words beforehand.

In the first step, we follow the three substeps. First, we construct the term-sentence matrix following the vector space model (Deerwester, Dumais, Landauer, Furnas, & Harshman, 1990) X of co-occurrences, where two words are said to co-occur in the same sentence if they appear in the same column (sentence) of the matrix, Here, X is an $n \times m$ matrix, where n is the number of words and m is the number of sentences in the training data. Second, we perform the singular value decomposition (SVD) of X to capture

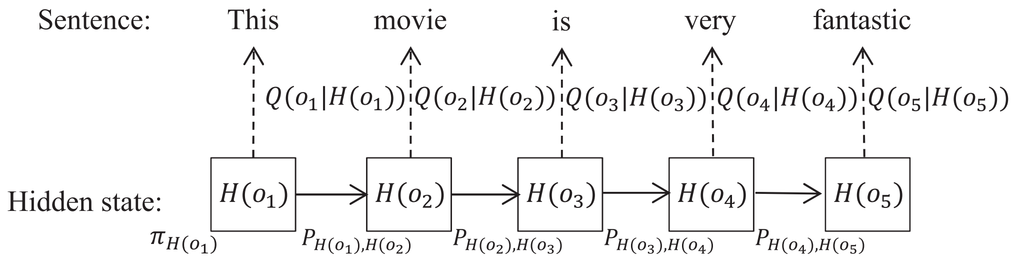

Fig. 2. Example of the used text-based hidden Markov model is shown.

the inherent dimensionality of the matrix. Notice that the matrix is generally large and sparse. The term-sentence matrix X is approximated by the product of three matrices:

$$X \cong \hat{X} = USV^T,$$

where U $(n \times d)$ is the matrix of left singular vectors, S $(d \times d)$ is the diagonal matrix of singular values (the square roots of the non-zero eigenvalues of X^TX in descending order), and V^T $(d \times m)$ is the matrix of right singular vectors. In this paper, instead of the full rank of X, we use the matrix S with rank d associated with the first d largest singular values so that the matrix (X) can represent the important and reliable patterns underlying the data X and ignore noisy and high-order effects (Deerwester et al., 1990). The reduced dimensionality d is a parameter that can vary by application: it can be set to a fixed number or it can be set so that a certain portion of the total variability in X may be captured. In this research, we chose a value of d that yields good performance by cross-validation which is an operational criterion. In particular, we searched for the best d among the possible d values for which the total variability in X is greater than 0.3. Third, words are clustered by the k-means clustering method using cosine similarity. The extent to which words have a similar occurrence pattern across the training dataset can be drawn from

$$\hat{X}^T\hat{X} = US^2\, U^T = US(US)^T,$$

the symmetric matrix that contains all of the term-to-term dot products. That is, the dot product between two rows of US represents the similarity between the two terms. Thus, we cluster the words in the training dataset by the k-means clustering with cosine similarity, Sim, between two rows u_i and u_j of US:

$$Sim(u_i, u_j) = \frac{u_i S^2 u_j^T}{||u_i S||\ ||u_j S||}, \quad 1 \le i, j \le n.$$

Hereafter, we set k in the k-means clustering as h, which is the number of the clusters and varies from 50 to 250 in the final ensemble algorithm. Expectedly, the words in the same cluster play a similar role in the sentences. We define the cluster of word o as $H(o) \in \{1, \cdots, h\}$, which will be used as the hidden state of the word o.

3.2. Constructing hidden Markov models

A hidden Markov model (HMM) consists of 3 parameters: a state transition matrix, an observation matrix, and an initial state matrix. We define a HMM, the parameter set Θ, by the following:

$$\Theta = (P, Q, \pi),$$

where P $(h \times h)$ is the state transition matrix with entries p_{ij} representing the transition probability from hidden state i to hidden state j. Recall that h is the number of clusters from the previous clustering step. We define the hidden state i for word o by the cluster from the previous step: $H(o) = i, 1 \le i \le h$. We define the transition probability P_{ij} by

$$P_{i,j} = P(\mathbf{H}_{t+1} = j|\mathbf{H}_t = i),$$

$$\sum_{j=1}^{h} P_{i,j} = 1,$$

where $\mathbf{H}_t$ is a random variable for the hidden state of the t-th word in sentence. That is to say, we use the clusters i and j as the observed hidden states of o_t and o_{t+1}, respectively. Let $Q_i(w_k)$ be the probability that the k-th word appears from hidden state i:

$$Q(w_k|i) = P(\mathbf{S}_\ell = w_k|\mathbf{H}_t = i), \quad 1 \le k \le n,$$

$$\sum_{k=1}^{n} Q(w_k|i) = 1,$$

where w_k is the observed k-th word at the term document matrix X and $\mathbf{S}_\ell$ is a random variable for the ℓ-th word in sentence. The initial state matrix, π $(h \times 1)$, is set by the frequency of the first hidden states in sentence. Unlike general HMMs, the hidden state of the word o is fixed through the cluster of word o, $H(o)$, in the previous step. This leads to the easy estimation of the parameters by counting the transitions between hidden states and the frequency of words in a cluster. Given a sentence, the sentiment orientation (SO) of the sentence s is calculated as follows:

$$P(s|\Theta) = \pi_{H(o_1)} Q(o_1|H(o_1)) P_{H(o_1),H(o_2)} Q(o_2|H(o_2)) \cdots P_{H(o_{L-1}),H(o_L)} \times Q(o_L|H(o_L)),$$

where Θ is the TextHMM constructed by the training texts of one attribute (either positive or negative, for example). Fig. 2 shows an illustrative flow for calculating the SO of a sentence.

In this paper, a classifier (TextHMM) is made up two HMMs, $\Theta_{Y=1}$ and $\Theta_{Y=-1}$, via clustering the input texts. One HMM, $\Theta_{Y=1}$, is formed from the data of one attribute (positive), and the other, $\Theta_{Y=-1}$, is made from the data of the other attribute (negative). We compute the orientation of sentence s by the two HMMs and compare them. If the positive orientation, $P(s|\Theta_{Y=1})$, is bigger than the negative orientation, $P(s|\Theta_{Y=-1})$, the sentence is deemed to expresses a positive opinion, otherwise it is negative:

$$\hat{Y} = \arg\max\{P(s|\Theta_{Y=1}), P(s|\Theta_{Y=-1})\} \qquad (2)$$

3.3. Building ensemble-TextHMM using boosting

In this step, we use an ensemble of TextHMMs to reflect the diverse pattern of sentences. One TextHMM may separate a certain pattern of training data well while it degenerates in sentences in which the pattern is relatively less expressed. There are various ways to express an opinion expression depending on region, age, gender, and so forth. Furthermore, users on the web often neglect grammar, using abbreviations and buzzwords. In addition, if we model one TextHMM with all of data using a parameter setting, the classifier might rely on a few dominant expression patterns. Thus, the proposed method in this paper extracts samples of the training data by bootstrapping and forms several TextHMMs from each bootstrap subset. These TextHMMs are combined by boosting.

The main idea behind the ensemble methodology is to weight several individual classifiers and then combining them in order to

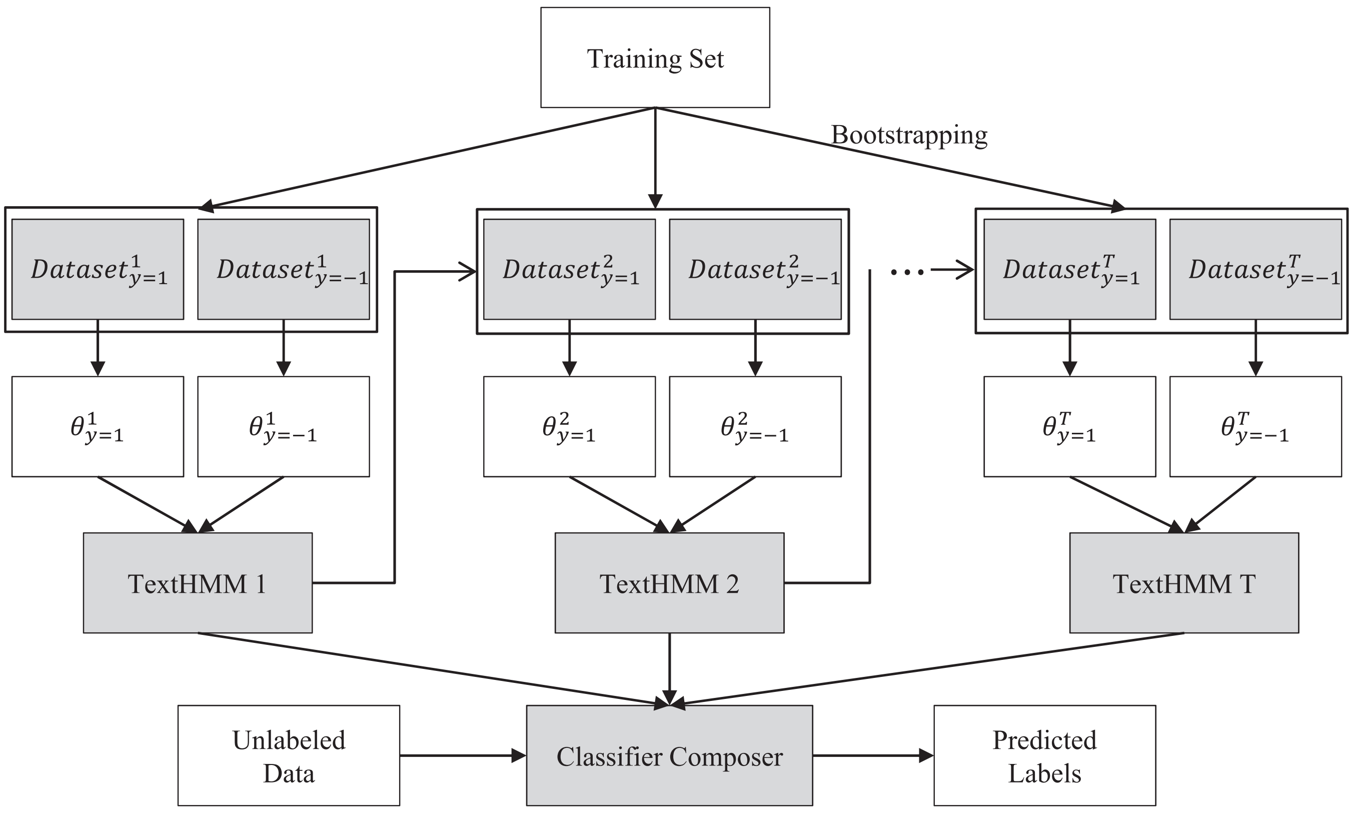

Fig. 3. Framework of the proposed Ensemble-TextHMM algorithm is shown.

obtain a final classifier that outperforms the individual classifiers. The more different and diverse classifier is, the stronger the ensemble becomes (Rokach, 2010). Hence we construct several TextHMMs by not only taking bootstrap samples but also differing the number of hidden states, h; at each iteration we randomly chose h between 50 and 250.

The way to construct an ensemble used in this paper is similar to Freund and Schapire's AdaBoost.M1 algorithm that deals with binary classification (Freund, Schapire, & Abe, 1999). The main idea of AdaBoost is to concentrate misclassified patterns by using the weights assigned to all of the data in the training set. The same weight is assigned to all data at first. In each iteration, the weights of the misclassified data increase, but the weights of correctly classified data decrease so that the next classifier may focus on the misclassified data. The weight proportion of the correctly classified data is allocated to each classifier. However, it differs from AdaBoost in that during each iteration, the proposed algorithm combines two TextHMMs built based on the bootstrap subset and internally tests all observations in the training data. The final classifier is weighted as the sum of each classifier in a way that higher weights are given to classifiers with higher accuracy. The framework of the proposed method, Ensemble-TextHMM, is shown in Fig. 3, also described in detail in the following algorithm.

Algorithm: Ensemble-TextHMM

```
Given $(s_1, y_1), (s_2, y_2), \ldots, (s_m, y_m)$ where $s_i \in S$, $y_i \in \{-1, 1\}$, and $T$ (the number of iterations)

1. $t \leftarrow 1$
2. $D_1(i) \leftarrow 1/m$, $i = 1, \ldots, m$.
3. Repeat
   (a) Extract $Dataset^t_{y=1}$ and $Dataset^t_{y=-1}$ from $S$ using bootstrapping
   (b) Train TextHMM$^{(t)}$, $\Theta^{(t)}_{Y=1}$ and $\Theta^{(t)}_{Y=-1}$, using $Dataset^t_{y=1}$ and $Dataset^t_{y=-1}$, respectively, as in Section 3.2
   (c) Estimate the sentiment orientation of $s_i \in S$: $\hat{y}^{(t)}_i = \arg\max\{P(s_i|\Theta^{(t)}_{Y=1}), P(s_i|\Theta^{(t)}_{Y=-1})\}$
   (d) Calculate error rate $\epsilon_t \leftarrow \sum^m_{i:y_i \neq \hat{y}^{(t)}_i} D_t(i)$
   (e) If $\epsilon_t > 0.5$, then go to step 3-a
   (f) $\alpha_t \leftarrow \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}$
   (g) $D_{t+1}(i) = D_t(i) \exp\{-\alpha_t y_i \hat{y}^{(t)}_i\}$, $i = 1, \ldots, m$
   (h) $D_{t+1}(i) \leftarrow \frac{D_{t+1}(i)}{\sum^m_k D_{t+1}(k)}$, $i = 1, \ldots, m$
   (i) $t \leftarrow t + 1$
4. until $t > T$
5. Estimate the sentiment orientation of $s$: $\hat{y} = \arg\max\{\sum^T_{t=1} \alpha_t P(s|\Theta^{(t)}_{Y=1}), \sum^T_{t=1} \alpha_t P(s|\Theta^{(t)}_{Y=-1})\}$
```

4. Experiments

In this section, we discuss the performance of the proposed method, Ensemble-TextHMM in comparison with a few of the previously proposed methods. We conducted experiments using five benchmark datasets, MR, CR, SST2, Subj, and Pros&Cons. Each dataset consists positive sentences and negative ones. Brief descriptions of the datasets are given as follows:

- **MR** : Movie reviews with positive and negative labels (Pang & Lee, 2005).
- **CR** : Customer reviews of electronic products (DVD player, Camera, MP3 etc.) (Hu & Liu, 2004).
- **SST2** : Stanford Sentiment Treebank in which neutral reviews removed (Socher et al., 2003).
- **Subj** : Subjectivity data with subjective and objective labels (treated as positive and negative) (Pang & Lee, 2004)

Table 1
Accuracy of the sentiment classification on the five datasets is shown.

Method	MR	CR	SST2	Subj	Pros&Cons
Ensemble-HMM	**82.39**	**87.0**	86.1	**98.1**	**92.5**
CNN-non-static	81.5	85.0	**88.1**	93.2*	–
CNN-rand	76.1*	79.8*	82.7	89.6*	–
CNN-Multi	81.1	85.0	**88.1**	93.2*	–
MV-RNN	79.0*	–	82.9	–	–
RAE	77.7*	–	82.4	–	–
NBSVM	79.4*	81.8*	–	93.2*	–
MNB	79.0*	80.0*	–.	93.6*	–
Tree-CRF	77.3*	81.4*	–	–	–
Opinion observer	–	–	–	–	90.2

Table 2
Some exemplary movie reviews in which Ensemble-TextHMM outperformed the previous methods are shown.

Sentiment	Review sentence
Negative	Director Hoffman, his writer and Klines agent should serve detention.
Negative	Resembles a soft porn Brian De Palma pastiche.
Negative	Seems like someone going through the motions.
Positive	A bodice-ripper for intellectuals.
Positive	Norton holds the film together.
Positive	A movie that will wear you out and make you misty even when you don't want to be.

- **Pros&Cons** : Opinions of competing products on the web with positive and negative labels (Liu, Hu, & Cheng, 2005)

We used 10-fold cross validation with each dataset except SST2. The SST2 dataset consists of three datasets: training, *dev*, and testing. In SST2, we trained the model with the training data and then used the *dev* dataset for selecting the number of classifiers for best precision. Common words were removed, and neither stemming nor the correction of misspelled words were used.

We compared our method with the performance of the state-of-the-art methods on the datasets described above. Nakagawa et al. (2010) proposed a dependency tree-based method using conditional random fields (CRFs) with hidden variables that cannot be observed in labeled data. The orientation (polarity) value of the whole sentence is calculated considering the interactions between the hidden variables. A matrix vector recursive neural network (MV-RNN) model (Socher, Huval, Manning, & Ng, 2012) learns the meaning of operators in propositional logic and natural language. In addition, convolutional neural networks using pre-trained vectors from word2vec for each task (CNN-non-Static), random initialization (CNN-rand), and a multichannel architecture (CNN-multi) (Kim, 2014); recursive autoencoders with pre-trained word vectors (RAE) (Socher, Pennington, Huang, Ng, Manning, 2011); naive Bayes support vector machines with uni-bigrams (NBSVM); and multinomial Naive Bayes with uni-bigrams (MNB) were also included in the comparison. Results in Table 1 show that the accuracy of Ensemble-HMM is generally higher than that of the methods in comparison. The boldfaced numbers represent best results in the datasets, and the numbers marked with an asterisk mean significant difference between Ensemble-HMM and the compared method (p-value < 0.1 by Wilcoxon rank-sum tests and p-value < 0.01 by t-test). We observe that the proposed method significantly outperforms CNN-rand, NBSVM, and MNB in the tested datasets except SST2. In SST2, the Ensemble-HMM method ranked second in terms of accuracy, surpassing CNN-rand, MV-RNN, and RAE.

To see the performance in detail, we present several exemplary sentences in the MR dataset, as shown in Table 2, in which Ensemble-TextHMM was correctly classified. The previous methods using sentiment lexicon failed to correctly classify the above reviews, requiring more information. The MV-RNN method could not classify the first and fourth sentences (Socher et al., 2012). However we admit that Ensemble-TextHMM had difficulty in classifying some sentences that have explicit words of both sentiment opinions. For example, "really quite funny" is explicitly positive but TextHMMs fail to classify it correctly. The reason for this is that the patterns of those sentences easily appear in both positive and negative datasets. The words "really" and "quite" are more often used in the negative dataset, although "funny" is more prevalent in the positive dataset. This is a weak point of TextHMMs since it does not use explicit sentiment words.

Fig. 4 shows the accuracy of the final classifier from the proposed method according to the number of classifiers, i.e., iterations in the MR dataset. One classifier comprised of a pair of TextHMMs (at iteration 1) is lower in accuracy, at about 65% for the training dataset and 57% for the test, when it just classifies a few typical patterns, but has difficulty in distinguishing diverse patterns. For this reason, we need an ensemble of TextHMMs to reflect the diversity of patterns. We trained TextHMMs by varying the ratio of data points, the number of chosen singular values in SVD, and the number of hidden states. As a result, we discovered a larger number of classifiers, the more accurate the final classifier gets. Fig. 5 shows the training and testing accuracy of the proposedm method in the datasets, Subj, CR, and SST2 with another round of MR.

We verified the effects of the number of hidden states and sampling on accuracy. First, we constructed TextHMMs by only varying the number of hidden states in the clustering process of Ensemble-TextHMM. As shown in Fig. 6(a), as the number of classifiers increases, the accuracy of the final classifier increases, but not significantly. The interpretation of such marginal improvement is that the variation of the number of hidden states is insufficient to catch diverse patterns in the training data. Next, we attempted to make a variety of TextHMMs by only sampling the training data. The result shown in Fig. 7 supports the point that the ensemble model built by sampled data is quite different from that built by the number of hidden states (in Fig. 6(b)), and they are able to reflect diverse patterns. The interpretation is that the model at each iteration learned and accumulated different patterns of opinions that are not easily revealed in the whole set of training data. The model learned from all of the training data tends to reflect a universal pattern. We found that the result of sampling was better than that for differing the number of hidden states, in that the accuracy increased faster and to a higher value. After all, the proposed method combined the two effects and showed an excellent capability to learn diverse patterns in the training data, as shown in Fig. 4.

5. Applications to real-life datasets

This section shows the application of Ensemble-TextHMM to two real-life datasets of cloth reviews and paper titles in Google Scholar. The cloth reviews written in Korean were collected from the online shopping site www.11st.co.kr. We crawled 1000 positive and 1000 negative reviews about women's clothes. Before testing, only special letters were removed and neither the correction of the spacing nor the typos was used. The 3-fold cross validation was used for evaluation. As shown in Table 3, Ensemble-TextHMM per-

Table 3
Results of the classification for 3-class titles of papers; One vs Rest approach determines type of title by using threshold, multiclass approach determines title on type of data having max SO value.

	One vs rest (%)	Multiclass (%)
Computer	96.8	92.4
Business	97.3	93.2
Education	95.8	90.2
Average	96.6	91.9

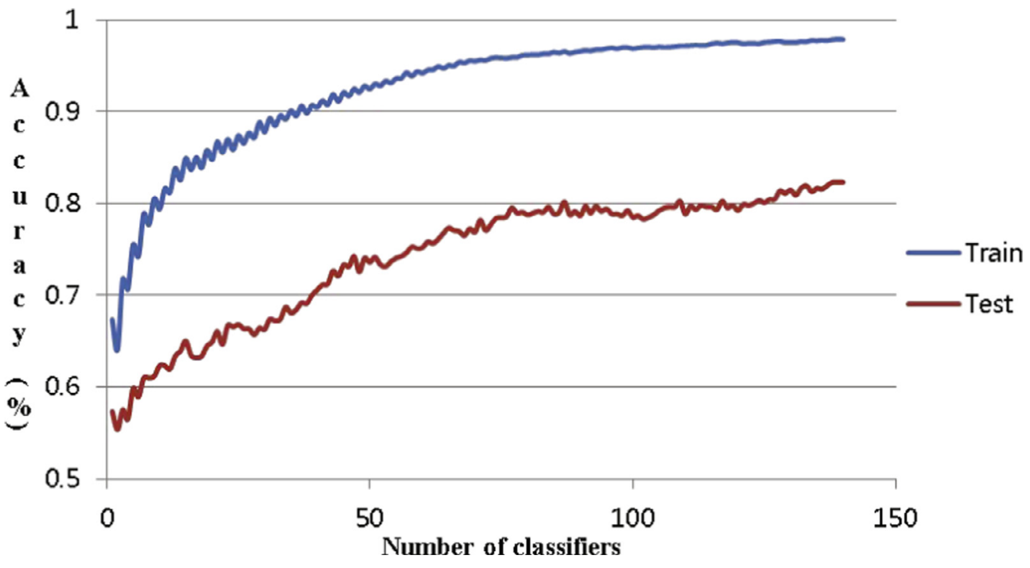

Fig. 4. Accuracy of Ensemble-TextHMM according to the number of iterations (the number of TextHMMs) for the movie review data.

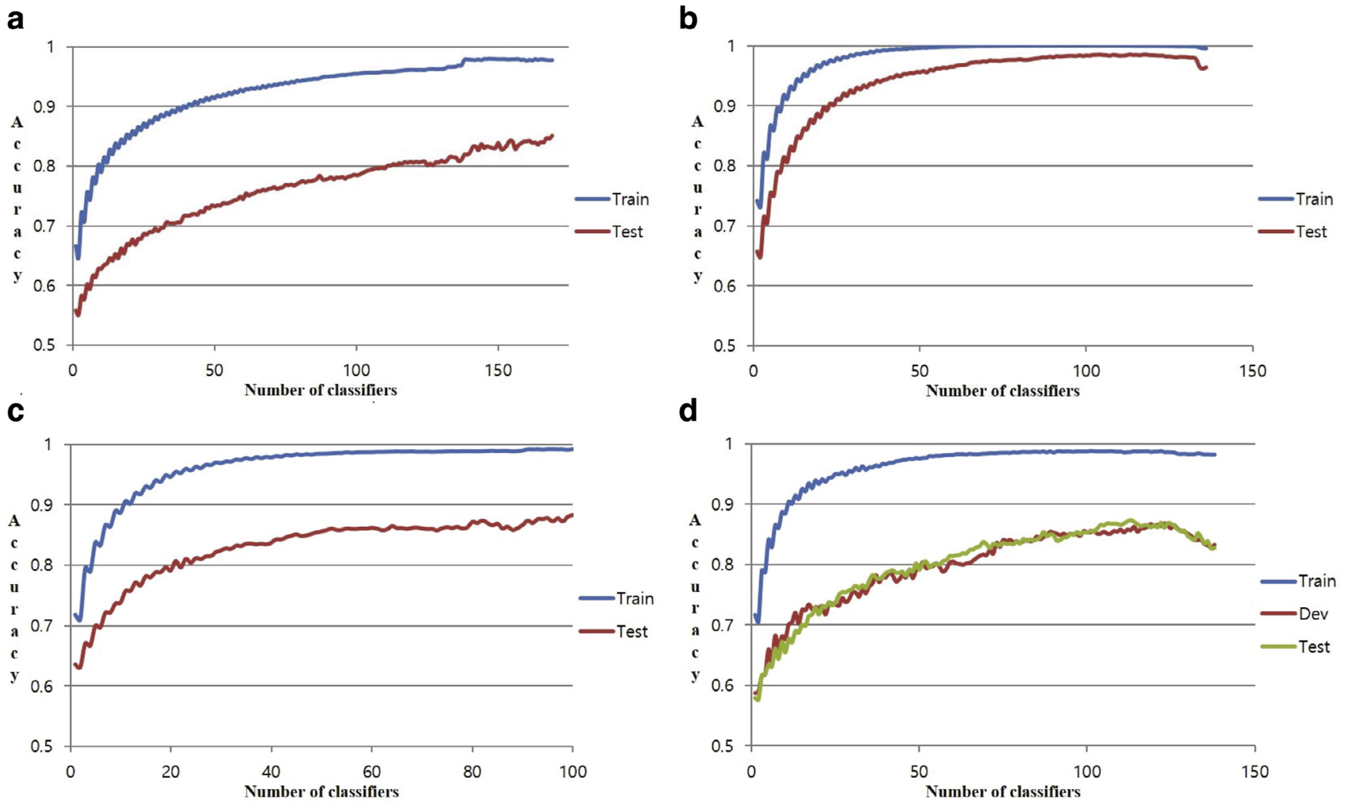

Fig. 5. Accuracy of Ensemble-TextHMM according to the number of iterations (the number of TextHMMs) for benchmark datasets, (a) MR, (b) Subj, (c) CR and (d) SST2 is shown. Accuracy is measured by the average of 10-fold cross validation results from train/test datasets. In SST2, consisting training/*dev*/testing, we used *dev* for selecting the number of classifiers.

formed quite well with the data set in classifying the labels of the reviews. For example, at iteration 10 the accuracy reached 85.3% for the testing as shown in Fig. 8. The method needed no predefined sentiment words other than the training data with labels.

The second application relates to multi-label text classification with paper titles in Google Scholar, which consists of 360 titles in computer science, business, and education, respectively. When we classify papers as one label (for example, computer science), we used the labeling strategy of the one versus the rest. The 3-fold cross validation was also used for evaluation. Fig. 9 shows the calculated SO values for the training of 240 sentences with labels from computer science versus the others, which clearly separates the two labels. For the sake of multi-label classification, we constructed three Ensemble-TextHMMs for the labels and determined the type of sentences by the model of the maximum SO value. We summarized the classification accuracy in Table 3, in which the accuracy is more than 90%. We note that the performance in

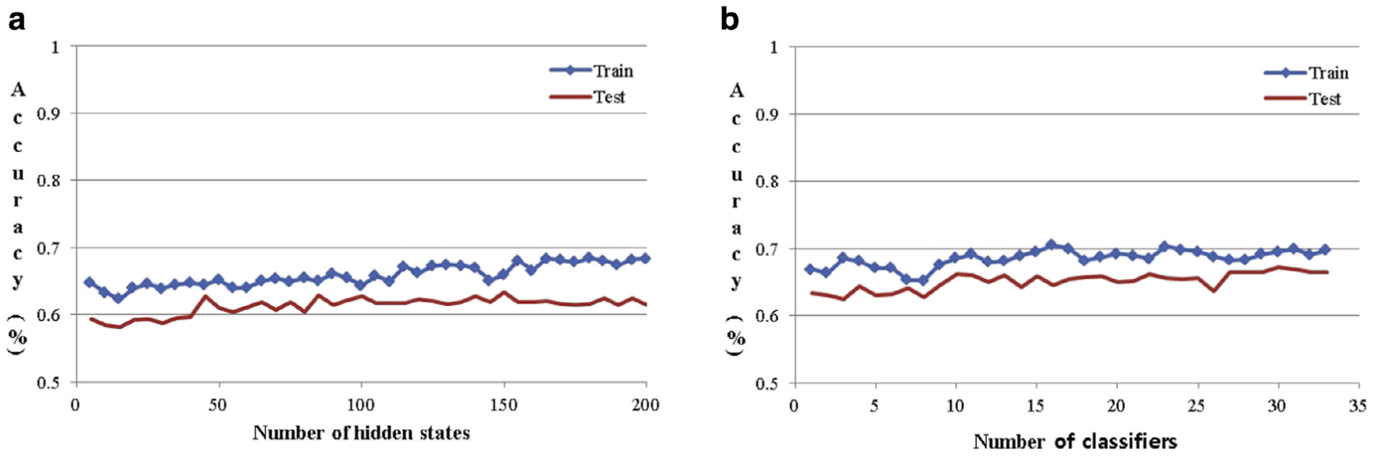

Fig. 6. The effect of the number of hidden states on accuracy are shown (a) when it is added sequentially and (b) when it is randomly chosen and added while building an ensemble of TextHMMs.

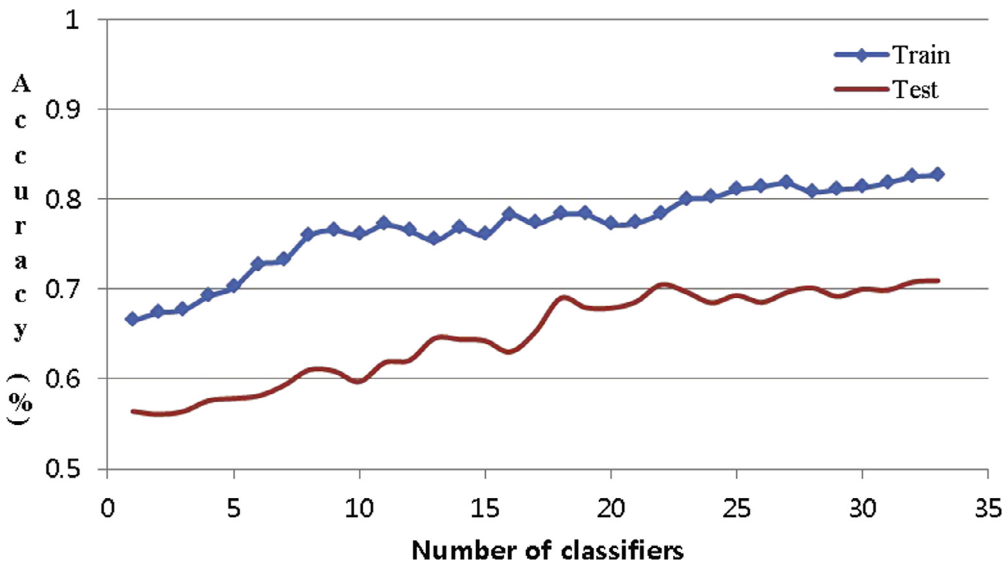

Fig. 7. The effect of sampling on accuracy in building an ensemble of TextHMMs is shown.

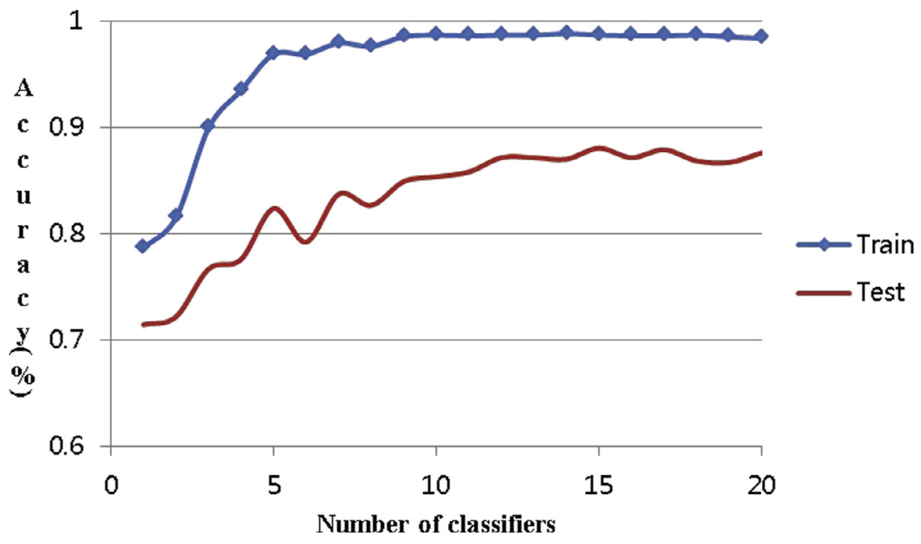

Fig. 8. Performance of the proposed method for the cloth reviews is shown.

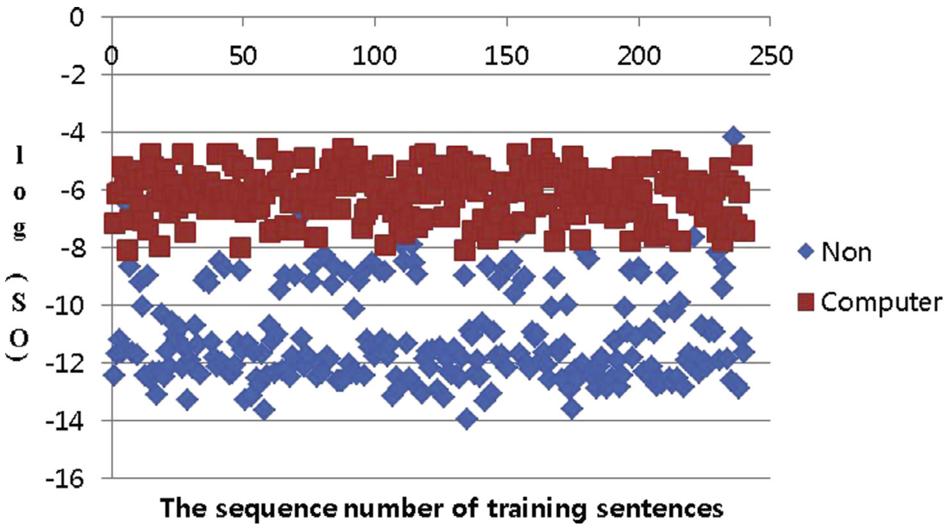

Fig. 9. Sentence orientation (SO) values with threshold for 240 papers with computer science versus non-computer science.

the binary classification setting is better than that in the multi-classification.

6. Conclusion

This paper proposed a new sentiment analysis method using an ensemble of text-based hidden Markov models, the Ensemble-TextHMM method. Instead of relying on extracted sentiment lexicons or predefined keywords, it uses labeled training texts to reflect diverse patterns of sentiments. For that purpose, we built a classifier using a text-based hidden Markov model through clustering the words, and we applied an ensemble of such classifiers by varying the number of hidden states and sampling the training texts. In the experiments, we showed that the proposed method, by producing a high accuracy, effectively captures diverse sentiment patterns in comparison to two other models without the ensemble strategy. We also showed that the proposed method has comparative advantages over sentences without sentiment words and outperformed some previous methods on the benchmark movie-review dataset. We demonstrated the applicability of the proposed method in two real-life datasets. As some misclassified sentences are represented by explicit and common sentiment words, we hope to improve the model by considering some obvious sentiment lexicons.

Acknowledgment

This research was supported by the grant (NRF-2015S1A5A2A03047963) funded by the Ministry of Education and National Research Foundation of Korea. This research was also supported by the National Safety Promotion Technology Development Program (201600000002094, Smart crime prevention solution development through machine learning based on Image Big Data), funded by the Ministry of Trade, Industry and Energy (MOTIE). This research was also supported by the grant (C0507566) funded by Small and Medium Business Administration (SMBA) in the Republic of Korea and Korea Association of University, Research Institute and Industry (AURI).

References

Bellegarda, J. R., Butzberger, J. W., Chow, Y. L., Coccaro, N. B., & Naik, D. (1996). A novel word clustering algorithm based on latent semantic analysis. In *Proceedings of 1996 IEEE international conference on acoustics, speech, and signal processing, ICASSP-96.: vol. 1* (pp. 172–175). IEEE.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science, 41*(6), 391–407.

Ding, X., Liu, B., & Yu, P. S. (2008). A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 international conference on web search and data mining* (pp. 231–240). ACM.

Duric, A., & Song, F. (2012). Feature selection for sentiment analysis based on content and syntax models. *Decision Support Systems, 53*(4), 704–711.

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence, 14*(771–780), 1612.

Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 168–177). ACM.

Jin, W., Ho, H. H., & Srihari, R. K. (2009). A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th annual international conference on machine learning* (pp. 465–472). Citeseer.

Kim, Y. (2014). Convolutional neural networks for sentence classification. ArXiv preprint arXiv:1408.5882.

Liu, B. (2012). Synthesis lectures on human language technologies. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers.

Liu, B., Hu, M., & Cheng, J. (2005). Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on world wide web* (pp. 342–351). ACM.

Matsumoto, S., Takamura, H., & Okumura, M. (2005). Sentiment classification using word sub-sequences and dependency sub-trees. In *Advances in knowledge discovery and data mining* (pp. 301–311). Springer.

Meena, A., & Prabhakar, T. (2007). *Sentence level sentiment analysis in the presence of conjuncts using linguistic analysis*. Springer.

Nakagawa, T., Inui, K., & Kurohashi, S. (2010). Dependency tree-based sentiment classification using CRFS with hidden variables. In *Proceedings of the 2010 annual conference of the North American chapter of the association for computational linguistics on human language technologies* (pp. 786–794).

Pang, B., & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on association for computational linguistics, ACL '04*.

Pang, B., & Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics* (pp. 115–124).

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on empirical methods in natural language processing: 10* (pp. 79–86). Association for Computational Linguistics.

Ravi, K., & Ravi, V. (2015). A survey on opinion mining and sentiment analysis: tasks, approaches and applications. *Knowledge-Based Systems, 89*, 14–46.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33*(1–2), 1–39.

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 1201–1211). Association for Computational Linguistics.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., & Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 151–161). Association for Computational Linguistics.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., et al. (2003). Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on empirical methods in natural language processing, EMNLP* (pp. 168–177).

Turney, P. D. (2002). Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics* (pp. 417–424). Association for Computational Linguistics.

Vinodhini, G., & Chandrasekaran, R. (2014). Opinion mining using principal component analysis based ensemble model for e-commerce application. *CSI Transactions on ICT, 2*(3), 169–179.

Xia, R., Zong, C., & Li, S. (2011). Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences, 181*(6), 1138–1152.