INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(3), 408-428, June 2018.

# Latent Semantic Analysis using a Dennis Coefficient for English Sentiment Classification in a Parallel System

V.N. Phu, V.T.N. Tran

**Vo Ngoc Phu***
Institute of Research and Development,
Duy Tan University-DTU
Da Nang, Vietnam
*Corresponding author: vongocphu03hca@gmail.com; vongocphu@dtu.edu.vn

**Vo Thi Ngoc Tran**
School of Industrial Management (SIM),
Ho Chi Minh City University of Technology - HCMUT,
Vietnam National University
Ho Chi Minh City, Vietnam
vtntran@hcmut.edu.vn

**Abstract:** We have already survey many significant approaches for many years because there are many crucial contributions of the sentiment classification which can be applied in everyday life, such as in political activities, commodity production, and commercial activities. We have proposed a novel model using a Latent Semantic Analysis (LSA) and a Dennis Coefficient (DNC) for big data sentiment classification in English. Many LSA vectors (LSAV) have successfully been reformed by using the DNC. We use the DNC and the LSAVs to classify 11,000,000 documents of our testing data set to 5,000,000 documents of our training data set in English. This novel model uses many sentiment lexicons of our basis English sentiment dictionary (bESD). We have tested the proposed model in both a sequential environment and a distributed network system. The results of the sequential system are not as good as that of the parallel environment. We have achieved 88.76% accuracy of the testing data set, and this is better than the accuracies of many previous models of the semantic analysis. Besides, we have also compared the novel model with the previous models, and the experiments and the results of our proposed model are better than that of the previous model. Many different fields can widely use the results of the novel model in many commercial applications and surveys of the sentiment classification.
**Keywords:** English sentiment classification; parallel system; Cloudera; Hadoop Map and Hadoop Reduce; Dennis Measure; Latent Semantic Analysis

## 1 Introduction

In this survey, our novel model is performed as follows: Firstly, we use the Dennis Coefficient (DNC) to identify the valences and polarities of the sentiment lexicons of the basis English sentiment dictionary (bESD). Then, the Latent Semantic Analysis (LSA) is improved by using the sentiment lexicons. All the positive documents of the training data set are transferred into one LSAV, called the positive LSAV group. All the negative documents of the training data set are transferred into one LSAV, called the negative LSAV group. Each document in the documents of the testing data set is transferred into one LSAV. We use the DNC to cluster this LSAV into either the positive LSAV group or the negative LSAV group. One similarity measure between the LSAV and the positive LSAV group is calculated certainly, called Measure_1 and one similarity measure between the LSAV and the negative LSAV group is calculated certainly, called Measure_2. If Measure_1 is greater than Measure_2, it means that the LSAV being close to the positive LSAV group is greater than the LSAV being close to the negative LSAV group, so the LSAV (corresponding to the document of the testing data set) is clustered into the positive. If Measure_1 is less than Measure_2, it means that the LSAV being close to the positive LSAV group is less than the LSAV being close to the negative LSAV group, so the LSAV (corresponding to the document of the testing data set) is clustered into the negative. If Measure_1 is as equal as Measure_2, it means that the LSAV being close to the positive LSAV group is as equal as the LSAV being close to the negative LSAV group, so the LSAV (corresponding to the document of the testing data set) is not clustered into both the positive and the negative. The LSAV is clustered into the neutral polarity. Therefore, the sentiment classification of this document is identified successfully. Finally, the sentiment classification of all the document of the testing data set is identified fully.

We firstly implement all the above things in the sequential system, and then, we perform all the above things in the parallel network environment to shorten the execution times of the proposed model.

Our proposed model has the crucial contributions to many areas of research as well as commercial applications as follows: (1) Many surveys and commercial applications can use the results of this work in a significant way; (2) The algorithms are built in the proposed model; (3) This survey can certainly be applied to other languages easily; (4) The algorithm of data mining is applicable to semantic analysis of natural language processing; (5) Millions of English documents are successfully processed for emotional analysis; (6) Our proposed model can be applied to many different parallel network environments such as a Cloudera system; (7) This study can be applied to many different distributed functions such as Hadoop Map (M) and Hadoop Reduce (R); (8) The LSA − related algorithms are proposed in this survey; (9) The DNC − related algorithms are built in this work.

## 2 Related work

We summarize many researches which are related to our research. The surveys related the similarity coefficients to calculate the valences of words are in [1, 13–18]. In the research [1], the authors generate several Norwegian sentiment lexicons by extracting sentiment information from two different types of Norwegian text corpus, namely, news corpus and discussion forums. The methodology is based on the Point wise Mutual Information (PMI), etc.

The English dictionaries are [19–21] and there are more than 55,000 English words (including English nouns, English adjectives, English verbs, etc.) from them.
There are the works related to the Dennis Coefficient (DNC) in [2, 3, 5]. The authors in [3] collected 76 binary similarity and distance measures used over the last century and reveal their correlations through the hierarchical clustering technique, etc.

There are the researches related the Latent Semantic Analysis (LSA) in [4, 6, 7]. The study in [4] presents a novel statistical method for factor analysis of binary and count data which is closely related to a technique known as Latent Semantic Analysis, etc.

The latest researches of the sentiment classification are [8–12]. In the research [9], the authors have explored different methods of improving the accuracy of sentiment classification. The authors' proposed method based on the combination of TermCounting method and Enhanced Contextual Valence Shifters method has improved the accuracy of sentiment classification, etc.

## 3 Methodology

Our methodology comprises 3 sub-sections as follows: (1) First sub-section: Creating the sentiment lexicons of the bESD; (2) Second sub-section: Improving the LSA according to the sentiment lexicons of the bESD a sequential environment and a distributed network system; (3) Third sub-section: Using the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in both a sequential environment and a parallel distributed system.

We built our the testing data set including the 11,000,000 documents in the movie field, which contains the 5,500,000 positive and 5,500,000 negative in English. We also built our the training data set including the 5,000,000 documents in the movie field, which contains the 2,500,000 positive and 2,500,000 negative in English. All the English documents in our testing data set and training data set are automatically extracted from millions of the documents of English Facebook, English websites and social networks; then we labeled positive and negative for them.

## 3.1 Creating the sentiment lexicons of the bESD

### Calculating a valence of one word (or one phrase) in English

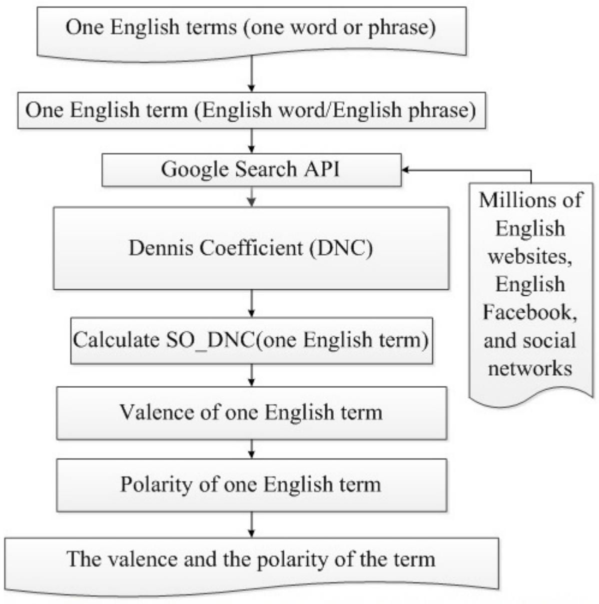
One English terms (one word or phrase)
One English term (English word/English phrase)
Google Search API
Millions of English websites, English Facebook, and social networks
Dennis Coefficient (DNC)
Calculate SO_DNC(one English term)
Valence of one English term
Polarity of one English term
The valence and the polarity of the term

Figure 1: Overview of identifying the valence and the polarity of one term in English using a Dennis coefficient (DNC)

In this part, we calculated the valence and the polarity of one English word (or phrase) by using the DNC through a Google search engine with AND operator and OR operator, as the following diagram in Figure 1 shows.
According to [1, 13–18], Pointwise Mutual Information (PMI) between two words wi and wj has the equation

$$PMI(w_i, w_j) = \log_2 \left[ \frac{P(w_i, w_j)}{P(w_i) \times P(w_j)} \right] \tag{1}$$

and SO (sentiment orientation) of word wi has the equation

$$SO(w_i) = PMI(w_i, positive) - PMI(w_i, negative) \tag{2}$$

In the research [1], the authors generate several Norwegian sentiment lexicons by extracting sentiment information from two different types of Norwegian text corpus, namely, news corpus and discussion forums. The methodology is based on the Point wise Mutual Information (PMI), etc. The authors in [13] used the Ochiai Measure through the Google search engine with AND operator and OR operator to calculate the sentiment values of the words in Vietnamese. The authors in [14] used the Consine Measure through the Google search engine with AND operator and OR operator to identify the sentiment scores of the words in English. The authors in [15] used the Sorensen Coefficient through the Google search engine with AND operator and OR operator to calculate the sentiment values of the words in English. The authors in [16] used the Jaccard Measure through the Google search engine with AND operator and OR operator to calculate the sentiment values of the words in Vietnamese. The authors in [17] used the Tanimoto Coefficient through the Google search engine with AND operator and OR operator to identify the sentiment scores of the words in English.

With the above proofs, we had the information about the measures as follows: PMI was used with AltaVista in English, Chinese, and Japanese with the Google in English; Jaccard was used with the Google in English, Chinese, and Vietnamese. The Ochiai was used with the Google in Vietnamese. The Consine and Sorensen were used with the Google in English.

According to [1,13–18], PMI, Jaccard, Consine , Ochiai, Sorensen, Tanimoto, and DNC were the similarity measures between two words, and they can perform the same functions and with the same chracteristics; so DNC was used in calculating the valence of the words. In addition, we proved that DNC can be used in identifying the valence of the English word through the Google search with the AND operator and OR operator.

With the Dennis coefficient (DNC) in [2,3,5], we had the equation of the DNC:

$$DNC(a,b) = \frac{[(a \cap b) \times [(\neg a) \cap (\neg b)] - [(\neg a) \cap b] \times [a \cap (\neg b)]]}{\sqrt{n \times [(a \cap b) + [(\neg a) \cap b]] \times [(a \cap b) + [a \cap (\neg b)]]}} \tag{3}$$

with a and b are the vectors.

In this study, we chose n=1. Therefore, we had eq. (4) as follows:

$$DNC(a,b) = \frac{[(a \cap b) \times [(\neg a) \cap (\neg b)] - [(\neg a) \cap b] \times [a \cap (\neg b)]]}{\sqrt{[(a \cap b) + [(\neg a) \cap b]] \times [(a \cap b) + [a \cap (\neg b)]]}} \tag{4}$$

From the eq. (1), (2), (3), we proposed many new equations of the DNC to calculate the valence and the polarity of the English words (or the English phrases) through the Google search engine as the following equations below. In eq. (3), when a had only one element, a is a word. When b had only one element, b is a word. In eq. (3), a was replaced by w1 and b was replaced by w2.

$$DennisMeasure(w_1, w_2) = DennisCoefficient(w_1, w_2) = DNC(w_1, w_2) = \frac{B}{\sqrt{A}} \tag{5}$$

with
(1) B=P($w_1$,$w_2$)× P(¬$w_1$,¬$w_2$)-P(¬$w1$,$w_2$) × P($w_1$,¬$w_2$);
(2) A=[P($w_1$,$w_2$)+P(¬$w_1$,$w_2$)]× [P($w_1$,$w_2$)+P($w_1$,¬$w_2$)].

Eq. (5) was similar to eq. (1). In eq. (2), eq. (1) was replaced by eq. (4). We had eq. (6) as follows:

$$Valence(w) = SO\_DNC(w) = DNC(w, positive\_query) - DNC(w, negative\_query) \tag{6}$$

In eq. (5), $w_1$ was replaced by w and $w_2$ was replaced by position_query. We had eq. (7). Eq. (7) was as follows:

$$DNC(w, positive\_query) = \frac{B7}{\sqrt{A7}} \tag{7}$$

with

(1) B7=P(w,positive_query) $\times$ P($\neg w$, $\neg positive\_query$)-P($\neg w$,positive_query) $\times$ P(w,$\neg positive\_query$);

(2) A7=[P(w,positive_query)+P($\neg w$,positive_query)] $\times$ [P(w,positive_query)+P(w,$\neg positive\_query$)].

In eq. (5), $w_1$ was replaced by w and $w_2$ was replaced by negative_query. We had eq. (8). Eq. (8) was as follows:

$$DNC(w, negative\_query) = \frac{B8}{\sqrt{A8}} \tag{8}$$

with:

(1) B8=P(w,negative_query) $\times$ P($\neg w$,$\neg negative\_query$)-P($\neg w$,negative_query) $\times$ P(w,$\neg negative\_query$);

(2) A8=[P(w,negative_query)+P($\neg w$,negative_query)] $\times$ [P(w,negative_query)+P(w,$\neg negative\_query$)].

We had the information about w, $w_1$, $w_2$, and etc. as follows:

(1) w, $w_1$, $w_2$ : were the English words (or the English phrases);

(2) P($w_1$, $w_2$): number of returned results in Google search by keyword ($w_1$ and $w_2$). We use the Google Search API to get the number of returned results in search online Google by keyword ($w_1$ and $w_2$);

(3) P($w_1$): number of returned results in Google search by keyword $w_1$. We use the Google Search API to get the number of returned results in search online Google by keyword $w_1$;

(4) P($w_2$): number of returned results in Google search by keyword $w_2$. We use the Google Search API to get the number of returned results in search online Google by keyword $w_2$;

(5) Valence(W) = SO_DNC(w): valence of English word (or English phrase) w; is SO of word (or phrase) by using the Dennis coefficient (DNC);

(6) positive_query: active or good or positive or beautiful or strong or nice or excellent or fortunate or correct or superior with the positive_query is the a group of the positive English words;

(7) negative_query: passive or bad or negative or ugly or week or nasty or poor or unfortunate or wrong or inferior with the negative_query is the a group of the negative English words;

(8) P(w, positive_query): number of returned results in Google search by keyword (positive_query and w). We used the Google Search API to get the number of returned results in search online Google by keyword (positive_query and w);

(9) P(w, negative_query): number of returned results in Google search by keyword (negative_query and w). We used the Google Search API to get the number of returned results in search online Google by keyword (negative_query and w);

(10) P(w): number of returned results in Google search by keyword w. We used the Google Search API to get the number of returned results in search online Google by keyword w;

(11) P($\neg w$,positive_query): number of returned results in Google search by keyword ($\neg w$ and positive_query). We used the Google Search API to get the number of returned results in search online Google by keyword ($\neg w$ and positive_query);

(12) P(w, $\neg positive\_query$): number of returned results in the Google search by keyword (w and ( $\neg positive\_query$)). We used the Google Search API to get the number of returned results in search online Google by keyword (w and ($\neg positive\_query$));

(13) P($\neg w$, $\neg$*positive_query*): number of returned results in the Google search by keyword ($\neg w$ and ($\neg$*positive_query*)). We used the Google Search API to get the number of returned results in search online Google by keyword (($\neg w$) and ($\neg$*positive_query*));

(14) P($\neg w$,negative_query): number of returned results in Google search by keyword ($\neg w$ and negative_query). We used the Google Search API to get the number of returned results in search online Google by keyword ($\neg w$ and negative_query);

(15) P(w, $\neg$*negative_query*): number of returned results in the Google search by keyword (w and ($\neg$*negative_query*)). We used the Google Search API to get the number of returned results in search online Google by keyword (w and ($\neg$*negative_query*));

(16) P($\neg w$,$\neg$*negative_query*): number of returned results in the Google search by keyword ($\neg w$ and ($\neg$*negative_query*)). We used the Google Search API to get the number of returned results in search online Google by keyword ($\neg w$ and ($\neg$*negative_query*)).

We have the information about the DNC as follows: (1) DNC(w, positive_query) $\geq$ 0 and DNC(w, positive_query) $\leq$ 1. (2) DNC(w, negative_query) $\geq$ 0 and DNC (w, negative_query) $\leq$ 1. (3) If DNC (w, positive_query) = 0 and DNC (w, negative_query) = 0 then SO_DNC (w) = 0. (4) If DNC (w, positive_query) =1 and DNC (w, negative_query) = 0 then SO_DNC (w) = 0. (5) If DNC (w, positive_query) = 0 and DNC (w, negative_query) = 1 then SO_DNC (w) = -1. (6) If DNC (w, positive_query) =1 and DNC (w, negative_query) = 1 then SO_DNC(w) = 0.
So, SO_DNC (w) $\geq$ -1 and SO_DNC (w) $\leq$ 1.
The polarity of the English word w is positive polarity If SO_DNC (w) < 0. The polarity of the English word w is negative polarity if SO_DNC (w) < 0. The polarity of the English word w is neutral polarity if SO_DNC (w) = 0. In addition, the semantic value of the English word w is SO_DNC (w). The result of calculating the valence w (English word) is similar to the result of calculating valence w by using AltaVista. However, AltaVista is no longer.
In summary, by using eq. (6), eq. (7), and eq. (8), we identified the valence and the polarity of one word (or one phrase) in English by using the DNC through the Google search engine with AND operator and OR operator.

**Creating a basis English sentiment dictionary (bESD) in a sequential environment**

In this part, we calculated the valence and the polarity of the English words or phrases for our bESD by using the DNC in a sequential system in the algorithm 1. According to [19–21], we had at least 55,000 English terms, including nouns, verbs, adjectives, etc.

**Algorithm 1** Performing a bESD in a sequential environment

```
1: Input: the 55,000 English terms; the Google search engine
2: Output: a bESD.
3: for all Each term in the 55,000 terms do
4:     By using eq. (5), eq. (6), eq. (7) and eq. (8) of the calculating a valence of one word
       (or one phrase) in English in the sub-section [Overview of identifying the valence and the
       polarity of one term in English using a DNC], the sentiment score and the polarity of this
       term were identified. The valence and the polarity were calculated by using the DNC through
       the Google search engine with AND operator and OR operator.
5:     Add this term into the basis English sentiment dictionary (bESD)
6: end for
7: Return bESD
```

Our basis English sentiment dictionary (bESD) had more 55,000 English words (or English phrases) and bESD was stored in Microsoft SQL Server 2008 R2.

### Creating a basis English sentiment dictionary (bESD) in a distributed system

In this part, we calculated the valence and the polarity of the English words or phrases for our bESD by using the DNC in a parallel network environment in the algorithm 2 and the algorithm 3. According to [19–21], we had at least 55,000 English terms, including nouns, verbs, adjectives, etc. This section included two phases: the Hadoop Map (M) phase and the Hadoop Reduce (R) phase. The input of the Hadoop Map phase was the 55,000 terms in English in [19–21]. The output of the Hadoop Map phase was one term which the sentiment score and the polarity are identified. The output of the Hadoop Map phase was the input of the Hadoop Reduce phase. Thus, the input of the Hadoop Reduce phase was one term which the sentiment score and the polarity are identified. The output of the Hadoop Reduce phase was the basis English sentiment dictionary (bESD).

**Algorithm 2** Performing the Hadoop Map phase

```
1: Input: the 55,000 English terms; the Google search engine.
2: Output: one term which the sentiment score and the polarity are identified.
3: for all Each term in the 55,000 terms do
4:     By using eq. (5), eq. (6), eq. (7) and eq. (8) of the calculating a valence of one word
       (or one phrase) in English in the sub-section [Overview of identifying the valence and the
       polarity of one term in English using a DNC], the sentiment score and the polarity of this
       term were identified. The valence and the polarity were calculated by using the DNC through
       the Google search engine with AND operator and OR operator.
5:     Return this term
6: end for
7: Return this term
```

**Algorithm 3** Implementing the Hadoop Reduce phase

```
1: Input: one term which the sentiment score and the polarity are identified − The output of
   M.
2: Output: a bESD.
3: Add this term into the basis English sentiment dictionary (bESD);
4: Return bESD
```

Our bESD had more 55,000 English words (or English phrases) and bESD was stored in Microsoft SQL Server 2008 R2.

## 3.2 Improving the LSA according to the sentiment lexicons of the bESD a sequential environment and a distributed network system

### Reforming the LSA based on the sentiment lexicons of the bESD

According to the Latent semantic analysis (LSA) [4,6,7], it is a technique in natural language processing, that provides a theory and method for extracting and representing the contextual-usage and meaning of words by statistical computations applied to a large corpus of text. It closely approximates many aspects of human language learning and understanding. LSA produces a set of concepts (themes) exposed by the document analysis, which are based on the terms contained in the documents. It assumes that words that are similar in meaning occur in similar pieces of texts. We have the basic step involved in LSA based on the LSA [4, 6, 7] as follows: Computing Term-Passage Matrix -This is the document-term matrix where each row represents

Table 1: One Latent semantic analysis vector - LSAV

| Documents | Latent | Semantic | Analysis | is | very | useful | slow |
|---|---|---|---|---|---|---|---|
| d1 | 1 | 1 | 1 | 1 | 2 | 1 | 0 |
| D2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

a document and the columns represent the terms (occurrence) in the document. Typically, given two documents d1and d2 with d1= "Latent Semantic Analysis is very very useful" and d2= "Latent Semantic Analysis is slow". Then, the document-term matrix is as shown below (called one Latent semantic analysis vector - LSAV) in Table 1. ⇒ The Latent semantic analysis vector:

$$LSAV = \begin{pmatrix} 1 & 1 & 1 & 1 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

As known, in one sentence, there are sometimes many terms (meaningful words or meaningful phrases) bearing the neutral sentiment, the positive sentiment, or the negative sentiment. For example, we assume that in the bESD, "useful" is the positive sentiment and its valence is +2.3. "very" is the positive sentiment and its sentiment score is +0.3. "slow" is the negative sentiment and its valence is -1.1.

We see that the neutral terms are not the important role in a sentence. Thus, if we still use them in calculating the sentiment of the sentence, there are many noises for this calculating. We also see that the negative terms make many noises for calculating the sentiment of the positive polarity and the positive terms make many noises for calculating the sentiment of the negative polarity. We use the valences of the terms combined with their frequencies to remove many noises of identifying the sentiment classification of one sentence. We apply the valences of the sentiment lexicons of the bESD into the Vd1 and Vd2 as follows: According to the bESD, it is assumed that "Latent" is 0 of its valence; "Semantic" is 0 of its sentiment value; "Analysis" is 0 of its sentiment score; "useful" of +2.3 of its valence; "is" is 0 of its sentiment score; "document" is 0 of its valence; "classification" is 0 of its sentiment value; "an" is 0 of its valence; "very" is +0.3 of its sentiment score; the sentiment value of "slow" is $-1.1$. Therefore, we have d1 and d2 in Table 2.

⇒ d1 = (0, 0, 0, 0, 0.6, 2.3, 0) and d2 = (0, 0, 0, 0, 0, 0, 0)
⇒ emphasizing on the positive terms and the negative terms in one sentence. ⇒ The Latent semantic analysis vector:

$$LSAV = \begin{pmatrix} 1 & 1 & 1 & 1 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

In one LSA vector (LSAV), the value of each element is (its valences) × (its frequency).

Table 2: The value of d1 and the value of d2

| Documents | Latent | Semantic | Analysis | is | very | useful | slow |
|---|---|---|---|---|---|---|---|
| d1 | 1 × (0) | 1 × (0) | 1 × (0) | 1 × (0) | 2 × (+0.3) | 1 × (+2.3) | 0 × (-1.1) |
| D2 | 1 × (0) | 1 × (0) | 1 × (0) | 1 × (0) | 0 × (+0.3) | 0 × (+2.3) | 0 × (-1.1) |

**Transferring all the documents of the testing data set and the training data set into the LSAVs in a sequential environmnent**

In this section, we proposed the algorithm 4, the algorithm 5, the algorithm 6, and the algorithm 7 in the sequential system as follows: All the positive documents of the training data set were transferred into one LSAV, called the positive LSAV group. All the negative documents of the training data set were transferred into one LSAV, called the negative LSAV group. Each document in the documents of the testing data set was transferred into one LSAV.

We implemented the algorithm 4 to create an order list of the LSAV which comprises all the meaningful terms of both the testing data set and the training data set in the sequential system. We proposed the algorithm 5 to transfer one document of the testing data set into one LSAV in the sequential system.
We built the algorithm 6 to transfer the positive documents of the training data set into one positive LSAV in the sequential system, called the positive LSAV group.

We proposed the algorithm 7 to transfer the negative documents of the training data set into one negative LSAV in the sequential system, called the negative LSAV group.

**Algorithm 4** Creating an order list of the LSAV

```
 1: Input: the documents of the training data set and the testing data set
 2: Output: an order list of the LSAV − AnOrderListOfTheLSAV
 3: Set AnOrderListOfTheLSAV ← ∅
 4: for all Each document into the documents of the training data set and the testing data set
    do
 5:     Split this document into the sentences
 6:     for all Each sentence in the sentences do
 7:         Split this sentence into the meaningful terms based on the sentiment lexicons of the
    bESD;
 8:         for all Each term in the meaningful terms do
 9:             if checking this term in AnOrderListOfTheLSAV is false then
10:                 Add this term into AnOrderListOfTheLSAV
11:             end if
12:         end for
13:     end for
14: end for
15: Return AnOrderListOfTheLSAV
```

**Transferring all the documents of the testing data set and the training data set into the the LSAVs in a distributed network system**

In this section, we implemented many algorithms in the distributed network system as follows: All the positive documents of the training data set were transferred into one LSAV, called the positive LSAV group. All the negative documents of the training data set were transferred into one LSAV, called the negative LSAV group. Each document in the documents of the testing data set was transferred into one LSAV.

This section comprises the algorithm 8, the algorithm 9, the algorithm 10, the algorithm 11, the algorithm 12, the algorithm 13, the algorithm 14, and the algorithm 15.

We created an order list of the LSAV which comprises all the meaningful terms of both the testing data set and the training data set in the distributed network system in the algorithm 8 and the algorithm 9. This stage included two phases: the Hadoop Map phase (M) and the Hadoop Reduce phase (R). The input of M was the documents of the testing data set and the training data set. The output of R is one term. The input of R was the output of M, thus, the input of R was one term. The output of R was an order list of the LSAV − AnOrderListOfTheLSAV.

We transferred one document of the testing data set into one LSAV in the parallel system in the algorithm 10 and the algorithm 11. This stage included two phases: the Hadoop Map phase

**Algorithm 5** Transferring one document of the testing data set into one LSAV in the sequential system

```
1: Input: one document in English and an order list of the LSA − AnOrderListOfTheLSAV;
2: Output: one LSAV;
3: Set Columns ← the length of AnOrderListOfTheLSAV
4: Set Rows ← the positive documents of the training data set
5: Set LSAV ←  with its column is Columns and its rows is Rows
6: Set i ← 0
7: for all Each term in AnOrderListOfTheLSAV  do
8:     Number := Count this term in this document
9:     Valence := Get the valence of this term based on the sentiment lexicons of the bESD
10:    Set LSAV[0][i] ← Number × Valence
11:    Set i ← i + 1
12: end for
13: for j:=1; j < Rows; j++ do
14:    for i:=0; i < Columns; i++ do
15:        Set LSAV[j][i] ← 0
16:    end for
17: end for
18: Return LSAV
```

(M) and the Hadoop Reduce phase (R). The input of M was one document in English and an order list of the LSA − AnOrderListOfTheLSAV. The output of M was one row of LSAV. The input of R was the output of M, thus, the input of R was one row of LSAV. The output of R was one LSAV of this document.

We transferred the positive documents of the training data set into one positive LSAV in the distributed system, called the positive LSAV group in the algorithm 12 and the algorithm 13. This stage included two phases: the Hadoop Map phase (M) and the Hadoop Reduce phase (R). The input of M was the positive documents in English and an order list of the LSA − AnOrderListOfTheLSAV. The output of M was one row of PositiveLSAV. The input of R was the output of M, thus, the input of R was one row of PositiveLSAV. The output of R was PositiveLSAV.

We transferred the negative documents of the training data set into one negative LSAV in the parallel system, called the negative LSAV group in the algorithm 14 and the algorithm 15. This stage included two phases: the Hadoop Map phase (M) and the Hadoop Reduce phase (R). The input of the Hadoop Map phase was the negative documents in English and an order list of the LSA − AnOrderListOfTheLSAV. The output of the Hadoop Map phase was one row of NegativeLSAV. The input of the Hadoop Reduce phase was the output of the Hadoop Map, thus, the input of the Hadoop Reduce phase was one row of NegativeLSAV. The output of the Hadoop Reduce phase was NegativeLSAV.

**Algorithm 6** Transferring the positive documents of the training data set into one positive LSAV in the sequential system

```
1: Input: the positive documents of the training data set and an order list of the LSA – AnOrderListOfTheLSAV
2: Output: one positive LSAV - the positive LSAV group
3: Set Columns ← the length of AnOrderListOfTheLSAV
4: Set Rows ← the positive documents of the training data set
5: Set PositiveLSAV ← with its column is Columns and its rows is Rows
6: for j := 0; j < Rows; j++ do
7:     Set i ← 0
8:     for all Each term in AnOrderListOfTheLSAV do
9:         Number := Count this term in the document (j) of the positive documents of the training data set
10:        Valence := Get the valence of this term based on the sentiment lexicons of the bESD
11:        Set PositiveLSAV[j][i] ← Number × Valence
12:        Set i ← i + 1
13:    end for
14: end for
15: Return PositiveLSAV
```

## 3.3 Using the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in both a sequential environment and a parallel distributed system

### Using the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in a sequential environment

This sub-section has the algorithm 16, and the algorithm 17. In this section, we used the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in a sequential environment.

We built the algorithm 16 to cluster one LSAV (corresponding one document of the testing data set) into either positive polarity or the negative polarity in the sequential environment.

We proposed the algorithm 17 to cluster all the documents of the testing data set into either the positive or the negative in the sequential system by using the LSA and the DNC.

### Using the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in a distributed system

This part includes the algorithm 18, the algorithm 19, the algorithm 20, and the algorithm 21.

In this part, we used the LSA and a DNC to cluster the documents of the testing data set into either the positive or the negative in a distributed system.

We clustered one LSAV (corresponding one document of the testing data set) into either positive polarity or the negative polarity in the distributed environment in the algorithm 18 and the algorithm 19. This stage comprised two phases: the Hadoop Map phase (M) and the Hadoop Reduce phase (R). The input of M was one LSAV (corresponding one document of the testing data set); the positive LSAV group and the negative LSAV group. The output of M was the result of the sentiment classification of one document. The input of R was the output of M, thus, the input of R was the result of the sentiment classification of one document. The output of R

**Algorithm 7** Transferring the negative documents of the training data set into one negative LSAV in the sequential system

```
1: Input: the negative documents of the training data set and an order list of the LSA – AnOrderListOfTheLSAV
2: Output: one negative LSAV – the positive LSAV group
3: Set Columns ← the length of AnOrderListOfTheLSAV
4: Set Rows ← the negative documents of the training data set
5: Set NegativeLSAV ← with its column is Columns and its rows is Rows
6: for j := 0; j < Rows; j++ do
7:     Set i ← 0
8:     for all Each term in AnOrderListOfTheLSAV do
9:         Number := Count this term in the document (j) of the positive documents of the training data set
10:        Valence := Get the valence of this term based on the sentiment lexicons of the bESD
11:        Set NegativeLSAV[j][i] ← Number × Valence
12:        Set i ← i + 1
13:    end for
14: end for
15: Return NegativeLSAV
```

was the result of the sentiment classification of this document of the testing data set.

We used the LSA and the DNC to cluster the documents of the testing data set into either the positive or the negative in the distributed system in the algorithm 20 and the algorithm 21. This stage comprised two phases: the Hadoop Map phase (M) and the Hadoop Reduce phase (R). The input of M was the documents of the testing data set. The output of M was the result of the sentiment classification of one document. The input of R was the output of M, thus, the input of R is the result of the sentiment classification of one document. The output of R was the results of the sentiment classification of the documents of the testing data set.

## 4 Experiment

To implement the proposed model, we have already used Java programming language to save the 11,000,000 documents of the testing data set and the 5,000,000 documents of the training data set, and to save the results of emotion classification. The proposed model was implemented in both the sequential system and the distributed network environment.
The configuration of one server is: IntelÂŽ Server Board S1200V3RPS, IntelÂŽ PentiumÂŽ Processor G3220 (3M Cache, 3.00 GHz), 2GB CC3-10600 ECC 1333 MHz LP Unbuffered DIMMs; and the operating system of the server is: Cloudera.
Our novel model related to the Latent Semantic Analysis and a Dennis Coefficient is implemented in the sequential environment with the configuration as follows: The sequential environment in this research includes 1 node (1 server).
The proposed model related to the Latent Semantic Analysis and a Dennis Coefficient is performed in the Cloudera parallel network environment with the configuration as follows: This Cloudera system includes 9 nodes (9 servers). All 9 nodes have the same configuration information
In Table 3, we show the accuracy and the results of our novel model in the testing data set.
The average time of the classification of our new model for the English documents in testing data set are displayed in Table 4.

**Algorithm 8** Performing the Hadoop Map phase

```
1: Input: the documents of the testing data set and the document of the training data set
2: Output: one term;//the output of M.
3: Input the documents of the testing data set and the document of the training data set into
   M in the Cloudra system
4: for all Each document into the documents of the training data set do
5:     Split this document into the sentences
6:     for all Each sentence in the sentences do
7:         Split this sentence into the meaningful terms based on the sentiment lexicons of the
   bESD
8:         for all Each term in the meaningful terms do
9:             Return this term;//the output of the Hadoop Map.
10:            for all Each document in the documents of the testing data set do
11:                Split this document into the sentences
12:                for all Each sentence into the sentences do
13:                    Split this sentence into the meaningful terms according to the sentiment
   lexicons of the bESD
14:                    for all Each term in the meaningful terms do
15:                        Return this term;//the output of M
16:                    end for
17:                end for
18:            end for
19:        end for
20:    end for
21: end for
22:
```

Table 3: The accuracy and The results of our novel model in the testing data set.

| | Testing Dataset | Correct Classification | Incorrect Classification | Accuracy |
|---|---|---|---|---|
| Negative | 5,500,000 | 4,884,751 | 615,249 | |
| Positive | 5,500,000 | 4,878,849 | 621,151 | 88.76% |
| Summary | 11,000,000 | 9,763,600 | 1,236,400 | |

**Algorithm 9** Implementing the Hadoop Reduce phase

```
1: Input: one term from M
2: Output: an order list of the LSAV – AnOrderListOfTheLSAV
3: Receive on term from M;
4: if checking this term in AnOrderListOfTheLSAV is false then
5:     Add this term into AnOrderListOfTheLSAV
6: end if
7: Return AnOrderListOfTheLSAV
```

**Algorithm 10** Performing the Hadoop Map phase

```
1: Input: one document in English and an order list of the LSA – AnOrderListOfTheLSAV
2: Output: one row of LSAV; //the output of M.
3: Input one document in English and an order list of the LSA – AnOrderListOfTheLSAV into
   the Hadoop Map in the Cloudera
4: Set Columns ← the length of AnOrderListOfTheLSAV
5: Set Rows ← the positive documents of the training data set
6: Set OneRowOfLSAV ← with its columns is Columns
7: Set i ← 0
8: for all Each term in AnOrderListOfTheLSAV do
9:     Number := Count this term in this document
10:    Valence := Get the valence of this term based on the sentiment lexicons of the bESD
11:    Set OneRowOfLSAV[i] ← Number × Valence
12:    Set i ← i + 1
13: end for
14: Return OneRowOfLSAV
```

**Algorithm 11** Implementing the Hadoop Reduce phase

```
1: Input: one row of LSAV; //the output of M.
2: Output: one LSAV
3: Receive one row of LSAV
4: Add this row into LSAV
5: for j:= 1; j < Rows; j++ do
6:     for i := 0; i < Columns; i++ do
7:         Set LSAV[j][i] ← 0
8:     end for
9: end for
10: Return LSAV
```

**Algorithm 12** Performing the Hadoop Map phase

```
1: Input: the positive documents in English and an order list of the LSA – AnOrderListOfTheL-
   SAV
2: Output: one row of LSAV; //the output of M.
3: Set Columns ← the length of AnOrderListOfTheLSAV
4: Set Rows ← the positive documents of the training data set
5: for j := 0; j < Rows; j++ do
6:     Set OneRowOfPositiveLSAV ← with its column is Columns
7:     Set i ← 0
8:     for all Each term in AnOrderListOfTheLSAV do
9:         Number := Count this term in the document (j) of the positive documents of the
   training data set
10:        Valence := Get the valence of this term based on the sentiment lexicons of the bESD
11:        Set OneRowOfPositiveLSAV[i] ← Number × Valence
12:    end for
13:    Set i ← i + 1
14: end for
15: Return OneRowOfPositiveLSAV; //the output of M
```

**Algorithm 13** Implementing the Hadoop Reduce phase

```
1: Input: one row of LSAV; //the output of M.
2: Output: PositiveLSAV
3: Receive one row of PositiveLSAV
4: Add this row into PositiveLSAV
5: Return PositiveLSAV
```

**Algorithm 14** Performing the Hadoop Map phase

```
1: Input: the negative documents in English and an order list of the LSA – AnOrderListOfTheLSAV
2: Output: one row of LSAV; //the output of M.
3: Set Columns $\leftarrow$ the length of AnOrderListOfTheLSAV
4: Set Rows $\leftarrow$ the negative documents of the training data set
5: for j := 0; j < Rows; j++ do
6:     Set OneRowOfNegativeLSAV $\leftarrow$ with its column is Columns
7:     Set i $\leftarrow$ 0
8:     for all Each term in AnOrderListOfTheLSAV do
9:         Number := Count this term in the document (j) of the negative documents of the training data set
10:        Valence := Get the valence of this term based on the sentiment lexicons of the bESD
11:        Set OneRowOfNegativeLSAV[i] $\leftarrow$ Number $\times$ Valence
12:        Set i $\leftarrow$ i + 1
13:    end for
14: end for
15: Return OneRowOfPositiveLSAV; //the output of M
```

**Algorithm 15** Implementing the Hadoop Reduce phase

```
1: Input: one row of LSAV; //the output of M.
2: Output: NegativeLSAV
3: Receive one row of NegativeLSAV
4: Add this row into NegativeLSAV
5: Return NegativeLSAV
```

Table 4: Average time of the classification of our new model for the English documents in testing data set

| | Average time of the classification 11,000,000 English documents. |
|---|---|
| The Latent Semantic Analysis and a Dennis Coefficient in the sequential environment | 43,460,194 seconds |
| The Latent Semantic Analysis and a Dennis Coefficient in the Cloudera distributed system with 3 nodes | 13,120,064 seconds |
| The Latent Semantic Analysis and a Dennis Coefficient in the Cloudera distributed system with 6 nodes | 7,360,032 seconds |
| Latent Semantic Analysis and a Dennis Coefficient in the Cloudera distributed system with 9 nodes | 4,851,132 seconds |

**Algorithm 16** Clustering one LSAV (corresponding one document of the testing data set) into either positive polarity or the negative polarity in the sequential environment

```
1: Input: one LSAV (corresponding one document of the testing data set); the positive LSAV group and the negative LSAV group.
2: Output: positive, negative, neutral;
3: Measure_1 := Similarity measure between this LSAV and the positive LSAV group by using the eq. (3) of the calculating a valence of one word (or one phrase) in English in the sub-section [Overview of identifying the valence and the polarity of one term in English using a DNC].
4: Measure_2 := Similarity measure between this LSAV and the negative LSAV group by using the eq. (3) of the calculating a valence of one word (or one phrase) in English in the sub-section [Overview of identifying the valence and the polarity of one term in English using a DNC].
5: if Measure_1 is greater than Measure_2 then
6:     Return positive
7: elseMeasure_1 is less than Measure_2
8:     Return negative
9: end if
10: Return neutral
```

**Algorithm 17** Clustering all the documents of the testing data set into either the positive or the negative in the sequential system by using the LSA and the DNC

```
1: Input: the documents of the testing data set and the training data set.
2: Output: positive, negative, neutral;
3: the creating a bESD in a sequential environment in the sub-section [Creating a basis English sentiment dictionary (bESD) in a sequential environment].
4: the algorithm 4 to create an order list of the LSAV which comprises all the meaningful terms of both the testing data set and the training data set in the sequential system.
5: the algorithm 6 to transfer the positive documents of the training data set into one positive LSAV in the sequential system, called the positive LSAV group.
6: the algorithm 7 to transfer the negative documents of the training data set into one negative LSAV in the sequential system, called the negative LSAV group.
7: Results := null;
8: for all Each document in the documents of the testing data set do
9:     One LSAV := the algorithm 5 to transfer one document of the testing data set into one LSAV in the sequential system.
10:    OneResult := the algorithm 16 to cluster one LSAV (corresponding one document of the testing data set) into either positive polarity or the negative polarity in the sequential environment.
11:    Add OneResult into Results;
12: end for
13: Return Results;
```

**Algorithm 18** Implementing the Hadoop Map phase

```
1: Input: one LSAV (corresponding one document of the testing data set); the positive LSAV
   group and the negative LSAV group.
2: Output: the result of the clustering – OneResult; //the output of M.
3: Measure_1 := Similarity measure between this LSAV and the positive LSAV group by using
   the eq. (3) of the calculating a valence of one word (or one phrase) in English in the sub-
   section [Overview of identifying the valence and the polarity of one term in English using a
   DNC].
4: Measure_2 := Similarity measure between this LSAV and the negative LSAV group by using
   the eq. (3) of the calculating a valence of one word (or one phrase) in English in the sub-
   section [Overview of identifying the valence and the polarity of one term in English using a
   DNC]
5: if Measure_1 is greater than Measure_2 then
6:     OneResult := positive;
7: else
8:     if Measure_1 is less than Measure_2 then
9:         OneResult := negative;
10:    else
11:        OneResult := neutral;
12:    end if
13: end if
14: Return OneResult; //the output of M
```

**Algorithm 19** Performing the Hadoop Reduce phase

```
1: Input: OneResult; //the output of M.
2: Output: positive, negative, neutral;
3: Receive OneResult;
4: Return OneResult;
```

**Algorithm 20** Implementing the Hadoop Map phase

```
1: Input: the documents of the testing data set and the training data set.
2: Output: positive, negative, neutral;
3: the creating a bESD in a distributed system in the sub-section [Creating a basis English
   sentiment dictionary (bESD) in a distributed system].
4: the algorithm 4 to create an order list of the LSAV which comprises all the meaningful terms
   of both the testing data set and the training data set in the sequential system.
5: creating an order list of the LSAV which comprises all the meaningful terms of both the
   testing data set and the training data set in the distributed network system in the algorithm
   8 and the algorithm 9.
6: transferring the positive documents of the training data set into one positive LSAV in the
   sequential system, called the positive LSAV group in the algorithm 12 and the algorithm 13.
7: transferring the negative documents of the training data set into one negative LSAV in the
   sequential system, called the negative LSAV group in the algorithm 14 and the algorithm 15.
8: Input the documents of the testing data set, the positive LSAV group and the negative LSAV
   group into the Hadoop Map in the Cloudera system;
9: for all Each document in the documents of the testing data set do
10:    One LSAV := transferring one document of the testing data set into one LSAV in the
   parallel system in the algorithm 10 and the algorithm 11.
11:    OneResult :=clustering one LSAV (corresponding one document of the testing data set)
   into either positive polarity or the negative polarity in the distributed environment in the
   algorithm 18 and the algorithm 19.
12:    Return OneResult;//the output of M
13: end for
14: Return OneResult;//the output of M
```

**Algorithm 21** Performing the Hadoop Reduce phase

```
1: Input: OneResult − the result of the sentiment classification of one document (the input of
   R is the output of M).
2: Output: the results of the sentiment classification of the documents of the testing data set;
3: Receive OneResult − the result of the sentiment classification of one document.
4: Add this OneResult into the results of the sentiment classification of the documents of the
   testing data set;
5: Return the results of the sentiment classification of the documents of the testing data set;
```

## 5 Conclusion

In this survey, a new model has been proposed to classify sentiment of many documents in English using the Latent Semantic Analysis and a Dennis Coefficient with Hadoop Map (M) /Reduce (R) in the Cloudera parallel network environment. Based on our proposed new model, we have achieved 88.76% accuracy of the testing data set in Table 3, and this is better than the accuracies of many previous models of the semantic analysis. Besides, we have also compared the novel model with the previous models, and the experiments and the results of our proposed model are better than that of the previous model. Until now, not many studies have shown that the clustering methods can be used to classify data. According to Table 4, the average time of the sentiment classification of using the Latent Semantic Analysis and a Dennis Coefficient in the sequential environment is 43,460,194 seconds / 11,000,000 English documents and it is greater than the average time of the sentiment classification of using the Latent Semantic Analysis and a Dennis Coefficient in the Cloudera parallel network environment with 3 nodes which is 13,120,064 seconds / 11,000,000 English documents. The average time of the sentiment classification of using the Latent Semantic Analysis and a Dennis Coefficient in the Cloudera parallel network environment with 9 nodes is 4,851,132 seconds / 11,000,000 English documents, and It is the shortest time in the table. Besides, the average time of the sentiment classification of using the Latent Semantic Analysis and a Dennis Coefficient in the Cloudera parallel network environment with 6 nodes is 7,360,032 seconds / 11,000,000 English documents
The accuracy of the proposed model is dependent on many factors as follows: (1) The LSA − related algorithms. (2) The testing data set. (3) The documents of the testing data set must be standardized carefully. (4) Transferring one document into one LSAV.

Table 5: Comparisons of our model′s positives and negatives the surveys related to the Latent Semantic Analysis (LSA)

| Studies | Approach | Positives | Negatives |
|---|---|---|---|
| [4] | Unsupervised Learning by Probabilistic Latent Semantic Analysis | The survey presents perplexity results for different types of text and linguistic data collections and discusses an application in automated document indexing. The experiments indicate substantial and consistent improvements of the probabilistic method over standard Latent Semantic Analysis. | No mention |
| [6] | The latent semantic analysis theory of acquisition, induction, and representation of knowledge. | A new general theory of acquired similarity and knowledge representation, latent semantic analysis (LSA), is presented and used to successfully simulate such learning and several other psycholinguistic phenomena. | No mention |
| Our work | LSA using A DNC | The positives and negatives of the proposed model are given in the Conclusion section. | |

The execution time of the proposed model is dependent on many factors as follows: (1) The parallel network environment such as the Cloudera system. (2) The distributed functions such as Hadoop Map (M) and Hadoop Reduce (R). (3) The LSA − related algorithms. (4) The performance of the distributed network system. (5) The number of nodes of the parallel network environment. (6) The performance of each node (each server) of the distributed environment. (7) The sizes of the training data set and the testing data set. (8) Transferring one document into one LSAV.

The proposed model has many advantages and disadvantages. Its positives are as follows: It uses the Latent Semantic Analysis and a Dennis Coefficient to classify semantics of English documents based on sentences. The proposed model can process millions of documents in the shortest time. This study can be performed in distributed systems to shorten the execution time of the proposed model. It can be applied to other languages. Its negatives are as follows: It has a low rate of accuracy. It costs too much and takes too much time to implement this proposed model.
To understand the scientific values of this research, we have compared our model's results with many studies in the tables below. Our novel model has more benefits than the studies in the tables, and the results of this model are better than that of the works in the tables.
In Table 5, we present the comparisons of our model′s positives and negatives the surveys related to the Latent Semantic Analysis (LSA).
The comparisons of our model′s benefits and drawbacks with the studies related to the DENNIS coefficient (DNC) are shown in Table 6.

Table 6: Comparisons of our model′s benefits and drawbacks with the studies related to the DENNIS coefficient (DNC)

| Studies | Approach | Benefits | Drawbacks |
|---|---|---|---|
| [5] | Analysis of Macromolecular Polydispersity in Intensity Correlation Spectroscopy: The Method of Cumulants | A method is described by which the distribution function of the decay rates (and thus the extent of polydispersity) can be characterized, in a light scattering experiment, by calculation of the moments or cumulants. | No mention |
| [3] | A Survey of Binary Similarity and Distance Measures | The authors collected 76 binary similarity and distance measures used over the last century and reveal their correlations through the hierarchical clustering technique | No mention |
| Our work | LSA using A DNC | The advantages and disadvantages of this survey are shown in the Conclusion section. | |

## Conflict of interests

The authors declare that there is no conflict of interests.

## Bibliography

[1] Bai, A.; Hammer, H.; Yazidi, A.; Engelstad, P. (2014); Constructing sentiment lexicons in Norwegian from a large text corpus, *2014 IEEE 17th International Conference on Computational Science and Engineering*, 231-237, 2014.

[2] Baldocchi, D.D.; Hincks, B.B.; Meyers, T.P.(1988); Measuring Biosphere-Atmosphere Exchanges of Biologically Related Gases with Micrometeorological Methods, *Ecology society of America*, 59(5), 1331-1340, 1988.

[3] Choi, S.-S; Cha, S.-H.; Tappert, C.C. (2010); A Survey Of Binary Similarity And Distance Measures, *Systemics, Cybernetics And Informatics*, 8(1), 43-48, 2010.

[4] Hofmann, T. (2001); Unsupervised Learning by Probabilistic Latent Semantic Analysis, *Machine Learning*, 42(1-2), 177-196, 2001.

[5] Koppel, D.E. (1972); Analysis of Macromolecular Polydispersity in Intensity Correlation Spectroscopy: The Method of Cumulants, *The Journal of Chemical Physics*, 57(11), 4814, 1972.

[6] Landauer, T.K.; Dumais, S. T. (1997); A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge, *Psychological Review*, 104(2), 211-240, 1997.

[7] Landauer, T.K.; Foltz, P. W.; Laham, D. (2009); An introduction to latent semantic analysis, *Discourse Processes*, 25(2-3), 259-284, 2009.

[8] Ngoc, P.V.; Ngoc, C.V.T.; Ngoc, T.V.T. et al. (2017); A C4.5 algorithm for english emotional classification, *Evolving Systems*, 1-27, 2017.

[9] Phu, V.N. ; Tuoi, P.T. (2014); Sentiment classification using Enhanced Contextual Valence Shifters, *International Conference on Asian Language Processing (IALP)*, 224-229, 2014.

[10] Phu, V.N.; Dat, N.D.; Tran, D.T.N.; Chau, V.T.N.; Nguyen, T.A.(2017); Fuzzy C-Means for English Sentiment Classification in a Distributed System, *International Journal of Applied Intelligence*, 45(3), 717-738 2017.

[11] Phu, V.N.; Chau, V.T.N.; Tran, D.T.N. (2017); SVM for English Semantic Classification in Parallel Environment, *International Journal of Speech Technology*, 20(3), 487-508, 2017.

[12] Phu, V.N.; Tran, V.T.N.; Chau, V.T.N. et al. (2017); A Decision Tree using ID3 Algorithm for English Semantic Analysis, *International Journal of Speech Technology*, 20(3), 593-613, 2017.

[13] Phu, V.N.; Chau, V.T.N.; Tran, V.T.N. et al. (2017); A Vietnamese adjective emotion dictionary based on exploitation of Vietnamese language characteristics, *International Journal of Artificial Intelligence Review (AIR)*, 1-67, 2017

[14] Phu, V.N., Chau, V.T.N., Dat, N.D. et al. (2017); A Valences-Totaling Model for English Sentiment Classification, *International Journal of Knowledge and Information Systems*, 53(3), 579-636, 2017.

[15] Phu, V.N.; Chau, V.T.N.; Tran, V.T.N(2017); Shifting Semantic Values of English Phrases for Classification, *International Journal of Speech Technology*, 20(3), 579-636, 2017.

[16] Phu, V.N., Chau, V.T.N., Tran, V.T.N. et al. (2017); A Valence-Totaling Model for Vietnamese Sentiment Classification, *International Journal of Evolving Systems*, 1-47, 2017.

[17] Phu, V.N., Tran, V.T.N., Chau, V.T.N. et al. (2017); Semantic Lexicons of English Nouns for Classification, *International Journal of Evolving Systems*, 1-69, 2017.

[18] Turney, D. P.; Littman, M.L. (2002); Unsupervised Learning of Semantic Orientation from a Hundred-Billion-Word Corpus, *arXiv:cs/0212012, Learning*, 2002.

[19] Cambridge English Dictionary (2017); *http://dictionary.cambridge.org/*

[20] Longman English Dictionary (2017); *http://www.ldoceonline.com/*

[21] Oxford English Dictionary (2017); *http://www.oxforddictionaries.com/*