Language Independent Semantic Kernels for Short-text Classification

Kwanho Kim[a], Beom-suk Chung[b], Yerim Choi[b], Seungjun Lee[b], Jae-Yoon Jung[a,*], Jonghun Park[b]

[a]*Department of Industrial and Management Systems Engineering, Kyung Hee University, Yongin, Gyeonggi 446-701, Republic of Korea*
[b]*Department of Industrial Engineering, Seoul National University, Seoul 151-744, Republic of Korea*

Abstract

Short-text classification is increasingly used in a wide range of applications. However, it still remains a challenging problem due to the insufficient nature of word occurrences in short-text documents, although some recently developed methods which exploit syntactic or semantic information have enhanced performance in short-text classification. The language-dependency problem, however, caused by the heavy use of grammatical tags and lexical databases, is considered the major drawback of the previous methods when they are applied to applications in diverse languages. In this article, we propose a novel kernel, called language independent semantic (LIS) kernel, which is able to effectively compute the similarity between short-text documents without using grammatical tags and lexical databases. From the experiment results on English and Korean datasets, it is shown that the LIS kernel has better performance than several existing kernels.

Keywords: Short-text document classification, Kernel method, Similarity measure, Language independent semantic kernel

*Corresponding author
Email address: jyjung@khu.ac.kr (Jae-Yoon Jung)

1. Introduction

In the past decade, short-text documents have been widely used in various applications in diverse languages. Many recent resources on the Web regardless of the languages exist in the form of short-text documents, including Website summaries, document snippets, image captions, and news comments. In social networks and micro-blogging services, users usually write short-texts to describe their ideas, feelings, and opinions within a few sentences such as tweets on Twitter and status updates on Facebook (Musiał & Kazienko, 2011). In addition, the spreading of short-text documents has been limited not only to Web-based services but also to mobile applications.

Therefore, an effective method for classifying short-text documents becomes increasingly important in various research areas such as information retrieval, recommendation, and social network analysis (Leong et al., 2012; Liu et al., 2012; Tomas & Vicedo, 2012). Short-text document classification is still considered a challenging problem mainly because of the following natures of short-text documents. First, the small number of words in a short-text document is not so enough to effectively classify the documents compared to that of a lengthy text document (Taksa et al., 2007). Second, the low occurrence rate of a word across documents causes the small number of words in common among documents (Sheth et al., 2005).

To address the issues, a few classification methods for short-text documents were recently proposed that attempt to calculate the similarity between documents by utilizing learning-based techniques (Faguo et al., 2010; Sriram et al., 2010). It is said that these methods are language dependent in that they mainly exploit grammatical tags and lexical databases to reflect syntactic or semantic features of documents.

The previous methods have limitations in extracting syntactic and semantic features and calculating the similarity between documents due to the heavy use of the grammatical tags and lexical databases which are often unavailable in many languages. As a matter of fact, only a few number of lexical databases such as WordNet (Fellbaum, 2010) and FameNet (Baker et al., 1998) have currently been developed, and most of them are dedicated to English and Chinese. In addition, there is no natural language processor that produces grammatical tags such as part of speech (POS) tags and predicate argument structure (PAS) tags based on syntactic analysis cannot be used for documents in most languages. Moreover, the previous methods separately address the syntactic and semantic features of documents, which might not result in the satisfactory results.

Motivated by the above remarks, we propose a kernel, called language independent semantic (LIS) kernel, which aim to effectively calculate the similarity between short-text documents by utilizing both the syntactic and semantic features of documents without relying on grammatical tags of words and ready-made lexical databases. Unlike previous kernels, the LIS kernel accommodates the two features, syntactic and semantic, in a single kernel by extracting syntactic patterns and annotating semantic information on words appearing in a document. Specifically, LIS kernel extracts the syntax-based patterns from a document. To address the small number and low occurrence rate of words problem, we considers the three levels of semantic annotations, word, document, and category, on each of syntactically extracted pattern.

The remainder of this paper is organized as follows. First, previous kernels for short-text document classification are described in Section 2. The proposed syntax-based pattern extraction and annotation methods are presented in Section 3. Experiment results are presented to show the effectiveness of the proposed kernels in Section 4. Finally, we conclude this article in Section 5.

2. Related work

Existing kernels developed for text classification can be divided into: word occurrences, syntax features, semantic features, and both syntactic and semantic features. Table 1 summarizes kernels in terms of their considered features and language constraints. The most widely used kernels for text classification is Bag-of-word (BOW) kernel, which calculates the similarity between documents

Table 1
Kernels for text classification in previous work (G and L represent grammatical tags and lexical databases, respectively)

Considered feature	Kernel	Language constraints	Author
Word occurrence	Bag-of-word (BOW) kernel	None	Joachims (1998)
Word sequence	String kernel	None	Lodhi et al. (2002)
Syntactic structure	Syntactic parse tree (ST) kernel	G	Collins & Duffy (2001)
Word similarity	Semantic smoothing (SS) kernel	L	Siolas & d'Alché Buc (2000)
Word similarity	Latent semantic (LS) kernel	None	Cristianini et al. (2002)
Syntactic structure and word similarity	Syntactic semantic tree (SST) kernel	G and L	Bloehdorn & Moschitti (2007)

based on the number of word occurrences (Joachims, 1998), which implies that it does not use any syntactic or semantic features.

Many studies have been conducted focusing on expanding feature spaces by incorporating syntactic and semantic features. Firstly, String kernel includes syntactic features by using substrings of a document for representing the document (Lodhi et al., 2002). Specifically, the main idea of string kernel is that the more substrings two documents have in common, the more similar the documents are. Next, syntactic parse tree (ST) kernel uses a syntactic parse tree of a document as its syntactic feature (Collins & Duffy, 2001). This kernel computes the similarity between documents by comparing the production of all possible pairs of nodes and counting the number of common sub-trees.

The limitation of the kernels, BOW kernel, String, and ST kernels, is that they compute the similarities based on the number of common features such as words, substrings, and sub-trees, respectively. It means that when there are few words in common among documents, they cannot show satisfactory performance.

To resolve the problem, there have been some efforts to design kernels that incorporate semantic features by using a priori semantic knowledge such as semantic smoothing (SS) and latent semantic (LS) kernels. SS kernel utilizes a lexical database called WordNet to obtain semantic features of a document (Siolas & d'Alché Buc, 2000). In LS kernel, words in a document are annotated with semantically related words which are extracted from a semantic space where the document is implicitly mapped (Cristianini et al., 2002). More recently, the semantic and syntactic kernel uses predicate-argument structures to consider the lexical dependencies between words (Moschitti, 2009).

Especially for short-text classification, syntactic semantic tree (SST) kernel combines both syntactic and semantic features (Bloehdorn & Moschitti, 2007). The concept of SST kernel is based on ST kernel, which represents a document as a syntactic parse tree by using grammatical tags, and SS kernel, which uses a lexical database to incorporate semantic features. In terms of language independency, BOW, String, and LS kernels are language independently applicable, whereas other kernels, ST, SS, and SST, are not applicable in some languages due to their dependency to grammatical tags and lexical databases. Accordingly, we attempt to suggest a kernel that utilizes both the syntactic and semantic features of documents without relying on grammatical tags and lexical databases.

3. Language independent semantic (LIS) kernel

In this section, we introduce a language independent semantic (LIS) kernel which is developed for short-text classification. LIS kernel composed of three parts as shown in Figure 1: pattern extraction, semantic annotation, and similarity computation. First, it extracts patterns from a document by considering its syntactic information. Second, each extracted pattern is annotated in terms of three annotation levels, word, document, and category. Finally, LIS kernel computes the similarity between documents by using their extracted patterns according to the three annotation levels. For classification tasks, LIS kernel is then applied in a kernel machine which aims to classify new documents into one

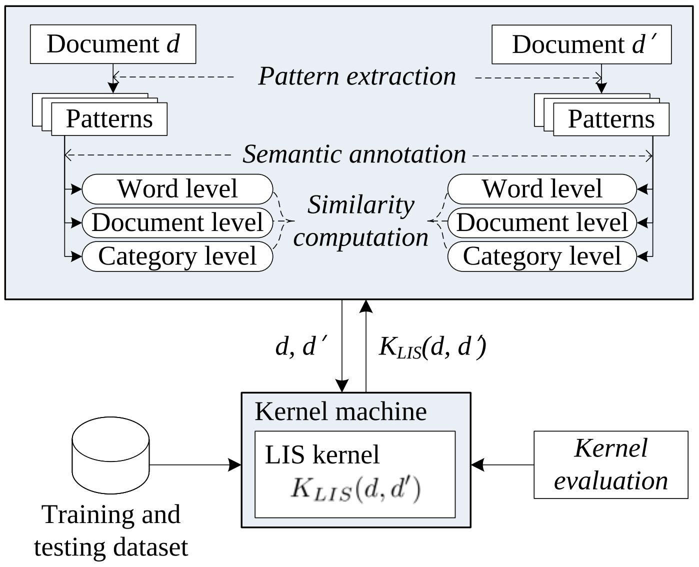


Fig. 1. An overview of LIS kernel.

of predefined categories based on the similarities between a new document and an existing documents in the category (Sanchez A, 2003). It provides the similarity between documents in a dataset to the kernel machine during its training and testing stages.

3.1. Syntax-based pattern extraction from short-text documents

To consider the syntactic features of documents when calculating similarity between documents, a syntax-based pattern extraction method is introduced. Here, a syntax-based pattern extracted from a document refers to a set of words appearing the document based on the syntax of a specific language. There exist some methods to extract patterns from a document based on different features of text documents. While a pattern is considered as a word that appear in a document in the word occurrence based method (Joachims, 1998), it consists of a set of consecutive words that appear in a document in the word sequence based method (Lodhi et al., 2002). On the other hand, in the parse tree based method based on syntactic structure, a pattern is extracted from the tree of a document by considering not only the word occurrence but also the word sequence in the document (Collins & Duffy, 2001).

Therefore, we focus on the pattern extraction method based on syntactic parse tree in this paper, since the method gives advantages to extract patterns through providing both occurrence and sequence of words compared to the alternative methods. We note that the alternatives can be utilized to extract the patterns from a document when its syntactic parse tree is unavailable.

Specifically, the tree based pattern extraction method utilizes the syntactic parse tree of a document. The underlying idea has been originally proposed for ST kernel that uses the sub-trees of the syntactic parse tree obtained from a document (Collins & Duffy, 2001). Since ST kernel extracts all the sub-trees which include more than one node, the extracted patterns from a document include not only the sequence of words but also their grammatical tags. Unlike the pattern extraction in ST kernel, the tree based method extract patterns by utilizing word sequences based on syntax without grammatical tags.

Each document is parsed by using an language processor that indicates the phrases in the document as a form of tree. For the parse tree of a document, all of its complete sub-trees which represent phrases in the document are considered. Subsequently, based on a complete sub-tree, a pattern is extracted by using an algorithm, called PATTERN_EXTRACTION algorithm shown in Figure 3. The examples shown in Figure 2 (a) and (b) depict the syntactic parse tree and the complete sub-trees of a document, "Patient billing service located in northern California", respectively, and Figure 2 (c) shows the final extracted patterns from the document.

In detail, the syntax-based patterns of a document are extracted based on the syntactic parse tree of the document by utilizing the algorithm that appends the words appearing on the leaf nodes related to a given node, *cu*, to a given empty pattern, *pa*. To keep the order of words in a document, the algorithm performs the breadth-first traversal of a tree. For a given node, *cu*, the algorithm

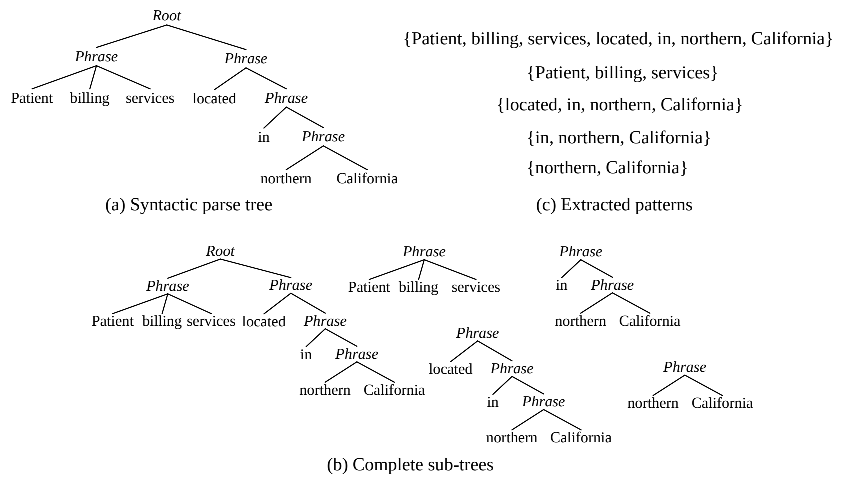


Fig. 2. An example of pattern extraction using parse tree.

```
PATTERN_EXTRACTION ($cu$, $pa$)
   1. for every child node $ch$ of $cu$ do
   2.      if $ch$ is a leaf node then
   3.        $append$($pa$, word on $ch$)
   4.      PATTERN_EXTRACTION ($ch$, $pa$)
   5. return $pa$
```

Fig. 3. Pattern extraction algorithm.

checks whether or not a child node, *ch*, is a leaf node by visiting every child node of *cu* (line 1 in Figure 3). In the case that *ch* is a leaf node, the word on *ch* is used for generating the syntax-based pattern of the tree, *pa* (line 2). The word on *ch* becomes the first word when *ch* is the first visited leaf node of a sub-tree rooted in *cu*, while the word is appended to existing *pa* by using a function, *append*(*pa*,word), which adds the word to the end of *pa* (line 3). The algorithm examines the remaining unvisited nodes of a tree until every node on the tree is visited.

3.2. Semantic annotation to patterns

Semantic information is annotated on each extracted pattern by considering three semantic levels: word, document, and category. First, the words that

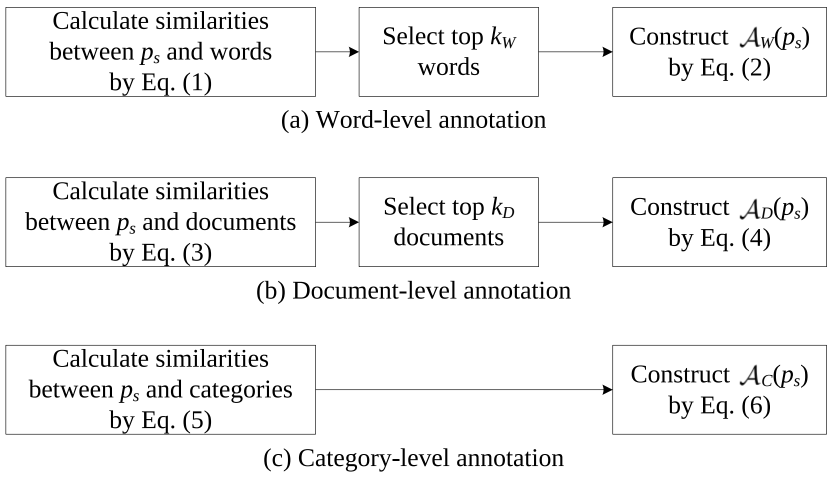

Calculate similarities between p_s and words by Eq. (1)
Select top k_W words
Construct $\mathcal{A}_W(p_s)$ by Eq. (2)
(a) Word-level annotation
Calculate similarities between p_s and documents by Eq. (3)
Select top k_D documents
Construct $\mathcal{A}_D(p_s)$ by Eq. (4)
(b) Document-level annotation
Calculate similarities between p_s and categories by Eq. (5)
Construct $\mathcal{A}_C(p_s)$ by Eq. (6)
(c) Category-level annotation

Fig. 4. Flowcharts of constructing three levels of semantic annotations on pattern p_s.

frequently co-occur within a pattern are used to define the meaning of the pattern in word-level annotation. Next, the meaning of a pattern is expanded by using words that appear in the similar documents to the pattern in document-level annotation. Lastly, the association between a pattern and a category based on their similarity is additionally considered in category-level annotation.

Three levels of semantic annotations on a pattern, called semantic pattern annotation (SPA), has been widely applied to enrich the meanings of patterns with the help of related words and documents (Luo et al., 2011; Mei et al., 2007). In SPA method, the meaning of a pattern is defined and expanded by annotating words, sentences, and related patterns. On the contrary, the similar words to a pattern are used for the document and category-levels on the pattern in this research. Moreover, while SPA method focuses on discovering the meaning of a pattern for human understanding, the representation scheme of semantic annotation enables to automatically compute the similarity between patterns for classification tasks.

We consider N documents, $D = \{d_n \mid n = 1, \ldots, N\}$, and M categories,

$C = \{c_m \mid m = 1, \ldots, M\}$, and document $d_n \in D$ is associated with $c_m \in C$. A set of distinct words that appear in documents in D is denoted by $W = \{w_l \mid l = 1, \ldots, L\}$, where and L are the number of distinct words. A document, $d_n \in D$, is represented as an L-dimensional vector, $d_n =< x_{1n}, \ldots, x_{Ln} >$, where x_{ln} is one if w_l appears in d_n, and zero otherwise. We also define a set of patterns as $P = \{p_s \mid s = 1, \ldots, S\}$, where S is the total number of distinct patterns extracted from documents in D by using the pattern extraction method presented in Section 3.1. A pattern, $p_s \in P$, is denoted by $p_s =< y_{1s}, \ldots, y_{Ls} >$, where y_{ls} is one if w_l appears in p_s, and zero otherwise.

Each pattern, p_s, is annotated semantically in word-, document-, and category-levels, as illustrated in the flowcharts of Figure 4. In the three types of annotations, the similarities are first calculated by comparing the pattern with words, documents, and categories. Specifically, in the former two annotations, only top similar words or documents are selected because the numbers of words and documents are too huge to apply all of them to semantic annotations directly. Finally, the selected words or documents in the former two levels and the similarities in three levels are used to construct three types of semantic annotation vectors, denoted by $\mathcal{A}_W(p_s)$, $\mathcal{A}_D(p_s)$, and $\mathcal{A}_C(p_s)$.

First, in word-level annotation, to calculate the similarity between p_s and w_l, mutual information (Zhao, 2010) is applied, denoted by m_{sl}, as:

$$mi(w_l, p_s) = \log \frac{Pr(w_l \wedge p_s)}{Pr(w_l)Pr(p_s)} \tag{1}$$

where $Pr(w_l \wedge p_s)$ is the probability that both w_l and p_s appear in a document, and $Pr(w_l)$ and $Pr(p_s)$ are the probabilities that w_l and p_s appear in a document, respectively.

Based on the similarities of all words in W with respect to pattern p_s, top k_W words, denoted by W_s, are selected to annotate semantics in the word level. The word-level annotation on p_s is finally generated in form of an L-dimensional vector, $\mathcal{A}_W(p_s) =< \alpha_{1s}, \cdots, \alpha_{Ls} >$, where

$$\alpha_{ls} = \begin{cases} mi(w_l, p_s), & \text{if } w_l \in W_s \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Next, the document-level annotation of p_s is constructed similar to the word-level. The cosine similarity metric is adopted to compute the similarity between d_n and p_s as:

$$cos(d_n, p_s) = \frac{\sum_{l=1}^{L} x_{ln} y_{ls}}{\sqrt{\sum_{l=1}^{L} x_{ln}^2}\sqrt{\sum_{l=1}^{L} y_{ls}^2}} \tag{3}$$

After calculating the similarities of all documents in D with respect to pattern p_s, top k_D documents, denoted by D_s, are selected to annotate semantics in the document level on p_s. The document-level annotation is then represented as an L-dimensional vector, $\mathcal{A}_D(p_s) =< \beta_{1s}, \cdots, \beta_{Ls} >$, where β_{ls} means the existence of w_l in p_s and is defined as:

$$\beta_{ls} = \delta\left(\sum_{d_i \in D_s} x_{li}\right) \tag{4}$$

where $\delta(\cdot)$ returns 1 if the given value is positive, and zero otherwise.

Finally, the category-level annotation on pattern p_s is obtained based on the similarity between the pattern and each of the categories in C. The similarity between p_s and c_m is measured as the ratio of the documents that contains the pattern in c_m, calculated as:

$$rat(c_m, p_s) = \frac{cf(c_m, p_s)}{M_m} \tag{5}$$

where M_m is the number of documents in c_m.

According to the category similarities with respect to p_s, the category-level annotation of p_s is represented as an M-dimensional vector, $\mathcal{A}_C(p_s) =< \gamma_{1s}, \cdots, \gamma_{Ms} >$, where γ_{ms} is the relative occurrence of p_s in c_m. Here, γ_{ms} is defined as:

$$\gamma_{ms} = rat(c_m, p_s) \tag{6}$$

For instance, consider extracted pattern that contains "Patient", "billing", "service", shown in Figure 2. The associated words in its word-level annotation include "patient", "electronic", "medical", "billing", and "services" when $k_W = 5$, while those in its document-level annotation are "hospital", "payment", "Internet", "surgery", and "patient" when $k_D = 5$. In the category-level annotation of the pattern, the associated categories to the pattern are "medical" and "information".

3.3. Similarity computation using LIS kernel

In this section, we describe how to calculate the similarity between two documents by using LIS kernel. First of all, components of LIS kernel are defined in Definition 1.

Definition 1. A pattern kernel, $K_P(p_s, p_{s'})$, for $s, s' = 1, \cdots, S$, which calculates the similarity between two patterns, p_s and $p_{s'}$, based on their semantic annotations is defined as:

$$K_P(p_s, p_{s'}) = \lambda_W K_W(p_s, p_{s'}) + \lambda_D K_D(p_s, p_{s'}) + \lambda_C K_C(p_s, p_{s'}) \tag{7}$$

In the equation, $K_W(p_s, p_{s'})$, $K_D(p_s, p_{s'})$, and $K_C(p_s, p_{s'})$ represent kernels that calculate similarities between two patterns based on word-, document-, and category-level annotations, and λ_W, λ_D, and λ_C are the similarity coefficients of three kernels, where $\lambda_W + \lambda_D + \lambda_C = 1$, $\lambda_W \geq 0$, $\lambda_D \geq 0$, and $\lambda_C \geq 0$. In detail, each kernel calculates the inner product between two corresponding annotations, and three kernels are respectively defined as:

$$K_W(p_s, p_{s'}) = \mathcal{A}_W(p_s) \cdot \mathcal{A}_W(p_{s'}) \tag{8}$$

$$K_D(p_s, p_{s'}) = \mathcal{A}_D(p_s) \cdot \mathcal{A}_D(p_{s'}) \tag{9}$$

$$K_C(p_s, p_{s'}) = \mathcal{A}_C(p_s) \cdot \mathcal{A}_C(p_{s'}) \tag{10}$$

By Definition 1, a pattern kernel is defined as a weighted linear combination of three types of kernels, which are for the similarity calculation according to the corresponding levels of semantic annotation. The similarity calculation based on three levels of semantic information is introduced for the pattern kernel to effectively reflect the characteristics of short-text documents. The semantics in terms of words and documents are critical elements to the pattern which should be used to classify documents. Note that two patterns with similar semantics are likely to often co-occur in the same document and also appear with many common words in documents. Moreover, semantics based on categories is specifically considered as additional semantic information on the pattern because two patterns with similar semantics are often found simultaneously in the same category. It can be said that the category-level of semantic annotation complements the other two levels of annotations by overcoming the insufficient number of words in common which are frequently found in short-text classification.

Using the similarity coefficients, λ_W, λ_D, and λ_C, the weights of the three types of kernels are decided, and the influence of the three levels of semantic annotation can be adjusted depending on datasets. The three levels can be altered or expanded depending on the purpose of semantic annotation.

Definition 2. LIS kernel that calculates the semantic similarity between two text documents based on a pattern kernel, $K_P(d_n, d_{n'})$, $n, n' = 1, \cdots, D$, is defined as:

$$K_{LIS}(d_n, d_{n'}) = \sum_{p_s \in pe(d_n)} \sum_{p_{s'} \in pe(d_{n'})} K_P(p_s, p_{s'}) \tag{11}$$

where $pe(d_n)$ represents a set of all possible patterns extracted from document d_n by using the pattern extraction method described in Section 3.1.

Propositions 1 and 2 show that K_{LIS}, K_P, K_W, K_D, and K_C are valid, so-called Mercer kernels.

Proposition 1. K_W, K_D, and K_C are Mercer kernels

Proof of Proposition 1. According to the theorem appeared in (Berg, 1990), a kernel, K, is a Mercer kernel if $K : \mathfrak{P} \times \mathfrak{P} \rightarrow \mathbb{R}$ is symmetric and positive semi-definite, where $\mathfrak{P}$ is a pattern space of document d_n, $n = 1, \cdots, N$. Therefore, K_W is a Mercer kernel, if $K_W(p_s, p_{s'}) = K_W(p_{s'}, p)$, $\forall p_s, p_{s'} \in \mathfrak{P}$, and $\sum_{s,s'=1}^{S} h_s h_{s'} K_W(p_s, p_{s'}) \geqq 0$, $\{h_1, \ldots, h_n\} \subseteq \mathbb{R}$.

This implies that, for K_W to be a Mercer kernel, the gram matrix of K_W, whose elements are $K_W(p_s, p_{s'})$, $\forall p_s, p_{s'} \in \mathfrak{P}$, must be positive semi-definite. K_D and K_C can also be proved to be Mercer kernels.

Since $\mathcal{A}_W(p_s)$, $\mathcal{A}_D(p_s)$, and $\mathcal{A}_C(p_s)$, $s = 1, \cdots, S$, are positive, the entries of gram matrices, K_W, K_D, and K_C, which are the inner products of the vectors, are also positive. The inner product is a symmetric function, implying that the gram matrices are also symmetric since $K_W(p_s, p_{s'}) = K_W(p_{s'}, p_s)$. Therefore, the gram matrices of K_W, K_D, and K_C are symmetric matrices with positive entries. They are diagonalized and all of their eigenvalues are positive, so the gram matrices are positive semi-definite. Hence, by definition, K_W, K_D, and K_C are Mercer kernels. $\square$

Based on Proposition 1 that shows that K_W, K_D, and K_C are Mercer kernels, those kernels, the pattern kernel and LIS kernel are built by using kernel combinations. The validity of K_P and K_{LIS} is proved in Proposition 11 by inducing that they are also Mercer kernels.

Proposition 2. K_P and K_{LIS} are Mercer kernels

Proof of Proposition 2. Output functions of kernel combinations such as the sum of kernels and the positively weighted kernels are Mercer kernels if the primitive kernels are Mercer kernels (Shawe-Taylor & Cristianini, 2004). Hence, pattern kernel, K_P, which is a sum of positively weighted component kernels, is a Mercer kernel since the component kernels, K_W, K_D, and K_C, are proved to be Mercer kernels by Proposition 2.

The convolution kernel is defined as a sum of composition kernels and if the composition kernels are Mercer kernels, the convolution kernel is also a Mercer kernel (Haussler, 1999). LIS kernel, K_{LIS}, which is a convolution kernel composed of K_P, is a Mercer kernel because K_P is proved to be a Mercer kernel. $\square$

The computational complexity of the proposed semantic annotation method is $O(L \times S)$. In addition, the complexity to calculate the similarity between documents by using LIS kernel is $O(S^2)$ because all possible similarities among patterns in a dataset are considered in the worst case. Note that the semantic annotation task can be done in the learning time before the classification of documents in the run-time.

4. Experiments

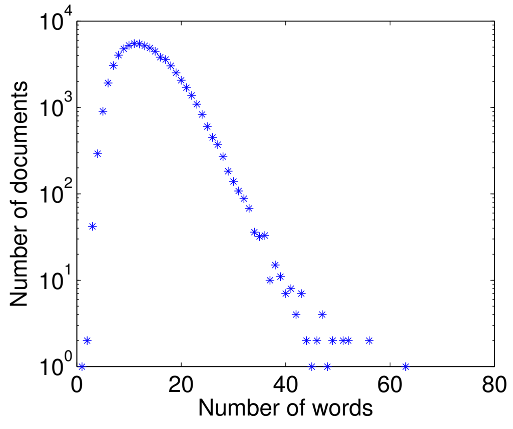
4.1. Datasets

To evaluate the proposed method, two languages, English and Korean, were selected for experiments. For the English dataset, Web-site information obtained from open directory project (ODP) (http://www.dmoz.org), the largest human-edited Web-site directory service. It provides hierarchically categorized Web-sites according to 15 top level categories. We randomly selected 10 top level categories, and total 68,625 Web-sites from the selected top level categories were gathered for the English dataset.

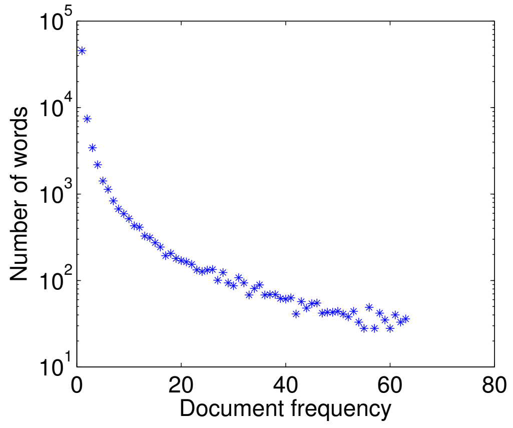
For the Korean dataset, we gathered Web-site information from Daum directory (http://directory.search.daum.net) which is the largest Korean Web-site directory service that provides 15 top level categories. Similar to the English dataset, we randomly selected 10 top level categories, and total 14,652 Web-sites from the selected top level categories were collected for the Korean dataset.

We built a text document by combining the title and the description of each Web-site. A half of the gathered documents were randomly selected for the parameter selection of the proposed kernel and training of a classifier, and the rest of the documents were used as testing data. The Korean and English documents which were prepared for the experiments consisted of 71,800 and 16,617 distinct words, respectively.

Figures 5 and 6 show the distributions of the number of words per document and the number of word occurrences across documents for the English and Korean dataset, respectively. The graphs imply that most documents contain



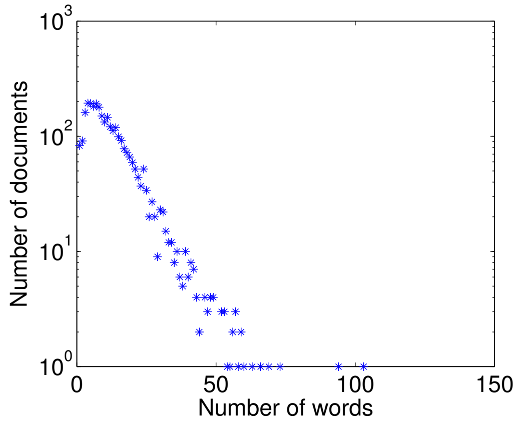
(a)



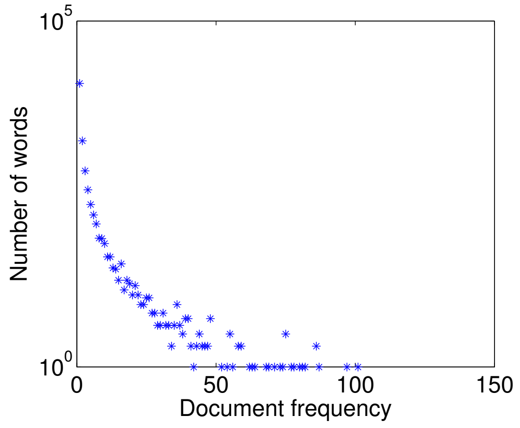
(b)

Fig. 5. Frequency distributions of document length and the number of documents per word in the English dataset.

the small number of words, and most words appear in the small number of documents. In the English dataset, 89.08% of the documents contain less than 20 words, and more than 99% contain less than 29 words. On the other hand, in the Korean dataset, 82.25% of the documents contain less than 20 words, and more than 99% contain less than 50 words.



(a)



(b)

Fig. 6. Frequency distributions of document length and the number of documents per word in the Korean dataset.

4.2. Experiment setting

4.2.1. Baselines

For comparison with the proposed kernel, we also evaluated three well-known kernels, BOW (Joachims, 1998), String (Lodhi et al., 2002), and ST (Collins & Duffy, 2001). We adopted the term frequency (TF) and inverted document frequency (TF-IDF) method to represent the bag-of-words of each document in BOW kernel. The values of the bag-of-words of a document then were normalized with zero-one. In String kernel, the window parameter that determines the number of consecutive words for pattern extraction was set to 3, and no significant performance change was observed in the different values of the window parameter. Unfortunately, in this experiments, kernels such as SS (Siolas & d'Alché Buc, 2000) and SST (Bloehdorn & Moschitti, 2007) were not applicable to the Korean dataset due to their language dependency.

4.2.2. Natural language processor

To implement experiments, we employed a natural language processor developed by the Stanford Natural Language Group (Klein & Manning, 2003) for the English documents, and KKM language processor (Lee et al., 2010) for the Korean documents.

4.2.3. Text document classifier

Text document classification tasks in the experiments were conducted by utilizing support vector machine (SVM), which is a binary classifier that attempts to divide two types of instances based on geometric principle. It has shown successful result from various text document classification tasks (Fu & Lee, 2012; Saleh et al., 2011). We adopted a SVM library called libsvm (Chang & Lin, 2011) and modified it to map documents into the semantic space we defined by replacing the existing similarity function with LIS kernel. In addition, we implemented the baselines, BOW, String, and ST kernels. Based on SVM, the one-versus-all method was applied for multi-class discrimination.

4.2.4. Evaluation metric

Prediction results can be categorized into four cases in the confusion matrix, as shown in Table 2, according to actual and predicted class labels of documents. Based on the notations in the confusion matrix, the classification performance was evaluated in terms of classification accuracy calculated as:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{12}$$

Table 2
Confusion matrix which defines four possible cases according to the actual and the predicted class labels for a category

	Predicted positive	Predicted negative
Actual positive	True positive (TP)	False negative (FN)
Actual negative	False positive (FP)	True negative (TN)

4.3. Parameter selection

To select the parameters of LIS kernel, we evaluated the classification performances obtained by using LIS kernel under various values of the annotation parameters and the similarity coefficients. Due to a great number of possible parameter combinations, we apply an empirical approach which results in the satisfactory results in many applications such as ranking (Liu et al., 2010), information retrieval (Can et al., 2008), text classification (Zhang & Lee, 2006), and similarity calculation of text documents (Wang & Oard, 2006).

Tables 3 and 4 represent the classification performances in terms of accuracy achieved by using LIS kernel based on the English and Korean datasets, respectively. In the tables, the values of accuracy are shown while annotation parameters, k_W and k_D, were varying whereas similarity coefficients, λ_W, λ_D, and λ_C, were fixed. The best classification performances were represented in bold.

In both datasets, the results show that the classification performances of LIS kernel were able to be significantly improved compared to those obtained without semantic annotation, represented by $k_W = 0$ and $k_D = 0$, even though only a few words and documents were used for annotating semantic information. Moreover, the classification performances were quite robust across different values of $k_W, k_D \in \{5, 10, 15, 20\}$.

Particularly, the best accuracy was observed under $k_W = k_D = 5$ for the English dataset, whereas it was found under $k_W = 5$ and $k_W = 20$ for the Korean dataset. When k_W and k_D became higher than 20 in both datasets, no improvements of the classification performances was found in both datasets. This implies that too much annotated words and documents can cause negative

Table 3
Classification performances obtained by using LIS kernel based on the English dataset obtained by varing k_W and k_D under the fixed similarity coefficients, $\lambda_W = \lambda_D = \lambda_C = 1/3$.

k_W	k_D	Accuracy	k_W	k_D	Accuracy
	0	0.318		0	0.749
	5	0.678		5	0.787
0	10	0.682	10	10	0.787
	15	0.682		15	0.790
	20	0.682		20	0.790
	0	0.738		0	0.772
	5	**0.801**		5	0.787
5	10	0.798	15	10	0.790
	15	0.798		15	0.790
	20	0.798		20	0.790

Table 4
Classification performances obtained by using LIS kernel based on the Korean dataset obtained by varing k_W and k_D underm the fixed similarity coefficients, $\lambda_W = \lambda_D = \lambda_C = 1/3$

k_W	k_D	Accuracy	k_W	k_D	Accuracy
	0	0.238		0	0.646
	5	0.707		5	0.718
0	10	0.707	10	10	0.713
	15	0.707		15	0.713
	20	0.707		20	0.713
	0	0.608		0	0.641
	5	0.718		5	0.724
5	10	0.713	15	10	0.707
	15	0.713		15	0.713
	20	**0.729**		20	0.718

effects in classifying documents, caused by too much information in semantic annotations. Therefore, the appropriate values of the parameters might be essential not only to improve classification performances but also to reduce computational costs for annotation tasks.

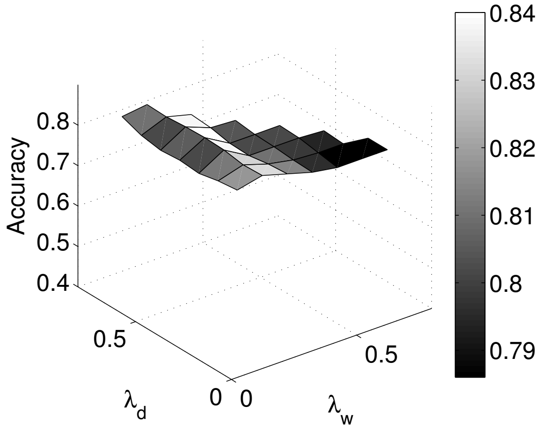
In addition, we also conducted the sensitivity analysis for classification performance with respect to similarity coefficients, λ_W, λ_D, and λ_C, which is designed to investigate how much the similarity coefficients affect the classification performance. Two graphs in Figure 7 show accuracy values according to datasets under different values of $0 \leq \lambda_W \leq 1$ and $0 \leq \lambda_D \leq 1$ at an interval of 0.1. Note that $\lambda_C = 1 - \lambda_W - \lambda_D$, and $k_W = k_D = 5$ were applied for the classification of the English dataset, while $k_W = 5$ and $k_D = 20$ for the classification of the Korean dataset. The best accuracy values were observed under $\lambda_W = 0.4$, $\lambda_D = 0.2$, and $\lambda_C = 0.4$ for the English dataset, and under $\lambda_W = 0.3$, $\lambda_D = 0.5$, and $\lambda_C = 0.2$ for the Korean dataset, respectively.

4.4. Comparison results

In this section, the performance evaluation results of LIS kernel in short-text classification are presented compared to three baseline kernels, BOW, String, and ST, on the English and Korean dataset.

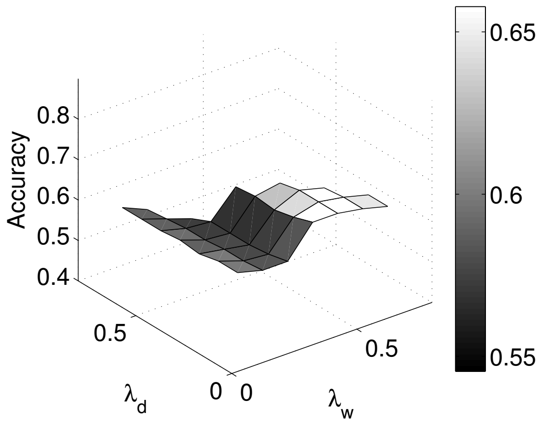
For the English dataset, the classification performances in terms of accuracy are shown in Figure 8. In detail, Figure 8 (a) depicts accuracy values of LIS kernel and three baseline kernels according to the number of categories. Categories were randomly selected in each number of categories, and the average accuracy was calculated across 30 repeated tests in each case. In each number of categories, LIS kernel outperformed three baseline kernels, and the average improvements of accuracy against BOW, String, and ST kernels were 9.03%, 33.66%, and 23.66%, respectively.

Figure 8 (b) depicts accuracy of LIS kernel and the baseline kernels according to the number of words per document. It also shows that LIS kernel performed better than the baseline kernels when the number of words was less than 15. We note that the proportion of documents whose number of words



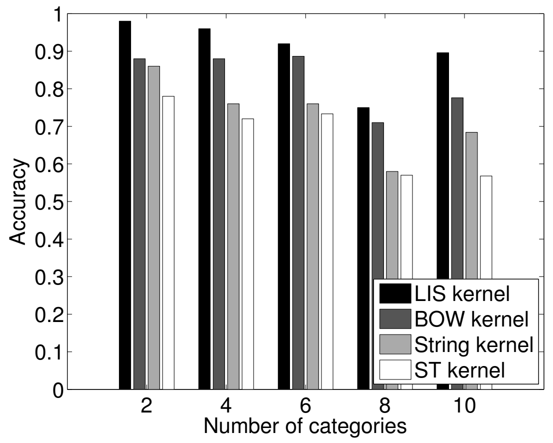
h[]

(a) English dataset

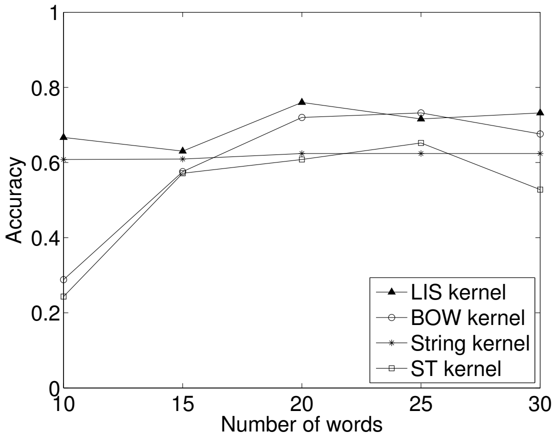


(b) Korean dataset

Fig. 7. Classification performances according to similarity coefficients, λ_W, λ_D, and λ_C when $k_W = k_D = 5$ for the English dataset, and $k_D = 5$ and $k_D = 20$ for the Korean dataset.



(a)



(b)

Fig. 8. Classification performances on the English dataset.

less than 15 was 67.04% in the English datasets. While BOW kernel resulted in the better accuracy than LIS kernel only when the number of words was 25, the other baseline kernels show less accuracy in all the cases considered. This means that the baseline kernels might suffer from a few words in common across documents when the number of words per document is very small. On the contrary, the consideration of semantic annotation for each pattern aiming to address the insufficient number of words in common across documents was effective to calculate the similarity between short-text documents.

For the Korean dataset, the classification performances of LIS kernel and three baseline kernels are captured in Figure 9.

Although String kernel shows better accuracy than LIS kernel when the number of categories and the number of words were 2 and 5, respectively, LIS kernel outperformed the baseline kernels including String kernel in the other cases. We remark that the proportion of documents each of which contains less than 5 words was less than 23%, implying that LIS kernel outperformed the baselines in most cases.

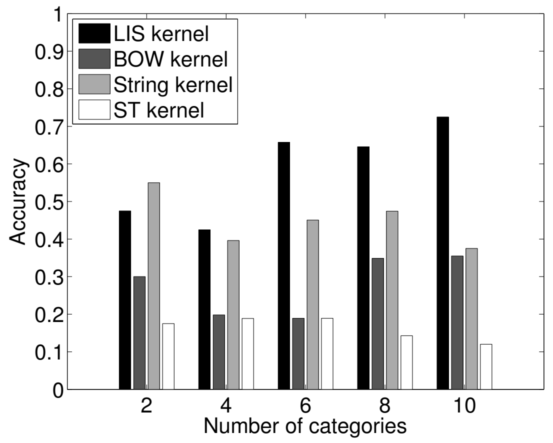
Interestingly, String kernel yielded relatively robust performances across the two languages while the other baseline kernels resulted in the sharply decreased performances in the Korean dataset compared to the English dataset. It is considered that different results might be caused by the different character usages in English and Korean.

Finally, we visualized the document by document similarity matrix obtained by LIS kernel in Figure 10. To visualize the similarity matrix of 200 documents, 20 documents for each category were randomly selected for each dataset. In the figure, darker colored areas represent higher similarity values, and we can say that LIS kernel generally performs on both English and Korean datasets.

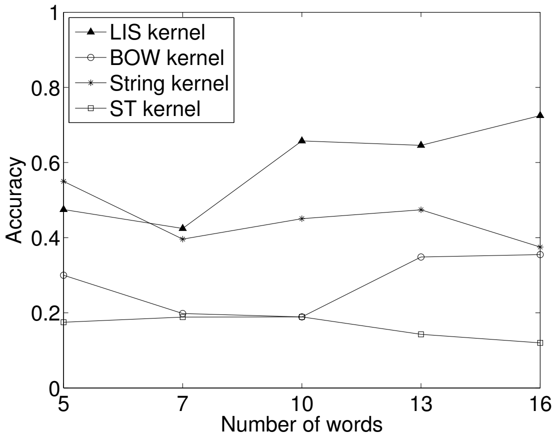
5. Conclusions

In this article, we proposed a novel kernel, called language independent (LIS) kernel based on semantic annotation. LIS kernel was designed to effectively classify short-text documents independently of languages without requiring grammatical tags and lexical databases, which are often regarded as a critical obstacle in many languages. We considered both syntactic and semantic features by utilizing three levels of semantic annotations to address the language dependency problem.

To evaluate the performances of the proposed kernel, two real-world language datasets, Korean and English, were used, and the performances of LIS kernel were compared to the existing methods such as BOW, String, and ST kernels.



(a)

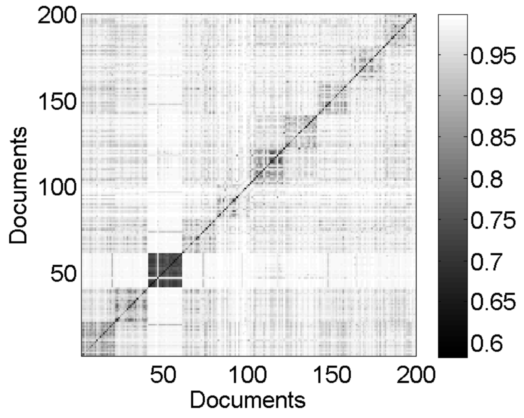


(b)

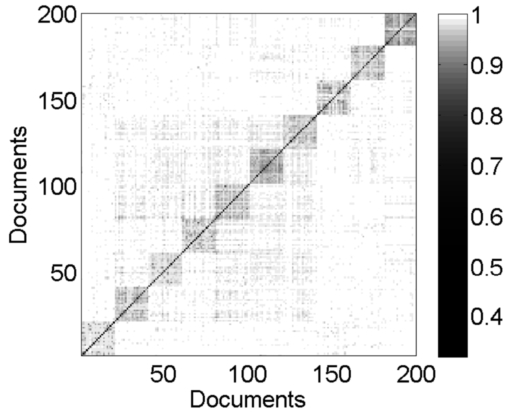
Fig. 9. Classification performances on the Korean dataset.

Experiment results showed that LIS kernel outperformed the others, and it is also quite robust against the number of categories as well as the number of words per document regardless of the considered languages.

The proposed LIS kernel is expected to serve as a tool of significantly improving the performance of short-text classification in various fields by alleviating the language independency problem.



(a) English dataset



(b) Korean dataset

Fig. 10. Similarity matrix of 200 documents obtained by LIS kernel.

Acknowledgements

This research was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea Government (MEST) (Nos. 2012-0002032, 2011-0030814, and 2012R1A6A3A01038452), and partly by Engineering Research Institute at Seoul National University.

References

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1* (pp. 86–90).

Berg, C. (1990). *Positive Definite and Related Functions on Semigroups*. Walter de Gruyter.

Bloehdorn, S., & Moschitti, A. (2007). Combined syntactic and semanitc kernels for text classification. In *Proceedings of the European Conference on IR Research* (pp. 307–318).

Can, F., Kocberber, S., Balcik, E., Kaynak, C., Ocalan, H. C., & Vursavas, O. M. (2008). Information retrieval on turkish texts. *Journal of the American Society for Information Science and Technology*, *59*, 407–421.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*, 27:1–27:27.

Collins, M., & Duffy, N. (2001). Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14* (pp. 625–632). MIT Press.

Cristianini, N., Shawe-Taylor, J., & Lodhi, H. (2002). Latent semantic kernels. *Journal of Intelligent Information Systems*, *18*, 127–152.

Faguo, Z., Fan, Z., Bingru, Y., & Xingang, Y. (2010). Research on short text classification algorithm based on statistics and rules. In *Proceedings of Third International Symposium on Electronic Commerce and Security* (pp. 3–7).

Fellbaum, C. (2010). Wordnet. In *Theory and Applications of Ontology: Computer Applications* (pp. 231–243). Springer.

Fu, J., & Lee, S. (2012). A multi-class svm classification system based on learning methods from indistinguishable chinese official documents. *Expert Systems with Applications*, *39*, 3127 – 3134.

Haussler, D. (1999). *Convolution Kernels on Discrete Structure*. Technical Report UCSC-CRL-99-10 University of California in Santa Cruz, Computer Science Department.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*. Springer.

Klein, D., & Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Lee, D., Yeon, J., Hwang, I., & goo Lee, S. (2010). Kkma : A tool for utilizing sejong corpus based on relational database. *Journal of Korea Information Science Society*, *16*, 1046–1050.

Leong, C. K., Lee, Y. H., & Mak, W. K. (2012). Mining sentiments in sms texts for teaching evaluation. *Expert Systems with Applications*, *39*, 2584 – 2589.

Liu, S., Jiang, X., & Xia, C. (2010). Rss item ranking based on implicit feedback. In *Proceedings of the 7th International Conference on Fuzzy Systems and Knowledge Discovery* (pp. 2507–2512). volume 6.

Liu, X., Rujia, G., & Liufu, S. (2012). Internet news headlines classification method based on the n-gram language model. In *Proceedings of the International Conference on Computer Science and Information Processing* (pp. 826–828).

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, *2*, 419–444.

Luo, Q., Chen, E., & Xiong, H. (2011). A semantic term weighting scheme for text categorization. *Expert Systems with Applications*, *38*, 12708 – 12716.

Mei, Q., Xin, D., Cheng, H., Han, J., & Zhai, C. (2007). Semantic annotation of frequent patterns. *ACM Transactions on Knowledge Discovery from Data*, *1*.

Moschitti, A. (2009). Syntactic and semantic kernels for short text pair categorization. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics* (pp. 576–584). The Association for Computer Linguistics.

Musiał, K., & Kazienko, P. (2011). Social networks on the internet. *World Wide Web*, (pp. 1–42).

Saleh, M. R., Martn-Valdivia, M., Montejo-Rez, A., & Urea-Lpez, L. (2011). Experiments with svm to classify opinions in different domains. *Expert Systems with Applications*, *38*, 14799 – 14804.

Sanchez A, V. D. (2003). Advanced support vector machines and kernel methods. *Neurocomputing*, *55*, 5–20.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Sheth, A., Aleman-Meza, B., Arpinar, I. B., Bertram, C., Warke, Y., Ramakrishanan, C., Halaschek, C., Anyanwu, K., Avant, D., Arpinar, F. S., & Kochut, K. (2005). Semantic association identification and knowledge discovery for national security applications. *Journal of Database Management*, *16*, 33–53.

Siolas, G., & d'Alché Buc, F. (2000). Support vector machines based on a semantic kernel for text categorization. In *Proceedings of the International Joint Conference on Neural Networks* (pp. 205–209).

Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., & Demirbas, M. (2010). Short text classification in twitter to improve information filtering. In *Pro-*

ceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (pp. 841–842).

Taksa, I., Zelikovitz, S., & Spink, A. (2007). Using web search logs to identify query classification terms. *International Journal of Web Information Systems*, *3*, 315–327.

Tomas, D., & Vicedo, J. (2012). Minimally supervised question classification on fine-grained taxonomies. *Knowledge and Information Systems*, (pp. 1–32).

Wang, J., & Oard, D. (2006). Clef-2005 cl-sr at maryland: Document and query expansion using side collections and thesauri. (pp. 800–809). volume 4022 of *Lecture Notes in Computer Science*.

Zhang, D., & Lee, W. S. (2006). Extracting key-substring-group features for text classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 474–483).

Zhao, H. (2010). Matching attributes across overlapping heterogeneous data sources using mutual information. *Journal of Database Management*, *21*, 91–110.