A novel semantic smoothing kernel for text classification with class-based weighting

Berna Altınel[1]
Computer Engineering Department of
Marmara University
Istanbul, Turkey
berna.altınel@marmara.edu.tr

Banu Diri[2]
Computer Engineering Department of
Yıldız Technical University
Istanbul, Turkey
banu@ce.yildiz.edu.tr

Murat Can Ganiz[3]
Computer Engineering Department of
Doğuş University
Istanbul, Turkey
mcganiz@dogus.edu.tr

Abstract*—In this study, we propose a novel methodology to build a semantic smoothing kernel to use with Support Vector Machines (SVM) for text classification. The suggested approach is based on two key concepts; class-based term weights and changing the orthogonality of vector space model. A class-based term weighting methodology is used for transformation of documents from the original space to the feature space. This class-based weighting basically groups terms based on their importance for each class and consequently smooths the representation of documents which changes the orthogonality of the vector space model by introducing class-based dependencies between terms. As a result, on the extreme case, two documents can be seen as similar even if they do not share any terms but their terms are similarly weighted for a particular class. The resulting semantic kernel can directly make use of class information in extracting semantic information between terms, therefore it can be considered as a supervised kernel. For our experimental evaluation, we analyze the performance of the suggested kernel with a large number of experiments on benchmark textual datasets and present results with respect to varying experimental conditions. To the best of our knowledge, this is the first study to use class-based term weights in order to build a supervised semantic kernel for SVM. We compare our results with kernels that are commonly used in SVM such as linear kernel, polynomial kernel and radial basis function (RBF) kernel and with several corpus-based semantic kernels. According to our experimental results the proposed method promisingly improves classification accuracy over linear kernel and several corpus-based semantic kernels in terms of both accuracy and speed.

Keywords—support vector machines; text classification; semantic kernel;semantic smoothing kernel; class-based term weighting.

1. INTRODUCTION

In recent years, with the ever accumulating online information on the Internet and Social Media, text categorization has become one of the key techniques for organizing and handling textual data. Automatically processing these huge amounts of textual data is an essential problem. Text classification can be defined as the utilization of a supervised learning methodology to assign predefined class labels to documents using a model learned from labels of the documents in the training set. An important requirement of efficient and accurate text classification systems is to organize documents into pre-determined categories that contain similar documents. In order to achieve this goal, they depend on similarity or distance measures that compare pairs of text documents similarity. It is also known that vector space representation of textual documents yields high dimensionality and related to this; sparsity. This is especially a problem when there are a large number of category labels but limited amount of training data. It is thus crucial that a good text classification algorithm should scale well with the increasing number of features and classes. Most importantly, words in textual data carry semantic information, i.e., the sense carried by the terms of the documents. Consequently, an ideal text classification algorithm should be able to make use of this semantic information.

Bag-of-words (BOW) feature representation is well accepted as the fundamental approach in the domain of text classification. In BOW approach documents are characterized by the frequencies of individual words or terms and each term represents a dimension in a vector space independent of other terms in the same document [1]. It basically focuses on the frequency of words. The BOW approach over simplifies the representation of terms in documents by ignoring the several different syntactic or semantic relations between terms in natural language, e.g. it treats polysemous words (i.e., words with multiple meanings) as a single entity. For instance the term “bank” may have different meanings like financial institution or a river side based on the context it appears. Additionally, the BOW feature representation maps synonymous words into different components [2]. In principle, as Steinbach et al. [3] investigate, each class of documents has two kinds of vocabulary: one is “core” vocabulary which are intimately associated to the topic of that class, the other type is "general" vocabulary (e.g. stop words) those may have similar distributions on different classes. Therefore, two different documents from different classes may share many general words and will have high similarity based on their BOW representations.

To address the above mentioned weaknesses of BOW model, several methods are proposed in word sense disambiguation, text classification and information retrieval. These methods which enhance the representation of documents can be categorized as domain knowledge-based systems, statistical approaches, hybrid methods, word sequence enhanced systems and linguistic enriched methods. These studies are discussed in related work section.

Another issue in the BOW model is how to weight a term. There are different weighting approaches to assign appropriate weights to the terms to improve the classification performance including binary, term frequency (TF), term frequency-inverse document frequency (TF-IDF) [[4], gain ratio, information gain, odds ratio, [[6],[7]], term frequency-relevance frequency (TF-RF) [8] and term frequency–inverse class frequency (TF-ICF) [9],[10] . These methods are summarized in Section 2.3.

In our previous studies, we introduced a number of corpus-based semantic kernels: Higher-Order Semantic Kernel (HOSK) [11], Higher-Order Term Kernel (HOTK) [12] , Iterative Higher-Order Semantic Kernel (IHOSK) [13] and Class Meaning Kernel (CMK) [60] for SVM. In those studies, we extend the traditional linear kernel (i.e. a dot product between document vectors) for text classification by embedding higher-order relations between terms and documents into the kernel. In the CMK, we employed meaningfulness calculations for terms and used these values to build a semantic kernel. These methods are discussed in Section 2.2.

In this study, we propose a novel approach for building a semantic smoothing kernel which makes use of the class-based term weights to improve the performance of SVM especially for text classification. The proposed approach is called Class Weighting Kernel (CWK). This class-based weighting basically groups terms based on their importance for each class and consequently smooths the representation of documents which changes the orthogonality of the vector space model by introducing class-based dependencies between terms. As a result, on the extreme case, two documents can be seen as similar even if they do not share any terms but their terms are similarly weighted for a particular class.

The suggested approach smooths the terms of a document in BOW representation with a methodology includes the calculation of the terms' discriminating power for each class in the training set. This in turn increases the importance of core or in other words significant terms specific to a particular class while reducing the importance of general terms that have a similar distribution in all classes. Since this approach is used in the transformation phase of a kernel function from input space into a feature space, it considerably reduces the effects of above mentioned disadvantages of BOW. We observe that CWK improves the accuracy of SVM compare to the linear kernel by increasing the importance of class specific concepts, which can be synonymous or very closely related in the context of a class. The CWK uses a semantic smoothing matrix in the transformation of the original space into the feature space. This semantic smoothing mechanism maps the similar documents to nearby positions in the feature space of SVM if they are written using semantically closer sets of terms on the same topic/class.

The first advantage of our suggested solution is the capability of CWK to perform much better than standard kernels in terms of classification accuracy. To demonstrate performance improvements, we conduct several experiments on varied benchmark datasets with several different test environments. According to our experimental results CWK exceeds the performance of linear kernel, is one the state of the art algorithms for text classification as it is mentioned in [14], [15]. Additionally, experimental results show that CWK is superior to both polynomial kernel and RBF kernel. In linear kernel, the inner product between two document vectors is used as kernel function, which utilizes the information about shared term in these two documents. This method can be categorized as a first-order method as its scope or context comprises of a single document only [52]. However, CWK can take advantage of class-based weighting of terms, therefore extending the context from a single document to a class of documents. In this way, semantic relation between two terms is composed of class-based weighting values of these terms for all classes. So if these two terms are significant terms in the same class then the semantic relatedness value between them will be higher.

The second benefit of CWK is about its execution time. We evaluate CWK by comparing it to the traditional kernels as well as the corpus-based kernels of IHOSK, HOSK, HOTK [11][12], [13] and CMK [60] by using several benchmark datasets. The CWK outperforms other corpus-based semantic kernels in many cases in terms of accuracy with less execution time. We provide a brief time complexity comparison analysis of these algorithms in Section 5.

The third advantage of the proposed approach is about its simplicity, flexibility, robustness, and independency of the outside semantic sources such as WordNet. As a result it can be applied to any domain without adjustments or parameter optimizations. To show this wide applicability of our kernel we present results with different experimental settings, such as: (i) several datasets from different domains such as newsgroups postings and movie reviews classified into sentiments, (ii) different training portions of the dataset in order to observe the effect of sparsity, and (iii) varying values of misclassification parameter of the SVM.

The other benefit of CWK is that it also forms a foundation that can easily be combined with other term-based semantic similarity in other words unsupervised semantic similarity measures. It is also easy to take advantage of similarities between terms derived from a semantic source like WordNet or Wikipedia.

The experimental results show significant improvements in the classification accuracy when class-based term weighting calculation is used in SVM, compared to the performance of the two types of previously mentioned baselines; linear kernel and our former semantic kernels like IHOSK and HOTK. To the best of our knowledge, class-dependent weighting values of words are conducted to the transformation phase of SVM for the first time in the bibliography and give significant insights on the semantic smoothing of terms for text classification.

The current work extends beyond our previous studies on semantic kernels in [11][12], [13].The main contributions of this work, which distinguish it from our former works, can be summarized in the following:

- Embedding the weighting values of terms which reflect the effects of terms on classes, into semantic kernel definition to smooth the similarity and the representation of the text documents.
- Building a kernel with the capability of reaching more accurate classification performance in compare to both linear kernel and our previous semantic kernels.
- Desinging a smoothing mechanism in a kernel which has a less execution time than IHOSK and HOTK.
- A detailed comparative evaluation against the linear kernel and the semantic kernels presented in [11][12], [13] shows that the proposed kernel performs favourably against both linear kernel and our previous semantic kernels.

The rest of the paper is organized as follows: In Section 2, we briefly introduce SVM and discuss the related work in the field of term weighting calculations and semantic smoothing kernels, with emphasis to the task of text classification. Section 3 describes and analyzes the suggested kernel for text classification algorithm. Experimental setup is given in Section 4, the corresponding experiment results including some discussion points are given in Section 5. Finally, we conclude this paper in Section 6 with a discussion on probable future extension points of the current work.

2. RELATED WORK

2.1 Support Vector Machines for Classification

The Support Vector Machines (SVM) is a very efficient machine learning algorithm stem from Statistical Learning Theory (SLT). SVM was proposed by Vapnik, Guyon and Boser [16] and analyzed in [17]. The basic goal of SVM is to find the optimal separating hyperplane with the maximal margin between two classes. Compared with traditional machine learning methods, it has many advantages such as finding global optimal solution, getting good generalization performance, having a good robustness and capability to apply kernel trick [17],[61].

An important property of a kernel function that it has to satisfy Mercer's condition as it is mentioned in [18]. A kernel function in SVM can be considered as a kind of similarity function, since it calculates the similarity values of data points, documents in our case, in the transformed space. Consequently, defining an appropriate kernel has the direct effect on finding a better representation of these data points as discussed in [2][19][30]. The popular kernel functions for the document vectors d_p and d_q:

- Linear kernel: $\kappa(d_p, d_q) = d_p d_q$ (1)
- Polynomial kernel: $\kappa(d_p, d_q) = (d_p d_q + 1)^b, b = 1,2...etc.$ (2)
- RBF kernel: $\kappa(d_p, d_q) = \exp(\gamma \|d_p - d_q\|^2)$ (3)

2.2 Semantic Kernel Designs for Text Classification

Linear kernel has a widely usage in the domain of text classification since it is the simplest kernel function. As shown in Eq. (1) the kernel values are calculated based on the inner products of feature vectors of the documents. So a linear kernel captures similarity between two documents as much as the terms they share. This could yields certain problems due to the nature of natural language such as synonymy and polysemy since it is not considering semantic relations between terms. This can be addressed by integrating semantic information between words using semantic kernels as mentioned in [2],[11],[12][13],[20][21][22],[23],[24],[27],[30]].

According to the definition given in [2][16][18][24][27] any function in the following form Eq. (4), is a valid kernel function:

$$\kappa(d_p, d_q) = \langle \bar{\phi}(d_p), \bar{\phi}(d_q) \rangle \quad (4)$$

In Eq. (4), d_p and d_q are input space vectors and $\bar{\phi}$ is a suitable mapping from input space into a feature space.

Like it is mentioned in Section 1, studies enhance the representation of documents can be categorized according to their design principles as:

- Domain knowledge-based systems: These systems use ontology or thesaurus to capture the concepts in the documents and incorporate the domain knowledge of separate terms into the words for representation in textual data. They enhance the representation of terms by taking advantages of semantic relatedness among terms. These include [2][24][25][26][27][28][29][30]. For instance in [30], WordNet [31] is used to calculate semantic similarity between English words. The study in [24] uses super concept declaration with some distance measures between words from WordNet like Wu-Palmer Measure, Inverted Path Length (IPL), Lin Measure and Resnik Measure. A recent study of this kind is [32] that uses HowNet as a Chinese semantic knowledge-base. In [34], the authors present distinct types of approaches to combine the background knowledge in ontologies into the representation of textual data and show the improvement in the classification accuracy. Similar works also can be found in [29] [33].

- Statistical approaches: These systems use statistical analysis depending on the relations of terms in the set of training documents in order to expose latent similarities between them [36]. One of the well-known corpus-based systems is Latent Semantics Analysis (LSA) [35]which partially solves the synonymy problem.
- Hybrid methods: These approaches combine the information gathering from ontology and statistical analysis results from the corpus like in [28]. Also there is a recent survey in [26] about these studies.
- Word sequence enhanced systems: This type of representations treat the words as string sequences. Typically the main idea is based on a word sequence taken out from documents by customary string matching technique. N-gram based representation [37] and similar works in [38] [39], [40] are conventional examples of this kind of systems.
- Linguistic enriched methods: These methods make use of syntactic and lexical rules of phrases in order to extract the noun phrases, terminologies and entities from documents and improve the representation using these linguistic units. For instance in [41] multi-words are used to expand the effectiveness of text-retrieval system. Also Lewis [42] compares the word-based indexing and phrase-base indexing for representation of document categorization

Siolas and d'Alché-Buc in [30] proposes a semantic kernel that is intuitively generated from the semantic relations of English words in WordNet which is a popular network of semantic connections between words. The hierarchies and connections between terms in WordNet are used by the authors [30] calculate semantic relatedness between two words. They benefit from this information to enrich the Gaussian kernel. According to this study, using a semantic similarity metric as a smoothing technique increases the correct classifications.

Bloehdorn et al. [24], uses a semantic kernel with super-concept declaration. Their purpose is to create a kernel function that captures the knowledge of topology that belongs to their super-concept expansion. This kernel function is given in Eq. (5), where *Q* is a semantic smoothing matrix. The *Q* is composed of *P* and P^T which contains super-concept information about the corpus. Their results show that they get remarkable improvement in performance, particularly in situations where the feature representations are highly sparse or little training data exists [24].

$$\kappa(d_p, d_q) = d_p P \ P^T d_q^T \tag{5}$$

Bloehdorn and Moschitti [50] designed a Semantic Syntactic Tree Kernel (SSTK) by combining syntactic dependencies like linguistic structures into a semantic knowledge that is gathered from WordNet. Similarly, in [27] WordNet is used as a resource of semantic background information. Nevertheless, they express that WordNet's coverage is not sufficient and a more extensive background knowledge resource is required. This is also one of the key reasons that other studies aim to use resources with wider coverage such as Wikipedia.

One of those works is [2]. The similarity between two documents in the kernel function designed as in Eq. (5), but in this time *P* is a semantic proximity matrix that is created from Wikipedia. The semantic proximity matrix is composed of three measures:

1) content-based measure which depends on Wikipedia articles' BOW representation,

2) out-link-category-based measure that brings an information related to the out-link categories of two associative articles in Wikipedia,

3) distance measure which is calculated as the length of the shortest path connecting the two categories of two articles belong to, in the Wikipedia's category taxonomy. The authors state that adding semantic knowledge that is extracted from Wikipedia into document representation overcomes some of the shortages of the BOW approach improves the categorization accuracy.

The study in [22] is also used WordNet to build a semantic proximity matrix based on Omiotis [23], which is a knowledge-based measure for computing the relatedness between terms. Nasir et al. incorporated this measure into a TF-IDF weighting approach. They show that their Omiotis-embedded methodology is superior to standard BOW representation. Nasir et al., further broadened their work by taking just top-k semantically related terms and utilizing some evaluation metrics on larger text datasets [28].

Semantic Diffusion Kernel is presented and studied in [21]. Such a kernel is obtained by an exponential transformation on a given kernel matrix as in

$$\kappa(\lambda) = \kappa_0 \exp(\lambda \kappa_0) \tag{6}$$

where κ_0 is the gram or kernel matrix of the corpus in BOW representation and λ is the decay factor . As it is stated in [20] the kernel matrix κ_0 is created by

$$G = D \ D^T \tag{7}$$

where *D* is the term by document of the corpus. In [20][21] it has been demonstrated that $\kappa(\lambda)$ relates to a semantic matrix $\exp(\lambda \frac{G}{2})$ as in Eq. (8):

$$S = \exp(\frac{\lambda}{2}G) = \frac{1}{2}(2I + \lambda G + \frac{\lambda^2 G^2}{2!} + \ldots + \frac{\lambda^\theta G^\theta}{0!} + \ldots) \quad (8)$$

where G is a generator that displays the initial semantic similarities between words and S is the semantic matrix of the exponential of the generator. According to the experiments in [20] the diffusion matrix exploits higher-order co-occurrences to gather latent semantic relationships between terms in the WSD tasks from SensEval.

Zhang et al. [51], focuses on the usage of multi-word for text representation in the task of text classification. In their work they extract multi-word by using the syntactical structure of the noun multi-word phrases. They present two strategies which are called general concept representation and subtopic representation, to represent the documents using extracted multi-word [51]. Their first strategy is based on the usage of general concepts for the representation of documents. The second strategy uses the subtopics of the general concepts of representation. Then they carry out a serious of experiments with SVM in linear kernel and non-linear kernels in order to see the effects of multi-word in a kernel function. They express the benefits of using multi-word with the following aspects [51]: Firstly, using multi-word decreases the number of dimensions. Secondly, acquiring multi word is an easy task. Thirdly, multi-word carries more semantics than individual words. According to their experimental results, their approach with multi-word linear kernel outperforms than standard linear kernel [51].

Data in SVM is typically projected into a high-dimensional space by means of kernels functions. This technique for space transformation can be called global as it is mentioned in [48], since the projection is done over the whole dataset. However, in text classification data generally have complicated class structure since these classes of documents are categorized by a subset of terms. Therefore; data is usually modeled by some subspaces, instead of the entire space as it is discussed in [48]. Each of these subspaces typically includes a class of documents. In other words, a class-dependent projection has more opportunity to explain the document categories [49]. In VSM representation, the document vectors may be presented as $d_1, d_2, \ldots, d_N$, where $d_i \in \Re^D$ and D denotes the number of unique words in the existing model, N stands for the total number of training documents. Table 1 shows a small corpus having 5 different documents, namely d_1, d_2, d_3, d_4, d_5 and three classes namely C_1, C_2, C_3, respectively.. The documents are represented by the terms t_1, t_2, t_3, t_4 and t_5. According to Table 1, C1 can be represented by t_1, t_2, t_3 and t_5 without t_4; class C_2 represented with only t_3, t_4 and t_5; class C_3 is represented by only t_2 and t_3. This shows that the classes of documents are positioned in different subspaces of the entire space. This requires a methodology that describes the different relevance of a term to the documents classes. By inspiring this phenomenon in the study [48], a classifier that is based on a class-dependent projection is presented. They develop a new supervised term weighting algorithm called Classification Using Projection (CUP), which learns class-dependent weighting values of each term from training data and build a class-dependent subspace for each document category. In testing phase, CUP classifies new documents based on a weighted distance value in the individual subspaces. Their experimental results demonstrate that CUP has higher classification accuracy than some commonly used classifiers such as kNN and SVM with linear kernel [48].

Table 1
An example of BOW vector space representation of documents with term frequencies

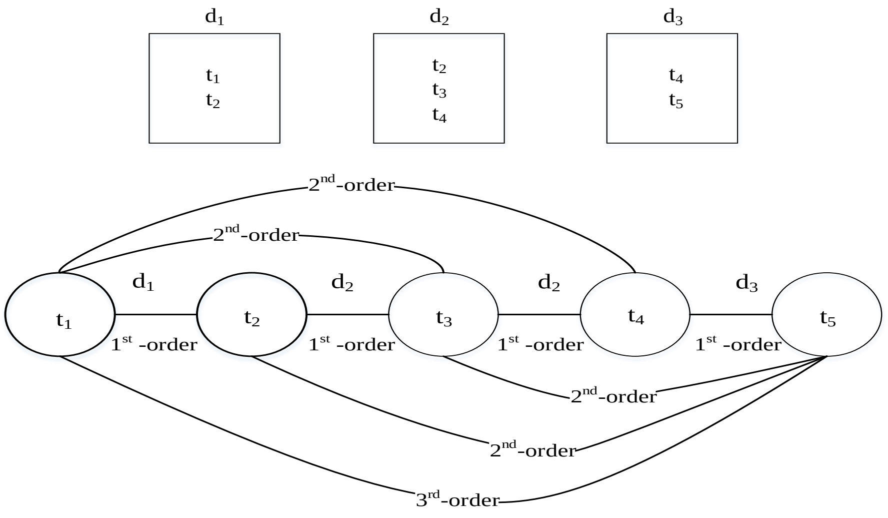
Class	Doc	t_1	t_2	t_3	t_4	t_5
C_1	d_1	8	16	4	0	4
	d_2	16	4	8	0	4
C_2	d_3	0	0	14	14	3
	d_4	0	0	20	2	2
C_3	d_5	0	14	16	0	0

In our previous studies [10][11][12] we proposed semantic kernels for SVM. First three of them [11][12][13] takes the advantages of higher-order paths. There are several systems with higher-order co-occurrences in the field of text classification. One of the most extensive of them is the Latent Semantic Indexing (LSI) algorithm. The work in [19] verified mathematically that performance of LSI has a straight correlation with the higher-order paths, In other words LSI's higher-order paths extract "latent semantics" [19] [52]. Based on these studies, the authors in [[45], [52]constructed a new Bayesian classification framework called Higher-Order Naive Bayes (HONB) which proposes that terms in documents are powerfully connected by such higher-order paths and that they can be exploited for getting better performance for classification. Both HONB [45] and HOS [46] [47] are based on Naïve Bayes.

Benefits of using on higher-order paths between documents [11] and between terms [13] [45], [47] are shown in Fig. 1. There are three documents, d_1, d_2 and d_3 which consist of a set of terms $\{t_1, t_2\}$, $\{t_2, t_3, t_4\}$ and $\{t_4, t_5\}$, respectively. Using an ordinary similarity measure that is based on the number of shared terms (e.g. dot product), the similarity value between documents d_1 and d_3 will be clearly zero since they do not have any common terms. But this kind of measure is inadequate since these two documents have some connections in the scope of the dataset over d_2 [13] as it can be seen in Fig. 1. This supports the idea that in cases, where two documents are written on the same topic using different but semantically closer sets of terms, using higher-order paths between documents, it is possible to obtain a non-zero similarity value between them which is not possible in the BOW representation.

In our previous study in [11]; we suggested a semantic kernel with the name of *HOSK* which makes use of higher-order paths between documents. In HOSK, the simple dot product between the features of the documents gives a first-*order matrix (F)*, where

its second order directly gives second-order matrix *(S)* between documents; which is used as kernel matrix in HOSK's transformation from input space into feature space. The results in [12] show that *HOSK* gains an improvement on accuracy over not only
linear kernel (one of the best performing algorithms for text classification system [14][15] but also polynomial kernel and RBF.



First order term co-occurrence {t_1, t_2}, {t_2, t_3}, {t_3, t_4}, {t_2, t_4}, {t_4, t_5}
Second order term co-occurrence {t_1, t_3}, {t_1, t_4}, {t_2, t_5}, {t_3, t_5}
Third order term co-occurrence {t_1, t_5}

Fig. 1. Graphical demonstration of first-order, second-order and third-order paths between terms through documents. (Adapted from [60])

Following this in [13], (IHOSK) we focused on the higher-order paths between documents and terms together in an iterative form. The semantic relatedness between both documents and terms were inspired from [53], in which the authors formulate an iterative technique namely χ-Sim to learn the similarity matrix between documents using similarity matrix between terms and vice-versa. The document similarity matrix is produced iteratively using *SR* (a similarity matrix between documents) and *SC* (a similarity matrix between terms). Their SR_t and SC_t formulas [13][53]are given as;

$$SR_t = D\ SC_{t-1}\ D^T \bullet NR \text{ with } NR_{i,j} = \frac{1}{|v_i||v_j|} \tag{9}$$

$$SC_t = D^T\ SR_{t-1}\ D \bullet NC \text{ with } NC_{i,j} = \frac{1}{|v_i||v_j|} \tag{10}$$

where *D* is the document by term matrix, D^T is the transpose of *D* matrix, *SR* is the document similarity matrix, *SC* is the word similarity matrix, and *NR* and *NC* are document and word normalization matrices, respectively. Bisson and Hussain claim that they repeat SR_t and SC_t calculations up to a pre-defined number of iterations such as *four* [53]. We tuned this number to *two* depending on our experiments [12]. Term similarity matrix (SC_t) is computed and used in the kernel function as follows:

$$\kappa_{IHOSK}(d_p, d_q) = d_p\ SC_t\ SC_t^T\ d_q^T \tag{11}$$

where κ_{IHOSK} is the kernel value of documents d_p and d_q, respectively.

Experiment results show that the classification performance increases relative to the common traditional kernels used in the SVM such as the linear kernel, polynomial kernel and RBF kernel.

In our study in [12] we consider less complex higher-order paths, HOTK, that is based on the outcomes of higher-order paths between terms only this time. The semantic kernel transformation in HOTK is done using the following equation:

$$\kappa_{HOTK}(d_p, d_q) = d_p \; S \; S^T \; d_q^T \tag{12}$$

where *S* has higher-order co-occurrence relationships between terms in the training-corpus-set only. HOTK is much simpler than IHOSK [13] and also requires less computation time.

In our very recent study in [60] we also propose a novel approach for building a semantic kernel for SVM, which we name Class Meaning Kernel (CMK). The suggested approach smoothes the terms of a document in BOW representation by class-based meaning values of terms. This in turn, increases the importance of significant or in other words meaningful terms for each class while reducing the importance of general terms which are not useful for discriminating the classes. This approach reduces the above mentioned disadvantages of BOW and improves the prediction abilities in comparison with standard linear kernels by increasing the importance of class specific concepts which can be synonymous or closely related in the context of a class. The main novelty of our approach is the use of this class specific information in the smoothing process of the semantic kernel. The meaning values of terms are calculated according to the Helmholtz principle from Gestalt theory [62] [63][64][65] in the context of classes. We conducted several experiments on various document datasets with several different evaluation parameters especially in terms of the training set amount. Our experimental results show that CMK widely outperforms the performance of the other kernels such as linear kernel, polynomial kernel and RBF kernel.

In one of the commonly used textual datasets, we compare the accuracy results of all our algorithms HOSK [11], IHOSK [13], HOTK [12] and CMK [60] with our novel class-based semantic kernel CWK in the Section 5.

2.3 Term weighting methods

As it is mentioned in Section 1, there are different aspects to assign appropriate weights to the terms to improve the classification performance: For example; binary, Term Frequency (TF), Term Frequency-Inverse Document Frequency (TF-IDF) [4][5]and its variants are the traditional methods borrowed from IR field and belong to the unsupervised term weighting methods. Also there are approaches which have proper place in the supervised term weighting category since term weights are calculated according to the category membership information of training documents. One type of them is to weight terms by using feature selection metrics, i.e. gain ratio, information gain (IG), odds ratio and so on, in [6][7]. Another recent approach that improves the terms' discriminating power for text categorization task is Term Frequency-Relevance Frequency (TF-RF) [8]which considers only the frequency of relevant documents (i.e. those which contain this term). Furthermore [43] [44] is inspired from Term Frequency–Inverse Class Frequency TF-ICF [9] [10] and extends the boundaries of traditional weighting method TF-IDF [4] by the contribution of category information of terms in the training set.

TF-IDF[4] is one of the popular term weighting and its formula is given in Eq. (14), where tf_w represents the frequency of the term *w* in the document and IDF is the inverse of the document frequency of the term in the dataset. IDF's formula is also given in Eq. (13) where $|D|$ denotes the number of documents and df_w represents the number of documents which contains term *w*. TF indicates the occurrence of word *w* in document d_i. The TF-IDF has proved extraordinarily robust and difficult to beat, even by much more carefully worked out models and theories [58].

$$IDF(w) = \frac{|D|}{df_w} \tag{13}$$

$$TF - IDF(w, d_i) = tf_w \times \log(IDF(w)) \tag{14}$$

A similar but supervised version of TF-IDF is called TF-ICF whose formula given in Eq. (16) as in [9] [10]. In Eq. (15), $|C|$ represents number of classes and cf_w indicates the number of classes which contain term *w*.

$$ICF(w) = \frac{|C|}{cf_w} \tag{15}$$

$$TF - ICF\left(w, c_j\right) = \sum_{d \in c_j} tf_w \times \log\left(ICF\left(w\right)\right) \tag{16}$$

In [8] proposed a new term weighting method with the idea of simplifying a multi-label classification problem into multiple independent binary classification problems. In their methodology, a chosen category is tagged as the positive category and all the remaining categories in the same dataset are combined together as the negative category. As it is discussed in [8] their consideration is: the more focused a high frequency term is in the positive category than in the negative category, the more contributions it makes in selecting the positive samples from the negative samples. Their term weightings formula is as follows:

$$TF - RF = tf_w \times \log\left(2 + \frac{a}{\max(1, c)}\right) \tag{17}$$

where tf_w is the term frequency of word w, a is the number of documents in the positive category which contain term w and c is the number of documents in the negative category which contain term w. Table 2 demonstrates the difference between the discriminative powers of both IDF and RF. Table 2 lists the IDF and RF values of four terms based on two categories, namely, 00_acq and 03_earn respectively. The first two terms, acquir and stake, are closely related to the theme discussed in category 00_acq while the last two terms, payout and dividend, are closely related to the theme discussed in category 03_earn. However, as the IDF factor disregards the category or label information of the training set. Thus, each of these four terms is weighted equally by the IDF factor even in terms of the two different categories. On the other hand, by using the RF scheme which pays attention to category information, each term is assigned more appropriate weights in terms of different categories [8].

Table 2
Comparison of the weighting value of four features in Category 00_acq and 03_earn (Adapted from [8])

Feature	Category:00_acq		Category:03_earn	
	IDF	RF	IDF	RF
acquir	3.553	4.368	3.553	1.074
stake	4.201	2.975	4.201	1.082
payour	4.999	1	4.999	7.820
dividend	3.567	1.033	3.567	4.408

By being inspired from the idea of both IDF and ICF, a new term weighting method which is designed as a part of a feature extraction algorithm is proposed in [43][44]. According to their approach the effect of a term over a class is calculated as follows:

$$W_{w,c} = \log(tfc_{w,c} + 1) \times \log(\frac{N}{N_w}) \tag{18}$$

where $tfc_{w,c}$ is the total number of frequency that word w in the class c, N is the total number of documents in the corpus and N_w is the total number of documents those contain term w and $W_{w,c}$ is the total weighting value of term w on class c. By using this class-dependent term weighting scheme they develop a feature extraction method for text classification. They carry out some experiments on some benchmark datasets to compare the classification performance of their method with well-known feature extraction algorithms. Their experimental results show that using this feature extraction with a class-dependent term weighting scheme enhances classification performance on most of the classifiers when compared with other feature extraction methods.

3 Class Weighting Kernel (CWK)

In our previous studies [11] [12] [13], [60] we take advantage of higher-order paths and meaning calculations. In those studies we showed that the performance improvements between first-order and higher-order representation of features [11][12][13] and the power of meaning calculations [60]. In this paper we investigate the use of a new type of semantic smoothing kernel for text classification. The main idea behind Class Weighting Kernel (CWK) is to take advantage of the class-based term weighting of terms in the semantic kernel building process by establishing the semantic relations between terms based on their relative weights for classes. This contributes to give more importance on core words of each class during the transformation phase of SVM from input space to feature space. Since corpus-based weighting matrix has extra information related to the terms in compare to BOW representation, this yields revealing semantic similarities between terms and documents by smoothing the similarity and the representation of the text documents. Term weighting calculation used in this study is inspired by [44] which is motivated by TF-RF [8] and TF-ICF [9] [10] as mentioned in Section 2.3. This term-weighting calculation has been applied to a feature extraction domain in previous works [43] [44] as it is discussed in Section 2.3. In these studies, a text document is represented by terms and their class-based weighting values. A term has a more discriminative power on a class if it has higher weighting value for that class. In other words, the more a word occurred in only a specific class the higher it gets a weighting value and conversely the more a word occurred in all classes the less it gets a weighting value. Although, this class-based weighting calculation has been used in feature extraction domain, to the best of our knowledge, our work is the first to apply this technique to a kernel function. The training and testing algorithms for CWK method are shown in

Fig. 22.

The VSM shows a document collection by a term-to-document matrix. In the initial step of our methodology a document d is represented in the VSM with the following BOW approach:

$$\phi : d : \phi(d) = [tf(t_1,d), tf(t_2,d), tf(t_3,d), tf(t_4,d), tf(t_5,d), ..., tf(t_D,d)] \in \Re^D \tag{19}$$

where $tf(t_i,d)$ is the TF of term t_i in document d, and D is the total number of terms in the dictionary of the *corpus*. In above expression $\phi(d)$ represents the document d as a TF vector, respectively. This function however, can be any other mapping from a document to its VSM representation (e.g., TF-IDF).

To enrich the BOW representation with semantic information, we build the semantic relatedness matrix S using class-based term weighting approach mentioned in [43] [44]. Specifically, the i, j element of S quantifies the semantic relatedness between terms t_i and t_j. The class-based weighting calculations and formulas have been described in detail in the previous section. We take benefits of calculated weighting values of terms in the mapping schema of our kernel function as

$$S = WW^T \tag{20}$$

where W is a class-based term weighting matrix that is mentioned in Section 2.3 and calculated with Eq. (18).In our system S is a semantic smoothing matrix to transform documents from input space to feature space. Thus, S is a symmetric term-by-term matrix. Mathematically, the semantically enriched BOW representation of a document d is given as

$$\bar{\phi}(d) = \phi(d)\ S \tag{21}$$

Although the feature space defined above can be directly used in many classification methods; in a text classification case where we have high dimensionality with sparcity, it will be helpful to define the feature space implicitly via the kernel function. As it is mentioned in Section 2.1 and Eq. (4), the kernel function computes the inner product between documents p and q in the feature space. For our case, this can be written as:

$$\kappa_{CWK}(d_p, d_q) = \langle \bar{\phi}(d_p)\ , \bar{\phi}(d_q)^T \rangle = \phi(d_p) SS^T \phi(d_q)^T \tag{22}$$

where $\kappa\ (d_p,\ d_q)$ is the similarity value between documents d_p and d_q , S is the class-based semantic matrix which represents the weighting values of terms according to Eq. (18) for each class by using the documents in our training set. In other words, here S is a semantic proximity matrix of terms and classes.

As it is mentioned in [20], for SVM and other kernel-based approaches the information is stored in Gram matrix or kernel matrix which is given by:

$$G_{p,q} = \kappa_{CWK}(d_p, d_q) \tag{23}$$

Consequently the Gram matrix or kernel matrix are essentially equivalent. By operating one of these matrixes, it is easy to encode the data in a more appropriate way for mining and learning like it is discussed in [20]. Additionally, as it is mentioned in Section 2.1, to be a valid kernel function, the Gram matrix that is formed from the kernel function must satisfy the Mercer's conditions [18]. These conditions are satisfied when the Gram matrix is positive semi-definite. It has been shown in [54] that the matrix G formed by the kernel function Eq. (23) with the outer matrix product SS^T is indeed a positive semi-definite matrix. According to Eq. (18) when in such a case that if a word does not occur in a class, as a weighting value for that class it gets zero as it will be appreciated. After all calculations we get W as a term-by-class matrix which includes the weighting values of terms in all classes of the corpus. We observe that these weighting values reflect the importance of those words in order to distinguish the classes. Indeed after calculations, terms semantically close to the theme discussed in the documents of that class, gain the highest weighting values in the range. In other words semantically related terms of that class, i.e. "core" words like it is mentioned in [3], gain importance while semantically isolated terms, i.e. "general" words, loose their importance. So terms are ranked based on their importance.

Consequently we argue that S which is based on W and indirectly is built by using Eq. (18) performs a similar kind of semantic smoothing as in Eq. (22). For instance, if the word "data" is highly present while the words "information" and "knowledge" are less, the application of semantic smoothing will increase the values of the last two terms because "data", "information" and "knowledge" are strongly related concepts [60]. The new representation of the documents is richer than the standard representation with TF-IDF since; supplementary statistical information is directly calculated from our training corpus and embedded into the kernel function. In other words transformations in Eq. (22) smooth the simple term vector representation using semantic ranking while passing from the original input space to a feature space through kernel transformation functions $\bar{\phi}(d_p)$ and $\bar{\phi}(d_q)$ for the documents d_p and d_q ,respectively as in [60]. As it is explicitly mentioned in [55], the presence of S in Eq. (22), changes the orthogonality of the document vectors, as this mapping introduces term dependence. Documents can be seen as similar even if they do not have any common terms by eliminating orthogonality.

We also observe that the class-based weighting calculation degrades stop words by assigning them very small weights similar to the [60]. Let us consider the following two cases, which are represented in similar to [60]. According to , t_1 and t_2 are occurred in one or more documents of c_1, not in remaining classes; c_2, c_3 and c_4 while t_3 and t_4 are occurred in one or more documents of c_3, not in remaining classes; c_1, c_2 and c_4, respectively. In other words t_1 and t_2 are significant words of the theme discussed in c_1, while t_3 and t_4 are significant terms of the topic discussed in c_3; getting high weighting values according to Eq. (18); since the number of documents that term w occurs in the entire corpus that is N_w is inversely proportional to the weighting value. Also the total frequencies of t_1 and t_2 in the documents c_1 and the frequencies of t_3 and t_4 in the documents c_3 are directly proportional to the weighting values of those terms. According to Eq. (18), the more a word occurred in only a specific class the higher it gets a weighting value and conversely the more a word occurred in all classes the less it gets a weighting value. This statement can also be represented with , since t_1 and t_2 occurred in only c_1 and t_3 and t_4 occurred in only c_3 while t_5 and t_6 are occurred in every classes of the corpus. It is highly possible that these two words, t_5 and t_6, are in the type of “general” words since they are seen in every class of the corpus.

Table 3
Term frequencies on different classes [60]

	c_1	**c_2**	**c_3**	**c_4**
t_1	1	0	0	0
t_2	1	0	0	0
t_3	0	0	1	0
t_4	0	0	1	0
t_5	1	1	1	1
t_6	1	1	1	1

```
Module Calculating Weighting Values of Words
Input
    Training set
Output
    Weighting Matrix $\overline{W}_{w,k}$ for each word $t_w$ and class $c_k$
Local variables
    $tfc_{w,k}$ : total frequency of word $w$ in class $k$
    $tf_{w,d}$ : total frequency of word $w$ in document $d$
    $N$ : total number of documents in the training set
    $\vec{N}_w$ : vector shows the total number of documents those contain word $w$
    $\overline{W}_{w,k}$ : matrix shows the weighting value of word $w$ in class $k$
begin
    for each word $w$
        for each document $d_i$ contain word $w$ in the training set
            $\vec{N}_w = \vec{N}_w + 1$        //Increment the number of documents contain word $w$
        end for
    end for
    for each word $w$
        for each document $d_i$ in class $k$
            $tfc_{w,k} = tfc_{w,k} + tf_{w,d}$        //Increment the class frequency of word $w$
        end for
        $\overline{W}_{w,k} = (log(tfc_{w,k}) +1)*(log (N/ \vec{N}_w))$        //Calculate the weighting value of the word $w$ in class $k$
    end for
end
Module Training        //Building semantic smoothing kernel
Input
    Training set
Output
    $\overline{G}_{d_p,d_q}$ : Gram matrix shows the kernel value between documents $d_p$ and $d_q$
Local variables
    $\overline{W}_{w,k}$ : Weighting matrix shows the weighting value of word $w$ in class $k$
    $\overline{S}_{w_i,w_j}$ : Semantic smoothing matrix shows the relatedness between words $w_i$ and $w_j$
begin
    $\overline{S}_{w_i,w_j} = \overline{W}_{w,k} (\overline{W}_{w,k})^T$        //Building semantic smoothing matrix
    for each document $d_p$ in the training set
        for each document $d_q$ in the training set
            $\overline{G}_{d_p,d_q} = d_p S S^T d_q$        //Calculating the kernel value between documents $d_p$ and $d_q$
        end for
    end for
end
Module Testing   //Calculating the kernel value between training and testing documents and classification
Input
    Training set and Testing set
Output
    $\overline{G}_{d_p,d_q}$ : Gram matrix shows the kernel value between documents $d_t$ and $d_p$
Local variables
    $\overline{S}_{w_i,w_j}$ : Semantic smoothing matrix shows the relatedness between words $w_i$ and $w_j$
begin
    for each document $d_t$ in the testing set
        for each document $d_p$ in the training set
            $\overline{G}_{d_t,d_p} = d_t S S^T d_p$        //Calculating the kernel value between documents $d_t$ and $d_p$
        end for
    end for
    for each document $d_t$ in the testing set
        return the estimated class label        //Classification
    end for
end
```

Fig. 2. Training and testing algorithms for CWK

4 EXPERIMENT SETUP

We embedded our kernel function into the implementation of the SVM algorithm in WEKA [56][57]. In order to see the performance of CWK on text classification, we conducted a series of experiments on several benchmark textual datasets which are shown in Table 4. Our first dataset IMDB[1] is a collection of movie reviews. It contains 2,000 reviews with two types of labels (i.e. positive and negative) about several movies in IMDB. The number and distribution of labels are balanced in both training and test sets that we used in our experiments. The second type of datasets we use are the variants of popular 20 Newsgroup[2] dataset. This data set is a collection of approximately 20,000 newsgroup documents, divided evenly across 20 different newsgroups and commonly used for text classification and text clustering. We used four basic subgroups; ”SCIENCE”, “POLITICS”, “COMP”, and ”RELIGION from the 20 Newsgroup dataset. The documents are evenly distributed to the classes including 500 documents per class. The sixth dataset we use is the mini-newsgroups[3] dataset which has 20 classes and also has a balanced class distribution. This is a subset of the 20 Newsgroup[2] dataset, too. Basic properties of these datasets are represented in Table 4.

We apply stemming and stopword filtering as we did in our previous studies [[11][12][13][60]]. Furthermore, we filter rare terms which occur in less than three documents. We also apply attribute selection and select the most informative 2000 terms using Information Gain as described [11], [12][13][45][46][47][52].

Table 4
Properties of datasets [60]

Dataset	#classes	#instances	#features
IMDB	2	2,000	16,679
20News-POLITICS	3	1,500	2,478
20NewsGroup-SCIENCE	4	2,000	2,225
20News-RELIGION	4	1,500	2,125
20News-COMP	5	2,500	2,478
Mini-NewsGroups	20	2,000	12,112

In order to observe the behaviors of our semantic kernel under different training set size conditions, we use the following percentage values for training set size: 5%, 10%, 30%, 50%, 70%, 80% and 90%. Remaining documents are used for testing. This is essential since we expect that the advantage of using semantic kernels should be more observable when there is inadequate labeled data.

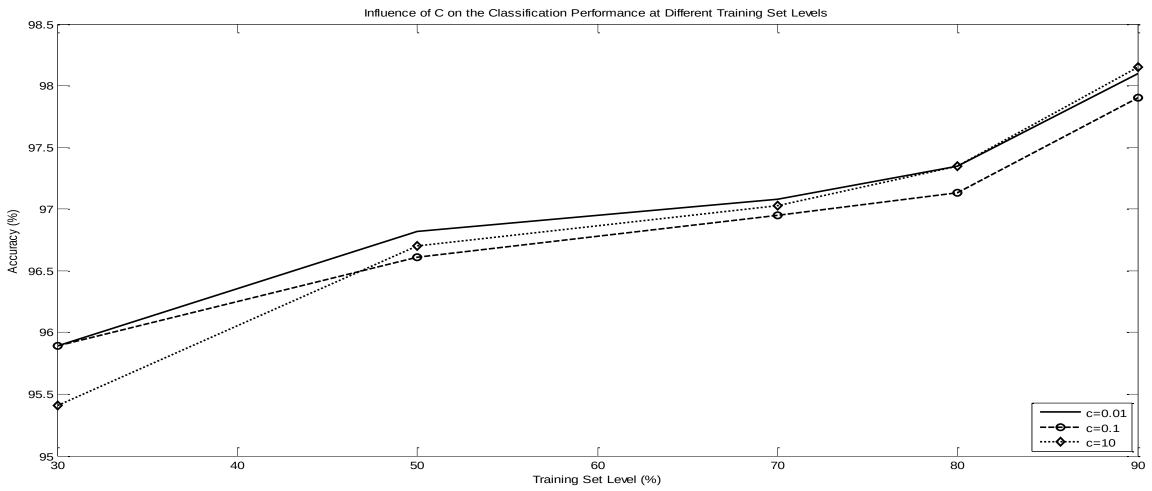


Fig. 3. Influence of *C* on the classification performance at different training set levels on SCIENCE dataset

[1] http://www.imdb.com/interfaces

[2] http://www.cs.cmu.edu/~textlearning

[3] http://archive.ics.uci.edu/ml/

One of the main parameters of SMO [57] algorithm is the misclassification cost (C) parameter. We conducted a series of optimization experiments on all of our datasets with the values of {10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}}. For all the training set percentages we selected the best performing one. Fig. 3 shows the influence of the misclassification parameter *C* on the classification performance (%) between training levels.

30% and 90% on SCIENCE dataset. The accuracy of CWK are equal with C parameters 0.01 and 0.1 at training set percentage 30% as it is shown in Fig. 3. Starting from training set percentage 30% the accuracy with C parameter 0.01 is better than the accuracy with C parameters both 0.1 and 10 until training set percentage 80%. At training level 80% accuracies with C parameters 0.01
and 10 are equal. Also at training set percentage 90% the accuracies with C parameters 0.01 and 10 are close to each other as it can be observed from Fig. 3. The varying accuracy results with different C parameters could probably because of overfitting and underfitting as it is mentioned in [18].

We run the algorithms on 10 random splits for each of the training set ratios with their optimized C values, then we report the average of these 10 results as in [12][13] which is a more comprehensive way of well-known n-fold cross validation which splits the data into n sets and train on n-1 of them while the remaining used as test set.

The main evaluation metric in our experiments is the accuracy and in the results tables we also provide standard deviations as in our previous studies [11][12][13][60].

In order to highlight the performance differences between baseline algorithms and our approach we report performance gain calculated using the following equation;

$$Gain_{CWK} = \frac{(P_{CWK} - P_x)}{P_x} \tag{24}$$

where P_{CWK} is the accuracy of SMO with CWK and P_x stands for the accuracy result of the other kernel. The experimental results are presented in Table 5 to 10. These tables include training set percentage (TS), the accuracy results of linear kernel, IHOSK, HOTK, CMK and CWK. Also the “Gain” columns in the corresponding results tables demonstrate the (%) gain of CWK over linear kernel calculated as in Eq. (24). Moreover, Students t-Tests for statistical significance are provided. We use $\alpha = 0.05$ significance level which is a commonly used level. In the training sets, where CWK significantly differs over linear kernel based on Students t-Tests, we indicate this with “*”. Furthermore we also provide the term coverage ratio by;

$$TermCoverage = \frac{n}{N} \times 100 \tag{25}$$

where *n* is the number of different terms seen in the documents of training set percentages and *N* is the total number of different terms in our corpus; respectively. We observe a reasonable relevance between the accuracy differences and term coverage ratios while passing from one training set percentage to another, which will be discussed in the following section in a detailed way.

5 Experimental Results and Discussion

CWK outperforms our baseline kernel at all training set percentages also making a significant difference based on Students t-Tests results on SCIENCE dataset. This can be observed from Table 5. The performance gain is specifically obvious at training set percentages 5%, 10%, 30% and 50%. For instance at training set percentages 5%, 10% , 30% and 50% the accuracies of *CWK* are 84.31%, 90.94%, 95.89% and 96.82% while the accuracies of linear kernel are 71.44%, 77.97%, 86.73 and 88.94%; respectively. *CWK* also has better performance than our previous semantic kernels IHOSK, and HOTK at all training set percentages as shown in Table 5. Also it should be noted that, CWK is superior to our recent study CMK at training set percentages 5%, 10%, 30%, 50% and 70%. Furthermore there is another point which deserves attention is that by using only 5% of the training set the performance gain of CWK over linear kernel is 18.02%, which is of great importance since usually it is difficult and expensive to obtain labeled data in real world applications. Additionally according to Table 5 we can conclude that the performance differences of CWK while passing from one training set percentage to another are compatible with the term coverage ratios at those training set percentages. For instance at training set percentage 10%, term coverage jumps to 82.28% from its previous value at 5% that is 63.99%. Similar behavior can be observed at performance of CWK while going through 5% training set percentage to 10% training set percentage; where it generates the accuracies 84.31% and 90.94%; respectively. This means an accuracy change of 6.63% between 5% and 10% training set percentages. Also, in all training set percentages CWK has an absolute superiority than both polynomial kernel and RBF on SCIENCE dataset. Actually this superiority on polynomial and RBF remains the same on all datasets in this study, but because of space limitation we cannot provide the results of polynomial kernel and RBF in the experiment results tables.

Additional to CWK, that is calculated with Eq. (22) we also built a second-order version of CWK with the name Second-Order Class Weighting Kernel (SO-CWK) with the following equation:

$$\bar{\phi}(d) = \phi(d)\ SS \tag{26}$$

$$\kappa_{SO-CWK}\left(d_p, d_q\right) = \bar{\phi}\left(d_p\right)\ \bar{\phi}\left(d_q\right)^T = \phi(d_p)\ SS\ (SS)^T \phi(d_q)^T \tag{27}$$

where $\bar{\phi}(d_p)$ and $\bar{\phi}(d_q)$ are transformation functions of kernel from input space into feature space for the documents d_p and d_q, respectively.

Table 5
Accuracy of different HO kernels on SCIENCE dataset with varying training set percentages

TS%	Linear	Polynomial	RBF	IHOSK	HOTK	CMK	CWK	Gain	Term Coverage
5	71.44±4.30	45.65±3.23	49.16±3.78	84.15±2.87	76.63±2.67	64.51±4.86	**84.31±2.77**	18.02*	63.99
10	77.97±3.73	55.77±4.73	51.72±4.64	90.37±0.81	82.47±2.02	82.19±3.58	**90.94±1.72**	16.63*	82.28
30	86.73±1.32	70.34±2.43	59.19±1.03	94.31±1.09	89.24±0.74	95.07±0.87	**95.89±0.51**	10.56*	98.01
50	88.94±1.16	76.42±0.99	63.60±1.80	94.97±0.90	90.84±1.12	96.71±0.61	**96.82±0.3**	8.86*	99.90
70	90.58±0.93	79.57±2.00	66.82±1.97	95.35±0.88	92.06±1.28	**97.12±0.59**	97.08±0.68	7.18*	99.99
80	91.33±1.41	81.60±2.13	68.15±1.78	96.23±1.19	93.38±1.43	**97.60±0.66**	97.35±0.56	6.59*	100.00
90	91.40±1.56	81.40±2.58	68.45±3.06	96.85±1.70	94.20±1.36	97.75±0.89	**98.20±0.71**	7.44*	100.00

We also recorded and compared the total kernel computation time of our previous semantic kernels IHOSK and HOTK and CWK. All the experiments presented in this paper are carried on our experiment framework, Turkuaz, which directly uses WEKA [56][57] on a computer with two Intel(R) Xeon(R) CPUs at 2.66 GHz with 64 GB of memory. Our semantic kernel's computation time on each dataset is recorded in terms of seconds and they are proportionally converted into time units. According to this conversion, for instance on SCIENCE dataset; IHOSK [13], SO-CWK, HOTK [12] and CWK estimates the following time units in order;100, 60, 56 and 40, respectively which is shown in Fig. 4.

These values are not surprising since the complexity and running time analysis of them supports this order as we show in the following. In IHOSK [13], there is an iterative similarity calculation between documents and terms, which completes totally in 4 steps including corresponding matrix calculations as in shown in Eq. (9) and Eq. (10). As it is discussed in [53] producing the similarity matrix of terms SC_t $(n \times n)$ has overall complexity $O(tn^3)$ where t is the number of iterations and n is the number of training terms. Similarly, calculating the similarity matrix of documents SR_t $(m \times m)$ has $O(tm^3)$ where m is the number of training documents. Since both matrices need to be computed iteratively the overall complexity is bounded by the matrix multiplications of term and document similarity matrices. In our experiments only two iterations are performed (t=2).. On the other hand, HOTK [12] has $O(n^3)$ complexity where n is the number of terms as noted in Eq. (12). Although both IHOSK and HOTK is bounded by the complexity of matrix multiplications, the IHOSK involves much more matrix multiplications due to the iterative computation of both term and document similarity matrices and consequently runs much longer than HOTK in practice. The CWK is also bounded by the complexity of matrix multiplication. However, the matrix is much smaller; $(n \times c)$, where n is the number of term and c is the number of classes. In addition, the generation of this term by class matrix is also very fast with $O(logn)$ complexity. These differences make CWK faster than HOTK as can be seen in Fig. 4. On the other hand the, SO-CWK includes an additional matrix multiplication of a term by term matrix, therefore it runs slightly longer than HOTK as shown in Fig. 44. Since the complexity of IHOSK is much higher than both HOTK and the proposed work of the CWK, it is not practical to apply IHOSK on large datasets.

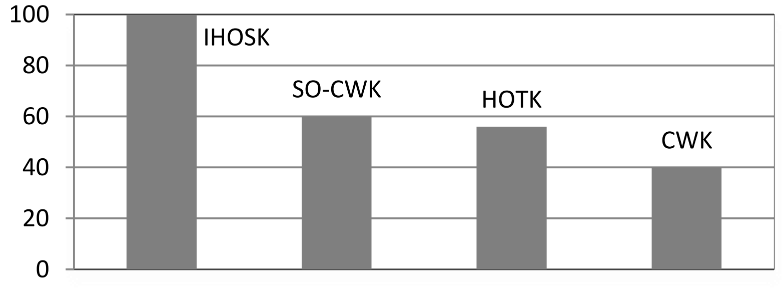


Fig. 4. The total kernel computation time units of IHOSK, SO-CWK, HOTK and CWK on SCIENCE dataset at 30% training set percentage

We also compare CWK with TF-ICF on SCIENCE dataset. The formulation of TF-ICF is given in Eq. (16). TF-ICF is a supervised approach as it is mentioned in Section 2.3. The results are shown in Fig. 5. According to Fig. 5, CWK has much better performance than TF-ICF in all training set percentages.

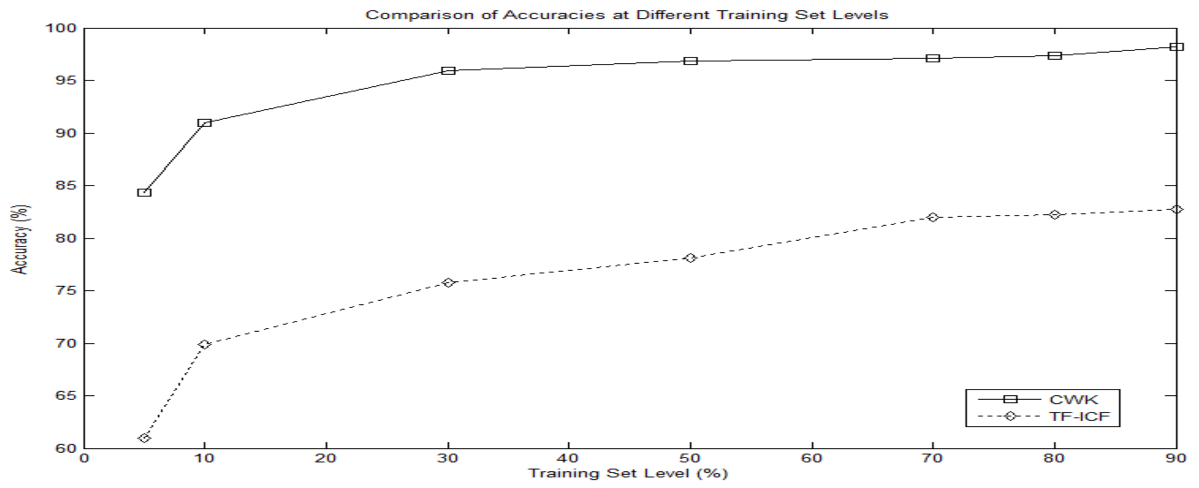


Fig. 5. Comparison of the accuracies of TF-ICF and CWK at different training set percentages on SCIENCE dataset

According to our experiments, CWK demonstrates a notable performance gain on IMDB dataset, which can be seen in Table 6. At 30% training set level, the performance of CWK is 89.66% while the performance of linear kernel is only 85.57%. It is also very promising to see that CWK is superior to both linear kernel and our previous algorithms IHOSK [13] and HOTK [12] throughout all training set percentages.

Table 6
Accuracy of different kernels on IMDB dataset with varying training set percentages

TS%	Linear	IHOSK	HOTK	CWK	Gain	Term Coverage
5	76.85±1.31	76.98±1.14	74.21±0.24	**77.62±2.45**	1.00	48.00
10	82.99±1.76	82.55±2.32	82.23±0.42	**84.32±1.19**	1.60	61.51
30	85.57±1.65	87.16±1.64	85.63±1.69	**89.66±0.53**	4.78	86.35
50	88.46±1.89	89.40±1.91	87.20±0.33	**91.48±0.69**	3.41	95.91
70	89.93±1.18	91.31±0.87	90.41±0.55	**92.75±1.05**	3.14	99.17
80	90.65±1.09	92.38±1.43	91.37±0.98	**92.73±1.09**	2.29	99.71
90	91.75±1.14	92.63±1.19	91.59±0.27	**93.60±1.52**	2.02	99.98

Table 7 denotes experiment results on POLITICS dataset. In this dataset, again CWK's performance is better than linear kernel's at all training set percentages. Similarly, CWK performs better than both IHOSK and HOTK at all training set percentages. Additionally, CWK statistically significantly outperforms our baseline kernel at 5% training set level based on Students t-Tests.

Table 7
Accuracy of different kernels on POLITICS dataset with varying training set percentages

TS%	Linear	IHOSK	HOTK	CWK	Gain	Term Coverage
5	79.01±2.65	82.27±4.60	80.72±1.56	**83.49±4.16**	5.67*	58.60
10	84.69±1.24	88.61±2.10	84.89±2.15	**88.73±2.29**	4.77	75.02
30	92.04±1.06	93.61±1.08	88.31±1.22	**94.90±0.97**	3.11	96.37
50	93.73±0.57	93.55±3.58	90.29±0.79	**96.15±0.80**	2.58	99.43
70	94.55±1.21	93.24±3.08	90.15±1.15	**95.87±0.90**	1.40	99.97
80	94.03±0.91	95.30±1.82	92.50±1.60	**96.80±1.09**	2.95	100.00
90	94.86±1.26	95.80±2.28	92.46±2.01	**96.27±1.97**	1.49	100.00

For COMP dataset, CWK significantly outperforms linear kernel at all training set percentages similar to the situation in SCIENCE dataset as shown in Table 8. The highest gain of CWK over linear kernel on this dataset is at 5% training set level with 18.52% gain. Overall, the gains of CWK over linear kernel are usually much larger in COMP dataset compare to the POLITICS. This may due to the larger number of classes in COMP dataset.

Table 8
Accuracy of different kernels on COMP dataset with varying training set percentages

TS%	Linear	IHOSK	HOTK	CWK	Gain	Term Coverage
5	56.75±4.72	**68.12±1.04**	60.22±3.00	67.26±2.53	18.52*	48.26
10	65.45±2.77	72.71±0.43	66.70±1.14	**76.60±1.53**	17.04*	65.19
30	75.38±2.12	78.71±0.04	75.97±1.04	**84.67±0.58**	12.32*	91.51
50	77.89±1.60	82.18±1.13	78.68±0.71	**87.09±0.77**	11.81*	98.92
70	79.63±1.59	84.67±2.83	80.97±1.18	**87.63±1.53**	10.05*	99.83
80	79.00±2.25	85.81±0.54	81.58±1.85	**88.10±0.96**	11.52*	99.98
90	81.40±2.47	85.96±0.69	81.32±1.46	**87.88±2.56**	7.96*	100.00

Experiment results on RELIGION dataset are presented in Table 9. These results show that CWK has superiority over other kernels starting from 10% training set level. For instance at training set percentage 30% CWK's gain over linear kernel is 7.55%. Also in training set percentage 30% CWK shows a significant improvement over linear kernel.

Table 10 presents the experiment results on mini-newsgroups dataset. According to these results CWK outputs better accuracy results than linear kernel at all training set percentages. Furthermore, it should be noted that at 5% training set level, the gain of CWK over linear kernel is 21.55% which is a very important advantage on the classification accuracy given the very limited labeled information. On the other hand, at training percentages 50%, 70%, 80% and 90% HOTK generates higher accuracy than CWK. This can be explained with the capability of HOTK to capture latent relations between terms with its higher-order approach as explained in Section 2.2.

Table 9
Accuracy of different kernels on RELIGION dataset with varying training set percentages

TS%	Linear	IHOSK	HOTK	CWK	Gain	Term Coverage
5	74.73±2.47	**77.73±2.47**	65.33±1.70	75.32±2.87	0.79	41.80
10	80.98±2.69	81.19±1.92	72.10±1.95	**82.63±1.97**	2.04	59.03
30	83.87±0.78	84.85±1.84	83.50±1.58	**90.20±1.08**	7.55*	88.18
50	88.39±0.93	88.96±2.30	86.19±1.35	**92.41±0.44**	4.55	96.16
70	89.68±1.41	90.62±1.18	87.26±0.31	**92.62±0.99**	3.28	99.37
80	90.70±1.12	91.00±0.20	88.90±0.24	**93.17±1.21**	2.72	99.80
90	91.65±1.63	91.70±1.73	89.00±2.37	**93.20±1.66**	1.69	99.99

Table 10
Accuracy of different kernels on MINI-NEWSGROUP dataset with varying training set percentages

TS%	Linear	IHOSK	HOTK	CWK	Gain	Term Coverage
5	52.38±5.53	61.29±1.03	49.69±5.64	**63.67±3.31**	21.55*	34.90
10	59.85±3.88	64.15±0.54	66.24±3.81	**71.66±1.22**	19.73*	50.08
30	72.84±3.56	75.51±0.31	81.82±2.04	**81.93±1.13**	12.48*	76.16
50	78.87±2.94	79.24±0.31	**85.54±1.20**	84.44±1.14	7.06*	87.65
70	80.05±1.96	79.73±0.45	**87.28±1.13**	84.85±1.56	6.00*	94.27
80	82.63±1.36	83.05±0.58	**88.15±1.58**	86.43±1.16	4.60	96.22
90	84.65±2.48	85.38±1.28	**88.10±2.80**	85.05±3.93	0.47	98.55

Since some of the datasets used in this study are also used in other studies such as [[45],[59]] we have the opportunity to compare our results with them. The first algorithm we compare our results is Higher-Order SVM (HOSVM) [45]. For instance at 30% training set level of COMP dataset; 75.38%, 78.71%, 75.97% and 84.67% accuracies are obtained by linear kernel, IHSOK, HOTK and CWK respectively as mentioned above. On the same training set level, HOSVM achieves 78% accuracy according to the Fig. 2(d) in [45]. This comparison shows CWK outperforms HOSVM by approximately 8.55% gain. Actually CWK's superiority on HOSVM can also be seen on other datasets such as RELIGION, SCIENCE and POLITICS. For instance at 30% training set level on POLITICS dataset while HOSVM' performance is about 91%, where CWK reaches 94.9% accuracy, which corresponds to 4.29% gain. Very similar picture can be seen at a higher training set level such as 50%. For example, the accuracy values of 88.94%, 92%, 94.97%, 90.84%, 96.82% are achieved by linear kernel, HOSVM, IHSOK, HOTK and CWK respectively on SCIENCE dataset at this level.

Moreover, we also have the chance to compare our results with the study in [59]. Harish et al. proposes a text classification algorithm in [59] which uses B-Tree and preserves the term sequence with a data structure called Status Matrix. One of the datasets they use is mini-newsgroups dataset. They do not apply any attribute selection technique such as IG in their preprocessing phase. For instance at training set percentage 50% on mini-newsgroups dataset; 78.87%, 79.24%, 85.54% and 84.44% accuracies are achieved by linear kernel, IHSOK, HOTK and CWK respectively as shown in Table 10. On the same training set percentage, the maximum accuracy gathered by the study in [59] is 68.95. According to this comparison we observe that CWK outperforms the algorithm in [59] by 15.49%.

6 CONCLUSIONS AND FUTURE WORK

We introduce a new semantic kernel for SVM called Class Weighting Kernel (CWK). CWK is based on weighting the values of terms in the context of classes according to [43][44]. CWK smooths the terms of a document in bag-of-words (BOW) representation by making use of the terms' discriminating power for each class in the training set. This in turn increases the importance of significant or in other words, core terms for each class while reducing the importance of general terms that exist in all classes. Since this method is used in the transformation phase of a kernel function (from input space into a feature space), it reduces the effects of the several disadvantages of the BOW representation which is discussed in Section 1. We demonstrate that CWK considerably increase the accuracy of SVM compare to the linear kernel by assigning more weight to class specific terms which can be synonymous or very closely related in the context of a class. In other words, CWK uses a class weighting based semantic smoothing matrix during the transformation from the original space into the feature space of the kernel function. This semantic smoothing mechanism map the similar documents to nearby positions in the feature space of SVM even if they are written using different but semantically closer sets of terms in the context of classes. In our semantic kernel setting, the document term vectors are smoothed based on weighting values of terms in the context of classes. As a result it can be considered as a supervised semantic kernel which directly makes use of class information.

Our experimental results show the promise of CWK as a semantic kernel for SVM in the text classification domain. CWK performs better than commonly used kernels in the literature such as linear kernel, polynomial kernel and RBF. The CWK also demonstrates better accuracies than other corpus-based semantic kernels such as IHOSK [13] and HOTK [12], in most of the datasets we used. According to our experimental results, CWK outperforms our baseline kernel at all training set percentages also making a significant difference based on Students t-Tests results on both SCIENCE and COMP datasets. Furthermore CWK gives higher accuracies than our baseline linear kernel at all of the training set percentages for all of the datasets we used in our experiments. Additionally, on SCIENCE dataset by using only 5% of the training set, the performance gain of CWK over linear kernel is 18.02%, which is of great importance since usually it is difficult and expensive to obtain labeled data in real world applications. A very similar situation is occurred for mini-newsgroups dataset at again 5% training set level, the performance gain of CWK over linear kernel is 21.55%. Moreover it should be noted that CWK has the capability of reaching more accurate classification performance in compare to both linear kernel and our previous semantic kernels with less execution time than both IHOSK and HOTK. We also believe that CWK forms a foundation that is open to several improvements. For instance, the CWK can easily be combined with other semantic kernels which smooth the document term vectors using term to term semantic relations such as the ones using WordNet or Wikipedia.

As future work, we would like to analyze and shed light on how our approach implicitly captures semantic information in the context of a class when calculating the similarity between two documents. We also plan to implement different class-based document or term similarities in supervised classification and compare their results to the results of the CWK.

ACKNOWLEDGMENT

This work is supported in part by The Scientific and Technological Research Council of Turkey (TÜBİTAK) grant number 111E239. Points of view in this document are those of the authors and do not necessarily represent the official position or policies of the TÜBİTAK.

REFERENCES

[1] Salton, G., Yang, C.S., 1973. On the Specification Of Term Values In Automatic Indexing. Journal of Documentation 29 (4) ,pp. 11–21.

[2] Wang, P. , Domeniconi, C., 2008. Building Semantic Kernels For Text Classification Using Wikipedia. In Proceeding of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 713-721.

[3] Steinbach, M. Karypis, G., Kumar, V., 2000. A Comparison Of Document Clustering Techniques. In: Proceedings of the KDD Workshop on Text Mining.

[4] Jones, K. S., 1972. A Statistical Interpretation Of Term Specificity And Its Application In Retrieval. Journal of documentation 28.1: 11-21.

[5] Salton, G., and Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. 24(5):pp. 513–523.

[6] Debole, F., Sebastiani, F., 2003. Supervised term weighting for automated text categorization, in SAC '03: Proceedings of the 2003 ACM symposium on Applied computing. New York, NY, USA: ACM Press, 2003, pp. 784–788.

[7] Deng, Z.-H., Tang, S.-W., Yang, D.-Q., Zhang, M., Li, L.-Y., Xie, K. Q.,2004. A comparative study on feature weight in text categorization, in APWeb, vol. 3007. Springer-Verlag Heidelberg, March 2004, pp. 588 – 597.

[8] Lan, M., Tan, C. L., Su, J., & Lu, Y., 2009. Supervised and traditional term weighting methods for automatic text categorization. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(4), 721-735.

[9] Ko, Y., Seo, J., 2000. Automatic Text Categorization By Unsupervised Learning. In: Proceedings of the 18th conference on Computational linguistics-Volume 1. Association for Computational Linguistics.

[10] Lertnattee, V., Theeramunkong, T.,2004. Analysis Of Inverse Class Frequency In Centroid-Based Text Classification. In IEEE International Symposium on Communications and Information Technology, (ISCIT), Vol. 2.

[11] Altınel, B., Ganiz, M.C., Diri, B., 2013. A Novel Higher-order Semantic Kernel. In:Proceedings of the IEEE 10th International Conference on Electronics Computer and Computation(ICECCO) , pp. 216-219.

[12] Altınel, B., Ganiz, M.C., Diri, B., 2014b. A Simple Semantic Kernel Approach for SVM using Higher-Order Paths. In: Proceedings of IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA) , pp. 431-435.

[13] Altınel, B., Ganiz, M.C., Diri, B., 2014a. A Semantic Kernel for Text Classification Based on Iterative Higher–Order Relations between Words and Documents. In: Proceedings of the 13th International Conference on Artificial Intelligence and Soft Computing (ICAISC) , Lecture Notes in Artificial Intelligence(LNAI),vol.8467,pp.505–517.

[14] Dumais, S., Platt, J., Heckerman, D., Sahami, M., 1998. Inductive Learning Algorithms And Representations For Text Categorization. In: Proceedings of the Seventh International Conference on Information Retrieval and Knowledge Management (ACM-CIKM), pp. 148–155.

[15] Joachims, T., 1998. Text Categorization With Support Vector Machines: Learning With Many Relevant Features, pp.137-142. Springer Berlin Heidelberg.

[16] Boser, B. E., Guyon, I. M., Vapnik,V. N., 1992. A Training Algorithm for Optimal Margin Classifier. In Proceedings of the 5th ACM Workshop, Comput. Learning Theory, pp. 144–152

[17] Vapnik, V.N., 1995. The Nature of Statistical Learning Theory, Springer, New York.

[18] Alpaydın, E.,2004. Introduction to Machine Learning, MIT press.

[19] Kontostathis, A., Pottenger, W.M., 2006. A Framework for Understanding LSI Performance.Journal of Information Processing & Management, pp. 56-73.

[20] Wang, T., Rao, J., Hu, Q., 2014. Supervised Word Sense Disambiguation Using Semantic Diffusion Kernel. Engineering Applications of Artificial Intelligence, Elsevier ,(27),167-174.

[21] Kandola, J., Shawe-Taylor, J., Cristianini, N., 2004. Learning Semantic Similarity. Advances in Neural Information Processing Systems 15,657–664,2003.

[22] Nasir, J. A., Karim, A. , Tsatsaronis, G. Varlamis, I., 2011. A Knowledge-Based Semantic Kernel For Text Classification ,String Processing and Information Retrieval, 261-266, Springer.

[23] Tsatsaronis, G., Varlamis, I. Vazirgiannis,M., 2010. Text Relatedness Based On A Word Thesaurus. Journal of Artificial Intelligence Research 37,1–39.

[24] Bloehdorn, S., Basili, R., Cammisa, M., Moschitti, A., 2006. Semantic kernels for text classification based on topological measures of feature similarity. In: Proceedings of The Sixth International Conference on Data Mining (ICDM), pp. 808–812.

[25] Budanitsky, A., Hirst, G.,2006. Evaluating WordNet-based Measures of Lexical Semantic Relatedness, Journal Comput. Ling. 32 (1) (2006) 13–47.

[26] Lee, J. Ho, Kim, M. H., Lee, Y. J., 1993. Information Retrieval Based On Conceptual Distance in IS-A hierarchies. Journal of Documentation, 49(2):188–207.

[27] Luo, Q., Chen, E., Xiong, H., 2011. A Semantic Term Weighting Scheme for Text Categorization. Journal of Expert Systems with Applications, 38, pp.12708-12716.

[28] Nasir, J. A., Varlamis, I., Karim, A., Tsatsaronis, G., 2013. Semantic Smoothing For Text Clustering. Knowledge-Based Systems, 54, 216-229.

[29] Scott, S., Matwin, S. ,1998. Text Classification Using Wordnet Hypernyms. In: Proceedings of the COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems, pp. 45–52.

[30] 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 (IJCNN), IEEE, Vol. 5, pp. 205-209.

[31] Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller K., 1993. Five Papers on WordNet, Technical report, Stanford University.

[32] Zhang, PY., 2013.A HowNet-Based Semantic Relatedness Kernel for Text Classification. Indonesian Journal of Electrical Engineering (TELKOMNIKA) 11(4).

[33] Rodriguez, M.B. et al., 2000. Using WordNet to complement training information in text categorization, in: Proceedings of 2nd International Conference on Recent Advances in Natural Language Processing II: Selected Papers from RANLP'97, vol. 189 of Current Issues in Linguistic Theory (CILT), pp. 353–364.

[34] Hotho, A. et al., Ontologies Improve Text Document Clustering, in: Proceedings of the 3rd IEEE International Conference on Data Mining, 2003, pp. 541–544.

[35] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K.., Harshman, R., 1990. Indexing by Latent Semantic Analysis. J. of the American Society for Information Science, 41, 6, pp. 391-407.

[36] Zhang, Z., Gentile, A. L., Ciravegna, F., 2012. Recent Advances In Methods Of Lexical Semantic Relatedness–A Survey. Natural Language Engineering, 1(1), 1-69.

[37] Cavnar, W.B., Trenkle, J.M., 1994. N-Gram based text categorization, in: Proceedings of 3rd Annual Symposium on Document Analysis and Information Retrieval, pp. 161–169.

[38] Fung, B.C.M. et al., 2003. Hierarchical document clustering using frequent itemsets, in: Proceedings of SIAM International Conference on Data Mining, pp.59–70.

[39] Ho, T.B. , Funakoshi, K., 1998. Information retrieval using rough sets, Journal of the Japanese Society for Artificial Intelligence 13 (3), pp. 424–433.

[40] Ho, T.B., Nguyen, N.B., 2000. Non-hierarchical document clustering based on a tolerance rough set model, International Journal of Intelligent Systems 17, pp.199–212.

[41] Papka, R., Allan, J., 1998. Document classification using multiword features, in: Proceedings of the Seventh International Conference on Information and Knowledge Management Table of Contents, Bethesda, Maryland, United States, pp. 124–131.

[42] Lewis, D.D., 1992. An evaluation of phrasal and clustered representation on a text categorization task, in: SIGIR'92: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 37–50.

[43] Biricik, G., Diri, B., Sonmez, A. C., 2009. A new method for attribute extraction with application on text classification. In Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control(ICSCCW). Fifth International Conference on IEEE, pp. 1-4.

[44] Biricik, G., Diri, B., Sonmez, A. C., 2012. Abstract feature extraction for text classification. Turkish Journal of Electrical Engineering & Computer Sciences, 20(Sup. 1), pp. 1137-1159.

[45] Ganiz, M. C., Lytkin, N. I., & Pottenger, W. M., 2009. Leveraging Higher Order Dependencies Between Features For Text Classification. In: Proceedings of the Conference Machine Learning and Knowledge Discovery in Databases (ECML/PKDD), pp. 375-390.

[46] Poyraz, M., Kilimic, Z.H., Ganiz, M.C., 2012. A Novel Semantic Smoothing Method Based on Higher-order Paths for Text Classification.In IEEE International Conference on Data Mining (ICDM), pp. 615-624.

[47] Poyraz, M., Kilimic, Z.H., Ganiz, M.C., 2014. Higher-Order Smoothing: A Novel Semantic Smoothing Method for Text Classification. Journal Of Computer Science and Technology,Vol.29， No.3, 2014, pp.376-391.

[48] Chen, L., Guo, G., Wang, K., 2011. Class-dependent projection based method for text categorization. Pattern Recognition Letters, 32(10), pp.1493-1501.

[49] Chen, L., Guo, G., 2010. Using class-dependent projection for text categorization. In:Proc. Internat. Conf. on Machine Learning and Cybernetics, pp. 1305–1310.

[50] Bloehdorn, S. , Moschitti, A., 2007. Combined Syntactic and Semantic Kernels For Text Classification, Springer.

[51] Zhang, W., Yoshida, T., Tang, X., 2008. Text classification based on multi-word with support vector machine. Knowledge-Based Systems, 21(8), pp.879-886.

[52] Ganiz, M.C., George, C., Pottenger, W.M., 2011. Higher-order Naive Bayes: A Novel Non-IID Approach to Text Classification.In: IEEE Transactions on Knowledge and Data Engineering(TKDE), vol. 23, no. 7, pp. 1022-1034.

[53] Bisson, G., Hussain, F., 2008. Chi-Sim: A New Similarity Measure for the Co-clustering Task. In: Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications(ICMLA), pp. 211–217.

[54] Cristianini, N., Shawe-Taylor, J., & Lodhi, H., 2002. Latent semantic kernels. Journal of Intelligent Information Systems, 18(2-3), pp. 127-152.

[55] Wittek P., Tan, C., 2009. A Kernel-Based Feature Weighting For Text Classification.In Proceedings of IJCNN-09, IEEE International Joint Conference on Neural Networks., pp. 3373–3379.

[56] Hall, M, Frank, E. Homes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[57] Kamber, I.H., Frank, E., 2005. Data Mining: Practical Machine Learning Tools And Techniques, 2nd Edition, Morgan Kaufmann, San Francisco.

[58] Robertson, S. E. 2004. Understanding Inverse Document Frequency: On Theoretical Arguments for IDF. Journal of Document. 60, 5, 503-520.

[59] Harish, B. S., S. Manjunath, D. S. Guru. "Text Document Classification: An Approach Based On Indexing." International Journal of Data Mining & Knowledge Management Process (IJDKP) 2.1 (2012): 43-62.

[60] B. Altinel, M.C. Ganiz, B. Diri, “A Corpus-Based Semantic Kernel for Text Classification by using Meaning Values of Terms” , Elseiver, Engineering Applications of Artificial Intelligence Volume 43, August 2015, Pages 54–66.(doi:10.1016/j.engappai.2015.03.015).

[61] N. Cristianini and J. Shawe-Taylor, “An Introduction to Support Vector Machines and Other Kernel-based Learning Methods,” Cambridge University Press, 2000.

[62] Balinsky, A., Balinsky, H., Simske, S., 2010. On the Helmholtz principle for Documents Processing. In: Proceedings of the 10th ACM Document Engineering (DocEng).

[63] Balinsky, A., Balinsky, H., Simske, S., 2011a. On the Helmholtz Principle For Data Mining. In: Proceedings of Conference on Knowledge Discovery, Chengdu, China.

[64] Balinsky, A., Balinsky, H., Simske, S., 2011b. Rapid change Detection And Text Mining. In: Proceedings of 2nd Conference on Mathematics in Defence (IMA), Defence Academy, UK.

[65] Balinsky, H., Balinsky, A., Simske, S., 2011c. Document Sentences As A Small World. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 2583-2588.