A new hybrid semi-supervised algorithm for text classification with class-based semantics

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

Murat Can Ganiz[2]
Department of Computer Engineering,
Marmara University
Istanbul, Turkey
murat.ganiz@marmara.edu.tr

Abstract— Vector Space Models (VSM) are commonly used in language processing to represent certain aspects of natural language semantics. Semantics of VSM comes from the distributional hypothesis, which states that words that occur in similar contexts usually have similar meanings [1]. In our previous work, we proposed novel semantic smoothing kernels based on class specific transformations [2][3]. These kernels use class-term matrices, which can be considered as a new type of VSM. By using the class as the context, these matrices can extract class specific semantics by making use of word distributions both in documents and in different classes. The classification algorithms which are built on kernels like Support Vector Machines (SVM) can make use of these strictly supervised semantic kernels to achieve higher accuracy compare to traditional VSM based classifiers for text classification. Following this we also proposed a simple yet effective classifier based on direct usage of class-term matrix values [4]. In this study, adapt two supervised semantic methods to build a novel and high performance semi-supervised text classification algorithm, which effectively utilizes both labeled and unlabeled data. In supervised learning systems, only labeled samples are used for building a classification model, which is then used to predict the class memberships of the unlabeled samples. However, obtaining labeled data is usually very expensive, time consuming and difficult in real-life practical situations as labeling a data set usually requires the efforts of human experts. On the other side, unlabeled data are often plentiful which makes it relatively inexpensive and easier to achieve. Semi-Supervised Learning (SSL) approaches strive to utilize this abundant source of unlabeled instances to improve the learning capacity of the classifier especially when amount of labeled instances are limited. Since SSL techniques reach higher accuracy and require less human effort, they attract a substantial amount of attention both in practice and theoretical research. The new hybrid semi-supervised algorithm we propose combines two different methods in order to predict the class labels of unlabeled examples in a corpus and incorporate them into the training set to build a better classifier. The proposed algorithm use Helmholtz principle based calculation of term meanings for initial classification and a class-based term weighting based semantic kernel with SVM for the final classification model. Term meaning calculations depend on the Helmholtz principle from the Gestalt theory and calculated in the context of classes. Both meaning calculations and class-based weighting essentially group words based on their semantic similarity for each class. We perform various experiments on popular benchmark textual datasets and report the results with respect to wide range of experimental conditions in order to evaluate the proposed approach. Our results show our new hybrid methodology increase the classification accuracy significantly.

Keywords— semantics; semi-supervised classification; text classification; semantic smoothing kernel; class-based transformations.

1. INTRODUCTION

Vector Space Models (VSM) are commonly used language processing to represent some aspects of natural language semantics. In this model, documents are simply represented as points in a space and closeness of two points is proportional to their semantic similarity. There are several categories of VSM including word-context, pair-pattern and term-document matrices [5]. As pointed out in [5], semantics of VSM comes from the distributional hypothesis, which states that words that occur in similar contexts usually have similar meanings [1]. One of our main motivations is to use a class of documents as the context. In our previous work, we proposed novel semantic smoothing kernels based on class specific transformations [2][3]. Since these kernels use class labels explicitly, they are strictly supervised. Furthermore, they can be considered as a new type of VSM consist of term-class matrices specific to the class labeled documents. By using the class as the context, these matrices can extract class specific semantics by making use of word distributions both in documents and in different classes. These semantic kernels can be integrated into supervised classifiers i.e. SVM for text classification and they could be able to outperform baseline classifiers using document based traditional VSM. In this study, by combining one of these supervised semantic kernel based classification algorithm [3] with class meanings based methodology to classify unlabeled documents which is inspired from [4], we propose a novel and high performance semi-supervised text classification algorithm.

In machine learning applications there are two conventional strategies; supervised learning and unsupervised learning. Traditional supervised learning algorithms need a set of sufficient labeled data as training set to build the classification model, which will be used to predict the class memberships of the unlabeled examples. On the other hand, unsupervised learning, solely based on unlabeled samples, doesn't need any labeled data to learn a model. So as to train a classifier, it attempts to discover the indirect

structure of unlabeled data [6]. There have been massive amounts of accumulated data on the web, particularly on blogs, forums, social networks and continue to increase day by day without any doubt. But unfortunately most of the available data does not have pre-assigned labels which limit their use in several practical machine learning application fields such as text classification, sentiment recognition, speech recognition. Moreover, generally it can be time-consuming, tedious and expensive to assign labels to them manually. Most importantly, learning a classifier with only a few labeled training data may not generate sufficient performance. In situations where labeled data is inadequate, many algorithms have suggested exploiting and utilizing the unlabeled data to support to learning process for better classification. SSL approaches utilize not only labeled data but also unlabeled data to increase the classification accuracy. Recently, SSL has become popular and gained increased attention of both academic and commercial platforms as a new machine learning strategy. SSL is different from two ordinary classification approaches by the usage of unlabeled data to mitigate the effect of insufficient labeled data on classifier accuracy. Many SSL systems have been offered in the past years, like co-training [7], self-training [8, 9] graph-based methods [10], semi-supervised support vector machines [6], EM with generative mixture models [11], transductive support vector machines [12].

Classification of texts requires special techniques to transform unstructured text into structured format required for classification algorithms; usually a vector of features. In this domain, documents are usually symbolized by terms and their corresponding frequencies. This kind of representation methodology is actually the most popular one in the literature and it is named as Bag of Words (BOW) feature demonstration. BOW is known as the simplest feature representation method where each term comprises a dimension in a vector space, being independent of other terms in the same document [13]. A bag is mathematically similar to set with the difference that there can be duplicate values. As in sets, the order of words is lost in bag representation. Similarly, a bag can be represented as a vector and a group of bags also can be represented as a matrix where the rows are documents and columns are term frequencies. This also called Vector Space Model (VSM). Although, the VSM and BOW approach is very simple and commonly used, they have several limitations as discussed in [2]. One of the restrictions is the assumption of independency between terms. Documents are represented only with their term frequencies, disregarding their position in the document or their semantic or syntactic links between other words. This is a big problem since it clearly ignores the multi-word expressions by separating them. Moreover; it cannot handle polysemous words (i.e. words with multiple meanings) since it treats them as a single entity. Furthermore, it maps synonymous words into different components; as discussed in [14]. In principle, as Steinbach et al. [15] mention, each class has two kinds of vocabulary: one is “core” vocabulary which are closely correlated to the theme of that class, the other type is “general” vocabulary those may have similar distributions on different classes. Consequently, two documents from different classes may commonly have many general words and can be categorized as similar in the BOW demonstration.

In our recent study in [2], we offer a novel method for constructing a supervised semantic smoothing kernel for SVM, which we name Class Meaning Kernel (CMK). The proposed method smoothens the words of a document in BOW demonstration by class-based meaning values of terms. The meaning scores of words are calculated based on the Helmholtz principle from Gestalt theory [16, 17, 18, 19] in the scope of classes. According to our experimental results, CMK is superior to the traditional kernels such as linear kernel, polynomial kernel and RBF kernel. In one of our previous studies, we suggest a new classifier for textual data, named as Supervised Meaning Classifier (SMC) [4]. The SMC classifier uses meaning calculations, which is based on Helmholtz principle from Gestalt Theory. In SMC, meaningfulness of terms in the scope of classes are calculated and used for classification of a document. According to the experiment results this new SMC classifier outperforms Multinomial Naïve Bayes (MNB) and SVM specifically on insufficient training data.

In another recent study of ours [3], we offer a novel approach for constructing a supervised semantic kernel for SVM, which we name Class Weighting Kernel (CWK). The proposed method smoothens the terms of a document in BOW representation by class-based weights of words. The weights of terms are calculated based on a new term weighting approach that is designed as a part of a feature extraction algorithm is presented in [20, 21]. According to our experimental results the classification performance of CWK is higher than the classification performance of the other commonly used kernels (i.e. linear kernel, polynomial kernel and RBF kernel).

Inspired by the advantages of CMK, CWK and SMC, and motivated by the fact that there is insufficient labeled data in real life practical applications, we build a new hybrid semi-supervised form of CWK and SMC, which is called Hybrid Class Semantics Classifier (HCSC). More precisely, in this article we suggest a novel non-iterative semi-supervised methodology that uses class-based meaning values and weights of terms. The suggested approach utilizes both labeled and unlabeled data in order to build a classifier. First it smoothens the terms of the labeled documents in BOW demonstration (document vector represented by term frequencies) by class-based meanings of words as the same way it is done in CMK [2] and SMC [4]. Then, it attempts to find suitable labels for unlabeled instances. It succeeded this labeling process by weighting the terms of the unlabeled documents in BOW representation with the help of meaning calculations [4]. Following this, HCSC combines the original labeled data with newly classified unlabeled data. Finally, CWK is applied on this enlarged labeled dataset in order to predict the labels of the samples in the test dataset. The smoothing process in HCSC increases the significance of important (i.e. meaningful) words specific to a particular class while reducing the importance of general terms those have a similar distribution in all classes. Since this method is used in the transformation phase of a kernel function from input space into a feature space, it considerably decreases the effects of above mentioned disadvantages of BOW. We note that HCSC advances the accuracy of SVM compare to the linear kernel by giving more significance to the class specific concepts, those may possibly be synonymous or very closely associated in the scope of a class.

The HCSC 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 close positions in the feature space of SVM if they are written using semantically nearby sets of terms on the same subject. The main novelty of our approach is the use of this meaning information in the both labeling of unlabeled data and smoothing process of the semantic kernel. The meanings of words are calculated based on the Helmholtz principle from Gestalt theory [16, 17, 18, 19] in the scope of both classes and documents as in [2]. HCSC directly incorporates class information to the semantic kernel for labeled instances. Additionally it also incorporates unlabeled instances in the training process. Therefore, it can be considered as a semi-supervised approach.

We performed a number of experiments on several document datasets with numerous different labeled/unlabeled/test splits of the corpus. According to our experimental results HCSC broadly outperforms the performance of baseline algorithms.

The first gain of our proposed solution is the classification capability of HCSC. To show the performance difference and robustness of HCSC we perform some experiments on various textual datasets with different labeled/unlabeled portions of the dataset. Experimental results demonstrate that HCSC exceeds the performance of baselines including the semi-supervised form of linear kernel. In linear kernel the inner product between two document vectors is used as kernel function, which exploits information about shared term in these two documents. This approach can be categorized as first-order method since its scope comprises of just a single document. Though, HCSC can take advantage of meaning values of terms in the context of classes. In this way semantic relation between two terms is composed of corresponding class-based and instance-based meaning scores of these words for all classes. Consequently if these two words are significant words in the same class then the resulting semantic relatedness value between them will be greater.

The second benefit of the offered approach is its relatively low complexity since HCSC does not need the processing of a large external knowledge base like WordNet or Wikipedia. Besides, as HCSC is built on corpus based statistics it is always up to date. As a result it can be applied to any field without adjustments or parameter optimizations.

The other improvement of HCSC is that it also forms a foundation that can easily be combined with other term based similarity measures. It is also easy to take advantage of similarities between terms derived from a semantic resource such as Wikipedia or WordNet.

According to the experimental results, there is an important improvement in the classification accuracy when class-dependent meaning values of terms are used in SVM, compared to the performance of the baseline kernels. To the best of our knowledge, class-dependent meaning values of words are used to assign labels to unlabeled instances and are utilized in the transformation phase of SVM for the first time in the bibliography and also give important insights on the semantic smoothing of words for text classification.

The rest of the paper is organized as follows: First, a brief introduction to supervised classification algorithms, mainly SVM, and semi-supervised algorithms in general for text classification, methods for building class-term matrix based VSMs including meaningfulness calculations in the class context and class-based term weighting are given in Section 2. Section 3 explains and analyzes the offered kernel for text classification algorithm. Experimental setup, the corresponding experiment results with some discussion points are presented in Section 4 and Section 5. Finally, we conclude the article in Section 6 and provide a discussion on probable future extension points of the current work.

2. RELATED WORK

2.1 Support Vector Machines for Classification

SVM is a very effective machine learning algorithm stem from Statistical Learning Theory (SLT). SVM is based on the structural risk minimization principle from computational learning theory [22, 23, 24], whose basic goal is try to find the optimal separating hyperplane with the maximal margin between two classes which guarantees the lowest error for an unseen test instance. If the problem is not linearly separable, the SVM algorithm may be altered by adding a softening variable that allows some samples passing the hyperplane margins with certain penalty. When it is not possible to find a separation linear hyperplane, a solution is to build a non-linear classier using a kernel function. The kernel function projects the problem from the original space to a higher dimensional space where it is possible to separate the data linearly. A kernel function in SVM can be considered as a kind of similarity function, since it computes the data points' similarity values, in the transformed space. A significant property of a kernel function that it needs to obey to Mercer's condition as it is stated in [25]. Therefore, defining a suitable kernel has the straight influence on finding a better representation of these data points as discussed in [14, 26, 27]. Some common 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: $\kappa(d_p, d_q) = (d_p d_q + 1)^b, b = 1,2 \ldots etc.$ (2)
- RBF kernel: $\kappa(d_p, d_q) = \exp(\gamma \left\| d_p - d_q \right\|^2)$ (3)

Transductive Support Vector Machines (TSVMs) is an extension of traditional SVM with the contribution of unlabeled data. Traditional SVM utilize only the labeled data in order to build the classifier and the aim is to find a maximum margin linear boundary in the feature space. On the other hand, not only the labeled data but also the unlabeled data is used in TSVM. The aim

is to predict the labels of the unlabeled data and use them in the training step; therefore a linear boundary has the maximum margin on the labeled data (the original labeled data with the addition of labeled unlabeled data this time). Intuitively, the decision boundary has the smallest generalization error on unlabeled data, since unlabeled data leads the linear boundary especially in dense regions [23].

2.2 Semi-Supervised Learning Approaches

SSL methods utilize unlabeled examples for building better classifiers with higher accuracy when there are a few amounts of labeled training examples. In a typical SSL scenario there are labeled training examples (L) and unlabeled examples (U). Co-training and Self-training are two popular SSL systems. Self-training basically works as follows [8]; a classifier is constructed from L and utilized to estimate the labels for samples in U. Then m unlabeled samples that the classifier has high classification confidence in U are assigned labels and moved to extend L. After that, the classifier is re-trained using the enlarged data set L. Although it is a very simple algorithm; since it is not easy to promise the convergence of it, the latter three stages are commonly repeated for a pre-defined maximum iteration number of times or reaching up until some heuristic convergence standard (i.e., there is no remaining unlabeled instances in U).

Co-training works like self-training except that it supposes that attributes can be divided into two different views. According to co-training algorithm input features are logically partitioned into two independent groups, and two separate classifiers are trained on these two subsets in labeled data set (L) [7]. Following this, each classifier is attempted to label the unlabeled samples in U; to put it in more detail; for each classifier, the instances in U with the highest classification confidence are selected and added to the labeled data set L. Consequently these two classifiers can assist for enlarging the data set L. Both classifiers are retrained on this expanded data set, and the stages are re-performed a fixed number of times. Thus each classifier then categorizes the unlabeled data, and teaches the other classifier with some unlabeled instances (those have their newly estimated labels with high classification confidence). Each classifier will be trained again with the supplementary training examples specified by the other classifier, and the procedure is repeated for higher accuracy [6, 7, 28]. As it is discussed in [29 the main idea in co-training is that a classifier may give suitable labels to some samples whereas it may be challenging for the other classifier to do so. Hence, each classifier can enlarge the training set with instances which are useful and important for the other classifier.

Different kinds of self-training and co-training algorithms have been offered by researchers over the years. One variant of them utilizes the whole unlabeled data in each iteration therefore there is no need for the selection criterion. One example of this kind is presented in [30]. The labels of the all unlabeled examples are predicted and then used to extend the training set and update the classifier at all iterations. In [31], co-EM is presented which uses all the unlabeled samples rather than a number of samples chosen from the data pool. Another type of approaches is to use active learning technique to choose unlabeled examples and then ask some human experts to label them which yields no mislabeled examples will occur, in principle. In [32], a system with active learning is applied to choose unlabeled examples for the multi-view semi-supervised Co-EM algorithm. In [33], in each iteration, uncertainty sampling is used to select unlabeled instances, then a cost-sensitive classifier is built on the extended labeled data and all unlabeled examples with assigned categories. Nevertheless active learning methods are difficult to apply since they cannot be accomplished without human experts.

Confidence selection is a popular instance selection criterion, which selects unlabeled examples to add to the training set which are classified with high classification confidence [7, 8, 31, 33, 34]. Other selection approaches have also been suggested by researchers. For instance, Wang et al., [35], offered an adapted Value Difference Metric as the selection technique in self-training. Their approach is based on decision tree classifiers and used to classify sentences as subjective or objective. They use the Naive Bayes trees algorithm, in order to build a Naive Bayes Classifier at each leaf of the tree. Their approach work well with very small datasets. In [36], a new data editing approach, named SETRED, is presented. Their approach benefits from the information of the neighbors of each self-labeled instance to recognize and remove the mislabeled samples from the self-labeled data. In [37], ISBOLD selection strategy is used to roughly prevent possible performance degradation in self-training and co-training.

Li et al. [38] presents a new methodology which uses three learners. According to [38] L denotes the labeled example set, h_1, h_2 and h_3 indicate initial learners and U show the unlabeled instance set and x is an instance in U. Firstly, three classifiers are trained from labeled instances. Then, any two of those classifiers are used to label the unlabeled sample x, if two of them predict the same label; then that instance will be utilized to teach the third classifier. It repeats this procedure until none of h_1, h_2 and h_3 changes. The final estimation is accomplished with a majority vote among all the learners.

Preferably, the selected unlabeled examples (together with the assigned labels) can finally assist to learn a better classifier. However, [39] stated that unlabeled data may reduce classification accuracy in some extreme situations and when the model assumptions are not correct. For instance in [40], an extensive empirical study was conducted on several popular SSL systems (including co-training and self-training) using different base Bayesian classifiers. According to their results on 26 UCI datasets, if the classifier has poor performance and incorrectly assigns labels to some self-labeled examples, there will be accumulated mislabeled data which yields the final performance will be jeopardized. McCallum and Nigam [11] mention that, they get better classification performance by combining a small set of labeled examples with a large set of unlabeled data with EM. Unfortunately, there are many studies showing that unlabeled examples are quite often detrimental to the performance of classifier in many situations [39]. According to those studies the more unlabeled data are joined with a fixed number of labeled instances, the poorer is the classifi-

cation performance of the corresponding classifier. Therefore, it is obvious that, the classifier should have a good classification performance on the original labeled data if it desires to have good classification performance on test data. More accurately, utilizing the accuracy on the original labeled data to select more reliable unlabeled samples seems critical for the final classification performance of the SSL algorithm.

In [41], a *C4SVM* algorithm is presented, which includes misclassification costs into the optimization function of a semi-supervised SVM. In some algorithms only one base learner is applied, which use the unlabeled samples iteratively based on its own knowledge. Some approaches include using Estimation-Maximization algorithm to estimate posterior parameters of a generative model, Naive Bayes, by labeling each unlabeled sample, i.e. a probability for each class as it is done in [11]; using the unlabeled data to search for a better configuration of Bayesian Network [42]; using a transductive inference for SVM on a special [43]. The self-training algorithm [31] is of that kind, where all iterations the learner converts the most confidently classified unlabeled sample of each class into a labeled training example.

These methods and their variants are also described, analyzed and compared in [12]. Furthermore there is a comprehensive survey that includes almost all of traditional semi-supervised learning algorithms in [6] and in [30].

Also there is a recent work [44, 45] whose aim is to create a collection of commonly used 'polarity concepts' i.e. common sense concepts with comparatively strong positive or negative polarity. Sentic Computing [46, 47] is a novel opinion mining and sentiment analysis paradigm which exploits AI and Semantic Web techniques to better identify, interpret and process sentiments and opinions in natural language text. In Sentic Computing, whose term derives from the Latin 'sentire' (the root of words such as sentiment and sensation) and 'sense' (intended as common sense), the analysis of text is not based on statistical learning models but rather on common sense reasoning tools [48] and domain-specific ontologies [49, 50]. Sentic Computing enables the analysis of documents not only on the page or paragraph-level but even on the sentence level.

ConceptNet was created as a demonstration for the knowledge gathered by the Open Mind Common Sense project [51]], that tries to use an interactive way to collect new statements from visitors to the site by asking them some questions. In detail, ConceptNet expresses concepts (i.e. phrases and words which can be removed from natural language text) and assertions of the ways that these concepts communicate to each other. There are a few publicly released versions of ConceptNet; besides the latest one which is called as ConceptNet 5 contains many representational improvements [52]. AffectiveSpace [54, 76] is an n-dimensional vector space built from ConceptNet and WordNet-Affect, a linguistic resource for the lexical representation of affective knowledge [53].

2.3 Helmholtz principle from Gestalt theory and meaningfulness calculation

According to the Helmholtz principle from the Gestalt theory in image processing [17]; events which have large deviations from randomness or noise can be perceived simply by human beings. It is demonstrated in Fig. 1. In the left hand side of Fig. 1, there is a group of five aligned points nevertheless with the existence of the high noise (i.e. large number of randomly placed dots) as the alignment probability of five points increases it is difficult to identify it at first glance. However; if we eliminate the number of randomly placed points (i.e. the right hand side of Fig. 1), we can easily notice the alignment pattern because it is very unlikely to happen by chance as it discussed in [17].

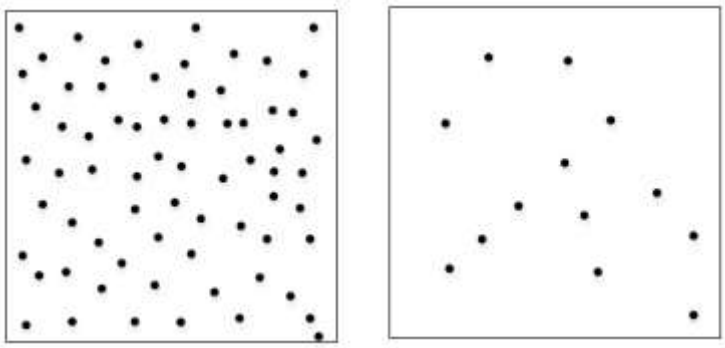

Fig. 1. The Helmholtz principle in human perception. Adopted from [17]

Consequently, the above illustrative example and other examples in [17] point out that interesting events and meaningful features appear in large deviations from randomness. This meaningfulness fundamentally related to calculations of expectations and it is originated from the approaches in statistical physics as it is mentioned in [17].

In the document mining field, the textual data is composed of natural structures which are made up of topics, documents, paragraphs and sentences. In [17], the authors work on the textual data by utilizing the human perceptual model of Helmholtz principle from Gestalt Theory. Therefore they assign a meaning score to each term or word in order to establish meaningfulness of

these structures. Their novel methodology for meaningful keyword extraction is founded on two important principles as mentioned in [17] and utilized in [2]:

1) The keywords should be defined both in the document scope and in the scope of other documents. This is similar to the traditional TF-IDF approach.

2) The topics are indicated by "unusual activity", a new topic can be noticed by a sharp rise in the frequencies of particular terms. They state that sharp increase in frequencies can be used in rapid change detection. In order to detect the change of a topic or occurrence of new topics in a stream of documents, bursts on the frequencies of words need to be observed [55].

Stemming from the theories given above, new approaches are implemented for a number of associated application areas including text summarization [18], for information extraction and unusual behavior detection from small documents [56], keyword extraction and rapid change detection in data streams and documents [16, 17] and also for defining relations between sentences using social network analysis and properties of small world phenomenon [19].

According to their theories, for calculating the meaning of a term *w* whose frequency is *m* in a context (document, paragraph, and sentence), its Number of False Alarms (NFA) value can be described as in Eq. (4). If the NFA (expected number) is less than one, then the frequency of *m* can be reflected as a meaningful event since it is not expected by our calculations even though it already exists. Thus, word *w* can be considered as a meaningful or significant term in the given scope. According to the NFA, the meaning values of terms are calculated with Eq. (4) and Eq. (5) in [19]:

$$NFA(w,P,D) = \binom{K}{m} \frac{1}{N^{m-1}} \tag{4}$$

$$Meaning(w,P,D) = -\frac{1}{m} \log NFA(w,P,D) \tag{5}$$

where *w* denotes a word, *P* shows a part of the document like a paragraph or a sentence, and *D* indicates the entire document. Moreover, *m* represents the frequency of term *w* in *P* and *K* denotes the frequency of word *w* in *D*. $N = L / B$ in which *L* is the length of *D* and *B* is the length of *P* in words [19] as in Eq. (6). In order to define *Meaning* function, the logarithmic value of NFA is used based on the observation that NFA values can be exponentially large or small [17].

In supervised meaning calculations, those are given in Eq. (7) and Eq. (8), the parameter c_j indicates documents which belong to class *j* and *S* denotes the whole training set, *k* shows the frequency of word *w* in the dataset *S*, and *m* indicates the frequency of the word *w* in the documents of class c_j.

$$L = \sum_{d \in S} \sum_{w \in d} tf_w , \qquad B = \sum_{d \in c_j} \sum_{w \in d} tf_w \qquad , \quad N = \frac{L}{B} \tag{6}$$

$$NFA(w,c_j,S) = \binom{k}{m} \frac{1}{N^{m-1}} \tag{7}$$

Based on NFA, the meaning value of the term *w* in a class c_j is defined as:

$$meaning(w,c_j) = -\frac{1}{m} \log NFA(w,c_j,S) \tag{8}$$

The bigger the meaning score of a word *w* in a class c_j means it is more significant or informative word for that class.

2.4 Term weighting methods

There are different ways to calculate suitable weights for the words to increase the classification performance: For instance; Term Frequency (TF), binary, Term Frequency-Inverse Document Frequency (TF-IDF) [13, 57] and its variants are the customary approaches borrowed from Information Retrieval (IR) field and are categorized as the unsupervised term weighting methods. Similarly there are supervised term weighting methods in which term weights are calculated according to the category membership information of training documents. Gain Ratio, Information Gain (IG), odds ratio are the traditional types of this category [58, 59].

TF-IDF [57] is the most popular term weighting method. Its formula is given in Eq. (10), where tf_w signifies the frequency of the word *w* in the document and IDF is the inverse of the document frequency of the word in the dataset. IDF's formula is also given in Eq. (9) where $|D|$ stand for the total number of documents and df_w denotes the number of documents which contains term *w*. TF shows the frequency of term *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 [63].

$$IDF(w) = \frac{|D|}{df_w} \tag{9}$$

$$TF - IDF(w, d_i) = tf_w \times \log(IDF(w)) \tag{10}$$

A similar but supervised kind of TF-IDF is called Frequency–Inverse Class Frequency (TF-ICF) whose formula given in Eq. (12) as in [61, 62]. In Eq. (11), $|C|$ shows the number of classes and cf_w represents the number of classes which includes term w.

$$ICF(w) = \frac{|C|}{cf_w} \tag{11}$$

$$TF - ICF(w, c_j) = \sum_{d \in c_j} tf_w \times \log(ICF(w)) \tag{12}$$

In [60] a new term weighting method, namely Term Frequency-Relevance Frequency (TF-RF) which considers only the frequency of relevant documents (i.e. those which contain this term). According to the study in [60], for the multi-label text classification task, when a text classifier is built for each category, then this category is labeled as positive category and all the other categories are grouped as negative category. Thus, their new proposed term weighting method has the idea of simplifying a multi-label classification problem into multiple independent binary classification problems. In their approach, 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. Hence, the documents in the positive category focus on one theme or several themes close to each other while the remaining documents in the negative category are spread over a wide range of subjects since all non-positive categories are actually grouped together as negative [60]. Therefore, the high frequency terms focused in the positive category are useful discriminators to select the positive examples from among the numerous negative samples. Their term weightings formula is as follows:

$$TF - RF = tf_w \times \log\left(2 + \frac{a}{\max(1, c)}\right) \tag{13}$$

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 those contain term w. Table 1 demonstrates an illustrative example about the discriminative powers of both IDF and RF. There are four terms in Table 1 namely; *acquire*, *stake*, *payout* and *dividend*. Table 1 lists the IDF and RF scores of these four words based on two categories, namely, 00_acq and 03_earn; respectively. The first two terms, acquire and stake, are thoroughly correlated to the subject mentioned in category 00_acq while the other two terms, payout and dividend, are thoroughly correlated to the subject mentioned in category 03_earn. Nevertheless, as the IDF factor ignores the category information of the training set, each of these four terms is weighted almost equally by the IDF factor even in terms of the two different labels. Conversely, by using the RF weighting approach that concentrates on the category information, each term is assigned more suitable weights in terms of different categories [60].

By being inspired from the idea of both IDF and ICF, a new term weighting technique which is built as a part of a feature extraction algorithm is presented in [20, 21]. According to their approach the influence of a term over a class is calculated as follows:

$$W_{w,c} = \log(tfc_{w,c} + 1) \times \log\left(\frac{N}{N_w}\right) \tag{14}$$

where $tfc_{w,c}$ is the total term frequency of word w in the documents of class c, N is the total number of documents in the corpus and N_w is the total number of documents those include term w. By utilizing this class-dependent term weighting scheme they develop a feature extraction technique for text classification. They conduct experiments on benchmark datasets to compare the classification performance with traditional feature extraction approaches. According to their experimental results, this feature extraction with a class-dependent term weighting scheme improves the classification performance on the classifiers they use when compared with other feature extraction approaches.

Table 1
Comparison of the weights of four features in Category 00_acq and 03_earn ([60])

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

3. HYBRID CLASS SEMANTICS CLASSIFIER (HCSC)

Linear kernel is commonly used as the kernel of choice in SVM for text classification. This is partly due to the fact that the BOW representation of documents is quite high dimensional usually consisting of thousands of features. The high dimensionality of the text makes the other kernels, which maps the input space into a higher dimensional space with the hope of finding a better separating hyperplane, generally useless. As the simplest kernel that can be used in SVM, the linear kernel is formulated in Eq. (1). It basically consists of just a dot product between the feature vectors of two instances (text vectors in our case). This yields producing a simple similarity value between two documents depending on just the number of common words. As we mentioned and analyzed in [2, 3] this may cause an important difficulty particularly when two documents have the same theme using two different sets of words which are indeed semantically very near to each other. Therefore, their calculated similarity value will be zero since it depends on only the number of shared terms those contain the exactly the same characters. But as it is illustrated in [2] with Fig. 1, in many real world cases documents can be written with different but semantically similar terms. Also, it will be very hard to detect reliable patterns between documents in circumstances where training data is limited. In this case the feature vectors will be quite sparse. This concludes that utilizing only dot product to calculate the similarity value between two documents will not always produce adequately exact similarity value between those documents. As mentioned before, in order to achieve a better classification performance it is essential to discount general words and stress on core words (which are strictly correlated to the theme of that class) as is analyzed in [15]. So as to surmount the problems described above, semantic smoothing kernels encode semantic dependencies between terms [27, 64].

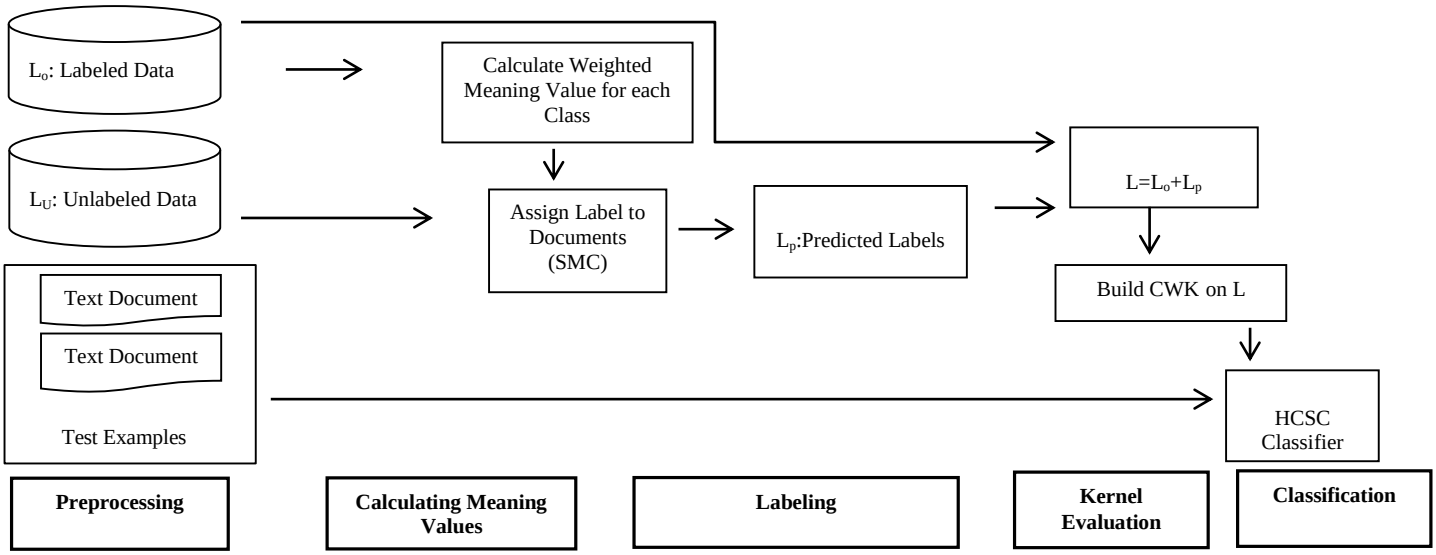

Fig. 2. The architecture of HCSC System

In this article, we introduce a new hybrid semi-supervised approach, which uses class-based meaning values and weights of terms. HCSC is mainly composed of five independent modules as shown in Fig. 2 including preprocessing, meaning calculation, labeling, kernel evaluation and classification. The first step of the system is preprocessing and it involves the conversion of input documents into formatted information including stemming, stopword filtering. In meaning calculation step, the meanings of the words for the classes are calculated based on Eq. (8). Then the unlabeled instances are classified and given labels depend on the meaning scores of the words in the labeling step. After that we construct our proposed kernel for kernel evaluation. Finally, in the classification step CWK [3] predict the labels of the test instances with the model built in the training phase. The details of all these steps are given in the following sub-sections.

3.1 Preprocessing

In the preprocessing step we apply stemming and stopword filtering on the text documents. Furthermore, we filter and remove infrequent terms those appear less than three times in documents. Moreover, we apply attribute selection and select the most informative 2,000 terms using IG as used in [2, 3, 65, 66, 67, 68, 69, 70]. This preprocessing increases the classification performance of the models by reducing the noise. We perform this preprocessing equally in all experiments that we report in the following section.

3.2 Meaning Calculations

In our approach $D_{labeled}$ is the data matrix consisting of only the labeled instances with r rows (documents) and t columns (terms). In $D_{labeled}$, d_{iq} represents the frequency of the q^{th} term in the i^{th} document. We use Eq. (8) in order to calculate the meaning

values of the terms in this labeled set which produces M_{train} class-based term meaning matrix is made up of t rows (terms) and j columns (classes). The $M_{labeled}$ matrix shows the meaningfulness of the words in the labeled set for each class. If a term is appeared only once in a class then its meaning score for that class is calculated as zero based on Eq. (8). If a term is not appeared at all in a class, it gets minus infinity based on Eq. (8) as a meaning score for that class; but in order to make calculations more practical we give the next smallest score to that term according to the range of meaning scores we get for all the terms in our labeled documents. After all calculations M is built as a term-by-class matrix that has the meaning values of words in all classes of the labeled documents. We have noticed that these meaning values are large for those words which give us the opportunity to differentiate the classes. Certainly words semantically near to the subject mentioned in the class get the highest meaning values. In other words semantically connected words of that class, i.e. "core" words as mentioned in [15], get importance while semantically isolated words, i.e. "general" words lose their importance. Consequently, if a term is very significant word for a class (i.e. a core word) then its meaning value will be higher for that class amongst the other words. On the other hand if a word is not a very important word for a class (i.e. a general word) then its meaning score will be smaller for that class in compare to the other meaning values. Also as it is mentioned in [16] and utilized in [2] with Table 1, meaning calculation automatically filters stop words by assigning them very small amounts of meaning values.

Our approach, HCSC, utilizes not only labeled instances but also unlabeled instances. In order to incorporate unlabeled examples into the classifier model in SVM, we calculate the total meaning value of an unlabeled document using Eq. (15):

$$TM(d_i, c_j) = \sum_{n=1}^{t} M_{labeled}(w_{nj}) \times tfw_{n,d_i} \tag{15}$$

where $M_{labeled}(w_{nj})$ is the meaning value of the term w_n for the class c_j as mentioned above, tfw_{n,d_i} is the number of occurrence of the term w_n in the document d_i and $TM(d_i, c_j)$ is the total meaning value of the document d_i for the class c_j.

Eq. (15) produces a matrix as demonstrated in Fig. 3.

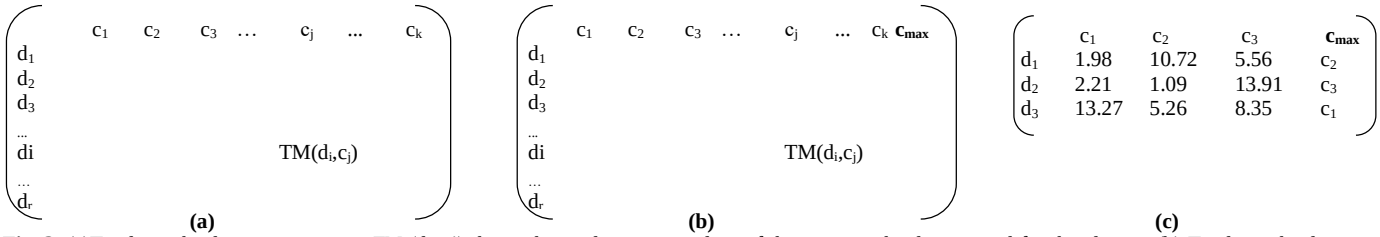

Fig. 3. (a)Total weighted meaning matrix, *TM (d_i, cj)* shows the total meaning values of the terms in the document d_i for the class c_j. (b) Total weighted meaning matrix, *TM (d_i, cj)* with c_{max} column signifies that the document gets maximum total meaning value for which class. (c)A quite simple illustrative example shows that the total meaning values and their greatest class value for the unlabeled instances of d_1, d_2 and d_3.

3.3 Labeling

In $TM(d_i, c_j)$ matrix, shown in Fig. 3 (a), $d_{ik} = [d_{i1},\ldots,d_{ik}]$ is the document vector showing the document d_i's total meaning values from the aspects of all the classes, respectively. We simply add another column into this matrix as demonstrated in Fig. 3(b). This new column just presents for the column (class number) with the greatest value in $d_{ik} = [d_{i1},\ldots,d_{ik}]$ document vector. A simplified sample case is given in Fig. 3 (c). According to Fig. 3 (c) an unlabeled document d_1 has the following meaning values of its terms: 1.98, 10.72, and 5.56 for the classes c_1, c_2 and c_3 respectively. Since the greatest value is 10.72 for the class c_2, then the unlabeled document d_1 is assigned a label as c_2. It is very likely that document d_1 contains terms, which are top ranked for the class c_2. The similar situations are valid for the documents d_2 and d_3, which are labeled as c_3 and c_1 as shown in Fig 3(c). After completing this labeling-step, all the unlabeled instances are assigned labels and the updated version of the labeled instances are found as follows:

$$L = L_o + L_p \tag{16}$$

where L_o is the original labeled examples, L_p is the previously unlabeled examples with their current predicted labels and L is the total of L_o and L_p; respectively. The labeling process in HCSC is actually done as in the similar way that classification is done in SMC [4].

3.4 Kernel Evaluation

When we come to the kernel evaluation step there are labeled instances (L; the composition of both L_o and $L_{p)}$ that will form the training set like a traditional supervised classification problem. After assigning class labels to all the unlabeled instances and combining them with the originally labeled dataset in order to form an augmented training set we basically run the CWK which is proposed in our previous study [3]. In CWK, we construct the class-based term weights matrix W with weighting calculations formulated in Eq. (18). The W matrix represents the weights of the words for each class. Then S matrix is calculated based on W in order

to expose class based semantic relationships between words. Precisely, the *i, j* element of *S* shows the semantic closeness between words t_i and t_j.

$$S = WW^T \tag{17}$$

The *S* is a symmetric term-by-term semantic smoothing matrix. Mathematically, the kernel value between two documents is given as

$$k_{\mathrm{HCSC}}(d_1, d_2) = d_1 SS^T d_2^T \tag{18}$$

where $k_{HCSC}(d_1, d_2)$ is the similarity value between documents d_1 and d_2. After using Eq. (19) the new representation of the documents is richer than the ordinary TF-IDF representation as; further statistical information which is directly calculated from our training corpus is added into the kernel. In other words transformations in Eq. (19) smooth the basic term vector demonstration using semantic ranking while passing from the original input space to a feature space through kernel transformation functions $\phi(d_1)$ and $\phi(d_1)$ for the documents d_1 and d_2 respectively:

$$\phi(d_1) = d_1 S \text{ and } \phi(d_2) = S^T d_2^T \tag{19}$$

As mentioned in [71], the presence of *S* in Eq. (18) and in Eq. (19) changes the orthogonality of the vector space model, as this mapping introduces term dependence. Documents can be seen as similar even if they do not share any terms by removing orthogonality.

3.5 Classification

We embedded our kernel function into the implementation of the SVM algorithm in WEKA [72] and consequently it can be directly used with Platt's Sequential Minimal Optimization (SMO) classifier [73]. In the classification-step all the test instances' labels are predicted in the experiments and the classification error rate is calculated. The details of them are explained in the following section. The algorithm of HCSC is given in Fig. 4.

Module Labeling *//Labeling* Unlabeled Documents
Input
$\overline{L_u}$: Unlabeled documents set
$\overline{M}_{o_{w,j}}$: matrix shows the meaning values of words for all classes in $\overline{L_o}$
Output
$\overline{L_p}$: Matrix shows the unlabeled documents with their predicted labels
Local variables
$\overline{TM}_{d_i,c_j}$: Matrix shows the unlabeled documents with their total meaning values for all classes (i ,j^{th} element shows the total meaning value of the document d_i in class c_j)
begin
$\overline{TM}_{d_i,c_j} = \overline{L_u}_{w,j} (\overline{M}_{w,j})$
for each document d_i in the $\overline{TM}_{d_i,c_j}$
$L_{P_i} = max(\overrightarrow{TM_{d_i}})$ *//Assigning the label of document* d_i
end for
end
Module Training *//Building semantic smoothing kernel*
Input
$\overline{L}$: $\overline{L_o} + \overline{L_p}$ (All the labeled documents)
Output
$\overline{G}_{d_p,d_q}$: Gram matrix shows the kernel value between documents d_p and d_q
Local variables
$m_{w,j}$: total frequency of word w in class j
$k_{w,d}$: total frequency of word w in L
N : the ratio of the length of the dataset and the class

$\overline{W}_{w,j}$: matrix shows the weights of words for all classes(w ,j^{th} element shows the weight of the word w in class j)

$\overline{S}_{w_p,w_q}$: Semantic smoothing matrix shows the relatedness between words w_p and w_q

begin

for each word w

for each class j

$$\overline{W}_{w,j} = \log(tfc_{w,j} + 1) \times \log(\frac{N}{N_w})$$ //Calculate the term weight of the word w in class j

end for

end for

$\overline{S} = \overline{W}\ \overline{W}^T$ *//Building semantic smoothing matrix*

for each document d_p in $\overline{L}$

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

Fig. 4. The Algorithm of HCSC

4. EXPERIMENT SETUP

4.1 Datasets

In order to HCSC on text classification, we used several textual datasets. IMDB[1] is our first dataset which is a collection of movie reviews and contains 2,000 reviews about several movies in IMDB. It has two types of labels; positive and negative. These labels are balanced in both training and test sets in our experiments. Our other four datasets are variations of popular 20 Newsgroup[2] dataset. 20 Newsgroup dataset is a group of approximately 20,000 newsgroup documents, divided evenly across 20 different newsgroups and commonly used in machine learning applications, particularly for text classification. We used four basic subgroups "POLITICS","SCIENCE","RELIGION" and "COMP" from the 20 Newsgroup dataset. The documents are evenly distributed to the classes, each class includes 500 documents. The sixth dataset we use is the mini newsgroups[3] dataset which has 20 classes and also has a balanced class distribution with 100 documents per class. This is a subset of the 20 Newsgroup[2] dataset, too. The last dataset is a subset of the 20 Newsgroups dataset which is used to compare our method with a semi-supervised classifier from a related study. Properties of these data sets are given in Table 2.

Table 2
Properties of datasets before attribute selection

Dataset	***#classes***	***#instances***	***#features***
IMDB	2	2.000	16.679
20 Newsgroups-POLITICS	3	1.500	2.478
20 Newsgroups-SCIENCE	4	2.000	9.615
20 Newsgroups-RELIGION	3	1.500	2.125
20 Newsgroups-COMP	5	2.500	2.478
MiniNewsGroup	20	2.000	12.112
20 Newsgroups-pcmac	2	1.943	3.290

4.2 Experiment Setting and Evaluation

We apply stemming and stopword filtering to these datasets. Moreover, we refine infrequent words whose frequency is less than three. We also apply attribute selection and select the most informative 2,000 terms using IG as described in [68, 69]. This preprocessing increase the performance of the classifier models by reducing the noise. We perform this preprocessing equally in all experiments.

In order to observe the behavior of our semi-supervised algorithm for each data set, 20% data are reserved as to evaluate the classification performance of learned model, whereas the remaining 80% data are divided into labeled set and unlabeled set. We use 1%, 2%, 3%, 5%, 7%, 10%, 15% and 30% data are used as labeled instances while 79%, 78%, 77%, 75%, 73%, 70%, 65% and 50% of the data are used as unlabeled examples; respectively. Note that the class distributions in these splits are similar to that in the original data set. This is essential since we expect that the benefit of using semantic kernels should be more observable when there is insufficient labeled data.

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

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

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

We tune the misclassification cost (*C*) parameter of SMO [74] to 1. We perform 10 random trials for each training set percentage by randomly selecting the instances to form training set. We report the average of these 10 accuracy results as in [66, 67]. The central evaluation metric in our experiments is the classification accuracy and in the results tables we also report standard deviations. In order to underline the classification performance differences between baseline algorithms and our method we provide performance gain calculated using the Eq. (20);

$$Gain_{HCSC} = \frac{(P_{HCSC} - P_x)}{P_x} \tag{20}$$

where P_{HCSC} is the accuracy of SMO with *HCSC* and P_x shows the accuracy result of the SMO with other kernel. The experimental results are represented in Table 3. This table includes labeled set percentage, unlabeled set percentage, testing set percentage, the accuracy results of linear kernel, baseline algorithms and HCSC. Also the "Gain" column in Table 3 demonstrates the (%) gain of HCSC over baseline algorithm calculated as in Eq. (20).

Additionally, Student's 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 HCSC significantly differs over baseline algorithm based on Students t-Tests, we indicate this with "*".

4.3 Baseline Algorithms

HCSC exploits semantic relatedness of terms among classes to assign labels to unlabeled examples and then merge labeled and unlabeled (with their labels) examples. Thus the baseline algorithm which will be compared to our algorithm is expected to utilize a customary approach to label unlabeled examples. In order to compare the results of HCSC we use four baseline algorithms. First of them is the traditional linear kernel. Please note that linear kernel is the traditional state of the art algorithm in SVM for text classification [43, 73]. The second of the baseline algorithms is called SSL-Linear. SSL-Linear first classifies unlabeled examples by using linear kernel that is trained by only the labeled examples. Then, like HCSC it merges the labeled examples and unlabeled examples with their pre-labels and builds the trainer by using standard linear kernel. After that it again attempts to classify unlabeled examples via the last built model and compares the labels of an instance. If an instance is classified into a different class by the second classifier then its label is updated since the final model is more comprehensive than the first model and is expected to produce predictions with higher classification confidence. The self-training process ends when either there is no change in the predictions or it reached 100 iterations. Our third and fourth baseline algorithms are our previous supervised kernels; CMK [2] and CWK [3]. For comparison, all the baseline algorithms are run on the same labeled/unlabeled/test splits as those used for evaluating HCSC.

5. EXPERIMENTAL RESULTS AND DISCUSSION

HCSC outperforms all of the baseline kernels we used (i.e., linear kernel, SSL-Linear, CMK and CWK) clearly in most of the labeled/unlabeled/test splits on SCIENCE except labeled set percentage 30% which can be observed from Table 3. According to Table 3 at labeled set percentages: 1%, 2% , 3% , 5% ,7%, 10% and 15% the accuracies of HCSC are 56.6%, 65.97%, 67.3%, 85.93%, 89.33%, 90.93% and 92.28% while the accuracies of SSL-Linear are 50.58%, 57.23%, 65.28%, 68.1%,72.65%, 75.22% and 80.8%; respectively. HCSC also has better performance than our previous supervised semantic kernels CMK and CWK at most of the labeled set percentages expect labeled set percentage 30% as shown in Table 3. The highest gain of HCSC over SSL-Linear kernel on this dataset is at 5% labeled set percentage which is 26.18% that is of great importance since usually it is difficult and expensive to get labeled data in real world applications. Moreover it should be observed that, there are performance gains of HCSC over linear kernel at all of the labeled set percentages.

Experiment results on RELIGION dataset show that HCSC has superiority at all labeled set percentages among all of the other kernels except at 30% labeled set percentage. For example at labeled set percentage 3% HCSC' gain over the baseline algorithm is 9.97%. Also, it should be paid attention that in every labeled set percentage between 2% and 30% HCSC shows a significant improvement over SSL-Linear kernel according to Student's t-Tests as presented in Table 3.

Table 3 also contains the experiment results on the POLITICS dataset. In this dataset, HCSC's performance is higher than the performance of linear kernel in all labeled set percentages except 1%. Furthermore, HCSC performs better than SSL-linear in all labeled set percentages. HCSC also gives higher accuracies than both CMK and CWK in almost all labeled set percentages except 30%. Only in labeled set percentage 30%, CMK gives better accuracy than HCSC, but HCSC still remains better than both linear kernel and the SS-Linear at this labeled set percentage as shown in Table 3.

Experiment results on COMP dataset are similar to the experimental results on POLITICS dataset. In COMP dataset, HCSC's performance is higher than the first baseline kernel's and second baseline kernel's in all labeled set percentages. HCSC shows a significant improvement over SSL-Linear kernel at all of the labeled set percentages according to Student's t-Tests as presented in Table 3. Furthermore, HCSC performs better than CMK and CWK in almost all labeled set percentages except 30% on POLITICS

dataset. Only in labeled set percentage 30%, CMK gives better accuracy than HCSC, but HCSC still remains better than both linear kernel and the SS-Linear at this labeled set percentage.

Table 3
Experiment results of HCSC and other kernels

Labeled %	Unlabeled %	Test %	Dataset	Baseline-1: Linear	Baseline-2: SSL-Linear	Baseline-3: CMK	Baseline-4: CWK	HCSC	Gain (over Base-line-2)
1	79	20	SCIENCE	51.80±5.33	50.58±5.41	39.42±6.78	51.4±6.84	**56.6±14.56**	11.90*
2	78	20		59.10±5.49	57.23±5.95	50.30±6.00	65.38±4.59	**65.97±11.49**	15.27*
3	77	20		66.03±3.61	65.28±3.77	53.40±7.78	67.13±5.23	**67.30±10.3**	3.090
5	75	20		70.10±4.34	68.1±4.96	70.03±5.07	81.63±3.92	**85.93±2.42**	26.18*
7	73	20		72.72±4.47	72.65±3.47	78.53±5.07	85.5±3.03	**89.33±2.93**	22.96*
10	70	20		76.68±2.07	75.22±3.58	87.48±4.81	90.1±2.11	**90.93±2.57**	20.89*
15	65	20		83.53±2.68	80.8±2.74	89.95±1.71	92.13±1.28	**92.98±1.77**	15.07*
30	50	20		86.28±2.27	84.13±1.99	**95.28±0.95**	94.88±1.37	95.25±1.31	13.22*
1	79	20	RELIGION	49.67±7.15	49.37±7.61	40.3±4.22	49.63±7.63	**50.3±5.56**	1.88
2	78	20		61.17±5.88	59.77±5.7	49.63±6.45	59.17±6.67	**63.07±5.27**	5.52*
3	77	20		67.50±5.05	65.93±5.41	56.6±5.25	67.93±5.42	**72.50±2.87**	9.97*
5	75	20		73.00±2.50	70.43±3.25	74.07±5.26	74.63±3.59	**77.13±4.10**	9.51*
7	73	20		75.47±4.11	73.87±5.44	76.47±4.94	79.37±2.66	**81.13±3.25**	9.83*
10	70	20		80.00±2.60	78.4±2.5	80.57±3.01	83.3±2.7	**84.73±3.24**	8.07*
15	65	20		81.80±3.12	80.6±2.94	86.63±2.69	87.1±2.6	**88.37±1.95**	9.64*
30	50	20		87.17±2.22	86±2.36	**92.37±1.36**	92.17±1.85	91.97±1.74	6.94*
1	79	20	POLITICS	**52.60±5.69**	50.9±5.98	38.60±2.26	49.53±3.35	51.87±6.75	1.910
2	78	20		64.60±6.34	62.6±6.07	47.97±4.64	66.57±3.98	**75.83±5.83**	21.13*
3	77	20		69.60±5.28	68.4±6.1	64.60±10.43	74.83±3.08	**80.40±5.03**	17.54*
5	75	20		73.23±3.92	72.3±3.29	78.03±4.46	81.43±2.75	**84.50±2.53**	16.87*
7	73	20		78.33±5.30	77.07±4.81	80.03±5.24	84.53±3.36	**89.17±2.67**	15.70*
10	70	20		82.00±2.38	81.1±2.09	87.13±2.13	88.3±1.32	**90.17±1.50**	11.18*
15	65	20		84.67±4.92	84.07±4.81	91.50±1.69	91.4±1.49	**92.33±1.54**	9.830*
30	50	20		90.07±1.91	87.73±2.47	**94.43±1.05**	94.4±0.93	94.20±0.96	7.370*
1	79	20	COMP	36±4.51	31.2±3.45	32.34±3.93	33.44±1.32	**47.08±6.37**	15.88*
2	78	20		41.22±3.93	35.56±2.98	35.9±4.49	37.9±2.93	**50.82±7.02**	15.26*
3	77	20		46.02±3.4	37.38±2.66	44.9±5.95	48.3±2.54	**60.44±6.17**	23.06*
5	75	20		50.36±4.35	43.5±2.99	54.06±8.96	57.62±3.66	**69±1.83**	25.50*
7	73	20		57.2±3.07	47.06±2.28	61.78±4.21	64.86±0.13	**71.08±3.27**	24.02*
10	70	20		62.12±3.43	52.92±3.08	72.54±1.71	73.42±1.14	**75.52±3.34**	22.60*
15	65	20		66.04±3.28	54.88±2.58	77.82±2.66	78.25±0.36	**80.48±1.82**	25.60*
30	50	20		73.94±3.26	64.02±1.89	**83.62±2.26**	84.21±1.64	83.58±2.03	19.56*
1	79	20	Mini News-Groups	36.70±4.24	31.7±4.16	19.23±2.59	31.25±3.98	**43.10±7.54**	35.96*
2	78	20		38.42±5.47	31.68±4.63	18.43±2.48	32.17±3.93	**39.75±6.04**	25.47*
3	77	20		46.33±4.32	39.42±5.38	26.63±3.43	38.38±3.62	**52.25±2.28**	32.55*
5	75	20		50.00±5.49	42.53±5.64	35.78±3.15	49.13±3.05	**61.68±1.94**	45.03*
7	73	20		55.15±4.72	49.15±5.93	47.23±3.18	57.18±2.44	**68.08±4.32**	38.51*
10	70	20		57.03±2.79	49.68±3.72	53.65±3.27	64.88±2.63	**71.65±1.86**	44.22*
15	65	20		62.48±4.28	56.03±4.16	61.83±3.24	71.5±3.59	**71.80±2.99**	28.15*
30	50	20		69.55±4.50	64.88±4.12	70.68±3.38	**77.72±1.49**	74.22±2.29	14.40*
1	79	20	IMDB	**65.75±7.79**	65.6±8.37	61.33±6.4	61.65±5.73	62.83±1.45	-2.77
2	78	20		69.3±3.28	70.13±2.82	67.97±6.57	68.25±5.17	**71.08±0.89**	0.95
3	77	20		72.8±3.28	71.15±3.83	74.25±3.91	**74.6±4.52**	73.35±6.15	2.20
5	75	20		77.03±2.55	75.88±2.78	80.33±3.79	**80.45±3.45**	79.33±5	3.45
7	73	20		79.45±1.76	78.53±2.49	82.38±1.88	**82.68±1.91**	82.83±2.73	4.30
10	70	20		79.7±2.86	78.83±3.28	**84.65±1.94**	84.23±1.73	83.88±1.94	5.05*
15	65	20		81.72±2.47	80.35±1.42	**86.98±1.57**	86.65±1.51	85.55±1.66	5.20*

30	50	20	85.8±1.37	84.78±1.29	**90.88±1.28**	89.63±0.94	88.33±1.14	3.55

Table 3 presents the experiment results on MiniNewsGroup dataset. According to these results HCSC outputs better accuracy than SSL-Linear kernel at all labeled set percentages also making a significant difference based on Students t-Tests. For instance at labeled set percentage 5% the performance gain of HCSC over SSL-Linear is 45.03% which is of great importance because of the scarcity of labeled data in real world applications. Also it should be noted that, HCSC performs better than CMK and CWK in almost all labeled set percentages except the labeled set percentage 30% on MiniNewsGroup dataset. Only in labeled set percentage 30%, CWK gives better accuracy than HCSC, but HCSC still remains better than linear kernel, SS-Linear and CMK at this labeled set percentage.

Table 3 also contains the experiment results on IMDB dataset. In IMDB dataset, HCSC's performance is superior to both linear kernel and SSL-Linear in all labeled set percentages except the labeled set percentage 1%. Furthermore HCSC gives better accuracies than SSL-Linear at the labeled set percentages 10% and 15% with yielding a significant difference based on Student's t-Tests on IMDB dataset. According to the experimental results on Table 3, CWK gives better accuracies than HCSC at labeled set percentages 3%, 5% and 7% while CMK gives better accuracies than HCSC at labeled set percentages 10%, 15% and 30%. The different trend of the accuracies on this dataset might be depending on the type of the dataset and the term coverage at labeled set percentages. The classification accuracies of all the algorithms on COMP and RELIGION datasets are shown in Fig. 5 and Fig. 6.

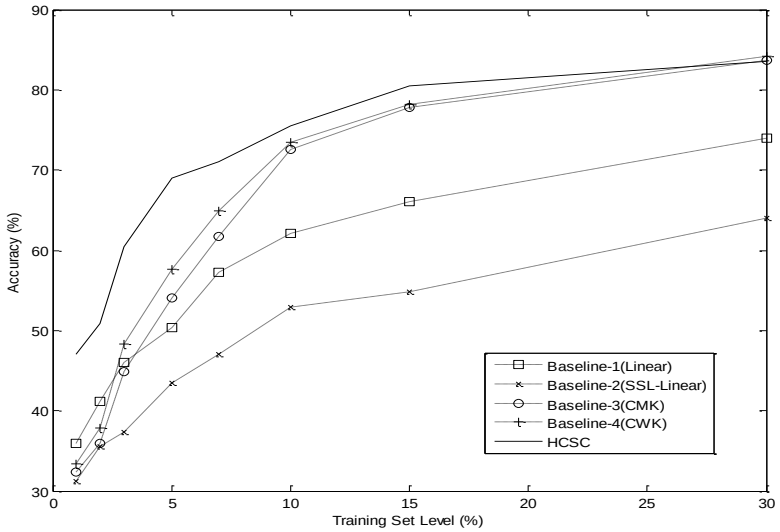

Fig. 5. Comparison of Accuracies at Different Training Set Levels on COMP Dataset

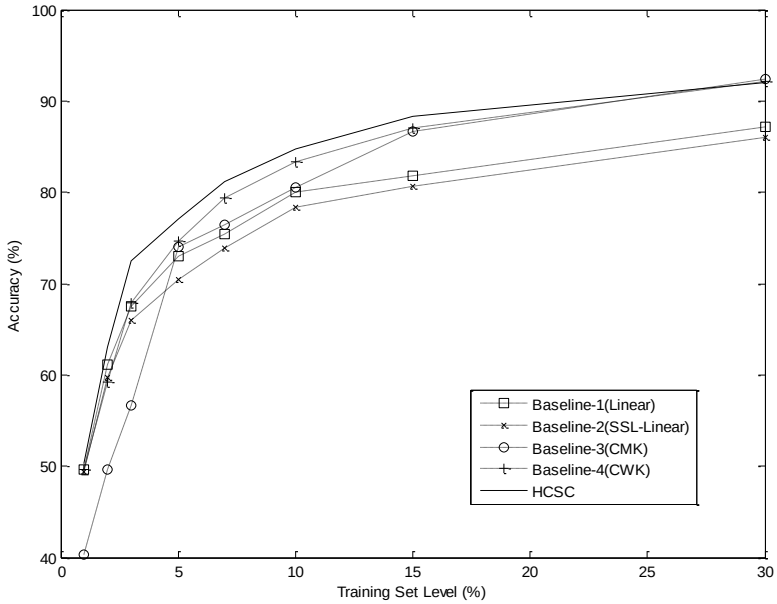

Fig. 6. Comparison of Accuracies at Different Training Set Levels on RELIGION Dataset

In terms of scalability, first part of algorithm is quite scalable. Populating a term-class matrix by calculating the meaning values for each term in the scope of each class is comparable with the training phase of Naïve Bayes algorithm, which is one of the fastest classifiers. On the classification phase, we sum the class based meaning values of the terms that exist in the test document for each class and compare the sums. Again it is comparable with the Naïve Bayes and consequently quite efficient. Second part of the algorithm uses the SVM classifier with a supervised semantic kernel. Our semantic kernel is obtained using again a term-class matrix multiplications. This time the term-class matrix is obtained using class based term weighting. This constitutes the main complexity in SVM training phase. However, it is important to note that the number of classes is usually smaller than the number of terms in many text classification problems and matrix multiplications can be easily made parallel to reduce the runtimes of experiments.

We also compare HCSC with additional methods from related work to confirm the effect of our proposed kernel. In [75] authors present a new algorithm for semi-supervised learning problems involving large, sparse datasets. The study in [75] is based on a Deterministic Annealing (DA) approach which is significantly more efficient and scalable than currently used dual techniques. DA is actually a variant of TSVM and involves multiple switching of labels [75]. We choose pcmac[1] dataset, a small subset of the 20 Newsgroups data popularly used in semi-supervised learning literature (e.g. in [12]), from this work. We prepared this dataset exactly with the same number of instances (labeled instances, unlabeled instances, test instances) as in the study [75]. After that we conduct this dataset to our experimental environment. We run our algorithm on this dataset and report the results as the mean of 10 random runs like it is done in [75]. The authors in [75] compared DA to SVM and TSVM. We also compared our results to not only DA but also SVM and TSVM. The experiment results are shown in Table 4 and Table 5.

Table 4 demonstrates the gain differences of DA and HCSC over SVM based on classification performances with the corresponding number of labeled, unlabeled and test instances. The first three columns in Table 4 presents the number of labeled, unlabeled and test instances; respectively. The fourth column shows the performance gain difference of DA over SVM results that is reported in the study and the fifth column shows the performance gain difference of HCSC over SVM results in our experiments. Finally the last column in Table 4 presents the differences between these performance gains of DA and HCSC over SVM. According to Table 4 HCSC shows better performance compare to DA for all of the test cases reported in Table 4. For instance, the classification performance gain of HCSC over SVM is 16.49 while the classification performance gain of DA over SVM is 15.61 with 37 labeled instances and 1423 unlabeled instances. The performance gains are also shown in Fig. 7.

Table 4
Gain of DA[75] and HCSC over respective SVM baselines on pcmac dataset

# Labeled Instances	***#Unlabeled Instances***	***#Test Instances***	***Gain DA over SVM***	***Gain HCSC Over SVM***	***Gain Difference Between HCSC and DA***
37	1423	486	15.61	16.49	0.88
73	1387	486	8.08	9.57	1.49
110	1350	486	5.65	6.71	1.06
146	1314	486	3.59	5.43	2.14
183	1277	486	3.24	5.30	2.06
220	1240	486	2.57	4.89	2.32

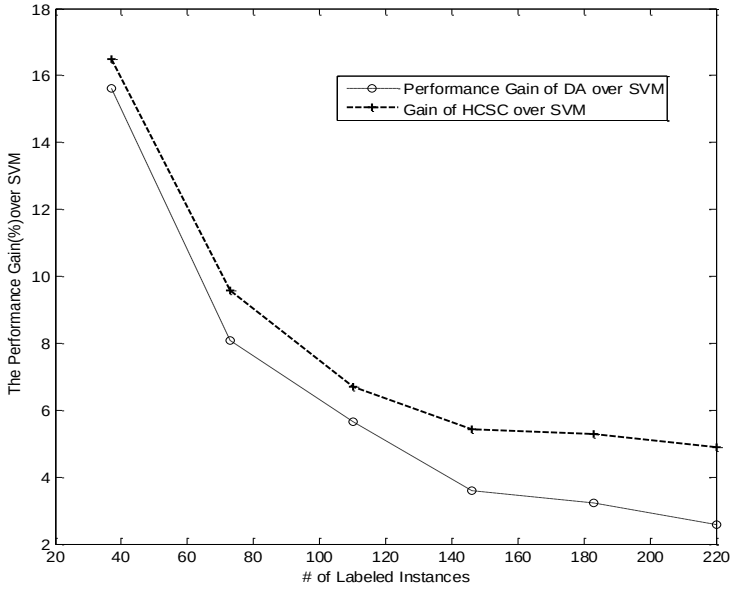

Fig. 7. The Performance Gain of DA and HCSC over SVM on pcmac Dataset

[1] http://vikas.sindhwani.org/datasets/lskm/matlab/

Table 5 demonstrates the gain differences of DA and HCSC over TSVM based on classification performances with the corresponding number of labeled, unlabeled and test instances. The first three columns in Table 5 presents the number of labeled, unlabeled and test instances; respectively as similar to Table 4. The fourth column shows the performance gain difference of DA over TSVM and the fifth column shows the performance gain difference of HCSC over TSVM. Finally the last column in Table 5 presents the differences between these performance gains of DA and HCSC over TSVM.

According to Table 5, in terms of accuracy improvements over their respective SVM results as baselines, HCSC is superior to both DA and TSVM for all of the test cases reported in Table 5. For instance, the classification performance gain of HCSC over TSVM is 3.14 while the classification performance gain of DA over TSVM is 2.38 with 37 labeled instances, 1423 unlabeled instances. The performance gains are also shown in Fig. 8.

Table 5
Gain of DA[75] and HCSC over TSVM and TSVM(S=max) on pcmac dataset

# Labeled Instances	***#Unlabeled Instances***	***#Test Instances***	***Gain DA over TSVM***	***Gain HCSC over TSVM***	***Gain Difference Between HCSC and DA***
37	1423	486	2.38	3.14	0.76
73	1387	486	2.04	3.01	1.42
110	1350	486	1.49	2.81	1.02
146	1314	486	1.06	2.85	1.79
183	1277	486	0.74	2.75	2.01
220	1240	486	0.42	2.69	2.27

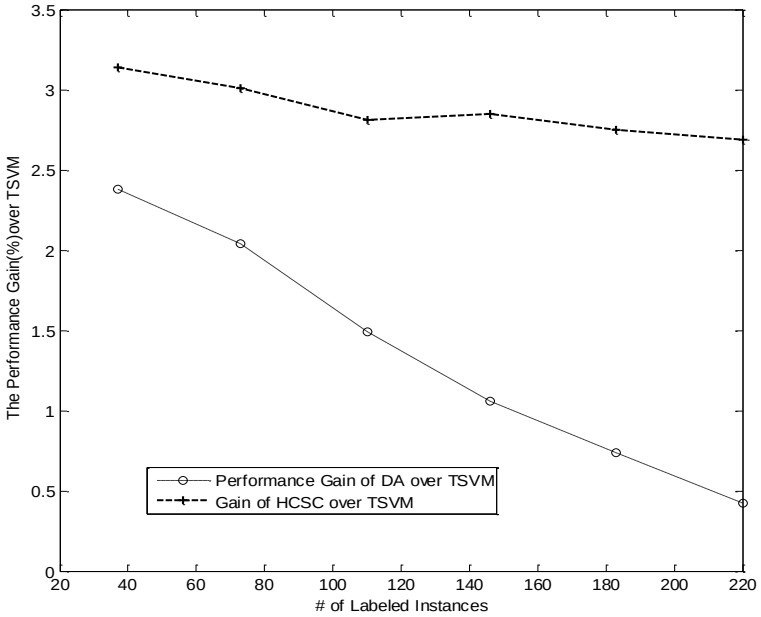

Fig. 8. The Performance Gain of DA and HCSC over TSVM on pcmac Dataset

6. CONCLUSIONS AND FUTURE WORK

Vector Space Models (VSM) are commonly used language processing to represent some aspects of natural language semantics. In this model, documents are simply represented as points in a space and proximity of two points is proportional to their semantic similarity [5]. Semantics of VSMs comes from the distributional hypothesis, which states that words that occur in similar contexts usually have similar meanings [1]. From this point of view, the hybrid semi-supervised text classification algorithm that we propose consists of a novel combination of two supervised semantic classification approaches. Both of these approaches can be considered as a new type of VSM since they make use of term-class matrices. We use distributional hypothesis by setting the context as the class and effectively making use of class based term transformations. In our case distributional hypothesis states that words occur in similar classes will usually have similar meanings. We call it class-based semantics. As a result, by setting the context as the class of documents, our individual methods can extract semantic relations between terms much better than traditional VSMs using the document context for text classification. Furthermore, our semi-supervised algorithm called Hybrid Class Semantics Classifier (HCSC), the novel combination and slight modification of these methods can achieve much better performance by exploiting the unlabeled data.

The meaning values used to populate term-class matrix that is used for labeling unlabeled documents are calculated according to the Helmholtz Principle, which is mainly based on Gestalt theory and already, has some practical applications on several text mining problems including document summarization, and feature extraction [16, 17, 18, 19]. The labeled documents are combined with training set to form a much larger but at the mean time noisier training set. This in turn used for training another class-term matrix based algorithm (CWK [3]) which employs term by class matrix as a supervised semantic kernel in SVM. CWK predicts the labels of the test instances by using the model built in the training phase. In our CWK, we use term weights to smoothen the document term vectors. The main contribution of our approach is; a non-iterative yet effective way of assigning labels to unlabeled instances and augmenting the training set with them in order to build a better performing model using the Class Weighting Kernel (CWK) which is recently proposed in our previous study [3]. It can be considered as the semi-supervised version of the CWK and also a hybrid version of SMC [4] and CWK [3].

Our experimental results show the promise of the HCSC as a semi-supervised method that utilizes class based semantics in the text classification domain. We use four baseline algorithms, namely linear kernel, SSL-Linear CMK and CWK, in order to compare the results of HCSC. According to our experimental results HCSC outperforms both SVM based supervised baselines in most of our experiments. The HCSC also outperforms our previous supervised class based semantic algorithms; CMK [2] and CWK in most of the datasets. In other words, HCSC achieves higher classification accuracy by adding unlabeled data into the usually small amount of labeled. Our experimental results show that HCSC outperforms our two baseline kernels, linear kernel and SSL-Linear for almost all of the test cases on each datasets. Additionally, HCSC achieves higher accuracies than our previous

semantic kernels for the most of the labeled set percentages on all of the datasets except the IMDB dataset. We attribute this exception to the relatively much lower number of classes in this dataset. Since both of the approaches in our hybrid algorithm use term-class based VSMs, the number of classes seems to play a crucial role in class based transformations. Our promising experimental results shows that we succeed in building a non-iterative semi-supervised version of CWK [3] that can benefit from unlabeled data.

Moreover, the HCSC forms a foundation that is open to several improvements. For example, the HCSC 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. Additionally, the initial labeling of unlabeled data can be improved by using an iterative approach as in many traditional SSL studies. As the future work, we would like to develop an iterative version of HCSC and compare it with the traditional SSL studies. In addition to this, 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.

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] Harris, Z. S., 1954. Distributional structure. Word, 10(2-3), pp. 146-162.

[2] Altınel, B., Ganiz, M.C., Diri, B., 2015. A Corpus-Based Semantic Kernel for Text Classification by Using Meaning Values of Terms. Engineering Applications of Artificial Intelligence , (43), pp. 54-66.

[3] Altınel, B., Diri, B., Ganiz, M.C., 2015. A Novel Semantic Smoothing Kernel for Text Classification with Class-based Weighting. Knowledge-Based Systems ,89, pp. 265-177.

[4] Ganiz, M.C., Tutkan, M., Akyokuş, S., 2015. A Novel Classifier Based on Meaning for Text Classification. Innovations in Intelligent Systems and Applications , September 2-4, Madrid, Spain.

[5] Turney, P. D., & Pantel, P., 2010. From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research, 37(1), 141-188.

[6] Zhu, X. J., 2005. Semi-supervised learning literature survey, Technical Report, Department of Computer Sciences, University of Wisconsin at Madison, Madison, WI, pp. 1530, 2006.

[7] Blum, A. and Mitchell, T., 1998. Semi-supervised learning literature survey., Proc. Conf. on Computational Learning Theory, pp. 92-100.

[8] Yarowsky, D., 1995. Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, pp. 189–196.

[9] Rosenberg, C. et al.,2005. Semi-supervised self-training of object detection models, In Proc. 7th Workshopn on Applications of Computer Vision, (1), pp. 29-36.

[10] Zhou, D. et al., 2004. Semi-supervised learning on directed graphs, Advances in Neural Information Processing Systems, 16, pp. 1633-1640.

[11] Nigam, K. et al., 2000. Text classification from labeled and unlabeled documents using EM, Machine Learning, 39(2/3), pp. 103-134.

[12] Chapelle, O. and Zien, A., 2005. Semi-supervised classification by low density separation, Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, pp. 57-64.

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

[14] 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.

[15] Steinbach, M. Karypis, G., Kumar, V., 2000. A Comparison Of Document Clustering Techniques. In: Proceedings of the KDD Workshop on Text Mining, 400 (1), pp. 525-526.

[16] Balinsky, A., Balinsky, H., Simske, S., 2010. On the Helmholtz principle for Documents Processing. In: Proceedings of the 10th ACM Document Engineering (DocEng), pp. 283-286..

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

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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

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

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

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

[27] 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, 5, pp. 205-209.

[28] Goldman, S. and Zhou, Y., 2000. Enhancing supervised learning with unlabeled data , In: Proceedings of the 17th ICML, San Francisco, CA, Morgan Kaufmann, pp. 327-334.

[29] Jin, Y., Huang, C., & Zhao, L., 2011. A Semi-Supervised Learning Algorithm Based on Modified Self-training SVM. Journal of Computers, 6(7), pp.1438-1443.

[30] Grira, N., Crucianu, M. and Boujemaa, N., 2004. Unsupervised and semi-supervised clustering: A brief survey. Report of the MUSCLE European Network of Excellence (FP6), pp. 1-12.

[31] Nigam, K. and R. Ghani, 2000. Analyzing the effectiveness and applicability of co-training. In Proceedings of the 9th ACM International Conference on Information and Knowledge Management, Washington, DC, pp. 86–93,.

[32] Muslea, I., Minton, S., Knoblock, C.A., 2002. Active semi-supervised learning in robust multi-view learning. In: Proceedings of the Nineteenth International Conference on Machine Learning.

[33] Liu, A., Jun, G., Ghosh, J., 2009. A self-training approach to cost sensitive uncertainty sampling. Machine Learning 76, pp. 257–270.

[34] Chapelle, O., Scholkopf, B., Zien, A. (eds.), 2006. Semi-supervised learning. MIT Press, Cambridge.

[35] Wang, B., Spencer, B., Ling, C.X., Zhang, H., 2008. Semi-supervised self-training for sentence subjectivity classification. In: The 21st Canadian Conference on Artificial Intelligence, pp. 344–355.

[36] Li, M., & Zhou, Z. H., 2005. SETRED: Self-training with editing. In Advances in Knowledge Discovery and Data Mining, pp. 611-621, Springer Berlin Heidelberg.

[37] Guo, Y., Zhang, H., Liu, X., 2011. Instance selection in semi-supervised learning. In: Proc. 24th Canadian Conference on Artificial Intelligence. pp. 158–169.

[38] Li, K., Zhang, W., Ma, X., Cao, Z., & Zhang, C. , 2008. A novel semi-supervised SVM based on tri-training. In Intelligent Information Technology Application, 2008. IITA'08. Second International Symposium, IEEE, 3, pp. 47-51.

[39] Cozman, F.G. et al., 2003. Semi-Supervised Learning of Mixture Models, Proceedings of the Twentieth International Conference on Machine Learning(ICML-2003), pp. 99-106.

[40] Guo, Y., Niu, X., Zhang, H., 2010. An extensive empirical study on semi-supervised learning. In: The 10th IEEE International Conference on Data Mining, pp. 186-195.

[41] Li, Y.F., Kwok, J.T., Zhou, Z.H., 2010. Cost-sensitive semi-supervised support vector machine. In: Proc. 24th AAAI Conference on Artificial Intelligence. pp. 500–505.

[42] Cohen, I., Cozman, F.G., Sebe, N., Cirelo, M.C., Huang, T.S., 2004. Semi-supervised learning of classier: theory, algorithms, and their application to human-computer interaction. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, pp.1553-1567.

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

[44] Cambria, E., Speer, R., Havasi, C. and Hussain, A., 2010. SenticNet: A Publicly Available Semantic Resource for Opinion Mining., AAAI fall symposium: Commonsense Knowledge,10, pp. 2216-2217.

[45] Cambria, E., Olsher, D. and Rajagopal, D., 2014. SenticNet 3:a common and common-sense knowledge base for cognition-driven sentiment analysis, in: AAAI, Quebec City, 2014, pp. 1515–1521.

[46] Cambria, E., Hussain, A., Havasi, C., and Eckl, C., 2010. Sentic Computing: Exploitation of Common Sense for the Development of Emotion-Sensitive Systems, volume 5967 of LNCS. Berlin Heidelberg: Springer-Verlag, pp. 148–156.

[47] Cambria, E., Hussain, A., 2015. Sentic Computing: A Common-Sense-Based Framework for Concept-Level Sentiment Analysis. Springer, Cham.

[48] Cambria, E., Hussain, A., Havasi, C. and Eckl, C., 2009b. Common Sense Computing: From the Society of Mind to Digital Intuition and Beyond, volume 5707 of LNCS. Berlin Heidelberg: Springer–Verlag, pp. 252–259.

[49] Cambria, E., Grassi, M., Hussain, A. and Havasi, C., 2012. Sentic computing for social media marketing. Forthcoming, 59(2), pp. 557-577.

[50] Cambria, E., Hussain, A., Havasi, C., Eckl, C. and Munro, J., 2010a. Towards crowd validation of the uk national health service. In WebSci10, pp. 1-5.

[51] Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T. and Zhu, W. L., 2002. Open Mind Common Sense: Knowledge acquisition from the general public. In On the Move to Meaningful Internet Systems, DOA/CoopIS/ODBASE 2002 Confederated International Conferences DOA, CoopIS and ODBASE 2002, pp. 1223–1237, London, UK. Springer-Verlag.

[52] Speer, R. and Havasi, C., 2012. Representing general relational knowledge in Conceptnet 5. In: International conference on language resources and evaluation(LREC), pp. 3679—3686.

[53] Strapparava, C., and Valitutti, A., 2004. Wordnet-affect: an affective extension of wordnet. In LREC04, 4, 1083-1086.

[54] Cambria, E., Fu, J., Bisio, F., Poria, S., 2015. Affectivespace 2: Enabling affective intuition for concept-level sentiment analysis. In: Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 508–514.

[55] Kleinberg, J. 2002. Bursty and Hierarchical Structure in Streams. In Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), 7(4), pp.373-397.

[56] Dadachev, B., Balinsky, A. Balinsky, H.; Simske, S. 2012. On the Helmholtz Principle for Data Mining. In International Conference on Emerging Security Technologies (EST), pp.99 – 102.

[57] Jones, K. S., 1972. A Statistical Interpretation Of Term Specificity And Its Application In Retrieval. Journal of documentation 28(1), pp. 11-21.

[58] 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, 87787877SA: ACM Press, 2003, pp. 784-788.

[59] 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.

[60] 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), pp.721-735.

[61] 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, pp. 453-459.

[62] 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), 2, pp. 1171-1176.

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

[64] 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.

[65] 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.

[66] 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), 8467, pp.505–517.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] 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.

[72] Hall, M, Frank, E. Homes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The WEKA Data Mining Software: An Update; SIGKDD Explorations, 11 (1), pp. 10-18.

[73] 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.

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

[75] Sindhwani, V. and Keerthi, S., 2006. Large Scale Semi-supervised Linear SVMs, Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 477-484.

[76] Cambria, E., Hussain, A., Havasi, C. and Eckl, C., 2009a. Affectivespace: Blending common sense and affective knowledge to perform emotive reasoning. In CAEPIA09, pp. 32-41.