A Semantic Kernel for Text Classification Based on Iterative Higher–Order Relations between Words and Documents

Berna Altinel[1], Murat Can Ganiz[2], Banu Diri[3]

[1]Department of Computer Engineering, Marmara University, Istanbul, Turkey
berna.altinel@marmara.edu.tr
[2]Department of Computer Engineering, Dogus University, Istanbul, Turkey
mcganiz@dogus.edu.tr
[3]Department of Computer Engineering, Yildiz Technical University, Istanbul, Turkey
banu@ce.yildiz.edu.tr

Abstract. We propose a semantic kernel for Support Vector Machines (SVM) that takes advantage of higher-order relations between the words and between the documents. Conventional approach in text categorization systems is to represent documents as a "Bag of Words" (BOW) in which the relations between the words and their positions are lost. Additionally, traditional machine learning algorithms assume that instances, in our case documents, are independent and identically distributed. This approach simplifies the underlying models, but nevertheless it ignores the semantic connections between words as well as the semantic relations between documents that stem from the words. In this study, we improve the semantic knowledge capture capability of a previous work in [1], which is called χ-Sim Algorithm and use this method in the SVM as a semantic kernel. The proposed approach is evaluated on different benchmark textual datasets. Experiment results show that classification performance improves over the well-known traditional kernels used in the SVM such as the linear kernel (one of the state-of-the-art algorithms for text classification system), the polynomial kernel and the Radial Basis Function (RBF) kernel.

Keywords: machine learning, support vector machine, text classification, higher-order paths, semantic kernel.

1 Introduction

Text categorization is a popular task which aims to label documents via using predefined category labels. There are large amounts of textual data accumulated both in organizations and on the internet especially on social networks, microblogging sites, blogs, forums, new, etc. This huge set of documents continues to increase by the contributions of millions of people every day. Automatically processing and extracting meaning from these increasing amounts of textual data is one of the most important problems for both research and commercial entities. The text classification plays a very important role in several popular and widely used applications such as document filtering, sentiment classification, information extraction, summarization and question answering.When processing textual data, either in information retrieval or text classification; it is common to use the bag of words (BOW) feature representation. In this approach the documents are represented only by the occurrences or the frequencies of the words or terms independent from their positions in the document. Although this approach is very popular due to its simplicity, it has several drawbacks. First of all, it breaks multi-word expressions into pieces, secondly it treats synonymous words as different terms; and thirdly it treats polysemous words (i.e., words with multiple meanings) as one single component, as it is mentioned in [2]. However, in order to enhance the prediction capabilities of text classification algorithms, it is important to benefit from the semantic relations between the words and even between the documents.

In this study, we introduce a new kernel for Support Vector Machines (SVM) called *Normalized Iterative-Higher-Orders Semantic Kernel* (N-IHOSK) which is based on higher-order paths between documents as well as the terms. Our approach is motivated by the studies of higher-order Naïve Bayes [3], [4] and Higher-Order Smoothing [5], [6] which makes use of the higher-order paths between terms, and recently introduced work of [7] which focus on the higher-order paths between documents. In this study, we improve the semantic knowledge capture capability of a previous work in [1], which is called χ-Sim algorithm and use this method in the SVM as a semantic kernel. Our target is to capture latent semantic information between the terms and between the documents. In our experiments, our proposed framework is compared with other traditional kernel methods for SVM such as linear kernel, polynomial kernel and Radial Basis Function (RBF) kernel. It is important to note that SVM with linear kernel is one the state of the art algorithms for text classification [8], [9].

These traditional kernels can be considered as first-order methods since their context is a single document and they model just the first-order co-occurrences of the terms. However, N-IHOSK can make use of the higher-order paths that include several different terms and documents in the context of the whole dataset. Our experiments running the N-IHOSK on several benchmark datasets show that the classification performance of SVM improves considerably over the first-order kernels.

The remainder of the paper is organized as follows: background information and related work including the SVM, semantic kernels and higher-order paths are summarized in Section 2. Section 3 presents and analyzes the proposed kernel for text classification algorithm. The experiment setup and corresponding results including some discussion points are given in Section 4. Finally, in Section 5 we provide the conclusion and the future work.

2 Background Information and Related Work

2.1 Support Vector Machines for Classification Problem

The SVM in general is a linear classifier which finds the optimal separating hyperplane between the classes. It is possible to use a kernel function in SVM which can map the data into a higher dimensional feature space if it is not possible to find a hyperplane in the original space [8]. We can consider a kernel function as a kind of similarity function, which can give the similarity of data points in the original space. Therefore, defining a suitable kernel is a direct way of finding a good representation of these data points as it is mentioned in [2], [10] and [11]. The SVM algorithm which is first introduced by Vapnik, Guyon and Boser [12] in 1992, has become one of the popular algorithms in real-world-problems producing good accuracies even with high-dimensional and sparse data [8]. Although the SVM is a binary classifier by its nature, it can be used for multi-class categorization using "one-against-the-rest" or "one-against-one" strategies [13]. Because of these benefits the SVM with linear kernel is one of the state of the art algorithms in text classification domain since textual data represented using BOW approach is very high-dimensional and quite sparse. Thus, considering the nature of the text classification (high-dimensional and sparse data), we decided to design a higher-order semantic kernel for SVM.

2.2 Semantic Kernels for Text Classification

According to the definition mentioned in [12], [10], and [2] and [14], any function in the following form (Eq.1) is a valid kernel function.

$$k(d_1, d_2) = \langle \phi(d_1), \phi(d_2) \rangle \quad (1)$$

In Eq.1, d_1 and d_2 are input space vectors and ϕ is a suitable mapping from input space into a feature space.

In [10], Siolas et al. propose a semantic kernel which is based on WordNet [15], which could be seen as a semantic network, for getting the term similarity information. In their work an estimation of two words semantic relation is supplied by WordNet's hierarchical tree structure. The authors in [10] have included this knowledge into the definition of Gaussian kernel. Their results show that the existence of semantic proximity metric increases the classification accuracy in SVM [10]. However, their approach treats multi-word concepts as single terms and does nothing to handle polysemy.

Semantic kernels with super concept declaration were studied in [14]. The aim of their work is to create a kernel algorithm which includes the topological knowledge of their super concept expansion. They apply this mapping with the help of a semantic smoothing matrix **Q** that is shown to be composed of **P** and $\mathbf{P^T}$ which includes super-concept information about their corpus. The proposed kernel function is given in Eq. 2.

$$k(d_1, d_2) = d_1 \cdot P \cdot P^T \cdot d_2^T \quad (2)$$

Their results show that they get a coherent progress in performance for super-concept semantic smoothing kernels in those cases in which little training data exists or the feature representations are highly sparse [14]. However their experiments were kept introductory and did not use a word sense disambiguation strategy. [14]

Similarly, in [16] the WordNet is used as a semantic information resource. But they ([10], [14] and [16]) stated that the coverage of WordNet is not sufficient and as a result, several following studies focused on information sources of wider coverage such as Wikipedia [1].

[1] http://www.wikipedia.org/

Wang et al. [2] combined the background knowledge gathered from Wikipedia into a semantic kernel for enriching the representation of documents. The similarity value between two documents in their kernel function formed as in Eq.3 where **S** is a semantic matrix which is created as a composition of the

contributions from Wikipedia, d_1 and d_2 are term-frequency vectors of documents d_1 and d_2, respectively. This composed **S** matrix consists of three measures. First of them is a content-based measure which is based on the BOW representation of Wikipedia articles. Second measure is the out-link-category-based measure which gives an information related to the out-link categories of two associative articles [2]. Third measure is a distance measure that is calculated as the length of the shortest path connecting the two categories of two articles belong to, in the acyclic graph schema of Wikipedia's category taxonomy [2].

$$k(d_1, d_2) = d_1 \cdot S \cdot S^T \cdot d_2 \qquad (3)$$

Their method is stated to overcome the shortages of the BOW approach. Their results demonstrate adding semantic knowledge into document representation by means of Wikipedia improves the categorization accuracy.

2.3 Iterative Higher-Order Relations between Words and Documents

Illustration of using the higher-order paths is given in Table 1. There are three documents, d_1, d_2 and d_3 which include sets of terms $\{t_1, t_2\}$, $\{t_3\}$ and $\{t_2, t_3\}$ are depicted. With a classical similarity measure which uses the number of shared terms (e.g. the dot product), the similarity value between documents d_1 and d_2 (in Table 1) is calculated as zero since they do not share any terms. But in fact these two documents are similar to a certain degree through d_3. So using a higher-order approach, it is possible to obtain a similarity value between d_1 and d_2 which is larger than zero. We can explain this phenomenon with the statement that two documents are written about the same topic using two different but semantically closer sets of terms. In this case the terms belonging to each set frequently co-occur in other documents relating to this topic, forming a connection pattern which can be revealed by using second-order paths.

Table 1. Illustration of Higher-Order Paths

D	***t₁***	***t₂***	***t₃***
d_1	1	1	0
d_2	0	0	1
d_3	0	1	1

In our study we are motivated by the work of [4] which uses higher-order paths between terms to exploit latent semantics and by the work of [1] which builds iterative higher-order-paths between documents and terms. In [1], the authors devise an iterative method to learn the similarity matrix between documents using similarity matrix between terms and vice-versa. They build a co-similarity algorithm which is called χ-Sim. The document similarity matrix is generated iteratively using **SR** (a similarity matrix between documents) and **SC** (a similarity matrix between terms). The major steps of their algorithm are described below:

1. Initialize the similarity matrices **SR** (documents) and **SC** (words) with the identity matrix **I**. This is a reasonable starting point since similarity between a document (or a term) and itself equals one and it equals to zero in the other cases. They denote these matrices as $\mathbf{SC_0}$ and $\mathbf{SR_0}$. [1]

2. At each iteration *t*, they calculate a new similarity matrix between documents $\mathbf{SR_t}$ by using the similarity matrix between words $\mathbf{SC_{t-1}}$ previously computed. They use the Hadamard product (denoted by "•") in order to multiply their similarity values with normalized weights by the normalization matrix **NR**. [1]

Their $\mathbf{SR_t}$ and $\mathbf{SC_t}$ formulas are given as

$$\mathrm{SR}_t = (\mathrm{D} \,.\, \mathrm{SC}_{(t-1)} \,.\, \mathrm{D}^{\mathrm{T}}) \bullet \mathrm{NR} \quad \text{with } \mathrm{nr}_{i,j} = \frac{1}{|\mathrm{r}_i| \,.\, |r_j|} \qquad (4)$$

$$\mathrm{SC}_t = (\mathrm{D}^{\mathrm{T}} \,.\, \mathrm{SR}_{(t-1)} \,.\, \mathrm{D}) \bullet \mathrm{NC} \quad \text{with } \mathrm{nc}_{i,j} = \frac{1}{|\mathrm{r}_i| \,.\, |r_j|} \qquad (5)$$

where **D** is the document corpus, $\mathbf{D^T}$ is the transpose of **D** matrix, **SR** is row (document) similarity matrix, **SC** is column (word) similarity matrix, and **NR** and **NC** are row and column normalization matrices, respectively. They state that they repeat $\mathbf{SR_t}$ and $\mathbf{SC_t}$ calculations for a limited number of times such as t=4 [1]. *r* represents the row vector of the given matrix.

3 Methodology

In our approach, $\mathbf{D_t}$ is the data matrix having *r* rows (documents) and *c* columns (words) formed from the training set. In this matrix d_{ij} shows the occurrence frequency of the j^{th} word in the i^{th} document; $d_i = [d_{i1} .. d_{ic}]$ is the row vector representing the document *i* and $d_j = [d_{1j} ..d_{rj}]$ is the column vector corresponding to word *j*.

We also tried several term weighting methods. First of them is TF-IDF (Term Frequency- Inverse Document Frequency) which is a statistical measure used to evaluate the importance of a word for a document in a corpus [17]. The formula for TF-IDF is given in Eq. 6.

$$TF - IDF(t,d) = p_{td} \times \log \frac{N}{n} \tag{6}$$

where p_{td} equals the number of times that *t* occurs in document *d*, *N* is the number of documents in the corpus and *n* is the number of documents that term *t* occurs. Another weighting approach we tried is from Dumais's research in [19]. In this approach, terms are represented in a document after multiplying by a value that is the global weight of the term in the whole corpus. The local weight of a term *t* in a document *d* is calculated as taking the log value of the total frequency of *t* in *d*. The global weight of a term is the entropy of that term in the corpus and according to [19] the entropy equals

$$Entropy(t,d) = 1 - \sum_{i=1}^{N} \frac{p_{td} \log(p_{td})}{\log(N)} \tag{7}$$

where *N* is the number of documents and p_{td} equals the number of times that *t* occurs in *d* divided by the total number of times that *t* occurs.

However, since we get better accuracies for linear kernel with the only TF (Term-Frequency) schema without any weighting, we use TF instead of TF-IDF or Entropy weighting approaches in our experiments for both linear kernel and our algorithm.

We use our term-frequency document corpus for χ-Sim's **SC** and **SR** similarity matrix calculations. We calculate up to four iterations. Similar to [1], we calculate $\mathbf{SR_0}$, $\mathbf{SC_0}$, $\mathbf{SR_1}$, $\mathbf{SC_1}$, $\mathbf{SR_2}$, $\mathbf{SC_2}$, $\mathbf{SR_3}$, $\mathbf{SC_3}$, $\mathbf{SR_4}$, $\mathbf{SC_4}$ matrices and after that we use these SC matrices, which contain iterative higher-order relations between terms, into our kernel by using Eq. 8:

$$k_{IHOSK}(d_1, d_2) = d_1 \cdot S.S^T \cdot d_2^T \tag{8}$$

where $K_{IHOSK}(d_1, d_2)$ is the similarity value between documents d_1 and d_2 , **S** is a semantic matrix which is gathered from the previously mentioned calculations of $\mathbf{SC_2}$ and d_1 and d_2 are term-frequency vectors of the documents. The **S** is a semantic matrix is based on *iterative higher-order paths* between documents and between terms. This kernel function means that the transformation of a document vector from input space to a feature space can be done by multiplying it with a semantic matrix as given in Eq.9:

$$\phi(d_1) = d_1 \cdot S \text{ and } \phi(d_2) = S^T \cdot d_2^T \tag{9}$$

In Eq.9. $\phi(d_1)$ and $\phi(d_2)$ are the transformations of document vectors d_1 and d_2 from their original input space into the feature space as required in the definition of kernel which is mentioned in Section 2.

After performing experiments up to four iterations of **SC** matrices, we conclude that the best results are obtained with the second iteration matrices ($\mathbf{SR_2}$, $\mathbf{SC_2}$). The following experimental results section reflects the results of our approach using these matrices.

Since we work with textual datasets which are high dimensional and highly sparse, we think that it is possible to benefit from normalization methods which could be applied on the similarity matrices. We experiment with several matrix normalization methods including row-level normalization (dividing each value in a row by the maximum value in that row), column-level normalization (dividing each value in a column by the maximum value in that column), document-length normalization (dividing each term frequency in a row with the corresponding documents length) and several other techniques which are used and explained in [9] (e.g., complement, weight normalization) and also some common methods from the literature which are explained in [18] such as z-score normalization, min-max normalization, etc. We obtained best accuracy results with length normalization which is defined in Eq.10.

$$\forall i, j \in 1...r \quad N\text{-IHOSK}(d_i, d_j) = \frac{\text{IHOSK}(d_i, d_j)}{|d_i| . |d_j|} \tag{10}$$

In Eq. 10, r is the number of documents in our corpus, IHOSK is similarity value between documents d_i and d_j, N-IHOSK is the normalized similarity value of these documents d_i and d_j and $|d_i|$ and $|d_j|$ are the lengths of these documents depending on the number of terms they have, respectively.

Then, we use this kernel function in SVM by plugging in the SMO WEKA's [21] implementation. In other words we built such a kernel function that is directly applicable in Platt's SMO (Sequential Minimal Optimization) [22] learner.

4 Experiment Setup

In order to examine the performance of N-IHOSK in SVM, we run it on several commonly used textual datasets. We use a variant of 20 Newsgroups dataset which is called 20News-18828[1]. This dataset has hierarchical class labels consist of four main groups namely SCIENCE, POLITICS, RELIGION and COMP and a total of 20 groups under them. We use the POLITICS and SCIENCE subsets of 20News-18828 dataset which consist of 3 classes and 4 classes, respectively. These subsets are also used in [3] and [4] for evaluating another higher-order classifiers HONB and HOSVM. We also make our experiments with COMP and RELIGION subsets of 20News-18828 dataset which are composed of 5 classes and 4 classes, respectively. Our third dataset is five-class version of the WebKB[1] dataset, namely WEBKB5, which includes web pages collected from computer science departments of different universities. It is important to note that while 20News-18828 subsets include the same number of documents in each class, WebKB5 dataset has a highly skewed class distribution. Fourth dataset we use is Mini-NewsGroups[2] dataset which has 20 classes. Properties of these datasets are given in Table 2.

We apply stemming and stopword filtering to the datasets. Terms occur less than three times in the documents are filtered. Furthermore, we used Information Gain in order to select most informative 2000 terms. This preprocessing increase the performance of the classifier models by reducing the noise.

Table 2. Properties of Datasets

Dataset	#classes	#instances	#features
WEBKB5	5	4,336	12,841
20NewsGroup			
20News-SCIENCE	4	2,000	2,225
20News-POLITICS	3	2,500	2,478
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.

One of the most important parameter of SMO [21] algorithm is misclassification-cost (C) parameter. We performed a series of exhaustive optimization trials on all of our datasets with the values in the set of $\{10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}\}$. For every training-set value of our all datasets we performed these optimization experiments and we selected the best performing value of that training-set of the corresponding dataset. After getting best performing C values for linear kernel which is our baseline algorithm at each training-set value we also use those C values for our proposed kernels of N-IHOSK, too. Optimized C values for each dataset are shown in Table 3.

After running algorithms on 10 random splits for each of the training set percentages with their corresponding optimized C values, we report average of these 10 results as in [4] and [8]. This is a more comprehensive way of well-known classical k-Fold cross validation which divides the data into k sets and train on k-1 of them while the remaining used as test set. However, the training set size in this approach is fixed (for instance it is %90 in 10-fold cross validation) and we cannot analyze the performance of the algorithm under scarce labeled data conditions. It is prohibitively expensive to obtain large amounts of labeled data in many real world applications and therefore it is important to develop methods that perform better with small training sets.

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

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

Table 3. Optimized C Values for Our Datasets

TS %	Optimized C Values for 20News SCIENCE	Optimized C Values for 20News POLITICS	Optimized C Values for WEBKB5	Optimized C Values for MINI-NEWSGROUP
5	1	10^{-1}	1	1
10	1	10^{-1}	1	1
30	1	10^{-1}	1	10^{2}
50	1	10^{-1}	1	10^{2}
70	1	1	1	10^{2}
80	1	1	1	10^{2}
90	1	10^{-1}	1	10^{1}

We run our experiments using our experiment framework called Turkuaz which closely uses WEKA [21] library. The main evaluation metric in our experiments is accuracy and in the results tables we also provide standard deviations. Additionally, Students t-Tests for statistical significance are provided. We use $\alpha = 0.05$ significance level which is a commonly used level. In order to highlight the performance differences between baseline algorithms and our approach we report performance gain calculated using the simple formula in Eq. 11;

$$Gain_{N-IHOSK} = \frac{(P_{N-IHOSK} - P_x)}{P_x} \tag{11}$$

where $P_{N\text{-}IHOSK}$ is the accuracy of SMO with N-IHOSK and P_x stands for the accuracy result of the other kernels (linear, polynomial or RBF). The experimental results are demonstrated in Table 4, Table 5, Table 6 and Table 7. These tables include training set percentage (TS), the accuracy results of linear kernel, polynomial kernel, RBF Kernel and N-IHOSK. Also the last columns show the (%) gain of N-IHOSK over linear kernel calculated as in Eq. 11.

5 Experiment Results

According to Table 4, N-IHOSK outperforms our baseline kernel (linear kernel, which is one of the state-of-the-art kernels in text classification [8], [9]) by extensive boundaries in all training set percentages. For instance at training levels 30%, 50% and 70% the accuracies of N-IHOSK are 94.31%, 94.97% and 95.35% while the accuracies of linear kernel are 86.73%, 88.94% and 90.37% ,respectively. The performance gain is obvious at all training set levels. It is important to note that high performance gains are especially visible at low training set levels. For instance at training levels 5%, and 10% N-IHOSK outperforms linear kernel with the gains of 18.64% and 16.25%, respectively. As mentioned above, this performance is of great importance since usually it is difficult and expensive to obtain labeled data in real world applications.

Table 4. Accuracy of Different Kernels on 20News SCIENCE Dataset with Varying Training Set Size

TS %	SMO-linear kernel	SMO-polynomial kernel	SMO-RBF Kernel	SMO-N-IHOSK	Gain
5	70.93±3.89	45.65±3.23	49.16±3.78	84.15±2.87	18.64
10	77.74±3.52	55.77±4.73	51.72±4.64	90.37±0.81	16.25
30	86.73±1.32	70.34±2.43	59.19±1.03	94.31±1.09	8.74
50	88.94±1.16	76.42±0.99	63.60±1.80	94.97±0.90	6.78
70	90.37±0.93	79.57±2.00	66.82±1.97	95.35±0.88	5.51
80	91.25±1.56	81.60±2.13	68.15±1.78	96.23±1.19	5.46
90	91.15±1.73	81.40±2.58	68.45±3.06	96.85±1.70	6.25

Table 5. Accuracy of Different Kernels on 20News POLITICS Dataset with Varying Training Set Size

TS %	SMO-linear kernel	SMO-polynomial kernel	SMO-RBF Kernel	SMO-N-IHOSK	Gain
5	78.33±3.40	56.69±6.79	55.74±6.43	82.27±4.60	5.03
10	84.66±2.09	62.45±6.67	65.33±3.96	88.61±2.1	4.67
30	91.98±1.24	83.30±4.57	80.34±4.05	93.61±1.08	1.77
50	91.21±0.89	89.43±2.03	87.95±2.18	93.55±3.58	2.57
70	92.29±1.22	91.02±1.50	87.84±1.79	93.24±3.08	1.03
80	93.7±0.79	90.77±1.50	88.5±1.12	95.3±1.82	1.71
90	93.69±2.04	92.2±1.81	89.8±2.18	95.8±2.28	2.25

On 20News POLITICS dataset, N-IHOSK gives better accuracies than linear kernel in all of the training levels which can be observable from Table 5.

Very similar situations are observed on the other subgroups of 20NewsGroup, namely RELIGION and COMP. In all training set levels N-IHOSK outputs higher accuracies compare to the baseline kernel. Since we got very similar and parallel outcomes we cannot provide their result-tables based on the reality of the space limitation here.

Same trend can be seen for WEBKB5 dataset which has a highly skewed class distribution. In this dataset again our algorithm N-IHOSK outperforms than all of the kernels including linear kernel, polynomial kernel and RBF Kernel. This can be seen in Table 6.

Table 6. Accuracy of Different Kernels on WEBKB5 Dataset with Varying Training Set Size

TS %	SMO-linear kernel	SMO-polynomial kernel	SMO-RBF Kernel	SMO-N-IHOSK	Gain
5	72.77±1.43	60.63±2.90	49.05±1.39	76.12±1.39	4.60
10	79.12±2.18	78.09±1.22	74.69±2.44	82.41±2.32	4.16
30	86.10±1.52	85.21±1.16	81.67±1.53	88.27±1.62	2.52
50	90.16±1.11	86.61±0.56	85.55±1.41	91.89±1.08	1.92
70	90.60±1.93	87.20±1.52	86.07±1.36	92.31±1.41	1.89
80	91.00±1.45	88.73±1.82	86.57±1.01	93.10±1.77	2.31
90	91.93±2.52	90.00±1.86	88.33±2.34	93.13±1.54	1.31

For us one of the most satisfactory results is observed in Mini-NewsGroups dataset. This dataset has the largest number of classes. Again in all training levels starting from 5% up until 90% N-IHOSK gives higher accuracies than other kernels. This can be seen from Table 7. This is especially obvious at 5% training level; the performance gain of N-IHOSK on linear kernel is 17.79%

Table 7. Accuracy of Different Kernels on Mini-NewsGroups Dataset with Varying Training Set Size

TS %	SMO-linear kernel	SMO-polynomial kernel	SMO-RBF Kernel	SMO-N-IHOSK	Gain
5	52.03±5.95	41.21±1.27	38.61±3.18	61.29±1.03	17.79
10	59.31±4.58	51.31±2.37	50.21±4.48	64.15±0.54	8.16
30	72.61±4.23	68.33±3.23	66.33±4.13	75.51±0.31	4.00
50	76.02±4.24	70.12±3.14	67.06±3.34	79.24±0.31	4.24
70	77.61±2.76	75.80±2.66	70.40±1.26	79.73±0.45	2.73
80	80.70±2.20	76.83±1.20	71.83±2.10	83.05±0.58	2.91
90	83.25±4.05	77.55±4.65	72.15±2.35	85.38±1.28	2.56

The particularly high accuracies of the proposed method on 20News-SCIENCE dataset may be explained with the less average sparsity of the documents of this dataset compare to the other datasets. It is possible that having

more terms in documents of this dataset give us the opportunity to generate more higher-order paths between documents.

At small training data levels first-order methods give zero as the similarity of two documents that do not contain common words. But by the use of higher-order paths the similarity between those two instances can be larger than zero. We think that this is the main reason that the difference between N-IHOSK and other first-order kernels (linear kernel. polynomial kernel and RBF Kernel) is most visible at small training levels like 5% and 10%. Through the experiments we observed remarkable gains such as 18.64%, 16.25%, and 17.79% at only using 5% and 10% of the labeled data as training set. This has important implications on real world applications where the labeled data is generally difficult to obtain. In many real world applications serious costs are associated with the labeling of the data.

6 Conclusion

It has been shown that higher-order co-occurrence relations between documents and terms catch "latent semantics" and result higher accuracies in text classification area [1], [3], [20] and [4]. Motivated by these studies, we propose a semantic kernel for the SVM named N-IHOSK. N-IHOSK exploits the semantic information in higher-order paths between documents as well as the higher-order paths between terms based on the methodology in [1]. We have performed detailed experiments on several popular textual datasets and compared N-IHOSK with traditional SVM kernels including state of the art linear kernel for text classification. Experiment results show that N-IHOSK outperforms the linear kernel, polynomial kernel, and RBF in all of our datasets under different training set size conditions. Our results show the usefulness of N-IHOSK as a semantic kernel for SVM in text classification.

As future work, we want to analyze the improved performance of N-IHOSK. Especially, we would like to shed light into if and how our approach implicitly captures semantic information such as synonyms and word sense disambiguation when calculating similarity between documents. Additionally, we plan to get more observations about under what type of conditions N-IHOSK performs better than other algorithms.

Acknowledgments

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. G. Bisson and F. Hussain, "Chi-Sim: A New Similarity Measure for the Co-clustering Task," Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications, 2008, pp. 211–217.
2. P. Wang and C. Domeniconi, "Building Semantic Kernels for text classification using Wikipedia.", in Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 713-721, ACM Press, New York, 2008.
3. M.C. Ganiz, N. Lytkin, W.M. Pottenger, "Leveraging Higher Order Dependencies between Features for Text Classification." in Proceedings of ECML/PKDD Conference, September, Bled, Slovenia , 2009.
4. M.C. Ganiz, C. George, W.M. Pottenger, "Higher Order Naive Bayes: A Novel Non-IID Approach to Text Classification", in IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 7, pp. 1022-1034, 2011.
5. M. Poyraz, Z.H. Kilimci, M.C. Ganiz, "Higher-Order Smoothing: A Novel Semantic Smoothing Method for Text Classification", Journal Of Computer Science and Technology, (Accepted), 2014.
6. M. Poyraz, Z.H. Kilimci, M.C. Ganiz, "A Novel Semantic Smoothing Method Based on Higher Order Paths for Text Classification", in IEEE International Conference on Data Mining (ICDM), Brussels, Belgium , 2012.
7. B. Altinel, M.C. Ganiz, B. Diri. "A Novel Higher-order Semantic Kernel". ICECCO 2013 (The 10th International Conference on Electronics Computer and Computation), November 7-9, Ankara, Turkey, 2013.
8. T. Joachims, "Text Categorization with Many Relevant Features", in Proceedings of European Conference on Machine Learning, Springer Verlag, 1998.
9. S. Dumais, J. Platt, D. Heckerman, & M. Sahami. " Inductive learning algorithms and representations for text categorization". In Proceedings of the Seventh International Conference on Information Retrieval and Knowledge Management (ACM-CIKM-98),pp. 148–155, 1998.
10. G. Siolas and F. D'Alche-Buc, "Support vectors machines based on a semantic kernel for text Categorization", in Proceedings of the International Joint Conference on Neural Networks, IEEE Press, Como , 2000.

11. E. Leopold and J. Kindermann, “Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?”, Machine Learning, vol. 46, pp. 423-444, 2002.
12. B.E. Boser, I.M. Guyon, and V.N. Vapnik, “A Training Algorithm for Optimal Margin Classifier”, in Proc. 5th ACM Workshop, Comput. Learning Theory, pp. 144–152, Pittsburgh, 1992.
13. C.W. Hsu, C.J. Lin, “A Comparison of Methods for Multi-Class Support Vector Machines”, IEEE Transactions on Neural Networks, pp. 415–425, 2002.
14. S. Bloehdorn, R. Basili, M. Cammisa and A. Moschitti, ”Semantic kernels for text classification based on topological measures of feature similarity.”, in ICDM ’06: Proceedings of The Sixth International Conference on Data Mining, pp. 808–812, 2006.
15. G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, “Five Papers on WordNet”, Technical report, Stanford University ,1993.
16. Q. Miller and E. Chen and H. Xiong,”A Semantic Term Weighting Scheme for Text Categorization”, in Journal of Expert Systems with Applications, 2011.
17. G. Salton and C. Buckley. "Term-weighting approaches in automatic text retrieval". Information Processing & Management, 24 (5). 1988.
18. J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, Morgan Kaufmann, Third Edition, 2012.
19. S. Dumais. “LSI meets TREC: A status report”. In Hartman, D., ed., The first Text Retrieval Conference: NIST special publication 500-215, pp. 105–116, 1993.
20. A. Kontostathis and W.M. Pottenger,”A Framework for Understanding LSI Performance”, in Information Processing & Management, pp. 56-73, 2006.
21. H. I. Witten and E. Frank. “Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations”. Morgan Kaufmann, 1999.
22. J. C. Platt, “Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines”, in Advances in Kernel Method: Support Vector Learning, MIT Press, pp. 185–208, 1998.