A Simple Semantic Kernel Approach for SVM using Higher-Order Paths

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

Murat Can Ganiz[2]
Computer Engineering Department of
Doğuş University
Istanbul, Turkey
mcganiz@dogus.edu.tr

Banu Diri[3]
Computer Engineering Department of
Yildiz Technical University
Istanbul, Turkey
banu@ce.yildiz.edu.tr

***Abstract*— The bag of words (BOW) representation of documents is very common in text classification systems. However, the BOW approach ignores the position of the words in the document and more importantly, the semantic relations between the words. In this study, we present a simple semantic kernel for Support Vector Machines (SVM) algorithm. This kernel uses higher-order relations between terms in order to incorporate semantic information into the SVM. This is an easy to implement algorithm which forms a basis for future improvements. We perform a serious of experiments on different well known textual datasets. Experiment results show that classification performance improves over the traditional kernels used in SVM such as linear kernel which is commonly used in text classification.**

Keywords*— higher-order relations; machine learning; support vector machine; text classification; semantic kernel.

I. Introduction

Text classification can be defined as automatically classifying documents according to predefined category-labels, usually by using machine learning algorithms. There are large amounts of textual data accumulated both in organizations and especially on the World Wide Web (WWW) through social networks, blogs, news, forums…etc. This huge set of documents continues to increase by the contribution of millions of people every day. Automatically processing these increasing amounts of textual data is one of the critical problems for research and commercial entities. Text classification is the basis for several important applications such as document filtering and sentiment or opinion classification.

One of the most common approaches for representing document is the bag of words (BOW) feature representation. In this approach, the documents are represented only by occurrences or frequencies of the words, independent from their position in the document or the semantic or syntactic connections between other words. To be more precise; it turns a blind eye to the multi-word expressions by breaking them apart. Furthermore, it handles treats polysemous words (i.e., words with multiple meanings) as a single entity. Additionally, it maps synonymous words into different components; as it is mentioned in [1]. Thus, in order to reduce the effect of these disadvantages and improve the prediction abilities of text classification algorithms, it is important to make use of semantic relations between words.

In this work, we introduce a simple approach for building a semantic kernel for Support Vector Machines (SVM) called *Higher-Order Term Kernel* (HOTK) which is based on higher-order paths between terms (or words) through documents. In our experiments, we compare HOTK with other traditional kernel methods for SVM such as the linear kernel. Please note that SVM with linear kernel is one the state of the art algorithms for text classification [8], [22]. These traditional kernels can be considered as first-order methods since their context or scope consist of a single document, only. However, HOTK can make use of higher-order paths that include several different words and documents in the context of the whole dataset. Experimental results show that HOTK exceeds the performance of the other first-order kernels on several benchmark datasets.

Our approach is motivated by the studies of higher-order Naïve Bayes (HONB) [3] and Higher-Order Smoothing (HOS) [4], [5] and recently introduced works of Higher-Order Semantic Kernel (HOSK) [6] and Iterative Higher-Order Semantic Kernel (IHOSK) [7] which mainly focus on the higher-order paths between documents.

Both HONB and HOS are Naïve Bayes based methods. They are designed to work on binary term frequency data and they make use of the higher-order paths between terms. On the other hand, HOSK and IHOSK are semantic kernel methods for SVM. HOSK is our first attempt to use higher-order paths as a semantic kernel in SVM and it is based on higher-order paths between the documents. This approach is further explained in related work section. The following work, IHOSK is similar to the HOSK since they both propose a semantic kernel for SVM by using higher-order relations. However, IHOSK makes use of the higher-order paths between both the documents and the terms iteratively. Although, the performance of IHOSK is superior, its complexity is much higher than the previous works such as HOSK, and the proposed work of HOTK.

HOTK is our first attempt to use higher-order paths between terms as a semantic kernel for SVM. In this sense it is similar to the previous term based higher-order leaning algorithms HONB and HOS. HOTK is much simpler than the IHOSK. Using higher-order paths between terms instead of between documents (as in HOSK) or both the documents and terms (as in IHOSK) forms a foundation that is open to several improvements. For instance HOTK can easily be combined with other term based semantic kernels such as the ones using WordNet or Wikipedia. Furthermore, it will be much easier to apply different path filters and normalizations based on the role of terms in different classes and observe their affects.

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

II. Related Work

A. Support Vector Machines for Classification Problem

SVM was first studied by Vapnik, Guyon and Boser [10]. A more detailed analysis is given in [11]. In general, SVM is a linear classifier that aims to finds the optimal separating hyperplane between two classes. It is possible and common to use a kernel function in SVM which can map or transform the data into a higher dimensional feature space if it is impossible or difficult to find a separating hyperplane between classes in the original space [8]. We can consider a kernel function as a kind of similarity function, which calculates the similarity values of data points in the transformed space. Therefore, defining an appropriate kernel has the direct effect on finding a better representation of these data points as it is mentioned in [1], [12] and [13] . SVM algorithm can be used for multi-class categorization by the "one-against-the-rest" and "one-against-one" approaches [20]. SVM can work very well on high dimensional and sparse data [8]. Because of these benefits SVM with linear kernel is one of the state of the art algorithms in text classification domain since textual data represented using BOW approach as a document by term matrix is high dimensional and quite sparse.

B. Semantic Kernels for Text Classification

According to the definition mentioned in [10], [14], and [1] and [15], 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 [19], the authors propose a semantic kernel which is based on WordNet [21]. WordNet is a network of semantic relations between words. These relations and hierarchies can be used to measure similarities between words. The authors use WordNet's hierarchical tree structure to measure semantic similarity between two words. They used this information to enrich the Gaussian kernel. Their results show that using the above mentioned semantic proximity metric increases the classification accuracy in SVM. However, their approach treats multi-word concepts as single terms and does nothing to deal the problem of polysemy.

Semantic kernels with super concept declaration were studied in [15]. 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 $\boldsymbol{Q}$ that is shown to be composed of $\boldsymbol{P}$ and $\boldsymbol{P}^T$ which includes super-concept information about their corpus. The proposed kernel function is given in Eq. 2. Their results show that they get significant improvements in the performance, especially in the cases where little training data exists or the feature representations are highly sparse [15]. However, they coming short of a word sense disambiguation strategy [15].

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

Similarly, in [12] and [13] the WordNet is used as a semantic information resource. However, they stated in their work that the coverage of WordNet is not sufficient and this is one of the main reasons that several following studies concentrated on some other wider coverage such as Wikipedia[1].

In one of these works [1], the authors 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 like in the previous equation Eq.2. this time where $\boldsymbol{P}$ 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 [1]. 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 [1].The authors claim that their method overcomes some of the shortages of the BOW approach. Their results demonstrate that adding semantic knowledge that is extracted from Wikipedia into document representation improves the categorization accuracy.

C. Higher-order Paths

Our approach is motivated by the studies of higher-order Naïve Bayes (HONB) [2],[3] and Higher-Order Smoothing (HOS) [4], [5] which makes use of the higher-order paths between terms, and recently introduced works of [6] and [7] which focus on the higher-order paths between documents instead. HONB and HOS are both Naïve Bayes based methods.

Advantages of using higher-order paths between documents as in [5] are illustrated in Fig. 1. In this figure, there are three documents, d_1, d_2 and d_3 which include a set of terms $\{t_1, t_2, t_3\}$, $\{t_3, t_4, t_5\}$ and $\{t_4, t_5, t_7\}$ respectively. Using a traditional similarity measure which is based on the shared terms (e.g. dot product), similarity value between documents d_1 and d_3 will be zero since they do share any terms. But in fact these two documents have some similarities in the context of the dataset through d_2 as it can be seen in Fig. 1. This supports the idea that using higher-order paths between documents, it is possible to obtain a non-zero similarity value between d_1 and d_3 which was not possible in BOW representation. This value becomes larger if there are many interconnecting documents like d_2 between d_1 and d_3. This may stem from the reason that the two documents are written on the same topic using two different but semantically closer sets of terms.

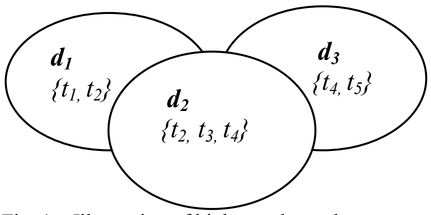

Fig. 1. Illustration of higher-order paths

Advantages of using higher-order paths between terms as in [3], [5] can also be seen in Fig. 1. In this figure, there is a higher-order path between t_1 and t_3. This is an example of a novel second-order relation since these two terms do not co-occur in any of these documents and can be gone unnoticed in traditional BOW models. However, we know that t_1 co-occurs with t_2 in document d_1, and t_2 co-occurs with t_3 in document d_2. The same principle that is mentioned above applies in here. The similarity between t_1 and t_3 becomes more eminent if there are many interconnecting terms such as t_2 or t_4 and interconnecting documents like d_2. The regularity of these second order paths may reveal latent semantic relationships such as synonymy [5].

III. Approach

In our proposed method, $\boldsymbol{D_{train}}$ is the data matrix having r rows (documents) and t columns (words) formed using the training set. In this matrix d_{ij} represents the occurrence frequency of the j^{th} word in the i^{th} document; $d_i = [d_{i1},...,d_{it}]$ is the row vector showing the document i and $d_j = [d_{1j},...,d_{rj}]$ is the column vector belongs to word j.

It is important to note that binary term occurrences are used in the premier studies which use higher-order paths between terms since it simplifies the definition and counting of the higher-order paths. However, in this study, we experiment with term frequencies (tf). This is similar to the initial attempt to use term frequencies in [5].

We use the training set to extract higher-order paths between terms. The $\boldsymbol{S}$ matrix which shows the amount or weight of higher-order (second-order in this case) relations between terms is obtained by using the formula in Eq. 3. This approach is motivated by algorithm which is explained in [5]. However, in this study, we are using term frequencies instead of binary term occurrences and we are not filtering any paths.

$$S = D_{train}^T \cdot D_{train} \tag{3}$$

In Eq.3, $\boldsymbol{D_{train}}$ is document by term matrix of the training set $\boldsymbol{S}$ is a symmetric square matrix whose dimensions are the number of the terms in the training set. The $\boldsymbol{S}$ matrix displays the first-order relations, in other words just co-occurrences of the terms. In order to get higher co-occurrence relations or in other words higher-order paths we multiply the $\boldsymbol{S}$ by itself. For instance, the square of $\boldsymbol{S}$ reveals the second-order relations between the terms. Since the second order-paths reveal latent semantic relations [2] we use the following Eq.4 as a simple semantic kernel.

$$k_{HOT-K}(d_1, d_2) = d_1 \cdot S.S^T \cdot d_2^T \tag{4}$$

The proposed kernel function in Eq.4 means that the transformation of a document vector from input space to a semantic feature space can be accomplished by multiplying it with a semantic matrix as shown in Eq. 5.

$$\phi(d_1) = d_1 \cdot S \text{ and } \phi(d_2) = S^T \cdot d_2^T \tag{5}$$

In Eq.5, $\phi(d_1)$ and $\phi(d_2)$ vectors are the transformations of documents d_1 and d_2 vectors from their original input space into the feature space as required in the definition of kernel which is mentioned in Section 2.

We also experimented with different 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 approaches those explained in [16] (e.g., complement, weight normalization) and [17] such as z-score normalization, min-max normalization, etc. We obtained best accuracy results with length normalization which is defined in Eq.6.

$$\forall i, j \in 1...r \quad k_{norm}(d_i, d_j) = \frac{k(d_i, d_j)}{|d_i| . |d_j|} \tag{6}$$

In Eq.6 $|d_i|$ and $|d_j|$ are the lengths of these documents measured by the sum of the term occurrences.

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

IV. Experiment Setup

We integrated our kernel function into the SVM implementation of SVM algorithm in WEKA [18]. In other words we created such a kernel function that is possible to directly use and plugged it in Platt's Sequential Minimal Optimization (SMO) [19] learner.

In order to examine the performance of HOTK in SVM, we run it on several commonly used textual datasets. We used a variant of 20 Newsgroups dataset which is called 20News-18828[1]. We used this dataset as in three basic subgroup forms namely "POLITICS","COMP" and "SCIENCE". Fourth dataset we use is mini-newsgroups [2] dataset which has 20 classes. Properties of these datasets are given in Table 1.

We apply stemming and stopword filtering to these datasets as it is a common approach. Additionally, we filter the terms which occur less than in three documents. We also apply attribute selection and select the most informative 2000 terms using Information Gain.

Table I. Properties of Datasets

Dataset	#classes	#instances
20News-SCIENCE	4	2,000
20News-POLITICS	3	1,500
20News-COMP	5	2,500
mini-newsgroups	20	2,000

In order to observe the performance of HOTK under different training set size conditions, we use the following percentage values for training set size of 5%, 10%, 30%, 50%, 70%, 80%, 90%. Remaining documents are used for testing. This important because we expect that the benefit of using semantic kernels should be more visible when there is insufficient labeled data.

One of the most important parameter of SMO [18] 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 all the training set percentages of the all datasets we optimized the *c* value by trying all the values above and selecting the best performing one. The optimized *c* values for each dataset at different training levels are shown in Table II.

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 [2] and [4]. This is a more comprehensive way of well-known *n*-fold cross validation which divides the data into *n* sets and train on *n-1* of them while the remaining used as test set. Since the training set size in this approach is fixed (for instance it is 90% for 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/

We run our experiments using our experiment framework called Turkuaz which closely uses WEKA [18] library. The main evaluation metric in our experiments is accuracy and in the results tables we also provide standard deviations.

In Table II, we provide the optimized *c* values for our experiments. This is interesting because the values change quite a bit from dataset to dataset.

Table II. Optimized C Values for Our Datasets

TS %	*20News SCIENCE*	*20News POLITICS*	*20News COMP*	*Mini-newsgroups*
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}

In order to highlight the performance differences between baseline algorithms and our approach we report performance gain calculated using the simple formula in Eq. 7;

$$Gain_{HOT-K} = \frac{(P_{HOT-K} - P_x)}{P_x} \tag{7}$$

where P_{HOTK} is the accuracy of SMO with normalized *HOTK* and P_x stands for the accuracy result of the other kernel (linear kernel). The experimental results are demonstrated in Table III, Table IV, As expected, the performance improvement is most visible in small training set levels which can be seen from Table III. For mini-newsgroups dataset, *HOTK* outperforms linear kernel in almost all of the training levels. This can be seen from Table VI.

Table V and Table VI. These tables include training set percentage (TS), the accuracy results of linear kernel, Polynomial Kernel, and HOTK. Also the last columns demonstrate the (%) gain of HOTK over linear Kernel calculated as in Eq.14.

V. Experiment Results and Discussion

According to our experiments HOTK demonstrates a notable performance on 20NewsSCIENCE dataset, which can be seen in Table III. HOTK outperforms our baseline kernel (linear kernel, which is one of the state-of-the-art kernels in text classification) in all training set percentages. The performance gain is specifically obvious at low training set levels. For instance, at training levels 5%, and 10% HOTK outperforms linear kernel with the gains of 7.26% and 5.77% on linear kernel respectively. 20NewsSCIENCE dataset is also used in our previous studies [5], [7]. Therefore we use this dataset to compare the results of HOSK [6] and IHOSK [7] with HOTK. Although the HOTK be able to outperform the baseline (linear kernel), the performance of IHOSK is superior to the HOSK and

HOTK. However, the complexity of IHOSK is much higher than the previous works such as HOSK, and the proposed work of the HOTK. This prevents the IHOSK to be applied on large datasets. HOSK also performs slightly better than HOTK but it is based on the higher-order paths between documents. The semantic relations between the documents are not as clear as the relations between the terms. HOTK is our first attempt to use the higher-order paths between terms as a semantic kernel for SVM. Using higher-order paths between terms instead of between documents (as in HOSK) or both the documents and terms (as in IHOSK) forms a foundation that is open to several improvements. For instance HOTK can easily be combined with other term based semantic kernels such as the ones using WordNet or Wikipedia. Furthermore, it will be much easier to apply different path filters and normalizations based on the role of terms in different classes and observe their affects.

TABLE III. ACCURACY OF DIFFERENT HO KERNELS ON 20NEWSSCIENCE DATASET WITH VARYING TRAINING SET SIZE

TS%	*Linear*	*HOSK*	*IHOSK*	*HOTK*	*Gain*
5	71.44±4.30	85.69±1.80	**90.37±0.81**	76.63±2.67	7.26
10	77.97±3.73	87.87±1.34	**94.31±1.09**	82.47±2.02	5.77
30	86.73±1.32	93.11±0.77	**94.97±0.90**	89.24±0.74	2.89
50	88.94±1.16	94.18±0.48	**95.35±0.88**	90.84±1.12	2.14
70	90.58±0.93	95.07±0.86	**96.23±1.19**	92.06±1.28	1.63
80	91.33±1.41	95.40±0.87	**96.85±1.70**	93.38±1.43	2.24
90	91.40±1.56	**96.00±1.80**	94.31±1.09	94.2±1.36	3.06

For the remaining datasets we report the results of HOTK compared to the baseline kernels of linear and the polynomial. 20NewsPOLITICS is an exceptional dataset in terms of the performance of HOTK. We only see improvements at very low training set percentages. This may due to the size of the dataset. 20NewsPOLITICS is our smallest dataset with 3 classes and 1500 documents. We observe that the discussions are centered around a smaller number of topics compare to the other datasets. In our opinion in this dataset, the classes are easier to discriminate, giving more advantage to the document based methods.

TABLE IV. ACCURACY OF DIFFERENT KERNELS ON 20NEWSPOLITICS DATASET WITH VARYING TRAINING SET SIZE

TS%	*Linear*	*Polynomial*	*HOTK*	*Gain*
5	79.01±2.65	56.69±6.79	**80.72±1.56**	2.16
10	84.69±1.24	62.45±6.67	**84.89±2.15**	0.24
30	**92.04±1.06**	83.30±4.57	88.31±1.22	-4.05
50	**93.73±0.57**	89.43±2.03	90.29±0.79	-3.67
70	**94.55±1.21**	91.02±1.50	90.15±1.15	-4.65
80	**94.03±0.91**	90.77±1.50	92.50±1.60	-1.63
90	**94.86±1.26**	92.20±1.81	92.46±2.01	-2.53

For 20NewsCOMP dataset, *HOTK* outperforms linear kernel in all training levels. This can be seen from As expected, the performance improvement is most visible in small training set levels which can be seen from Table III. For mini-newsgroups dataset, HOTK outperforms linear kernel in almost all of the training levels. This can be seen from Table VI.

Table V. 20NewsCOMP is a larger dataset than the 20NewsPOLITICS It has five classes. As expected, the performance improvement is most visible in small training set levels which can be seen from Table III. For mini-newsgroups dataset, HOTK outperforms linear kernel in almost all of the training levels. This can be seen from Table VI.

TABLE V. ACCURACY OF DIFFERENT KERNELS ON 20NEWSCOMP DATASET WITH VARYING TRAINING SET SIZE

TS%	*Linear*	*Polynomial*	*HOTK*	*Gain*
5	56.75±4.72	37.23±3.57	**60.22±3.00**	6.11
10	65.45±2.77	44.36±3.07	**66.70±1.14**	1.91
30	75.38±2.12	60.90±3.00	**75.97±1.04**	0.78
50	77.89±1.60	64.6±2.18	**78.68±0.71**	1.01
70	79.63±1.59	66.87±2.25	**79.97±1.18**	0.43
80	79.00±2.25	65.70±3.97	**80.38±1.85**	1.75
90	81.40±2.47	67.48±2.29	**81.52±1.46**	0.15

TABLE VI. ACCURACY OF DIFFERENT KERNELS ON MINI-NEWSGROUP DATASET WITH VARYING TRAINING SET SIZE

TS%	*Linear*	*Polynomial*	*HOTK*	*Gain*
5	**56.75±4.72**	41.21±1.27	49.69±5.64	-12,44
10	65.45±2.77	51.31±2.37	**66.24±3.81**	1,21
30	75.38±2.12	68.33±3.23	**81.82±2.04**	8,54
50	77.89±1.60	70.12±3.14	**85.54±1.20**	9,82
70	79.63±1.59	75.80±2.66	**87.28±1.13**	9,61
80	79.00±2.25	76.83±1.20	**88.15±1.58**	11,58
90	84.65±2.48	77.55±4.65	**88.10±2.80**	4,08

VI. CONCLUSIONS AND FUTURE WORK

In this work, we introduce a simple approach for building a semantic kernel for Support Vector Machines (SVM) called *Higher-Order Term Kernel* (HOTK) which is based on higher-order paths between words (or terms) through documents. Our approach is motivated by the studies of Higher-order Naïve Bayes [2], [3] and Higher-Order Smoothing [4] and [5] which makes use of the higher-order paths between terms, and recently introduced works of [5] and [7] which focus on the higher-order paths between documents instead. In this study, we demonstrate a preliminary and as a result a simple approach for creating a semantic kernel based on the second-order term co-occurrences in the training set. Our results show the promise of HOTK as a semantic kernel for SVM in text classification which can be further improved by applying some of the techniques in [5].

HOTK is our first attempt to use the higher-order paths between terms as a semantic kernel for SVM. Using higher-order paths between terms instead of between documents (as in HOSK [6]) or both the documents and terms (as in IHOSK [7]) forms a foundation that is open to several improvements. For instance HOTK can easily be combined with other term based semantic kernels such as the ones using WordNet or Wikipedia. Furthermore, it will be much easier to apply different path filters and normalizations based on the role of terms in different classes and observe their affects.

As future work, we want to study on different weighting and normalization approaches on our algorithm. We also would like to analyze and shed light on how our approach implicitly captures semantic information such as synonyms, and performs word sense disambiguation for polysemous terms when calculating similarity between documents. Additionally we plan to get more insights about under which conditions HOTK performs well.

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] 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.

[2] M. 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

[3] Ganiz, M., Lytkin, N. , Pottenger, W.M. (2009). Leveraging Higher Order Dependencies between Features for Text Classification. In Proceedings of ECML/PKDD Conference, September, 2009, Bled, Slovenia.

[4] 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.

[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,Vol.29, No.3, 2014, pp.376-391

[6] Altınel, B., Ganiz, M.C., Diri, B., (2013). A Novel Higher-order Semantic Kernel. ICECCO 2013 (The 10th International Conference on Electronics Computer and Computation) , November 7-9, Ankara, Turkey.

[7] Altınel, B., Ganiz, M.C., Diri, B., (2014).(Accepted) A Semantic Kernel for Text Classification Based on Iterative-Higher–Order Relations between Words and Documents. ICAISC 2014 (The 13th International Conference on Artificial Intelligence and Soft Computing) , June 1-5, Zakopane, Poland.

[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, (1998). 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).

[10] 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.

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

[12] 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.

[13] 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.

[14] Q. Luo and E. Chen and H. Xiong,”A Semantic Term Weighting Scheme for Text Categorization”, in Journal of Expert Systems with Applications, 2011.

[15] 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.

[16] J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger. Tackling the poor assumptions of Naive Bayes classifiers. In Proceedings of International Conference on Machine Learning, pages 616–623, 2003.

[17] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, Morgan Kaufmann, Third Edition,2012

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

[19] 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

[20] 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.

[21] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, “Five Papers on Wordnet”, Technical report, Stanford University ,1993.

[22] S. Dumais, J. Platt, D. Heckerman, & M. Sahami, (1998). 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).