A Kernel-based Feature Weighting for Text Classification

Peter Wittek and Chew Lim Tan

***Abstract*— Text classification by support vector machines can benefit from semantic smoothing kernels that regard semantic relations among index terms while computing similarity. Adding expansion terms to the vector representation can also improve effectiveness. However, existing semantic smoothing kernels do not employ term expansion. This paper proposes a new non-linear kernel for text classification to exploit semantic relations between terms to add weighted expansion terms.**

I. Introduction

Building an automated text classification system consists of two key subtasks. The first task is text representation which converts the content of documents into compact format so that they can be further processed by the text classifiers. Another task is to learn the model of a text classifier which is used to classify the unlabeled documents. This paper proposes a substantially new model for text representation to improve effectiveness of text classification.

Since the early days of the vector space model, it has been debated whether it is a proper carrier of meaning of texts [1], arguing if distributional similarity is an adequate proxy for lexical semantic relatedness [2]. With the statistical, i.e. devoid of word semantics approaches there is generally no way to improve both precision and recall at the same time, increasing one is done at the expense of the other. For example, casting a wider net of search terms to improve recall of relevant items will also bring in an even greater proportion of irrelevant items, lowering precision. In the meantime, practical approaches have been proliferating, especially with developments in kernel methods in the last decade [3], [4], [5]. Some researchers suggested a more general mathematical framework to accommodate the needs that the vector space model cannot satisfy [6]. This paper explores the opportunities of this representation in the domain of text classification by introducing it as a new nonlinear semantic kernel.

By automatically selecting expansion terms for a text classification system to expand a document vector by adding terms that are related to the terms already in the document, performance can be improved [7]. The new terms can either be statistically related to the original terms or chosen from lexical resources such as thesauri, controlled vocabularies, ontologies and the like.

Using statistical relations to expand document vectors is attractive since the relations are easily generated from the documents in a collection. Unfortunately, such methods have had little success in improving overall effectiveness [8].

Using lexical resources as a source of related terms has met with success in some experiments. Expansion by synonyms improved performance but expansion by broader or narrower terms selected from a hierarchical thesaurus was too inconsistent to be generally useful [9].

A fundamental question often overlooked is whether the expansion terms extracted are equally related to the document and are useful for text classification. This paper proposes a form of term expansion with decreasing importance of those terms that are less related, as contrasted with rigid term expansion.

This paper is organized as follows. Section II overviews text classification by support vector machines, expanding on traditional text similarity measures II-A, semantic smoothing kernels II-B, term expansion strategies II-C, and finally introduces our semantic kernels in the L_2 space II-D. Section III discusses experimental results and Section IV concludes the paper.

II. Text Classification with Support Vector Machines

Text categorization is the task of assigning unlabeled documents into predefined categories. Given a collection of $\{d_1, d_2, \ldots, d_N\}$ documents, and a $C = \{c_1, c_2, ..., c_{|C|}\}$ set of predefined categories, the task is, for each document d_j ($j \in \{1, 2, \ldots, N\}$), to assign a decision to file d_j under c_i or a decision not to file d_j under c_i ($c_i \in C$) by virtue of a function Φ, where the function Φ is also referred to as the classifier, or model, or hypothesis, or rule. Supervised text classification is a machine learning technique for creating the function Φ from training data. The training data consist of pairs of input documents, and desired outputs (i.e., classes).

Support vector machines have been found the most effective by several authors [3], [10]. The proposed semantic text classification method is grounded in the kernel methods underlying support vector machines.

A support vector machine is a kind of supervised learning algorithm. In its simplest, linear form, a support vector machine is a hyperplane that separates a set of positive examples from a set of negative examples with maximum margin [5]. The strength of kernel methods is that they allow a mapping $\phi(.)$ of $\mathbf{x}$ to a higher dimensional space. In the dual formulation of the mathematical programming problem, only the kernel matrix $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)'\phi(\mathbf{x}_i)$ is needed in the calculations.

A. Traditional Text Similarity Measure

Intuitively, if a text fragment of two documents address similar topics, it is highly possible that they share lots of substantive terms. After having removed the stopwords and stemmed the rest e.g. by the Porter stemmer [11], the stemmed terms construct a vector representation for each text

Peter Wittek and Chew Lim Tan are with the Department of Computer Science, National University of Singapore (email: {wittek, tancl}@comp.nus.edu.sg).

document. Let $\mathbf{a}_j$ be a document vector in the vector space model, that is,

$$\mathbf{a}_j = \sum_{k=1}^{M} a_{kj}\mathbf{e}_k, \tag{1}$$

where M is the number of index terms, a_{kj} is some weighting (e.g., term frequency), and $\mathbf{e}_k$ is a basis vector of the M-dimensional Euclidean space. This representation is also referred to as the bag-of-words (BOW) model.

Given this representation, semantic relatedness of a pair of text fragments is computed as the cosine similarity of their corresponding term vectors which is defined as:

$$S(\mathbf{a}_i, \mathbf{a}_j) = \frac{\mathbf{a}_i\mathbf{a}_j}{|\mathbf{a}_i||\mathbf{a}|_j}. \tag{2}$$

B. Linear Semantic Kernels

One enrichment strategy is to use a semantic smoothing kernel while calculating the similarity between two documents. Any linear kernel for texts is characterized by

$$K(\mathbf{a}_i, \mathbf{a}_j) = \mathbf{a}_i' S' S \mathbf{a}_j, \tag{3}$$

where S is an appropriately shaped matrix commonly referred to as semantic smoothing matrix [12], [5], [13], [14]. The presence of S changes the orthogonality of the vector space model, as this mapping should introduce term dependence. A recent attempt tried to manually construct S with the help of a lexical resource [12]. The entries in the symmetric matrix S express the semantic proximity between the terms i and j. Entries in this matrix are inversely proportional to the length of the WordNet hierarchy path linking the two terms. The performance, measured over the 20NewsGroups corpus, showed an improvement of 2 % over the the basic vector space method. Moreover, the semantic matrix S is almost fully dense, hence computational issues arise. In a similar construction, [14] defined the matrix entries as weights of superconcepts of the two terms in the WordNet hierarchy. Focusing on special subcategories of Reuters-21578 and on the TREC Question Answering Dataset, they showed consistent improvement over the baseline. As [13] pointed out, polysemy will remain a problem in semantic smoothing kernels.

An early attempt to overcome the untenable orthogonality assumption of the vector space model was proposed under the name of generalized vector space model [15]. The article which proposed the model did not provide empirical results [15], and since then the model has been regarded of large theoretical importance with less impact on actual applications. The model takes a distributional approach, focusing on term co-occurrences. The underlying assumption is that term correlations are captured by the co-occurrence information. That is, two terms are semantically related if they co-occur often in the same documents [15]. By eliminating orthogonality, documents can be seen as similar even if they do not share any terms. The term co-occurrence matrix is AA', hence the model takes A' as the semantic proximity matrix S. A major drawback of the generalized vector space model is that it replaces the orthogonality assumption with another questionable assumption. The computational needs are tremendous too, if the dimensions of A are considered. Moreover, the co-occurrence matrix is not sparse anymore.

Latent semantic indexing (or latent semantic analysis) was another attempt to bring more linguistic and psychological aspects to language processing via a kernel. Conceptually, latent semantic indexing is similar to the generalized vector space model, it measures semantic information through co-occurrence analysis in the corpus. From the algorithmic perspective it is an enormous problem that textual data have a large number of relevant features. This results in huge computational needs and the classification models may overfit the data. The number of features can be reduced by multivariate feature extraction methods. In latent semantic indexing, the dimension of the vector space is reduced by singular value decomposition [16].

Using rank reduction, terms that occur together very often in the same documents are merged into a single dimension of the feature space. The dimensions of the reduced space correspond to the axes of greatest variance. For latent semantic indexing, by dual representation the kernel matrix is $K = V\Sigma_k^2 V'$, where Σ_k is a diagonal matrix containing the k largest singular values of the singular value decomposition of the vector space, and V holds the right singular values of the decomposition. The new kernel matrix can be obtained directly from K by applying an eigenvalue decomposition of K [4]. The computational complexity of performing an eigenvalue decomposition on the kernel matrix is a major drawback of latent semantic indexing.

C. Text Representation Enrichment Strategies by Term Expansion

In order to eliminate the bottleneck of the traditional BOW representation, previous approaches in term expansion enriched this convention by external lexical resources such as WordNet.

As a first step, these methods generate new features for each document in the dataset. These new features can be synonyms or homonyms of document terms as in [17], [18], or expanded features for terms, sentences and documents as in [19].

Then, the generated new features replace the old ones or are appended to the document representation, and construct a new vector representation $\hat{\mathbf{a}}_i$ for each text document. The similarity measure of document pairs is defined as:

$$S(\hat{\mathbf{a}}_i, \hat{\mathbf{a}_j}) = \frac{\hat{\mathbf{a}}_i\hat{\mathbf{a}_j}}{|\hat{\mathbf{a}}_i||\hat{\mathbf{a}_j}|}. \tag{4}$$

D. Our Framework

The basic assumption of our framework is that terms can be arranged in an order such that consecutive terms are semantically related. Hence each term acquires a unique position, and this position ties the term to its semantically related neighbors. However, given a BOW representation with a cosine similarity measure, this position would not improve classification performance. Therefore we suggest to associate a mathematical function with each term, thus mapping terms

and documents to the L_2 space, and using the inner product of this space to express similarity. The choice of function will determine to which extent neighboring terms, i.e., the enriching terms, are considered in calculating the similarity between two documents. This section first introduces an algorithm that produces the aforementioned semantic order, then the semantic kernels in the L_2 space are discussed.

1) An Algorithm for a Semantic Ordering of Terms: The proposed kernels assume that there is a semantic order between terms. To produce such an order, two problems have to be addressed:

1) Measuring semantic relatedness between terms. Various lexical resource-based [2] and distributional measures [20] have been proposed to measure semantic relatedness and distance between terms.
   Distributional measures use statistics acquired from large text corpora to determine how similar the contexts of two terms are. These measures are also used as proxies to measures of semantic similarity as terms found in similar contexts tend to be semantically similar.
   Lexical resource-based measures typically rely on WordNet or other hierarchical network [2]. These measures employ some form of edge counting, that is, counting the number of edges in the hierarchy between two nodes (senses). This approach is similar to the ones used by certain semantic smoothing kernels (Section II-B). Other factors, such as link density [21], or weights of edges [22] can also be used in calculating the measure.
   Terms can be corpus- or genre-specific. Manually constructed general-purpose lexical resources include many usages that are infrequent in a particular corpus or genre of documents. For example, one of the 8 senses of *company* in WordNet is a *visitor/visitant*, which is a hyponym of *person* [23]. This sense of the term is practically never used in newspaper articles, hence distributional attributes should be taken into consideration. Composite measures that combine the advantages of both approaches have also been developed [24], [25]. This paper relies on the Jiang-Conrath composite measure [25], which has been shown to be superior to other measures [2], and we also found that this measure works the best for the purpose. The Jiang-Conrath metric measures the distance between two senses by using the hierarchy of WordNet. By denoting the lowest super-ordinate of two senses s_1 and s_2 in the hierarchy with LSuper(s_1,s_2), the metric is calculated as follows:

$$d(s_1, s_2) = \mathrm{IC}(s_1) + \mathrm{IC}(s_2) - 2\mathrm{IC}(\mathrm{LSuper}(s_1, s_2)),$$

   where IC(s) is the information content of a sense s based on a corpus. Distance between two terms is calculated according to the following equation: $\mathrm{d}(t_1, t_2) = \max_{s_1 \in \mathrm{sen}(t_1), s_2 \in \mathrm{sen}(t_2)} d(s_1, s_2)$, where t_1 and t_2 are two terms, and $\mathrm{sen}(t_i)$ is the set of senses of t_i.
2) Creating "islands of meaning". Various distributional term clustering algorithms have been proposed to collapse similar terms onto one feature [26], [27], [28]. Lexical resource-based distances could probably also be used in term clustering. However, in the model proposed here the aim was... and use all semantic relations between them to improve classification performance. This section proposes an algorithm to provide an appropriate semantic ordering of terms.

Let V denote a set of terms $\{t_1, t_2, \ldots, t_n\}$ and let $d(t_i, t_j)$ denote the semantic distance between the terms t_i and t_j. The initial order of the terms is not relevant, though it is assumed to be alphabetic.

Let $G = (V, E)$ denote a weighted undirected graph, where the weights in the set E are defined by the distances between the terms.

Finding a semantic ordering of terms can be translated to a graph problem: a minimum-weight Hamiltonian path G' of G gives the ordering by reading the nodes from one of the paths to the other. G is a complete graph, therefore such a path always exists, but finding it is an NP-complete problem. The following greedy algorithm is similar to the nearest neighbor heuristic for the solution of the traveling salesman problem. It creates a graph $G' = (V', E')$, where $V' = V$ and $E' \subset E$. This G' graph is a spanning tree of G in which the maximum degree of a node is two, that is, the minimum spanning tree is a path between two nodes.

Step 1 Find the term at the highest stage of the hierarchy in a lexical resource.

$$t_s = \mathrm{argmin}_{t_i \in V} \mathrm{depth}(t_i).$$

This seed term is the first element of V', $V' = \{t_s\}$. Remove it from the set V:

$$V := V \backslash \{t_s\}.$$

Step 2 Let t_l denote the leftmost term of the ordering and t_r the rightmost one. Find the next two elements of the ordering:

$$t'_l = \mathrm{argmin}_{t_i \in V} d(t_i, t_l),$$

$$t'_r = \mathrm{argmin}_{t_i \in V \backslash \{t'_l\}} d(t_i, t_r).$$

Step 3 If $d(t_l, t'_l) < d(t_r, t'_r)$ then add t'_l to V', $E' := E' \cup \{e(t_l, t'_l)\}$, and $V := V \backslash \{t'_l\}$. Else add t'_r to V', $E' := E' \cup \{e(t_r, t'_r)\}$ and $V := V \backslash \{t'_r\}$.

Step 4 Repeat from *Step 2* until $V = \emptyset$.

The computational cost of the algorithm is $O(n^2)$. The above algorithm can be thought of as a modified Prim's algorithm, but it does not find the optimal minimum-weight spanning tree.

Given that the distance function involves counting edges of a semantic network, the validity of the above algorithm is discussed as follows.

- The ordering is possible: Starting from the seed term, the remaining set V will always contain elements which

either share the same hypernym or are hypernyms of each other.
- The ordering is good enough: The quality will also depend on the lexical resource in question. Further, the complexity of human languages makes the creation of even a near perfect semantic network of its concepts impossible. Thus in many ways the lexical resource-based measures are as good as the networks on which they are based.

2) Semantic Kernels in the L_2 Space: The L_2 space shares resemblance with a real vector space. Real-valued vectors are replaced by square-integrable functions, and the dot product is replaced by the following inner product:

$$(f_i, f_j) = \int f_i f_j dx,$$

for some f_i, f_j in the given L_2 space.

Lately, Hoenkamp has also pointed out that the L_2 space can be used for information retrieval when he introduced a Haar basis for the document space [29]. He utilized a signal processing framework within the context of latent semantic indexing. In order to apply an L_2 representation for text classification, the problem is approached from a different angle than by Hoenkamp, taking discounting expansion terms as our point of departure.

Assigning a function $w(x-k)$ to the term in the kth position in a semantic order, a document j can be expressed as follows:

$$f_j(x) = \sum_{k=1}^{M} a_{kj} w(x-k), \tag{5}$$

where x is in $[1, M]$, and it is the variable of integration in calculating the inner product of the L_2; x can be regarded as a "dummy" variable carrying no meaning in itself. The above formula will be referred to as a document function. In the experiments, the function $\exp(-bx^2)$ was used as $w(x)$, with b as a free parameter reflecting the width of the function expressing how many neighboring expansion terms are considered.

The inner product of the $L_2([1, M])$ space is applied to express similarity between two documents in similar vein as the dot product does in a real-valued vector space:

$$(f_i, f_j) = \int_{[1,M]} f_i(x) f_j(x) dx, \tag{6}$$
$$f_i, f_j \in L_2([1, M]),$$

where f_i and f_j are the representations of the documents in the L_2 space.

With the above formula, a matching term in two documents will be counted to its full tfidf score, while semantically related terms will be counted less and less according their semantic proximity to the matching term. Assuming that the terms *brand*, *brand name*, and *trade name* follow each other in the semantic order, consider the following example. The first document has the term *brand name*, and so does the second document. In Figure 1, it can be seen brand name

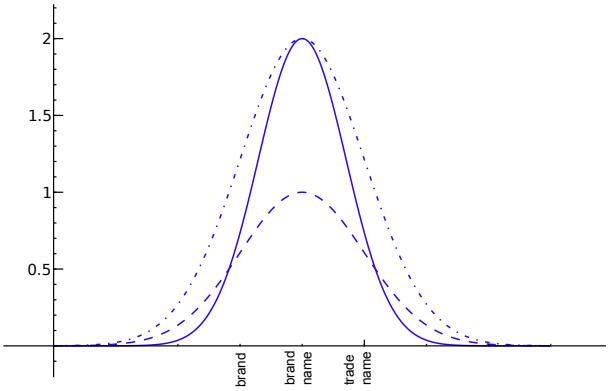

Fig. 1. Two documents with matching term *brand name*. Dotted line: Document-1. Dashed line: Document-2. Solid line: Their product as in Equation (6).

is counted the same way as it would be in a BOW model, brand and trade name are counted to a lesser extent, while other related terms are considered even less.

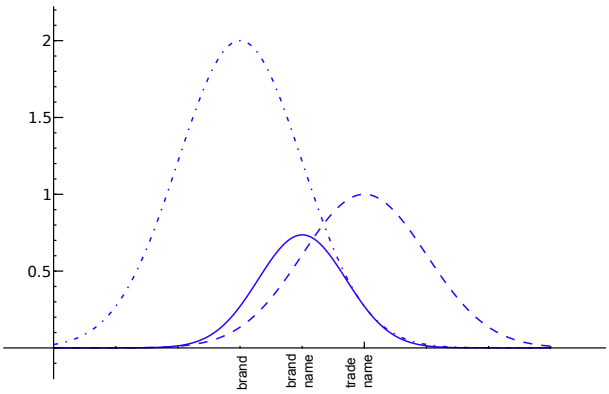

Fig. 2. Two Documents with no matching term but with related terms *brand* and *trade name*. Dotted line: Document-1. Dashed line: Document-2. Solid line: Their product as in Equation (6).

Now if the two documents do not share the exact term, only related terms occur, for instance, *trade name* and *brand*, respectively, then the term *brand name*, placed between *trade name* and *brand* in the s semantic order, will be considered to some extent for the calculation of similarity (see Figure 2).

The proposed kernel is mathematically valid. The similarity measure of the L_2 model defined by Equation 6 is an inner product. An inner product is a nondegenerate sesquilinear form, moreover it is positive definite. Let ϕ be the mapping that creates theL_2 functions from the discrete data, and X be a subspace of $\mathbb{R}^M$:

$$\phi : X \to L_2([1, M]).$$

The kernel underlying the model is

$$K(\mathbf{a}_i, \mathbf{a}_j) = (\phi(\mathbf{a}_i), \phi(\mathbf{a}_j))_{L_2}. \tag{7}$$

This kernel is positive definite, therefore the conditions of Mercer's Theorem are satisfied, hence the kernel is valid for any support vector machine.

The proposed kernel does not change the computational complexity of support vector machines other than the kernel evaluation cost. Given a linear kernel

$$K(\mathbf{a}_i, \mathbf{a}_j) = (\mathbf{a}_i, \mathbf{a}_j),$$

the computational cost of a kernel evaluation is O(M), if M is the number of features:

$$(\mathbf{a}_i, \mathbf{a}_j) = \sum_{k}^{M} a_{ki} a_{kj}.$$

To calculate the evaluation cost for the L_2 kernels, let us start from Equation 7:

$$K(\mathbf{a}_i, \mathbf{a}_j) = (\phi(\mathbf{a}_i), \phi(\mathbf{a}_j))_{L_2} = \tag{8}$$
$$\sum_{k=1}^{M} \sum_{l=1}^{M} a_{ki} a_{lj} \int_{[1,M]} \mathrm{w}(x-k)\mathrm{w}(x-l) d\lambda(x).$$

The above formula indicates a magnitude increase in the complexity (O(M^2)), however, it can be simplified. Since only the vicinity of a coordinate is interesting for the approximating function w(x), and the function is zero or almost zero for the rest of the domain, it is reasonable to calculate the score only for that vicinity, that is, for a feature j, the interval $[\max\{j - \lceil b \rceil, 1\}, \min\{j + \lceil b \rceil, M\}]$. Hence the above equation simplifies to

$$K(\mathbf{a}_i, \mathbf{a}_j) = (\phi(\mathbf{a}_i), \phi(\mathbf{a}_j))_{L_2} = \tag{9}$$
$$\sum_{k=1}^{M} \sum_{l=\max\{k-\lceil b \rceil, 1\}}^{\min\{j+\lceil b \rceil, M\}} a_{ki} a_{lj} \int_{[1,M]} \mathrm{w}(x-k)\mathrm{w}(x-l) d\lambda(x).$$

Thus the eventual kernel evaluation cost is O(bM).

III. Experimental Results

Text classification systems are normally evaluated by measures of effectiveness rather than efficiency, that is, its ability to take the right classification decisions. One approach is to calculate precision and recall with regard to a class c_k. Precision and recall should be interpreted together, they are not sensible measures of effectiveness in themselves. The F_1 measure is a composite measure of precision and recall.

The most widely used benchmark corpus is the Reuters-21578 collection. For benchmarking purposes, the ModApte split was adapted. 9603 documents were used as the training set and 3299 as the test set in the experiments. Only those ninety text categories which had at least one positive example in the training set were included in the benchmark. Another benchmark data corpus we used was the 20 Newsgroups corpus, which is a collection of approximate 20,000 newsgroup documents nearly evenly divided among 20 discussion groups and each document is labeled as one of the 20 categories corresponding to the name of the newsgroup that the document was posted to.

Prior to the semantic ordering, terms were assumed to be in alphabetic order. Measuring the Jiang-Conrath distance between adjacent terms, the average distance was 1.68 on the Reuters corpus (Figure 3). Note that the Jiang-Conrath distance was normalized to the interval $[0, 2]$. There were few terms with zero or little distance between them. This is due to terms which are related and start with the same word or stem. For example, *account, account executive, account for, accountable*.

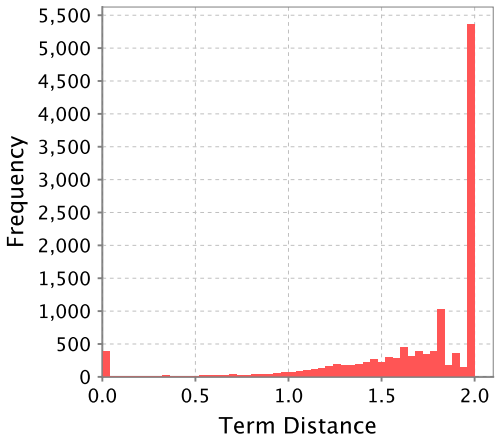

Fig. 3. Distribution of Distances Between Adjacent Terms in an Arbitrary Order

The same average distance after reordering the terms with the proposed algorithm and the Jiang-Conrath distance was 0.56 on the same corpus (Figure 4). About one third of the terms had very little distance between each other. Nevertheless, over 10 % of the total terms still had the maximum distance. This is due to the non-optimal nature of the proposed term-ordering algorithm. These terms add noise to the classification. The noisy terms occur typically at the two sides of the scale, that is, the leftmost terms and the rightmost terms. While it is easy to find close terms in the beginning, as the algorithm proceeds, fewer terms remain in the pool to be chosen. For instance, *brand, brand name, trade name, label* are in the 33rd, 34th, 35th and 36th position on the left side counting from the seed respectively, while *windy, widespread, willingly, whatsoever, worried, worthwhile* close the left side, apparently sharing little in common. The noise can be reduced by the appropriate choice of the parameter b in $\exp(-bx^2)$, so the impact of adjacent, but distantly related terms can be minimized.

We used the `libSVM` library [30] for benchmarking the classification performance. Table I shows the results on the two benchmark corpora with the baseline kernels. For all the kernels, the results with the best parameter settings are shown. Polynomial kernels were benchmarked between degrees 2 and 5.

The most direct relative to the proposed kernels is the linear semantic smoothing kernels. We constructed this kernel the same way as in [12], that is, the entries in the semantic proximity matrix (see Equation 3) were the inverse of the

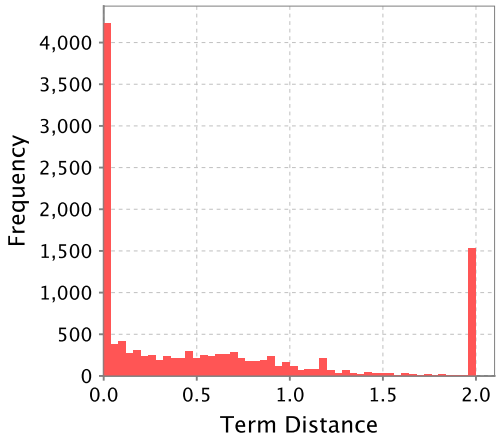

Fig. 4. Distribution of Distances Between Adjacent Terms in a Semantic Order Based on Jiang-Conrath Distance

Kernel	Reuters Micro-F_1	Reuters Macro-F_1	20News Micro-F_1	20News Macro-F_1
Linear	0.900	0.826	0.801	0.791
Linear (Semantic)	0.908	0.826	0.807	0.794
Poly	0.903	0.824	0.796	0.788
L_2	0.911	0.835	0.813	0.799

TABLE I

MICRO- AND MACRO-AVERAGE F_1 RESULTS

length of the path linking the two corresponding words in the WordNet hypernym hierarchy. Indeed, this linear semantic kernel was more compelling than linear and polynomial kernels. However, the proposed L_2 kernels are able to outperform all the baseline kernels, and the differences in micro-averaged results are statistically significant in all cases.

L_2 kernels were benchmarked with width b between 1 and 8, where b refers to the width of the approximating function. With $b = 1$, the results are identical with the linear kernel, which was expected, since if we do not consider neighboring terms, then the L_2 kernel acts as a linear kernel. The performance peaked at 4 in all cases (Table II). As we introduce more and more consecutive terms, the results get noisy, hence a drop in performance.

IV. CONCLUSIONS

Information systems are in great need of automated intelligent tools, but existing algorithms and methods cannot be pushed much further. Most existing techniques are impaired by the semantically poor but widespread representation of information and knowledge. This research proposes an entirely new, more intuitive analogue representation of natural language and hence human knowledge.

The proposed model combines term expansion with the semantic relations and semantic relatedness used in semantic smoothing kernels. This slightly unusual approach needs to transform the real vector representation to the L_2 space, and the experimental results show that this new representation can improve text classification effectiveness.

Two, seemingly contradicting linguistic theories are reconciled at different levels in the model. The distributional hypothesis meets the referential theory of meaning first at the semantic ordering of terms. While high-quality lexical resources enable such ordering in themselves, the ordering can benefit from data derived from a specific corpus being studied: semantic relatedness measures such as the Jiang-Conrath similarity operate this way. Once the ordering is done, weights expressing statistical relationships between terms and documents are borrowed from the vector space model to form the basis for constructing hypothetical signals of content: the documents as continuous functions.

Kernel b	Reuters Micro-F_1	Reuters Macro-F_1	20News Micro-F_1	20News Macro-F_1
1	0.900	0.826	0.801	0.791
2	0.910	0.830	0.805	0.796
4	0.911	0.835	0.813	0.799
8	0.902	0.821	0.802	0.788

TABLE II

SENSITIVITY ANALYSIS OF PARAMETER b

REFERENCES

[1] V. Raghavan and S. Wong, "A critical analysis of vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 37, no. 5, pp. 279–287, 1986.

[2] A. Budanitsky and G. Hirst, "Evaluating WordNet-based measures of lexical semantic relatedness," *Computational Linguistics*, vol. 32, no. 1, pp. 13–47, 2006.

[3] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proceedings of ECML-98, 10th European Conference on Machine Learning*. Chemnitz, Germany: Springer-Verlag, London, UK, April 1998, pp. 137–142.

[4] N. Cristianini, J. Shawe-Taylor, and H. Lodhi, "Latent semantic kernels," *Journal of Intelligent Information Systems*, vol. 18, no. 2, pp. 127–152, 2002.

[5] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge University Press, 2004.

[6] C. J. van Rijsbergen, *The Geometry of Information Retrieval*. New York, NY, USA: Cambridge University Press, 2004.

[7] J. Hu, L. Fang, Y. Cao, H. Zeng, H. Li, Q. Yang, and Z. Chen, "Enhancing text clustering by leveraging Wikipedia semantics," in *Proceedings of SIGIR-08, 31st ACM International Conference on Research and Development in Information Retrieval*. Singapore: ACM Press, New York, NY, USA, July 2008, pp. 179–186.

[8] H. Peat and P. Willett, "The limitations of term co-occurrence data for query expansion in document retrieval systems," *Journal of the American Society for Information Science*, vol. 42, no. 5, pp. 378–383, 1991.

[9] G. Salton and M. Lesk, "Computer evaluation of indexing and text processing," *Journal of the ACM*, vol. 15, no. 1, pp. 8–36, 1968.

[10] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization," in *Proceedings of CIKM-98, 7th International Conference on Information and Knowledge Management*. Bethesda, MD, USA: ACM Press,New York, NY, USA, 1998, pp. 148–155.

[11] M. Porter, "An algorithm for suffix stripping," *Program: Electronic Library & Information Systems*, vol. 14, no. 3, pp. 130–137, 1980.

[12] G. Siolas and F. d'Alché Buc, "Support vector machines based on a semantic kernel for text categorization," in *Proceedings of IJCNN-00, IEEE International Joint Conference on Neural Networks*. Austin, TX, USA: IEEE Computer Society Press, Los Alamitos, CA, USA, 2000.

[13] D. Mavroeidis, G. Tsatsaronis, M. Vazirgiannis, M. Theobald, and G. Weikum, "Word sense disambiguation for exploiting hierarchical thesauri in text classification," *Proceedings of PKDD-05, 9th European Conference on the Principles of Data Mining and Knowledge Discovery*, pp. 181–192, October 2005.

[14] S. Bloehdorn, R. Basili, M. Cammisa, and A. Moschitti, "Semantic kernels for text classification based on topological measures of feature similarity," *Proceedings of ICDM-06, 6th IEEE International Conference on Data Mining*, December 2006.

[15] S. Wong, W. Ziarko, and P. Wong, "Generalized vector space model in information retrieval," in *Proceedings of SIGIR-85, 8th ACM International Conference on Research and Development in Information Retrieval*. Montréal, Québec, Canada: ACM Press, New York, NY, USA, 1985, pp. 18–25.

[16] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[17] A. Hotho, S. Staab, and G. Stumme, "WordNet improves text document clustering," in *Proceedings of SIGIR-03, 26th ACM International Conference on Research and Development in Information Retrieval*. Toronto, Canada: ACM Press, New York, NY, USA, July 2003.

[18] M. Rodriguez and J. Hidalgo, "Using WordNet to complement training information in text categorisation," in *Procedings of RANLP-97, 2nd International Conference on Recent Advances in Natural Language Processing*. John Benjamins Publishing, Amsterdam, The Netherlands, 1997.

[19] E. Gabrilovich and S. Markovitch, "Feature generation for text categorization using world knowledge," in *Proceedings of IJCAI-05, 19th International Joint Conference on Artificial Intelligence*, vol. 19. Edinburgh, UK: Lawrence Erlbaum Associates Ltd, 2005.

[20] S. Mohammad and G. Hirst, "Distributional measures as proxies for semantic relatedness," *Submitted for publication*, 2005.

[21] R. Richardson and A. Smeaton, "Using WordNet in a knowledge-based approach to information retrieval," in *Proceedings of the 17th BCS-IRSG Colloquium on IR Research*, Manchester, UK, April 1995.

[22] M. McHale, "A comparison of WordNet and Roget's Taxonomy for measuring semantic similarity," in *Proceedings of COLING-ACL Workshop on Usage of WordNet in Natural Language Processing Systems*. Montréal, Québec, Canada: ACl, Morristown, NJ, USA, August 1998, pp. 115–120.

[23] D. Lin, "Automatic retrieval and clustering of similar words," in *Proceedings of COLING-ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, vol. 98. Montréal, Québec, Canada: ACL, Morristown, NJ, USA, August 1998, pp. 768–773.

[24] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," in *Proceedings of IJCAI-95, 14th International Joint Conference on Artificial Intelligence*, vol. 1, Montréal, Québec, Canada, August 1995, pp. 448–453.

[25] J. Jiang and D. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," in *Proceedings of the International Conference on Research in Computational Linguistics*, Taipei, Taiwan, 1997, pp. 19–33.

[26] F. Pereira, N. Tishby, and L. Lee, "Distributional clustering of English words," in *Proceedings of the 31st Annual Meeting on ACL*. ACL, Morristown, NJ, USA, 1993, pp. 183–190.

[27] D. Koller and M. Sahami, "Toward optimal feature selection," in *Proceedings of ICML-96, International Conference on Machine Learning*, L. Saitta, Ed. Bari, Italy: Morgan Kaufmann Publishers, San Francisco, CA, USA, July 1996, pp. 281–289.

[28] L. Baker and A. McCallum, "Distributional clustering of words for text classification," in *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*. Melbourne, Australia: ACM Press, New York, NY, USA, August 1998, pp. 96–103.

[29] E. Hoenkamp, "Unitary operators on the document space," *Journal of the American Society for Information Science and Technology*, vol. 54, no. 4, pp. 314–320, 2003.

[30] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, 2001, URL: http://www.csie.ntu.edu.tw/˜cjlin/libsvm.