Wikipedia-based Kernels for Text Categorization

Zsolt Minier Zalán Bodó Lehel Csató
Department of Mathematics and Computer Science
Babeş-Bolyai University
RO-400084, Cluj-Napoca, Romania
Email: {minier, zbodo, csatol}@cs.ubbcluj.ro

Abstract

In recent years several models have been proposed for text categorization. Within this, one of the widely applied models is the vector space model (VSM), where independence between indexing terms, usually words, is assumed. Since training corpora sizes are relatively small – compared to $\approx \infty$ what would be required for a realistic number of words – the generalization power of the learning algorithms is low.

It is assumed that a bigger text corpus can boost the representation and hence the learning process. Based on the work of Gabrilovich and Markovitch *[6], we incorporate Wikipedia articles into the system to give word distributional representation for documents. The extension with this new corpus causes dimensionality increase, therefore clustering of features is needed. We use Latent Semantic Analysis (LSA), Kernel Principal Component Analysis (KPCA) and Kernel Canonical Correlation Analysis (KCCA) and present results for these experiments on the Reuters corpus.*

1. Introduction

Text categorization is a thoroughly examined task in information retrieval (IR), see for example [1]. The amount of textual information available these days makes necessary the intelligent and *efficient* methods that help navigating in this virtual "document space".

We study a "classical" categorization or classification task: given the training data $D = \left\{(\mathbf{d}_i, y_i)\right\}_{i=1}^{N}$, one has to find a candidate $\widehat{f}$ which "best" approximates f based on the given data. It is assumed that the *data* are possibly noisy observations of a *true* classifier function $f : \mathcal{D} \to 2^C$ where $\mathcal{D}$ is the set of all documents in a chosen representation and $C = \{1, \ldots, c\}$ denotes the set of categories with 2^C the set of all subsets thereof. From this setup, it is evident that a multi-label task is addressed: a document can belong to one or more classes; thus we model a multi-class document classification system.

In categorizing textual data, the problem is commonly separated in two phases: (1) term selection and (2) category learning. Both are equally important parts of a text categorization system: a good term selection might help removing "noise" or irrelevant features that might lead the system into wrong direction. On the other hand, machine learning techniques like Support Vector Machines for classification [17] and Rocchio's method [11] can filter out irrelevant terms.

The most commonly used model in information retrieval – hence in text categorization – is the vector space model (VSM) introduced in [13]. The textual document is transformed into *a single* vector where the dimensions – indices's or index term in what follows – are words or stems taken from the training documents, also referred to as corpus. From the VSM settings follows that index terms are considered to be independent despite the fact that words are evidently not semantically independent.

Due to its simplicity and good performance, VSM is the most popular representation method used in the IR community. Although other, often more sophisticated models were explored – like generalized vector space model (GVSM), latent semantic indexing/analysis (LSI/LSA), probabilistic models, fuzzy set models, see for example [1, Chapter 2] – they did not provide a significantly better performance. We use the vector space model representation in this article. An improvement to the VSM is its augmentation with a weighting scheme, where descriptive terms will get a higher weight in the representation. The term frequency×inverse document frequency (`tfidf`) weighting of a *word* in a document is defined as follows

$$w_{ij} = \mathrm{freq}(i, j) \cdot \log \frac{n_D}{n_i}$$

where w_{ij} denotes the weight of word i in document j; $\mathrm{freq}(i, j)$ is the frequency of word i in the jth document; n_D is the total number of documents in the corpus; and n_i

is the number of documents in which word i appears. This gives higher weights for more descriptive, more frequent, terms in a document – called term frequency `tf` – and also higher weights for terms which have more discriminative power – inverse document frequency `idf`. Let us use $\mathbf{D}$ for the term×document matrix, where the entry w_{ij} contains the frequency of term i in the jth document, written as

$$\mathbf{D} = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n_D} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n_T1} & w_{n_T2} & \cdots & w_{n_Tn_D} \end{pmatrix}. \qquad (1)$$

n_T and n_D denote the number of terms and documents respectively. The `tfidf` transformation is $\Phi(\mathbf{d}) = \mathbf{Pd}$, where $\mathbf{d}$ denotes the document vector – a row of $\mathbf{D}$ – and $P = \mathrm{diag}(\mathbf{t})$, where $t_i = \log \frac{n_D}{n_i}$, $i = 1, 2, \ldots, n_T$.

For a good comparative study on the basic feature selection techniques in text categorization see the work of [19]. [16] gives a broad overview on different machine learning techniques applied to the problem of text categorization.

In this paper we study the use of Wikipedia-derived knowledge in enhancing text categorization. In the next section we describe the method of using Wikipedia as an external knowledge source in text categorization. Section 3 presents dimensionality reduction techniques which help us cope with the extreme dimensionality of Wikipedia. Section 4 shortly describes the machine learning algorithm we use. The paper ends with sections 5 and 6 discussing the results and conclusions drawn from the experiments.

2. Document classification using Wikipedia

According to the VSM model, each document $\mathbf{d}_j$ is *represented* in the vector space where each word has its associated dimension. Since documents have a *limited number* of words – compared to the set of all words from the corpus, document representations can be very sparse. We apply a linear transformation to document vectors, $\Phi(\mathbf{d}) = \mathbf{Pd}$, and we study the different possibilities for this transformation. Our approach can be considered as a transformation of the documents into the Wikipedia concept space and then to a latent space using dimensionality reduction techniques like PCA, KPCA, CCA and KCCA (see Figure 1).

Throughout the article any vector is a *column* vector.

For classification we use Support Vector Machines, one of the most successful supervised learning techniques on the text categorization task.

2.1. Wikipedia

The Wikipedia[1] is the largest encyclopedia edited collaboratively on the internet comprising of approximately

[1] http://wikipedia.org

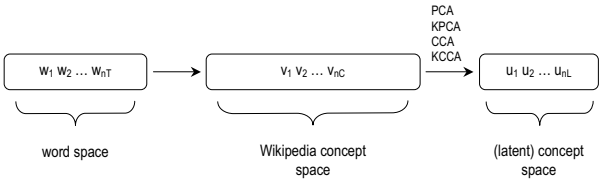

Figure 1. Schematic representation of our method

1.6 million articles totalling to ≈ 8 gigabytes of textual data (including metadata, i.e. XML tags). It is written in a clear and coherent style with many concepts sufficiently explained, making it an excellent resource for natural language research.

The disadvantage of the VSM model is that it considers words as unrelated or independent, which is evidently a false in natural languages. Sometimes we introduce term- (word-) relatedness based either on corpus statistics or some external knowledge. Here we use the latter approach, assuming that introducing information from an external knowledge source would help us solving the problem. Our need for a semantic relatedness metric between words could be easily approached using the distribution of words in the different Wikipedia concepts. Throughout the article we use the notion of Wikipedia concept and article interchangeably.

Gabrilovich and Markovitch [6] use word distributional representation for measuring word and text *relatedness*, using Wikipedia articles. We adopt their technique to represent documents in this concept space. Given a document $\mathbf{d}$ it can be transformed to the Wikipedia concept space by

$$\underbrace{\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1n_T} \\ c_{21} & c_{22} & \cdots & c_{2n_T} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_C1} & c_{n_C2} & \cdots & c_{n_Cn_T} \end{pmatrix}}_{\mathbf{W}} \cdot \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n_T} \end{pmatrix}}_{\mathbf{d}}$$

where the w_{ij} are the weights of the words from eq. (1), and c_{ij} represents the weight of the word i in Wikipedia concept j; n_C denotes the number of Wikipedia concepts. It is easy to observe that the matrix $\mathbf{W}$ is a concept×term matrix, which transforms documents to the concept space. The transformation is similar to the map used in GVSM-kernels [18], [4], the co-occurrence matrix now built upon external information,

$$k_{\mathrm{GVSM}}(\mathbf{d_1}, \mathbf{d_2}) = \mathbf{d}_1' \mathbf{W}' \mathbf{W} \mathbf{d}_2$$

where $\mathbf{W}'\mathbf{W}$ is the term×term co-occurrence matrix, which in the GVSM is simply $\mathbf{D}'\mathbf{D}$. This construction creates the connection between two terms by looking at

the concepts/documents where these occur together, that is entry ij will be nonzero if there is at least one article/document where both the terms show up.

Although both the matrices $\mathbf{D}$ and $\mathbf{W}$ are sparse (with a density of 0.67% and 0.2%, respectively), their product results in a much more denser structure. Though – similarly to [6] – we filtered out Wikipedia articles considered less important, the resulting matrix remains still huge and dense, thus making difficult to store and actually use the constructed document vectors.

Kernels are used to map the data points to a higher dimensional feature space and perform the classification – or some other – task there. Mercer's theorem states that any continuous symmetric, positive definite kernel function can be expressed as a dot product in a high-dimensional space [15, pp. 37–38]. Therefore to apply kernelization to a problem it has to be rewritten in terms of dot products between data points. Kernel methods cover all geometric constructions involving angles, distances and lengths. We consider the general purpose kernels: linear, polynomial and RBF or Gaussian kernel [15]:

$$\begin{aligned} k_{\text{linear}}(\mathbf{x}_i, \mathbf{x}_j) &= \mathbf{x}_i \cdot \mathbf{x}_j = \sum_k \mathbf{x}_{ik}\mathbf{x}_{jk} \\ k_{\text{poly}}(\mathbf{x}_i, \mathbf{x}_j) &= (b\mathbf{x}_i \cdot \mathbf{x}_j + c)^d \\ k_{\text{rbf}}(\mathbf{x}_i, \mathbf{x}_j) &= \exp\left\{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right\} \end{aligned}$$

2.2. Link structure of Wikipedia

Wikipedia possesses a link structure too, which can be used efficiently by propagating frequencies through these connections, thus lexical distributions become semantic distributions. Consider the concept×concept link matrix $\mathbf{E}$ which is defined as $E_{ij} = 1$ if there is a link from the ith to the jth concept and 0 otherwise. Because we want to keep the already assigned weights we set the main diagonal to be all zeros. Thus our updated Wikipedia-matrix becomes $\widetilde{\mathbf{W}} = \mathbf{E}'\mathbf{W}$.

2.3. Concept weighting

While words/dimensions in the training corpus get varying importance by using the `tdidf` weighting the concepts are treated of equal importance in Wikipedia. A possible weighting scheme could be achieved by ranking Wikipedia articles based on citation or reference analysis, so these importances could be extracted from the link structure discussed in the previous section. The famous PageRank algorithm [12] of Google does this considering only the link or citation structure of a set of hyperlinked pages: a page has a high rank if it has many back-links or there are only a few but important back-links. This can formulated recursively by

$$\mathbf{r}_i = \sum_{j \in N^{-1}(i)} \frac{\mathbf{r}_j}{|N^{-1}(j)|}$$

where $\mathbf{r}(i)$ denotes the rank of the ith page and $N^{-1}(i)$ is the set of backward neighbors of i pointing to it. If $\mathbf{P}$ denotes the adjacency matrix having $P_{ij} = 1/|N(i)|$, where $N(i)$ represents the forward neighbors of i, then the ranks can be calculated by solving the eigenproblem $\mathbf{r} = \mathbf{P}'\mathbf{r}$. PageRank can be viewed as a random walk on a graph and the ranks are the probabilities that a random surfer is at the page at time step k. If k is sufficiently large then the probabilities converge to a unique fixed distribution. The only problem with the above formulation is that the graph may have nodes with no forward neighbors and groups from which no forward links go out. To solve these problems one can add a little randomness to the surfing process,

$$\mathbf{r} = c\mathbf{P}'\mathbf{r} + (1-c)\mathbf{1}, \quad c \in [0,1]$$

where denotes the all 1 column vector of size n_p having n_p pages. In our experiments we used the value $c = 0.85$ according to [12].

Using the ranks produced by the PageRank algorithm we replace $\mathbf{W}$ by $\text{diag}(\mathbf{r})\mathbf{W}$. Figure 2 shows the magnitude of the ranks obtained while on Table 1 some of the best and worst ranked Wikipedia article titles are shown. A discussion about the resulting ranking would be quite interesting but because of lack of space we omit it here.

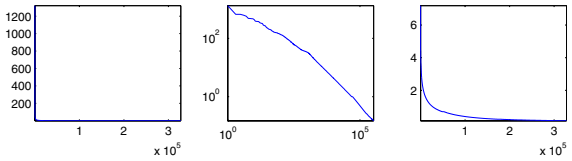

Figure 2. The decrease of the PageRank index in normal (left); and on log-log (ceentre) scale; on the right is the decrease with our rank transformation $\widetilde{\mathbf{r}} = \log(\mathbf{r} + \mathbf{1}_{n_p})$

3. Dimensionality reduction

To address the high dimensionality of vector spaces in natural language processing, dimensionality reduction techniques are often used. Dimensionality reduction helps at making feature vectors small enough to be handled by machine learning methods and many times has the beneficial effect of removing noise and thus slightly increasing classification accuracy and reducing overfitting.

Place	Article title	No.	Rank	Place	Article title	No.	Rank
1.	united states	231 770	1320.84328	20.	wikipedia:people by year/reports/canadians/for ...	139 249	0.15015
2.	united kingdom	12 118	667.21293	19.	wikipedia:dead external links/301/m	310 077	0.15014
3.	2006	13 420	666.54867	18.	wikipedia:dead external links/301/c	310 064	0.15014
4.	2005	13 417	622.37032	17.	wikipedia:dead external links/301/a	310 060	0.15014
5.	france	288 348	571.75942	16.	wikipedia:reference desk archive/humanities/ja ...	238 364	0.15014
6.	2004	13 367	481.86905	15.	wikipedia:reference desk archive/science/janua ...	238 365	0.15013
7.	world war ii	12 585	480.33496	14.	wikipedia:reference desk archive/miscellaneous ...	212 383	0.15013
8.	germany	4415	470.65791	13.	wikipedia:dead external links/301/s	310 084	0.15012
9.	england	3421	467.52380	12.	wikipedia:reference desk archive/science/octob ...	212 380	0.15012
10.	canada	271 492	440.91171	11.	list of songs containing overt references to r ...	71 582	0.15012
11.	2003	13 419	391.87985	10.	list of performers on top of the pops	302 455	0.15010
12.	australia	264 104	382.22329	9.	wikipedia:version 1.0 editorial team/computer a ...	281 658	0.15010
13.	japan	5829	382.05746	8.	wikipedia:version 1.0 editorial team/physics ar ...	279 053	0.15010
14.	english language	3393	377.01480	7.	wikipedia:version 0.5/biglist2	322 821	0.15008
15.	europe	3390	376.93508	6.	wikipedia:version 0.5/full list	322 828	0.15007
16.	india	5432	346.36223	5.	wikipedia:version 1.0 editorial team/assessment ...	269 869	0.15007
17.	2002	13 353	338.49065	4.	wikipedia:version 1.0 editorial team/release ve ...	327 212	0.15006
18.	london	6705	332.89629	3.	wikipedia:wikiproject automobiles/articles	295 708	0.15005
19.	2001	13 052	326.38888	2.	wikipedia:airline destination lists	247 538	0.15005
20.	italy	5431	325.82783	1.	wikipedia:bad links/http1	221 406	0.15005

Table 1. Best and worst ranked articles in the reduced Wikipedia set – $327\,653$ pages

3.1. Latent semantic analysis and latent semantic kernels

Latent Semantic Analysis (LSA) or Latent Semantic Indexing (LSI) was introduced by [5] in information retrieval. It is a dimensionality reduction technique based on singular value decomposition. Given the term×document matrix $\mathbf{D}$, it is decomposed as $\mathbf{D} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}$ where $\mathbf{U}$ and $\mathbf{V}$ are matrices of orthonormal columns called singular vectors while $\mathbf{S}$ is diagonal and contains the singular values, usually in decreasing order – of course the singular vectors are arranged according to $\mathbf{S}$. There is a connection between eigenvectors and singular vectors: $\mathbf{U}$ contains the eigenvectors of $\mathbf{D}\mathbf{D}'$, $\mathbf{V}$ contains the eigenvectors $\mathbf{D}'\mathbf{D}$ and the diagonal of $\sqrt{\mathbf{S}}$ constitutes the common eigenvalues of these matrices. By truncating $\mathbf{S}$, i.e. keeping only the first k largest values one obtains the best k rank approximation of $\mathbf{D}$ in terms of the Frobenius norm, therefore SVD is a powerful dimensionality reduction technique as Principal Component Analysis (PCA) removing the noise from the data by neglecting the irrelevant dimensions.

Document-document and term-term similarities in this semantic space induced by SVD can be written as

$$\mathbf{D}'\mathbf{D} = \mathbf{V}\mathbf{S}_k\mathbf{U}'\mathbf{U}\mathbf{S}_k\mathbf{V}' = \mathbf{V}\mathbf{S}_k^2\mathbf{V}' = \mathbf{V}\mathbf{S}_k(\mathbf{V}\mathbf{S}_k)'$$

and

$$\mathbf{D}\mathbf{D}' = \mathbf{U}\mathbf{S}_k\mathbf{V}'\mathbf{V}\mathbf{S}_k\mathbf{U}' = \mathbf{U}\mathbf{S}_k^2\mathbf{U}' = \mathbf{U}\mathbf{S}_k(\mathbf{U}\mathbf{S}_k)'$$

that is documents and terms can be viewed as the rows of $\mathbf{V}\mathbf{S}_k$ and $\mathbf{U}\mathbf{S}_k$ respectively. To fold in documents into this space we use the transformation

$$\mathbf{U}_k'\mathbf{d}$$

which comes directly from the decomposition of $\mathbf{D}$. To obtain $\mathbf{V}\mathbf{S}_k$ we simply multiply $\mathbf{D}$ by $\mathbf{U}_k'$ from left; thus a document becomes $\widehat{\mathbf{d}} = \mathbf{U}_k'\mathbf{d}$.

Our aim is to reduce the space of words using Wikipedia, that is map the documents to a concept space – not Wikipedia concepts but concepts formed by the initial indexing terms. Hence the SVD of $\mathbf{W}$ would be needed but it is simpler to perform the eigenvalue decomposition of $\mathbf{W}'\mathbf{W}$. If we decompose $\mathbf{W} = \mathbf{U}\mathbf{S}\mathbf{V}'$ – here $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}$ are not the same matrices as before – then $\mathbf{W}$ can be approximated by the lower rank matrix $\widehat{\mathbf{W}}$, $\mathbf{W} \approx \widehat{\mathbf{W}} = \mathbf{U}\mathbf{S}_k\mathbf{V}'$. The kernel can now be written as $k(\mathbf{d}_1, \mathbf{d}_2) = \mathbf{d}_1'\widehat{\mathbf{W}}'\widehat{\mathbf{W}}\mathbf{d}_2 = (\mathbf{S}_k\mathbf{V}'\mathbf{d}_1)'(\mathbf{S}_k\mathbf{V}'\mathbf{d}_2)$. We neglect the singular values $\mathbf{S}_k$ taking $\mathbf{I}$ instead, because using it some dimensions got a very high value which resulted in decrease of performance. Another interpretation would be the following: as the first component of the SVD decomposition of $\mathbf{D}$ contain the eigenvectors or principal components of the feature space in LSA, in this case $\mathbf{V}$ will contain it; we assume that Wikipedia articles yield a better covariance. That is we simply replace $\mathbf{D}\mathbf{D}'$ by $\mathbf{W}'\mathbf{W}$.

Finally our kernel becomes

$$k(\mathbf{d_1}, \mathbf{d_2}) = \mathbf{d}_1'\mathbf{V}_k\mathbf{V}_k'\mathbf{d}_2 \qquad (*)$$

where $\mathbf{V}_k$ denotes the eigenvectors of $\mathbf{W}'\mathbf{W}$ arranged in columns. Our approach is similar to the one presented in [4] which is a kernelization of LSA.

3.2. Kernel PCA

In the previous section we did not need the full SVD of $\mathbf{W}$ but only the eigenvectors of $\mathbf{W}'\mathbf{W}$, which are the principal components of the covariance of the words in Wikipedia. PCA is a linear technique thus unable to capture non-linear dependencies between words. One way to overcome this limitation is by using kernel methods.

For a set of centered data $\mathbf{S}_x$ consisting of $\mathbf{x}_i$ points the correlation matrix is $C = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i\mathbf{x}_i'$ that yields the stan-

dard eigenproblem $\lambda\mathbf{v} = \mathbf{C}\mathbf{v}$ whose solutions are the principal components. [14] showed that this eigenproblem can be cast in terms of kernel methods by rewriting the correlation matrix in kernel form: $K_{ij} = \frac{1}{m}\Phi(\mathbf{x}_i)'\Phi(\mathbf{x}_j) = \frac{1}{m}k(\mathbf{x}_i, \mathbf{x}_j)$, $i, j = 1, \ldots, m$, which can be solved in a similar way to the simple version.

The difference between obtaining the principal components in the input and the feature space is that the eigendecomposition is performed on the covariance of the words in the feature space (kernel matrix) instead of covariance matrix in input space. Hence our document kernel is the same as $(*)$ but $\mathbf{V}_k$ now contains the first k eigenvectors of $\mathbf{K}$ arranged in columns.

3.3. Kernel CCA

The approaches presented previously can be considered as unsupervised techniques revealing the hidden structure of words in Wikipedia. In order to reconcile the strucure of words present in Wikipedia and the training corpus we propose using Canonical Correlation Analysis (CCA).

CCA was introduced by [8] and is an optimization problem of finding basis vectors ($\mathbf{w}_x$ and $\mathbf{w}_y$) for two sets of random vectors ($\mathbf{S}_x$ and $\mathbf{S}_y$) such that the projections of the vectors on the bases have maximal correlation.

$$\rho = \max_{\mathbf{w}_x, \mathbf{w}_y} \frac{\langle \mathbf{S}_x\mathbf{w}_x, \mathbf{S}_y\mathbf{w}_y \rangle}{\|\mathbf{S}_x\mathbf{w}_x\| \, \|\mathbf{S}_y\mathbf{w}_y\|}$$

Denoting the covariance matrix of $\mathbf{x}$ and $\mathbf{y}$ as

$$\mathbf{C}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} \mathbf{C}_{\mathbf{xx}} & \mathbf{C}_{\mathbf{xy}} \\ \mathbf{C}_{\mathbf{yx}} & \mathbf{C}_{\mathbf{yy}} \end{bmatrix}$$

this maximization problem can be rewritten in the form

$$\rho = \max_{\mathbf{w}_x, \mathbf{w}_y} \frac{\mathbf{w}_x'\mathbf{C}_{xy}\mathbf{w}_y}{\sqrt{\mathbf{w}_x'\mathbf{C}_{xx}\mathbf{w}_x\mathbf{w}_y'\mathbf{C}_{yy}\mathbf{w}_y}}$$

[7] introduced a kernelized version of this problem by replacing the covariance matrix with a kernel matrix and adding regularization to overcome the overfitting that is very likely in the high dimensional kernel space. The optimization problem thus becomes:

$$\rho = \max_{\alpha, \beta} \frac{\alpha'\mathbf{K}_x\mathbf{K}_y\beta}{\sqrt{(\alpha'\mathbf{K}_x^2\alpha + \kappa\alpha'\mathbf{K}_x\alpha) \cdot \left(\beta'\mathbf{K}_y^2\beta + \kappa\beta'\mathbf{K}_y\beta\right)}}$$

[7] shows that the α vectors that maximize this function can be obtained as the eigenvectors of the following standard eigenproblem:

$$\left(\mathbf{K}_x + \kappa\mathbf{I}\right)^{-1} \mathbf{K}_y \left(\mathbf{K}_y + \kappa\mathbf{I}\right)^{-1} \mathbf{K}_x\alpha = \lambda^2\alpha$$

The choice of the κ regularization parameter is usually done empirically.

In our case $\mathbf{K}_x$ and $\mathbf{K}_y$ is the kernel matrix over words in the training corpus and Wikipedia respectively. Using KCCA (kernel CCA) the solutions α of the above eigenproblem will constitute the first k columns of the projection $\mathbf{V}_k$ in $(*)$.

4. Learning the training data

First we built the bag-of-words document representation with `tfidf` weighting of the training corpus. These vectors are transformed with the studied linear transformations of the form $\Phi(\mathbf{d}) = \mathbf{V}_k'\mathbf{d}$ and the resulting vectors are used as the input of the learning method.

For learning the decision boundaries we applied support vector machines using the LIBSVM [2] library. Support vector machines (SVMs) were introduced by Vapnik for binary classification. Although the formulation of the problem can be extended in some ways to handle multi-class classification, in most cases – because of the lower computational cost – binarization techniques like one-vs-rest, one-vs-one, error correcting output coding (ECOC), etc. are used for supporting non-binary classification.

SV classifiers maximizes the margin of the hyperplane which separates positive and negative examples. In its Wolfe dual form the problem is written as the maximization of the convex function

$$\mathbf{a}' \cdot \mathbf{1} - \frac{1}{2}\mathbf{a}'\mathbf{G}\mathbf{a}$$

in terms of the Lagrange multipliers $\mathbf{a}$ such that

$$0 \leq \alpha_i \leq C, i = 1, \ldots, m \quad \text{and} \quad \mathbf{a}'\mathbf{y} = 0$$

where $\mathbf{a} = (\alpha_1 \ldots \alpha_m)'$, $\mathbf{y} = (y_1 \ldots y_m)'$, $G_{ij} = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$, $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \in X \times Y \subseteq \mathbb{R}^n \times \{-1, 1\}$. After solving the problem the decision function becomes

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i\alpha_i^* k(\mathbf{x}, \mathbf{x}_i) + b^*\right),$$

$$b^* = y_i - \mathbf{w}^{*\prime}\Phi(\mathbf{x}_i) = y_i - \sum_{j=1}^{m} y_j\alpha_i^* k(\mathbf{x}_j, \mathbf{x}_i)$$

The separating hyperplane of maximal margin depends only on the *support vectors*, vectors supporting the marginal hyperplanes. Joachims [9] experimentally proved that the classes induced by the documents from the Reuters corpus are more or less linearly separable using the vector space model. We assumed the same holds for our model.

5. Experimental methodology and results

The first step constituted building an inverted index for the words appearing in Wikipedia (we used the dump of

November, 2006), excluding stop words and also the first 300 most frequent words. Because the large amount of Wikipedia articles turns the problem into one of unmanageable size, as we have mentioned earlier, we filtered out some of them, namely we eliminated those containing less than 500 words or having less or equal than 5 forward links to other Wikipedia articles. In this way we were left with 327 653 articles. The words appearing in less than 10 Wikipedia articles were neglected too. When building the feature vector of the words in Wikipedia the single occurences of words were removed considering them as "random" appearances.

For testing the model we used the Reuters-21578 text categorization corpus, ModApté split with $90+1$ categories. The Reuters collection contains documents which appeared on the Reuters newswire in 1987. These documents were manually categorized by the personnel from Reuters Ltd. and Carnegie Group Inc. in 1987. The collection was made available for scientific research in 1990. Originally, there were 21 578 documents, but some of them, namely 8681 were unused in the split, moreover, some of it were not categorized – they were put in the unknown category. Removing this "virtual" class, the training and the test corpus contains 9583 and 3744 documents respectively, defining 90 classes. Some of the documents are assigned to more than one category, the average number of classes per document being 1.24 [3].

In the preprocessing step we selected the top 5209 word stems of the Reuters corpus using the χ^2 term ranking method [19]. For these 5209 terms the inverted index was built based on the Wikipedia, that is, each term is represented as a vector of occurences in the vector space of Wikipedia concepts. The number of dimensions of this vector space is 327 653.

The best results we obtained are shown on Table 2 while we also listed the performance results obtained using only the first 1000 eigenvectors extracted with KPCA and KCCA on Table 3.

The performance is measured using the common precision-recall breakeven point – the intersection of the precision and recall curves if such a point exists – and the F_1 measure.

We measured the baseline performance of the system with the terms extracted using the χ^2 ranking method [19] from the corpus itself. We used these results as a baseline for another term selection method which resulted in 5209 terms [10] and we used this number of terms for baseline here as well.

To show the usefulness of PCA, we transformed the documents with the word covariance derived from Wikipedia (Table 2 line 2). The low results obtained here could be explained by the presence of the term $\mathbf{S}^4$ derived from the decomposition of $\mathbf{W}'\mathbf{W}$ that appears in the covariance distorting the weights of certain terms.

The covariance matrix was decomposed using PCA and the best performance is reported in Table 2, line 3.

We performed the link structure inclusion described in Section 2.2 and the PageRank weighting of the concepts presented in Section 2.3. The results can be seen in Table 2, lines 4 and 5.

To improve the performance of PCA by capturing nonlinear dependencies between words in Wikipedia we performed KPCA (Section 3.2) decomposition of the kernel matrix. We used the polynomial kernel with generic parameters ($b = 1/n_C$, $c = 1$, $d = 2$) and the rbf kernel also with default parameter setting ($2\sigma^2 = n_C$). The results are in Table 2, lines 6 and 7.

Last KCCA (Section 3.3) was used to discover the common structure of words in Reuters and Wikipedia using linear, polynomial and rbf kernels with the same settings as for KPCA.

6. Conclusions

Using the Wikipedia-based representation for word similarity, the authors of [6] obtained a much higher correlation with human judgement than previous methods. We used the same word representation while documents were built as the weighted sum of the words appearing in it. One possible explanation of the below-baseline performances could be the new document representations can not be discriminated well.

Using kernels did not yield better performance than the linear methods and this shows that there is sufficient information in the distribution of words over Wikipedia concepts to establish a good relatedness measure with linear methods.

Another explanation for the results of the experiments could be that the distribution of words in Reuters is significantly different from the distribution of Wikipedia so that no useful principal components can be extracted. This is somewhat supported by the fact that KCCA performed better than KPCA, especially when only a limited number (1000) of principal components were used. Another linguistic argument is that the usage of words has somewhat shifted from 1987 – when Reuters was written – to this day.

One possible improvement could be to cut off extremities from the transformation matrix $E'E$ to increase the sparseness of document representations and decrease possible noise.

The method should be tested on other corpora, to verify that semantic relatedness – derived from Wikipedia – can help categorization. The Reuters collection is very unbalanced, causing serious difficulties for text categorization systems.

These methods using Wikipedia can be used as semantic similarity measures between different chunks of text so they

	#eigenvectors	mP	mR	mBEP	mF1	MP	MR	MBEP	MF1
χ^2	–	88.46	84.59	86.52	86.48	71.61	61.25	66.43	66.02
Wikipedia covariance	–	48.95	35.42	42.18	41.10	8.79	5.35	7.07	6.65
PCA	4500	88.05	83.65	85.85	85.80	62.48	54.38	58.43	58.15
PCA+links	4500	87.89	83.71	85.80	85.75	65.11	56.05	60.58	60.24
PCA+PageRank	4000	87.44	83.68	85.56	85.52	59.75	53.86	56.80	56.65
poly KPCA	3500	87.80	83.97	85.89	85.84	65.88	57.21	61.54	61.24
rbf KPCA	3500	88.11	83.73	85.92	85.87	66.54	55.49	61.02	60.52
CCA ($\kappa = 10^4$)	4000	87.80	84.35	86.07	86.04	68.66	59.54	64.10	63.77
poly KCCA ($\kappa = 10$)	3000	87.66	84.27	85.97	85.93	63.78	56.08	59.93	59.68
rbf KCCA ($\kappa = 10^{-1}$)	1000	87.74	83.95	85.85	85.80	62.56	56.43	59.49	59.33

Table 2. Performance for the Reuters corpus in percentages. Notation: mP=micro-precision, mR=micro-recall, mBEP=micro-breakeven, mF1=micro-F_1, MP=macro-precision, MR=macro-recall, MBEP=macro-breakeven, MF1=macro-F_1

	mP	mR	mBEP	mF1	MP	MR	MBEP	MF1
PCA	83.26	78.12	80.69	80.61	52.64	41.65	47.13	46.49
PCA+links	83.95	80.02	81.98	81.94	55.84	46.79	51.32	50.92
PCA+PageRank	83.98	80.66	82.32	82.29	52.36	46.43	49.39	49.22
poly KPCA	80.23	75.21	77.72	77.64	47.14	39.83	43.48	43.18
rbf KPCA	84.20	80.42	82.31	82.27	60.15	50.59	55.37	54.96
CCA ($\kappa = 10^4$)	87.60	83.63	85.62	85.57	61.13	52.79	56.96	56.65
poly KCCA ($\kappa = 1$)	87.49	83.65	85.57	85.53	63.53	55.18	59.35	59.06
rbf KCCA ($\kappa = 10^{-1}$)	87.74	83.95	85.85	85.80	62.56	56.43	59.49	59.33

Table 3. Results for KPCA and KCCA using the first 1000 eigenvectors

could be used to segment documents that is a key step in our previous feature selection method [10].

Our experiments show that still the bag-of-words model performs better than the semantic space model of documents.

7. Acknowledgements

We acknowledge the support of the grants CEEX/1474, CNCSIS/TD-77 and CNCSIS/TD-35 by the Romanian Ministry of Education and Research.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[2] C. Chang and C. Lin. LIBSVM: a library for support vector machines (version 2.31), Sept. 07 2001.

[3] K. Crammer and Y. Singer. A new family of online algorithms for category ranking. In *SIGIR*, pages 151–158. ACM, 2002.

[4] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *J. Intell. Inf. Syst*, 18(2-3):127–152, 2002.

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, June 1990.

[6] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI)*, January 2007.

[7] D. R. Hardoon, S. Szedmák, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12), 2004.

[8] H. Hotelling. Relation between two sets of variables. *Biometrica*, 28:322–377, 1936.

[9] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In C. Nedellec and C. Rouveirol, editors, *Machine Learning: ECML-98, 10th European Conference on Machine Learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer, 1998.

[10] Z. Minier, Z. Bodó, and L. Csató. Segmentation-based feature selection for text categorization. In *Proceedings of the 2nd International Conference on Intelligent Computer Communication and Processing*, pages 53–59. IEEE, September 2006.

[11] A. Moschitti. A study on optimal parameter tuning for rocchio text classifier. In F. Sebastiani, editor, *Proceedings*

of ECIR-03, 25th European Conference on Information Retrieval, pages 420–435, Pisa, IT, 2003. Springer Verlag.

[12] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[13] G. Salton, A. Wong, and A. C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:229–237, 1975.

[14] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[15] B. Schölkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

[16] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[17] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[18] Y. Yang, J. G. Carbonell, R. D. Brown, and R. E. Frederking. Translingual information retrieval: Learning from bilingual corpora. *Artificial Intelligence*, 103(1–2):323–345, 1998.

[19] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, 1997.