Semantic Smoothing for Text Clustering

Jamal A. Nasir[a], Iraklis Varlamis[b], Asim Karim[a], George Tsatsaronis[c]

[a]*KADE Lab, Lahore University of Management Sciences, Pakistan*
[b]*Department of Informatics and Telematics, Harokopio University of Athens, Greece*
[c]*BIOTEC, Technical University of Dresden, Germany*

Abstract

In this paper we present a new *semantic smoothing* vector space kernel (*S-VSM*) for text documents clustering. In the suggested approach semantic relatedness between words is used to smooth the similarity and the representation of text documents. The basic hypothesis examined is that considering semantic relatedness between two text documents may improve the performance of the text document clustering task. For our experimental evaluation we analyze the performance of several semantic relatedness measures when embedded in the proposed (*S-VSM*) and present results with respect to different experimental conditions, such as: (i) the datasets used, (ii) the underlying knowledge sources of the utilized measures, and, (iii) the clustering algorithms employed. To the best of our knowledge, the current study is the first to systematically compare, analyze and evaluate the impact of semantic smoothing in text clustering based on '*wisdom of linguists*', e.g., *WordNets*, '*wisdom of crowds*', e.g., *Wikipedia*, and '*wisdom of corpora*', e.g., large text corpora represented with the traditional *Bag of Words* (*BoW*) model. Three semantic relatedness measures for text are considered; two knowledge-based (*Omiotis* [1] that uses *WordNet*, and *WLM* [2] that uses *Wikipedia*), and one corpus-based (*PMI* [3] trained on a semantically tagged *SemCor* version). For the comparison of different experimental conditions we use the *BCubed F-Measure* evaluation metric which satisfies all formal constraints of good quality cluster. The experimental results show that the clustering performance based on the *S-VSM* is better compared to the traditional *VSM* model and compares favorably against the standard *GVSM* kernel which uses word co-occurrences to compute the latent similarities between document terms.

Keywords:
Text Clustering, Semantic Smoothing Kernels, WordNet, Wikipedia,

1. Introduction

Document clustering plays an important role in indexing, retrieval, browsing and mining of large and high dimensional text data. Document clustering algorithms aim at organizing documents into meaningful groups that contain highly similar documents, and which are distant from the documents of other groups [4]. For this purpose, they rely on document similarity or distance measures, with which they typically compare pairs of text documents. Therefore, similarity measures play a crucial role in the task of document clustering. The performance of similarity measures in data mining tasks depends on the type of data, on the particular domain, on the dataset and on the nature of the examined task. In the case of document clustering, the textual data have usually large volume, they are high-dimensional, and carry also semantic information, i.e., meaning conveyed by the text terms. Therefore, the clustering algorithm and the similarity measures that are employed for the task should be able to address these parameters effectively.

In the task of document clustering documents are typically represented by their terms. Terms are either single words or composite (multi-word terms), which form as a whole the language vocabulary of the underlying text corpus. Terms of either category are usually associated with a positive real value acting as a weight for the respective term. Furthermore, the weight of each term corresponds to its importance/relevance to the document it appears in.

More formally, given a collection of documents D, the vocabulary V of D may be defined as the set of all distinct terms appearing in D. For each term t_i of a document $d_j \in D$, the weight $w_{ij} > 0$ of the t_i in d_j may be computed, usually, through a measure that takes into account the frequency of occupancies of t_i in d_j. This representation is widely known as the *Vector Space Model* (*VSM*) [5].

VSM is very simple and commonly used; yet, it has several limitations. Its main restriction is that it assumes independency between the vocabulary terms and ignores all the conceptual relations between terms that potentially exist. As a consequence, two terms that are semantically close, e.g., synonyms, are treated differently. Furthermore, polysemous terms, i.e., terms with multiple meanings, are considered the same in all contexts they appear. For example the term '*bank*' may have the meaning of a *financial institution* when it appears in a context related to economy, or the meaning of a *river side* when it appears in a context that refers to landscapes or geographical locations. In principle a document contains usually more terms that are '*general terms*', i.e., that may appear in all clusters, than '*cluster dependent terms*', i.e., '*core terms*' [6] that characterize the documents of a single cluster. *VSM* cannot consider that differentiation as it cannot examine similarities between terms that have different surface strings. For the *VSM* model, the similarities between documents and the similarities between a document and the cluster centroid are only based on the matched term strings. Hence, the need of smooth semantically the *VSM* model, i.e., by employing semantic smoothing *VSM* kernels, arises. This embedding may increase the importance of *core words* by considering the terms' relations, and in parallel downsize the contribution of *general terms*, leading to better text clustering results.

In this article, we propose a novel semantic smoothing *VSM* kernel (*S-VSM*), which smooths the *VSM* representation with the semantic similarity between terms[1]. The proposed *S-VSM* allows any semantic similarity or relatedness measure to be employed, both measures that use linguistic resources, e.g., knowledge bases, ontologies, and thesauri, but also measures that are based on statistical information extracted from the analysis of large text corpora. Hence, the first advantage of the suggested solution is that it offers a very flexible kernel that can be applied within any domain or with any language. To showcase the wide applicability of the suggested kernel, for the purposes of this work we examine the embedding of three novel semantic relatedness measures into the *S-VSM*; the first employs the *WordNet*-based similarity measure of *Omiotis* [1], the second is *Wikipedia*-based and employs the measure of Milne and Witte [2], and the third is based on statistical analysis of text corpora and uses a *Pointwise Mutual Information* similarity measure for the computation of terms' similarity [3].

The second advantage of the suggested solution is the ability of the *S-VSM* to perform much better than the *VSM* in the task of text clustering. In addition, an extension of the *S-VSM* that we propose, namely the *top-k S-VSM*, which considers only the top-k semantically related terms, does not only perform better than the *VSM*, but it also conducts the task of text clustering very

[1]Though there are slight conceptual differences between the terms '*semantic similarity*' and '*semantic relatedness*', for the purposes of this work this differentiation is not important. Therefore, the two terms might be used interchangeably for the remaining of the paper

efficiently in terms of time and space complexity. The proposed *S-VSM* and its extension are evaluated on five datasets: (1) *Reuters-Transcribed-set*[2], (2) *R8 Reuters-21578*[3], (3) *4 Universities Data Set* (*WebKB*)[4], (4) *Email-1431*[5], and, (5) *Movie Reviews*[6]. To evaluate *S-VSM* and *top-k S-VSM* we use both *agglomerative* and *partitional* clustering for conducting the experiments, and two baselines; the traditional Bag of Word (*BoW*) model which uses the *VSM* model for document representation, and the standard *Generalized Vector Space Model* kernel *GVSM*, which considers the term-to-document matrix to compute latent similarities between terms based on their co-occurrence.

The clustering results show significant improvements in the clustering accuracy when *S-VSM* and *top-k S-VSM* are used, compared to the performance of the two aforementioned baselines. In addition, we provide a thorough analysis on the effect of the number of the top-k semantically related terms used for the smoothing, which, to the best of our knowledge, is conducted for the first time in the bibliography, and gives important insights on how the semantic smoothing can be optimized computationally.

This work capitalizes on our previous work on semantic kernels [7]. The main contributions of the current work, which differentiate it from our former work on *S-VSM* kernels and expand it, can be summarized in the following:

1. Extension of the $S-VSM$ to embed only the top-k semantically related terms.
2. Application to the task of text clustering.
3. An extended and thorough evaluation in text clustering, using a large variety of text datasets, employed clustering algorithms, and evaluation metrics.
4. Comparative evaluation against the standard *GVSM* kernel and the semantic kernel presented in [8], which shows that the suggested expanded *S-VSM* performs favorably against these two approaches.

[2]Available for download from `http://archive.ics.uci.edu/ml/datasets/Reuters+Transcribed+Subset`

[3]Available for download from `http://web.ist.utl.pt/~acardoso/datasets/datasets.zip`

[4]From the *WebKB* project, available for download from `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/webkb-data.gtar.gz`

[5]Available for download from `http://cogsys.imm.dtu.dk/toolbox/nmf/email.zip`

[6]Available for download from `http://www.cs.cornell.edu/people/pabo/movie-review-data/review_polarity.tar.gz`

The rest of the paper is organized as follows: Section 2 discusses the related work in the field of semantic smoothing kernels, with emphasis to the task of text clustering. Section 3 provides preliminary related information. Section 4 introduces the semantic smoothing kernel (*S-VSM*) and its *top-k* extension. Section 5 presents the experimental setup, and Section 6 presents and analyzes the experimental results. Finally, we conclude the paper in Section 7 and provide a discussion on the possible expansions of the current work.

2. Related Work

The idea of using background knowledge or gathered statistical information from large text corpora analysis in order to compute text similarity is well studied in the past [9, 10], and there exist many research works that introduce efficient similarity or relatedness measures between terms. With regards to works that employ such measures for document clustering, *WordNet* is one of the most widely used lexical thesauri [11, 12]. In principle, research works in document clustering, but also in text retrieval, that incorporate semantics in the *VSM* representation can be classified in three categories, depending on the type of information or the features used to index the document terms and expand the index with additional features: (i) embedding of concept features, (ii) embedding of multi-word phrases, and (iii) employing semantic kernels to embed semantically related terms or semantic relation information between terms to the documents' representation; the semantic similarity and relations may be retrieved from a word thesauri or ontology, or may be computed based on statistical analysis of a large text corpus.

Works in the first category, e.g., [11], use conceptual features to improve the clustering performance. *WordNet* is typically used as a background knowledge to obtain concept features, which are defined as set of words that describe the same high level concept; for example *penny*, *nickel*, *dime* and *quarter* describe the concept *c*oin. The weights of both concepts and terms are employed to represent documents, usually leading to a type of *hybrid* document representation in the vector space, that contains both concepts, i.e., meanings, but also concrete terms. Such representations were also applied in the past in text retrieval, with mixed performance outcome[7] [14]. Another re-

[7]Usually such hybrid representations improve the recall in information retrieval sys-

cent representative example of a work in this category is the method proposed by Egozi et al. [15]. The authors in that work proposed an augmentation of the keyword-based representation of documents with concept-based features that are extracted from *Wikipedia*. The overall methodology is based on *Explicit Semantic Analysis* (*ESA*), and their experimental evaluation shows an important improvement when the suggested approach is used in information retrieval, compared to the standard *BOW* document representation.

The idea behind the works of the second category is that phrases have more expressive power than single words, and, thus, they expand the vocabulary index of the terms with associated word phrases to enhance document representation. For example, Mao et al. [16] expand the *VSM* model with phrase-based representations of documents that they obtain from the *Unified Medical Language System* (*UMLS*), which is one of the largest ontologies for the biomedical domain. In another work of the same category, SanJuan et al. [17] combine the *VSM* representation of documents with what they call *symbolic approach*, to enrich the representation with multi-word terms. The application in both works was in the domain of text retrieval, and more precisely, using text data from the medical domain, and the results in both cases show promise in expanding the *VSM* with phrase-based features. In another work, Zhoo et al. [18] embedded multi word phrases in the document representation, with their approach being domain agnostic. For the purposes of covering any domain, they used in their approach a context-sensitive smoothing method that statistically maps phrases to individual document terms.

Methods of both the aforementioned categories, however, share some limitations, such as: (i) increased dimensionality, as they actually both expand the *VSM* and the indexing size of the documents, (ii) increased computational cost, since the index size is increased, (iii) information loss, e.g., in cases when the ontology concepts replace the original terms, and noise when the ontology concepts are used as additional features due to poor word sense disambiguation performance, (iv) limited coverage when a comprehensive ontology is not available; though in the case of the biomedical domain the *UMLS* thesaurus is considered a very rich resource that describes in large coverage the life sciences, such a resource is not always available for other

tems, due to the expansion in the concepts' dimension, but might drop precision due to the difficulty of transiting from terms to concepts, with the task of word sense disambiguation having major role and innate limitations [13].

domains, (v) limited usage of the available semantic relationships, as in the majority of the cases in these two categories, the structure of the underlying ontologies or resources is not taken into account, and they are rather used as dictionaries, and not so much as ontologies.

To remedy the shortcomings of these two categories of works, research efforts that may be classified in the third category attempt to exploit efficiently semantic information from ontologies and thesauri, or results from statistical analysis of large text corpora. Such works utilize as many features as possible from the underlying ontologies, e.g., structure and relations, when ontologies are used, or they compute semantically related terms by analyzing text corpora. The resulting information in both cases is embedded into the document similarity computation by employing *semantic kernels* [19], and their optimizations to cover also semi-supervised learning [20]. *Semantic kernels* constitute a very efficient means of incorporating background knowledge in the task of computing similarity between documents, and, thus, allowing applications in text clustering and text classification. The notion of kernel usage to embed background information allows straightforward application of the resulting function to popular learners such as *Support Vector Machines* and *kernel Fischer discriminant analysis*, which are based on kernels.

Concerns regarding efficiency of computation through the embedding of such large scale background knowledge, e.g., semantic similarity information between the term features, can be addressed by utilizing sparse representations of the prediction function [21], a-priori computation of similarities between the term features [7], and retaining the *core* semantic features through the usage of *synonyms* [22]. In principle, the past few years several works have started appearing that apply efficiently semantic smoothing kernels, overcoming the scalability and complexity problems.

Examples of such recent works with application to text clustering, which exploit and embed efficiently background semantic knowledge for documents' similarity are: the work of Zheng et al. [23], the approach by Jing et al. [8], and the method suggested by Hu et al. [24]. Zheng et al. [23] explore the effectiveness of hypernymy, hyponymy, holonymy and meronymy information for nouns and and noun phrases, using *WordNet* to retrieve the related semantic information. Their experimental results show that the inclusion of such semantic information improves the clustering performance. Jing et al. [8] authors introduce a knowledge-based *VSM*, where the semantic information is used to re-weight term frequency in the original *VSM*. Semantic similarity is computed with two methods; one based on the structure of hierarchies such as *WordNet* and *Medical Subject Headings*, and another based on extracting the semantic relations from analyzing text corpora. Their experimental evaluation shows, like in the previous case, that the inclusion of the semantic similarity information between terms improves the clustering performance compared to the traditional *VSM* document representation. Finally, Hu et al. [24] enhanced the document representation with *Wikipedia* concept and category information. The idea text is based on mapping text documents to *Wikipedia* concepts, and from there to *Wikipedia* categories. The associated *Wikipedia* concept and category information is then utilized in the similarity measure of the documents.

Motivated by the encouraging results reported in the literature on the merits of incorporating semantic information as a background knowledge to perform text clustering, in this work we propose a semantic smoothing kernel which is based on our previous work [7], and we analyze, for the first time to the best of our knowledge, the application of three different semantic similarity measures using the suggested kernel to the task of document clustering. In addition, to address space and time efficiency issues, we expand the suggested kernel to incorporate information only regarding the top-k semantically related terms to the original term features, and we report on the effect of the chosen value of k. Overall, the semantic kernel (*S-VSM*) and its top-k extension (*top-k S-VSM*) that we propose in this paper may be applied using any measure of semantic similarity or relatedness. The embedding of the background knowledge is made through the suggested kernel, via smoothing the weights of the feature terms and their similarity values, and, thus, the original dimensions of the *VSM* are not augmented. In order to experimentally evaluate the suggested kernel and its extension we use three similarity measures; one based on *WordNet*, one based on *Wikipedia*, and a corpus-based measure to compute semantic similarity between terms.

3. Preliminaries

In document clustering, the input is typically a collection of documents, and the aim is to split it in subgroups of documents (clusters) that share similar concepts or discuss the same or similar topics. Each document d in the collection is made up of words and is represented using the vector space model (*VSM* or *Bag Of Words-BOW* representation) in a vector space where each distinct term constitutes a dimension of the collection. In the following we are summarizing the most important preliminary concepts in document representation, semantic relatedness or similarity and document clustering algorithms, that are used in the remaining of the paper. We will use the symbols n, m and c to denote the number of documents, the number of terms, and the number of clusters, respectively, and D to denote the set of documents in the collection.

3.1. Document Representations

3.1.1. The Vector Space Model VSM

The *VSM* defines for each term t_i a vector $\vec{t_i}$ in the vector space that represents it. It then considers the set of all term vectors $\vec{t_i}$ to be the orthocanonical base of the vector space, thus the space basis. In the standard *VSM* model, each document, d, is represented as follows:

$$\phi : d \mapsto \phi(d) = [w_{t_1,d}, w_{t_2,d}, \ldots, w_{t_m,d}]^T \in \Re^D$$

where $w_{t_i,d} = tf\text{-}idf(t_i, d)$ is the *TF-IDF* weight of term t_i in document d and the superscript T denotes the transpose operator. In the above expression, the function $\phi(d)$ represents the document d as a weighted term vector in the m-dimensional space of all terms in the collection. This function, however, can be any other mapping from a document to its vector space representation. To represent the whole corpus of the n documents in the collection, the term to document matrix, $\mathcal{D}$, is introduced. $\mathcal{D}$ is a $m \times n$ matrix whose rows are indexed by the words and whose columns are indexed by the documents.

One of the main issues in any document clustering algorithm is to define the similarity measure between documents. A popular measure for text similarity, is the computation of the cosine of the angle between the two document vectors in the *VSM* representation, which may be computed fast, it is scale invariant, and does not depend on the text length. However, cosine similarity is bounded by the limitations of *VSM*, so it assumes pairwise orthogonality between terms, i.e., the vector space dimensions, and overlooks the semantic relatedness or similarity between terms. The cosine similarity measure between two documents in the *VSM* representation is given by Equation 1.

$$sim_{VSM} = cosine(d_p, d_q) = \frac{\sum_{i=1}^{m} w_{t_i,d_p} w_{t_i,d_q}}{\sqrt{\sum_{i=1}^{m} w_{t_i,d_p}^2}\sqrt{\sum_{i=1}^{m} w_{t_i,d_q}^2}} \qquad (1)$$

where d_p and d_q are two documents in the collection, and w_{t_i,d_p} denotes the weight of term t_i in document d_p (w_{t_i,d_p} respectively denotes the weight of t_i in d_q).

*3.1.2. The Generalized Vector Space Model (*GVSM*)*

The *Generalized Vector Space Model* (*GVSM*) [25], also known as 'the dual space' approach [26], extends *VSM*, by introducing term-to-term correlations, which deprecate the pairwise orthogonality assumption, but it keeps the assumption that the term vectors are linearly independent[8]. *GVSM* considers a new space, where the original m-dimensional term vector is expressed as a linear combination of vectors in the 2^n dimensional space. The similarity between two documents d_p and d_q, can be then defined as:

$$sim_{GVSM}(d_p, d_q) = \frac{\sum_{j=1}^{m} \sum_{i=1}^{m} w_{t_i,d_p} * w_{t_j,d_q} * (\vec{t_i} \cdot \vec{t_j})}{\sqrt{\sum_{i=1}^{m} w_{t_i,d_p}^2} \sqrt{\sum_{i=1}^{m} w_{t_i,d_q}^2}} \tag{2}$$

Term correlation $(\vec{t_i} \cdot \vec{t_j})$ can be implemented in several ways. It can be based on frequency co-occurrence statistics gathered from large corpora [25], or on the semantic correlations between terms [27]. Using large corpora to compute the term-to-term correlations, makes the assumption that two terms are considered semantically related if they frequently co-occur in the same documents. Thus, a document is represented by the embedding shown in Equation 3.

$$\bar{\phi}(d) = \phi(d)\mathcal{D} \tag{3}$$

The corresponding *GVSM* kernel between two documents d_p and d_q, is shown in Equation 4.

$$\bar{\kappa}(d_p, d_q) = \phi(d_p)\mathcal{D}\mathcal{D}^T\phi(d_q)^T \tag{4}$$

In Equation 4, the $(i,j)^{th}$ entry of the matrix $\mathcal{D}\mathcal{D}^T$ is given by the following formula, given that the term-document matrix $\mathcal{D}$ contains the *TF-IDF* weights for each of the terms, in each of the documents:

[8]It is known from linear algebra that if every pair of vectors in a set of vectors is orthogonal, then this set of vectors is linearly independent, but not the inverse.

$$(\mathcal{D}\mathcal{D}^T)_{ij} = \sum_{ij} \mathit{tfidf}(t_i, d) \cdot \mathit{tfidf}(t_j, d) \tag{5}$$

The matrix $\mathcal{D}\mathcal{D}^T$ has a nonzero entry $(\mathcal{D}\mathcal{D}^T)_{ij}$ if there is a document d in which the corresponding terms t_i and t_j co-occur.

3.2. Relatedness measures

The problem of computing semantic relatedness or similarity between two terms has been studied for many decades in the bibliography, and there are many approaches that may be categorized according to their design principles into: (i) knowledge-based, (ii) corpus-based, and, (iii) hybrid. Approaches of the first category use a word thesaurus or an ontology and its structure to compute the semantic similarity or relatedness between two terms. The second category approaches analyze statistically large corpora to infer latent similarities between terms, typically based on their co-occurrence in the analyzed corpus. Finally, approaches of the latter category combine typically the structure of an ontology and statistical information gathered for the concepts and their labels from the analysis of a large text corpus. For an overview of the most representative approaches in each category, the reader may wish to consult the work of Budanitsky and Hirst [9], or the more recent work of Zhang et al. [10]. In the following subsections we focus on the three measures that we are employing for the implementation of the suggested kernel.

3.2.1. Omiotis

The first measure that we use, is the *Omiotis* measure [1]. *Omiotis* defines the semantic relatedness between a pair of terms as shown in Equation 6, where the knowledge-base O is *WordNet* (*WN*).

$$SR_{Omiotis}(t_i, t_j) = \max_m \{\max_k \{SCM(S^m_{ij}, P^k_{ij}) \cdot SPE(S^m_{ij}, P^k_{ij})\}\} \tag{6}$$

where *SCM* and *SPE* are called *Semantic Compactness* and *Semantic Path Elaboration* respectively. Their product measures the weight of the path connecting the senses that can be assigned to the terms (S^m_{ij}), taking into account: the path length considering all possible paths connecting them (P^k_{ij}), the type of the semantic edges comprising it, and the depth of the intermediate nodes in the *WN* senses hierarchy. The semantic relatedness between two terms t_i, t_j, when $t_i \in WN$ and $t_j \notin WN$, or vice versa, is considered 0. The intuition behind Equation 6 is that the semantic relatedness between two terms should be computed based on the *highest value* path connecting any pair of senses of the two terms. The computation of the *value* takes into account in tandem all of the aforementioned factors.

3.2.2. Wikipedia-based relatedness

The *WLM Wikipedia*-based measure of Milne and Witten [2], is a low-cost solution for measuring relatedness between terms using the *Wikipedia* articles and link structure as a knowledge base. The semantic relatedness between two terms t_i and t_j according to *WLM* is defined as shown in Equation 7. The intuition behind this formula is that the semantic similarity between two terms becomes higher, as the number of articles pointing to both respective *Wikipedia* articles increases (i.e., as the percentage of the articles linking to both pages compared to the number of articles linking to either of them increases).

$$SR_{Wiki}(t_i,t_j) = \frac{log(\max\{|In(a_i)|,|In(a_j)|\}) - log(|In(a_i) \cap In(a_j)|)}{log(|W|) - log(\min\{|In(a_i)|,|In(a_j)|\})} \qquad (7)$$

where $In(a_i)$ is the number of *Wikipedia* articles that point to article a_i which corresponds to term t_i (similarly for $In(a_j)$) and $|W|$ is the number of *Wikipedia* articles.

3.2.3. Average of Omiotis and Wikipedia-based relatedness

Given equations 6 and 7, we can combine their values into a single relatedness value that considers them both by simply averaging the two scores. This average *Omiotis-Wikipedia* relatedness score, is shown in the following equation for two terms t_i and t_j:

$$SR_{OWavg}(t_i,t_j) = \frac{SR_{Omiotis}(t_i,t_j) + SR_{Wiki}(t_i,t_j)}{2} \qquad (8)$$

3.2.4. Pointwise Mutual Information

Pointwise Mutual Information (*PMI*) is a statistical measure of association, which has been used, for example, to discover collocations by examining how often words co-occur in a corpus [28, 29]. In our case, *PMI* is used to measure the relatedness of two words w_1, w_2 by examining how often these two words co-occur in sentences of a corpus. The intuition is that if w_1 and w_2 are related, the words will co-occur frequently. In the experiments, we used the *Tipster* corpus [9], which contains approximately 953 million tokens

[9]http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC93T3A

to estimate $PMI(w_1, w_2)$.

The following definition of *PMI* can be used for our purposes, where $P(w)$ is the probability of encountering a sentence containing a word occurrence w, and $P(w_1, w_2)$ is the probability of encountering a sentence with (at least) two word occurrences (not necessarily adjacent).

$$PMI(w_1, w_2) = \log \frac{P(w_1, w_2)}{P(w_1) \cdot P(w_2)} \tag{9}$$

If the occurrences of w_1 and w_2 in the corpus are completely independent, then $PMI(w_1, w_2) = 0$. If w_1 and w_2 always occur together, their *PMI* score is maximum, equal to $-\log P(w_1) = -\log P(w_2)$. If w_1 and w_2 never occur together, then their *PMI* score approximates $-\infty$.

More precisely, we use the following normalized form of *PMI* [30], which returns values in the range $(0, 1]$.

$$PMI(w_1, w_2) = \frac{1}{2} \cdot (\frac{1}{-\log P(w_1, w_2)} \cdot \log \frac{P(w_1, w_2)}{P(w_1) \cdot P(w_2)} + 1) \tag{10}$$

We note that Pecina [29] found *PMI* to be the best collocation extraction measure; and Newman et al. [31] found it to be the best measure of '*topical coherence*' for sets of words.

3.3. Document Clustering

Document clustering algorithms can be categorized into two classes: (i) partitioning algorithms, and, (ii) hierarchical algorithms. Partitioning algorithms divide the dataset into a number of clusters which are usually optimal in terms of some predefined criterion functions (in Section 3.3.3 representative functions are analyzed). Hierarchical algorithms group the data points into a hierarchical tree structure using bottom-up or top-down approaches. Agglomerative (bottom-up) approaches initially consider each data point as a single cluster and in each iteration build bigger clusters by grouping similar data points or clusters together until the entire dataset is merged in a single cluster in the top level of the hierarchy. Divisive (top-down) approaches split the dataset into smaller clusters until each data point is assigned a single cluster. Spectral clustering algorithms attempt to solve the clustering problem as an optimal partitioning problem.

3.3.1. Clustering algorithms

The *K-means* [32] is the most known partitioning clustering algorithm. It takes as input the dataset D and the desired number of clusters c. It initially randomly selects c documents from D as the cluster representatives (centroids) and then assigns each document to the most similar centroid. It iteratively recalculates centroids based on the clusters' documents and reassigns documents to the most similar centroid. The clustering iterations stop when the condition defined by the criterion function is met, or when the maximum number of iterations is exceeded.

The *Repeated Bisection* (*RB*) algorithm [33] obtains a c-way clustering solution by first bisecting the entire collection. Next, one of the two clusters is selected and it is further bisected leading to a total of three clusters. The process of selecting and bisecting a particular cluster is repeated until a number of c clusters is obtained. The aim of each bisection is to optimize a particular criterion function.

The *Biased Agglomerative* (*Bagglo*) algorithm [33] builds on the agglomerative paradigm, whose goal is to locally optimize (minimize or maximize) a particular clustering criterion function by merging two clusters. The merging process is repeated, as previously, until a number of c clusters is obtained.

Non-negative Matrix Factorization (*NMF*) [34, 35, 36] is a spectral clustering algorithm which finds an optimal partitioning of the document set by finding two nonnegative matrices whose product can well approximate the initial non-negative data matrix. The method assumes that D consists of c clusters and aims in factorizing the term-document matrix $\mathcal{D}$ into the non-negative $m \times c$ matrix W and the non-negative $c \times n$ matrix H that minimizes an objective function. W can be regarded as containing a basis that is optimized for the linear approximation of the data in $\mathcal{D}$.

3.3.2. Algorithms Complexity

The various clustering algorithms have different scalability characteristics. Table 1 summarizes the time and space complexity of the discussed clustering algorithms[10]. With respect to time and memory, the most scalable method is the repeated-bisecting algorithm [37]. The least scalable of the algorithms are the ones based on hierarchical agglomerative clustering.

[10]For reference in the algorithm complexities the reader may wish to consult the manual of *Cluto* clustering toolkit [37]

Table 1: Time and space complexity of the clustering algorithms. *NNZ* represents total number of non-zero values in the term-to-document matrix

Algorithm	Time Complexity	Space Complexity
$DirectMethod$	$O(c(NNZ + m))$	$O(NNZ + m * c)$
$RBMethod$	$O(NNZ * log(c)$	$O(NNZ)$
$Bagglo$	$O(n^2 * log(n))$	$O(n^2)$
NMF	$O(c * m * n)$	$O(m * c)$

3.3.3. Clustering Criterion Functions

Clustering algorithms use a global criterion function whose optimization controls the whole clustering process. Therefore, clustering algorithms typically compute a clustering solution for which the value of a particular criterion function is optimized. In this work we use the following three different clustering criterion functions:

1. $\mathcal{I}_2$ Criterion function: In this approach each cluster (S_r), is represented by its centroid (C_r) and the $\mathcal{I}_2$ criterion function (Eq.11) maximizes the similarity between each document and the centroid of the cluster that is assigned to.

$$\mathcal{I}_2 \text{ maximizes } \sum_{r=1}^{c} \sum_{d_i \epsilon S_r} cos(d_i, C_r) = \sum_{r=1}^{c} \parallel D_r \parallel \tag{11}$$

The term $\parallel D_r \parallel$ is square-root of the pairwise similarity between all documents in S_r.

2. $\mathcal{H}_2$ criterion function is the combination of $\mathcal{I}_2$(Eq.11) and $\mathcal{E}_1$(Eq.12).

$$\mathcal{E}_1 \text{ minimizes } \sum_{r=1}^{c} n_r cos(C_r, C) \Leftrightarrow \text{ minimize } \sum_{r=1}^{c} n_r \frac{D_r^t D}{\parallel D_r \parallel} \tag{12}$$

where $D_r^t D$ is sum of all document vectors in the corresponding cluster r and n_r is the size of corresponding cluster r.
The $\mathcal{E}_1$ criterion function (Eq. 12) computes the clustering by finding a solution that separates the documents of each cluster from the entire collection. so $\mathcal{H}_2$ would be:

$$\mathcal{H}_2 \text{ maximizes } \frac{\mathcal{I}_2}{\mathcal{I}_2} \Leftrightarrow \text{ minimize } \frac{\sum_{r=1}^{c} \parallel D_r \parallel}{\sum_{r=1}^{c} n_r D_r^t D / \parallel D_r \parallel} \tag{13}$$

3. $\mathcal{EUC}$ Criterion function: Given a nonnegative matrix T, nonnegative matrix factorization (NMF) aims to factor T into two nonnegative matrices W and H. Lee and Seung [35] formulated *NMF* as two types of optimization problems in which the approximation error should be minimized under the constraint that W and H are nonnegative. In the case of the Euclidean distance, the optimization problem is expressed as follows
$$\text{minimize } f(W,H) = \| T - WH\|^2 \tag{14}$$
subject to $H_{aj} \geq 0, W_{ia} \geq 0, \forall_{a,i,j}$
where $\| \cdot \|$ represents the Frobenius norm, that is,
$\| T - WH\|^2 = \sum_{ij}(T_{ij} - (WH)_{ij})^2$
as the objective function $f(W,H)$ is non-convex. Therefore, Lee and Seung [35] proposed the multiplicative update rule for finding a local optimal solution.

4. Semantic Smoothing Kernels

The Vector Space Model (*VSM*) represents a document collection by a term-to-document matrix, thus, resulting in a very sparse representation in a high dimensional space. Among its deficiencies is that it assumes term independence and thus fails to capture semantics and that its sparse representation is susceptible to noise [25]. The *Generalized Vector Space Model* (*GVSM*) improves *VSM* by removing the pairwise orthogonality assumption and taking into account the term-to-term correlations, which are based either on the semantic correlations between terms or on the frequency co-occurrence statistics computed in large text corpora. *GVSM* results in a denser representation model. *Latent Semantic Indexing* (*LSI*) attempts to improve *GVSM* by means of dimensionality reduction using a new space with fewer orthogonal dimensions (principal dimensions), which are computed on a training corpus and convey most of the variance of the observed data. This work focuses on *GVSM*; it uses a statistics based kernel as a baseline, evaluates a semantics *GVSM* kernel *S-VSM* that can use any semantic similarity measure and, finally, introduces a space efficient *GVSM* (*top-k S-VSM*) kernel in order to confront the high dimensionality problem.

4.1. A GVSM*-based Semantic kernel S-VSM*

In order to overcome the limitations of *VSM* in taking into account the semantic similarity of terms, we enrich the *BOW* representation with semantic information. Instead of extending it with additional dimensions that correspond to concepts, we embed the conceptual relationships of terms in the *VSM* document representation. More specifically, we re-adjust the weight values of each term in a document using the weights of its semantically related terms according to their pairwise similarities. This transformation maintains the original *VSM* dimensions, and, thus, does not increase the computational space complexity of the similarity function, and discounts the effects of very general terms in favor of conceptually related terms.

To enrich the BOW representation with semantic information, we construct the semantic relatedness matrix R using a semantic relatedness measure. Specifically, the $r_{i,j}$ element of matrix R is given by a knowledge or corpus-based similarity measure[11], which quantifies the semantic relatedness between terms $T : (t_i, t_j)$. Thus, R is a $D \times D$ symmetric matrix with 1's in the principal diagonal. This smoothing matrix can be used to transform the documents' vectors in such a way that semantically related documents are brought closer together in the transformed (or feature) space (and vice versa). More formally, the semantically enriched BOW representation of a document d is given as

$$\bar{\phi}(d) = (\phi(d)^T R)^T$$

Following the notation introduced in Section 3.1.1 we define the feature space implicitly via the kernel function. This is particularly important in kernel-based methods or kernel machines when the feature space is very large or even infinite in size. By definition, the kernel function computes the inner product between documents d_p and d_q in the feature space. For our case, this can be written as

$$\kappa(d_p, d_q) = \bar{\phi}(d_p)^T \bar{\phi}(d_q) = \phi(d_p)^T R R^T \phi(d_q) \tag{15}$$

For this to be a valid kernel function, the Gram matrix G (where $G_{pq} = \kappa(d_p, d_q)$) formed from the kernel function must satisfy the Mercer's conditions [38]. These conditions are satisfied when the Gram matrix is positive semi-definite. It has been shown in [38] that the matrix G formed by the kernel function (Eq. 15) with the outer matrix product RR^T is indeed a positive semi-definite matrix.

[11]in our case we employ *Omiotis*, *Wikipedia*-based or *PMI* similarities; the three measures that were discussed in the preliminaries section

After semantic smoothing, the new weight value of a term t_i in document d_p is calculated as shown in Equation 16.

$$\hat{w}_{t_i,d_p} = w_{t_i,d_p} + \sum_{j=1,j\neq i}^{m} \Psi_{t_i t_j}.w_{t_j,d_p} \tag{16}$$

where w_{t_i,d_p} is the original weight for term t_i in d_p, w_{t_j,d_p} are the original weights for all other terms and $\Psi_{t_i t_j}$ are the relatedness value between terms.

4.2. Top-k S-VSM

The semantic smoothing presented in Equation 16 corrects the weight of t_i by taking into account the relatedness of t_i with all the terms in the collection. It is straightforward to think that highly related terms will affect the weight of t_i more than terms with limited semantic relation or no relation at all. Though this is taken into account by considering lower weights for less semantically related terms, i.e., embedded as a property directly semantic relatedness is computed, there might be still many weakly semantically related terms that affect the overall re-weighting of the semantic smoothing.

In order to address this problem, in the following we extend the notion of the *S-VSM* semantic kernel that was presented in the previous section, so that it takes into account the background knowledge of the semantic similarities between terms that are highly related. The kernel retains only the top-k highly semantically related terms (*top-k S-VSM*) to a given dictionary term t_i. The basic intuition behind the following formulation is that each term t_i should be reweighed considering only terms with which t_i shares many highly semantically related neighbors. A similar formulation of this intuition was created by Bloehdorn et al. [39], with the difference being that they considered only weighted parents (hypernyms) of the terms, in an effort to expand the approach of Mavroeidis et al. [40] who considered unweighed parents (hypernyms). In the following formulation we consider any semantically related term, in contrast to the previous works that considered only hypernyms, and retain only the top semantically related terms for each term.

We define a virtual document collection V, where each document $d_i \in V$ has been created for every term $t_i \in T$. The document-to-term matrix $\boldsymbol{V}$ is, hence, a square matrix nxn, where $n = |T| = |V|$. Given the initialization of a positive integer parameter k, where $k \leq n$, in every cell of row i in $\boldsymbol{V}$ we add V_{ij} =*Omi*(t_i,t_j), iff t_j belongs to the set of the top k semantically related terms of t_i, or we add $V_{ij} = 0$ in any other case[12].

Finally, for two documents d_p and d_q from the original collection D we apply the *GVSM* semantic kernel, given $\boldsymbol{V}$, as shown in the next equation:

$$k(d_p, d_q) = \sum_{i,j} \phi(d_p)_i V'V \phi(d_q)_j \tag{17}$$

where

$$(V'V)_{ij} = \sum_{d=1}^{n} V_{id} V_{jd} \tag{18}$$

In the experimental evaluation, we present an analysis of how the selection of k, when using the *top-k S-VSM*, affects the clustering performance. The rationale behind its design is that the selection of few, the top-k most related terms per dimension, is definitely computationally faster than the *top-k S-VSM*, and the experimental evaluation actually shows that the clustering performance is not harmed, compared to the performance of the *S-VSM*, even when a small k is selected as a value.

5. Experimental Setup

To evaluate the effectiveness of *S-VSM* and its *top-k* extension for text clustering we conduct a series of experiments in benchmark text clustering datasets. We employ seven clustering algorithms (four partitioning, two hierarchical, and the non-negative matrix factorization), which we execute with and without semantic enrichment of the *VSM*, and compare the clustering performance in both cases over five datasets. This section presents our experimental setup; the next section analyzes the results of our experiments.

5.1. Data sets

We used six real data sets, namely *Reuters-Transcribed-set*, *R8 Reuters-21578*, *ReutersTD*, *WebKB*, *Email* and *Movie Reviews* to evaluate the clustering performance of the aforementioned approaches. In the following, a description of each data set is provided.

[12]We can also normalize V per row and column, so that the sum of each row adds to 1; the same with the sum of each column

1. *Movie Reviews* (*Movies*): It contains $2,000$ reviews of movies from the *Internet Movie Database Archive*. Half of them are positive sentiments about the movie, and half express a negative opinion. For the purpose of text clustering, we concentrated in discriminating between positive and negative ratings.
2. *Email*: This dataset comprises e-mails classified into three classes; *conference*, *job*, and *spam*.
3. *WebKB*: The *WebKB* dataset comprises web pages collected by the *World Wide Knowledge Base* project of the *CMU* text learning group. These pages are manually classified into seven different classes: *student*, *faculty*, *staff*, *department*, *course*, *project*, and *other*. We discarded the classes *staff* and *department* because there are only a few pages from each university. We also discarded the class *other* because pages were very different in content within this class.
4. *Reuters-Transcribed-set* (*RTS*): The dataset is created by selecting 20 files from each of the 10 largest classes in the *Reuters-21578* collection. It is available from the *UCI KDD Archive*.
5. *R8 Reuters-21578*: It contains $7,674$ documents from the *Reuters-21578* collection. We are following Sebastiani's convention to selection 8 of the 10 most frequent classes.

6. *ReutersTD*: It contains $1,504$ documents that belong to 13 out of the 135 topic categories *Reuters-21578* collection. The same dataset has been employed by [8] and we use it only in order to compare against their knowledge-based *VSM* model.

Table 2 summarizes the description of the five used datasets for the evaluation and comparison of the clustering performance.

5.2. Document Clustering

5.2.1. Algorithms

We use seven different algorithms to test whether the clustering performance could be improved by using the *S-VSM* or its *top-k* extension. The selection of such a number of clustering algorithms is imperative, to ensure that the reported results are not dependent on any particular clustering algorithm. Four of the seven algorithms are four *K-means* variants: (1) *Direct K-means* with criterion function I_2 $(d - i_2)$, (2) *Direct K-means* with criterion function H_2 $(d - h_2)$, (3) *Repeated Bisectioning K-means* with criterion

Table 2: Summary of data sets

Data	Source	No. of Docs.	No. of Terms	No. of Classes
$Movies$	*abc*	2000	29433	2
$Email$	$Cogsys$	1443	10429	3
$WebKB$	*abc*	4199	9457	4
RTS	UCI	200	4699	10
$R8$	UCI	7674	7852	8
$ReutersTD$	UCI[8]	1504	7275	13

function I_2 $(rb - i_2)$, and, (4) *Repeated Bisectioning K-means* with criterion function h_2 $(rb - h_2)$. These four implementations are provided from the *CLUTO Toolkit*[13], and their selection was based on their very good performance reported in the literature [6]. The fifth clustering algorithm is the *RBR* with criterion function H_2 $(rbr - h_2)$. Its implementation can also be found in the *CLUTO* toolkit. The algorithm is similar to the repeated-bisecting method but the overall solution is globally optimized to further satisfy best the clustering criterion function. The sixth algorithm is a hybrid hierarchical agglomerative clustering algorithm, which can also be found in *CLUTO*. The algorithm is called *Biased Agglomerative* (*Bagglo*), with criterion function H_2 $(b - h_2)$. For clustering n objects, *Bagglo* first computes a $\sqrt{n}$-way clustering solution using repeated bisections of individual objects or clusers of objects (*rb* method). Consequently, it augments the original feature space by adding $\sqrt{n}$ new dimensions. Each object has a non-zero value in the dimension that corresponds to its cluster, which is proportional to the similarity between the object and its cluster centroid. Then, given this augmented representation, the overall clustering solution is obtained by using the traditional agglomerative algorithm. It has been shown in the past that the *Bagglo* algorithm always produces better results than agglomerative clustering algorithms [41]. The seventh algorithm employs *Non-negative Matrix Factorization* (*NMF*) and the implementation for text clustering provided by *DTU:Toolbox*[14].

[13]Available for download from `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download`

[14]Available for download from `http://cogsys.imm.dtu.dk/toolbox/nmf`

5.2.2. Vector Similarity

Cosine measure is the commonly used method to find the similarity between two text documents $d_i, d_j \epsilon D$. It is measured by the cosine of the angle between the vectors $\vec{t_i}, \vec{t_j}$ representing them:

$$cos(\sphericalangle(\vec{t_i}, \vec{t_j})) = \frac{\vec{t_i}.\vec{t_j}}{\parallel \vec{t_i} \parallel . \parallel \vec{t_j} \parallel}$$

If the documents are identical then cosine measure is one, else it would be zero if there is nothing in common between them (i.e., the vectors are orthogonal to each other). For all datasets, we used a stopwords list to remove common words (non descriptive). After that, text documents are processed by the process of lemmatization. The reason behind using lemmatization is to increase the coverage. Tree Tagger[15] is used for lemmatization process. Each method is run 5 times, every time starting with a random initialization of the clusters, and results are reported as average $\pm$ standard deviation of the performance measure.

5.2.3. Evaluation Measures

The evaluation of the performance of clustering algorithms is done using external criteria (the labels are known for all documents). So, for our experiments, we set the number of clusters equal to the number of categories (labels, usually built using human assessors) for all the clustering algorithms. To evaluate their performance, we compare the clusters generated by these algorithms with the categories by computing BCubed FMeasure[42]. Amigo et al. [43] showed that BCubed measure is the only measure that satisfy all four desirable constraints(cluster homogeneity, cluster completeness, rag bag, and cluster size versus quantity) for a good measure for cluster evaluation. Let $L(i)$ and $C(i)$ be the category and the cluster of an item i, then correctness of the relation between i and i' that share a category is if and only if they are in the same cluster:

$$Correctness(i, i') = \begin{cases} 1, & \textit{iff } L(i) = L(i') \leftrightarrow C(i) = C(i') \\ 0, & \text{otherwise} \end{cases}$$

[15]Available for download from `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`

So BCubed precision and recall of an item i is defined as:

$$BCubedPrecision_i = \frac{Number\ of\ correct\ elements\ in\ output\ clsuter\ containing\ i}{Number\ of\ elements\ in\ the\ output\ cluster\ containing\ i}$$

$$BCubedRecall_i = \frac{Number\ of\ correct\ elements\ in\ output\ clsuter\ containing\ i}{Number\ of\ elements\ in\ the\ category\ containing\ i}$$

So the overall BCubed precision is the averaged precision of all items in the distribution, and BCubed recall is the averaged recall of all items in the distribution. To obtain a single evaluation measure, BCubed presision and recall are combined using the harmonic mean formula:

$$BF = 2 \times \frac{BP \times BR}{BP + BR}$$

The BCubed F-Measure ranges from 0 to 1 with larger values signifying better clusterings.

6. Experimental Results and Discussion

6.1. Performance of the Semantic VSM on Different Datasets

Our first set of experiments was focused on evaluating *S-VSM* representations on various clustering algorithms and datasets. The BCubed results for *VSM* and semantic *VSM* are shown in Table 3, where each row corresponds to one method and each column corresponds to one representation for a particular dataset. Here results are provided primarily for completeness and in order to evaluate the various methods. The first way of summarizing the results is to average the BCubed FMeasure for each representation over the five different datasets. A number of observations can be made by analyzing the results in Table 4. First, *GVSM* and *S-VSM* methods outperform the *VSM* for almost all datasets. Over the entire set of experiments, *GVSM* is 1.5%-12% and *S-VSM* is 3%-6% better than *VSM*. *GVSM* full and OW are very competitive and lead to the best solutions for most of the datasets. GVSM performed well on Movies and WebKB data but OW outperform *GVSM* on R8 and RTS data. PMI also performed better than *VSM* in most of the datasets. PMI performed better than OMI and Wiki where sentences are not more structured. When the relative performance of different methods is similar, the average BCubed Fmeasure will be quite similar. Hence to make

Table 3: Text clustering Performance (BCubed Fmeasure)

Dataset	Clu. Algo.	*VSM*	$GVSM$	*S-VSM*			
		TFIDF	full	Omi	Wiki	PMI	OW
RTS	$D-i_2$	0.319±0.02	0.376 ±0.02	0.378±0.02	0.358±0.02	0.369±0.04	0.378±0.01
	$D-h_2$	0.318±0.02	0.371 ±0.01	0.362±0.02	0.329±0.03	0.357±0.02	0.364±0.01
	$RB-i_2$	0.326±0.02	0.323 ±0.02	0.354±0.02	0.353±0.02	0.321±0.04	0.350±0.02
	$RB-h_2$	0.306±0.02	0.339 ±0.01	0.333±0.02	0.342±0.00	0.289±0.02	0.340±0.02
	$RBR-h_2$	0.347±0.02	0.345 ±0.01	0.365±0.02	0.369±0.03	0.323±0.01	0.363±0.02
	$Bagglo-h_2$	0.358±0.00	0.363 ±0.00	0.390±0.00	0.353±0.00	0.372±0.00	0.378±0.00
	NMF	0.341±0.00	0.365 ±0.01	0.308±0.02	0.342±0.00	0.241±0.00	0.352±0.00
Movies	$D-i_2$	0.526±0.01	0.659 ±0.01	0.555±0.00	0.542±0.01	0.565±0.05	0.542±0.01
	$D-h_2$	0.521±0.01	0.659 ±0.01	0.549±0.01	0.528±0.01	0.558±0.05	0.537±0.00
	$RB-i_2$	0.526±0.01	0.659 ±0.01	0.555±0.02	0.541±0.01	0.565±0.05	0.542±0.01
	$RB-h_2$	0.524±0.01	0.659 ±0.01	0.549±0.02	0.528±0.01	0.558±0.05	0.537±0.00
	$RBR-h_2$	0.518±0.01	0.695 ±0.01	0.546±0.02	0.529±0.01	0.560±0.05	0.537±0.01
	$Bagglo-h_2$	0.508±0.00	0.538 ±0.00	0.503±0.00	0.502±0.00	0.579±0.00	0.542±0.00
	NMF	0.509±0.04	0.609 ±0.02	0.551±0.01	0.569±0.00	0.532±0.00	0.515±0.00
WebKB	$D-i_2$	0.294±0.01	0.401 ±0.01	0.370±0.01	0.371±0.02	0.331±0.00	0.333±0.01
	$D-h_2$	0.284±0.01	0.389 ±0.01	0.344±0.01	0.356±0.02	0.330±0.00	0.304±0.01
	$RB-i_2$	0.304±0.01	0.415 ±0.01	0.325±0.02	0.395±0.02	0.329±0.00	0.344±0.01
	$RB-h_2$	0.297±0.01	0.393 ±0.02	0.342±0.02	0.355±0.01	0.330±0.00	0.319±0.01
	$RBR-h_2$	0.288±0.01	0.390 ±0.02	0.360±0.01	0.349±0.01	0.332±0.00	0.310±0.01
	$Bagglo-h_2$	0.294±0.00	0.338 ±0.00	0.315±0.00	0.296±0.00	0.392±0.00	0.291±0.00
	NMF	0.341±0.01	0.357 ±0.04	0.372±0.01	0.410±0.02	0.336±0.01	0.348±0.01
Email	$D-i_2$	0.734±0.01	0.712 ±0.05	0.723±0.01	0.741±0.01	0.746±0.03	0.723±0.01
	$D-h_2$	0.717±0.01	0.701 ±0.05	0.736±0.01	0.702±0.01	0.721±0.04	0.716±0.01
	$RB-i_2$	0.747±0.01	0.791 ±0.01	0.774±0.01	0.738±0.01	0.699±0.01	0.737±0.02
	$RB-h_2$	0.721±0.01	0.786 ±0.01	0.768±0.01	0.714±0.01	0.715±0.01	0.738±0.01
	$RBR-h_2$	0.717±0.01	0.784 ±0.01	0.744±0.01	0.719±0.01	0.713±0.01	0.728±0.01
	$Bagglo-h_2$	0.589±0.00	0.678 ±0.00	0.746±0.00	0.641±0.00	0.544±0.00	0.589±0.00
	NMF	0.747±0.02	0.753 ±0.01	0.773±0.01	0.700±0.01	0.695±0.01	0.741±0.01
R8	$D-i_2$	0.519±0.02	0.559 ±0.04	0.536±0.02	0.551±0.02	0.557±0.02	0.671±0.01
	$D-h_2$	0.516±0.02	0.561 ±0.04	0.540±0.02	0.539±0.03	0.553±0.02	0.552±0.02
	$RB-i_2$	0.550±0.04	0.486 ±0.02	0.581±0.03	0.595±0.03	0.608±0.03	0.614±0.01
	$RB-h_2$	0.517±0.04	0.473 ±0.01	0.532±0.03	0.536±0.03	0.533±0.03	0.533±0.01
	$RBR-h_2$	0.515±0.03	0.481 ±0.01	0.530±0.03	0.551±0.02	0.523±0.02	0.547±0.02
	$Bagglo-h_2$	0.439±0.00	0.463 ±0.00	0.528±0.00	0.455±0.00	0.520±0.00	0.487±0.00
	NMF	0.512±0.02	0.574 ±0.01	0.412±0.03	0.567±0.02	0.408±0.00	0.546±0.01

the comparisons of these representations easier, our second way of summarizing the results is to create a dominance matrix for various representations. We create a 6×6 dominance matrix (Table 5). The rows and columns of this matrix correspond to the various vector space models, whereas its values correspond to the number of dataset-algorithm pair for which the representation model of the row outperforms the model of the column. For example, the value in the entry of the row TFIDF and the column $GVSM$is 7, which means for 7 out of the 35 dataset-algorithm pairs, the TFIDF model outperforms the OMI based *VSM*. The values that are close to 35 indicate that the row method outperforms the column method. OMI outperforms the TFIDF as it performed 31 out of 35 times better than the *VSM*. Similarly Wiki and PMI were also 29 and 24 times better than *VSM* respectively. *GVSM* and *S-VSM* are very competitive. In most of the cases results of *S-VSM*are better than *GVSM* results. Within *S-VSM*, OMI and Omi-Wiki based methods are overall better than other representations.

Table 4: Average BCubed Fmeasure for various Vector Space Models on Different Datasets

Representation Dataset	*VSM*TFIDF	*GVSM*full	Omi	Wiki	PMI	OW_{avg}
RTS	0.330	0.355	0.355	0.349	0.325	0.361
$Movies$	0.519	0.640	0.544	0.534	0.560	0.536
$WebKB$	0.300	0.383	0.347	0.362	0.340	0.321
$Email$	0.710	0.744	0.752	0.708	0.690	0.710
$R8$	0.523	0.512	0.544	.554	0.555	0.583

Table 5: Dominance Matrix for various Vector Space Models evaluated by BCubed FMeasure

Representation	*VSM*TFIDF	*GVSM*full	Omi	Wiki	PMI	OW_{avg}
$VSMTFIDF$	0	7	4	6	11	5
$GVSMfull$	28	0	22	26	26	22
Omi	31	13	0	20	22	22
$Wiki$	29	9	15	0	18	17
PMI	24	9	13	17	0	15
OW_{avg}	29	13	11	17	19	0

In order to compare our $S-VSM$ model against other knowledge-based *VSM*s, we use the *ReutersTD* dataset, employed in the work by Jing et al. [8]. We compare against the methods reported in [8] using the same evaluation metrics for clustering, i.e. purity and entropy. The results of our *S-VSM* model using the four different term similarity metrics (Omi, $Wiki$, PMI, OW) and the first two clustering alternatives ($D-i_2$ and $D-h_2$) are reported in Table 6 along with the results of the methods reported in [8]. Lower entropy and higher purity values correspond to better clustering. As shown in Table 6 our *S-VSM* compares favorably against the method of Jing et al. and with the use of Omiotis similarity metric our best results outperform the best results of Jing et al.

Table 6: Text clustering performance of Knowledge based VSMs (*ReutersTD* dataset). Lower entropy and higher purity values in the table designate better clustering performance.

	method	*Entropy*	*Purity*
Jing et al [8]	$Term$	0.3789	0.6418
	$Term + concept$	0.3358	**0.7092**
	$Term + HS$	0.3463	0.6851
	$Term + SO$	0.3247	0.7013
	$Term + AL$	**0.3225**	0.6932
S-VSM	$Omi + D - i_2$	**0.304**	0.713
	$Omi + D - h_2$	0.309	**0.716**
	$Wiki + D - i_2$	0.335	0.698
	$Wiki + D - h_2$	0.341	0.661
	$PMI + D - i_2$	0.459	0.583
	$PMI + D - h_2$	0.471	0.602
	$OW + D - i_2$	0.331	0.697
	$OW + D - h_2$	0.336	0.701

6.2. Performance of Clustering Algorithms with Different Representations

To summarize the results for clustering algorithms we create a 7×7 dominance matrix that is shown in Table 7. The rows and columns of this matrix correspond to the various clustering algorithms whereas its values correspond to the number of dataset-VSM pair for which the clustering algorithm of the column outperforms the model of the column. The values that are close to 30 indicate that the row method outperforms the column method. For example, the value in the entry of the row Di_2 and the column Dh_2 is 26, which means that for 26 out of the 30 dataset-VSM pairs, the Di_2 outperforms the Dh_2 algorithm. Results validated the pervious findings that variants of K-means perform better than other algorithms. Di_2 performed 26 and 21 times better than Dh_2 and Bagglo respectively. Similarly Di_2 outclass 25 and 22 times RBh_2 and $RBRh_2$ respectively. NMF is better than Bagglo in most of the cases. Overall selection criterion i_2 is better than all other criterions.

6.3. Performance of the GVSMsemantic kernel using the Top-k% related terms

Based on their performance among all *S-VSM* representations, we decided to conduct experiments only for Omiotis and Omiotis-Wiki average based

Table 7: Dominance Matrix for various Clustering Algorithms evaluated by BCubed FMeasure

Algorithm	Di2	Dh2	RBi2	RBh2	$RBRh_2$	$Baggloh_2$	NMF
$Di2$	0	26	12	25	22	21	17
$Dh2$	3	0	8	13	12	20	15
$RBi2$	13	21	0	24	20	21	18
$RBh2$	4	11	5	0	14	21	13
$RBRh_2$	7	16	10	15	0	20	17
$Baggloh2$	6	10	7	9	10	0	10
NMF	13	15	11	16	13	20	0

semantic kernels using an increasing number of Top-k % terms (from 5 to 95% with a step of 5%). The reason for conducting these experiments is to observe:

1. the impact of less related terms on cluster quality,
2. the algorithms' stability with respect to varying k (top-k) values

Figures 1 and 2 provide in-depth views of the performance of the $GVSM$—Semantic Kernel methods when the two relatedness measures and only the top-k most related terms are used for smoothing ($K-VSMs$). The horizontal axis corresponds to the percentage of terms selected each time, whereas the vertical axis shows the BCubed FMeasure values in each dataset. To summarize the results of Top-k % terms, we divided the Top-k% related terms in 5 ranges i.e 5-20, 25-40,45-60,65-80, and 85-100. We create a range won matrix that is shown in Table 8. Range won matrix shows the number of times a particular range has better results on a particular dataset-algorithm pair. For example, the value for the entry of the row 5-20% top terms is 35, which means for 35 out of the 70 dataset-algorithm pairs, the 5-20% term range has better results than the other ranges.

From the results, we can easily conclude that:

1. Competitive results are achieved even when only 5% Top-k terms are used. Friedman test also validate that there is no overall statistically significant difference between the 20 slots.
2. In most of the cases, after 60% Top values Clustering algorithms stabilize. Bagglo algorithm is more effected due to variation of Top-k values. This is because hierarchical nature of the algorithm. Direct clustering algorithm is the most stabilized algorithm.

Table 8: Range Won Matrix for Top-k% related terms

Top k% Term Range	No. of times win
5 – 20%	35
25 – 40%	16
45 – 60%	11
65 – 80%	7
85 – 100%	1

3. Reduction in the number of terms used results in reduction of time and space complexities of clustering algorithms and the overall clustering process.

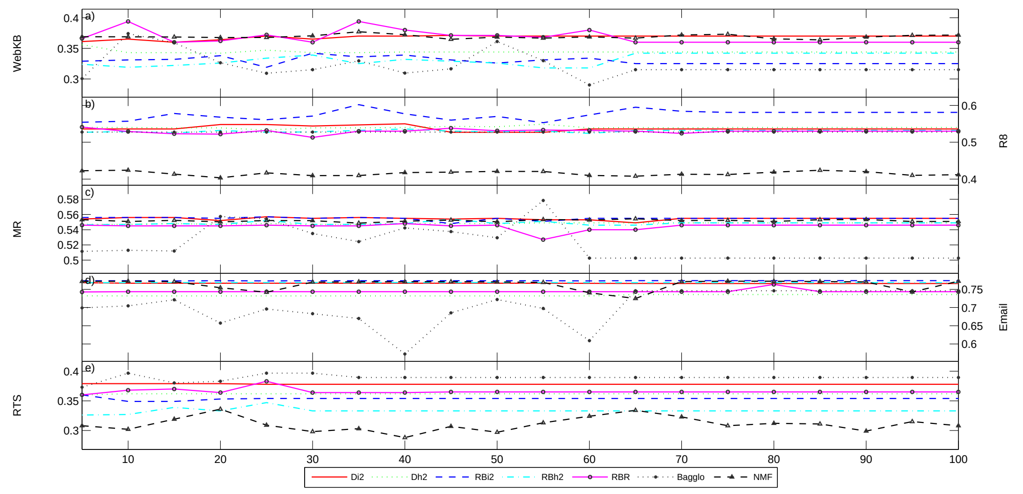

Figure 1: Omiotis-based *top-k S-VSM*

6.4. *Significance Analysis*

To verify the consistency and ranking of the observed results presented in Table 3, we applied the Friedman test with post-hoc tests on the observed differences in performances of all methods on all the data sets. The Friedman test is a non-parametric equivalent of ANOVA. The Friedman test works on the assumption that the data come from populations with the same continuous distributions and all observations are mutually independent. These

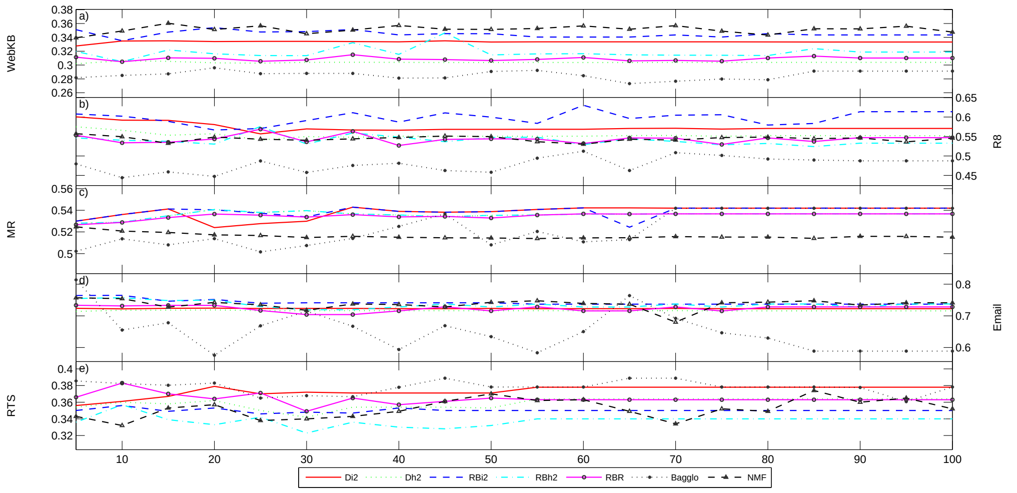

Figure 2: Top-k percent based on Omiotis-Wiki average K-VSM

assumptions are desirable for our case because clustering results from separate algorithms may be extremely variable [44].

The null hypothesis for the Friedman test is that there are no differences between the variables. If the calculated probability is low ($P \leq 0.05$) the null-hypothesis is rejected. There was a statistically significant difference in perceived effort depending on which type of vector space model is used, $\chi^2(2) = 39.414$, $\rho = 0.001$. Post-hoc analysis with Wilcoxon Signed-Rank Tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $\rho < 0.005$. There was a statistically significant difference in TFIDF vs. OMI trial (Z = -4.161, $\rho = 0.001$), TFIDF vs. Wiki trials (Z = -4.038, $\rho = 0.001$), TFIDF vs. OW trials (Z = -4.738, $\rho = 0.001$), and $GVSM$vs. TFIDF (Z=-4.120, ρ=0.000). There was no significant differences between the TFIDF and PMI trials (Z = -2.121, $\rho = 0.034$), OMI and $GVSM$(Z= -1.867, ρ=0.062), and Wiki and Omi (Z=-1.212, ρ=0.225). So it shows that TFIDF is outclass by *S-VSM*.

Similarly we conducted a Friedman test for the clustering algorithms. There is a statistically significant difference in perceived effort depending on which type of clustering algorithm is used, $\chi^2(2) = 31.030$, $\rho = 0.000$. Post-hoc analysis with Wilcoxon Signed-Rank Tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $\rho < 0.005$.

T here was a statistically significant difference in $D - i_2$ vs. Bagglo trial (Z = -3.436, ρ = 0.001).

6.5. Performance under Varying Data Size

In order to test the effect of the data size on the performance of our *S-VSM* in text clustering we used subsets of the WebKB dataset. Subsets were created using startified sampling of 500, 1000, 2000 and 3000 examples (there are total 4199 examples). To account for the high inherent sampling variance, this approach was repeated 10 times for each of the 4 subset sizes. $Direct - i_2$, $Direct - h_2$, and $RB - i_2$ clustering algorithms are run 5 times each on these subsets, resulting in a total 600 experiments. Figure 3 presents the performance of the three clustering algorithms using WebKB subsets of varying size. The x-axis shows the sample size and the y-axis shows the average BCubed Fmeasure values for each dataset. We see that the performance is not affected by the number of samples and thus by the dataset size.

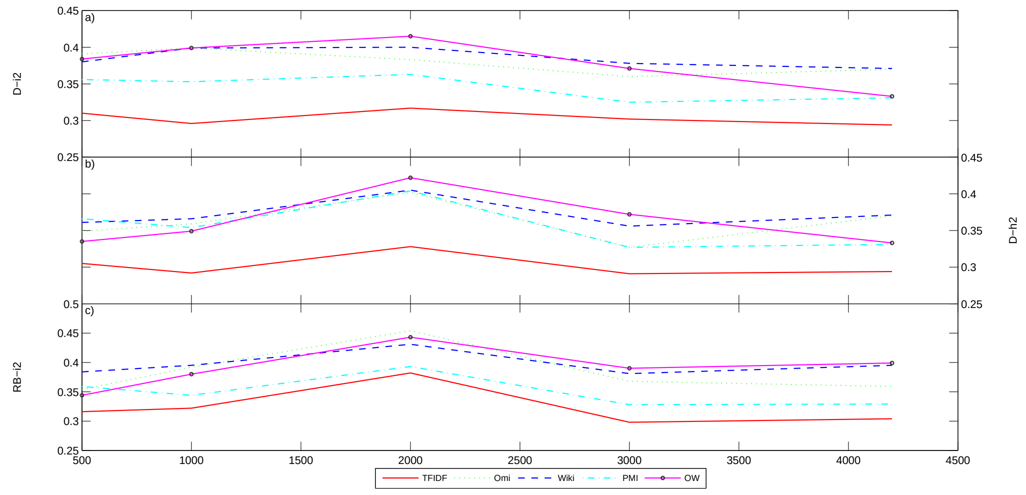

Figure 3: Performance on subsets of WebKB datasets

6.6. Performance with Varying number of clusters K

The next step was to test the effect of the number of clusters in the performance of our *S-VSM* models. For this reason, we used documents from four classes of the WebKB dataset and the Direct-i2 algorithm. As a result, the expected number of clusters was 4. We evaluated several clustering schemas as output, with a varying number of desired clusters (from 2 to 10), and measured their BCubed Fmeasure. Figure 4, shows that *S-VSM* (more especially the Wikipedia based metric) are more resilient than *VSM* to an increase in the number of clusters. In the case of Wikipedia metric the "knee" in four clusters is obvious.

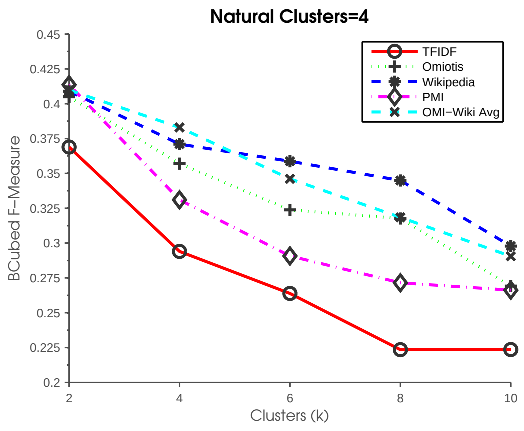

Figure 4: Performance with a varying number of expected clusters (K) on WebKB dataset using Direct-i2

7. Conclusions and Future work

In this paper, we presented a semantic smoothing vector space kernel (*S-VSM*) for document clustering, and exhaustively tested its performance against *VSM* and *GVSM* using different semantic relatedness measures between words, several document clustering algorithms and five popular document sets. The evaluation results demonstrated that *S-VSM* dominates *VSM* in performance in most of the combinations and compares favorably to *GVSM*, which uses word co-occurrences to compute the latent similarities between document terms and has increased space complexity. In order to further reduce the complexity of *S-VSM* we introduced an extension of it, namely the *top-k S-VSM*, which considers only the top-k semantically related terms. The *top-k S-VSM* outperforms *VSM* and is very efficient in terms of time and space complexity. As a next step, we plan to test our representation models to text classification. We also examine the possibility of defining a reduction of the GVSM semantic kernel, which will allow us to reach top performance, while reducing the overall complexity.

References

[1] G. Tsatsaronis, I. Varlamis, M. Vazirgiannis, Text relatedness based on a word thesaurus., J. Artif. Intell. Res. (JAIR) 37 (2010) 1–39.

[2] D. N. Milne, I. H. Witten, An effective, low-cost measure of semantic relatedness obtained from Wikipedia links., in: Proc. of the first AAAI Workshop on Wikipedia and Artificial Intelligence, 2008, pp. 24–30.

[3] M. Turney, Mining the web for synonyms: PMI-IR versus LSA on TOEFL, in: Proc. of the 12th ECML, 2001, pp. 491–502.

[4] C. Manning, P. Raghavan, H. Schutze, Introduction to Information Retrieval, Cambridge University Press, Cambridge, UK, 2008.

[5] G. Salton, A. Wong, C. S. Yang, A vector space model for automatic indexing, Communications of the ACM 18 (11) (1975) 613–620.

[6] M. Steinbach, G. Karypis, V. Kumar, A comparison of document clustering techniques, in: In KDD Workshop on Text Mining, 2000, pp. 525–526.

[7] J. A. Nasir, A. Karim, G. Tsatsaronis, I. Varlamis, A knowledge-based semantic kernel for text classification, in: SPIRE, 2011, pp. 261–266.

[8] L. Jing, M. K. Ng, J. Z. Huang, Knowledge-based vector space model for text clustering, Knowl. Inf. Syst. 25 (1) (2010) 35–55.

[9] A. Budanitsky, G. Hirst, Evaluating WordNet-based measures of lexical semantic relatedness, Computational Linguistics 32 (1) (2006) 13–47.

[10] Z. Zhang, A. Gentile, F. Ciravegna, Recent advances in methods of lexical semantic relatedness - a survey., Natural Language Engineering doi:10.1017/S1351324912000125.

[11] A. Hotho, S. Staab, G. Stumme, Wordnet improves text document clustering, in: In Proc. of the SIGIR 2003 Semantic Web Workshop, 2003, pp. 541–544.

[12] J. Sedding, D. Kazakov, Wordnet-based text document clustering, in: Proceedings of the 3rd Workshop on RObust Methods in Analysis of Natural Language Data, ROMAND '04, 2004, pp. 104–113.

[13] R. Navigli, Word sense disambiguation: A survey, ACM Computing Surveys 41 (2) (2009) 10:1–10:69.

[14] E. Voorhees, Using wordnet to disambiguate word sense for text retrieval, in: Proc. of the 16th SIGIR, ACM, 1993, pp. 171–180.

[15] O. Egozi, S. Markovitch, E. Gabrilovich, Concept-based information retrieval using explicit semantic analysis, ACM Trans. Inf. Syst. 29 (2) (2011) 8.

[16] W. Mao, W. W. Chu, The phrase-based vector space model for automatic retrieval of free-text medical documents, Data Knowl. Eng. 61 (1) (2007) 76–92.

[17] E. SanJuan, F. Ibekwe-Sanjuan, J. M. T. Moreno, P. Velázquez-Morales, Combining vector space model and multi word term extraction for semantic query expansion, in: NLDB, 2007, pp. 252–263.

[18] X. Zhou, X. Zhang, X. Hu, Semantic smoothing of document models for agglomerative clustering, in: IJCAI, 2007, pp. 2928–2933.

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

[20] C. Domeniconi, J. Peng, B. Yan, Composite kernels for semi-supervised clustering, Knowl. Inf. Syst. 28 (1) (2011) 99–116.

[21] Q. Zhang, J. Li, Z. Zhang, Efficient semantic kernel-based text classification using matching pursuit kfda, in: ICONIP (2), 2011, pp. 382–390.

[22] S. J. Fodeh, W. F. Punch, P.-N. Tan, On ontology-driven document clustering using core semantic features, Knowl. Inf. Syst. 28 (2) (2011) 395–421.

[23] H.-T. Zheng, B.-Y. Kang, H.-G. Kim, Exploiting noun phrases and semantic relationships for text document clustering, Inf. Sci. 179 (13) (2009) 2249–2262.

[24] X. Hu, X. Zhang, C. Lu, E. K. Park, X. Zhou, Exploiting wikipedia as external knowledge for document clustering, in: In Proc. of Int. Conf. on Knowledge Discovery and Data Mining (KDD 2009), 2009, pp. 389–396.

[25] S. K. M. Wong, W. Ziarko, P. C. N. Wong, Generalized vector spaces model in information retrieval, in: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '85, 1985, pp. 18–25.

[26] P. Sheridan, J. P. Ballerini, Experiments in multilingual information retrieval using the spider system, in: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '96, 1996, pp. 58–65.

[27] G. Tsatsaronis, V. Panagiotopoulou, A generalized vector space model for text retrieval based on semantic relatedness, in: Proceedings of the 12th EACL Conference: Student Research Workshop, EACL '09, 2009, pp. 70–78.

[28] C. Manning, H. Schutze, Foundations of Statistical Natural Language Processing, MIT Press, 2000.

[29] P. Pecina, An extensive empirical study of collocation extraction methods, in: Proc. of the ACL Student Research Workshop, Ann Arbor, MI, USA, 2005, pp. 13–18.

[30] G. Bouma, Normalized (pointwise) mutual information in collocation extraction, in: From Form to Meaning: Processing Texts Automatically, Proceedings of the Biennial GSCL Conference, Potsdam, Germany, 2009, pp. 31–40.

[31] D. Newman, J. Lau, K. Grieser, T. Baldwin, Automatic evaluation of topic coherence, in: Proceedings of the Annual Conference of the North American Chapter of ACL, Los Angeles, California, 2010, pp. 100–108.

[32] J. B. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297.

[33] Y. Zhao, G. Karypis, U. Fayyad, Hierarchical clustering algorithms for document datasets, Data Min. Knowl. Discov. 10 (2) (2005) 141–168.

[34] F. Shahnaz, M. W. Berry, V. P. Pauca, R. J. Plemmons, Document clustering using nonnegative matrix factorization, Inf. Process. Manage. 42 (2) (2006) 373–386.

[35] D. D. Lee, H. S. Seung, Algorithms for non-negative matrix factorization, in: Proceedings of NIPS, 2000, pp. 556–562.

[36] X. L. Wei Xu, Y. Gong, Document clustering based on non-negative matrix factorization, in: Proceedings of SIGIR, 2005, pp. 267–273.

[37] G. Karypis, Cluto: A Clustering Toolkit, University of Minnesota, Department of Computer Science (November 2003).

[38] N. Cristianini, J. S. Taylor, H. Lodhi, Latent Semantic Kernels, in: Proceedings of the Eighteenth International Conference on Machine Learning, 2001, pp. 66–73.

[39] S. Bloehdorn, R. Basili, M. Cammisa, A. Moschitti, Semantic kernels for text classification based on topological measures of feature similarity, in: Proceedings of the 2006 IEEE International Conference on Data Mining (ICDM'06), 2006, pp. 808–812.

[40] D. Mavroeidis, G. Tsatsaronis, M. Vazirgiannis, M. Theobald, G. Weikum, Word sense disambiguation for exploiting hierarchical thesauri in text classification, in: Proc. of the 9th PKDD, 2005, pp. 181–192.

[41] Y. Zhao, G. Karypis, Evaluation of hierarchical clustering algorithms for document datasets, in: Proceedings of the eleventh international conference on Information and knowledge management, CIKM '02, 2002, pp. 515–524.

[42] A. Bagga, B. Baldwin, Entity-based cross-document coreferencing using the vector space model, in: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL98), Montreal, USA, 1998, pp. 79–85.

[43] E. Amigó, J. Gonzalo, J. Artiles, F. Verdejo, A comparison of extrinsic clustering evaluation metrics based on formal constraints, Inf. Retr. 12 (4) (2009) 461–486.

[44] J. Demsar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.