SemaFor: Semantic Document Indexing using Semantic Forests

George Tsatsaronis
Biotechnology Center
Technische Universität
Dresden
Dresden, Germany
george.tsatsaronis@biotec.tu-dresden.de

Iraklis Varlamis
Department of Informatics and Telematics
Harokopio University of Athens
Athens, Greece
varlamis@hua.gr

Kjetil Nørvåg
Department of Computer and Information Science
Norwegian University of Science and Technology
Trondheim, Norway
Kjetil.Norvag@idi.ntnu.no

ABSTRACT

Traditional document indexing techniques store documents using easily accessible representations, such as inverted indices, which can efficiently scale for large document sets. These structures offer scalable and efficient solutions in text document management tasks, though, they omit the cornerstone of the documents' purpose: ***meaning***. They also neglect semantic relations that bind terms into coherent fragments of text that convey messages. When semantic representations are employed, the documents are mapped to the space of concepts and the similarity measures are adapted appropriately to better fit the retrieval tasks. However, these methods can be slow both at indexing and retrieval time. In this paper we propose *SemaFor*, an indexing algorithm for text documents, which uses semantic spanning forests constructed from lexical resources, like *Wikipedia*, and *WordNet*, and spectral graph theory in order to represent documents for further processing.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: [Retrieval models, Selection process]; H.3.1 [**Content Analysis and Indexing**]: [Linguistic processing, Thesauruses]

General Terms

Algorithms, Experimentation, Theory

Keywords

Document Indexing, Semantic Graphs, Text Representation

1. INTRODUCTION

Document indexing has been traditionally conducted with the use of a term to document mapping and its inverse, which takes into account only the frequency of occurrence of terms in the indexed documents, neglecting semantic relatedness between terms, and their syntactic role in the document. In this paper we propose *SemaFor*, a new document indexing algorithm that takes into account the semantic relatedness of terms within documents. *SemaFor* aims at: (1) extracting information from text, namely terms, and identify their semantic connections, (2) storing the semantic information in an efficient manner that can support fast processing of documents, and, (3) using publicly available resources for the task, and an efficient methodology that does not require any type of training, so that it can scale up for large document collections, and be applied across different domains (domain agnostic).

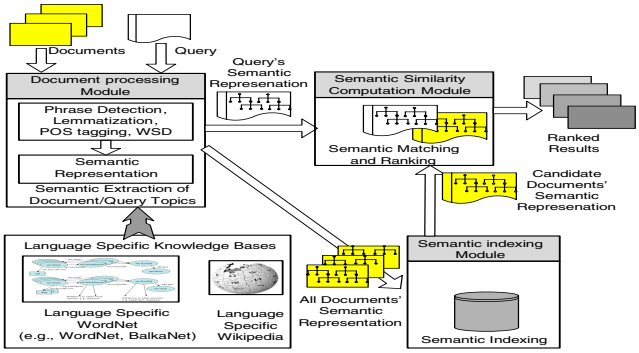

Figure 1: The high-level representation of the *SemaFor* process flow.

SemaFor lies in the formulation of semantic spanning trees (SSTs) and semantic spanning forests (SSFs). Each document is first parsed and transformed into a set of SSTs, each one corresponding to a document topic. The SSF that contains the respective SSTs is the document's semantic representation. The forests are indexed following an efficient methodology that allows fast retrieval of potential matches at query time. A similarity measure for semantic spanning forests based on spectral graph algebra is introduced in order to provide the ability of the index to produce similarity scores between documents (SSFs),i.e., for the purposes of documents clustering, or create a ranking of the potential matching *SSF*s to a given user query for the purposes of document retrieval. The constructed index allows for fast search and ranking of the semantic forests (documents), given a user query.

A high level representation of the process flow in *SemaFor*, is shown in Figure 1. Given an initial set of input text documents, the *semantic extraction of document topics* module pre-processes the texts and creates a semantic spanning forest for each document, as explained in Section 3. The *semantic indexing* module indexes doc-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'12, October 29–November 2, 2012, Maui, HI, USA.
Copyright 2012 ACM 978-1-4503-1156-4/12/10 ...$15.00.

uments in the form of semantic spanning forests. Similarity computation is performed using ***spectral graph algebra***, as explained in detail in Section 3.

2. RELATED WORK

The basic hypothesis behind our approach is that the use of semantic information for the representation of documents may improve the performance of the text clustering and retrieval tasks, both in precision and recall. The hypothesis is based on previously published scientific indications, e.g., [10].

In one direction, several approaches attempt to capture semantic relatedness between terms using statistical analysis of corpora. They attempt to group the terms of a document into subsets (topics) that contain statistically "related" terms, in order to represent documents as combinations of one or more topics [4, 5]. However, such approaches require extensive analysis of large text corpora, and the produced models cannot be easily transferred across domains. In another direction, the use of lexical and other knowledge resources is employed in order for the documents to be represented as graphs of terms [14]. Recent studies, e.g., [19] have shown that linguistic and crowdsource-based knowledge sources, for example *WordNet* and *Wikipedia* respectively, can be used complementary in this task. The processing of document semantics in *SemaFor* also results in a graph, which contains the document terms only. Though *SemaFor* does not perform topic detection literally, the SSTs of each indexed document can be seen as the document topics. Taking one step further to the aforementioned approaches, *SemaFor* indexes the document graph using a mechanism that facilitates storage and fast processing, and incorporates semantic information inside the indexing data structures. For the task of the graph creation it uses both *WordNet* and *Wikipedia*, combining the "*wisdom of linguists*" and "*wisdom of crowds*". Close to our approach are also the works that embed senses and semantic information for text document management, like for example *Generalized Vector Space Models* (GVSM) [12] and *semantic kernels* [2].

An important point in existing approaches is the consideration of *word sense disambiguation* methods (*WSD*) which can potentially offer the transit from terms to senses. In this paper we address word disambiguation by employing a very simple *WSD* algorithm that provides state of the art performance and is used as a very competitive baseline for *WSD* methods; the *first sense heuristic*, which selects the most frequently appearing sense of each word [9].[1].

Finally, with regards to semantic indexing methodologies, existing approaches map documents to graphs, yet they do not consider the semantic information at indexing level. In [7], each document is mapped to a graph with terms as vertices and 4 types of edges (based on *WordNet* relations). The graph structure is neither indexed, nor employed in the computation of similarity between documents. In [15] documents are mapped to semantic forests using the co-occurrence of terms (actually stems) and their semantic relations (as given by *WordNet*) in order to draw semantic relations between terms. During the indexing and document similarity computation phases, the graph information is neglected and each forest is perceived as a set of terms. In contrast to the aforementioned approaches, *SemaFor* introduces a lightweight representation of the document graph that keeps only the strongest edges and employs *spectral graph theory* in order to convert the spanning trees into an indexable format.

[1]In our implementation we are using *WordNet* as the main dictionary, or *Wikipedia* definitions if the term is ambiguous and does not appear in Wordnet

Algorithm 1 *SSF*(D)

INPUT: A text document D.
OUTPUT: The Semantic Spanning Forest (*SSF*) of D, *SSF*(D)
T: A set of term-POS pairs
G, *SSF*: Initially empty graphs
V: The set of vertices of G, E: The set of edges of G
T := PreProcessDoc(D)
for all $tp \in T$ and $tp \in$*WordNet* **do**
  Disambiguate(tp)
end for
for all $i = 1$ to $|T| - 1$ **do**
  for all $j = i + 1$ to $|T|$ **do**
    if $tp_i, tp_j \in$ *WordNet* **then**
      $S(tp_i, tp_j) = SR(tp_i, tp_j)$
    else
      $S(tp_i, tp_j) =$*WLM*(tp_i, tp_j)
    end if
    if $S(tp_i, tp_j) > 0$ **then**
      Add tp_i, tp_j in V and $\frac{1}{S(tp_i, tp_j)}$ in E
    end if
  end for
end for
for all $c \in$ connected components of G **do**
  SSF$\cup$ MinimumSpanningTree(c)
end for
RETURN *SSF*

3. THE SEMAFOR ARCHITECTURE

In this section we present the details of the *SemaFor* architecture as shown in the high level representation of Figure 1. In Section 3.1, we explain the operations of the *Document Processing module*, which constructs the semantic spanning forests given a set of documents. Section 3.2 explains the details of the Semantic Indexing module: (a) how the semantic spanning forest is transformed into a set of points in a metric space using *spectral graph theory*, and (b) what information is stored in the index for each semantic spanning forest. Section 3.3 illustrates the *spectral graph similarity computation* module, the details of the distance metric and the algorithm employed in *SemaFor* for document comparison and similarity computation.

3.1 Construction of Semantic Spanning Forests

Given a document D, the semantic spanning forest construction process (Alg. 1) comprises three steps: (a) the pre-processing of the document, i.e., part-of-speech tagging (POS tagging) and phrase detection, (b) word sense disambiguation, and (c) construction of the semantic spanning forest using measures of semantic relatedness. For a given document D of the collection, we initially perform POS tagging using the *Stanford Part of Speech Tagger* [11], which also enables us to perform sentence splitting in the document. Next, phrase detection is performed to recognize terms of more than two words. The phrase recognition takes place by simple dictionary look up, which in our case consists of *WordNet* and *Wikipedia*, examining only the noun phrases of each sentence. Finally, all stopwords are removed. In the *WSD* step we use the *first sense heuristic* approach to disambiguate the terms into their respective sense, by assigning to each term its most frequent sense consulting *WordNet* as the reference dictionary, or *Wikipedia* in the cases that the term does not exist in *WordNet*.

The algorithm of the $SSF(D)$ construction, for a given document D is described by Alg. 1. Given that D contains a set of

n term-*POS* pairs, namely $T = tp_1, tp_2, ..., tp_n$, in the remaining of this section we describe how a semantic spanning forest is constructed from this set. Primarily, note that for any given pair (tp_i, tp_j) with $i \neq j$ and both $i, j \in [1..n]$ the t part of the term-pair tp_i might be identical with the t part of the tp_j term pair, but then the p part in the two term pairs must differ (i.e, we keep the set of all distinct term-*POS* pairs for D).

Initially, we compute the semantic relatedness *S* between every term-pair combination in T. In our implementation, for pairs of terms that exist in *WordNet* we are using *Omiotis* [13]; for the rest we are using *WLM*, a *Wikipedia*-based measure [8]. *Omiotis*, which has been shown to outperform *WLM* in case both terms exist in *WordNet* [13]. Note that the suggested methodology is general enough to allow for the use of any other measure of semantic relatedness or similarity. Both used measures are in the range of $[0, 1]$, with 1 meaning totally related and 0 meaning totally unrelated, they are publicly available and their performance is state-of-the-art in their category of measures [19]. Since for both measures $S(tp_1, tp_2) = S(tp_2, tp_1)$ we need exactly $\frac{n \cdot (n-1)}{2}$ computations of semantic relatedness. Based on the above we define $S(tp_i, tp_j) = SR(tp_i, tp_j)$, if $tp_i, tp_j \in$ WordNet, else $S(tp_i, tp_j) = WLM(tp_i, tp_j)$.

Next, we construct a semantic graph which initially contains all the elements of T as nodes. Each node represents a term-*POS* element of D. We add an edge e_{tp_i,tp_j} between every pair of nodes (tp_i, tp_j) for which $S(tp_i, tp_j) > 0$, with weight $w_{tp_i,tp_j} = \frac{1}{S(tp_i,tp_j)}$, and $i \neq j$.

Once all the edges have been added, the semantic graph contains terms as nodes and reverse semantic relatedness values between them as edges. For each connected component of the graph, i.e., as this is defined by traditional graph theory, we apply the computation of the minimum spanning tree algorithm of Kruskal. D is now a set of *minimum semantic spanning trees*. We define this set as the *Semantic Spanning Forest* representing document D (*SSF*(D)), and each i-th semantic tree of D (*SST*$_i$(D)) as one of its topics.

3.2 Indexing of Semantic Spanning Forests

Having the documents in the form of semantic spanning forests (*SSF*), we now proceed in representing them to a metric space where we can compute similarity between documents. For their similarity, we are based on the *spectra* of the normalized Laplacian of the two bipartite graphs, following the basis of spectral graph theory [3]. The similarity between two semantic forests is eventually based on the computation of the *Hausdorff* distance [1] between the two *SSF*s, which considers the spectral properties of the two graphs, and more specifically the *sectional curvatures* of their edges. The *Hausdorff* distance has been shown to perform very well in the application of graph clustering in the field of computer vision [6]. Thus, the following procedure, though not new, it constitutes a novel embedding in our case, since it is for the first time, to the best of our knowledge, that it is applied in graphs representing documents, as a means of *SSF*s. We give details on the *Hausdorff* distance in the following section.

In the following we explain the details of the first application of this technique in text processing and more specifically we show how *SSF*s are transformed to facilitate spectral similarity computation. Initially, let $G(V, E)$ be a graph, which in our case represents a document as a means of a *SSF*, where V is the set of its vertices, and E the set of its edges. For reasons of simplicity, let us also assume that G is connected, forming a spanning tree. Primarily, for every such graph in our document collection, we compute the degree d_v of each vertex $v \in V$ as:

$$d_v = \sum_u w(v, u) \tag{1}$$

where vertex $u \in V$ is any adjacent node to vertex v and $w(v, u) = w(u, v)$ is the weight of the edge connecting them.

Then, the Laplacian L of G can be computed as follows:

$$L(u, v) = \begin{cases} d_v - w(v, v), & \text{if } u = v \\ -w(u, v), & \text{if u and v are adjacent} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

We also construct a diagonal matrix D, with $D(v, v) = d_v$, in order to compute the normalized Laplacian $\hat{L}$ of G. The $\hat{L}$ matrix is needed, as its eigenvalues constitute the spectrum of the initial graph. $\hat{L}$ is computed as follows:

$$\hat{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} \tag{3}$$

and the spectral decomposition of the normalized Laplacian $\hat{L}$ as:

$$\hat{L} = \Phi \Lambda \Phi^T \tag{4}$$

where Λ is the diagonal matrix with the ordered eigenvalues as its elements and Φ contains the eigenvectors as columns.

To measure the *Hausdorff* distance between two *SSF*s, we need to embed the nodes of each *SSF* into a vector space. There is a strong connection between the heat kernel of a graph and the manifold in which its node reside [17]. Thus, we initially compute the heat kernel h_t [3], which encapsulates the way information flows through the graph edges over time. Essentially the heat kernel can be computed by exponentiating the $\hat{L}$ matrix using a parameter t that stands for time. Higher values of t give more focus to the full graph (i.e., trust more the edges of the entirety of the full graph), in contrast to lower t values that focus on the locality of the graph. The heat kernel in our case can be computed as follows:

$$h_t = exp[-\hat{L}t] = \Phi exp[-t\Lambda]\Phi^T \tag{5}$$

We can obtain the matrix that contains the coordinates for each node in this new vector space. This can be done by applying the *Young-Householder* decomposition [18] of the heat kernel $h_t = Y^T Y$, where on Y the columns will represent the nodes as vectors in the vector space. As a result, the matrix of the resulting coordinates is expressed as:

$$Y = exp[-\frac{1}{2}t\Lambda]\Phi^T \tag{6}$$

In this new vector space, the Euclidean distance between nodes (u, v) of G can then be computed as [6]:

$$d_E^2(u, v) = \sum_{i=1}^{|V|} exp[-\lambda_i t](\phi_i(u) - \phi_i(v))^2 \tag{7}$$

where λ_i is the i-th eigenvalue in Λ (the non-zero value in the i-th row of the Λ matrix), and $\phi_i(u)$ is the value in position (i, u) of the eigenvector matrix Φ. Since for the computation of the *Hausdorff* distance we need the sectional curvature of the edges of G, we require the geodesic distance of the nodes (u, v), in addition to their Euclidean distance. This can be computed as follows [6]:

$$d_G(u, v) = floor_n\{\sum_{i=1}^{|V|}(1 - \lambda_i)^n \phi_i(u)\phi_i(v)\} \tag{8}$$

where n constitutes the length of the walk on the *SSF* with the smallest number of connecting edges. Eventually n is the smallest value for which the sum in Equation 8 becomes positive.

Algorithm 2 *SemaFor*(*SSF*, t)

1: **INPUT:** A semantic spanning forest *SSF*, the parameter t of the heat kernel.
2: **OUTPUT:** The indexing of *SSF* as a set of ordered lists of real values in a low-dimensional space.
3: $\hat{L}, \Lambda, \Phi, K$: Initially empty matrices
4: K_{set}: An initially empty set of ordered real values
5: L: An initially empty list of K_{set}
6: *SST*: An initially empty set of trees
7: **for all** $s \in SST$ **do**
8: $\hat{L}$:= NormalizedLaplacian(s)
9: Λ, Φ := EigenValueDecomposition($\hat{L}$)
10: **for all** (u, v) pairs $\in s$ **do**
11: $d_E^2(u,v) := \sum_{i=1}^{|V|} exp[-\lambda_i t](\phi_i(u) - \phi_i(v))^2$
12: $d_G(u,v) := floor_n\{\sum_{i=1}^{|V|}(1-\lambda_i)^n \phi_i(u)\phi_i(v)\}$
13: $K[u,v] := \frac{2\sqrt{6}(d_G(u,v) - d_E(u,v)^{\frac{1}{2}}}{d_G(u,v)^{\frac{3}{2}}}$
14: **end for**
15: K_{set} := OrderValuesOf(K)
16: L := AddToList(K_{set})
17: **end for**
18: Store *SSF* as L

Eventually, the sectional curvature of the edge (u, v) can be computed as follows (the proof can be found in [16]):

$$k(u,v) = \frac{2\sqrt{6}(d_G(u,v) - d_E(u,v)^{\frac{1}{2}}}{d_G(u,v)^{\frac{3}{2}}} \quad (9)$$

The sectional curvatures of the *SSF* are the only information that we index for our tree structures. Essentially, the sectional curvatures capture the topological structure of *SSF* and allows us to construct a low-dimensional feature space in which these values reside. Ultimately, we only need to index those values instead of the full *SSF* structure. Algorithm 2 describes the *SemaFor* document indexing algorithm. It assumes that the *SSF* of a given document D has already been computed (using Alg 1). Eventually, a list of sets of ordered real values are indexed for the *SSF*. Each set represents each *SST* of the *SSF*, and the values are the respective sectional curvatures of the *SST* edges.

3.3 The *Hausdorff* Distance

Given two graphs $G_1(V_1, E_1, K_1)$ and $G_2(V_2, E_2, K_2)$, where V_1, V_2 are the respective sets of their vertices, E_1, E_2 are the respective sets of their edges, and K_1, K_2 are the respective matrices with the sectional curvatures of their edges (e.g., $K_1(u, v)$ is the sectional curvature of edge (u, v) in G_1) we are using the *Hausdorff* distance [6] to compute the distance between G_1 and G_2 as follows:

$$Hausdorff(G_1, G_2) = \max_{i,j \in V_1} \min_{I,J \in V_2} ||k_2(I,J) - k_1(i,j)|| \quad (10)$$

The *Hausdorff* distance in our case is a *maximin* function between the sectional curvature matrices. Since we have assumed that the *SSF*s we are examining are connected, we will generalize Equation 10 to capture all the cases, i.e., cases that *SSF* may contain several semantic spanning trees (*SST*). The generalization takes place in a similar manner that the average-link works during the agglomeration step in the hierarchical agglomerative clustering (*HAC*). The reason is simple: given two sets (i.e., the *SSF*s) of elements (their *SST*s), we estimate the distance between sets based on the *Hausdorff* distance between elements. This is exactly the problem faced by the *HAC* algorithm.

Reuters Subset	VSM	LSI	CF	SemaFor
C1	0.64	0.64	0.74	0.94
C2	0.5	0.62	0.8	0.84
C3	0.25	0.34	0.48	0.71

Table 1: Overall clustering accuracies on the Reuters subsets.

Subset	Cat.	P	R	F1	MP	MR	MF1
C1	**Oil**	0.92	0.958	0.938	0.94	0.94	0.94
	Nat-Gas	0.96	0.923	0.941			
C2	**Coffee**	0.693	1.0	0.819	0.847	0.875	0.86
	Sugar	1.0	0.75	0.857			
C3	**Grain**	0.51	0.91	0.66	0.69	0.84	0.76
	Wheat	0.51	0.86	0.64			
	Ship	1.0	0.6	0.75			
	Crude	0.77	1.0	0.86			

Table 2: Detailed clustering results on the Reuters subsets.

The possible solutions are: (a) single-link, with the caveat of the effect of chaining, (b) complete-link, with the caveat that can be sensitive to outliers (i.e., small *SST* in our case, describing small document topics that are quite distant from the larger *SST*, meaning the larger document topics), and (c) average-link, which is a compromise between the sensitivity of complete-link to outliers and the lack of compactness of single-link. If solution (c) is chosen, the generalization of the *Hausdorff* distance between G_1 and G_2 becomes:

$$Hausdorff^*(G_1, G_2) = \frac{\sum_{i=1}^{|SST_{G_1}|} \sum_{j=1}^{|SST_{G_2}|} Haudorff(SST_i, SST_j)}{|SST_{G_1}| \cdot |SST_{G_2}|} \quad (11)$$

where $|SST_{G_1}|, |SST_{G_2}|$ is the number of *SST* in G_1 and G_2 respectively, and SST_i, SST_j are the i-th and j-th *SST* of G_1 and G_2 respectively.

In the remaining of the paper, we will be using Equation 11 whenever the distance computation between documents is required, e.g., document clustering using *HAC*, and its inverse, whenever similarity is required (e.g., similarity between a query and a document). Note that for two documents D_1 and D_2 that are identical, their *SSF*s are identical. In this case we do not use Equation 11, because it uses the average-link, and we assume $Hausdorff^*(G_1, G_2) = 0$ and the respective similarity being a very large positive constant.

4. EXPERIMENTAL EVALUATION

We experimentally evaluate *SemaFor* in the text clustering and retrieval tasks. We are using the *Reuters*-21578 collection for the former task and the *TREC* collection for the latter.

4.1 Text Documents Clustering

The application of *SemaFor* in clustering is straightforward, and can be easily embedded into the hierarchical agglomerative algorithm (HAC) (i.e, a distance between two documents is the *Hausdorff* distance of their *SSF*s). To evaluate the performance of *SemaFor* in text clustering we use the *Reuters*-21578 data set, comprising approximately $21,500$ files organized in 132 (possibly overlapping) categories. We are comparing its performance against a standard baseline, namely vector space document representation with *TF-IDF* term weights, *LSI*, and the *Concept Forest* text docu-

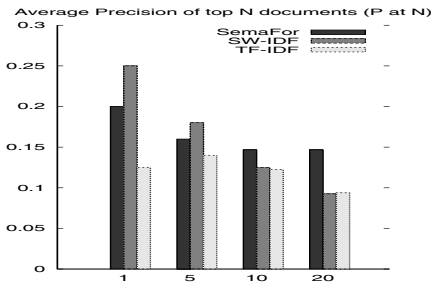

Figure 2: Average Precision of top N documents.

ment similarity approach [15]. In order to be compatible with the results presented in [15], we are using the same document subsets, produced as described in their respective work: (1) $C1$, comprising 50 documents in total from the *Oil* and *Nat-Gas* categories (25 documents from each category), (2) $C2$, comprising 100 documents in total from the *Coffee* and *Sugar* categories (50 documents from each category), and (3) $C3$, comprising 200 documents from the *Grain*, *Wheat*, *Ship* and *Crude* categories (50 documents in each category). For our evaluation, we compute precision, recall, and F-Measure (or F_1 score) for each category in every case ($C1, C2$, and $C3$), as well as their macro-averages, and overall accuracy. The accuracy results, that are directly comparable with the results reported in [15] are shown in Table 1. Table 2 contains the detailed results of *SemaFor* for each category, in each subset, where *MP*, *MR*, and *MF1* are macro-averaged precision, recall and F1-score respectively.

4.2 Text Retrieval

For the text retrieval evaluation of *SemaFor* we are using the *TREC2* document collection, and more specifically the *Wall Street Journal* articles from 1990, so that we can directly compare with the semantic indexing approach proposed by Kang and Lee [7]. This document set comprises $21,705$ articles, and the 50 query topics $101 - 150$ from the respective collection are used.

Figure 2 shows the average precision results of top N documents over all queries for *SemaFor*, the *SW-IDF* semantic indexing approach introduced in [7] and the standard baseline. The *SW-IDF* and *TF-IDF VSM* results are taken from [7].

The results show that the precision of *SemaFor* is higher than that of *SW-IDF* ([7]) in the top-10 ($k = 10$) and top-20 ($k = 20$) documents, which is the typical amount of retrieval results that a user examines in a search. Precision is always higher than that of the baseline method. The top results of *SemaFor* are better than that of its competitors.

5. CONCLUSIONS AND FUTURE WORK

In this work, we presented *SemaFor*, a novel document indexing algorithm that is based on the spectra of the documents' semantic graphs to represent and index documents. *SemaFor* uses thesauri (*WordNet* and *Wikipedia*) in order to extract the semantic relations between documents' terms. The indexing algorithm employs algebraic transformations from spectral graph theory in order to provide a reduced and compact representation for each document. The *Hausdorff* distance is used to define the distance between two documents. Evaluation in text clustering experiments shows that the spectrum-based graph representation of the documents can improve significantly the performance of the text clustering process, and has satisfactory performance in the retrieval task. Our next steps involve the optimization of the current implementation and its expansion in further applications.

6. REFERENCES

[1] M. Barnsley. *Fractals Everywhere*. Morgan Kaufmann, 2000.

[2] R. Basili, M. Cammisa, and A. Moschitti. A semantic kernel to exploit linguistic knowledge. In *Proc. of the AI*IA*, 2005.

[3] N. Biggs. *Algebraic Graph Theory*. Cambridge University Press, 1993.

[4] W. Buntine, J. Löfström, J. Perkiö, S. Perttu, V. Poroshin, T. Silander, H. Tirri, A. Tuominen, and V. Tuulos. A scalable topic-based open source search engine. In *Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 228–234, 2004.

[5] J. Chang, J. Lee, Y. Kim, and B. Zhang. Topic extraction from text documents using multiple-cause networks. In *Proc. of PRICAI*, pages 434–443, 2002.

[6] H. ElGhawalby and R. Hancock. Measuring graph similarity using spectral geometry. In *Proc. of the 5th International Conference on Image Analysis and Recognition*, 2008.

[7] B. Kang and S. Lee. Document indexing: A concept-based approach to term weight estimation. *Information Processing and Management*, 41:1065–1080, 2005.

[8] O. Milne and I. Witten. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proc. of the first AAAI Workshop on Wikipedia and AI*, 2008.

[9] R. Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2), Article 10, 2009.

[10] P. Schone, J. Towsend, T. Crystal, and C. Olano. Text retrieval via semantic forests. In *Proc. of the Sixth Text Retrieval Conference (TREC6)*, pages 761–773, 1997.

[11] K. Toutanova, D. Klein, C. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of HLT-NAACL*, pages 252–259, Canada, 2003. ACM.

[12] G. Tsatsaronis and V. Panagiotopoulou. A generalized vector space model for text retrieval based on semantic relatedness. In *Proc. of the EACL 2009 (Student Research Workshop)*, pages 70–78, 2009.

[13] G. Tsatsaronis, I. Varlamis, and M. Vazirgiannis. Text relatedness based on a word thesaurus. *Journal of Artificial Intelligence Research*, 37:1–39, 2010.

[14] G. Tsatsaronis, M. Vazirgiannis, and I. Androutsopoulos. Word sense disambiguation with spreading activation networks generated from thesauri. In *Proc. of the 20th IJCAI*, pages 1725–1730, 2007.

[15] J. Wang and W. Taylor. Concept forest: A new ontology-assisted text document similarity measurement method. In *Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 395–401, 2007.

[16] B. Xiao and E. Hancock. Geometric characterisation of graphs. In *Proc. of the International Conference on Image Analysis and Processing (ICIAP)*, pages 471–478, 2005.

[17] S. Yau and R. Scoen. *Differential Geometry*. Science Publication, 1988.

[18] G. Young and A. Householder. Discussion of a set of points in terms of their mutual distances. *Psychometrica*, 3, 1938.

[19] Z. Zhang, A. Gentile, and F. Ciravegna. Recent advances in methods of lexical semantic relatedness - a survey. *Natural Language Engineering*, doi:10.1017/S1351324912000125, 2012.