Computing Text Semantic Relatedness using the Contents and Links of a Hypertext Encyclopedia

Majid Yazdani[a,b], Andrei Popescu-Belis[a]

[a] *Idiap Research Institute*
1920 Martigny, Switzerland
[b] *EPFL, École Polytechnique Fédérale de Lausanne*
1015 Lausanne, Switzerland

Abstract

We propose a method for computing semantic relatedness between words or texts by using knowledge from hypertext encyclopedias such as Wikipedia. A network of concepts is built by filtering the encyclopedia's articles, each concept corresponding to an article. Two types of weighted links between concepts are considered: one based on hyperlinks between the texts of the articles, and another one based on the lexical similarity between them. We propose and implement an efficient random walk algorithm that computes the distance between nodes, and then between sets of nodes, using the visiting probability from one (set of) node(s) to another. Moreover, to make the algorithm tractable, we propose and validate empirically two truncation methods, and then use an embedding space to learn an approximation of visiting probability. To evaluate the proposed distance, we apply our method to four important tasks in natural language processing: word similarity, document similarity, document clustering and classification, and ranking in information retrieval. The performance of the method is state-of-the-art or close to it for each task, thus demonstrating the generality of the knowledge resource. Moreover, using both hyperlinks and lexical similarity links improves the scores with respect to a method using only one of them, because hyperlinks bring additional real-world knowledge not captured by lexical similarity.

Keywords: Text semantic relatedness, Distance metric learning, Learning to rank, Random walk, Text classification, Text similarity, Document clustering, Information retrieval, Word similarity

1. Introduction

Estimating the semantic relatedness of two text fragments – such as words, sentences, or entire documents – is important for many natural language processing or information retrieval applications. For instance, semantic relatedness has been used for spelling correction [1], word sense disambiguation [2, 3], or coreference resolution [4]. It has also been shown to help inducing information extraction patterns [5], performing semantic indexing for information retrieval [6], or assessing topic coherence [7].

Existing measures of semantic relatedness based on lexical overlap, though widely used, are of little help when text similarity is not based on identical words. Moreover, they assume that words are independent, which is generally not the case. Other measures, such as PLSA or LDA, attempt to model in a probabilistic way the relations between words and topics as they occur in texts, but do not make use of structured knowledge, now available on a large scale, to go beyond word distribution properties. Therefore, computing text semantic relatedness based on concepts and their relations, which have linguistic as well as extra-linguistic dimensions, remains a challenge especially in the general domain and/or over noisy texts.

In this paper, we propose to compute semantic relatedness between sets of words using the knowledge enclosed in a large hypertext encyclopedia, with specific reference to the English version of Wikipedia used in the experimental

Email addresses: majid.yazdani@epfl.ch, majid.yazdani@idiap.ch (Majid Yazdani), andrei.popescu-belis@idiap.ch (Andrei Popescu-Belis)

part. We propose a method to exploit this knowledge for estimating conceptual relatedness (defined in Section 2) following a statistical, unsupervised approach, which improves over past attempts (reviewed in Section 3) by making use of the large-scale, weakly structured knowledge embodied in the links between concepts. The method starts by building a network of concepts under the assumption that every encyclopedia article corresponds to a concept node in the network. Two types of links between nodes are constructed: one by using the original hyperlinks between articles, and the other one by using lexical similarity between the articles' content (Section 4).

This resource is used for estimating the semantic relatedness of two text fragments (or sets of words), as follows. Each fragment is first projected onto the concept network (Section 5). Then, the two resulting weighted sets of concepts are compared using a graph-based distance, which is computed based on the distance between two concepts. This is estimated using the *visiting probability (VP)* of a random walk over the network from one concept to another, following the two types of links (Section 6). Visiting probability integrates 'density of connectivity' and 'length of path' factors for computing a relevant measure of conceptual relatedness in the network. Several approximations based on truncation of paths are proposed (Section 7) and justified (Section 8) in order to make the computations tractable over a very large network, with 1.2 million concepts and 35 million links. Moreover, a method to learn an approximation of visiting probability using an embedding space (Section 9) is shown to be another solution to the tractability problem.

To demonstrate the practical relevance of the proposed resources – the network, the distance over it, and the approximations – we apply them to four natural language processing problems that should benefit from an accurate semantic distance: word similarity (Section 10), document similarity (Section 11), document clustering and classification (Sections 12 and 13 respectively), and information retrieval (Section 14), including learning to rank (Section 15). The results of our method are competitive on all four tasks, and demonstrate that the method provides a unified and robust answer to measuring semantic relatedness.

2. Semantic Relatedness: Definitions and Issues

Two samples of language are said to be semantically related if they are about things that are associated in the world, i.e. bearing some influence one upon the other, or being evoked together, in speech or thought, more often than other things. Semantic relatedness is a multi-faceted notion, as it depends on the scale of the language samples (words vs. texts) and on what exactly counts as a relation. In any case, the adjective 'semantic' indicates that we are concerned with relation between the senses or denotations, and not, e.g., surface forms or etymology.

2.1. Nature of Semantic Relations for Words and Texts

Semantic relations between words, or rather between their senses, have been well studied and categorized in linguistics. They include classical relations such as synonymy (identity of senses, e.g. 'freedom' and 'liberty'), antonymy (opposition of senses such as 'increase' vs. 'decrease'), hypernymy or hyponymy (e.g. 'vehicle' and 'car'), and meronymy or holonymy (part-whole relation such as 'wheel' and 'car'). From this point of view, semantic similarity is more specific than semantic relatedness. For instance, antonyms are related, but not similar. Or, following Resnik [8], 'car' and 'bicycle' are more similar (as hyponyms of 'vehicle') than 'car' and 'gasoline', though the latter pair may seem more related in the world. Classical semantic relations are listed in hand-crafted lexical or ontological resources, such as WordNet [9] or Cyc [10], or implicitly in Roget's Thesaurus (as used by Jarmasz [11]), or they can be inferred from distributional data as discussed below.

Additional types of lexical relations have been described as 'non-classical' by Morris and Hirst [12], for instance based on membership in similar classes (e.g. positive qualities), or on association by location, or due to stereotypes – but these relations do not qualify as similarity ones, and are generally not listed in lexical resources. Budanitsky and Hirst [1] point out that semantic 'distance' can be seen as the contrary of either similarity or relatedness. In this paper, our use of 'distance' will refer to our measure of semantic relatedness as defined below.

At the sentence level, semantic relatedness can subsume notions such as paraphrase or logical relations (e.g., entailment or contradiction). More generally, two sentences can be related by a similarity of topic, a notion that applies to multi-sentence texts as well, even though the notion of 'topic' is difficult to define. Topicality is often expressed in terms of the continuity of themes, i.e. referents or entities about which something is predicated, which ensures the coherence of texts. Linguists have analyzed coherence as being maintained by cohesive devices [13, 14], which include identity-of-reference, lexical cohesion, and similarity chains based on classical lexical relations [15, 16].

A key relation between the semantic relatedness of words and their occurrence in texts has long been exploited by researchers in natural language processing (NLP) under the form of distributional measures [17], despite certain limitations pointed out by Budanitsky and Hirst [1, Section 6.2]. The assumption that sentences and texts form coherent units makes it indeed possible to infer word meanings and lexical relations from distributional similarity [18], using vector-based models such as Latent Semantic Analysis [19], possibly enhanced with syntactic information [20]. In return, the hidden topical parameters that govern the occurrences of words can be modeled probabilistically (e.g. using PLSA [21] or LDA [22]), thus providing measures of text similarity.

2.2. Use of Encyclopedic Knowledge for Semantic Relatedness

Semantic relatedness has been mainly considered from the perspective of repertoires of semantic relations (often hand-crafted), or from the perspective of relations inferred from the distributional properties of words in collections of texts. However, the computation and use of relations founded on real-world knowledge has been considerably less explored, as it was made possible only recently by the emergence of large-scale hypertext encyclopedias such as Wikipedia. We believe that the use of encyclopedic knowledge may significantly complement semantic relatedness measures based on word distributions only: in the remainder of this section we briefly frame encyclopedic knowledge, outline our proposal, and discuss its task-based validation.

Encyclopedias are lists of general concepts and named entities, accompanied by descriptions in natural language. They differ from dictionaries as they describe concepts or entities, rather than define words, and provide significant factual knowledge for grounding them in the real world, rather than linguistic information only. While printed encyclopedias already include certain references from one entry to another, the linking mechanism is used much more extensively within hypertext encyclopedias such as Wikipedia. As a result, hypertext encyclopedias seem quite adept at capturing semantic relations between concepts, which range from culture-specific to universal ones, including classical and non-classical relations mentioned above.

To measure the extent to which two text fragments are semantically related according to an encyclopedia, two main operations are necessary. First, the concepts related to the texts must be identified. These can be either concepts directly mentioned in the texts, or otherwise related to it, in a sense that will be specified in Section 5 below. Second, the relatedness or proximity of the two sets of concepts thus identified must be measured. This presupposes the capacity to measure the relatedness of two concepts in the first place, taking advantage of the contents of the corresponding articles, and of the hyperlinks between them (see Section 6).

Empirical evidence should support the definition of relatedness and demonstrate its relevance to NLP applications such as those cited at the beginning of this paper. The measure proposed here will be judged and compared to others based on its performance over a variety of tasks, following an empirical stance similar to those expressed in the introductions to their articles by Budanitsky and Hirst [1] and Padó and Lapata [20], to cite only two examples. Our proposal will be applied to word similarity, document similarity, document clustering and classification, and information retrieval (Sections 10 to 15).

3. Related Work

This paper puts forward a new method for computing semantic relatedness, which makes use of a graph structure over Wikipedia to solve several NLP problems. Therefore, related work spans a large number of domains and approaches, and can be divided mainly into: (1) previous methods for computing semantic relatedness, including uses of Wikipedia or other networked resources, for one or more tasks in common with this paper; (2) previous algorithms for computing distances within graphs; and (3) state-of-the-art methods and scores for each of the targeted tasks. In fact, many combinations of tasks, methods and resources may share one or more elements with our proposal. This section will focus on the first two categories of previous work, while for the third one, performance comparisons with state-of-the-art methods will be made in each of the application sections. A synthetic view of previous work appears in Table 1 at the end of this section, with resources, algorithms, tasks, and data sets used for testing.

3.1. Word Semantic Relatedness: WordNet and Wikipedia

Many attempts have been made in the past to define word and text similarity distances based on word overlap, for various applications to language technology. One approach is to construct – manually or semi-automatically – a

taxonomy of word senses or of concepts, with various types of relations, and to map the text fragments to be compared onto the taxonomy. For instance, WordNet [9] and Cyc [10] are two well-known knowledge bases, respectively of word senses and concepts, which can be used for overcoming the strong limitations of pure lexical matching. A thesaurus such as Roget's can also be used for similar purposes [11, 23]. This approach makes use of explicit senses or concepts that humans can understand and reason about, but the granularity of knowledge representation is limited by the taxonomy. Building and maintaining these knowledge bases requires a lot of time and effort from experts. Moreover, they may cover only a fraction of the vocabulary of a language, and usually include few proper names, conversational words, or technical terms.

Several methods for computing lexical semantic relatedness exploit the paths in semantic networks or in WordNet, as surveyed by Budanitsky and Hirst [1, Section 2]. Distance in the network is one of the obvious criteria for similarity, which can be modulated by the type of links [24] or by local context, when applied to word sense identification [25]. Resnik [8, 26] improved over distance-based similarity by defining the information content of a concept as a measure of its specificity, and applied the measure to word sense disambiguation in short phrases. An information-theoretic definition of similarity, applicable to any entities that can be framed into a probabilistic model, was proposed by Lin [27] and was applied to word and concept similarity. This work and ours share a similar concern – the quest for a generic similarity or relatedness measure – albeit in different conceptual frameworks – probabilistic vs. hypertext encyclopedia.

Other approaches make use of unsupervised methods to construct a semantic representation of words or of documents by analyzing mainly co-occurrence relationships between words in a corpus (see e.g. Chappelier [28] for a review). Latent Semantic Analysis [19] offers a vector-space representation of words, which is grounded statistically and is applied to document representation in terms of topics using Probabilistic LSA [21] or Latent Dirichlet Allocation [22]. These unsupervised methods construct a low-dimensional feature representation, or concept space, in which words are no longer supposed to be independent. The methods offer large vocabulary coverage, but the resulting "concepts" are difficult for humans to interpret [29].

Mihalcea et al. [30] compared several knowledge-based and corpus-based methods (including for instance [25]) and then used word similarity and word specificity to define one general measure of text semantic similarity. Results of several methods and combinations are reported in their paper. Because it computes word similarity values between all word pairs, the proposed measure appears to be suitable mainly for computing similarity between short fragments – otherwise, the computation becomes quickly intractable.

One of the first methods to use a graph-based approach to compute word relatedness was proposed by Hughes and Ramage [31], using Personalized PageRank (PPR) [32] over a graph built from WordNet, with about 400,000 nodes and 5 million links. Their goal (as ours) was to exploit all possible links between two words in the graph, and not only the shortest path. They illustrated the merits of this approach on three frequently-used data sets of word pairs – which will be also used in this paper, see Section 10 – using several standard correlation metrics as well as an original one. Their method reaches "the limit of human inter-annotator agreement and is one of the strongest measures of semantic relatedness that uses only WordNet."

In recent years, Wikipedia has appeared as a promising conceptual network, in which the relative noise and incompleteness due to its collaborative origin is compensated for by its large size and a certain redundancy, along with availability and alignment in several languages. Several large semantic resources were derived from it, such as a relational knowledge base (DBpedia [33]), two concept networks (BabelNet [34] and WikiNet [35]) and an ontology derived from both Wikipedia and WordNet (Yago [36]).

WikiRelate! [37] is a method for computing semantic relatedness between two words by using Wikipedia. Each word is mapped to the corresponding Wikipedia article by using the titles. To compute relatedness, several methods are proposed, namely, using paths in the Wikipedia category structure, or using the contents of the articles. Our method, by comparison, also uses the knowledge embedded in the hyperlinks between articles, along with the entire contents of articles. Recently, the category structure exploited by WikiRelate! was also applied to computing semantic similarity between words [38]. Overall, however, WikiRelate! measures relatedness between two words and is not applicable to similarity of longer fragments, unlike our method. Another method to compute word similarity was proposed by Milne and Witten [39] using similarity of hyperlinks between Wikipedia pages.

3.2. Text Semantic Relatedness

Several studies have measured relatedness of sentences or entire texts. In a study by Syed et al. [40], Wikipedia was used as an ontology in three different ways to associate keywords or topic names to input documents: either (1) by cosine similarity retrieval of Wikipedia pages, or (2) by spreading activation through the Wikipedia categories of these pages, or (3) by spreading activation through the pages hyperlinked with them. The evaluation was first performed on three articles for which related Wikipedia pages could be validated by hand, and then on 100 Wikipedia pages, for which the task was to restore links and categories (similarly to [41]). The use of a private test set makes comparisons with other work uneasy. In another text labeling task, Coursey et al. [42] have used the entire English Wikipedia as a graph (5.8 million nodes, 65 million edges) with a version of Personalized PageRank [32] that was initialized with the Wikipedia pages found to be related to the input text using Wikify! [43]. The method was tested on a random selection of 150 Wikipedia pages, with the goal of retrieving automatically their manually-assigned categories.

Ramage et al. [44] have used Personalized PageRank over a WordNet-based graph to detect paraphrases and textual entailment. They formulated a theoretical assumption similar to ours: "the stationary distribution of the graph [random] walk forms a 'semantic signature' that can be compared to another such distribution to get a relatedness score for texts." Our proposal includes a novel method for comparing such distributions, and is applied to different tasks (we tested paraphrases in a previous paper [45]).

Explicit Semantic Analysis (ESA), proposed by Gabrilovich and Markovitch [46, 47], instead of mapping a text to a node or a small group of nodes in a taxonomy, maps the text to the entire collection of available concepts, by computing the degree of affinity of each concept to the input text. ESA uses Wikipedia articles as a collection of concepts, and maps texts to this collection of concepts using a term/document affinity matrix. Similarity is measured in the new concept space. Unlike our method, ESA does not use the link structure or other structured knowledge from Wikipedia. Our method, by walking over a content similarity graph, benefits in addition from a non-linear distance measure according to word co-occurrences.

ESA has been used as a semantic representation (sometimes with modifications) in other studies of word similarity, such as a cross-lingual experiment with several Wikipedias by Hassan and Mihalcea [48], evaluated over translated versions of English data sets (see Section 10 below). In a study by Zesch et al. [49], concept vectors akin to ESA and path length were evaluated for WordNet, Wikipedia and the Wiktionary, showing that the Wiktionary improved over previous methods. ESA also provided semantic representations for a higher-end application to cross-lingual question answering [50], and was used by Yeh et al. [51], to which we now turn.

Probably the closest antecedent to our study is the WikiWalk approach [51]. A graph of documents and hyperlinks was constructed from Wikipedia, then the Personalized PageRank (PPR) [32] was computed for each text fragment, with the teleport vector being the one resulting from ESA. A dictionary-based initialization of the PPR algorithm was studied as well. To compute semantic similarity between two texts, Yeh et al. simply compared their PPR vectors. Their scores for word similarity were slightly higher than those obtained by ESA [47], while the scores on document similarity (Lee data set, see Section 11 below) were "well below state of the art, and show that initializing the random walk with all words in the document does not characterize the documents well." By comparison, in our method, we also consider in addition to hyperlinks the effect of word co-occurrence between article contents, and use a different random walk and initialization methods. In particular, we have previously shown [45] that visiting probability improves over PPR, likely because it captures different properties of the network.

Mihalcea and Csomai [43] and Milne and Witten [41] discussed enriching a document with Wikipedia articles. Their methods can be used to add explanatory links to news stories or educational documents, and more generally to enrich any unstructured text fragment (or bag-of-words) with structured knowledge from Wikipedia. Both perform disambiguation for all n-grams, which requires a time-consuming computation of relatedness of all senses to the context articles. The first method detects linkable phrases and then associates them to the relevant article, using a probabilistic approach. The second one learns the associations and then uses the results to search for linkable phrases.

3.3. Distances between Nodes in Graphs

We now turn to abstract methods for measuring distances between vertices in a graph. Many graph-based methods have been applied to NLP problems (see for instance the proceedings of the *TextGraphs* workshops) and were recently surveyed by Navigli and Lapata [55] with an application to word sense disambiguation. A similar attempt was made by Ion and Ştefănescu [56], while Navigli [57] defined a method for truncating a graph of WordNet senses built

Article	Resource	Algorithm	Task	Data set
Jarmasz [11], Jarmasz and Szpakowicz [23]	Roget	Shortest path	Word sim.	M&C, R&G, Synonyms
Mihalcea et al. [30], corpus-based	Web / BNC	PMI-IR / LSA	Paraphrase	Microsoft
Mihalcea et al. [30], six knowledge-based	WordNet	Shortest path, IC, etc.	=	=
Hughes and Ramage [31]	WordNet	PPR	Word sim.	M&C, R&G, WS-353
Gabrilovich and Markovitch [46]	Wikipedia	ESA: TF-IDF + Cosine sim.	Word sim., Doc. sim.	WS-353, Lee
Agirre and Soroa [52]	~WordNet	PPR	WSD	Senseval-2, 3
Zesch et al. [49]	WordNet, Wikipedia, Wiktionary	Path length, concept vectors	Word sim.	M&C, R&G, WS-353 + German
Strube and Ponzetto [37]	Wikipedia	Shortest path, categories, text overlap	Word sim., coreference resolution	M&C, R&G, WS-353
Milne and Witten [39]	Wikipedia	Similarity of hyperlinks	Word sim.	M&C, R&G, WS-353
Hassan and Mihalcea [48]	Wikipedia	Modified ESA	Cross-lingual word sim.	Translated M&C, WS-353
Syed et al. [40]	Wikipedia	Cosine sim., spreading activation	Doc. classif.	3–100 handpicked docs
Coursey et al. [42]	Wikipedia	PPR	Doc. classif.	150 WP articles
Ramage et al. [44]	WordNet	PPR	Paraphrase, entailment	Microsoft, RTE
Gabrilovich and Markovitch [47]	Wikipedia	ESA: TF-IDF + Cosine sim.	Doc. clustering	Reuters, 20NG, OHSUMED, short docs
Yeh et al. [51]	Wikipedia	PPR	Word sim., Doc. sim.	M&C, WS-353, Lee
Present proposal	Wikipedia	Visiting Probability (*VP*)	Word sim., Doc. sim. and clustering, IR	See Sections 10–14.

Table 1: Comparison of the present proposal (last line) with previous work cited in this section, in terms of resources, algorithms, NLP tasks, and data sets. The abbreviations for the data sets in the rightmost column are explained in Section 10 on page 17. The methods are abbreviated as follows: ESA for Explicit Semantic Analysis [46, 47], LSA for Latent Semantic Analysis [19], IC for Information Content [8, 26], PMI-IR pointwise mutual information using data collected by information retrieval [53, 30], and PPR for the Personalized PageRank algorithm [32, 54].

from input text. Navigli and Lapata [55] focussed on measures of connectivity and centrality of a graph built on purpose from the sentences to disambiguate, and are therefore close in spirit to the ones used to analyze our large Wikipedia-based network in Section 4.3.

Two measures of node distance which have a similar goal as the visiting probability (*VP*) proposed in this paper are hitting time, a standard notion in graph theory, and Personalized PageRank (PPR) [32], surveyed by Berkhin [54]. Hitting time from vertex s_i to s_j is the number of steps a random walker takes on average to visit s_j for the first time when it starts from s_i. The difference between hitting time and visiting probability will be discussed at the end of Section 6.2 below, once our proposal is properly introduced. Hitting time has been used in several studies as a distance measure in graphs, e.g. for dimensionality reduction [58] or for collaborative filtering in a recommender system [59]. Hitting time has also been used for link prediction in social networks along with other graph-based distances [60], or for semantic query suggestion using a query/URL bipartite graph [61]. A branch and bound approximation algorithm has been proposed to compute a node neighborhood for hitting time in large graphs [62]. PageRank has been used for word sense disambiguation over a graph derived from the candidate text by Navigli and Lapata [55]. As for PPR, the measure has been used for word sense disambiguation by Agirre and Soroa [52] over a graph derived from WordNet, with up to 120,000 nodes and 650,000 edges. PPR has also been used for measuring lexical relatedness of words in a graph built from WordNet by Hughes and Ramage [31], as mentioned above.

4. Wikipedia as a Network of Concepts

4.1. Concepts = Nodes = Vertices

We built our concept network from Wikipedia by using the Freebase Wikipedia Extraction (WEX) dataset [63] (version dated 2009-06-16). Not all Wikipedia articles were considered appropriate to include in the network of concepts, for reasons related to their nature and reliability, but also to the tractability of the overall method, given the very large number of pages in the English Wikipedia. Therefore, we removed all Wikipedia articles that belonged to the following name spaces: Talk, File, Image, Template, Category, Portal, and List, because these articles do not describe concepts, but contain auxiliary media and information that do not belong into the concept network. Also, disambiguation pages were removed as well, as they only point to different meanings of the title or of its variants.

As noted by Yeh et al. [51], short articles are often not appropriate candidates to include in the concept network, for several reasons: they often describe very specific concepts which have little chances to occur in texts; they might correspond to incomplete articles (stubs); they contain an unreliable selection of hyperlinks; and their number considerably slows down computation in the network. In previous work, Yeh et al. [51] set a size limit of 2,000 non-stop words below which entries were pruned, and this limit decreased considerably the size of their network. As our goal is to minimize the risk of removing potentially useful concepts, and to respect as much as possible the original contents of Wikipedia, we set a cut-off limit of 100 non-stop words, thus pruning only very minor articles. This value is thus much lower than the value used by Yeh et al. [51], and is similar to the one used by Gabrilovich and Markovitch [47]. Out of an initial set of 4,327,482 articles in WEX, filtering removed about 70% of all articles based on namespaces and length cut-off, yielding a resulting set of 1,264,611 concepts.

Each concept has a main Wikipedia name, which is the title of the main page describing the concept. However, in many cases, other words or phrases can be used as well to refer to the concept. One such type of words can be determined by examining Wikipedia redirects, i.e. articles that have no content but point the user to a proper article with an alternative title. The titles of redirect pages were added as secondary titles to the titles of the articles they redirect to. In addition, for every hyperlink from one article to another, we extracted the corresponding anchor text and considered it as another possible secondary title for the linked article, thus capturing a significant part of the terminological variation of concept names (with some noise due to variability in linking practice). Therefore, each concept has three types of titles (see summary of data structure in Table 2): the original one, the anchor texts of the hyperlinks targeting it, and the variants provided by redirect pages, each specific title being listed only once.

4.2. Relations = Links = Edges

Relations between concepts can be determined in several ways. In a previous study [45], we considered four types of links between concepts: hyperlinks and links computed from similarity of content, of category, and of template. While each type of links captures some form of relatedness, we focus in the present study on the first two types, which are the most complementary. However, the proposed computational framework is general enough to accommodate more than two types of links, in particular if an optimal combination can be learned from training data.

The use of *hyperlinks between Wikipedia articles* embodies the somewhat evident observation that every hyperlink from the content of an article towards another one indicates a certain relation between the two articles. These are encyclopedic or pragmatic relations, i.e. between concepts in the world, and subsume semantic relatedness. In other words, if article A contains a hyperlink towards article B, then B helps to understand A, and B is considered to be related to A. Such links represent a substantial amount of human knowledge that is embodied in the Wikipedia structure. It must be noted that these links are essentially asymmetric, and we decided to keep them as such, i.e. to list for a given page only its outgoing links and not the incoming ones. Indeed, observations showed that while the target page of a link helps understanding the source one, the contrary is not always true or the relation is not specific enough. For each article, the XML text from WEX was parsed to extract hyperlinks, resulting in a total of 35,214,537 hyperlinks – a time-consuming operation that required also the ability to handle instances of ill-formed XML input.

The second type of links is based on *similarity of lexical content between articles* of Wikipedia, computed from word co-occurrence. If two articles have many words in common, then a topic-similarity relation holds between them. To capture content similarity, we computed the lexical similarity between articles as the cosine similarity between the vectors derived from the articles' texts, after stopword removal and stemming using Snowball.[1] We then linked every

[1] From the Apache Lucene indexing system available at http://lucene.apache.org/.

article to its k most similar articles, with a weight according to the normalized lexical similarity score (for non-zero weights). In the experiments described below, k was set to 10, so that each node has ten outgoing links to other nodes, based on lexical similarity.[2] The value of $k = 10$ was chosen to ensure computational tractability, and is slightly lower than the average number of hyperlinks per concept, which is about 28. As the Wikipedia articles are scattered in the space of words, tuning k does not seem to bring crucial changes. If k is very small then the neighborhood contains little information, whereas a large k makes computation time-consuming.

Concept ID	Integer
Names of the concept	Name of article in the encyclopedia
	Alternative names redirecting to the article
	Anchor texts of incoming hyperlinks
Description of the concept	Text
Relations to other concepts (outgoing links)	Hyperlinks from the description towards other concepts (no weights)
	Lexical similarity links to the ten closest concepts (weights from cosine similarity)

Table 2: Logical structure of each node in the network resulting from the English Wikipedia.

4.3. Properties of the Resulting Network

The processing of the English Wikipedia resulted in a very large network of concepts, each of them having the logical structure represented in Table 2. The network has more than 1.2 million nodes (i.e. vertices), with an average of 28 outgoing hyperlinks per node and 10 outgoing content links per node.

A natural question arising at this point is: how can the structure of the network be characterized, apart from putting it to work? It is not possible to visualize the entire network due to its size, and displaying only a small part, as done for instance by Coursey et al. [42, Figure 1], might not be representative of the entire network.[3] A number of quantitative parameters have been proposed in graph theory and social network analysis, and some have for instance been used to analyze WordNet (and an enhanced version of it) by Navigli and Lapata [55]. We compute below some well-known parameters for our network, and add a new, more informative characterization.

A first characteristic of graphs is their degree distribution, i.e. the distribution of the number of direct neighbors per node. For the original Wikipedia with hyperlinks, a representation [64] suggests that the distribution follows a power law. A more relevant property here is the network clustering coefficient, which is the average of clustering coefficients per node, defined as the size of the immediate neighborhood of the node divided by the maximum number of links that could connect all pairs of neighbors [65]. For our hyperlink graph, the value of this coefficient is 0.16, while for the content link graph it is 0.26.[4] This shows that the hyperlink graph is less clustered than the content link one, i.e. the distribution of nodes and links is more homogeneous, and that overall the two graphs have rather low clustering.[5]

We propose an ad-hoc measure offering a better illustration of the network's topology, aimed at finding out whether the graph is clustered or not – i.e., whether the communities of nodes based on neighborhoods have a preferred size, or are uniformly distributed. We consider a sample of 1000 nodes. For each node of the graph, the Personalized PageRank algorithm [32] is initialized from that node and run, thus resulting into a proximity coefficient for each node in the graph to the initial node. This is first done using hyperlinks, and then using the content links. The

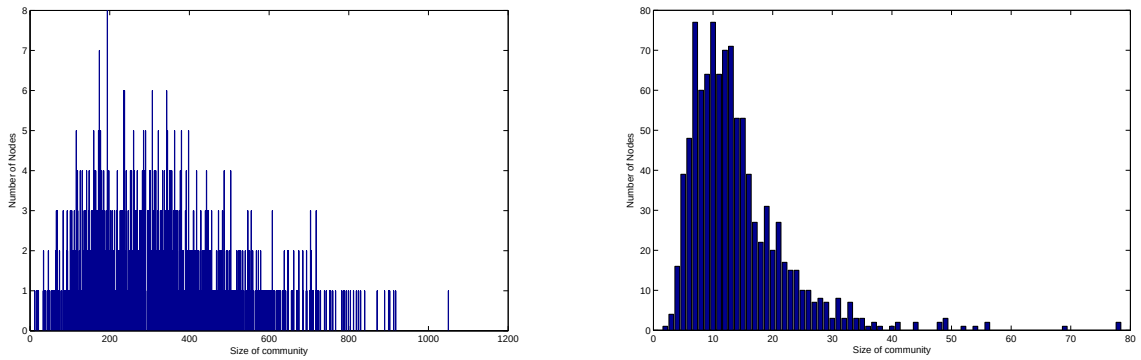
[2] Therefore, the outdegree of all nodes is 10, but as the indegree may vary (the number of incoming links), the graph is not strictly speaking a 10-regular graph.

[3] An example of neighborhood according to our relatedness measure is however shown in Section 6.3.

[4] For the content links, the coefficient was computed regardless of their weights. A recent proposal for computing it for weighted graphs could be applied too [66].

[5] The observed values, together with the power law degree distribution, suggest that our graph is a scale-free network – characterized by the presence of "hub" nodes – or a small-world network [65]. However, it is not clear what impact these properties defined mainly for social networks would have here.

community size for the node is computed by sorting all nodes with respect to their proximity and counting how many nodes contribute to 99% of the mass.



(a) Hyperlink graph

(b) Content link graph

Figure 1: Distribution of community sizes for a sample of 1000 nodes (see text for the definition of a community). For each community size (x-axis) the graphs show the number of nodes (y-axis) having a community of that size, for each of the two graphs built from Wikipedia. Both graphs have a tendency towards clustering, i.e. a non-uniform distribution of links, with an average cluster size of 150–400 for hyperlinks and 7–14 for content links.

The results shown in Figure 1 show that the distribution is neither flat nor uniformly decreasing, but has a peak, which provides an indication of the average size of clusters. This size is around 150–400 nodes for the hyperlink graph, without a sharp maximum, showing less clustering than for content links, for which this average is around 7–14 nodes. The latter value is partly related to the 10-link limit set for content links, but is not entirely due to it, as the limit concerns only outgoing links. The use of hyperlinks thus avoids local clusters and extends considerably the connectivity of the network in comparison to content similarity ones.

5. Mapping Text Fragments to Concepts in the Network

In order to use the concept network for similarity judgments between two text fragments, two operations are necessary, as explained at the end of Section 2.2. The first one, mapping each text fragment to a set of vertices of the network, is explained in this section, while Sections 6 and 7 define the distance between two weighted sets of vertices. For mapping, two cases must be considered, according to whether the text matches exactly the title of a Wikipedia page or not. Exact matching is likely to occur with individual words or short phrases, but not with entire sentences or longer texts.

If a text fragment consists of a single word or a phrase that *matches exactly the title of a Wikipedia page*, then it is simply mapped to that concept. In the case of words or phrases that may refer to several concepts in Wikipedia, we simply assign to them the same page as the one assigned by the Wikipedia contributors as the most salient or preferred sense or denotation. For instance, 'mouse' directs to the page about the animal, which contains an indication that the 'mouse_(computing)' page describes the pointing device, and that other senses are listed on the 'mouse_(disambiguation)' page. So, here, we simply map 'mouse' to the animal concept. However, for other words, no sense or denotation is preferred by the Wikipedia contributors, e.g. for the word 'plate'. In such cases, a disambiguation page is associated to that word or phrase. We chose not to include such pages in our network, as they do not correspond to individual concepts. So, in order to select the referent page for such words, we simply use the lexical similarity approach described in the next paragraph.

When a fragment (a word, phrase, sentence, or text) *does not match exactly the Wikipedia title of a vertex in our network*, it is mapped to the network by computing its lexical similarity with the text content of the vertices in the network, using cosine distance over stemmed words, stopwords being removed. Concept names (principal and

secondary ones) are given twice as much weight as the words found in the description of the concept. The text fragment is mapped to the k most similar articles according to this similarity score, resulting in a set of at most k weighted concepts. The weights are normalized, summing up to one, therefore the text representation in the network is a *probability distribution over at most k concepts*. Finding the k closest articles according to lexical similarity can be done efficiently using the Apache Lucene search engine (see note 1).

For example, consider the following text fragment to be mapped to our network: "Facebook: you have some serious privacy and security problems." When this fragment is mapped to the $k = 10$ most similar Wikipedia articles, the resulting probability distribution is the following one: 'Facebook' (0.180), 'Facebook Beacon' (0.119), 'Facebook history' (0.116), 'Criticism of Facebook' (0.116), 'Facebook features' (0.116), 'Privacy' (0.084), 'Privacy International' (0.080), 'Internet privacy' (0.080), 'Privacy policy' (0.054), 'Privacy Lost' (0.054).

This mapping algorithm has an important role in the performance of the final system, in combination with the network distance described hereafter. It must however be noted that the effects of wrong mappings at this stage are countered later on. For instance, when large sets of concepts related to two text fragments are compared, a few individual mistakes are not likely to alter the overall relatedness scores. Alternatively, when comparing individual words, wrong mappings are less likely to occur because the test sets for word similarity described in Section 10, page 17, also consider implicitly the most salient sense of each word, just as described above for Wikipedia.

6. Semantic Relatedness using Visiting Probability

In this section, we describe our framework for computing relatedness between two texts, represented as sets of concepts in a network. Although this is applied here to Wikipedia, the model is general enough to be applied to other networks too. The goal is to estimate a distance between two concepts or two sets of concepts that captures their relatedness by taking into account the global connectivity of the network, and without being biased by local properties. Indeed, the use of individual links and paths, e.g. when estimating conceptual relatedness as the length of shortest path, does not take into account their relative importance with respect to the overall properties of the network, such as the number and length of all possible paths between two nodes. Moreover, the length of the shortest path is quite sensitive to spurious links, which are frequent in Wikipedia. Therefore, a number of aggregated proximity measures have been proposed, including PageRank and hitting time, as reviewed in Section 3 above.

Our proposal uses a random walk approach, and defines the relatedness of concept sets as the *visiting probability*, in the long run, of a random walker going from one set of nodes to the other one. The walker can use either type of directed links between concepts described in Section 4.2, i.e. hyperlinks or lexical similarity ones. This algorithm is independent from the mapping algorithm introduced in the previous section.

6.1. Notations

Let $S = \{s_i | 1 \leq i \leq n\}$ be the set of n concepts or vertices in the network. Any two concepts s_i and s_j can be connected by one or more directed and weighted links, which can be of L different types ($L = 2$ in our case: hyperlinks and lexical similarity links). The structure of links of type l ($1 \leq l \leq L$) is fully described by the matrix A_l of size $n \times n$, where $A_l(i, j)$ is the weight of the link of type l between s_i and s_j, with possible weights being 0 or 1 for the hyperlink matrix, and actual lexical similarity scores (or 0) for the content similarity matrix. The transition matrix C_l gives the probability of a direct (one step) transition between concepts s_i and s_j, using only links of type l. This matrix can be built from the A_l matrix as follows:

$$C_l(i, j) = \frac{A_l(i, j)}{\sum_{k=1}^{n} A_l(i, k)}$$

In the random walk process using all link types ($1 \leq l \leq L$), let the weight w_l denote the importance of link type l. Then, the overall transition matrix C which gives the transition probability $C_{i,j}$ between any concepts s_i and s_j is $C = \sum_{l=1}^{L} w_l C_l$.

Finally, let $\vec{r}$ be the n-dimensional vector resulting from mapping a text fragment to the concepts of the network (Section 5), which indicates the probabilities of concepts in the network given a text, or the relevance of concepts to the text's words. The sum of its elements is 1.

6.2. Definition of Visiting Probability

Given a probability distribution $\vec{r}$ over concepts (i.e. a weighted set of concepts), and a concept s_j in the network, we first compute the probability of visiting s_j for the first time when a random walker starts from $\vec{r}$ in the network, i.e. from any of the concepts in $\vec{r}$ that have a non-zero probability.[6] To compute visiting probability, the following procedure provides a model of the state S_t of the random walker, i.e. the concept in which the random walker is positioned. Executing the procedure until termination gives the visiting probability *VP*.

Step 0: Choose the initial state of the walker with probability $P(S_0 = s_i|\vec{r}) = r_i$. In other words, position the random walker on any of the concepts of $\vec{r}$ with the probability stated in $\vec{r}$.

Step t: Assuming S_{t-1} is determined, if $S_{t-1} = s_j$ then return 'success' and finish the procedure. Otherwise, with probability α, choose a value for the next concept S_t according to the transition matrix C, and with probability $1 - \alpha$, return 'fail'. The possibility of 'failing', or absorption probability, introduces a penalty over long paths that makes them less probable.

We introduce C' as being equal to the transition matrix C, except that in row j, $C'(j, k) = 0$ for all k. This indicates the fact that when the random walker visits s_j for the first time, it can not exit from it and its probability mass drops to zero in the next step. This modified transition matrix was defined to account for the definition of visiting probability as the probability of *first* visit of s_j (a similar idea has been proposed by Sarkar and Moore [62]).

To compute the probability of success in the above process, we introduce the probability of success at step t, $p^t(success)$, which is $p^t(success) = \alpha^t(\vec{r}C'^t)_j$. In this formula, $(\vec{r}C'^t)_j$ is the j^{th} element of the vector $\vec{r}C'^t$, and C'^t is the power t of matrix C'. Then, the probability of success in the process, i.e. the probability of visiting s_j starting from $\vec{r}$, is the sum over all probabilities of success with different lengths:

$$p(success) = \sum_{t=0}^{\infty} p^t(success) = \sum_{t=0}^{\infty} \alpha^t(\vec{r}C'^t)_j.$$

The visiting probability of s_j starting from the distribution $\vec{r}$, as computed above, is different from the probability assigned to s_j after running Personalized PageRank (PPR) with a teleport vector equal to $\vec{r}$. In the computation of *VP*, the loops starting from s_j and ending to s_j do not have any effect on the final score, unlike the computation of PPR, for which such loops boost the probability of s_j. If some pages have this type of loops (typically "popular" pages), then after using PPR they will have high probability although they might not be very close to the teleport vector $\vec{r}$.

The visiting probability of s_j is also different from the hitting time to s_j, defined as the average number of steps a random walker would take to visit s_j for the first time in the graph. If we use the same notations, the hitting time from $\vec{r}$ to s_j is $H(\vec{r}, s_j) = \sum_{t=0}^{\infty} t(\vec{r}C'^t)_j$. Hitting time is more sensitive to long paths in comparison to *VP* (t in comparison with α^t in the formula), which might introduce more noise, while *VP* reduces the effect of long paths sooner in the walk. We have compared the performance of these three algorithms in a previous paper [45] and concluded that *VP* outperformed PPR and hitting time, which will not be used here.

6.3. Visiting Probability to Weighted Sets of Concepts and to Texts

To compute the *VP* from a weighted set of concepts to another set, i.e. from distribution $\vec{r}_1$ to distribution $\vec{r}_2$, we construct a virtual node representing $\vec{r}_2$ in the network, noted $s_R(\vec{r}_2)$. We then connect all concepts s_i to $s_R(\vec{r}_2)$ according to the weights in $\vec{r}_2$. We now create the transition matrix C' by adding a new row (numbered n_R) to the transition matrix C with all elements zero to indicate $s_R(\vec{r}_2)$, then adding a new column with the weights in $s_R(\vec{r}_2)$, and updating all the other rows of C as follows (C_{ij} is an element of C):

$$\begin{aligned} C'_{ij} &= C_{ij}(1 - (\vec{r}_2)_i) \text{ for } i, j \neq n_R \\ C'_{in_R} &= (\vec{r}_2)_i \text{ for all } i \\ C'_{n_R j} &= 0 \text{ for all } j. \end{aligned}$$

[6] A simpler derivation could first be given for the visiting probability to s_j starting from another concept s_i, but to avoid duplication of equations, we provide directly the formula for a weighted set of concepts.

These modifications to the graph are local and can be done at run time, with the possibility to undo them for subsequent text fragments to be compared.

To compute relatedness between two texts, we average between the visiting probability of $\vec{r}_1$ given $\vec{r}_2$ (noted $VP(\vec{r}_2, \vec{r}_1)$) and the visiting probability of $\vec{r}_2$ given $\vec{r}_1$ (noted $VP(\vec{r}_1, \vec{r}_2)$). A larger value of the measure indicates closer relatedness. It is worth mentioning that $(2 - (VP(\vec{r}_1, \vec{r}_2) + VP(\vec{r}_2, \vec{r}_1))$ is a metric distance, as it satisfies four properties: non-negativity, identity of indiscernible, symmetry, and triangle inequality, as it can be shown using the definition of *VP*.

6.4. Illustration of Visiting Probability 'to' *vs.* 'from' *a Text*

To illustrate the difference between *VP to* and *from* a text fragment, we consider the following example. Though any text fragment could be used, we took the definition of 'jaguar' (animal) from the corresponding Wikipedia article. Although the word 'jaguar' is polysemous, the topic of this particular text fragment is not ambiguous to a human reader. The text was first mapped to Wikipedia as described above. Then, using *VP* with equal weights on hyperlinks and content links, the ten closest concepts (i.e. Wikipedia pages) in terms of *VP* from the text, and respectively to it, were found to be the following ones:

- Original text: "The jaguar is a big cat, a feline in the Panthera genus, and is the only Panthera species found in the Americas. The jaguar is the third-largest feline after the tiger and the lion, and the largest in the Western Hemisphere."
- Ten closest concepts according to their *VP to* the text: 'John Varty', 'European Jaguar', 'Congolese Spotted Lion', 'Lionhead rabbit', 'Panthera leo fossilis', 'Tigon', 'Panthera hybrid', 'Parc des Félins', 'Marozi', 'Craig Busch'.
- Ten closest concepts according to their *VP from* the text: 'Felidae', 'Kodkod', 'North Africa', 'Jaguar', 'Panthera', 'Algeria', 'Tiger', 'Lion', 'Panthera hybrid', 'Djémila'.

The closest concepts according to *VP from* a text tend to be more general than closest concepts according to *VP to* the text. For instance, the second list above includes the genus and family of the jaguar species (Panthera and Felidae), while the first one includes six hybrid or extinct species related to the jaguar. Concepts close *to* a text thus bring detailed information related to the topics of the text, while concepts close *from* a text are more popular and more general Wikipedia articles. Note also that none of the closest concepts above is related to the Jaguar car brand, as found by examining each page, including the lesser-known ones (both persons cited are wildlife film makers and park founders). However, not all concepts are related in an obvious way (e.g. 'Algeria' is likely retrieved through 'Western Hemisphere').

The behavior illustrated on this example is expected given the definition of *VP*. A concept that is close *to* a text has more paths in the network towards the text with respect to other concepts in the network, which means that the concept is quite specific in relation to the topics in the text, as in the above example. Conversely, a concept that is close *from* a text (in terms of *VP* from a text) is typically related by many paths from the text to the concept, in comparison to other concepts. This generally means that it is likely that the article is a general and popular article around the topics of the text. If we consider the hierarchy between concepts from more general to more specific ones, then concepts close *to* a text are generally lower in the hierarchy with respect to the text, and concepts close *from* a text are generally higher.

7. Approximations: T-Truncated and ε-Truncated Visiting Probability

The above definition of the random walk procedure has one direct consequence: computation can be done iteratively and can be truncated after T steps when needed, especially to maintain computation time within acceptable limits. Truncation makes sense as higher order terms (for longer paths) get smaller with larger values of t because of the α^t factor. Moreover, besides making computation tractable, truncation reduces the effect of longer (hence less reliable) paths on the computed value of $p(success)$. Indeed, *VP* to popular vertices (i.e. to which many links point) might be quite high even when they are not really close to the starting distribution $\vec{r}$, due to the high number of long paths toward them, and truncation conveniently reduces the importance of such vertices. We propose in this section two methods for truncating *VP*, and justify empirically the level of approximation chosen for our tasks.

T-truncated Visiting Probability. The simplest method, called T-truncated *VP*, truncates the computation for all the paths longer than T. To compute an upper bound on the error of this truncation, we start by considering the possible state of the random walk at time T: either a success, or a failure, or a particular node. The sum of the probabilities of these three states equals 1, while the third value, i.e. the probability of returning neither success nor failure in the first t steps, can be computed as $\sum_{i\neq j}^{n} \alpha^t(\vec{r}C'^t)_i$ – this is in fact the probability mass at time t at all nodes except s_j, the targeted node. If $p_T(success)$ denotes the probability of success considering paths of length at most T, and ε_T the error made by truncating after step T, then by replacing $p_T(success)$ with the value we just computed, and noting that $p(success) + p_T(failure) \leq 1$, we obtain the following upper bound for ε_T:

$$\varepsilon_T = p(success) - p_T(success) \leq \sum_{i\neq j}^{n} \alpha^T(\vec{r}C'^T)_i.$$

So, if $p_T(success)$ is used as an approximation for $p(success)$ then an upper bound for this approximation error ε_T is the right term of the above inequality. This term decreases over time because α^T and $\sum_{i\neq j}^{n}(rC'^T)_i$ are both decreasing over time, therefore ε_T decreases when T increases.

ε-truncated Visiting Probability. A second approach, referred to as ε-truncated *VP*, truncates paths with lower probabilities in earlier steps and lets paths with higher probabilities continue more steps. Given that the probability of being at s_i at time step t is $\alpha^t(rC'^t)_i$, if this is neglected and set to zero, then the error caused by this approximation is $\alpha^t(rC'^t)_i$. Setting this term to zero means exactly that paths that are at s_i at time step t are no longer followed afterwards. So, in ε-truncation, paths with a probability less than ε are not followed, i.e. when $\alpha^t(rC'^t)_i \leq \varepsilon$. This approach is faster to compute than the previous one, but no upper bound of the error could be established. We use the ε-truncated *VP* in some of our experiments below, leading to competitive results in an acceptable computation time.

T-Truncated Visiting Probability from all Vertices to a Vertex. Given a dataset of N documents, some tasks such as document clustering (Section 12) require to compute the average *VP* between all N documents. Similarly, for information retrieval (Section 14), it is necessary to sort all the documents in a repository according to their relatedness to a given query. It is not tractable to compute exact *VP* values for all concepts, but we provide here a way to compute T-truncated *VP* from/to a distribution $\vec{r}$ for all concepts in the network, which is faster than repeating the computation for each concept in part.

As in Section 6.3 above, we consider the virtual concept $s_R(\vec{r})$ representing $\vec{r}$ in the network (noted simply s_R). It is then possible to compute *VP* from all concepts towards s_R at the same time, using the following recursive procedure to compute the T-truncated visiting probability VP^T. This procedure follows from the recursive definition of *VP* given in Section 6.2, which states that $VP(s_i, s_R) = \alpha \sum_k C'(s_i, s_k)VP(s_k, s_R)$. Therefore,

$$\begin{aligned}
&VP^T(s_i, s_R) = \alpha \textstyle\sum_k C'(s_i, s_k)VP^{T-1}(s_k, s_R) \text{ for } i \neq n_R\\
&VP^T(s_i, s_R) = 1 \text{ for } i = n_R\\
&VP^0(s_i, s_R) = 0.
\end{aligned}$$

Using dynamic programming, it is possible to compute T-truncated *VP* from all concepts to s_R in $O(ET)$ steps, where E is the number of edges of the network.

T-Truncated Visiting Probability from a Vertex to all Vertices. Conversely, to compute VP^T from $\vec{r}$ to all concepts in the network, the total computation time is $O(NET)$, where N is the number of concepts and E the number of edges, because VP^T must be computed for each concept. With our current network, this is very time consuming: therefore, we propose the following sampling method to approximate T-truncated *VP*. The sampling involves running M independent T-length random walks from $\vec{r}$. To approximate VP^T to concept s_j from $\vec{r}$, if s_j has been visited for the first time at $\{t_{k_1}, \cdots, t_{k_m}\}$ time steps in the M samples, then the T-truncated *VP* to s_j can be approximated by the following average: $\hat{VP^T}(\vec{r}, s_j) = (\sum_l \alpha^{t_{k_l}})/M$.

According to the proposed method, it is possible to approximate truncated *VP* from $\vec{r}$ to all concepts in the network in $O(MT)$ time, where M is number of samples. It remains to find out how many samples should be used to obtain the desired approximation level, a question that is answered by the following theorem. For any vertex, the estimated truncated *VP* approximates the exact truncated VP^T to that vertex within ε with a probability larger than $1 - \delta$ if the number of samples M is larger than $\frac{\alpha^2 \ln(2n/\delta)}{2\varepsilon^2}$. The proof of this result is given in Appendix A.

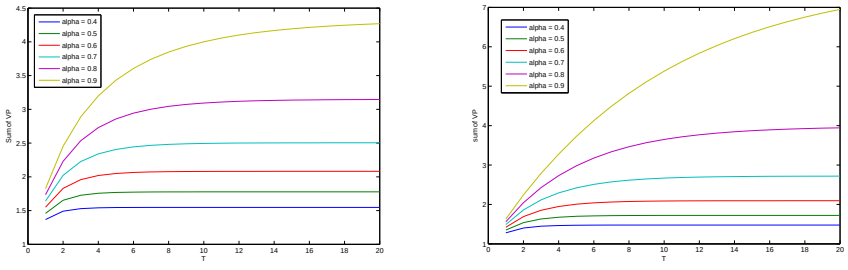
8. Empirical Analyses of *VP* and Approximations

We now analyze the convergence of the two approximation methods proposed above, i.e. the variation of the margin of error with T or ε. In the case of T-truncated approximation, when T increases to infinity, the T-truncated approximated *VP* converges towards the exact value, but computation time increases linearly with T. In the case of ε-truncation, when ε tends to zero the approximated value converges towards the real one, but again computation time increases. Therefore, we need to find the proper values for T and ε as a compromise between the estimated error and the computing time.

8.1. Convergence of the T-Truncated VP over Wikipedia

The value of T required for a certain level of convergence (upper bound on the approximation error) depends on the transition matrix of the graph. It was not possible to find a convenient theoretical upper bound, so we analyzed the relation between T and the approximation error empirically, by a sampling method. We chose randomly a set S of 1000 nodes in the graph, and computed the T-truncated *VP* from all nodes in the graph to each of the nodes in S. Then we computed the average of the values of T-truncated *VP* from all nodes to the nodes in S: $VP_{sum}(T) = \sum_{j \in S} \sum_{i \neq j} VP^T(i, j)/|S|$.

Given that S is a large random sample, we consider that the evolution of $VP_{sum}(T)$ with T is representative of the evolution of an "average" VP^T with T. Figure 2(a) shows the values of $VP_{sum}(T)$ depending on T for the Wikipedia graph with the lexical similarity (content) links, for various values of α. Figure 2(b) shows the same curves for the Wikipedia graph with hyperlinks. Both analyses show, as expected, that larger values of α correspond to a slower convergence in terms of T, because a larger α requires the random walker to explore longer paths for the same level of approximation. (The exact value toward which $VP_{sum}(T)$ converges is not important here.) The figures give some indication, for each given α, of the extent to which the paths should be explored for an acceptable approximation of the non-truncated *VP* value. In our experiments, we chose $\alpha = 0.8$ and $T = 10$ as a compromise between computation time and accuracy – the approximation error is less than 10% in Figures 2(a) and 2(b).



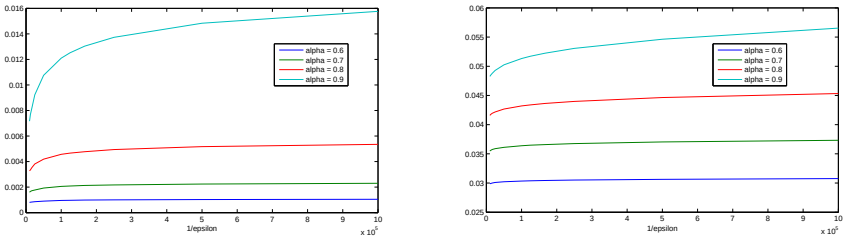
(a) Content link graph

(b) Hyperlink graph

Figure 2: $VP_{sum}(T)$ as an "average *VP*" for content links (a) and hyperlinks (b) over the Wikipedia graph, depending on T, for α varying from 0.4 to 0.9 (by 0.1, bottom to top). The shape of the curves indicates the values of T leading to an acceptable approximation of the exact, non-truncated value: $\alpha = 0.8$ and $T = 10$ were chosen in the subsequent experiments.

8.2. Convergence of ε-Truncated VP over Wikipedia

To analyze the error induced by ε-truncation when computing *VP* over the Wikipedia graph, we proceeded in a similar way. We chose 5000 random pairs of nodes and computed the sum of ε-truncated *VP* between each pair. Figure 3(a) shows the values of the sum depending on $1/\varepsilon$ for the Wikipedia graph with content links, and Figure 3(b) for the Wikipedia graph with the hyperlinks. Again, it appears from the curves that for a larger α, smaller values of ε are needed to reach the same level of approximation error, because for a larger α longer paths must be explored to reach the same approximation level. The ε-truncation is used for word similarity and document similarity below, with a value of $\varepsilon = 10^{-5}$, and a value of $\alpha = 0.8$ as above.



(a) Content link graph

(b) Hyperlink graph

Figure 3: Sum of ε-Truncated *VP* depending on $1/\varepsilon$ for the Wikipedia graph with content links (a) or hyperlinks (b), for α from 0.6 to 0.9 (bottom to top). The shape of the curves indicates the values of ε leading to an acceptable approximation of the non-truncated value: $\varepsilon = 10^{-5}$ and $\alpha = 0.8$ were chosen for subsequent experiments.

8.3. Differences between the Random Walk Model over Content Links and Direct Lexical Similarity between Articles

Performing a random walk over the Wikipedia graph leads to a relatedness measure that is different from direct lexical similarity (i.e. cosine similarity in the space of words), as we empirically show here through the following experiment. We randomly chose 1000 nodes and sorted all other nodes (Wikipedia articles) based on the cosine similarity with the randomly selected nodes, measured using TF-IDF vectors. In parallel, for each of the randomly selected nodes, all other nodes were also sorted based on the average T-truncated *VP* with $T = 10$.

The comparison of the two resulting sorted lists for each node shows the difference between the two measures. To perform this comparison, for each node we look at the intersection of the heads of the two sorted lists (neighborhoods of the node), varying the size of these subsets. Figure 4 shows the percentage of nodes in common depending on neighborhood sizes, for different values of α (which changes little to the results). It appears that for the closest neighbors, and in particular for *the* closest one, both lexical similarity and *VP* over content links return the same nodes. However, when expanding the size of the neighborhood, the size of the intersection decreases rapidly. For example, when looking at the ten closest nodes, only about 50% of the nodes on average are the same in the two lists. This is an empirical argument showing that walking over content links leads to a different relatedness measure than simply using lexical similarity between articles.

9. Learning Embeddings from Visiting Probabilities

In our approach to computing semantic relatedness, each text is mapped to a fixed number of concepts, and the distance between them is computed using *VP* at runtime. To make computation tractable, we have introduced a number of approximations in Section 7. In this section, we put forward a more principled approach, which learns from *VP* values a similarity measure between text vectors, then applies it with a much lower cost at run time. This approach does not require a fixed number of nodes to project a document to. Moreover, it can be integrated as prior knowledge to other learning algorithms for NLP and can be applied to very large scale problems.

At training time, given a series of samples – that is, pairs of texts with *VP* values from the first text to the second one – the goal is to learn a transformation from the space of words to a latent space, so that the similarity between the latent representation of the texts is as close as possible to the *VP* similarity. In other words, the goal is to approximate *VP* between two texts i and j by the matrix product $x_i A B' x'_j$, where x_i and x_j are the TF-IDF vectors of the two texts constructed from their words using a fixed dictionary. The size of matrices A and B is $n \times m$, with n being the size of the dictionary (number of words) and m the size of the latent space (akin to the number of topics in topic models). Two different matrices A and B are needed because *VP* values are not symmetric in i and j.

In principle, all pairs of Wikipedia articles (i.e., texts) corresponding to nodes in our network can be used for training, but this set is extremely large (ca. 1.4×10^{12}). Therefore, we formulate the following constraints for training: (1) training should focus on neighboring articles (articles with high *VP* values), and (2) the exact values of *VP* are

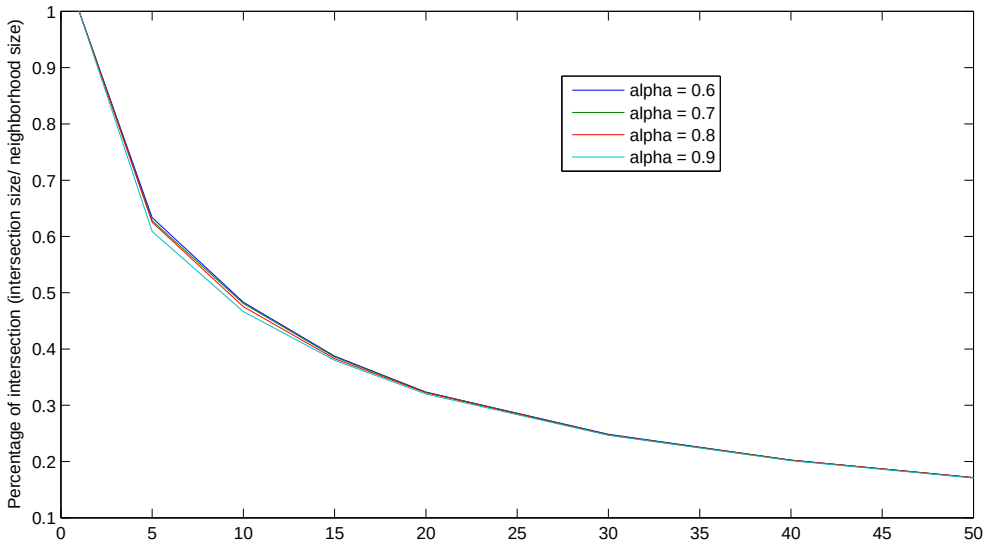


Figure 4: Average proportion of identical nodes among K closest neighbors using cosine similarity vs. average T-truncated VP, for varying sizes K of the neighborhood and various values of α (which have little influence on the result). The proportion decreases for larger neighborhoods, i.e. the two metrics give quite different results for smaller relatedness values.

replaced with the ranking of pairs of articles by decreasing VP. We show here that under these constraints valuable embeddings can be learned.

Let $VPto_k(i)$ be the set of the k closest articles *to* the article i according to VP similarity. We define a hinge loss function L as follows, so that the similarity between i and its k closest articles is larger than the similarity to all other articles by a fixed margin M.

$$L = \sum_{i \in \mathrm{WP}} \sum_{j \in VPto_k(i)} \sum_{z \notin VPto_k(i)} max(0, M - x_i AB' x'_j + x_i AB' x'_z)$$

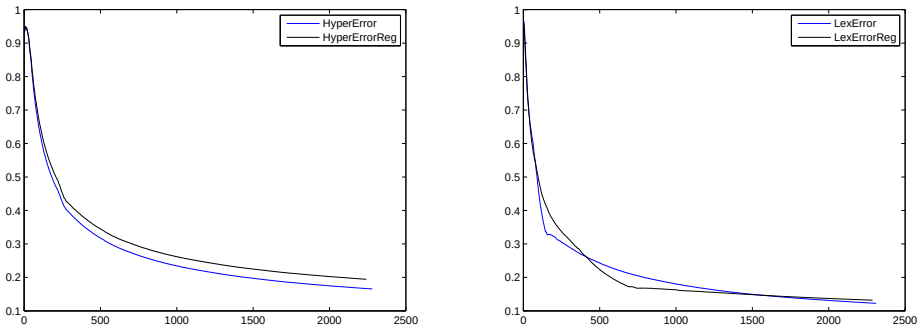
We optimize L with stochastic gradient descent: in each iteration we randomly choose one article i, then randomly choose one of the k closest articles to i (noted j) and one other article from the rest of documents (noted z).

In our experiments, we set $k = 10$ and $M = 0.2$. We computed the $VPto_k(i)$ values for all i using the approximations in Section 7 above. We built the dictionary by using the Snowball tokenizer from Lucene and removing the highest and lowest frequency words, keeping around 60,000 words. We set the number of topics to $m = 50$, because a larger m offers a higher learning capability, but also increases linearly the number of parameters to learn. Given the size of the training set, we chose a rather small number m to make the training possible in a reasonable time, and found a satisfactory prediction error.

Moreover, to perform regularization over matrices A and B when optimizing L, we impose the constraint that A and B are orthonormal. In order to apply this constraint, we project at every 1000 iterations both A and B to their nearest orthogonal matrix found by using SVD decomposition. The rationale for the constraint is the following: if we assume that each latent dimension corresponds to a possible topic or theme, then these should be as orthogonal as possible.

Figures 5(a) and 5(b) show the average training error at every 1000 iterations, with and without regularization, respectively for the hyperlink graph and for the content link one. The training error is computed as the number of text triples (i, j, z) for which the test in the hinge loss function is false, i.e. $x_i AB' x'_j - x_i AB' x'_z < M$.

To test the predictive power of the embeddings as a replacement for the computation of VP at runtime, we excluded 50,000 articles from the training set, and then built 50,000 triples of articles by choosing randomly, for each of the excluded articles, one article from the k closest ones and one randomly from the rest of the articles. The test set error, which is the number of triples from this set that do not respect the order given by VP similarities, is shown in Table 3.



(a) Hyperlink graph

(b) Content link graph

Figure 5: Average training error for learning embeddings, measured every 1000 iterations, with and without using regularization.

Training set for embedding	**Error (%)**
Hyperlinks	9.8
Hyperlinks with Regularization	13.5
Content Links	5.4
Content Links with Regularization	5.9

Table 3: Accuracy of embeddings learned over four different training sets. The error (percentage) is computed for 50,000 triples of articles from a separate test set, as the number of triples that do not respect the ordering of *VP* similarities.

The two main findings are the following. First, *VP* over the hyperlinks graph is harder to learn, which may be due to the fact that hyperlinks are defined by users in a manner that is not totally predictable. Second, regularization decreases the prediction ability. However, if regularization traded prediction power for more generality, in other words if it reduced overfitting to this problem and made the distance more general, then it would still constitute a useful operation. This will be checked in the experiments in Sections 12, 13 and 15. Moreover, these experiments will show that the embeddings can be plugged into state-of-the-art learning algorithms as prior knowledge, improving their performance.

10. Word Similarity

In the following sections, we assess the effectiveness of visiting probability by applying it to four language processing tasks. In particular, for each task, we examine each type of link separately and then compare the results with those obtained for combinations of links, in the attempt to single out the optimal combinations.

The word similarity task has been heavily researched using a variety of methods and resources – such as WordNet, Roget's Thesaurus, the English Wikipedia or Wiktionary – starting for instance with Resnik's seminal paper [8] and even earlier. We have reviewed in Section 3 some of the recent work on word similarity, focusing mainly on graph-based methods over Wikipedia but also WordNet. Recent studies, including also references to previous scores and several baselines, have been made by Bollegala et al. [67], Gabrilovich and Markovitch [46] (see also [47, Tables 2 and 3]), Zesch et al. [49], Agirre et al. [68], and Ramage et al. [44], among others.

Three test sets for the English word similarity task have been extensively used in the past. They consist of pairs of words accompanied by average similarity scores assigned by human subjects to each pair. Depending on the instructions given to the subjects, the notion of 'similarity' was sometimes rather interpreted as 'relatedness', as discussed for instance by Hughes and Ramage [31, Section 5]. The three sets, all of them reproduced in Jarmasz's thesis [11] for instance, were designed respectively by Rubenstein and Goodenough [69] (henceforth, R&G, 65 pairs,

51 judges), by Miller and Charles [70] (M&C, 30 pairs), and by Finkelstein et al. [71] (WordSimilarity-353, with 353 pairs).

We estimate the relatedness between words by mapping them to concepts and computing the ε-truncated *VP* distance between them with $\varepsilon = 10^{-5}$. We set the value of $\alpha = 0.8$ as explained in Section 8.2 above. The correlation with human judgments of relatedness is measured using the Spearman rank correlation coefficient ρ as well as the Pearson correlation coefficient r between the *VP* values for each pair and the human judgments.

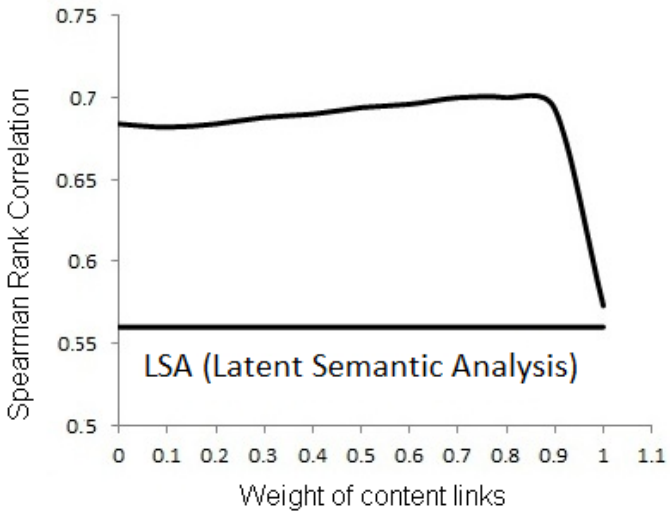


Figure 6: Spearman rank correlation ρ between automatic and human judgments of word similarity on the WordSimilarity-353 data set, depending on the weight of content links in the random walk (the weight of hyperlinks is the complement to 1). The best scores, $\rho = 0.70$, are reached when content links have more weight than hyperlinks (0.7–0.8 vs. 0.3–0.2). The result of LSA, $\rho = 0.56$, is quoted from [46], and is outperformed by other scores in the literature.

The values of the Spearman rank correlation coefficient ρ on the WordSimilarity-353 data set, for varying relative weights of content links vs. hyperlinks in the random walk, are shown in Figure 6. The best scores reach $\rho = 0.70$ for a combination of hyperlinks and content links, weighted between 0.3/0.7 and 0.2/0.8. As shown in the figure, results improve by combining links in comparison with using a single type. Results improve rapidly when hyperlinks are added to content ones (right end of the curve), even with a small weight, which shows that adding encyclopedic knowledge represented by hyperlinks to word co-occurrence information can improve significantly the performance. Conversely, adding content links to hyperlinks also improves the results, likely because the effect of spurious hyperlinks is reduced after adding content links.

To find out whether the two types of links encode similar relations or not, we examined to what extent results using *VP* with hyperlinks only are correlated with results using content links only. The Spearman rank correlation coefficient between these scores is $\rho = 0.71$, which shows that there is some independence between scores.

We also tested our method on the R&G and M&C data sets. The values of the Pearson correlation coefficient r for previous algorithms using lexical resources are given by Jarmasz [11, Section 4.3.2], by Gabrilovich and Markovitch [47, Table 3] and by Agirre et al. [68, Table 7], showing again that for word similarity, using lexical resources is very successful, reaching a correlation of 0.70–0.85 with human judgments. For our own algorithm, correlation r with human judgments on R&G and M&C is, respectively, 0.38–0.42 and 0.46–0.50, depending on the combination of links that is used. Lexically-based techniques are thus more successful on these data sets, for which only the lexical similarity relation is important, while our method, which considers linguistic as well as extra-linguistic relations, is less efficient.

However, if we examine the Spearman rank correlation ρ on the R&G and M&C data sets, considering therefore only the ranking between pairs of words and not the exact values of the relatedness scores, then our method reaches 0.67–0.69. Our scores are thus still lower than the best lexically-based techniques, which have a ρ between 0.74–0.81 and 0.69–0.86 on R&G and M&C, but the difference is now smaller. Our method is thus more suitable for capturing

the ranking of the pairs instead of their exact scores, an observation that was also made by Gabrilovich and Markovitch [47].

Gabrilovich and Markovitch [46, Table 4] (see also [47, Table 3]) and Agirre et al. [68, Table 9] provide the values of the Spearman rank correlation coefficient ρ of previous methods on the WordSimilarity-353 data set. The best results is obtained by Explicit Semantic Analysis with $\rho = 0.75$ and by a system combination of distributional and WordNet-based methods (Agirre et al. [68]) with $\rho = 0.78$. Apart from these methods, the best reported scores on this data are the ones reached by LSA with $\rho = 0.56$ oly. Our method outperforms this score by a margin that is similar to that of ESA or system combination [68], though our method does not quite reach their scores. Some authors have attempted to reproduce the ESA scores: Zesch et al. [49] reached only $\rho = 0.46$, while Ramage et al. [44] and Hassan and Mihalcea [48] reported results close to the original ones [46, 47]. A key factor that ensures high performance seems to be the cleaning procedure applied to concept vectors [47, 3.2.3]. Overall, to facilitate comparison of our scores to important scores in the literature, Table 4 provides a synthetic view.

WordSimilarity-353		**M&C data set**		
Study	ρ	**Study**	ρ	r
Finkelstein et al. [71]	0.56	Wu and Palmer [72]	0.78	0.78
Jarmasz [11]	0.55	Resnik [8]	0.81	0.80
Strube and Ponzetto [37]	0.48	Leacock and Chodorow [25]	0.79	0.82
Hughes and Ramage [31]	0.55	Lin [27]	0.82	0.83
Gabrilovich and Markovitch [46]	0.75	Jarmasz [11]	0.87	0.87
Agirre et al. [68]	0.78	Patwardhan and Pedersen [73]	N/A	0.91
		Bollegala et al. [67]	0.82	0.83
		Alvarez and Lim [74]	N/A	0.91
		Hughes and Ramage [31]	0.90	N/A
		Agirre et al. [68]	0.92	0.93
VP	0.70	*VP*	0.69	0.50

Table 4: A comparison of several word similarity scores, in terms of Spearman rank correlation ρ and Pearson correlation r, on two data sets: WordSimilarity-353 [71] and M&C [70]. Our scores for *VP* appear in the last line, and are for WordSimilarity-353 in the upper range, though not above state-of-the-art ones.

To improve our understanding of the types of links, Figure 7 shows the average frequency of the path lengths traveled for computing ε-truncated *VP* on the word pairs from WordSimilarity-353, for three different combinations of links. The results show that using hyperlinks shortens the length of the average path that is used for the computation of *VP*. Conversely, by using hyperlinks, the number of paths between words is increasing dramatically in comparison to using content links only, a fact that explains why adding hyperlinks, even with a small weight, to content links improves the results so rapidly in Figure 6.

11. Document Similarity

The estimation of document similarity is another task on which our proposal was assessed. The document similarity data set used in this experiment was gathered by Lee et al. [75], and contains average human similarity scores for all pairs of a set of 50 documents.

As in the experiment with word similarity (using the same parameters $\alpha = 0.8$ and $\varepsilon = 10^{-5}$), we tested our method using various combinations of weights for the two types of links, with results shown in Figure 8. Following Gabrilovich and Markovitch [46] – and because the averaged human judgments cover only a small range of possible values – we use the Pearson correlation coefficient r to evaluate how close our method approaches the human judgments. For these experiments, each document from the set was mapped to the 1,000 closest concepts in the network. Otherwise, the same random walk parameters as for the word similarity task were used.

Our findings are that the behavior of the system and its best results are similar to the previous experiment on word similarity. Adding hyperlinks to content links improves the correlation sharply, but adding content links to hyperlinks also improves the results (after an initial decrease), so that the best performance ($r = 0.676$) is reached with a combination of links, which appears to be very similar to the word similarity experiment.

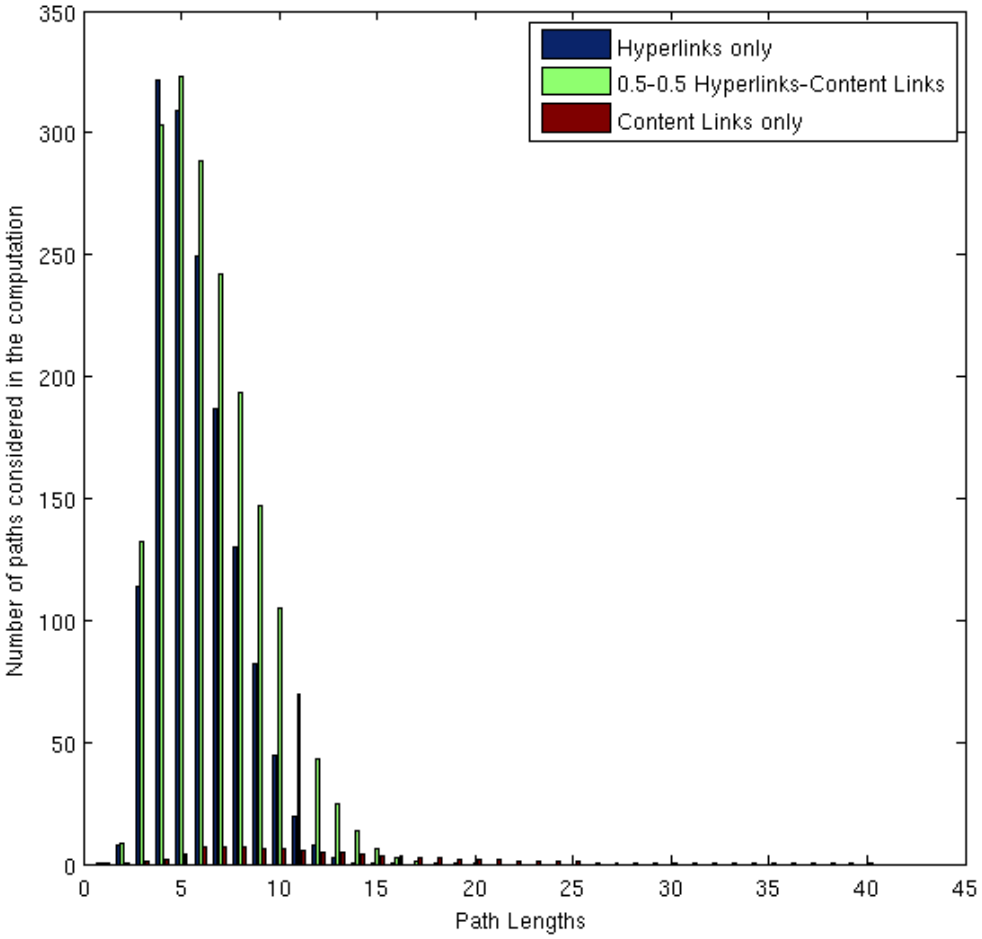


Figure 7: Average frequency of the path lengths contributing to *VP* on the WordSimilarity-353 data set, in three configurations (left to right for each integer value): hyperlinks only, equal weights, content links only.

The authors of the data set (Lee et al. [75]) report the results of several other methods on the document similarity task, the best one reaching $r = 0.60$ using LSA. However, in this case (also mentioned by Gabrilovich and Markovitch [46]), LSA was trained only a small document corpus of 314 news articles. When trained over a much larger corpus, the performance of LSA increases to $r = 0.69$, a value reported by Hassan and Mihalcea [76] after training LSA over the entire Wikipedia corpus. ESA [46] reaches a score of $r = 0.72$ on this task, which outperforms all other methods (as in the case of word similarity) including ours, although by a small margin. Indeed, with our method, the best observed combination reached $r = 0.676$. Therefore, although our method is below some of the best results in the literature, it reaches nevertheless high scores with a general semantic relatedness measure.

As we did for word similarity, we show in Figure 9 the frequency of the path lengths traveled for computing *VP*, averaged on the document similarity data set, for three different combination of links. The figure shows that using mostly hyperlinks shortens the average path length that is used for the computation, while using more content links lengthens the path lengths.

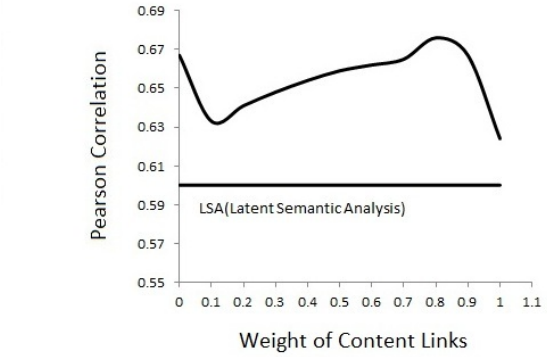


Figure 8: Pearson correlation coefficient r between VP results and human judgments on document similarity, depending on the weight of the hyperlinks in the combination (the weight of the content links is the complement to 1). The best score of $r = 0.676$ is reached when hyperlinks have less weight than content links, but are still used. The result of LSA, $r = 0.60$, is quoted from Lee et al. [75].

12. Document Clustering

This section describes the experimental setting and the results of applying the text relatedness measure defined above to the problem of document clustering over the 20 Newsgroups dataset.[7] The dataset contains about 20,000 postings to 20 news groups, hence 20 document classes with about 1,000 documents per class. We aim here at finding these classes automatically, using for testing the entire data set without using any part of it as a training set. The knowledge of our system comes entirely from the Wikipedia network and the techniques described in Sections 6–7 for computing distances between two texts projected onto the network. We also experimented with the embeddings trained over VP similarities described in Section 9.

12.1. Setup of Experiment on 20 Newsgroups

We first compute a similarity matrix for the entire 20 Newsgroups data set, with the relatedness score between any two documents being VP^T. For tractability, we fixed $T = 10$, a value that gives sufficient precision, as studied empirically in Section 8 above. Similarly, we empirically set the absorption probability of the random walk at $1 - \alpha = 0.2$. Based on α and T, the results in Section 7 allowed us to compute the error bound of the truncation. So, the choice of α and T was also guided by the fact that for a smaller α, fewer steps (T) are needed to achieve the same approximation precision because of the penalty set to longer paths. In this application, instead of computing VP^T between all possible pairs separately, we filled one row of the matrix at a time using the approximations we proposed. To use the embeddings we simply transform all the documents by the embeddings matrices and then run the clustering algorithm over the transformed documents in the projection space. We trained two matrices, A and B, for each Wikipedia graph. The qualitative behavior of the results by using A and B is very similar. Here we report only the results of the transformation by A to help readability of the results.

Clustering is performed using a k-means algorithm over each of the similarity matrices and feature representations. The similarity metric between two representation vectors is cosine similarity. Given the randomized initialization of the algorithm, the final clusterings are different in each run of the algorithm.

The quality of the clustering is measured using the Rand Index (RI), which counts the proportion of pairs of documents that are similarly grouped, i.e. either in the same, or in different clusters, in the reference vs. candidate clusterings. RI tends to penalize the smaller clusters, for example if the algorithm clusters together two classes and

[7] The dataset is distributed at http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/news20.html, see also [77, Chapter 6].

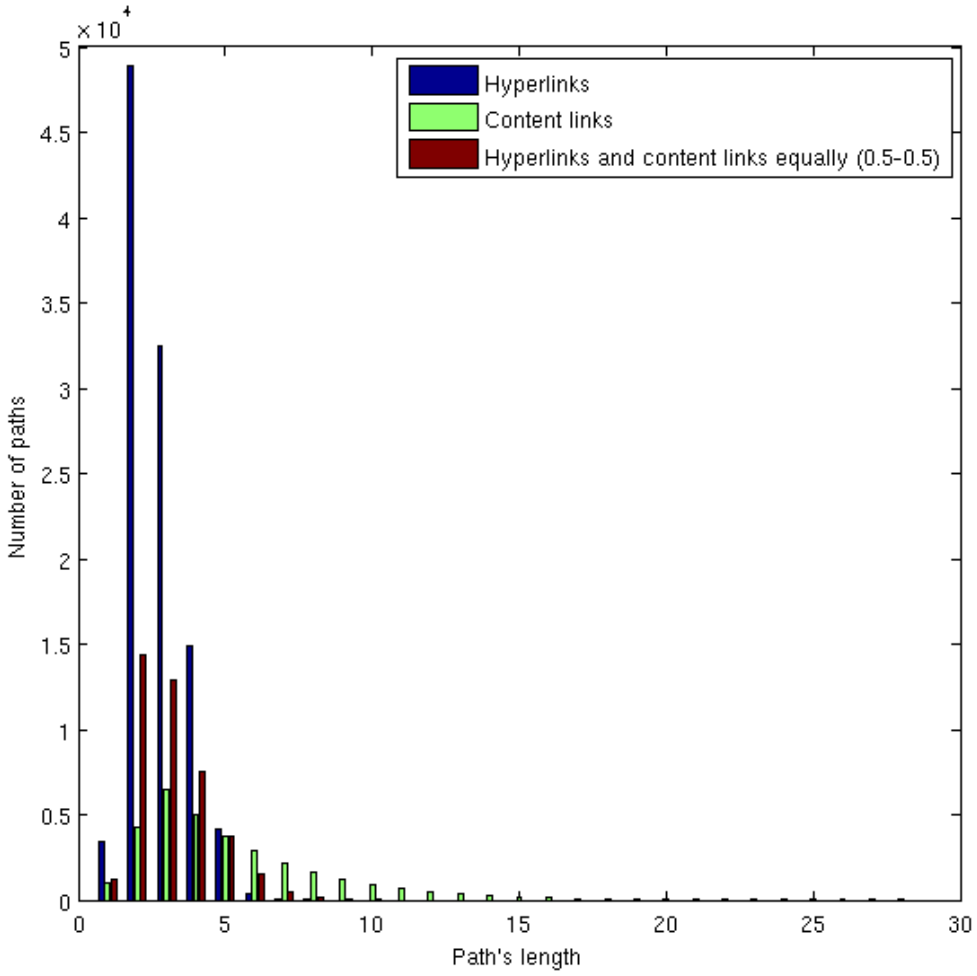


Figure 9: Frequencies of the path lengths contributing to *VP*, averaged over the document similarity data set, in three configurations of the system (left to right for each integer value): hyperlinks only, content links only, and equal weights for both types.

splits another class, it will receive a high penalty from RI. This happens often in 20 Newsgroups data set, given that some classes are very similar and some other classes are more general and potentially suitable for splitting. As a consequence, the RI values vary largely from one run to another run, making significance testing quite difficult. To gain more information about the quality of the clustering, we consider precision, recall and F-score.

Table 5 shows average results (over ten runs) of the clustering algorithm using various relatedness measures. The random walk model significantly outperforms the baseline cosine similarity between TF-IDF vectors for document clustering, and it also outperforms LSA in terms of F-score, with most of the relatedness measures. The RI scores do not allow conclusions as their variation does not allow significance judgments. The comparison with previous work is uneasy, as no systematic comparison of clustering methods on the 20 Newsgroups datasets was known to us. In a recent paper, Hu et al. [78] found an F-score of 14.8% for a baseline word vector method, improved to 19.6% by their use of Wikipedia, for agglomerative clustering. However, for partitional clustering with K-means, both scores increased more than twice (to 38.2% for the baseline and 41.8% for their best method), a result that could not be confirmed independently.

For our approach, the combination of hyperlinks and content links improves the results over using either of them alone. Using the embeddings of the content links reduces the performance in comparison to the computation of the

Distance Metric	Precision	Recall	F-score	RI
Cosine similarity of TF-IDF vectors	9.29	19.04	12.49	86.67
LSA	19.10	20.64	19.84	91.67
VP on content links	**24.13**	**37.15**	**29.13**	90.94
VP on hyperlinks	18.36	**39.22**	24.72	*87.78*
VP_{Comb} (0.6 content links)	**24.26**	**36.91**	**29.23**	91.03
Embedding from content links (ECL)	21.30	**26.13**	**23.47**	91.48
Embedding from hyperlinks (EHL)	**22.63**	**28.40**	**25.17**	91.56
ECL with regularization	**22.74**	**27.67**	**24.95**	91.68
EHL with regularization	**24.08**	**29.64**	**26.56**	91.80

Table 5: Rand Index (RI), precision, recall and F-score of different clustering methods. VP_{Comb} stands for *VP* over a combination of content links (weighted 0.6) and hyperlinks (0.4). Scores in bold are significantly higher than LSA, the one in italics significantly lower (t-test, $p < 0.001$).

VP values over the graph. On the contrary, using embeddings of the hyperlinks improves the results in comparison to using the *VP* values over the hyperlinks graph.

The embedding learned on *VP* similarities over the hyperlinks appears to provide a more general similarity measure, with does not overfit to the Hyperlinks graph of Wikipedia. The high recall it obtains is related to the larger extension of paths computed with hyperlinks, which can connect many documents together and attach them to the same class, while the high precision obtained using content links is due to their tendency to cluster into smaller neighborhoods.

Although the regularization imposed on the embeddings reduced their predictive power for the *VP* similarities, but it improves the performance on this task.

Computation time using embeddings is (as expected) greatly reduced, as computations are performed in the low-dimensional latent space. Moreover, other unsupervised clustering algorithms can be applied to the documents transformed by the embeddings, e.g. the state-of-the-art clustering algorithm proposed by Bordogna and Pasi [79].

12.2. Comparison of VP *and Cosine Similarity*

To find out in which cases the proposed method improves over a simple cosine similarity measure, we considered a linear combination of the cosine similarity and VP_{Comb} (*VP* over content links weighed 0.6 and hyperlinks weighed 0.4), namely $w \times VP_{Comb} + (1 - w) \times LS$, and varied the weight w from 0 to 1. Considering the k-nearest neighbors of every document according to this combined similarity, we define k-purity as the number of documents with the correct label over the total number of documents k in the computed neighborhood. The variation of k-purity with w, for several sample values of k, is shown in Figure 10.

The best purity appears to be obtained for a combination of the two methods, for all values of k that were tested. This shows that VP_{Comb} brings valuable additional information about document relatedness that cannot be found in *LS* only. Furthermore, when the size of the examined neighborhood k increases (lower curves in Figure 10), the effect of VP_{Comb} becomes more important, i.e. its weight in the optimal combination increases. For very small neighborhoods, *LS* is almost sufficient to ensure optimal purity, but for larger ones ($k = 10$ or 15), VP_{Comb} used alone ($w = 1$) outperforms *LS* used alone ($w = 0$). Their optimal combination leads to scores that are higher than those obtained for each of them used separately, and, as noted, the weight of VP_{Comb} in the optimal combination increases for larger neighborhoods.

These results can be explained as follows. For very small neighborhoods, the cosine lexical similarity score with the nearest 1–5 documents is very high, as they have many words in common, so *LS* is a good measure of text relatedness. However, when looking at larger neighborhoods, for which relatedness is less based on identical words, then VP_{Comb} becomes more effective, and *LS* performs poorly. Therefore, we can predict that VP_{Comb} will be most relevant when looking for larger neighborhoods, or in order to increase recall. VP_{Comb} should also be relevant when there is low diversity among document words, for instance when all documents are very short.

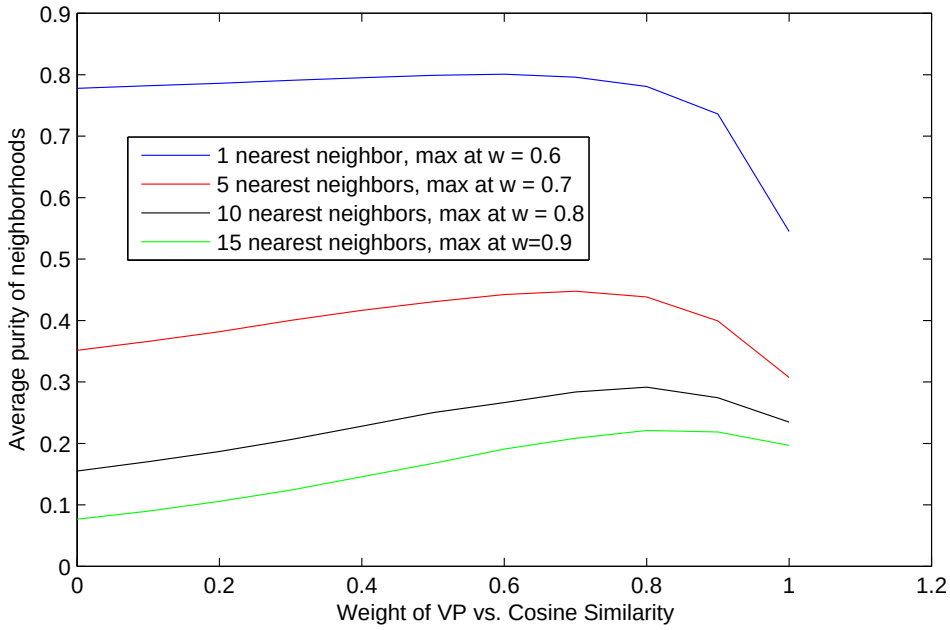


Figure 10: Values of k-purity (vertical axis) averaged over all documents, for neighborhoods of different sizes k. The horizontal axis indicates the weight w of visiting probability vs. cosine lexical similarity in the formula: $w \times VP_{Comb} + (1 - w) \times LS$.

13. Text Classification

We showed in the previous section that using the *VP* similarities over Wikipedia hyperlinks and content links graphs improved text clustering. Although text clustering is an important application, there have been more studies on text classification, i.e. when labeled examples are available for learning. In this section, we investigate this problem by using the embeddings that were learned over *VP* similarities in Section 9 above. Embeddings can easily be integrated with any distance learning algorithm as prior knowledge or as an initial state. We thus designed a distance learning algorithm for this purpose, which we compared (using various embeddings) to an SVM text classifier, outperforming its score when few training examples are available.

13.1. Distance Learning Classifier

We built a distance learning classifier which learns, given a training set, a similarity measure so that for each data point in the training set, its similarity to data points with the same label (or class) is higher than data points with different labels. This classifier is essentially very similar to a large margin nearest neighbor classifier [80], with some changes that make it applicable to large scale text classification problems with a large number of features (here, words).

We define the similarity between two documents i and j represented by TF-IDF vectors x_i and x_j as $x_i AA' x'_j$, where A is a matrix $n \times m$, n being the size of the feature dictionary and m size of the latent space. If $C(i)$ denotes the class (label) of document i, then we define the following loss function L over the training set:

$$L = \sum_i \sum_{C(j)=C(i)} \sum_{C(z)\neq C(i)} max(0, M - x_i AA' x'_j + x_i AA' x'_z)$$

M is a margin which is set to 0.2 in our experiments. We performed the optimization of L by stochastic gradient descent. At testing time, we proceeded similarly to the k-nearest neighbors algorithm: we chose the k closest documents from the training set according to the learned distance and then returned the class (label) resulting from the majority vote.

Our goal here is to show that starting from the prior knowledge obtained from *VP* similarities over Wikipedia graphs in the forms of the embeddings can improve the performance of the classification, especially when a small number of training samples is available.

13.2. 20 Newsgroups Classification

We applied the above method to classify texts from the 20 Newsgroups data set. We compared the results of the distance learning algorithm, with various initial points, to a linear SVM method (LIBSVM implementation [81]), which is a state-of-the-art text classifier. The classification accuracy is given in Table 6 for various sizes of the training set. We have trained two matrices over *VP* similarities, *A* and *B*, because *VP* similarity is not symmetric. We have experimented with initialization by either *A* or *B*, which gives similar results, therefore we show only the results using matrix *A*. The distance learning classifier with random initialization of the parameters performed poorly, so it is not reported here.

Method	**Size of the training set**						
	40	**100**	**200**	**500**	**800**	**1000**	**1500**
DL + CL embedding	*21.13*	*32.93*	*42.79*	56.57	62.54	65.90	70.57
DL + CL emb. + REG	*22.54*	*34.14*	*44.32*	*57.12*	63.82	66.43	71.10
DL + HL embedding	*21.40*	*34.31*	*44.09*	*57.18*	63.09	66.31	70.65
DL + HL emb. + REG	*22.50*	*35.29*	*46.17*	*58.64*	64.08	66.84	71.14
SVM	7.90	15.93	28.48	52.40	61.76	65.67	70.83

Table 6: Classification accuracy for different sizes of the training set over the 20 Newsgroups data set. The accuracy is the average of 10 times run on a randomly divided data set. 'DL' is distance learning classifier, 'CL' is the embedding learned over the content links graph and 'HL' the embedding learned over hyperlinks, 'REG' stands for regularization of the matrix. The numbers in italics are significantly better than the accuracy of the SVM (t-test, $p < 0.001$)

.

The first important observation is that, when the training set is small, the distance learning classifier initialized with the embeddings from *VP* similarities over Wikipedia graphs outperforms the baseline SVM classifier significantly. By adding more and more labeled data, the importance of prior knowledge appears to decrease, presumably because the distance learning algorithm can infer reliable decisions based only on the training data. A similar effect was shown for a method based on deep learning by Ranzato and Szummer [82], with their method and a TF-IDF/SVM method both reaching 65% accuracy for more than 50 samples per class (corresponding to 1000 total samples) in the training data.

The second observation is that the orthonormality regularization that we imposed on the embeddings again improved the performance. The generalization ability was improved at the price of decreasing slightly the precision in the approximation of the *VP* values. A third observation is that the accuracy using hyperlinks was slightly higher than using content links.

14. Information Retrieval

In this section, we apply the proposed distance to information retrieval data from TREC-7 and TREC-8 [83]. The application of our method to a large scale information retrieval task requires the computation of *VP* between a query and the representation of *all the documents* in the repository. By using the approximations proposed in Section 7, we can compute T-truncated *VP* between every query and documents in an acceptable amount of time. Firstly, we map all documents in the data set to the Wikipedia graph; then, at query time, each query is mapped to the graph; finally, for each query, VP^T is computed to and from all documents in the data set by using the proposed approximations. In this section, we use $\alpha = 0.8$, $T = 10$ and a combination of hyperlinks and content links in *VP* weighted 0.2/0.8, following observations from previous sections. The time-consuming operation is the first one – viz., mapping a large collection of documents to the Wikipedia graph – but this is done only once, before query time.

We used the TREC-7 and TREC-8 *Ad-hoc Test Collections*.[8] The data set includes a repository of 530,000 documents and two sets of 50 queries. For each query, the data set also provides relevance judgments for a subset of documents considered to be related to the query, as they were retrieved by a pool of search methods, a method that is intended to maximize recall. To study the effect of *VP* in comparison to lexical similarity and TF-IDF scoring in a single retrieval operation, we compute for each document and query a linear combination of VP^T with weight w and of lexical similarity with weight $1 - w$. The weight w thus sets the relative importance of conceptual relatedness vs. lexical similarity, and will serve to illustrate their respective contributions. We will refer to this as the *Combined* similarity. We measure the IR performance through the average precision of the first 10 returned documents (10 is the typical size of the first result page of Web search engines).

We computed all the scores for $w = 0$ to $w = 1$ with increments of 0.1, and found out that the highest score was reached for $w = 0.1$ for the TREC-7 data set. This optimized value was then tested over the TREC-8 query set (50 topics), leading to an average precision for *Combined* of 0.515 , which improves (+15.4 %) over the precision of lexical similarity alone ($w = 0$), which is 0.446. Egozi et al. [84] reported precision at 10 on TREC-8 data for various bag-of-words information retrieval systems (Xapian, Okapi, LM-KL-DIR) along with improved results (up to 14%) obtained by using a new retrieval algorithm, which integrates ESA [47] semantic similarity to the previous systems.[9] Here, we reported the mean average precision at 10, which is the lower bound of precision at 10, but is more informative than it. Direct comparison between our scores and the scores reported in [84] is not possible, as they used different base retrieval systems.

We also examined the precision score for every query separately. In particular, we counted the proportion of queries for which the *Combined* similarity returned more relevant documents than lexical similarity alone. *Combined* similarity outperformed the lexical one on 14 queries, while the reverse was true for 5 queries only, and the scores were identical on the remaining 31 queries. The average score difference between *Combined* and lexical similarity is 0.018 (maximum is 1). This value is not significantly above 0 at the usual levels (e.g. $p < 0.01$), but we found using a normal distribution that the probability of the true difference being zero, given the observations, is only $p = 0.11$. While this does not ensure that the *Combined* similarity is "significantly" better than the lexical one, it still provide encouraging evidence for the utility of *VP* on the TREC-8 test set.

The precision scores of both methods vary considerably across queries. We therefore examined separately the "difficult" queries, defined here as the queries on which lexical similarity had a score of 0.3 or less (meaning it returned between 0 and 3 relevant documents). There are 21 such queries out of 50 on the TREC-8 test set. Over these queries, the *Combined* similarity outperforms the lexical one on 7, while the reverse is true for only one, and 13 queries are a tie. Of course, it might seem unsurprising to see this difference as these queries are "difficult" for the lexical similarity by definition. However, when examining the 20 queries that are "difficult" for the *Combined* similarity, this measure still outperforms the lexical one on 5, while the reverse is true for only two, and 13 are a tie. These results show that *VP* provides complementary information that can improve the results of lexical similarity for IR, especially for queries on which lexical similarity performs less well.

15. Learning to Rank by Using *VP* Similarities

We have seen in the previous section that the linear combination of the *VP* and lexical similarities improved only slightly the retrieval results, although *VP* provides additional information. This motivated us to integrate the *VP* similarity to a learning to rank system [85], instead of a linear combination with lexical similarity. We have chosen an approach similar to the discriminative projection learning algorithm introduced by Yih et al. [86] which exhibits good performance. We have reimplemented the algorithm for the experiments in this section, and we first describe it briefly, then discuss the results.

Assume that for a query q, a document d_r is relevant and d_{nr} is not relevant. The algorithm learns a projection A from TF-IDF vectors of the articles to a latent space such that the similarity between the projections of q and d_r

[8] These are available at http://trec.nist.gov, and we used more specifically the test queries (topics) numbers 351–400 and 401–450, with the associated relevance judgments. The documents are available from the Linguistic Data Consortium.

[9] Precision at 10 was improved as follows: for Xapian from 0.472 to 0.478 (+1.3%), for Okapi from 0.488 to 0.522 (+7.0%), and for LM-KL-DIR from 0.442 to 0.506 (+14.4%).

is higher than the similarity between those of q and d_r. Given a training set consisting of (q_i, d_{r_i}, d_{nr_i}), the algorithm minimizes the following loss function L over the training set:

$$L = \sum_i max(0, M - q_i AA' d'_{r_i} + q_i AA' d'_{nr_i})$$

We will show that using the embeddings learned from *VP* as a starting point for minimizing L can help to improve the ranking performance.

To build the training and test sets, we used the 50 queries from TREC-8 and considered for each query the documents labeled as relevant (4728 documents), while unlabeled documents were considered as irrelevant. We divided evenly and randomly the pairs of queries and relevant documents into a training and a test set. The possible number of triples in the training set is very large due to the large number of irrelevant documents. We perform stochastic gradient descent over the training set by choosing at each iteration a query and a relevant document, with an irrelevant document chosen randomly from the rest of the documents. We stop when the training error is lower than a fixed threshold. To test the performance, we report average precision at 10. This is computed over the number of relevant documents that are not used for training. Therefore, when using a larger training set, fewer documents are left for testing and the precision at 10 scores necessarily decrease; however, our interest is in comparing scores for different methods on the same training set.

Method	**Size of the training set**				
	500	**1000**	**1500**	**2000**	**3000**
RL + CL embeddings	13.62	14.40	12.62	11.24	5.54
RL + CL emb. + REG	**17.48**	**17.84**	**14.02**	12.08	5.56
RL + HL embeddings	15.74	15.98	13.10	11.28	5.36
RL + HL emb. + REG	**19.44**	**18.60**	**14.34**	12.46	5.70
RL + random start	16.3	15.16	13.56	12.48	5.34
Cosine between TF-IDF vectors	**21.16**	14.74	9.62	7.98	2.70

Table 7: Precision at 10 for different ranking algorithms when training size varies. As in Table 6, 'RL' is the ranking learner, 'CL' is the embedding learned over the content links graph and 'HL' the embedding learned over hyperlinks, and 'REG' stands for regularization of the matrix.

Table 7 shows the precision at 10 of various algorithms by using different initial states and different number of training samples. The first observation is that only when the training size is very small, 10 documents on average for each query, the cosine similarity outperforms the learning to rank algorithms. Otherwise, the performance of the learning to rank algorithms is always better than the cosine similarity.

The second result is that when the number of training examples is small, the learning algorithm initialized with the regularized embeddings outperforms the random initialization. Gradually, when adding more training examples, it becomes less useful to leverage the prior knowledge, as the learning algorithm can solve the problem better simply by looking at training samples. Similarly to the classification experiments, the embeddings learned over the hyperlinks are more useful than the ones learned over the content links.

16. Perspectives

In addition to the above experiments, we examined two-stage random walks, in which at the first stage, the network is built using one set of weights on the links, and in the second stage using a different set. The hypothesis here is that some links might be more useful when explored first, while some others might be more useful when explored later, as discussed by Collins-Thompson and Callan [87].

For the word similarity task, we focused on ε-truncated two-stage walks in which one combination of links is used for the first three steps of the random walker, and a different combination from the fourth to the last steps. The choice of three steps was empirical, by looking at the average length of single-stage ε-truncated walks. We report the ρ scores of three significant combinations of weights for this scenario in Table 8, including the best we found, which reached $\rho = 0.714$, higher than the one-stage walks (see Section 10). As the optimization took place on the test data, this is

not a competitive score, but intends to show that two-stage walks are a promising approach, in particular exploring the hyperlinks first and then the content links.

A similar analysis to the one shown in Figure 7 explains why scores improve in the two-stage random walk in Table 8, which travels hyperlinks first (thus expanding the possible paths), and then content links (following precise neighborhoods). In the case of exploring hyperlinks first and content links second, there are longer paths in comparison to using only hyperlinks. In the case of exploring hyperlinks in the second stage, there are many long paths in comparison to other scenarios.

(Hyperlink weight, Content link weight)	ρ
(1.0, 0.0) for 3 steps, then (0.0, 1.0)	.684
(0.0, 1.0) for 3 steps, then (1.0, 0.0)	.652
(0.7, 0.3) for 3 steps, then (0.0, 1.0)	**.714**

Table 8: Spearman rank correlation coefficient ρ between automatic and human word similarity when two-stage random walk is used for *VP*. In parentheses the respective weights of hyperlinks and content links. The best result, $\rho = 0.714$ is found when both hyperlinks and content links are used for the first three stages (with weights 0.7 vs. 0.3), but only content links are used for latter stages.

The results of VP following two-stage random walks with several meaningful combinations of links are given in Table 9 for document similarity data set. The scores on document similarity can be slightly improved to $r = 0.680$ if hyperlinks are mostly explored in the first steps, and then only content links are followed which is congruent with our finding about two stage random walk for word similarities.

(Hyperlink weight, Content link weight)	r
(1.0, 0.0) for 3 steps, then (0.0, 1.0)	.667
(0.0, 1.0) for 3 steps, then (1.0, 0.0)	.635
(0.8, 0.2) for 3 steps, then (0.0, 1.0)	**.680**

Table 9: Pearson correlation coefficient r between two-stage *VP* results and human judgments on document similarity. The best result (0.680) is found when for the first three stages both hyperlinks (0.8) and content links (0.2) are used, but only content links are used for latter stages.

17. Conclusion

We have proposed a general framework for text semantic similarity based on knowledge extracted from Wikipedia. We have constructed a graph including Wikipedia articles and two different link structures between them. Our hypothesis was that using both word co-occurrence information and user-defined hyperlinks between articles could improve the resulting textual distance for application to a variety of tasks: word similarity; document similarity, clustering, and classification; information retrieval, and learning to rank.

We tested our approach on four different benchmark tasks, and found that results were often competitive compared to state-of-the-art results obtained by task-specific methods. Our experimental results supported the hypothesis that both types of links are useful, as the improvement of performance was higher when both were used together rather than separately.

We have introduced visiting probability (*VP*) to measure proximity between weighted sets of concepts, and proposed approximation algorithms to compute it efficiently for large graphs. One advantage of our approach is that the training phase (building the network of concepts) is not computationally demanding, as all the computation is done at query time. Therefore, the update of the data set and related network does not imply an additional cost of re-computation. At run time, we have shown how to make computation possible, using a k-nearest-neighbors graph in the random walk framework, for a large resource such as the English Wikipedia. The truncation of *VP* did not only speed up computation, but also improved the accuracy of results by reducing the importance of very popular vertices. To speed up computation at run time even more, we showed that it was possible to train embeddings to learn the proposed similarity measures. In addition to the gain in speed, we were able to integrate the proposed distance as prior knowledge, in the form of embeddings, to several learning algorithms: document clustering, document classification, and learning to rank.

Acknowledgments

This work has been supported by the Swiss National Science Foundation through the National Center of Competence in Research (NCCR) on Interactive Multimodal Information Management (IM2), http://www.im2.ch. We are very much indebted to the AI Journal reviewers and special issue editors for their numerous and insightful suggestions on previous versions of the paper. We would like to thank Michael D. Lee for access to the textual similarity dataset, as well as Ronan Collobert and Jean-Cédric Chappelier for helpful discussions on several issues related to the paper.

Appendix A. Estimate of Truncated *VP* to any Vertex

In Section 7, we have announced a formal probabilistic bound for the differences between the estimated truncated *VP*, noted $\hat{VP}^T$, and the exact truncated VP^T to any vertex. The complete proof of the result, inspired by the proof in [88], is given below.

Let us note the estimation of a variable X by $\hat{X}$, and suppose that concept s_j has been visited for the first time at $\{t_{k_1}, \cdots, t_{k_M}\}$ time steps in the M samples. We define the random variable X^l by $\alpha^{t_{k_l}}/M$, where t_{k_l} indicates the time step at which s_j was visited for the first time in l^{th} sampling. If s_j was not visited at all, then $X^l = 0$ by convention. The l random variables X^l ($k_1 \leq l \leq k_M$) are independent and bounded by 0 and 1 ($0 \leq X^l \leq 1$). We have $\hat{VP}^T(\vec{r}, s_j) = \sum_l X^l = (\sum_l \alpha^{t_{k_l}})/M$ and $E(\hat{VP}^T(\vec{r}, s_j)) = VP^T(\vec{r}, s_j)$. So, by applying Hoeffding's inequality, we have:

$$P(|\hat{VP}^T(\vec{r}, s_j) - E(\hat{VP}^T(\vec{r}, s_j))| \geq \varepsilon) \leq 2exp(-\frac{2M\varepsilon^2}{\alpha^2})$$

If the probability of error must be at most δ, then setting the right side lower than δ gives the bound for M that is stated in our theorem.

As a consequence, we have the following lower bound for M if we look for an ε-approximation for all possible s_j with probability at least $1 - \delta$. We use the union bound and Hoeffding's inequality to prove that:

$$P(\exists j \in \{1 \cdots n\}, |\hat{VP}^T(r, s_j) - E(\hat{VP}^T(r, s_j))| \geq \varepsilon) \leq 2n \times exp(-\frac{2M\varepsilon^2}{\alpha^2})$$

which gives the desired lower bound $M \geq \frac{\alpha^2 \ln(2n/\delta)}{2\varepsilon^2}$.

References

References

[1] A. Budanitsky, G. Hirst, Evaluating WordNet-based measures of semantic distance, Computational Linguistics 32 (2006) 13–47.
[2] S. Patwardhan, S. Banerjee, T. Pedersen, Using measures of semantic relatedness for word sense disambiguation, in: Proceedings of CICLing 2003 (4th International Conference on Computational Linguistics and Intelligent Text Processing), LNCS 2588, Mexico City, Mexico, pp. 241–257.
[3] U. S. Kohomban, W. S. Lee, Learning semantic classes for word sense disambiguation, in: Proceedings of ACL 2005 (43rd Annual Meeting of the Association for Computational Linguistics), Ann Arbor, MI, pp. 34–41.
[4] S. P. Ponzetto, M. Strube, Knowledge derived from Wikipedia for computing semantic relatedness, Journal of Artificial Intelligence Research 30 (2007) 181–212.
[5] M. Stevenson, M. Greenwood, A semantic approach to IE pattern induction, in: Proceedings of ACL 2005 (43rd Annual Meeting of the Association for Computational Linguistics), Ann Arbor, MI, pp. 379–386.
[6] M. Baziz, M. Boughanem, N. Aussenac-Gilles, C. Chrisment, Semantic cores for representing documents in IR, in: Proceedings of SAC 2005 (ACM Symposium on Applied Computing), Santa Fe, NM, pp. 1011–1017.
[7] D. Newman, J. H. Lau, K. Grieser, T. Baldwin, Automatic evaluation of topic coherence, in: Proceedings of HLT-NAACL 2010 (Annual Conference of the North American Chapter of the Association for Computational Linguistics), Los Angeles, CA, pp. 100–108.
[8] P. Resnik, Using information content to evaluate semantic similarity in a taxonomy, in: Proceedings of IJCAI 1995 (14th International Joint Conference on Artificial Intelligence), Montreal, pp. 448–453.
[9] C. Fellbaum (Ed.), WordNet: An electronic lexical database, The MIT Press, Cambridge, MA, 1998.
[10] D. B. Lenat, CYC: A large-scale investment in knowledge infrastructure, Communications of the ACM 38 (1995) 33–38.
[11] M. Jarmasz, Roget's Thesaurus as a Lexical Resource for Natural Language Processing, Master's thesis, University of Ottawa, 2003.
[12] J. Morris, G. Hirst, Non-classical lexical semantic relations, in: Proceedings of the HLT-NAACL 2006 Workshop on Computational Lexical Semantics, Boston, MA, pp. 46–51.

[13] M. Halliday, R. Hasan, Cohesion in English, Longman, London, 1976.
[14] J. Hobbs, Why is discourse coherent?, in: F. Neubauer (Ed.), Coherence in natural language texts, Buske, Hamburg, 1983, pp. 29–70.
[15] R. Hasan, Coherence and cohesive harmony, in: J. Flood (Ed.), Understanding reading comprehension: Cognition, language and the structure of prose, International Reading Association, Newark, DE, 1984, pp. 181–219.
[16] M. Halliday, R. Hasan, Language, context, and text: aspects of language in a social-semiotic perspective, Oxford University Press, London, 2nd edition, 1989.
[17] S. Mohammad, G. Hirst, Distributional measures as proxies for semantic distance: A survey, Available online (consulted on April 15, 2012) at: http://www.cs.toronto.edu/pub/gh/Mohammad+Hirst-2005.pdf, 2005.
[18] J. E. Weeds, Measures and applications of lexical distributional similarity, PhD thesis, University of Sussex, 2003.
[19] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, Indexing by Latent Semantic Analysis, Journal of the American Society for Information Science 41 (1990) 391–407.
[20] S. Padó, M. Lapata, Dependency-based construction of semantic space models, Computational Linguistics 33 (2007) 161–199.
[21] T. Hofmann, Probabilistic Latent Semantic Indexing, in: Proceedings of SIGIR 1999 (22nd ACM SIGIR Conference on Research and Development in Information Retrieval), Berkeley, CA, pp. 50–57.
[22] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent Dirichlet Allocation, Journal of Machine Learning Research 3 (2003) 993–1022.
[23] M. Jarmasz, S. Szpakowicz, Roget's thesaurus and semantic similarity, in: Proceedings of RANLP 2003 (Conference on Recent Advances in Natural Language Processing), Borovetz, Bulgaria, pp. 111–120.
[24] R. Rada, H. Mili, E. Bicknell, M. Blettner, Development and application of a metric to semantic nets, IEEE Transactions on Systems, Man and Cybernetics 19 (1989) 17–30.
[25] C. Leacock, M. Chodorow, Combining local context and WordNet similarity for word sense identification, in: C. Fellbaum (Ed.), WordNet: An electronic lexical database, The MIT Press, Cambridge, MA, 1998, pp. 265–283.
[26] P. Resnik, Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language, Journal of Artificial Intelligence Research 11 (1999) 95–130.
[27] D. Lin, An information-theoretic definition of similarity, in: Proceedings of ICML 1998 (15th International Conference on Machine Learning), Madison, WI, pp. 296–304.
[28] J.-C. Chappelier, Topic-based generative models for text information access, in: E. Gaussier, F. Yvon (Eds.), Textual Information Access: Statistical Models, ISTE / Wiley, London, UK, 2012, pp. 129–178.
[29] J. Chang, J. Boyd-Graber, S. Gerrish, C. Wang, D. M. Blei, Reading tea leaves: How humans interpret topic models, in: Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, A. Culotta (Eds.), Advances in Neural Information Processing Systems 22, The MIT Press, Cambridge, MA, 2009, pp. 288–296.
[30] R. Mihalcea, C. Corley, C. Strapparava, Corpus-based and knowledge-based measures of text semantic similarity, in: Proceedings of AAAI 2006 (21st National Conference on Artificial Intelligence), Boston, MA, pp. 775–782.
[31] T. Hughes, D. Ramage, Lexical semantic relatedness with random graph walks, in: Proceedings of EMNLP-CoNLL 2007 (Conference on Empirical Methods in Natural Language Processing and Conference on Computational Natural Language Learning), Prague, Czech Republic, pp. 581–589.
[32] T. H. Haveliwala, Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search, IEEE Transactions on Knowledge and Data Engineering 15 (2003) 784–796.
[33] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia – a crystallization point for the web of data, Journal of Web Semantics 7 (2009) 154–165.
[34] R. Navigli, S. P. Ponzetto, BabelNet: Building a very large multilingual semantic network, in: Proceedings of ACL 2010 (48th Annual Meeting of the Association for Computational Linguistics), Uppsala, Sweden, pp. 216–225.
[35] V. Nastase, M. Strube, B. Boerschinger, C. Zirn, A. Elghafari, WikiNet: A very large scale multi-lingual concept network, in: Proceedings of LREC 2010 (7th International Conference on Language Resources and Evaluation), Valletta, Malta.
[36] F. Suchanek, G. Kasneci, G. Weikum, Yago: A large ontology from Wikipedia and WordNet, Journal of Web Semantics 6 (2008) 203–217.
[37] M. Strube, S. P. Ponzetto, WikiRelate! Computing semantic relatedness using Wikipedia, in: Proceedings of AAAI 2006 (21st National Conference on Artificial Intelligence), Boston, MA, pp. 1419–1424.
[38] S. P. Ponzetto, M. Strube, Taxonomy induction based on a collaboratively built knowledge repository, Artificial Intelligence 175 (2011) 1737–1756.
[39] D. Milne, I. H. Witten, An effective, low-cost measure of semantic relatedness obtained from Wikipedia links, in: Proceedings of WIKIAI 2008 (1st AAAI Workshop on Wikipedia and Artificial Intelligence), Chicago, IL, pp. 25–30.
[40] Z. S. Syed, T. Finin, A. Joshi, Wikipedia as an ontology for describing documents, in: Proceedings of the Second International Conference on Weblogs and Social Media, Seattle, WA, pp. 136–144.
[41] D. Milne, I. H. Witten, Learning to link with Wikipedia, in: Proceedings of CIKM 2008 (17th ACM Conference on Information and Knowledge Management), Napa Valley, CA, pp. 509–518.
[42] K. Coursey, R. Mihalcea, W. Moen, Using encyclopedic knowledge for automatic topic identification, in: Proceedings of CoNLL 2009 (13th Conference on Computational Natural Language Learning), Boulder, CO, pp. 210–218.
[43] R. Mihalcea, A. Csomai, Wikify!: Linking documents to encyclopedic knowledge, in: Proceedings of ACM CIKM 2007 (16th ACM Conference on Information and Knowledge Management), Lisbon, Portugal, pp. 233–242.
[44] D. Ramage, A. N. Rafferty, C. D. Manning, Random walks for text semantic similarity, in: Proceedings of TextGraphs-4 (4th Workshop on Graph-based Methods for Natural Language Processing), Singapore, pp. 23–31.
[45] M. Yazdani, A. Popescu-Belis, A random walk framework to compute textual semantic similarity: a unified model for three benchmark tasks, in: Proceedings of IEEE ICSC 2010 (4th IEEE International Conference on Semantic Computing), Pittsburgh, PA, pp. 424–429.
[46] E. Gabrilovich, S. Markovitch, Computing semantic relatedness using Wikipedia-based explicit semantic analysis, in: Proceedings of IJCAI 2007 (20th International Joint Conference on Artificial Intelligence), Hyderabad, India, pp. 6–12.
[47] E. Gabrilovich, S. Markovitch, Wikipedia-based semantic interpretation for natural language processing, Journal of Artificial Intelligence

Research 34 (2009) 443–498.
[48] S. Hassan, R. Mihalcea, Cross-lingual semantic relatedness using encyclopedic knowledge, in: Proceedings of EMNLP 2009 (Conference on Empirical Methods in Natural Language Processing), Singapore, pp. 1192–1201.
[49] T. Zesch, C. Müller, I. Gurevych, Using Wiktionary for computing semantic relatedness, in: Proceedings of AAAI 2008 (23rd National Conference on Artificial Intelligence), volume 2, Chicago, IL, pp. 861–866.
[50] P. Cimiano, A. Schultz, S. Sizov, P. Sorg, S. Staab, Explicit vs. latent concept models for cross-language information retrieval, in: Proceedings of IJCAI 2009 (21st International Joint Conference on Artificial Intelligence), Pasadena, CA, pp. 1513–1518.
[51] E. Yeh, D. Ramage, C. D. Manning, E. Agirre, A. Soroa, WikiWalk: random walks on Wikipedia for semantic relatedness, in: Proceedings of TextGraphs-4 (4th Workshop on Graph-based Methods for Natural Language Processing), Singapore, pp. 41–49.
[52] E. Agirre, A. Soroa, Personalizing PageRank for word sense disambiguation, in: Proceedings of EACL 2009 (12th Conference of the European Chapter of the Association for Computational Linguistics), Athens, Greece, pp. 33–41.
[53] P. Turney, Mining the web for synonyms: PMI-IR versus LSA on TOEFL, in: Proceedings of ECML 2001 (12th European Conference on Machine Learning), Freiburg, Germany, pp. 491–502.
[54] P. Berkhin, A survey on PageRank computing, Internet Mathematics 2 (2005) 73120.
[55] R. Navigli, M. Lapata, An experimental study of graph connectivity for unsupervised word sense disambiguation, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (2010) 678–692.
[56] R. Ion, D. Ştefănescu, Unsupervised word sense disambiguation with lexical chains and graph-based context formalization, in: Proceedings of LTC 2009 (4th Language and Technology Conference), LNAI 6562, Poznan, Poland, pp. 435–443.
[57] R. Navigli, A structural approach to the automatic adjudication of word sense disagreements, Natural Language Engineering 14 (2008) 547573.
[58] M. Saerens, F. Fouss, L. Yen, P. Dupont, The principal components analysis of a graph, and its relationships to spectral clustering, in: Proceedings of ECML 2004 (15th European Conference on Machine Learning), Pisa, Italy, pp. 371–383.
[59] M. Brand, A random walks perspective on maximizing satisfaction and profit, in: Proceedings of the 2005 SIAM International Conference on Data Mining, Newport Beach, CA, pp. 12–19.
[60] D. Liben-Nowell, J. Kleinberg, The link prediction problem for social networks, in: Proceedings of CIKM 2003 (12th ACM International Conference on Information and Knowledge Management), New Orleans, LA, pp. 556–559.
[61] Q. Mei, D. Zhou, K. Church, Query suggestion using hitting time, in: Proceeding of CIKM 2008 (17th ACM International Conference on Information and Knowledge Management), Napa Valley, CA, pp. 469–478.
[62] P. Sarkar, A. Moore, A tractable approach to finding closest truncated-commute-time neighbors in large graphs, in: Proceedings of UAI 2007 (23rd Conference on Uncertainty in Artificial Intelligence), Vancouver, BC, pp. 335–343.
[63] Metaweb Technologies, Freebase Wikipedia Extraction (WEX), http://download.freebase.com/wex/, 2010.
[64] V. Grishchenko, Wikipedia as an ant-hill, Web page consulted on April 15, 2012: http://no-gritzko-here.livejournal.com/22900.html, 2008.
[65] D. J. Watts, S. H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (1998) 440–442.
[66] T. Opsahl, P. Panzarasa, Clustering in weighted networks, Social Networks 31 (2009) 155–163.
[67] D. Bollegala, Y. Matsuo, M. Ishizuka, Measuring semantic similarity between words using web search engines, in: Proceedings of WWW 2007 (16th International Conference on World Wide Web), Banff, Alberta, pp. 757–766.
[68] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, A. Soroa, A study on similarity and relatedness using distributional and WordNet-based approaches, in: Proceedings of HLT-NAACL 2009 (Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the ACL), Boulder, CO, pp. 19–27.
[69] H. Rubenstein, J. B. Goodenough, Contextual correlates of synonymy, Communications of the ACM 8 (1965) 627–633.
[70] G. A. Miller, W. G. Charles, Contextual correlates of semantic similarity, Language and Cognitive Processes 6 (1991) 1–28.
[71] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, E. Ruppin, Placing search in context: The concept revisited, ACM Transactions on Information Systems (TOIS) 20 (2002) 116–131.
[72] Z. Wu, M. Palmer, Verb semantics and lexical selection, in: Proceedings of ACL 1994 (32nd Annual Meeting of Association for Computational Linguistics), Las Cruces, NM, pp. 133–138.
[73] S. Patwardhan, T. Pedersen, Using WordNet-based context vectors to estimate the semantic relatedness of concepts, in: Proceedings of the EACL 2006 Workshop on Making Sense of Sense, Trento, Italy, pp. 1–8.
[74] M. A. Alvarez, S. J. Lim, A graph modeling of semantic similarity between words, in: Proceedings of ICSC 2007 (1st International Conference on Semantic Computing), Irvine, CA, pp. 355–362.
[75] M. D. Lee, B. Pincombe, M. Welsh, An empirical evaluation of models of text document similarity, in: Proceedings of CogSci 2005 (27th Annual Conference of the Cognitive Science Society), Stresa, Italy, pp. 1254–1259.
[76] S. Hassan, R. Mihalcea, Semantic relatedness using salient semantic analysis, in: Proceedings of AAAI 2011 (25th AAAI Conference on Artificial Intelligence), San Francisco, CA, pp. 884–889.
[77] T. M. Mitchell, Machine Learning, McGraw-Hill, New York, NY, 1997.
[78] X. Hu, X. Zhang, C. Lu, E. Park, X. Zhou, Exploiting Wikipedia as external knowledge for document clustering, in: Proceedings of KDD 2009 (15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining), Paris, France, pp. 389–396.
[79] G. Bordogna, G. Pasi, Hierarchical-hyperspherical divisive fuzzy C-means (H2D-FCM) clustering for information retrieval, in: Proceedings of WI-IAT 2009 (IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology), volume 1, Milan, Italy, pp. 614–621.
[80] K. Weinberger, J. Blitzer, L. Saul, Distance metric learning for large margin nearest neighbor classification, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems 18, The MIT Press, Cambridge, MA, 2006, pp. 1473–1480.
[81] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) article n. 27.
[82] M. Ranzato, M. Szummer, Semi-supervised learning of compact document representations with deep networks, in: Proceedings of ICML 2008 (25th International Conference on Machine Learning), Helsinki, Finland, pp. 792–799.

[83] E. M. Voorhees, D. Harman, Overview of the Eighth Text REtrieval Conference (TREC-8), in: Proceedings of TREC-8, Gaithersburg, MD, pp. 1–24.
[84] O. Egozi, S. Markovitch, E. Gabrilovich, Concept-based information retrieval using explicit semantic analysis, ACM Transactions on Information Systems (TOIS) 29 (2011) article n. 8.
[85] H. Li, Learning to Rank for Information Retrieval and Natural Language Processing, Morgan and Claypool, San Rafael, CA, 2011.
[86] W.-T. Yih, K. Toutanova, J. C. Platt, C. Meek, Learning discriminative projections for text similarity measures, in: Proceedings of CoNLL 2011 (15th Conference on Computational Natural Language Learning), Portland, OR, pp. 247–256.
[87] K. Collins-Thompson, J. Callan, Query expansion using random walk models, in: Proceedings of CIKM 2005 (14th ACM Conference on Information and Knowledge Management), Bremen, Germany, pp. 704–711.
[88] P. Sarkar, A. W. Moore, A. Prakash, Fast incremental proximity search in large graphs, in: Proceedings of ICML 2008 (25th International Conference on Machine Learning), Helsinki, Finland, pp. 896–903.