Using a Wikipedia-based Semantic Relatedness Measure for Document Clustering

Majid Yazdani
Idiap Research Institute and EPFL
Centre du Parc, Rue Marconi 19
1920 Martigny, Switzerland
majid.yazdani@idiap.ch

Andrei Popescu-Belis
Idiap Research Institute
Centre du Parc, Rue Marconi 19
1920 Martigny, Switzerland
andrei.popescu-belis@idiap.ch

Abstract

A graph-based distance between Wikipedia articles is defined using a random walk model, which estimates visiting probability (VP) between articles using two types of links: hyperlinks and lexical similarity relations. The VP to and from a set of articles is then computed, and approximations are proposed to make tractable the computation of semantic relatedness between every two texts in a large data set. The model is applied to document clustering on the 20 Newsgroups data set. Precision and recall are improved in comparison with previous textual distance algorithms.

1 Introduction

Many approaches have been proposed to compute similarity between texts, from lexical overlap measures to statistical topic models that are learned from large corpora. In this paper, we propose a method for using knowledge from a structured, collaborative resource – the Wikipedia hypertext encyclopedia – in order to build a measure of semantic relatedness that we test on a text clustering task.

The paper first describes the document graph derived from Wikipedia (Section 2), and then defines a network-based distance using visiting probability (Section 3), along with algorithms for its application to text clustering (Section 4). Results over the 20 Newsgroups dataset are shown to be competitive (Section 5), and the relative contributions of cosine lexical similarity and visiting probability are analyzed. Our proposal is discussed in the light of previous work in Section 6.

2 The Document Network

In the present proposal, knowledge about semantic relatedness is embodied into a document network, whose nodes are intended to represent concepts, while the links between nodes stand for various relations between concepts. The nodes of the network correspond to articles from the Wikipedia hypertext encyclopedia, and are derived as follows.

The network was built from Wikipedia, using the WEX dataset (Metaweb Technologies, 2010). All articles from the following categories were removed, as they do not correspond to proper concepts: Talk, File, Image, Template, Category, Portal, and List. Moreover, disambiguation pages and articles shorter than 100 non-stopwords were filtered out as well. Out of 4,327,482 articles in WEX, 1,264,611 articles were kept, forming the nodes of our network.

The first type of links in our document network are the hyperlinks between articles, because, in principle, each link between two articles indicates some form of relatedness between them. There are more than 35 million such links in our network.

The second type of links is derived from the similarity of lexical content between articles. This is computed using cosine similarity between the lexical vectors corresponding to the articles' texts, after stopword removal and stemming. Then, links are created by connecting every article to the 10 articles that are most similar to it, each link receiving a weight which is the normalized lexical similarity score. The number 10 was chosen to ensure computational tractability, and is in the same range as the average number of hyperlinks per node (30).

Proceedings of the TextGraphs-6 Workshop, pages 29–36,
Portland, Oregon, USA, 19-24 June 2011.

Computing semantic relatedness between two texts requires: (1) to estimate relatedness between two sets of nodes in the network, as described in Sections 3 and 4; and (2) to project each text onto a set of nodes, as we briefly explain here. The projection of a text onto the network is found by computing the text's lexical similarity with all articles, again using cosine distance over stemmed words, without stopwords. The text is mapped to the 10 closest articles, resulting in a probability distribution over the 10 corresponding nodes. Again, this value was chosen to be similar to the number of hyperlinks and content links per node, and to keep computation tractable. In fact, the numerous Wikipedia articles are scattered in the space of words, therefore tuning these values does not seem to bring crucial changes.

3 Computing Relatedness in the Network Using Visiting Probability (VP)

We have previously defined a random walk model (Yazdani and Popescu-Belis, 2010) to compute relatedness of sets of nodes as the visiting probability (VP) of a random walker from one set to another one, and we will review the model in this section. In the next section, we will explain how the model was extended for application to document clustering.

3.1 Notations

Let $S = \{s_i | 1 \leq i \leq n\}$ be the set of n nodes in the graph. Any two nodes 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). Links between nodes can thus be represented by L matrices A_l ($1 \leq l \leq L$) 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. The *transition matrix* C_l gives the probability of a direct transition between nodes 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 giving the transition probability $C_{i,j}$ between nodes s_i and s_j is : $C = \sum_{l=1}^{L} w_l C_l$.

One of the main parameters in this computation is the relative weight of the two types of links (lexical similarity and hyperlinks) in the random walk over the network. The settings for the experiments on document clustering (0.6 vs. 0.4) are explained in Section 5.1 below.

3.2 VP from a Set of Nodes to a Node

Let us consider a probability distribution $\vec{r}$ over nodes, corresponding to the projection of a text fragment onto the network of articles (Section 2). Given a new node s_j in the network, our model first estimates the probability of visiting s_j for the first time when a random walker starts from $\vec{r}$ in the graph. The model considers the state S_t of the random walker (its position node) and provides a procedure which, executed until termination, yields the value of VP. Namely, the initial state is chosen at random with probability $P(S_0 = s_i|\vec{r}) = r_i$ (where the r_i are the components of $\vec{r}$). Then, from state S_{t-1}, either $S_{t-1} = s_j$ and the procedure is finished, or the next node is chosen using the transition matrix C. Moreover, it is also possible to 'fail' the walk with a small probability, called 'absorption probability', which makes longer paths less probable.

3.3 Differences between VP and PageRank or Hitting Time

The VP of s_j starting from the distribution $\vec{r}$, as computed here, is different from the probability assigned to s_j after running Personalized PageRank (Haveliwala, 2003) with a teleport vector equal to $\vec{r}$. In the computation of VP, the loops starting from s_j and ending to the same s_j do not have any effect on the final score, unlike for PPR, for which such loops boost the probability of s_j. If some pages have this type of loops (typically, very "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 VP 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 starting from $\vec{r}$. Hitting time is more sensitive to long paths in comparison to VP, a fact that might introduce more noise.The performance of these three algorithms in computing se-

mantic similarity has been compared in (Yazdani and Popescu-Belis, 2010).

3.4 VP between Sets of Nodes

Generalizing now to the computation of VP from a weighted set of nodes $\vec{r}_1$ (a probability distribution) to another set $\vec{r}_2$, the model first constructs a virtual node representing $\vec{r}_2$ in the network, named by convention s_R, and then connects all nodes s_i to s_R according to their weights in $\vec{r}_2$. The transition matrix for the random walk is updated accordingly.

To compute relatedness of two texts projected onto the network as $\vec{r}_1$ and $\vec{r}_2$, the VP of $\vec{r}_1$ given $\vec{r}_2$ is averaged with the converse probability, of $\vec{r}_2$ given $\vec{r}_1$ – a larger probability indicating closer semantic relatedness.

3.5 Truncated VP

The computation of VP can be done iteratively and can be truncated after a number of steps, as the importance of longer paths grows smaller due to the absorption probability, leading thus to a T-truncated visiting probability noted VP^T. Besides making computation more tractable, truncation reduces the effect of longer paths, which seem to be less reliable indicators of relatedness.

We have computed an upper bound on the truncation error, which helps to control and minimize the number of steps actually computed in a random walk. To compute the upper bound of the truncation error we compute the probability of returning neither success (reaching s_j) nor failure (absorption) in first t steps, which 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. C' is the transition matrix that gives the probability of a transition between two nodes, modified to include the virtual node s_R in the network, and $1-\alpha$ is the absorption probability.

If $p_t(success)$ denotes the probability of success (reaching s_j) considering paths of length at most t, and ε_t the error made by truncating after step t, then we have:

$$\varepsilon_t = p(success) - p_t(success) \leq \sum_{i \neq j}^{n} \alpha^t(\vec{r}C'^t)_i$$

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.

4 Application of VP to Text Clustering

In this section, we describe the additional modeling that was done so that semantic relatedness based on VP could be applied efficiently to text clustering. Indeed, it is not tractable to individually compute the average VP between any two texts in the set of documents to be clustered, because the numbers of pairs is very large – e.g., 20,000 documents in the experiments in Section 5. Instead, we propose two solutions for computing, respectively, VP *to* a set of nodes (from all documents in the network), and respectively VP *from* a set of nodes to all documents.

4.1 Computing VP from All Nodes to a Subset

To compute the T-truncated visiting probability (noted VP^T) from all nodes in the network to a node s_R at the same time, the following recursive procedure is defined. Here, T is the number of steps before truncation, and s_R is a virtual node representing a probability distribution $\vec{r}$ from a text. The procedure is based on the definition of VP between nodes in Section 3 and uses the transition matrix C' that gives the probability of a transition between two nodes, modified to include the virtual node s_R in the network. If $1-\alpha$ is the absorption probability, then the recursive definition of VP^T from a node s_i to the virtual node s_R is:

$$VP^T(s_i, s_R) = \alpha \sum_k C'(s_i, s_k) VP^{T-1}(s_k, s_R)$$

Using dynamic programming, it is possible to compute VP^T from all nodes to s_R in $O(ET)$ steps, where E is the number of links in the network. The initialization of the procedure is done using $VP^T(s_R, s_R) = 1$ and $VP^0(s_i, s_R) = 0$ for any $i \neq R$.

4.2 Computing VP from a Subset to All Nodes

To compute the truncated VP *from* $\vec{r}$ to all nodes in the network, the total computation time using the definition of VP^T from Section 3 is $O(ETN)$, where N is the number of nodes in the network, because VP^T must be computed for each node separately. For a large data set, this is not tractable.

The proposed solution is based on a sampling method over the random walks to approximate VP^T. The sampling involves running M independent random walks of length T from $\vec{r}$. For a given

node s_j and a sample walk m, the first time (if any) when s_j is visited on each random walk starting from $\vec{r}$ is noted t_{j_m}. Then, VP^T can be estimated by the following average over sample walks, where $1-\alpha$ is again the absorption probability:

$$\hat{VP^T}(\vec{r}, s_j) = (\sum_m \alpha^{t_{j_m}})/M.$$

As a result, the estimate of VP^T can be computed in $O(MT)$ steps, where M is the number of sample paths.

Moreover, it is possible to compute a bound on the error of the estimation, $|VP^T - \hat{VP^T}|$, depending on the number of sample paths M. It can be shown that the error is lower than ε, with a probability larger than $1-\delta$, on condition that the number of sample paths is greater than $\alpha^2 \ln(2/\delta)/2\varepsilon^2$.

To prove this bound, we use inspiration from a proof by Sarkar et al. (2008). If the estimation of a variable X is noted $\hat{X}$, let us suppose that concept s_j has been visited for the first time at $\{t_{j_1}, \cdots, t_{j_m}\}$ time steps in the M sample walks. We define the random variable X^l by $\alpha^{t_{j_l}}/M$, where t_{j_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 ($j_1 \leq l \leq j_m$) are independent and bounded by 0 and 1 ($0 \leq X^l \leq 1$). We have:

$$\hat{VP^T}(\vec{r}, s_j) = \sum_l X^l = (\sum_l \alpha^{t_{j_l}})/M \text{ and}$$

$$E(\hat{VP^T}(\vec{r}, s_j)) = VP^T(\vec{r}, s_j).$$

So, by applying Hoeffding's inequality, we have:

$$P(|\hat{VP^T} - E(\hat{VP^T})| \geq \epsilon) \leq 2exp(-\tfrac{2M\epsilon^2}{\alpha^2}).$$

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 want ε-approximation for all possible s_j with probability at least $1-\delta$. We use union bound and Hoeffding's inequality:

$$P(\exists j \in \{1, \ldots, n\}, |\hat{VP^T} - E(\hat{VP^T})| \geq \epsilon) \leq 2n \times exp(-\tfrac{2M\epsilon^2}{\alpha^2})$$

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

5 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.[1] 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 document network and the techniques for computing distances between two texts projected onto it.

5.1 Setup of the Experiment

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 $T = 5$ that gives sufficient precision; a larger value only increased computation time. Instead of computing VP^T between all possible pairs separately, we fill one row of the matrix at a time using the approximations above.

We set the absorption probability of the random walk $1 - \alpha = 0.2$ for this experiment. Given α and T by using the formula in section 3.5, it is possible to compute the error bound of the truncation, and noting that for a smaller α, fewer steps (T) are needed to achieve the same approximation precision because of the penalty set to longer paths. Conversely, a larger α decreases the penalty for longer paths and requires more computation.[2]

For comparison purposes, four similarity matrices were computed. Indeed, the theoretical apparatus described above can be applied to various types of links in the document network. In Section 2, we introduced two types of links, namely lexical similarity and actual hyperlinks, and these can be used separately in the model, or as a weighted combination. The following similarities will be compared:

1. VP over hyperlinks only (noted VP_{Hyp});
2. VP over lexical similarity links (VP_{Lex});

[1] Distributed at `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/news20.html`, see also (Mitchell, 1997, Chapter 6).

[2] Note that in the extreme case when $\alpha = 0$, similarity to all nodes except the node itself is zero.

3. VP over a combination of hyperlinks (0.4) and lexical links (0.6) (noted VP_{Comb}) – these values gave the best results in our previous applications to word and document similarity tasks (Yazdani and Popescu-Belis, 2010);

4. no random walk, only cosine similarity between the tf-idf vectors of the documents to be clustered (noted LS, for lexical similarity).

5.2 Clustering Performance

Clustering is performed using a k-means algorithm over each of the four similarity matrices.[3] The quality of the clustering is first 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. Other methods exist (Pantel and Lin, 2002), including a Rand Index adjusted for chance (Vinh et al., 2009), but the RI suffices for comparative judgments in this subsection. However, in Subsection 5.3, we will also look at precision and recall, and in Subsection 5.4 we will use purity. As the clustering is performed over the entire data set, because there is no training vs. test data, confidence intervals are not available, though they could be computed by splitting the data. As a result, comparison with other scores on the same test set is absolute.

The scores in terms of Rand Index are, in decreasing order:

1. 90.8% for VP_{Comb}
2. 90.6% for VP_{Hyp}
3. 90.4% for VP_{Lex}
4. and only 86.1% for the LS cosine similarity.

The random walk model thus clearly outperforms the baseline LS approach. If counting only wrongly clustered document pairs, VP_{Comb} has 6.6% of such pairs, while VP_{Lex} has 8.4%, confirming the lower performance of the model using only lexical similarity links, i.e. the utility of hyperlinks.

[3]The semantic relatedness measure proposed here could be used with other clustering algorithms, such as the committee-based method proposed by Pantel and Lin (2002).

5.3 Comparison to Other Methods

To obtain a better understanding of the performance of the proposed method, we computed the clustering precision and recall of several well-known methods for statistical text representation, shown in Table 1. For Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and Latent Semantic Analysis (LSA) (Deerwester et al., 1990), we first mapped the documents in the latent space and then computed the cosine similarity between the documents in the latent space. The number of topics for LSA and LDA is set to 100 to make the computation tractable. Precision and recall are used, rather than the Rand Index, to show in more detail the performance of each method. The use of VP over our document network clearly increases both precision and recall in comparison to other tested approaches.

Similarity method	Precision	Recall
LS	7.50	18.38
LSA	8.63	9.99
LDA	19.93	31.50
VP_{Comb}	**23.81**	**35.32**

Table 1: Precision and Recall for k-means clustering over the 20 Newsgroups using several well-known methods to compute text similarity, in comparison to the present proposal.

5.4 Analysis of the Impact of VP with Respect to 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, noted $w \times VP_{Comb} + (1 - w) \times LS$, 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 values of k, is shown in Figure 1.

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. Further-

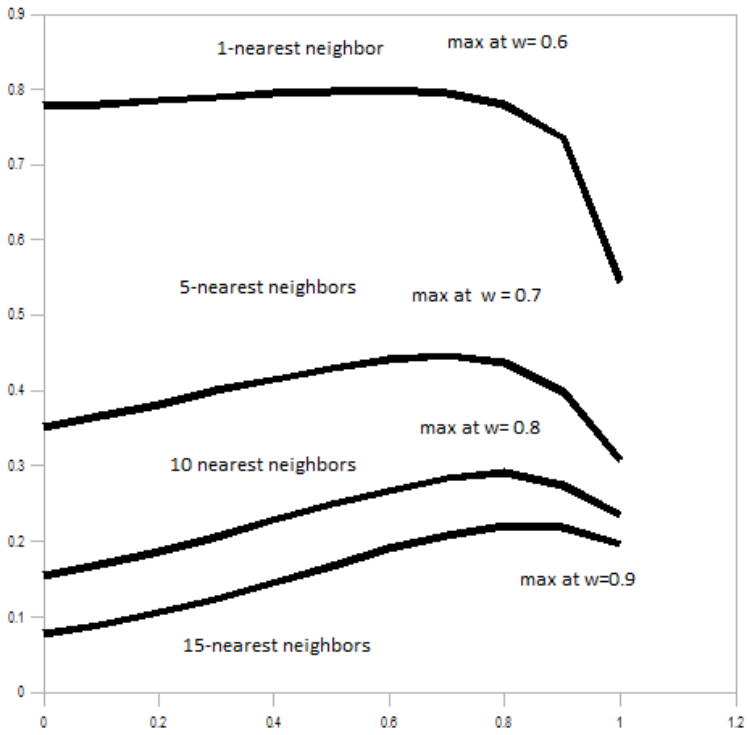

Figure 1: Values of k-purity (vertical axis) averaged over all documents, for neighborhoods of different sizes k. The horizontal axis indicates the weight w of visiting probability vs. cosine lexical similarity in the formula: $w \times VP_{Comb} + (1 - w) \times LS$.

more, when the size of the examined neighborhood k increases (lower curves in Figure 1), 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.

6 Related Work

Many attempts have been made to improve the overlap-based lexical similarity distance, for various applications to HLT. One approach is to construct a taxonomy of concepts and relations (manually or automatically) and to map the text fragments to be compared onto the taxonomy. For instance, Wordnet (Fellbaum, 1998) and Cyc (Lenat, 1995) are two well-known knowledge bases that can be used for enriching pure lexical matching. However, building and maintaining such resources requires considerable effort, and they might cover only a fraction of the vocabulary of a language, as they usually include few proper names or technical terminology.

Another approach makes use of unsupervised methods to construct a semantic representation of

documents by analyzing mainly co-occurrence relationships between words in a corpus. Latent Semantic Analysis (Deerwester et al., 1990), Probabilistic LSA (Hofmann, 1999) and Latent Dirichlet Allocation (Blei et al., 2003) are unsupervised methods that construct a low-dimensional feature representation or concept space, in which words are no longer supposed to be independent.

Mihalcea et al. (2006) compared knowledge-based and corpus-based methods, using word similarity and word specificity to define one general measure of text semantic similarity. Because it computes word similarity values between all word pairs, the proposed method appears to be suitable mainly to compute similarity between short fragments, otherwise the computation becomes intractable.

WikiRelate! (Strube and Ponzetto, 2006) computes semantic relatedness between two words by using Wikipedia. Each word is mapped to the corresponding Wikipedia article by using article titles. To compute relatedness, several methods are proposed, namely, using paths in the Wikipedia category structure or the articles' content. Our method, by comparison, also uses the knowledge embedded in the hyperlinks between articles, as well as the entire contents of articles, but unlike WikiRelate! it has been extended to texts of arbitrary lengths.

Explicit Semantic Analysis (Gabrilovich and Markovitch, 2007), 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. Similarity is measured in the new concept space. ESA does not use the link structure or other structured knowledge from Wikipedia. Moreover, by walking over a content similarity graph, our method benefits from a non-linear distance measure according to the paths consisting of small neighborhoods.

In the work of Yeh et al. (2009), a graph of documents and hyperlinks is computed from Wikipedia, then a Personalized PageRank (Haveliwala, 2003) is computed for each text fragment, with the teleport vector being the one resulting from the ESA algorithm cited above. To compute semantic relatedness between two texts, Yeh et al. (2009) simply compare their personalized page rank vectors. By comparison, in our method, we also consider in addition to hyperlinks the effect of word co-occurrence between article contents. The use of visiting probability also gives different results over personalized page rank, as it measures different properties of the network.

There are many studies on measuring distances between vertices in a graph. Two measures that are close to the visiting probability proposed here are hitting time and Personalized PageRank mentioned in Section 3.3. Hitting time has been used in various studies as a distance measure in graphs, e.g. for dimensionality reduction (Saerens et al., 2004) or for collaborative filtering in a recommender system (Brand, 2005). Hitting time was also used for link prediction in social networks along with other distances (Liben-Nowell and Kleinberg, 2003), or for semantic query suggestion using a query/URL bipartite graph (Mei et al., 2008). As for Personalized PageRank, it was used for word sense disambiguation (Agirre and Soroa, 2009), and for measuring lexical relatedness of words in a graph built from WordNet (Hughes and Ramage, 2007).

7 Conclusion

We proposed a model for measuring text semantic relatedness based on knowledge embodied in Wikipedia, seen here as document network with two types of links – hyperlinks and lexical similarity ones. We have used visiting probability to measure proximity between weighted sets of nodes, and have proposed approximation algorithms to make computation efficient for large graphs (more than one million nodes and 40 million links) and large text clustering datasets (20,000 documents in 20 Newsgroups). Results on the document clustering task showed an improvement using both word co-occurrence information and user-defined hyperlinks between articles over other methods for text representation.

Acknowledgments

The work presented in this paper has been supported by the IM2 NCCR (Interactive Multimodal Information Management) of the Swiss National Science Foundation (http://www.im2.ch).

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of EACL 2009 (12th Conference of the European Chapter of the Association for Computational Linguistics)*, pages 33–41, Athens, Greece.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Matthew Brand. 2005. A random walks perspective on maximizing satisfaction and profit. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, Newport Beach, CA.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Christiane Fellbaum, editor. 1998. *WordNet: An electronic lexical database*. MIT Press, Cambridge, MA.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of IJCAI 2007 (20th International Joint Conference on Artificial Intelligence)*, pages 6–12, Hyderabad.

Taher H. Haveliwala. 2003. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, 15:784–796.

Thomas Hofmann. 1999. Probabilistic Latent Semantic Indexing. In *Proceedings of SIGIR 1999 (22nd ACM SIGIR Conference on Research and Development in Information Retrieval)*, pages 50–57, Berkeley, CA.

Thad Hughes and Daniel Ramage. 2007. 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)*, pages 581–589, Prague.

Douglas B. Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of CIKM 2003 (12th ACM International Conference on Information and Knowledge Management)*, pages 556–559, New Orleans, LA.

Qiaozhu Mei, Dengyong Zhou, and Kenneth Church. 2008. Query suggestion using hitting time. In *Proceeding of CIKM 2008 (17th ACM International Conference on Information and Knowledge Management)*, pages 469–478, Napa Valley, CA.

Metaweb Technologies. 2010. Freebase Wikipedia Extraction (WEX). http://download.freebase.com/wex/.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of AAAI 2006 (21st National Conference on Artificial Intelligence)*, pages 775–782, Boston, MA.

Tom M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, New York.

Patrick Pantel and Dekang Lin. 2002. Document clustering with committees. In *Proceedings of SIGIR 2002 (25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval)*, pages 199–206, Tampere.

Marco Saerens, Franois Fouss, Luh Yen, and Pierre Dupont. 2004. The principal components analysis of a graph, and its relationships to spectral clustering. In *Proceedings of ECML 2004 (15th European Conference on Machine Learning)*, pages 371–383, Pisa.

Purnamrita Sarkar, Andrew W. Moore, and Amit Prakash. 2008. Fast incremental proximity search in large graphs. In *Proceedings of ICML 2008 (25th International Conference on Machine Learning)*, pages 896–903, Helsinki.

Michael Strube and Simone Paolo Ponzetto. 2006. Wikirelate! Computing semantic relatedness using Wikipedia. In *Proceedings of AAAI 2006 (21st National Conference on Artificial Intelligence)*, pages 1419–1424.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In *Proceedings of ICML 2009 (26th International Conference on Machine Learning)*, Montreal.

Majid Yazdani and Andrei Popescu-Belis. 2010. 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.

Eric Yeh, Daniel Ramage, Christopher D. Manning, Eneko Agirre, and Aitor Soroa. 2009. WikiWalk: random walks on Wikipedia for semantic relatedness. In *Proceedings of TextGraphs-4 (4th Workshop on Graph-based Methods for Natural Language Processing)*, pages 41–49, Singapore.