IDIAP RESEARCH REPORT

Inferring Document Similarity from Hyper-links

David Grangier [1]   Samy Bengio [2]

IDIAP–RR 05-21

May 2005

submitted for publication

[1] IDIAP, CP 592, 1920 Martigny, Switzerland, grangier@idiap.ch

[2] IDIAP, CP 592, 1920 Martigny, Switzerland, bengio@idiap.ch

IDIAP Research Institute **www.idiap.ch**

Rue du Simplon 4 P.O. Box 592 1920 Martigny – Switzerland

Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

IDIAP Research Report 05-21

Inferring Document Similarity from Hyper-links

David Grangier Samy Bengio

May 2005

submitted for publication

Abstract. Assessing semantic similarity between text documents is a crucial aspect in Information Retrieval systems. In this paper, we propose a technique to derive a similarity measure from hyper-link information. As linked documents are generally semantically closer than unlinked documents, we use a training corpus with hyper-links to infer a function $a, b \rightarrow sim(a, b)$ that assigns a higher value to linked documents than to unlinked ones. Two sets of experiments on different corpora show that this function compares favorably with *OKAPI* matching on document retrieval tasks.

1 Introduction

Automatic techniques to access and organize document collections are essential to fully benefit from large text corpora. Several of these methods require a measure to quantify semantic similarities between text items: e.g. clustering relies on document comparisons, while Information Retrieval (IR) depends on document/query similarities.

In this paper, we propose to infer a measure of similarity from hyper-linked documents. As stated in previous work [12], the presence of a link between two documents can be considered as an indicator of topic relatedness, i.e. linked documents tend to be semantically closer than unlinked documents. The goal of this paper is hence to identify a measure of similarity $a, b \rightarrow sim(a, b)$ such that, for any document d, the documents which are linked to it are considered more similar than those which are not:

$$\forall d, \forall l^+ \in L(d), \forall l^- \notin L(d), sim(d, l^+) > sim(d, l^-) \tag{1}$$

where $L(d)$ is the set of documents linked with d.

For that purpose, a gradient descent strategy [16] is adopted: we first introduce a parameterized measure of similarity $a, b \rightarrow sim_\theta(a, b)$ and a cost C which indicates how far sim_θ is from the condition (1), then gradient descent optimization is used to select the parameters θ which minimize C for a given training corpus D_{train}.

To evaluate this approach, the inferred similarity measure is compared with the state-of-the-art *OKAPI* matching measure [19] on a retrieval task (see Section 4): this task consists of finding the documents which are related to a given document (like the *similar pages* function of web search engines, e.g. [1]). In this context, our model *LinkLearn* is shown to improve both precision at top 10 documents and average precision (resp. 18% and 17% relative improvement). Another set of experiments performed over a TREC retrieval task (i.e. TREC9 queries for TDT-2 corpus) confirms these results. The remainder of this paper is organized as follows: Section 2 presents the related previous approaches, Section 3 describes the proposed method, Section 4 presents the experiments and results, and Section 5 draws some conclusions.

2 Related Work

The assessment of semantic similarity between text items has mostly been investigated in the context of IR [3]. In this framework, the documents are generally ranked according to their similarity with the query and the similarity measure used is hence a key aspect of such systems.

The first part of this section presents different approaches that have been used to assess document similarity (or document/query similarity) while the second part presents how hyper-link information has been used to improve such approaches.

2.1 Assessing Document Similarity

In most IR systems, documents and queries are compared according to the inner product of their vector representations [3]. Each document is represented by a vocabulary sized vector $d = (d_1, \ldots, d_V)$ where d_i denotes the weight of term i in the document and V is the vocabulary size. Similarly, each query is also assigned a vector representation $q = (q_1, \ldots, q_V)$ and document/query similarity is then computed as:

$$sim(q, d) = q \cdot d = \sum_{i=1}^{V} q_i d_i. \tag{2}$$

In order to improve the retrieval performance (i.e. to reliably estimate document/query similarity), different weighting schemes have been investigated. Query and document components can be, amongst others, binary values (stating the presence or absence of a term) or different variants of the weights known as $tf \cdot idf$:

$$tf_{d,i} \cdot idf_i = tf_{d,i} \cdot log(\frac{N}{df_i})$$

where $tf_{d,i}$ denotes the number of occurrences of term i in document d, N is the number of documents in the corpus, and df_i is the number of documents containing the term i.

Actually, in state-of-the-art IR systems, those weights are generally binary for the query and $OKAPI\ BM25$ weights for the documents [19]:

$$d_i = \frac{(K+1) \cdot tf_{i,d} \cdot idf_i}{K \cdot ((1-B) + B \cdot ndl_d) + tf_{i,d}}$$

where ndl_d is the normalized document length (i.e. the length of document d divided by the average document length) and K, B are two hyper-parameters (K is used to avoid the over-weighting of repetitions, while B is used to reduce the influence of document length). The document/query matching then becomes:

$$sim(q,d) = \sum_{i=1}^{V} q_i \cdot \frac{(K+1) \cdot tf_{i,d} \cdot idf_i}{K \cdot ((1-B) + B \cdot ndl_d) + tf_{i,d}} .$$

Several data-driven alternatives to these *a priori* weighting strategies have been proposed in the literature, the ultimate goal being to infer a more effective similarity measure (or document representation) from a training collection. Different types of approaches can be categorized according to their optimized criterion: least-square algorithms, maximum likelihood algorithms and distance learning methods.

Latent Semantic Analysis (LSA) is a well known least-square approach which applies Singular Value Decomposition (SVD) to the term-documents matrix X [13]. Given $k < V$, the goal of LSA is to find the least-square best approximation of X by a matrix of rank k. Each document can then be represented in a k dimensional space and document similarity can then be computed through the inner product in this new space. In some cases, this approach has shown to be more effective than the inner product in the original space since LSA captures term/term statistical relationships and documents are not compared on their set of shared terms alone. However, the relationship between the optimized criterion and the desired properties of the similarity measure is not clear and, moreover, the computational cost of SVD makes the application of LSA over large collections difficult.

Maximum likelihood approaches consist of probabilistic models which hypothesize that the observation of a term i in a document d results from a hidden generative process. Different models that assume different underlying processes have been proposed in the literature, including Probabilistic LSA (PLSA) [14], Latent Dirichlet Allocation (LDA) [5] and Multinomial Principal Component Analysis (MPCA) [7]. The parameters of these models are estimated through a maximum likelihood procedure, i.e. the parameters which maximizes the training data likelihood are selected. A document d and a query q can then be compared according to the estimate of the probability of q knowing d, $P(q|d)$. Like LSA, these models often lead to better results than empirical approaches but there is no guarantee that likelihood maximization should lead to a more suitable similarity measure.

To overcome this weakness, distance learning methods attempt to infer a distance metric through the explicit optimization of a cost related to the desired properties of the metric. In [21], the training data consists of pairs of data points that should be close according to the derived distance and parameter learning is performed through gradient descent. This method has been applied successfully to the clustering of different types of data such as breast cancer diagnoses or protein structures. Another distance learning approach has been applied to text data: ranking SVMs have been used to infer a distance between text items [20]. In this case, the training labels consist of preference constraints of the type: "query q should be closer to document d^+ than document d^-". Such constraints are similar to our condition (1): this model and the one we propose can hence be applied to similar tasks. However, in ranking SVM, parameter fitting is performed through quadratic optimization which makes this approach costly compared to a gradient descent procedure such as the one we propose. Hence, unlike ranking SVMs, our approach can be applied to very large constraint sets, as shown in Section 4.

2.2 Benefit from Linkage Information

As stated in [12], hyper-links convey semantic similarity information: the authors do not create links randomly, but use them to associate topically related documents. This linkage information can thus be used to improve the way document similarity is assessed and different methods to benefit from links have already been investigated.

In [6], document expansion is used: the weight of a term i in a document d is increased when the term i occurs in links pointing to d. Hence, according to inner product comparison, two linked documents are generally considered more similar than two unlinked documents. In [8], a different document expansion technique is investigated. A thesaurus is first built from link information: document connectivity is used to extract pairs of related terms (i.e. terms which frequently occur in linked documents but rarely occur in unlinked documents). Then, documents or IR queries can be expanded by adding terms which are related to those they already contain. Unlike the previous technique, this method relies on the whole link structure rather than focusing only on the links of the expanded document, which allows, for example, the expansion of text items with no link, like IR queries.

Linkage information has also been used in contexts other than document/query expansion. In [18], document/query similarity is computed iteratively relying on the following idea: the similarity between a document d and a query q, $sim(d, q)$, should depend on the similarity between q and the documents linked with d. Hence, in this framework, a document which is connected to documents highly similar to q, will itself be assigned an higher similarity value with respect to q. This approach, which is an extension of the page rank algorithm [15], hypothesizes that, among the documents having content similar to the query, the most valuable are those which are referred to by the others.

The PLSA model [14] has also been extended to benefit from hyper-link data [10]: in this case, a link to a document d is considered as a kind of new index term l_d and hence, when computing document similarity, document sharing pointers to the same references are considered closer.

Therefore, the above approaches all consist of modifying the way similarity values are computed such that condition (1) is satisfied for most documents. These modifications allow the similarity measure to be closer to human assessments which states that linked pages are more likely to be about the same topic than unlinked ones [12]. The model we propose also share the same goal, however our approach differs significantly from previous work: rather than relying on heuristics which make the similarity measure closer to (1), we define a loss function which penalizes measures that do not satisfy most of the constraints of (1) and we then pick, within a family of parameterized measure sim_θ, the function sim_{θ^*} which minimizes this loss. This explicit optimization of a cost related to (1) is shown to be effective experimentally (see Section 4).

3 The *LinkLearn* Model

This section describes the *LinkLearn* model that we propose in order to infer a document similarity measure from a hyper-linked corpus: a cost C related to (1) is first defined, the parameterization of the similarity measure is then introduced and the training procedure to select the parameters which minimize C is described.

3.1 Similarity Constraint Criterion

As mentioned above, it is desirable that, for any document d, the documents which are linked to it (i.e. the documents of $L(d)$) are considered more similar to d than any other documents (1). A simple cost would hence be the proportion of document triplet $d \in D_{train}$, $l^+ \in L(d)$, $l^- \notin L(d)$ for which the above property is not satisfied:

$$C^{0/1} = \frac{1}{|D_{train}|} \sum_{d \in D_{train}} C_d^{0/1} \tag{3}$$

where

$$C_d^{0/1} = \frac{1}{|L(d)| \cdot |\overline{L(d)}|} \sum_{\substack{l^+ \in L(d) \\ l^- \in \overline{L(d)}}} I\{sim(d,l^+) < sim(d,l^-)\},$$

$I\{\cdot\}$ is the indicator function, i.e. $I\{c\} = 1$ if c is true and zero otherwise and $L(d)$ is the set of documents linked with d (i.e. the documents referring to d and the documents referred to by d).

Similarly to the 0/1 loss (i.e. error rate) in the case of classification, $C^{0/1}$ cannot be directly minimized through gradient descent [4]. Hence, we propose to minimize an upper bound of this quantity:

$$C = \frac{1}{|D_{train}|} \sum_{d \in D_{train}} C_d \tag{4}$$

where

$$C_d = \frac{1}{|L(d)| \cdot |\overline{L(d)}|} \sum_{\substack{l^+ \in L(d) \\ l^- \in \overline{L(d)}}} \|1 - sim(d,l^+) + sim(d,l^-)\|_+,$$

and $x \to \|x\|_+$ is 0 for $x < 0$ and x otherwise. C is actually an upper bound of $C^{0/1}$ since

$$\forall x \in \mathbb{R}, I\{x < 0\} < \|1 - x\|_+.$$

Moreover, as stated in [11], the minimization of C can be interpreted as margin maximization and our optimization problem is hence close to the ranking SVM [20] with, however, a lower computational cost. In ranking SVM, the observed training complexity is at least quadratic in the number of training constraints, c, while the complexity of our approach grows linearly with c, which allows for the application of *LinkLearn* to large constraint sets.

3.2 Model Parameterization

Our model relies on the Vector Space Model (VSM): each document d is first represented with a vector $(d_1, \ldots, d_V)$, V being the vocabulary size, and the documents are then compared according to the inner product of their vectors:

$$sim(d,d') = \sum_{i=1}^{V} d_i \cdot d'_i\ .$$

Like for *OKAPI* weighting, the weight d_i of a term i in a document d is a function of

- $tf_{d,i}$, i.e the number of occurrences of i in d,
- idf_i, i.e. the inverse document frequency of i in the corpus,
- l_d, i.e. the total number of term occurrences in d.

This weight is written as

$$d_i = g(tf_{d,i}, idf_i, l_d),$$

g being a parametric function whose parameters are selected to minimize the loss C introduced above. This function g is defined as the product of three functions f_{tf}, f_{idf} and f_l:

$$\begin{aligned} d_i &= g(tf_{d,i}, idf_i, l_d) \\ &= f_{tf}(tf_{d,i}) \cdot f_{idf}(idf_i) \cdot f_l(l_d). \end{aligned} \tag{5}$$

where f_{tf}, f_{idf} and f_l are multi-layer perceptrons, MLP [4] (see figure 1), composed of three layers:

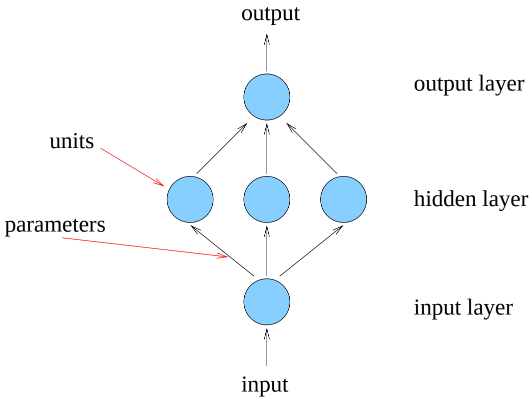

Figure 1: Multi-layer Perceptron Function

- the input layer is a single real value x, which is respectively tf, idf and l in our case,
- the hidden layer consists of k hyperbolic tangent units:

$$\forall h \in \{1, \dots, k\}, y_h^{(1)} = \tanh\left(\theta_{0,0,h} + \theta_{0,1,h} \cdot x\right)$$

- the output layer consists of one soft-plus unit:

$$y^{(2)} = \log\left(1 + \exp\left(\theta_{1,0,0} + \sum_{h=1}^{k} \theta_{1,1,h} \cdot y_h^{(1)}\right)\right)$$

  which ensures that the output is positive.

The functions f_{tf}, f_{idf}, f_l are therefore 3 different parametric functions whose respective parameters are:

$$\begin{aligned}
\theta^{(tf)} &= \left(\theta_{i,j,h}^{(tf)}, \forall (i,j,h) \in \{0,1\} \times \{0,1\} \times \{1, \dots, k_{tf}\}\right), \\
\theta^{(idf)} &= \left(\theta_{i,j,h}^{(idf)}, \forall (i,j,h) \in \{0,1\} \times \{0,1\} \times \{1, \dots, k_{idf}\}\right), \\
\theta^{(l)} &= \left(\theta_{i,j,h}^{(l)}, \forall (i,j,h) \in \{0,1\} \times \{0,1\} \times \{1, \dots, k_{l}\}\right),
\end{aligned}$$

where k_{tf}, k_{idf} and k_l denote the number of hidden units in each MLP.

This parameterization hence involves 3 different MLPs, each one having a real value input, which could be seen as a limitation with respect to a model where function g would be a unique MLP with a 3-dimensional input. Such a simplification is however necessary to apply this model to large corpora since the use of 3 MLPs significantly reduces the required computational cost: instead of evaluating an MLP function for all triplets $\forall d, i, (tf_{d,i}, idf_i, l_d)$, it is only necessary to evaluate it for each possible value of $tf_{d,i}$, idf_i and l_d. Without this simplification, the experiments presented in Section 4 could not have been performed since they would have required approximately $1,000$ times more MLP evaluations. Moreover, the experimental results show that this simplified parameterization does not prevent our model from reaching good performance.

3.3 Parameter Fitting

In order to select the MLP parameters which minimize the cost C (4), a stochastic gradient descent algorithm [16] is applied (see Algorithm 1): at each iteration, a document d is randomly selected (with replacement), we estimate the true gradient $\frac{\partial C(\theta)}{\partial \theta}$ with $\frac{\partial C_d(\theta)}{\partial \theta}$ and update the parameters in the (estimated) direction of greatest decrease in C, according to the learning rate λ:

$$\theta \leftarrow \theta - \lambda \frac{\partial C_d(\theta)}{\partial \theta}.$$

The termination criterion we used is known as *early stopping* [4]: a validation set D_{valid}, which is different from the training set D_{train} and the test set D_{test}, is used to estimate the generalization performance (i.e the expected performance over data that were not available during training) and the iterative training procedure is stopped when no further improvement or a degradation is observed on this estimate. This technique intends to avoid over-fitting, i.e. reaching good performance over training data but failing to generalize [16]. The validation data D_{valid} is also used in the same spirit to select the different hyper-parameters (k_{tf}, k_{idf}, k_l and λ). Note that since D_{valid} is used to estimate the hyper-parameters, it cannot also be used to estimate the expected performance of the system on unseen data, hence a separate set D_{test} is used for this purpose.

Algorithm 1 Training Procedure

Initialize θ
repeat
  Pick $d \in D_{train}$ randomly with replacement
  Compute the gradient $\frac{\partial C_d(\theta)}{\partial \theta}$
  Update weights $\theta \leftarrow \theta - \lambda \frac{\partial C_d(\theta)}{\partial \theta}$
until termination criterion

4 Experiments and Results

This section presents the experiments performed in order to assess the proposed method. The experimental setup is first described, model training and hyper-parameter selection are then presented and finally the results are discussed.

4.1 Experimental Setup

The experiments presented in the following are performed over the Wikipedia corpus [2]. This dataset consists of $\sim 450,000$ encyclopedia articles. In each article, the authors refer to other related articles using hyper-links. In order to evaluate the generalization properties of our model, the corpus is randomly split into three different parts: *train*, *valid* and *test*. This split results in three sets, each composed of $150,625$ documents. For all documents, the links pointing to articles in a different set are removed, such that each set can be used individually. Table 1 summarizes set statistics. For indexing, all three sets have been stemmed (using Porter's algorithm [17]) and stopped (i.e. terms occurring in more than $10,000$ documents have been removed). Moreover, terms occurring only once in the corpus have also been removed since this reduces greatly vocabulary size without any impact on document comparisons. The *train* and *valid* sets are used during model fitting: *train* is used for gradient descent (i.e. C is minimized over this set) while *valid* is used to assess the generalization performance during training and to select the hyper-parameters.

In contrast, the *test* set is considered to be unavailable during training and is only used for the final evaluation. This evaluation compares the results obtained with our inferred similarity measure to those obtained with the state-of-the-art *OKAPI* formula (see Section 2) over the same data. Two types of measures have been used for evaluation: the constraint error rate and IR measures. The constraint error rate $C^{0/1}$ has been introduced above (3): this quantity measures the percentage of constraints of type "a document d should be more similar to a document with which it shares a link with than to any other document" which are not respected. The IR evaluation is used in the following context:

Table 1: The 3 Subsets of the Wikipedia Corpus (vocabulary size and number of terms per document are measured after stopping and stemming).

	train	valid	test
Number of documents	150,625	150,625	150,625
Vocabulary size	229,003	227,018	230,198
Number of terms per doc.	83.3	83.5	83.4
Number of links per doc.	13.4	12.5	12.6

Table 2: TREC-9 queries for the TDT-2 Corpus (vocabulary size and number of terms per document are measured after stopping and stemming).

TDT-2/TREC-9 queries	
Number of documents	24,823
Number of terms per doc.	63.8
Vocabulary size	45,188
Number of queries	50
Number of terms per q.	6.3
Number of relevant doc. per q.	13.4

each document d is considered to be a query whose relevant documents consist of the documents which are linked with d. This task is related to the "similar pages" function that exists in web search engines (e.g. [1]): given a current document d, the user needs to find information which is related to the content of d. In this case, for each document d, all other documents are ranked by decreasing similarity with d and this ranking is then compared to the ideal case in which the documents of $L(d)$ appear above any other document. This comparison is performed according to mean average precision ($AvgP$), break-even point (BEP) and precision at top 10 documents ($P10$). This evaluation is hence performed over $150,625$ queries (each test document is used to retrieve its related articles) and reliable statistics can thus be extracted from such a large test set. To have a more complete evaluation, we also compared *LinkLearn* and *OKAPI* matching on TREC queries for the TDT-2 corpus [9], which are short retrieval queries (see Table 2). This second set of experiments aims at evaluating whether the measure inferred from one corpus (i.e. Wikipedia) can be reasonably applied to different data without model re-training or adaptation.

4.2 Model Training

As mentioned above, the hyper-parameters are selected in order to optimize the performance over validation data. Table 3 summarizes the selected values for both *LinkLearn* and *OKAPI*. The performance over *valid* obtained during training is reported on Figure 2. This figure shows that the performance of *LinkLearn* quickly reaches *OKAPI* level and continues to grow afterwards. Moreover, these plots also show that the optimization of C, which has been introduced to minimize the error rate $C^{0/1}$, not only leads to a lower $C^{0/1}$ but also leads to better results according to the standard retrieval measures, like $P10$ or $AvgP$.

We should further note that while there were about $160,000$ training documents, only about $54,000$ of them were randomly selected during training before early stopping (i.e. training stopped after $54,000$ iterations), which might seem strange. On the other hand, for each selected document d, all the other documents were also used, either in the $L(d)$ or the $\overline{L(d)}$ sets.

Next section describes the results on the *test* set, which are of greater interest than the results on

Table 3: Hyper-parameters for *LinkLearn* and *OKAPI*

LinkLearn	*OKAPI*
$k_{tf} = 5$	
$k_{idf} = 10$	$K = 1.5$
$k_l = 10$	$B = 0.6$
$\lambda = 0.001$	

the *valid* set in order to evaluate the ability of *LinkLearn* to generalize over new data.

4.3 Results

Table 4 presents the results obtained for the experiments performed over *test* data. According to all measures, the performance of *LinkLearn* is higher than the *OKAPI* matching performance. In all cases, the relative improvement is more than 15%. We further compared the results of *LinkLearn* and *OKAPI* for each of the $150,625$ queries in order to verify whether the advantage of *LinkLearn* could be due to a few queries. The Wilcoxon signed rank test rejected this hypothesis with 95% confidence for each retrieval measure.

The results obtained over TDT-2 data confirm those obtained over the Wikipedia corpus (see Table 5): the use of *LinkLearn* leads to an improvement with respect to *OKAPI* matching according to the different performance measures used (e.g. 18% improvement for *AvgP*, 34.5% vs 29.3%). This shows that the similarity measure derived from one training corpus (i.e. Wikipedia, a set of encyclopedia articles) can be used over a different type of data (i.e. TDT-2, a set of broadcast news transcripts and newswire articles). This means that a retrieval task over a given corpus, which may not even be hyper-linked, can benefit from larger hypertext datasets that may easily be found on the web nowadays.

As a final analysis of our approach, we have compared the inferred *LinkLearn* weighting with *OKAPI*. Figure 3 shows the weight d_i of a term i in a document d with respect to $tf_{d,i}$, df_i and l_d. These plots show that the inferred weighting (*LinkLearn*) and the one given *a priori* (*OKAPI*) are close to each other which may highlight the appropriateness of *OKAPI*'s parameterization. However, the plots also show some differences between the weighting schemes that could explain why *LinkLearn* yields higher performance. The main differences consists of:

Term Frequency tf For low tf values ($tf < 25$), the term weight grows much slower when tf increases for *LinkLearn* than for *OKAPI* which means that term repetitions are considered more important in *OKAPI* than in *LinkLearn*.

Document Frequency df The term weights of the two approaches are similar for low df but different for large df. In this case, *OKAPI* gives more weight to frequent terms, which means that general terms have more influence on *OKAPI* matching than on *LinkLearn* similarity.

Document Length l The two weighting schemes are similar for short documents, while *LinkLearn* gives less weight to longer documents, which may mean that long documents may contain a large amount of repetition about the same topic rather than being of richer content.

This analysis underlines that small changes in the weighting scheme can influence significantly retrieval performance. This also shows that the use of a learning technique may be an appropriate method to automatically infer the weighting scheme (or alternatively the document similarity measure) from hyper-link data.

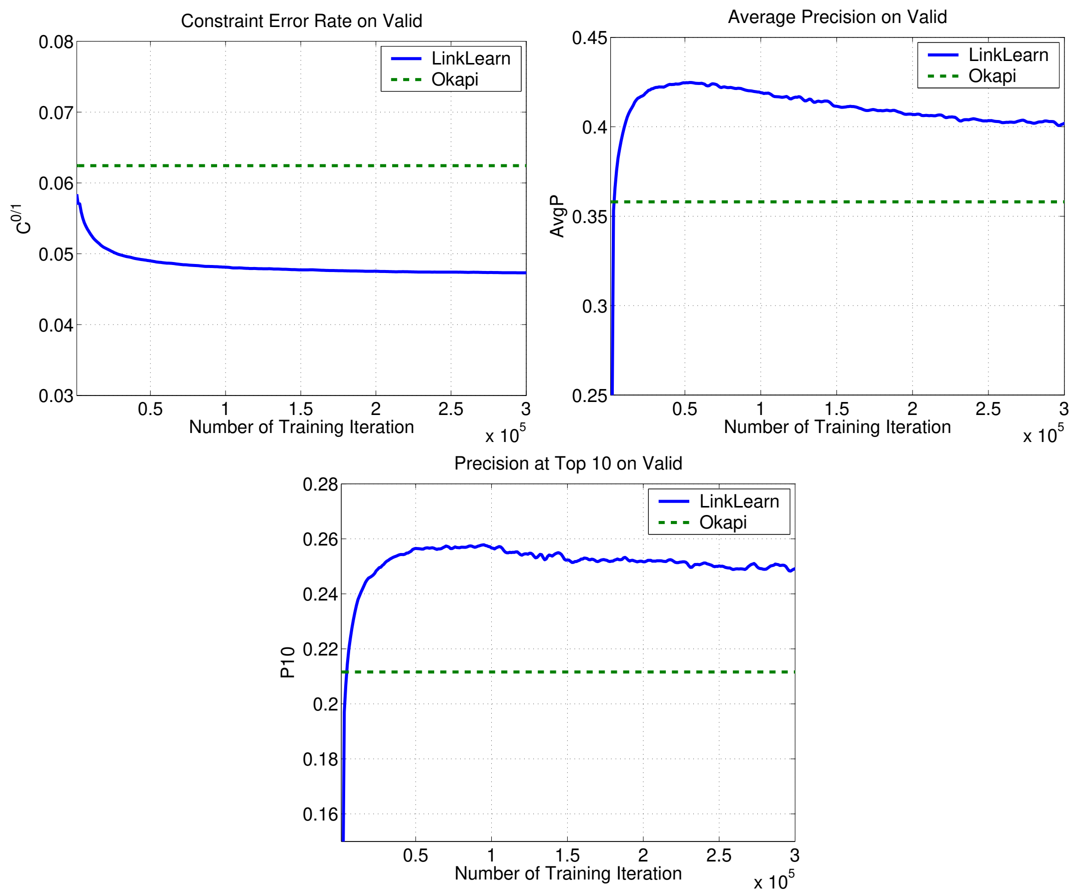

Figure 2: Validation Performance during Training

These plots show the performance, in terms of cost $C^{0/1}$, average precision $AvgP$, and precision at top 10 $P10$, up to $300,000$ iterations for readability, but early stopping criterion has actually stopped training before over-fitting on the $AvgP$ curve (i.e. after $54,000$ iterations).

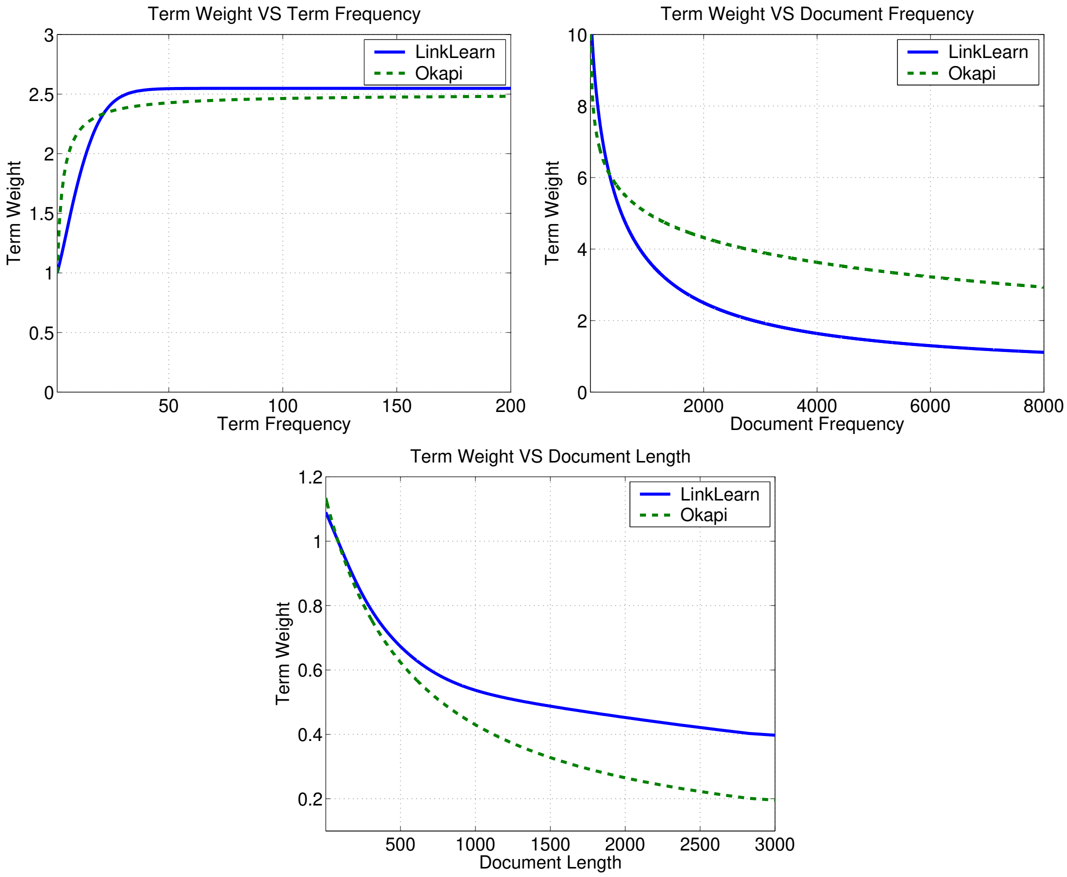

Figure 3: Term Weights for *OKAPI* and *LinkLearn*

5 Conclusions

In this paper, we introduced an approach to derive a document similarity measure from hyper-link information. As stated in previous work [6, 12], links between documents can be considered as an indicator of topical relatedness and a hyper-linked training corpus can hence be useful to identify a reliable measure of similarity between documents. Such an approach could benefit text mining applications, like I.R. or document clustering [3] in which the assessment of semantic similarity is a crucial point.

Our approach relies on a gradient descent strategy: a parameterized similarity measure $a, b \rightarrow sim_\theta(a, b)$ is first defined and θ is selected such that linked documents are considered closer to each other than to unlinked ones. For that purpose, a loss C which expresses how far sim_θ is from this ideal condition is introduced and the parameters which minimize C are then selected through gradient descent optimization.

This approach was compared to the state-of-the-art similarity measure used in *OKAPI* systems [19] on a retrieval task (which consists of finding the articles related to another article in an encyclopedia, see section 4). The proposed model is shown, in this context, to reach statistically significantly higher performance (e.g. precision at top 10 documents is 18% higher, 25.2 vs 21.5, when using sim_{θ^*} compared to *OKAPI* matching). This improvement was confirmed on another set of experiments performed with TREC9 queries for the TDT-2 corpus (e.g. average precision of 34.5% for *LinkLearn* vs 29.4% for *OKAPI*).

These results are promising and we plan to further investigate this approach on different retrieval tasks such as TREC-TIPSTER tasks and to combine it with other techniques which improve retrieval performance, such as query expansion. Moreover, we also intend to study different parameterizations for the similarity measure: e.g. the cost C could be optimized over a hyper-linked corpus to infer a matching measure relying on a linear projection parameterization, like in the LSA model [13].

Acknowledgments

The authors would like to thank Mikaela Keller for her useful suggestions and comments. This work is supported by the Swiss National Science Foundation through the National Center of Competence in Research on Interactive Multimodal Information Management (IM2). It was also supported by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, funded by the Swiss OFES.

References

[1] Google. www.google.com.

[2] Wikipedia, the free encyclopedia. www.wikipedia.org.

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

[4] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, London, England, 1995.

[5] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *International World Wide Web Conference*, pages 107–117, 1998.

[7] W. Buntine. Variational extensions to EM and multinomial PCA. In *European Conference on Machine Learning*, pages 23 – 34, 2002.

[8] Z. Chen, S. Liu, L. Wenyin, G. Pu, and W.Y Ma. Building a web thesaurus from web link structure. In *ACM Special Interest Group on Information Retrieval*, pages 48–55, 2003.

[9] C. Cieri, D. Graff, M. Liberman, N. Martey, and S. Strassel. The TDT-2 text and speech corpus. In *DARPA Broadcast News Workshop*, 1999.

[10] D. Cohn and T. Hofmann. The missing link a probabilistic model of document content and hypertext connectivity. In *Conference on Advances in Neural Information Processing Systems*, pages 430–436, 2000.

[11] R. Collobert and S. Bengio. Links between perceptrons, MLPs and SVMs. In *International Conference on Machine Learning*, 2004.

[12] B. D. Davison. Topical locality in the web. In *ACM Special Interest Group on Information Retrieval*, pages 272 – 279, 2000.

[13] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 6(41):391–407, 1990.

[14] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42:177–196, 2001.

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

[16] Y. Lecun, L. Bottou, G. B. Orr, and K. R. Mueller. Efficient backprop. In G. B Orr and K. R. Mueller, editors, *Neural Networks: Trick of the Trade*, chapter 1, pages 9–50. Springer, 1998.

[17] M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[18] M. Richardson and P. Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *Conference on Advances in Neural Information Processing Systems*, 2002.

[19] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *NIST Text Retrieval Conference*, pages 109–126, November 1994.

[20] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Conference on Advances in Neural Information Processing Systems*, 2003.

[21] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Conference on Advances in Neural Information Processing Systems*, pages 505–512, 2003.

Table 4: Results for the Similar Document Search Task (Wikipedia *test* set)

	OKAPI	*LinkLearn*
Error rate	5.4%	4.2% (−22%)
Precision at top 10	21.5%	25.2% (+18%)
Break Even Point	36.6%	42.1% (+15%)
Average Precision	37.3%	43.8% (+17%)

Table 5: Results for the Retrieval task (TREC9 queries for TDT-2 corpus)

	OKAPI	*LinkLearn*
Precision at top 10	38.8%	43.2% (+11%)
Break Even Point	30.3%	35.2% (+16%)
Average Precision	29.3%	34.5% (+18%)