

Too Many Mammals: Improving the Diversity of Automatically Recognized Terms

Ziqi Zhang, Lei Xia, Mark A. Greenwood and José Iria
The Department of Computer Science
The University of Sheffield, United Kingdom
{*initial.surname*}@dcs.shef.ac.uk

Abstract

Automatic Term Recognition systems extract domain-specific terms from text corpora. Unfortunately current systems fail to capture the whole of the domain covered by a corpus. To address this problem, we present a novel term re-ranking method that generates term lists containing terms that are not only individually salient, but also contribute to a globally diverse list that is truly representative of the corpus. We show that, even without any prior knowledge about the domain, our proposed method improves the diversity of the results produced by two popular automatic term recognition algorithms.

Keywords

Automatic Term Recognition, Diversity in Ranking, Random Walk, Semantic Similarity

1 Introduction

Automatic Term Recognition (ATR) is an important research area that deals with the recognition and extraction of technical terms from domain-specific corpora. ATR is often a processing step preceding more complex tasks, such as semantic search or ontology engineering [14, 3]. It can also be used as an end-user tool to, for instance, generate a list of terms that summarizes a text corpus provided by the user.

Whilst state-of-the-art ATR algorithms are reasonably successful in identifying the most relevant domain-specific terms from a corpus, the analysis we carried out of the output of these algorithms over several corpora led us to conclude that the ranking of terms rarely reflects the whole domain.

Having carefully studied the experimental outputs from [25], we have observed that terms from a subset of the sub-domains (of the domain covered by the corpus) tend to dominate the results, pushing other characteristic terms, which are perhaps not so globally relevant but nevertheless fundamental to get a comprehensive coverage of the domain, far down the ranking. For example, on the Wikipedia animal corpus described in Section 5, which contains 1051 random Wikipedia articles describing animals across roughly 30 scientific classes, in the top ranked 50 terms by the C-Value [10] algorithm there are the names of 13 mammals, 3 fish, 2 birds and 2 insects. Among these, 3 of the 13

mammals are species of whale, and 2 of the 3 fish are species of shark. Likewise, for the TF-IDF algorithm applied on the same corpus, in the top ranked 50 terms we obtain the names of 18 mammals, 6 birds and 4 insects. Hence terms belonging to these 3 or 4 classes dominate the results, preventing the remaining classes from being properly represented. Therefore, with current methods, taking the top-ranked terms is unlikely to produce a diverse list of terms that is fully representative of the entire domain. Unfortunately, for certain applications of ATR this is not a desirable behaviour, e.g., generating a list of terms that best summarizes a text corpus.

In this paper, we address the so-called “diversity in ranking” problem [26] in the context of ATR. Our goal is to generate term lists which contain terms that are not only individually salient, as produced by current methods, but at the same time contribute to a globally diverse list. By promoting diversity, we expect to balance the number of terms from different sub-domains appearing in the top results and provide the user with a better notion of the whole of the domain covered by the corpus. Our main contributions consist of: 1) designing and implementing a novel term re-ranking method, called TermHopper, that can be coupled with any existing ATR algorithm, 2) creating a new corpus for ATR based on Wikipedia¹, and 3) empirically showing that the proposed method provides an improved ranking of extracted terms. Furthermore, an attractive feature of the proposed approach is its domain independence, that is, it does not require any additional domain specific resources.

The remainder of this paper is structured as follows. In the following section we describe related work. Section 3 introduces our proposed ranking algorithm. Section 4 describes the application of the ranking algorithm to the ATR problem. In Section 5 we describe our experimental setup, namely the data collection and pre-processing steps, the design of the gold standard and the evaluation methodology. Sections 6 presents and discuss the results of our evaluation. We conclude with an outline of our plans for future work.

2 Related Work

[25] presented a comparison of several state-of-the-art ATR methodologies namely TF-IDF, Weirdness [1], C-Value [10], Glossex [14], and Termex [17]. TF-IDF

¹ <http://www.dcs.shef.ac.uk/~ziqizhang/resources/wiki.zip>

makes use of term frequencies and document frequencies in the target corpus; C-value makes use of term frequencies and the frequencies at which terms appear within longer terms. Terms which exhibit high frequency and are less often used within longer terms are given higher ranks; Weirdness compares term frequencies in both the target and a reference corpus; Glossex and Termex are similar to Weirdness in the way that they all utilise term frequencies in the target corpus versus those in a reference corpus. Glossex normalises the overall term frequencies with respect to the frequencies of the component words whilst Termex also captures domain concepts that exhibit high frequencies within a small subset of the corpus but are completely absent in the remainder of the corpus.

To the best of our knowledge, the diversity issue has been overlooked in traditional ATR methods – the closest related problem is term clustering. [2] apply the LEXTER algorithm to extract candidate terms, then applied the FASTR algorithm to cluster them. This essentially clusters terms by their canonical forms after morphological normalization and syntactic normalization. [13] run C/NC-value on 2,082 MEDLINE abstracts to extract candidate terms, then applied Nearest-Neighbour clustering to the top ranked terms. They defined contextual, functional and lexical similarity to collectively measure similarity between two terms. [20] perform similar experiments, in which they ran C/NC-value on the same corpus and then classified the extracted terms into UMLS classes. In order to do this, they extended Nenadic’s method of measuring term similarity by adding another dimension called term-class similarity, which is computed by co-occurrence strength of a term to a domain-specific verb that is usually a strong indicator of a class. [6] takes document clustering and word clustering as a co-clustering task, in which the output of one task (e.g., clusters of documents) induces another (e.g., clusters of words). They viewed documents in a corpus and their words as a graph connected by edges, and treated clustering as a graph partitioning problem in which optimum clusters are produced when the crossing edges between partitions have minimum weight.

Term clustering constitutes only part of the solution to the diversity problem as we need to understand how to produce a diverse ranked list of terms given the generated clusters. Methods to improve diversity in ranking include maximum marginal relevance (MMR) [5] in the context of text summarization, mixture models [24] in the context of adaptive information filtering systems, and subtopic diversity [22] and diversity penalty [23] in the context of document retrieval. The basic underlying idea of these methods is to penalize redundancy by lowering the rank of an item if it is similar to items already ranked.

Our proposed method uses a re-ranking approach based on absorbing random walks to improve the ranking of terms that describe the domain. Contrary to methods like MMR, which partly rely on heuristics, methods based on absorbing random walks have a principled mathematical model and strong empirical performance on artificial data.

The idea of using random walks in an absorbing Markov chain to improve diversity in ranking was first introduced in [26] where it was shown to effectively

improve ranking results on a text summarization task, and on a social network analysis task that identifies movie stars. Moreover, absorbing random walks have also found several other applications in the research literature. For example, [19] employ absorbing random walks in order to personalize the recommendation of items to users in a collaborative filtering task, while [18] apply them to modelling an expert finding problem. The method presented here is inspired by that of [26]. We use the same core absorbing random walks algorithm, but define a similarity metric and evaluation methodology appropriate for automatic term recognition tasks.

3 Ranking for Diversity

The ranking algorithm required to solve our problem must support the notions of centrality, diversity and prior:

1. **centrality** - a highly ranked term should be representative of a local group of terms;
2. **diversity** - the top terms should cover as many distinct groups as possible;
3. **prior** - it should be possible to incorporate an existing ranking, in our case the output from an existing ATR system, as prior knowledge.

Most ATR methods treat centrality and diversity separately and try to combine results *a posteriori*, sometimes using heuristic procedures. We, however, have chosen to adopt the GRASSHOPPER algorithm introduced in [26], which is based upon a principled mathematical model that combines centrality and diversity.

Graph-based ranking algorithms like GRASSHOPPER decide the centrality of a vertex from global information recursively drawn from the entire graph. The principle behind these models is that of voting or recommendation. When one vertex links to another one, it can be seen as casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher its importance. Plus, the importance of the vertex casting the vote determines how important the vote itself is. Hence, the score associated with a vertex is determined based on the votes that are cast for it, and the score of the vertices casting these votes.

In GRASSHOPPER diversity is addressed together with centrality by setting top-ranked vertices as absorbing states of a random walk over the vertices of the graph. Once the random walk reaches an absorbing state, it is absorbed and stays there. If we think about the expected number of visits to a node before absorption as its rank, we expect nodes “closer” (more similar) to the absorbing node to be less visited because the likelihood of “falling into” the nearby absorbing node is higher. This effectively places unranked vertices that are similar to absorbing nodes lower in the rank, thus encouraging diversity. In what follows we briefly describe the algorithm.

Given a graph W , represented by a $n \times n$ similarity matrix, where w_{ij} is the (non-negative) weight on the edge relating term i to term j ; a probability distribution r encoding the prior ranking, obtained from

a previously run ATR algorithm; and a tradeoff parameter $\alpha \in [0, 1]$ (that balances domain knowledge vs. prior), the algorithm produces a (re-)ranked list of the input terms such that the top terms are not only central but also diverse.

We start by finding the top ranked term using teleporting random walks. Let \tilde{P} be obtained by normalising the rows of W : $\tilde{P}_{ij} = w_{ij} / \sum_{k=1}^n w_{ik}$, so that \tilde{P}_{ij} is the probability that the walker moves to j from i . The walk is made teleporting by interpolating each row with the available prior information²:

$$P = \alpha \tilde{P} + (1 - \alpha) \mathbf{1} \otimes \mathbf{r},$$

where $\mathbf{1}$ is an all-1 vector, and $\mathbf{1} \otimes \mathbf{r}$ is the outer product. Due to the way it was designed (normalisation, teleportation), P is irreducible, aperiodic and ergodic, and therefore has a unique stationary distribution

$$\pi = P^\top \pi,$$

which gives the global visiting probabilities for each vertex. The states with large probabilities can be regarded as central vertices, an idea used in PageRank [4] and in many works in natural language processing such as text summarization [8] or keyword extraction [12]. The top ranked vertex is thus selected as being $a_1 = \arg \max_{i=1}^n \pi_i$.

The rest of the algorithm consists of an iterative procedure that takes the top ranked vertex from the previous step and sets it as being the absorbing state of the random walk at the current step. A vertex a can be turned into an absorbing state by setting $P_{aa} = 1$ and $P_{ai} = 0, \forall i \neq a$. Once the random walk reaches an absorbing state, it remains there, so we are no longer interested in the stationary distribution but rather in computing the expected number of visits to each node before absorption. The fundamental matrix

$$N = (\mathbf{I} - Q)^{-1}$$

gives the expected number of visits in the absorbing random walk [7], where Q is the submatrix of P obtained by re-arranging the terms so that those already ranked appear before unranked terms in the matrix:

$$P = \begin{bmatrix} \mathbf{I}_A & \mathbf{0} \\ R & Q \end{bmatrix}$$

The expected number of visits to vertex j is then given by the average over all possible starting states. In matrix notation:

$$\mathbf{v} = \frac{N^\top \mathbf{1}}{n - |A|}$$

where $|A|$ is the number of absorbed vertices. The vertex with the largest number of visits becomes the next term in the rank and an absorbing state for the remaining iterations: $a_{|A|+1} = \arg \max_{i=|A|+1}^n v_i$. The process is repeated until every vertex has been turned into an absorbing state.

4 TermHopper

Given the generic ranking algorithm introduced in the previous section, to define our TermHopper method we

² We add a small teleporting constant ϵ to the prior to ensure $P_{ij} > 0, \forall i, j$

now need to choose a similarity matrix W . Our approach to re-ranking the output of ATR takes terms as nodes in the graph, and uses a pair wise semantic similarity function between terms to assign weights to the edges in the graph. The reasoning behind using a semantic similarity function is the belief that similar nodes in the graph, i.e., terms, will cluster together and hence the algorithm presented in the previous section will choose nodes from many different clusters rather than many nodes from the same cluster.

There have been a number of different semantic similarity functions developed in the past years. For example, distributional similarity [11] would seem like a good match for the task of re-ranking ATR output. Unfortunately, distributional similarity requires a large amount of time and data to compute and was thus deemed inappropriate for this particular application. Instead we used a WordNet [9] based similarity function for assigning edge weights.

In WordNet synonymous words are grouped into synsets. Synsets are then linked by relations such as hyponymy and hypernymy. Different WordNet based similarity functions use different parts of this structure to determine the similarity between two words. In this study we used the Lin similarity measure³ [11].

The Lin similarity measures use corpus frequency counts to represent the informativeness of each node in WordNet, a technique developed by Resnik [16]. Nodes near the root of the hierarchy are not considered to be informative and have low values while those nearer the leaves have higher values, for example the concept *fish* would be more informative than *animal*. Numerical values representing the informativeness of each node are calculated from frequency counts of the words in that synset.

The information content (IC) for a synset, s , is calculated as $IC(s) = -\log(Pr(s))$ where $Pr(s)$ is the probability of synset s occurring in the corpus (estimated using word frequency counts). Resnik's similarity measure is provided by $sim_{Res} = IC(lcs(s_1, s_2))$, i.e. the similarity of a pair of nodes is defined to be the informativeness of their lowest common subsumer. Lin combined the same terms in a different formula:

$$sim_{Lin} = \frac{2 \times IC(lcs(s_1, s_2))}{IC(s_1) + IC(s_2)}$$

In our experiments we used information content values calculated over the BNC⁴.

TermHopper can work on the output of an existing ATR system simply by taking the scores output for each term as the prior vector r introduced in the previous section.

Finally, we restrict the similarity matrix W to contain, for each term, only its k neighbour (most similar) terms. Intuitively, this has the effect of reducing the potential noise introduced by the computation of all possible pair wise similarities.

³ We used a Java implementation of the Lin measure available at <http://nlp.shef.ac.uk/result/software.html>.

⁴ We used the information content file distributed with the Perl WordNet::Similarity library [15]

5 Experimental Setup

In this section we describe in detail our experimental setup which is designed to validate our approach. The experiments evaluate TermHopper against the original ATR algorithms and also a random baseline. First we describe the corpus and the gold standard used, and then we present the ATR algorithms selected for comparison and the proposed random baseline. Experimental results are presented in Section 6.

5.1 Dataset Collection and Processing

The experiments presented here were conducted on the AnimalWiki corpus, a manually built corpus of Wikipedia articles about 1,051 randomly selected animals. In total, the corpus contains 1.3 million words.

The corpus was created by extracting only the main textual content from the HTML pages and ignoring any formatting or navigational elements. The corpus was then POS tagged and the linguistic filters described by [10] were applied to extract nouns and noun phrases as candidate terms. The candidate list was then filtered by removing stop words.

5.2 Algorithms

Due to space limitations, we select two popular algorithms out of the collection of ATR algorithms available in the Java Automatic Term Recognition Toolkit (JATR⁵) [25]; namely the C-Value and TF-IDF algorithms. Please refer to the related work section for an overview of these algorithms. From the output of each algorithm we select the top 500 ranked terms for re-ranking by TermHopper, and present the comparison between our results and the results produced by the original algorithms, over the AnimalWiki corpus.

The random baseline, which we will call RandomHopper, can be modelled using a multivariate hypergeometric distribution or, equivalently, modelling the problem as a urn sampling problem without replacement. Under this model, to determine the next term in the rank we draw one term from the urn and observe its category. We plot a curve that shows the expected number of categories observed as the number of terms drawn from the urn grows. Because there is no closed form solution to this problem, we simply simulated the urn drawing process for an appropriately large number of runs and took the average of the observations.

We also check how TermHopper behaves when using a perfect similarity function (PerfectHopper):

$$sim_{Pft}(x, y) = \begin{cases} 1, & \text{if } cat(x) = cat(y) \\ 0, & \text{otherwise} \end{cases}$$

where *cat* is a function that returns the category of a term; that is, the perfect similarity is given by consulting the categories in the gold standard.

5.3 Gold Standard Evaluation Method

We designed the gold standard for evaluation by categorising terms into different semantic categories. In

it is actually their common	Scientific classification
	Kingdom: Animalia
	Phylum: Chordata
	Class: Mammalia
	Order: Primates
	Suborder: Haplorrhini

Fig. 1: A excerpt of a Wikipedia page showing the scientific classification of an animal.

Actinopterygii	Echinoidea
Amphibia	Gastropoda
Anthozoa	Insecta
Arachnida	Malacostraca
Aves	Mammalia
Bivalvia	Merostomata
Cephalaspidomorphi	Osteichthyes
Cephalopoda	Reptilia
Chondrichthyes	Sauropsida
Clitellata	Scyphozoa
Crustacea	Trilobita

Table 1: Category labels derived from Wikipedia Scientific Classification “Class”.

order to do this, we attempted to automatically obtain the category of a term by applying a few simple heuristics over the English section of Wikipedia, using the Java Wikipedia Library [21] and the February 2007 English Wikipedia dump. If the term denotes an animal, we retrieve its corresponding Wikipedia page, and extract the scientific classification for that animal as the category for the term (Figure 1).

Scientific classifications for animals in Wikipedia are subdivided into Kingdom, Phylum, Class, Order, Family, Genus, Species, etc. For the purposes of our study we have categorised terms according to Class, and we only apply the automatic labelling processes to the selected top section (500) of terms from each ATR algorithm considered. This produced 22 Wikipedia categories as listed in Table 1.

The automated process left many terms uncategorised, in particular those terms which do not denote an animal. These were manually labelled according to a further six categories, as illustrated in Table 2. *Adjective* is used to categorises terms that are used as adjectives; *Group* contains terms used for describing groups of animals; *Part* are terms used for describing body parts; *Place* and *Time* refer to terms which are generally places or time expressions; while for any other term missing a category we assign the label *Other*.

We are interested in measuring diversity in the ranking generated by the several algorithms. For that, we study how the number of observed categories grows with the number of ranked terms considered.

⁵ <http://www.dcs.shef.ac.uk/~ziqizhang>

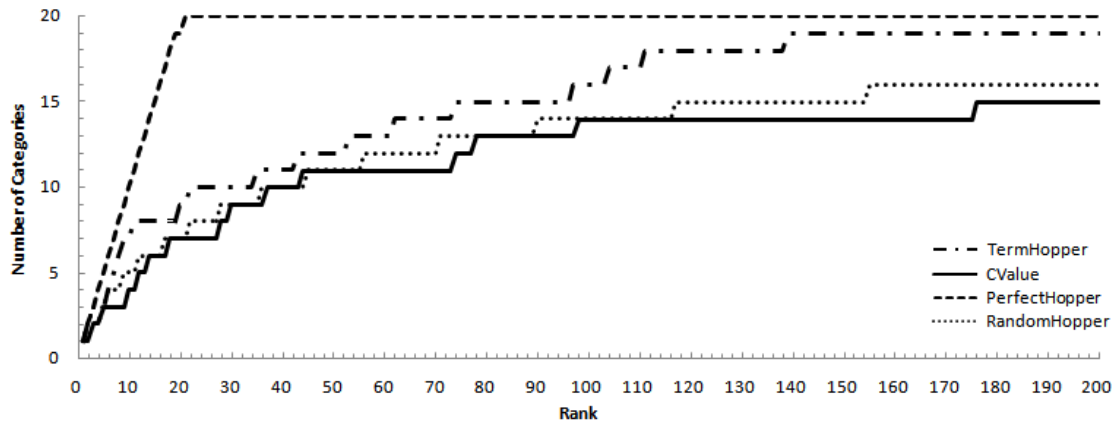


Fig. 2: Experimental results comparing the C-Value and TermHopper algorithms, $\alpha=0.8$ and $k=3$.

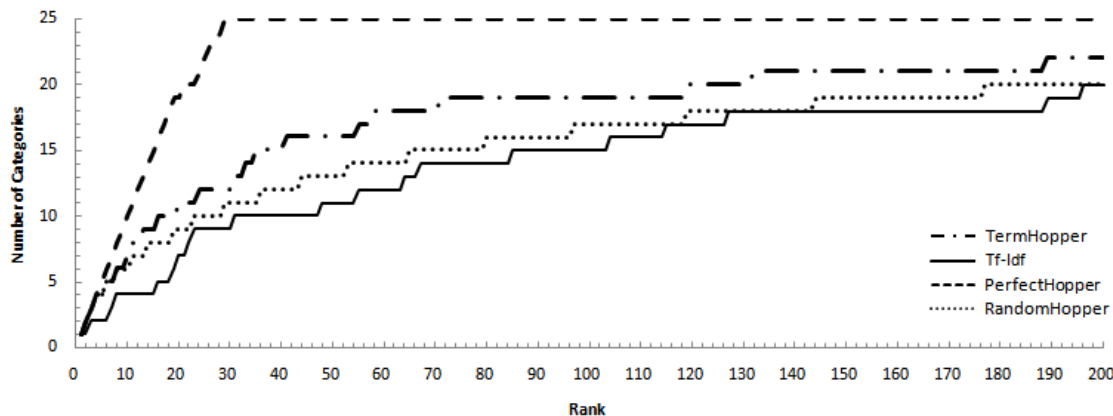


Fig. 3: Experimental results comparing the TF-IDF and TermHopper algorithms, $\alpha=0.6$ and $k=10$.

Category Label	Examples
Place	river, sea, America
Time	year, month, Ice Age
Adjective	black, hybrid, male
Group	pack, colony, species
Other	range, sense, devil
Part	head, nose, mouth

Table 2: Non-animal category labels.

6 Results and Discussion

The performance of TermHopper in comparison to the base ATR algorithms of C-Value and TF-IDF can be seen in Figures 2 and 3 respectively. These graphs show the number of observed categories against the number of ranked terms being considered. Both graphs also show the performance of RandomHopper and PerfectHopper for comparison. In both experiments the following parameters of TermHopper were tuned using a grid method:

- the tradeoff parameter $\alpha \in [0, 1]$
- the number of (most similar) neighbours, $k \in \{3, 10, 499\}$, in the similarity matrix W

Overall, our approach consistently outperforms both the random baseline and the rankings generated by the

original ATR systems. It is not surprising that the latter perform worse than the baseline, since they favour centrality only and have no notion of diversity, and thus place many (globally relevant) terms from the same category in the top ranked positions. TermHopper, on the other hand, shows more term categories sooner, both within the all-important first 10 or 20 results as well as beyond that, while at the same time ensuring, by design, that the top terms are the most central within their respective categories.

By tuning the parameters, we have observed that results improve when considering only a few (at most 10) neighbours in the similarity matrix instead of using a dense matrix with all the possible pair wise similarity values computed. This also matches our intuition that only the network of the few most similar terms should be used to semantically define a given term.

The gap between TermHopper’s and PerfectHopper’s curves indicates how strong the misalignment is between the adopted term similarity function and the desired gold standard classification. The proposed generic WordNet-based similarity function can be replaced with a more specific similarity function based on domain knowledge to bridge that gap. Unfortunately, doing so reduces the portability of the method. However, we believe that the results obtained are still very valuable, because they show that a considerable improvement over the baseline can be obtained

even with a domain-independent, off-the-shelf similarity function.

7 Conclusions and Future Work

ATR algorithms often fail to capture the whole of the domain covered by the corpus, which for some applications may be unacceptable or undesirable. For example, a corpus summarization system aiming to provide a short summary to the user in the form of keywords needs to be able to cover all of the sub-domains in as few keywords as possible – ideally using exactly one keyword per sub-domain.

To improve diversity in ranking the automatically recognized terms, we have presented a novel term re-ranking method, called TermHopper. We showed that, even without encoding any knowledge about the domain, i.e., using a generic WordNet-based term similarity function, the proposed method is successful in improving the diversity of the results produced by two popular ATR algorithms on the AnimalWiki corpus. One of the advantages of the proposed method is that it can be coupled to any existing ATR system, since it runs as a post-processing step.

As future work, we plan to experiment with several similarity metrics from the literature to replace the WordNet-based similarity used here, as long as their computation cost remains low, due to the exponential cost of computing the pair wise similarity. We also plan to study the impact of deploying the new diversity-improved ATR system in our existing ontology learning tools.

Acknowledgements

This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

References

- [1] K. Ahmad, L. Gillam, and L. Tostevin. University of Surrey Participation in TREC8: Weirdness Indexing for Logical Document Extrapolation and Retrieval (WILDER). In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, 1999.
- [2] D. Bourigault. Surface Grammatical Analysis for the Extraction of Terminological Noun Phrases. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING)*, 1992.
- [3] C. Brewster, J. Iria, Z. Zhang, F. Ciravegna, L. Guthrie, and Y. Wilks. Dynamic Iterative Ontology Learning. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 07)*, 2007.
- [4] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Computer Networks and ISDN Systems*, volume 30, pages 107–117, 1998.
- [5] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR'98*, Australia, 1998.
- [6] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274, San Francisco, 2001.
- [7] P. Doyle and J. Snell, editors. *Random Walks and Electric Networks*. Mathematical Assoc. of America, 1984.
- [8] G. Erkan and D. Radev. Lexrank: Graph-based centrality as salience in text summarization. In *Journal of Artificial Intelligence Research (JAIR)*, volume 22, pages 457–479, 2004.
- [9] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database and some of its Applications*. MIT Press, 1998.
- [10] K. T. Frantzi and S. Ananiadou. The C-value/NC-value Domain Independent Method for Multi-Word Term Extraction. *Journal of Natural Language Processing*, 1999.
- [11] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98)*, Madison, Wisconsin, 1998.
- [12] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, Barcelona, Spain, 2004.
- [13] G. Nenadic, I. Spasic, and S. Ananiadou. Term Clustering Using a Corpus-Based Similarity Measure. In *Proceedings of the 5th International Conference on Text, Speech and Dialogue*, pages 151–154, 2002.
- [14] Y. Park, R. J. Byrd, and B. K. Boguraev. Towards Ontologies on Demand. In *Proceedings of the Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data*, 2003.
- [15] T. Pedersen, S. Patwardhan, and J. Michelizzi. WordNet::Similarity - Measuring the Relatedness of Concepts. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, San Jose, CA, 2004.
- [16] P. Resnik. Using Information Content to evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 448–453, Montreal, Canada, 1995.
- [17] F. Sclano and P. Velardi. TermExtractor: A Web Application to Learn the Shared Terminology of Emergent Web Communities. In *Proceedings of the 3rd International Conference on Interoperability for Enterprise Software and Applications (I-ESA)*, 2007.
- [18] P. Serdyukov, H. Rode, and D. Hiemstra. Modeling Expert Finding as an Absorbing Random Walk. In *Proceedings of the 31st Annual International ACM SIGIR conference on Research and Development in Information Retrieval*, 2008.
- [19] A. P. Singh, A. Gunawardana, C. Meek, and A. C. Surendran. Recommendations using Absorbing Random Walks. In *Proceedings of NESCAI 2007*, 2007.
- [20] I. Spasic, G. Nenadic, K. Manios, and S. Ananiadou. Supervised Learning of Term Similarities. In *Proceedings of the Third International Conference on Intelligent Data Engineering and Automated Learning*, 2002.
- [21] T. Zesch, C. Muller, and I. Gurevych. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 2008.
- [22] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th Annual International ACM SIGIR Conference*, pages 10–17, Toronto, Canada, 2003.
- [23] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma. Improving web search results using affinity graph. In *Proceedings of the 28th Annual International ACM SIGIR Conference*, Salvador, Brazil, 2005. ACM.
- [24] Y. Zhang, J. Callan, and T. Minka. Novelty and redundancy detection in adaptive filtering. In *Proc. ACM SIGIR 2002*, pages 81–88. ACM Press, 2002.
- [25] Z. Zhang, J. Iria, C. Brewster, and F. Ciravegna. A Comparative Evaluation of Term Recognition Algorithms. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 2008.
- [26] X. Zhu, A. B. Goldberg, J. V. Gael, and D. Andrezejewski. Improving Diversity in Ranking using Absorbing Random Walks. In *Proceedings of NAACL/HLT*, 2007.