Automatic Subject Metadata Generation for Scientific Documents Using Wikipedia and Genetic Algorithms

Arash Joorabchi and Abdulhussain E. Mahdi

Department of Electronic and Computer Engineering, University of Limerick, Ireland
{Arash.Joorabchi, Hussain.Mahdi}@ul.ie

Abstract. Topical annotation of documents with keyphrases is a proven method for revealing the subject of scientific and research documents. However, scientific documents that are manually annotated with keyphrases are in the minority. This paper describes a machine learning-based automatic keyphrase annotation method for scientific documents, which utilizes Wikipedia as a thesaurus for candidate selection from documents' content and deploys genetic algorithms to learn a model for ranking and filtering the most probable keyphrases. Reported experimental results show that the performance of our method, evaluated in terms of inter-consistency with human annotators, is on a par with that achieved by humans and outperforms rival supervised methods.

Keywords: text mining, scientific digital libraries, subject metadata, keyphrase annotation, keyphrase indexing, Wikipedia, genetic algorithms.

1 Introduction

Automatic keyphrase annotation methods for scientific documents can be divided into two main categories:

1. Keyphrase extraction: keyphrases are picked from a set of candidate phrases extracted from the content of the document itself and are ranked and filtered based on their various statistical and/or semantical features, such as frequency, position, length, and coherence. The ranking function could be either (a) unsupervised, where it is heuristically defined by manual analysis of sample documents and encoding general properties of typical keyphrases, e.g., see [1, 2]; or (b) supervised, where it is automatically derived by a general-purpose ML algorithm from a training dataset, e.g., see [3-6]. Keyphrase extraction approach has two main weaknesses: 1) it is prone to generating phrases composed of a set or sequence of words that occur contiguously within the document and have statistically significant properties, such as high frequency (a.k.a statistically motivated phrases), but are ill-formed, grammatically wrong, or meaningless; 2) it limits the scope of potential candidates to the phrases explicitly appearing in the document.

2. Keyphrase assignment: keyphrases are picked from controlled vocabularies, such as taxonomies, thesauri, and subject heading systems (e.g., LCSH, MeSH, AGROVOC, Eurovoc) and are not confined to the phrases appearing in the document. In this approach, keyphrase annotation is treated as a multi-label text classification

problem and general-purpose ML algorithms (e.g., SVM, NB) are utilized to learn a model for each term in the controlled vocabulary from a set of manually annotated training documents. The learnt models are then applied to test documents for classification resulting in a set of high-probability classes (i.e., keyphrases) per document, e.g., see [7, 8]. Using this approach, assigned keyphrases are well formed, grammatically correct, and not limited to those appearing in the document. Therefore, it can cope with cases where a concept is discussed but not explicitly mentioned in the document. However, depending on the characteristics of the target domain, this approach may suffer one or more drawbacks common among supervised ML-based approaches to information retrieval in general, including lack of high quality and/or quantity training data, data sparsity and/or skewed distribution, and concept drift.

Medelyan and Witten [9, 10] proposed a hybrid approach as an intermediate between keyphrase extraction and keyphrase assignment which they have called keyphrase indexing. In this approach, candidate phrases are limited to a set of descriptors, i.e., preferred and commonly used terms for the represented concepts in a domain-specific thesaurus, which either themselves or their synonyms/alternative lexical forms (a.k.a non-descriptors, encoded in form of semantic relations in the thesaurus) occur in the document. This method of candidate generation eliminates the two above-mentioned weaknesses of keyphrase extraction approach as the generated candidate phrases are well-formed, semantically rich, and not restricted to those occurring in the document explicitly. Similar to keyphrase extraction, in this approach an unsupervised or supervised ranking function is deployed to model the general properties of keyphrases in order to rank and filter the most probable ones. This method of rank and filtering requires either no or limited training data, depending on the type of ranking function deployed. This is in contrast to keyphrase assignment approach which requires a set of annotated documents per descriptor. The main weakness of the keyphrase indexing approach is that it assumes there exists a comprehensive domain-specific thesaurus for the target domain, which is not always a feasible assumption. This weak point has been addressed by automatic construction of a universal thesaurus from Wikipedia [11, 12] and replacing the domain-specific thesauri with the thesaurus derived from Wikipedia [13, 14].

In this work, we aim to extend the keyphrase indexing approach, described above, by: (a) introducing a new set of features for the candidate phrases derived from Wikipedia, which enhances the performance of rank and filtering process, and (b) introducing a new supervised ranking function based on Genetic Algorithms (GA) which eliminates the need for manual feature selection and outperforms general-purpose ML algorithms used for keyphrase annotation.

2 Candidate Generation

Following the work of Medelyan and Witten [13, 14], we utilize an open-source toolkit called Wikipedia-Miner [15] for candidate generation. We use the topic detection functionality of the Wikipedia-Miner to extract all the Wikipedia topics (i.e., Wikipedia articles) whose descriptor or non-descriptor lexical representations occur in the document, and use the descriptors of the extracted topics as candidate phrases for the document. We have devised a set of twenty statistical, positional, and semantical

features for candidate topics/phrases to capture and reflect various properties of those candidates which have the highest keyphraseness probability:

1. Term Frequency (TF): the occurrence frequency of the candidate phrase (i.e., descriptor of the extracted Wikipedia topic) and its synonyms and alternative lexical forms/near-synonyms (i.e., non-descriptors of the extracted Wikipedia topic) in the document. The TF values are normalized by dividing them by the highest TF value in the document.
2. First Occurrence: the distance between the start of the document and the first occurrence of the candidate topic, measured in terms of the number of characters and normalized by the length of the document.
3. Last Occurrence: the distance between the end of the document and the last occurrence of the candidate topic, measured in terms of the number of characters and normalized by the length of the document.
4. Occurrence Spread: the distance between the first and last occurrences of the candidate topic, measured in terms of the number of characters and normalized by the length of the document. This feature reflects the observation that candidates which are more evenly spread within the document have a higher keyphraseness probability.
5. Length: the number of words in the candidate phrase, i.e., the descriptor of the candidate topic. This feature reflects the general observation that multi-word phrases have a higher keyphraseness probability as they tend to be more specific and less ambiguous. The keyphrase annotation studies which adopt this feature (e.g., see [10, 13, 14, 16, 17]) compute the length of a candidate phrase by simply counting its number of words or characters. However, our approach is to: (a) split the hyphenated words, (b) count the stopwords as 0.5 and non-stopwords as 1.0, (c) normalize the count value by dividing it by 10.0, (d) eliminate candidates which either have a normalized value greater than 1.0 or those which do not contain any letters (e.g., numbers, numerical dates).
6. Lexical Diversity: the descriptor and non-descriptors of a given topic could appear in a document in various lexical forms. We calculate the lexical diversity by (a) case-folding and stemming all the lexical forms of the candidate topic which appear in the document, using an improved version of Porter stemmer called the English (Porter2) stemming algorithm [18]; (b) counting the number of unique stems minus one, so that the lexical diversity value would be zero if there is only one unique stem. Lexical diversity values are normalized by dividing them by the highest possible lexical diversity value between all topics in Wikipedia. As explained in Section 3, this feature is only used in the supervised ranking function to balance and complement the lexical unity feature.
7. Lexical Unity: inverse of lexical diversity calculated as: $1.0 - \textit{lexical diversity}$. Our assumption is that the candidates with higher lexical unity values would have a higher keyphraseness probability.
8. Average Link Probability: the average value of the link probabilities of all the candidate topic's lexical forms which appear in the document. The link probability of a lexical form is the ratio of the number of times it occurs in Wikipedia articles as a hyperlink to the number of times it occurs as plain text.
9. Max Link Probability: the maximum value of all link probabilities of the lexical forms for a candidate topic which appear in the document. Both the average and max

link probability features are based on the assumption that candidate topics whose descriptor and/or non-descriptor lexical forms appearing in the document have a high probability of being used as a hyperlink in Wikipedia articles, also would have a high keyphraseness probability.

10. Average Disambiguation Confidence: in many cases a term from the document corresponds to multiple topics in Wikipedia and hence needs to be disambiguated. For example, the term "Java" could refer to various topics, such as "Java programming language", "Java Island", etc. As described in [19], the Wikipedia-Miner uses a novel ML-based approach for word-sense disambiguation which yields an F-measure of 97%. We have set the disambiguator to perform a strict disambiguation, i.e., each term in the document can only correspond to a single topic which has the highest probabilistic confidence. The value of the *average disambiguation confidence* feature for a candidate topic is calculated by averaging the disambiguation confidence values of its descriptor and non-descriptor lexical forms that appear in the document.

11. Max Disambiguation Confidence: the maximum disambiguation confidence value among the lexical forms of a candidate topic which appear in the document. Both the average and max disambiguation confidence features are incorporated into the ranking function to reduce the likelihood of candidate topics with low disambiguation confidence values being ranked as top keyphrases.

12. Link-Based Relatedness to Other Topics: the Wikipedia-Miner measures the semantic relatedness between topics using a new approach called Wikipedia Link-based Measure (WLM). In this approach the relatedness between two Wikipedia articles/topics is measured according to the number of Wikipedia topics which discuss/mention and have hyperlinks to both the two topics being compared (see [20] for details). The *link-based relatedness to other topics* feature value of a candidate is calculated by measuring and averaging its relatedness to all the other candidates in the document.

13. Link-Based Relatedness to Context: the only difference between this feature and the *link-based relatedness to other topics* is that the relatedness of the candidate topic is only measured against those of other candidate topics in the document which are unambiguous, i.e., their descriptor and non-descriptor lexical forms occurring in the document have only one valid sense. Both the *link-based relatedness to context* and *link-based relatedness to other topics* features are designed to increase the likelihood of those candidate topics with high semantic relevance to other topics in the document being picked as top keyphrases. However, the former only takes into account the unambiguous topics in the document and therefore has high accuracy but low coverage, whereas the latter also includes the ambiguous topics which have been disambiguated based on their surrounding unambiguous context (i.e., unambiguous topics in the document) and therefore has lower accuracy but conclusive coverage.

14. Category-Based Relatedness to Other Topics: Our study shows that as of July 2011, 95% of Wikipedia articles are classified and on average each classified article belongs to 3.82 categories. When a candidate topic is classified, we can utilize its categorization data to measure its semantic relatedness to other candidates in the document. We measure the category-based relatedness of two Wikipedia topics as:

$$\text{Relatedness}(topic_1, topic_2) = 1 - \frac{\text{Distance}(topic_1, topic_2) - 1}{2D - 3}, \quad (1)$$

where D is the maximum depth of the taxonomy, i.e., 16 in case of the Wikipedia dump used in this work. The distance function returns the length of the shortest path between $topic_1$ and $topic_2$ in terms of the number of nodes along the path. The term $2D-3$ gives the longest possible path distance between two topics in the taxonomy, which is used as the normalization factor, i.e., $2 \times 16 - 3 = 29$. The shortest possible distance between two nodes/topics is 1 (in case of siblings) and the longest is $2D-3$. Therefore subtracting one from the outcome of the distance function results in a highest possible relatedness value of 1.0, e.g., $1 - (1-1) / (2 \times 16 - 3) = 1.0$, and a lowest possible relatedness value of 0.03, e.g., $1 - (29-1) / (2 \times 16 - 3) = 0.03$. Changing the divisor from $2D-3$ to $2D-4$ reduces the lowest possible relatedness value to zero, however we have adopted the former and instead assign a zero value to relatedness when either $topic_1$ or $topic_2$ are amongst the 5% of Wikipedia topics which are not classified. The value for *category-based relatedness to other topics* for each candidate is calculated by measuring and averaging its category-based relatedness to all the other candidates in the document.

15. Generality: the depth of the topic in the taxonomy measured as its distance from the root category in Wikipedia, normalized by dividing it by the maximum possible depth, and inversed by deducting the normalized value from 1.0. It ranges between 0.0 for the topics farthest from the root and unclassified ones, and 1.0 for the root.

16. Speciality: inverse of generality calculated as: $1.0 - generality$. This feature is only used in the supervised ranking function (see Section 3) to balance and complement the generality feature.

17. Distinct Links Count: total number of distinct Wikipedia topics which are linked in/out to/from the candidate topic, normalized by dividing it by the maximum possible distinct links count value in Wikipedia.

18. Links Out Ratio: total number of distinct Wikipedia topics which are linked out from the candidate topic, divided by the *distinct links count* value of the candidate. Our preliminary experiments show that the candidates with a higher ratio of links out to links in, have a higher keyphraseness probability.

19. Links In Ratio: total number of distinct Wikipedia topics which are linked in to the candidate topic divided by the *distinct links count* value of the candidate. This feature is only used in the supervised ranking function (see Section 3) to balance and complement the *links out ratio*.

20. Translations Count: number of languages that the candidate topic is translated to in the Wikipedia, normalized by dividing it by the maximum possible translations count value in Wikipedia.

3 Rank and Filtering

As discussed in Section 1, the function applied to rank all the candidates and filter out those with highest keyphraseness probabilities could be either supervised or unsupervised. Since all the features defined in Section 2 are normalized to range from 0.0 to 1.0, a simple unsupervised function could be defined as:

$$\text{Score}(topic_j) = \sum_{i=1}^{|F|} f_{ij} \tag{2}$$

which computes the sum of all feature values of a given candidate topic, $topic_j$, as its keyphraseness score. The feature set, F, does not contain the inverse features f_6, f_{16}, and f_{19} as they are only designed to be used in the supervised function. The main advantage of this unsupervised approach is that it does not involve a training process and, therefore, does not require any manually annotated documents for learning a rank and filtering function from. Hence, it may be readily applied to document collections across all domains with minimum effort. However, this approach forces a number of naive assumptions on the general properties of keyphrases in research documents, which negatively impact the accuracy performance of the rank and filtering function:

- All the summed features carry the same weight in respect to their capacity for measuring the keyphraseness probability of a candidate. This is a virtually impossible assumption as shown previously (e.g., see [14]).
- All the features correspond and contribute to the keyphraseness probability of candidates linearly. This assumption does not hold neither intuitively nor empirically [14]. For example, in case of positional features such as first occurrence (f_2) and last occurrence (f_3), only extreme values (>~0.9) should have a significant effect on the overall keyphraseness scores of candidates. Therefore, an exponential correspondence between the values of these features and the scores of candidates better captures the behavior of these features.

The above issues may be addressed to a large degree by adding a weight, w_i, and a degree parameter, d_i, to each feature in equation 3:

$$\text{Score}(topic_j) = \sum_{i=1}^{|F|} w_i f_{ij}^{d_i} \tag{3}$$

In an unsupervised setting, knowledge engineers would require to heuristically find and assign the (near) optimum values to these two parameters via examination of a collection of manually annotated documents. However, the optimum values for these parameters could change from one domain or dataset to another. Hence, in order to achieve the best performance, the learning process needs to be repeated whenever the underlying nature of the dataset changes. In our supervised approach we automate this learning process by utilizing genetic algorithms to learn the optimum values for the weight and degree parameters from a set of manually annotated documents. In this approach, the learning process consists of the following steps:

1. Generating an initial population consisting of a set of ranking functions with random weight and degree parameter values from within a predefined range.
2. The fitness of each individual ranking function in the population is evaluated via applying it to a set of training documents to rank and filter their most probable keyphrases, and comparing the resulted top n keyphrases per document with those assigned by human annotators.
3. Based on their fitness, a number of individuals in the current population are stochastically selected, crossed, and mutated to form a new population for the next generation.
4. Steps 2 and 3 are repeated successively until one of the following termination conditions is reached:

(a) A ranking function with the highest possible fitness is found, i.e., for all the documents in the training set, all the top n keyphrases resulted from applying the ranking function match those assigned by human annotators. In practice we use the inter-indexer consistency measure to evaluate the fitness of individual ranking functions (see Section 4 for details of the evaluation measure used).

(b) The threshold on the number of generations is invoked. We have defined a greedy thresholding scheme which initially allows a predefined number of generations specified by the *threshold* variable to pass, and from that point on it counts the number of generations that pass without any improvement in the fitness of their fittest individual compared to that of the previous generation. Each time there is an improvement, the counter is reset to zero. The iteration process terminates when the number of generations passed without improvement equals half the total number of generations passed. It should be noted that since we use elitism the fitness of the best individual in each generation is guaranteed to be equal or higher than the fitness of the best individual in the previous generation, and the proposed thresholding mechanism would not work effectively without elitism.

5. The best individuals of all the generations built since the last generation with an improvement up to the last one before termination, are stored to be used for rank and filtering high probability keyphrases in unseen documents.

The degree parameters are float numbers with values between 0.0 and 2.0, allowing each feature, f_i, to be scaled logarithmically ($0.0<d_i<1.0$), linearly ($d_i=1.0$), exponentially ($1.0<d_i\leq 2.0$), or to become neutralized ($d_i=0.0$). The weight parameters have the same type and range as the degree parameters, allowing the weight of each feature, i.e., the magnitude of its impact on the total keyphraseness score of a given candidate, to become nil ($w_i=0.0$), a fraction of neutral ($0.0<w_i<1.0$), neutral ($w_i=1.0$), or up to twice bigger than neutral ($1.0<w_i\leq 2.0$). The defined ranges allow the genetic algorithm to render the features that are too noisy or counterproductive redundant via setting their degree to zero, or setting their weight to zero (or close to it). This in effect automates the feature selection process.

In the unsupervised setting, the universal ranking function defined in Equation 2 is applied to unseen documents and the top n keyphrases with the highest keyphraseness probability are filtered out. In the supervised setting however, the data stored at the fifth step of the learning process is used to apply an individual or a set of ranking functions defined in Equation 3, whose weight and degree parameters are adjusted according to the general properties of keyphrases in the target dataset. We have developed and evaluated two different methods to this:

- **Last best:** in this method, simply the individual function with the highest fitness from the last generation before termination is applied to the documents to rank and filter their top n keyphrases.
- **Unique bests:** in this method, the final score of each candidate keyphrase in a given document is calculated as the sum of its scores from all the unique individual ranking functions created during the learning process, which have yielded the best fitness value (achieved before termination) with different weight and degree value sets. This ensembled scoring method takes into account all the variations of the weight and degree value sets which have yielded the final best fitness value.

4 Experimental Results & Evaluation

For evaluating the performance of our keyphrase annotation method, we have used a dataset called wiki-20 [21] created by Medelyan and Witten [13, 14]. The wiki-20 collection consists of 20 Computer Science (CS) related technical research reports, each manually annotated by fifteen different human teams independently. Each team consisted of two senior undergraduate and/or graduate CS students. The teams were instructed to assign about five keyphrases to each document from a controlled vocabulary of over two million terms which served as article titles (i.e. topic descriptors) in Wikipedia at the time the dataset was compiled. We follow the evaluation approach from [13, 14] and use the inter-indexer consistency formula proposed by Rolling [22] to measure the quality of keyphrases assigned to the test documents by our method via comparing them with those assigned by each team of human annotators:

$$\text{Inter - indexer consistency}(A,B) = \frac{2c}{a+b} \tag{4}$$

where *A* and *B* represent the two annotators whose inter-consistency is being measured, *a* and *b* are the number of terms assigned by each annotator, and *c* is the number of terms they have in common. The overall inter-indexer consistency score of an annotator is calculated by first measuring and averaging its inter-consistency with all the other annotators per document, and then averaging the results over all the documents. This measure is also used in the second step of the learning process described in Section 3, to evaluate the fitness of individual ranking functions in a given population.

In order to achieve a thorough evaluation of the performance of our method, we have conducted three rounds of evaluation, each consisting of three sets of experiments: (ExpA) 2-fold cross-validation, (ExpB) 4-fold cross-validation, and (ExpC) 20-fold cross-validation, which corresponds to Leave-One-Out Cross-Validation (LOOCV). The data and results of all the experiments are available for download (http://www.skynet.ie/~arash/zip/KA_Wiki20_WM1.2-R233_ECJ20.zip).

Table 1 presents the results of the three rounds of evaluation. In the first round, we set the threshold to 400 (population-size multiplied by 10) which forces the GA to go through a minimum of 800 generations before terminating the learning process. In the second round, we reduced the threshold value from 400 to 200 to speed up the learning process and measure the effect it has on the quality of results. In the third round however, we doubled the threshold from 400 to 800 to examine if having a larger threshold and lengthier learning process would result in an improved overall performance. The results indicate underfitting in case of the second round and overfitting in case of the third round, both resulting in an underperforming model. Table 2 compares the performance of our machine annotator on the wiki-20 dataset with human annotators, a baseline machine annotator based on TFIDF, two un-supervised machine annotators: the work of Grineva et al. [1], and CKE [2], and two supervised machine annotators: KEA++ (KEA-5.0) [13, 17] and Maui [14].

Table 1. results of the three rounds of evaluation.

Round	Dataset	Train				Test					Settings	
						Supervised			Avg. Un-supervised	Avg. Highest Possible	*nk*	5
		Avg. # of Gens	Avg. Time Mins.	Avg. Fitness	Avg. Highest Possible	Avg. Unique Bests	Avg. # of Unique Bests	Avg. Last Best			Selection method	Tournament
First	ExpA (2-fold CV)	800	20.93	37.1%	48.5%	33.4%	202	32.7%	30.7%	48.5%	*Elites*	1
	ExpB (4-fold CV)	3137	131.03	37.4%	48.5%	32.7%	100	32.8%	30.7%	48.5%	*Population*	40
	ExpC (LOOCV)	1510	75.19	36.7%	48.5%	33.5%	73	33.5%	30.7%	48.5%	Genomes	40
Second	ExpA (2-fold CV)	672	17.07	37.6%	48.5%	32.8%	84	33.5%	30.7%	48.5%	Chunks	2
	ExpB (4-fold CV)	584	24.34	36.8%	48.5%	31.4%	54	31.5%	30.7%	48.5%	Crossover	2 ps
	ExpC (LOOCV)	1000	49.83	36.5%	48.5%	32.9%	53	32.6%	30.7%	48.5%	*w,d*	[0,2]
Third	ExpA (2-fold CV)	2420	62.22	38.3%	48.5%	33.3%	194	33.5%	30.7%	48.5%	Mutation	Reset
	ExpB (4-fold CV)	2487	102.55	37.5%	48.5%	32.9%	218	33.1%	30.7%	48.5%	*Mutation probability*	0.05
	ExpC (LOOCV)	3104	154.38	37.1%	48.5%	32.8%	149	31.8%	30.7%	48.5%		

Table 2. Performance comparison with human annotators and rival machine annotators.

Method	Learning Approach	Number of Keyprases Assgined per document, *nk*	Avg. inter consistency with human annotators (%)		
			Min.	Avg.	Max.
TFIDF (baseline)	n/a - unsupervised	5	5.7	8.3	14.7
KEA++ (KEA-5.0)	Naïve Bayes	5	15.5	22.6	27.3
Grineva et al.	n/a - unsupervised	5	18.2	27.3	33.0
Maui	Naïve Bayes (all 14 features)	5	22.6	29.1	33.8
Maui	Bagging decision trees (all 14 features)	5	25.4	30.1	38.0
Human annotators (gold standard)	n/a - senior CS students	Varied, with an average of 5.7 per document	21.4	30.5	37.1
CKE	n/a - unsupervised	5	22.7	30.6	38.3
Current work	**n/a - unsupervised**	**5**	**19.1**	**30.7**	**37.9**
Maui	Bagging decision trees (13 best features)	5	23.6	31.6	37.9
Current work (LOOCV)	GA, threshold=800, unique bests method	5	12.3	32.8	58.1
Current work (LOOCV)	GA, threshold=200, unique bests method	5	13.9	32.9	56.7
Current work (LOOCV)	**GA, threshold=400, unique bests method**	**5**	**14.0**	**33.5**	**58.1**

5 Conclusion

In this paper, we introduced an unsupervised and a supervised GA-based keyphrase annotation method for scientific literature utilizing a large number of features derived from Wikipedia. We evaluated the performance of both methods in terms of the consistency of their resulted keyphrases for a collection of test documents with those assigned by human annotators. The results of the three rounds of evaluation show that both methods outperform their rivals and yield a performance above the average performance of human annotators in terms of overall inter-indexer consistency.

6 References

1. Grineva, M., Grinev, M., Lizorkin, D.: Extracting key terms from noisy and multi-theme documents. 18th international conference on World wide web. Madrid, Spain (2009)
2. Mahdi, A.E., Joorabchi, A.: A Citation-based approach to automatic topical indexing of scientific literature. Journal of Information Science 36, 798-811 (2010)
3. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: KEA: practical automatic keyphrase extraction. fourth ACM conference on Digital libraries. ACM, Berkeley, California, United States (1999)
4. Turney, P.D.: Learning Algorithms for Keyphrase Extraction. Inf. Retr. 2, 303-336 (2000)
5. Turney, P.D.: Coherent keyphrase extraction via web mining. Proceedings of the 18th international joint conference on Artificial intelligence, pp. 434-439. Mexico (2003)
6. Nguyen, T.D., Kan, M.-Y.: Keyphrase extraction in scientific publications. Proceedings of the 10th international conference on Asian digital libraries, pp. 317-326.Vietnam (2007)
7. Markó, K.G., Hahn, U., Schulz, S., Daumke, P., Nohama, P.: Interlingual Indexing across Different Languages. Computer-Assisted Information Retrieval - RIAO, pp. 82-99 (2004)
8. Pouliquen, B., Steinberger, R., Ignat, C.: Automatic annotation of multilingual text collections with a conceptual thesaurus. Ontologies and Information Extraction. Workshop at EUROLAN'2003, (2003)
9. Medelyan, O., Witten, I.H.: Thesaurus based automatic keyphrase indexing. Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries, pp. 296-297. USA (2006)
10. Medelyan, O., Witten, I.H.: Domain-independent automatic keyphrase indexing with small training sets. Journal of the American Society for Information Science and Technology 59, 1026-1040 (2008)
11. Milne, D., Medelyan, O., Witten, I.H.: Mining Domain-Specific Thesauri from Wikipedia: A Case Study. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 442-448. IEEE Computer Society (2006)
12. Medelyan, O., Milne, D., Legg, C., Witten, I.H.: Mining meaning from Wikipedia. Int. J. Hum.-Comput. Stud. 67, 716-754 (2009)
13. Medelyan, O., Witten, I.H., Milne, D.: Topic Indexing with Wikipedia. first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08). AAAI Press, US (2008)
14. Medelyan, O.: Human-competitive automatic topic indexing. Department of Computer Science, vol. PhD thesis. University of Waikato, New Zealand (2009)
15. Milne, D.: An open-source toolkit for mining Wikipedia. New Zealand Computer Science Research Student Conference, (2009)
16. Turney, P.D.: Learning to Extract Keyphrases from Text. National Research Council, Institute for Information Technology (1999)
17. Barker, K., Cornacchia, N.: Using Noun Phrase Heads to Extract Document Keyphrases. Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence, pp. 40-52. (2000)
18. Snowball, http://snowball.tartarus.org/algorithms/english/stemmer.html
19. Milne, D., Witten, I.H.: Learning to link with wikipedia. Proceedings of the 17th ACM conference on Information and knowledge management, pp. 509-518. USA (2008)
20. Milne, D., Witten, I.H.: An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08), Chicago, I.L (2008)
21. Wiki20, http://maui-indexer.googlecode.com/files/wiki20.tar.gz
22. Rolling, L.: Indexing consistency, quality and efficiency. Information Processing & Management 17, 69-76 (1981)