# Keyphrase Extraction by Synonym Analysis of *n*-grams for E-Journals Categorisation

Richard Hussey, Shirley Williams, Richard Mitchell
School of Systems Engineering
University of Reading
Reading, United Kingdom
{r.j.hussey, shirley.williams, r.j.mitchell}@reading.ac.uk

***Abstract*—Automatic keyword or keyphrase extraction is concerned with assigning keyphrases to documents based on words from within the document. Previous studies have shown that in a significant number of cases author-supplied keywords are not appropriate for the document to which they are attached. This can either be because they represent what the author *believes* the paper is about not what it actually is, or because they include keyphrases which are more classificatory than explanatory e.g., "University of Poppleton" instead of "Knowledge Discovery in Databases". Thus, there is a need for a system that can generate appropriate and diverse range of keyphrases that reflect the document. This paper proposes a solution that examines the synonyms of words and phrases in the document to find the underlying themes, and presents these as appropriate keyphrases. The primary method explores taking n-grams of the source document phrases, and examining the synonyms of these, while the secondary considers grouping outputs by their synonyms. The experiments undertaken show the primary method produces good results and that the secondary method produces both good results and potential for future work.**

***Keywords- Automatic tagging, Document classification, Keyphrases, Keyword extraction, Single document, Synonyms, Thesaurus***

## I. INTRODUCTION

Keywords are words used to identify a topic, theme, or subject of a document, or to classify a document. They are used by authors of academic papers (such as all papers about "metaphor" or "leadership"), by libraries to allow people to locate books (such as all books on "Stalin" or "romance"), and other similar uses. The keywords for a document indicate the major areas of interest within it.

A keyphrase is a short phrase of, perhaps, one to five words, which fulfils a similar purpose, but with broader scope for encapsulating a concept. While this may be considered the authors' contention, it is inferred that a short phrase of a few linked words contains more meaning than a single word alone, e.g., the phrase "natural language processing" is more useful than just the word "language".

Frank et al. [1] discuss two different ways of approaching the problem of linking keyphrases to a document. The first, keyphrase assignment, is to assume a set and given list of keyphrases or categories which can be assigned to the document. The computational problem for this approach is then to determine a mapping between documents and categories using already classified documents as learning aids. The second approach, keyphrase extraction, assumes there is no restricted list and instead attempts to use any phrase from the document (or ones constructed via a reference document) to serve as the keyphrases.

Previous research [2][3] has shown that for any given group of documents with keyphrases, there is a small number which are frequently used (examples include "shopping" or "politics" [3]) and a large number with low frequency (examples include "insomnia due to quail wailing" or "streetball china" [3]). The latter set is too idiosyncratic for widespread use; generally, even reuse by the same author is unlikely. Therefore, part of the issue of both keyphrase assignment and extraction is locating the small number of useful keyphrases to apply to the documents.

This project is concerned with keyphrase extraction and, as such, this paper covers the background research into keyword/keyphrase generation, outlines a proposed solution to the problem, and compares the performance to manually assigned keyphrases. The main aim is to take an arbitrary document (in isolation from a corpus) and analyse the synonyms of word-level *n*-grams to extract automatically a set of useful and valid keywords, which reflect the themes of that document. The words of the document are analysed as a series of *n*-grams, which are compared to entries in a thesaurus to find their synonyms and these are ranked by frequency to determine the candidate keywords. The secondary aim is to look at a method of grouping the theme outputs into clusters, so that the results did not just show the most common theme swamping out any others.

The rest of the paper comprises the background and state-of-the-art (Section II), the implementation and results gained (Section III), a discussion (Section IV), and conclusions and suggestions for future work (Section V).

## II. BACKGROUND

The background research into automatic keyword generation has shown that existing work in these areas focuses on either cross analysing a corpus of multiple documents for conclusions or extrapolating training data from manual summaries for test documents. While manual summaries generally require multiple documents to train upon they do not need to compare each component of the corpus to all other components. Instead, they try to extrapolate the patterns between the pairs of documents and manual summaries in the training set.

## A. *Single Documents*

Single document approaches make use of manual summaries or keyphrases to achieve their results. Tuning via manual summaries attempts to replicate the process by which a human can identify the themes of a document and reduce the text down to a summary/selection of keyphrases. The general approach taken involves a collection of documents (with associated human summaries) and a given method is applied to draw relationships between the document and the summary. From this, new documents (generally a test corpus that also contains human summaries) are subject to the derived relationships to see if the summaries produced by the system are useful and usable.

For creating summaries, Goldstein et al. [4] set out a system based upon assessing every sentence of the document and calculating a ranking for its inclusion in a summary. The authors made use of corpora of documents for which assessor-ranked summary sentences already existed, and attempted to train the system to produce similar or identical sentences.

A different approach was taken by the *Stochastic Keyword Generator* [5], a proposed system for classifying help desk problems with short summaries. Submitted e-mails varied in their description of the problem and often contained duplicated or redundant data. Therefore, the authors created a system that would attempt to create a summary similar to those manually created by the help desk staff: concise, precise, consistent, and with uniform expressions. Their system uses a corpus of e-mails with manual summaries and ranks source words for inclusion based on the probability that they will occur based on the probability from its training data.

For producing keyphrases, Barker and Cornacchia [6] propose a system that takes into account not only the frequency of a "noun phrase" but also the head noun. For example, tracking "the Canadian Space Agency" should also track counts of "the Space Agency" or "the Agency". Wermter and Hahn [7] examine a method of ranking candidate keyphrases using the limited paradigmatic modifiability (LPM) of each phrase as a guide to locating phrases with low frequency but high interest to the document.

## B. *Multiple Documents*

Multiple document approaches take a corpus and attempt to analyse relationships between the component elements to create methods for dealing with unseen elements. Most of these approaches are based on examining parts of an individual document in the corpus and then examining how that differs across the other documents.

"*TagAssist*" [2] makes use of a continually updated corpus of blog posts (supplied by [3]) and author-supplied tags to suggest tags for new blog posts. The system compares the author's tags and content of blog posts to work out the relationships that prompt the former to be chosen to represent the latter. Their baseline system worked on a simple frequency count for determining output. Evaluated by ten human judges (unaware of which system produced each tags), the results showed that the original tags were the most appropriate (48.85%) with *TagAssist* coming in second (42.10%), and the baseline system last (30.05%).

The *C-Value* and *NC-Value* [8] are presented as methods for ranking "term words" taking into account phrase length and frequency of its occurrence as a sub-string of another phrase. *TRUCKS* [9] extends the *NC-Value* work, combining it with [10], to use contextual information surrounding the text to improve further the weightings used in the *NC-Value*.

Extra data may be used to gain more information on the relationships between the components, often gained from reference documents. Joshi and Motwani [11] make use of a thesaurus to obtain extra meaning from keywords so their program, "*TermsNet*", can observe keywords in their original context in attempt to link said keywords in "non-obvious ways". Scott and Matwin [12] use the *WordNet* lexical database [13] to find the hyponyms and feed this information to the Ripper machine learning system. Wei et al. [14] demonstrate such a system that uses *WordNet* to generate keywords for song lyrics. Their approach clusters the words of a song using *WordNet's* data to link words across the song. Keywords are then found at the centres of these links.

## III. Implementation and Results

The basis of this work is the examination of a document with reference to its synonyms and therefore the main bulk of the coding of the system related to this and the associated thesaurus file. The input thesaurus corpus for analysis was Roget's "*Thesaurus of English Words and Phrases*" [15] and was chosen due to availability and because initial prototyping had shown that WordNet [13] (normally used within the discipline) performed less well for this application (see also Section V).

The system was tested on a number of papers taken from a collection of online e-journals, Academics Conferences International (ACI) [16]. There were five e-journals in this collection, each on a different topic and they were analysed separately. The topics were *Business Research Methods* (EJBRM), *E-Government* (EJEG), *E-Learning* (EJEL), *Information Systems Evaluation* (EJISE), and *Knowledge Management* (EJKM).

For each of these e-journals the authors supply keywords/phrases. The baseline evaluation of this work is to compare the keyphrases supplied by the author with those identified by the system under consideration. A match is assumed if one author's keyphrase matches a system-supplied keyphrase using a naïve text-matching method. This method would match the word "know" with both the words "know" and "knowledge".

Table I shows the baseline results for the study, which were established by running the system with only unigrams and no clustering, and outputting only the most common keyphrase for each paper.

For each of the methods described below the thesaurus was loaded into the program and stored as a list of linked pairs of data, consisting of a unique Key (base word in the thesaurus) and an associated Value (its synonyms). The keys ranged from unigram word entries up to 7-gram phrases.

TABLE I. BASE LINE RESULTS

| Journal | Papers | Matched | Percentage |
|---|---|---|---|
| EJBRM | 72 | 2 | 2.78% |
| EJEG | 101 | 3 | 2.97% |
| EJEL | 112 | 16 | 14.29% |
| EJISE | 91 | 6 | 6.59% |
| EJKM | 110 | 15 | 13.64% |
| **Average** | | | 8.47% |

The project was split into two methods: the *n*-gram study and the clustering study. The following sections outline these approaches and the results from each.

## A. *The n-gram study*

For the *n*-gram study, the words from the source document were split into a number of *n*-gram lists, from unigrams up to 7-grams. For all of the lists the entries overlapped so that all combinations of words from the text were included. E.g., if the source text were "The quick fox jumped" then the bigrams would be "The quick", "quick fox", and "fox jumped" and the trigrams would be "The quick fox", and "quick fox jumped". For each document, the results of each of the *n*-grams were combined and considered together to determine the overall output.

Each time the *n*-gram appeared in the source text, its frequency in its word list was increased by n. The unigrams were then stemmed (to remove plurals, derivations, etc.) using the Porter Stemming Algorithm [17], and added to the list with combined frequencies from each of the unigrams that reduced to that stem. The resultant corpus of *n*-grams and stems was then compared to the entries in the thesaurus:

- For each word (Key) in the thesaurus, compare the *n*-gram to the associated synonyms (Value).
- For each synonym that matches, add the word (Key) to a list, and increase its frequency value by the value of the *n*-gram.
- Sort the list by frequency and output the top *r* ranked items (in this study, *r* was chosen to be 5).

The *n*-gram results showed a reasonable improvement (52%) over the baseline, as can be seen in Table II. The increase measures the performance compared to the results from Table I.

TABLE II. RESULTS OF *N*-GRAM STUDY

| Journal | Papers | Matched | Percentage | Increase |
|---|---|---|---|---|
| EJBRM | 72 | 25 | 34.72% | 31.94% |
| EJEG | 101 | 71 | 70.30% | 67.33% |
| EJEL | 112 | 72 | 64.29% | 50.00% |
| EJISE | 91 | 39 | 42.56% | 35.97% |
| EJKM | 110 | 94 | 85.45% | 71.84% |
| **Average** | | | 60.69% | 52.22% |

## B. *The clustering study*

The clustering algorithm attempts to extend the *n*-gram algorithm to group the keyphrases into "clusters" by finding the keyphrases that are of a similar theme and returning a single keyphrase for that group. For example, the word "recovery" can mean either "acquisition" or "taking" [15]. The base system therefore could return multiple versions of the same concept as keyphrases. By clustering the results, the attempt was to prevent a single, "popular", concept dominating and allow the other themes to be represented. The method for this was:

- For each word (Key) in the thesaurus, compare the *n*-gram to the associated synonyms (Value).
- For each synonym that matches, add the word (Key) to a list, and increase its frequency value by the value of the *n*-gram divided by the number of associated synonyms (number of entries in Value).
- Then, for each Key entry in the thesaurus check to see if the frequency is equal to the highest frequency value in the found in the preceding step.
- For each synonym entry associated with the Key, add the synonym to a second list of words and increase its value by one.
- Sort the second list by frequency and output the top *r* ranked items (in this study, *r* was chosen to be 5).

The clustering results show only a small improvement over the *n*-gram alone, as can be seen in Table III. The increase measures the performance compared to the results from Table I.

TABLE III. RESULTS OF CLUSTERING STUDY

| Journal | Papers | Matched | Percentage | Increase |
|---|---|---|---|---|
| EJBRM | 72 | 31 | 43.06% | 40.28% |
| EJEG | 101 | 73 | 72.28% | 69.31% |
| EJEL | 112 | 77 | 68.75% | 54.46% |
| EJISE | 91 | 46 | 50.55% | 43.96% |
| EJKM | 110 | 94 | 85.45% | 71.81% |
| **Average** | | | 64.72% | 56.25% |

## IV. DISCUSSION

The results show that while using *n*-grams on their own produce an increase in the matched output, extending the algorithm to include clustering produced a small improvement. However, when the clustering algorithm was run on a system using only unigrams, it performed just as well (identically in fact) as with a greater number of *n*-grams. This suggests that the clustering algorithm is of more use to the project aims than the initial *n*-gram work, and provides an interesting area to expand upon in future work.

In addition to this, some of the keywords submitted by the authors may be tags instead and these display meta-data that can often be irrelevant to the understanding of the document. An example seen in the corpus was the keyword "University of Birmingham" because the author of that paper worked there. This is valid as a tag but as a keyword, it does not indicate a topic or a theme to which the document holds (other than in a rare case where the paper is about the University of Birmingham). This would therefore lower the chances of keyphrases being matched as the comparison data is filled with `noise'.

The synonyms are currently analysed context-free, and thus for a word with multiple meanings (e.g., "recovery" can mean "acquisition", "improvement", or "restoration" [15]) every occurrence of that word is treated the same. This means that a document equally about "improvement" and "restoration" could end up with the theme of "recovery" which (while a correct assumption) may not give the right meaning.

## V. Conclusion and Further Work

The approach to synonym analysis developed in this paper showed good results for the test corpora used and potential for future study. Further study is required to compare the system to ones developed in similar areas, but this should provide a solid framework for taking the project forward.

The results also show that the use of *n*-grams, while useful on its own, has no effect upon the percentage match for the system. This does not, however, mean that the keywords produced may not be more useful to the user, as they could be different enough not to match the success criteria but still relevant.

The results themselves were evaluated against the keywords submitted by the authors of the papers. *TagAssist* [2] showed that in 54.15% of cases, author keywords were judged as being inappropriate for the work with which they were associated. Therefore, when interpreting the results (which averaged around 60% matches) it should be remembered that they are produced by matching the output against the author keywords, which may be less than perfect for the task. A new method of evaluating the results is, therefore, needed.

Another area of further work for this project is conducting more experiments to determine why the *WordNet* thesaurus performed worse in initial trials, and whether it is better at certain subject classes of documents (e.g., a medical corpus vs. a computer science corpus).

## Acknowledgment

The authors would like to thank the School of Systems Engineering for the studentship, which enabled this project, and the contributions from the reviewers to this paper.

## References

[1] E. Frank, G.W. Paynter, I.H. Witten, C. Gutwin, and C.G. Nevill-Manning. 1999. "Domain-Specific Keyphrase Extraction", Proceedings 16th International Joint Conference on Artificial Intelligence, pp. 668–673. San Francisco, CA Morgan Kaufmann Publishers.

[2] S.C. Sood, S.H. Owsley, K.J. Hammond, and L. Birnbaum. 2007. "TagAssist: Automatic Tag Suggestion for Blog Posts". Northwestern University. Evanston, IL, USA. http://www.icwsm.org/papers/2--Sood-Owsley-Hammond-Birnbaum.pdf [Last accessed: 13 December 2010]

[3] Technorati. 2006. "Technorati". http://www.technorati.com [Last accessed: 13 December 2010]

[4] J. Goldstein, M. Kantrowtiz, V. Mittal, and J. Carbonell. 1999. "Summarising Text Documents: Sentence Selection and Evaluation Metrics", ACM, pp. 121–128. Language Technologies Institute, Carnegie Mellon University, Pittsburgh, USA.

[5] C. Li, J. Wen, and H. Li. 2003. "Text Classification Using Stochastic Keyword Generation", Twentieth International Conference on Machine Learning (ICML), pp 464–471. Washington DC. https://www.aaai.org/Papers/ICML/2003/ICML03-062.pdf [Last accessed: 13 December 2010]

[6] K. Barker and N. Cornacchia. 2000. "Using Noun Phrase Heads to Extract Document Keyphrases", AI '00: Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence. pp. 40–52). London.

[7] J. Wermter and U. Hahn. 2005. "Paradigmatic Modifiability Statistics for the Extraction of Complex Multi- Word Terms". Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP) pp. 843–850. Vancouver Association for Computational Linguistics.

[8] K. Frantziy, S. Ananiadou, and H. Mimaz. 2000. "Automatic Recognition of Multi-Word Terms: the C-value/NC-value Method", International Journal on Digital Libraries , 3 (2), pp. 117-132.

[9] D. Maynard and S. Ananiadou. 2000. "TRUCKS: a model for automatic multi-word term recognition". Journal of Natural Language Processing, 8 (1), pp. 101-125.

[10] D. Maynard and S. Ananiadou. 1999. "Term extraction using a similarity-based approach". Recent Advances in Computational Terminology, pp. 261–278.

[11] A. Joshi and R. Motwani. 2006. "Keyword Generation for Search Engine Advertising", IEEE International Conference on Data Mining, pp. 490–496.

[12] S. Scott and S. Matwin. 1998. "Text Classification Using WordNet Hypernyms", Proceedings of the Association for Computational Linguistics, pp. 38–44.

[13] G.A. Miller, C. Fellbaum, R. Tengi, P. Wakefield, and H. Langone. 2005. "WordNet". Princeton University. http://WordNet.princeton.edu [Last accessed: 13 December 2010]

[14] B. Wei, C. Zhang, and M. Ogihara. 2007. "Keyword Generation for Lyrics", Austrian Computer Society (OCG). Comp. Sci. Dept., U. Rochester, USA. http://ismir2007.ismir.net/proceedings/ISMIR2007_p121_wei.pdf [Last accessed: 13 December 2010]

[15] P.M. Roget. 1911. "Roget's Thesaurus of English Words and Phrases (Index)". http://www.gutenberg.org/etext/10681 [Last accessed: 13 December 2010]

[16] Academics Conferences International. 2009. "ACI E-Journals". http://academic-conferences.org/ejournals.htm [Last accessed: 13 December 2010]

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