TESTING THE USE OF N-GRAM GRAPHS IN SUMMARIZATION SUB-TASKS

GEORGE GIANNAKOPOULOS, VANGELIS KARKALETSIS, AND GEORGE VOUROS

Abstract. Within this article, we sketch the set of generic tools we have devised and used within the summarization process and the domain of summary evaluation, focusing on how the tools were used within the TAC 2008 summarization update challenge. The tools have a common underlying theory and provide utility in various aspects of the Natural Language Processing domain. Within this study we elaborate on query expansion, content matching and filtering, redundancy removal as well as summary evaluation.

1. Introduction

We have been developing methods of text representation and corresponding algorithms that may offer generic utility on NLP tasks. Within the scope of this research we have used the n-gram graph representation, which offers rich information on the cotopy and contextual relations between character or word n-grams, throughout the whole pipeline of the proposed summarization system. The application of the aforementioned representation and a set of generic algorithms on the tasks of multi-document summary extraction and automatic evaluation of summaries have offered promising results and interesting conclusions.

The presented representation and algorithms have specifically been used in query expansion, content matching and filtering, and redundancy removal, within the summarization system. However, the summarization system is only a prototype, lacking the sophistication to sketch the method's true potential. We have also used the AutoSummENG evaluation method [GKVS08] on the TAC 2008 corpus with promising results.

2. System Overview

The system we have developed and applied on the TAC 2008 corpus is a prototype system, based on a few basic concepts and practices of existing summarization literature. In brief, the system finds sentences that contain chunks of words resembling the query text and adds the most promising sentences to the candidate summary, making sure no information is repeated. The process is repeated until we reach the desired length of summary text. Within our runs we have used a variation of the system that applies trivial query expansion on the queries before judging promising sentences. The simple process we have adopted poses the questions about how the query and the candidate sentences are represented, how similarity is measured between sentences and the query, how the query expansion is performed and how redundancy is avoided when selecting sentences. Within the following sections we elaborate on the answers of these questions, but first we introduce the reader to the representation and corresponding algorithms that have been used throughout the whole system.

2.1. **Representation and Basic Algorithms.** In the domain of natural language processing, there have been a number of uses for the analysis of texts in *n-grams*. An n-gram is a, possibly ordered, set of words or characters, containing n elements. N-grams have been used in summarization and summary evaluation [BV04, LH03, CS04]. In the automatic summarization domain, n-grams appear as word n-grams, as happens in the ROUGE/BE family of evaluator methods [HLZ05, Lin04]. Another related application is the n-gram fuzzy matching, which detects similar portions of text, even if other words appear between the n-gram words in the text [Lin04].

Trying to remain language independent, while allowing for different types of the same word, as well as trying to capture higher order relations between words (*i.e.* "neighbor of a neighbor" and sequence information), our method represents texts by using character n-grams positioned within a context-indicative graph. We shall call this construct an *n-gram graph*.

To create the n-gram graph, a window of length D_{win} runs over the summary text. We consider the window to be centered at the beginning of the current n-gram, we will call N_0. If N_0 is located at position p_0 in the text, then the window will span from $p_0 - [\frac{D_{\text{win}}}{2}]$ to $p_0 + [\frac{D_{\text{win}}}{2}]$, taking into account *both preceding and following* characters or words. An edge is created for every n-gram that can be found within the given window, near N_0. Each neighbourhood indicative edge is weighted based on the number of window co-occurrences of the neighbours within the text. Therefore, if we find the n-gram "do" to be within neighborhood distance from the n-gram "it" five times within a given text, the corresponding graph will contain an edge between "do" and "it" and this edge will have a weight of 5.

To represent a character sequence or text we can use a set of n-gram graphs, for various n-gram ranks (*i.e.* lengths), instead of a single n-gram graph. To compare a sequence of characters in the form of a chunk, a sentence, a paragraph or a whole document, we apply variations of a single algorithm that acts upon the n-gram graph representation of the character sequences. The algorithm is actually a similarity measure between two n-gram graph sets corresponding to two texts T_1 and T_2.

To compare the texts (or character sequences in general) T_1 and T_2, we need to compare their representations. Given that the representation of a text T_i is a set of graphs $\mathbb{G}_i$, containing graphs of various ranks, we use the *Value Similarity (VS)* for every n-gram rank, indicating how many of the edges contained in graph G^i are contained in graph G^j, considering also the weights of the matching edges. In this measure each matching edge e having weight w_e^i in graph G^i contributes $\frac{\text{VR}(e)}{\max(|G^i|,|G^j|)}$ to the sum, while not matching edges do not contribute (consider that if an edge $e \notin G^i$ we define $w_e^i = 0$). The *ValueRatio (VR)* scaling factor is defined as:

$$\text{VR}(e) = \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)} \quad (1)$$

The equation indicates that the *ValueRatio* takes values in $[0, 1]$, and is symmetric. Thus, the full equation for *VS* is:

$$\mathrm{VS}(G^i, G^j) = \frac{\sum_{e \in G^i} (\mu(e, G^j) \times \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)})}{\max(|G^i|, |G^j|)} \tag{2}$$

$\mu(e, G^j)$ is the membership function, which returns 1 if e belongs to G^j, else it returns 0. VS is a measure converging to 1 for graphs that share both the edges and similar weights, which means that a value of VS = 1 indicates perfect match between the compared graphs. Another important measure is the *Averaged Value Similarity (AVS)*, which is computed as:

$$\mathrm{AVS}(G^i, G^j) = \frac{VS}{\frac{\min(|G^i|, |G^j|)}{\max(|G^i|, |G^j|)}} \tag{3}$$

The fraction $\mathrm{SS}(G^i, G^j) = \frac{\min(|G^i|, |G^j|)}{\max(|G^i|, |G^j|)}$, is also called Size Similarity. The overall similarity $\mathrm{VS}^O$of the sets $\mathbb{G}_1, \mathbb{G}_2$ is computed as the weighted sum of the VS over all ranks:

$$\mathrm{VS}^O(\mathbb{G}_1, \mathbb{G}_2) = \frac{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r \times \mathrm{VS}^r}{\sum_{r \in [L_{\min}, L_{\mathrm{MAX}}]} r} \tag{4}$$

where VS^r is the VS measure for extracted graphs of rank r in $\mathbb{G}$, and $L_{\min}$, L_{MAX} are arbitrary chosen minimum and maximum n-gram ranks.

Given two instances of n-gram graph representation G_1, G_2, there is a number of operators that can be applied on G_1, G_2 to provide the n-gram graph equivalent of union, intersection and other such operators of set theory. For example, let the the merging of G_1 and G_2 corresponding to the union operator in set theory be $G_3 = G_1 \cup G_2$, which is implemented by adding all edges from both graphs to a third one, while making sure no duplicate edges are created. Two edges are considered duplicates of each other, when they share the equal vertices[1]. The invention of the operator is actually non-trivial, because a number of questions arise, such as the handling of weights on common edges after a union operation. In our implementation we have decided that in the union operator will average existing edge weights for every common edge into the corresponding new graph edge[2].

Overall, we have defined the following operators.

- The merging or union operator $\cup$, returning all the edges of two graphs, both common and uncommon, with averaged weights where applicable.
- The intersection operator $\cap$, returning the common edges of two graphs, with averaged weights.
- The delta operator (also called *all-not-in* operator) $\triangle$ returning the subgraph of a graph G_1 that is not common with a graph G_2. This operator is non-symmetric, *i.e.* $G_1 \triangle G_2 \neq G_2 \triangle G_1$, in general.
- The inverse intersection operator $\bigtriangledown$ returning all the edges of two graphs that are not common between them. This operator is symmetric, *i.e.* $G_1 \bigtriangledown G_2 = G_2 \bigtriangledown G_1$.

[1] The equality between vertices can be customized. Within our applications two vertices are the same if they refer to the same n-gram, which can be checked by simple string matching.

[2] We do not provide formalized futher definitions of the operators, because of space limitations.

Finally, the empty graph $\emptyset$ is considered to be a graph with no nodes and no edges.

2.2. **Content matching and filtering.** Concerning the content matching part of the presented summarization system, the following basic assumptions have been made.

- The *content* $C_{\mathbb{U}}$ of a *text set (corpus)* $\mathbb{U}$ is considered to be the intersection of all the graph representations of the texts in the set: $C_{\mathbb{U}} = \bigcap^{t \in \mathbb{U}} t$.
- A sentence S is considered more similar to the content $C_{\mathbb{U}}$ of a text set, as more of the sentence's *chunks* (sub-strings of a sentence) have an n-gram graph representation similar to the corresponding content representation. Every chunk's similarity to the content counts for the overall similarity of a sentence to the content.

The chunks of a sentence are extracted using an entropy-based approach. We first use a corpus, that can be different from the given corpus, to determine the probability $P(c|S_n)$ that a single given character c will follow a given character n-gram S_n, for every character c apparent in the corpus. The probabilities can then be used to calculate the entropy of the next character for a given character n-gram S_n.

The entropy measure indicates uncertainty, thus we have supposed that substrings of a character sequence where the entropy of $P(c|S_n)$ surpassed a statistically computed threshold represent candidate delimiters. Within the text to be chunked we seek the delimiters, after the end of which a new chunk begins. In our application we have only checked for unigrams, simple letters, as delimiters even though delimiters of higher rank can be determined. So, given a character sequence S_n and a set of delimiters $\mathbb{D}$, our chunking algorithm splits the string after every occurrence of a delimiter $d \in \mathbb{D}$.

Given the content definition and the chunking process, each sentence is assigned a score, which is actually the sum of the similarities of its chunks to the content. This process offers an ordered list of sentences. Then, a naive selection algorithm would select the highest-scoring sentences from the list, until the summary word count limit is reached. However, this would not take redundancy into account and thus this is where redundancy removal comes in.

2.3. **Redundancy Removal - Intra-summary and User-modeled Redundancy.** The redundancy removal process has two aspects, the *intra-summary* redundancy and the *inter-summary* or *user-modeled* redundancy. The intra-summary redundancy refers to the redundancy of a sentence in a summary, given the rest of the content of the summary. The inter-summary or user-modeled redundancy refers to the redundancy of information apparent when the summarization process does not take into account information already available to the reader.

In order to ensure intra-summary non-redundancy, one has to make sure that every sentence added only minimally repeats already existing information. To achieve this goal, we use the following process:

- Extract the n-gram graph representation of the summary so far, indicated as G_{sum}.
- Keep the part of the summary representation that does not contain the content of the corresponding document set $\mathbb{U}$, $G'_{\text{sum}} = G_{\text{sum}} \triangle C_{\mathbb{U}}$.

- For every candidate sentence (in the ranked list) that has not been already used
    - extract its n-gram graph representation, G_{cs}.
    - keep only $G'_{cs} = G_{cs} \triangle C_{\mathbb{U}}$, because we expect to judge redundancy for the part of the n-gram graph that does not refer to the necessary content.
    - assign the similarity between $G'_{cs}, G_{sum'}$ as the sentence redundancy score.
- For all candidate sentences (in the ranked list)
    - Set the score of the sentence to be its rank based on the similarity to $C_U U$ minus the rank based on the redundancy score.
- Select the sentence with the highest score as the best option and add it to the summary.
- Repeat the process until the word limit has been reached or no other sentences remain.

In the TAC 2008 corpus, systems are supposed to take into account the first of two sets per topic, set A, as prior user knowledge for the summary of set B of the same topic. We have used the content of the given set A, $C_{\mathbb{U}A}$, in the redundancy removal process by further merging the content of set B, $C_{\mathbb{U}B}$, to G_{sum} after the first step of the process. In other words, the content of set A appears to always be included in the current version of the summary and, thus, new sentences avoid redundancy.

2.4. **Query Expansion.** Query expansion is based on the assumption that a set of words related to an original query can be used as part of the query itself to improve the recall and usefulness of the returned results. In the literature much work has indicated that query expansion should be carefully applied in order to improve results [Voo94, QF93].

In our approach, we have used query expansion in a very simplistic way, looking up all query words in WordNet [MBF^{+}90] and appending the resulting *"overview of senses"*-contained words to the query. This approach does not function effectively, because much noise is inserted within the query; on the other hand, this experimentation with query expansion offers some insight concerning the usefulness of query expansion for our approach on the query-based summarization task.

More precisely, in the query expansion process we use a data structure we call *semantic index*. The semantic index represents links between n-grams and their semantic counterparts, implemented as WordNet definitions. The semantic index also supplies a facility to compare the meanings of two terms, by comparing the n-gram graph representation of their WordNet definitions. The semantic index uses another construct, called the *Symbolic Graph*, to link n-grams of various ranks, indicating which n-gram consists of which n-grams of lower rank. Thus, the semantic index can extract the meaning of n-grams that do not have a meaning on their own (*e.g.* sub-word parts), by assigning to an n-gram the meanings of any sub-n-grams it comes from. Only n-grams that have a definition in the WordNet are directly assigned a definition; all other n-grams inherit the meaning from their constituents.

For example, if we do not have a definition for the word "superman", but we have the definitions of "super" and "man", then our algorithm will return the joined meaning of "super" and "man". This helps assign meaning, even if a spelling error has been made, or the word is complex and no direct meaning has been attached to it.

For a given word w, a set of senses' overviews is returned by the semantic index; from these senses $s_i, i > 0$ we only utilize senses s_j with graph representations G_{s_j} that have more in common with the content $C_{\mathbb{U}}$ than a given threshold: $G_{s_j} \cap C_{\mathbb{U}} \neq \emptyset$ and $\mathrm{VS}(G_{s_j}, C_{\mathbb{U}}) > t, t \in \mathbb{R}^+$. Finally, the query is integrated in the content definition by merging the representation of the original query G_q and the representations G_{s_j} of all the j additional extracted senses to the original content, giving a new query-based content definition $C_{\mathbb{U}}'$. Having calculated $C_{\mathbb{U}}'$, we can judge important sentences simply by comparing the graph representation of each sentence to the $C_{\mathbb{U}}'$.

Even though the query expansion process was finally rather successful, in the original query expansion process noise was added, due to chunks like "an", "in" and "o" which were directly assigned the meanings of "angstrom", "inch" and "oxygen", correspondingly (also see experiments in section 3.2). This lowered the evaluation scores of our submitted runs. Using the senses' filter as shown here, the deficiency has been avoided.

2.5. **Summary Evaluation.** The representations and algorithms we have implemented[3] and presented briefly in section 2.1 were primarily used for evaluation purposes. The method is called AutoSummENG [GKVS08] and uses the character n-gram graph representation to represent both model and peer summaries, further applying the value similarity measure to compare a given summary to a set of models. The average similarity over model summaries of a given peer summary offers the *summary score.* The average of the summary scores of a given system offers the *system score.*

The parameters of the evaluation method include the minimum and maximum character n-gram sizes taken into account, as well as the maximum distance between n-gram taken into consideration to form the edges between neighboring n-grams. These parameters are derived from an a priori parameter estimation process that separates n-grams into meaningful ones, called *symbols*, and useless ones, called *non-symbols.* The distinction between symbols and non-symbols is based on statistical measures (see [GKVS08] for more on symbols, non-symbols and parameter estimation) and, as such, is language independent.

3. Experiments

The experiments conducted upon the TAC 2008 corpus were numerous, mostly to research aspects of the summarization process, but also to test our evaluation method upon the extracted summaries. Therefore, the experiments conducted are divided into the following categories.

Summary evaluation: We have applied our summarization system evaluation method to see how it correlates to human judgement. Furthermore, we have tried to create a complex evaluation method, based on the results of existing evaluation methods (like ROUGE/BE and AutoSummENG) and

[3] The toolkit used throughout the summarization process, as well as its evaluation part is called JInsect and is publicly available at `http://www.ontosum.org/?q=static/AutomaticSummarization` under LGPL licence.

AutoSummENG to ...	*Spearman*	*Kendall*	*Pearson*
Overall Responsiveness	0.8953 (< 0.01)	0.7208 (< 0.01)	0.8945 (< 0.01)
Linguistic quality	0.5390 (< 0.01)	0.3819 (< 0.01)	0.5307 (< 0.01)

Table 1. Correlation of the *system* AutoSummENG score to human judgement for peers only (p-value in parentheses)

AutoSummENG to ...	*Spearman*	*Kendall*	*Pearson*
Overall Responsiveness	0.3788 (< 0.01)	0.2896 (< 0.01)	0.3762 (< 0.01)
Linguistic quality	0.1982 (< 0.01)	0.1492 (< 0.01)	0.1933 (< 0.01)

Table 2. Correlation of the *summary* AutoSummENG score to human judgement for peers only (p-value in parentheses)

have tried to argue on why evaluation results cannot be significantly improved simply by using combinations of these methods. Throughout the next experiments AutoSummENG has been used to provide the measure of summary quality.

Query expansion: We have tested whether query expansion improves the overall quality of our summarization system.

We have purposely left out of the experiments the content selection and filtering method, because it is rather elementary and much has to be researched in order to achieve efficiency. It should be noted that we have used an elementary sentence splitter based on regular expressions, which further lowered performance. However, heavy experimenting with various parameters is ongoing and we plan to offer more promising results in future versions of the system. The current version of the summarization system offers a performance in the lower ranks of the evaluation on the TAC 2008 corpus, with system IDs 9 and 39 for the two runs based on different configurations (with and without query expansion).

3.1. **Summary evaluation.** Two aspects of the summary evaluation method have been examined:

- the correlation (Spearman's rho, Kendall's tau and Pearson correlation) of the *system* evaluation scores to the human system judgements (average overall responsiveness and average grammaticality). The system evaluation scores are calculated by the average scores of the summaries provided by a single system.
- the correlation of the *summary* evaluation scores to human judgement (overall responsiveness and linguistic quality). The summary evaluation score is the AutoSummENG score of a single summary given a set of model summaries.

The tables 1 and 2 indicate two important aspects of the summarization evaluation. The first has to do with the fact that the AutoSummENG method is good enough to judge system performance rankings. The second indicates that we should research a measure to indicate summary performance, in contrast to system performance. The latter problem is much harder and would also solve the system ranking problem, as the system performance is calculated as the average of the system summaries' performance.

We attempted to create a meta-estimator of system quality using n-grams of various ranks both at the word and character level. The performance of each system was described as a vector, the dimensions of which were the AutoSummENG performance of the system for different configurations (various n-gram sizes, word or character n-grams, various window sizes) as well as ROUGE/BE values. The performance only shifted slightly, which could be the work of chance and on some methods it even performed worse than a single evaluation method. The meta-estimator was created using a set of various machine learning techniques (decision trees, linear regression, multi-layer perceptron, SVM-based regression) but all had the same results: *no significant change*. We also tried using Principal Component Analysis (see [TK03] for more) to determine more important features, but little changed since it showed that all features should be used equally into a single vector. This have us a hint that using existing evaluation techniques combined does not offer the expected results. The reason for this can be explained by correlation: the results of existing evaluation methods are highly correlated statistically (also see [Dan05, Dan06]). This is normal, because they all aim to give an overall judgement of responsiveness. But, how about the other textual quality aspects?

We note that the correlation between overall responsiveness and linguistic quality is 0.3788 (Kendall's tau, p-value < 0.01). This means that they are correlated, but not strongly. We also deduce from the table that there are aspects of textual quality that cannot be well estimated at this point in time, like the linguistic quality. As this quality is important and not strongly correlated to the overall responsiveness measure, it seems that the reason for not being able to surpass the current level of performance in evaluating summaries and summarization systems is that we lack *statistically independent* judgements concerning orthogonal aspects of textual quality. If these judgements were performed, we would be able to judge quality better as a composition of the independent judgements.

3.2. **Query expansion.** As already noted, two runs were submitted based on our summarization system with system IDs 9 and 39. The run with ID 9 used query expansion, while the one with ID 39 did not. We have tried to see whether query expansion offers improvement over the original configuration. To judge the performance of the system we have performed a t-test over the summary AutoSummENG scores, which according to a Cramer-von Mises test [Tho02] can be considered normal. More precisely, we tested the average scores of the summaries (compared to all the corresponding topic models) in a paired t-test: the difference in the means of performance values was lowered by 0.07 (p-value < 0.01) when using the query expansion.

The original query expansion process (used in the submitted runs), simply extended the query by concatenating the overview of senses returned by the concept extraction process to the original query. This induced *much* noise, as shown in example 3.1.

Example 3.1. Query: *Airbus A380 Describe developments in the production and launch of the Airbus A380.*
Expansion terms: *air angstrom, angstrom unit, A describe, depict, draw development inch, in production angstrom, angstrom unit, A establish, set up, found, launch oxygen, O, atomic number 8 air angstrom, angstrom unit, A*

However, after the use of filtering on the senses, using the *exact same algorithms and representation* which was utilized throughout the system, we easily managed to only extract relevant senses. Even though in the original runs the query expansion module *seriously worsened* the results, after applying this simple sense filtering the performance was *slightly improved.* The example 3.2 shows the result of the improved expansion process over the query of example 3.1.

Example 3.2. Expansion terms: *air, describe, depict, draw, development, production*

As shown in the literature, query expansion should be implemented very carefully to offer improvement over non-expansion results. It should be noted that a paired t-test over the average summary performance per topic gave a 0.00005 difference between the means of summary AutoSummENG average values. This value accounts to about 0.1% of the standard deviation of the AutoSummENG values of all systems and, consequently, is not unimportant. It should be noted that the query expansion process most of the times offered few or no new words. However, the times it did provide new words, these words must have been useful, since the overall performance was slightly improved.

4. Future work

The summarization system we have implemented uses a common set of principles to face the problem of summarization from multiple documents. These principles depend on the wide applicability of a set of algorithms concerning the graph representation of texts, as well as on the operators that can be applied upon these representations. Even though the application has proved to be very promising on the summary evaluation domain, its performance as a generic tool for summarization modules depends on its exact use. In our case, the results indicated that there is much space for improvement, especially in basic functions of the system, as the sentence splitting module, that can seriously affect the quality of the output summary.

The experiments related to our summarization system specifically indicated that

- it is non-trivial to add query expansion techniques to improve the results summaries. The use of the designated query expansion method only slightly improves performance, given the described selection process. Further experiments should determine the optimal balance between strictness in the selection of relevant senses for the expansion and usefulness of the expansion module.
- we can extract senses from a text chunk by using statistial methods, like entropy-based chunking, and algorithms like the ones implemented in the Semantic Index and the Concept Extractor (see section 3.2). It would be interesting to attempt evaluation using a representation based on the extracted senses of a text and not its original form.
- the same approaches used for the summary evaluation process can be used to test redundancy and impose redundancy removal within the summarization process.
- the system we have implemented does not use any machine learning techniques, but offers promising results. This hints us that the combination of

the generic tools described herein together with machine learning optimization may offer even more positive results.

The experiments concerning AutoSummENG evaluation method indicated that:

- the correlation between existing evaluation methods and the measures of overall responsiveness and linguistic quality is rather different. This hints the need for new evaluation measures aiming for other *aspects* of a summary.
- the correlation of existing evaluation methods' results does not allow for an efficient combination of them as a new evaluation measure. In terms of information, very little is offered from the use of a second method in our effort to evaluate a system. This means that the axes of evaluation should be uncorrelated to each other, in order to function as independent judges and allow for the invention of a composite evaluation method through machine learning.

References

[BV04] Michele Banko and Lucy Vanderwende. Using n-grams to understand the nature of summaries. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 1–4, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[CS04] T. Copeck and S. Szpakowicz. Vocabulary usage in newswire summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 19–26. Association for Computational Linguistics, 2004.

[Dan05] H. T. Dang. Overview of DUC 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

[Dan06] H. T. Dang. Overview of DUC 2006. In *Proceedings of HLT-NAACL 2006*, 2006.

[GKVS08] G. Giannakopoulos, Vangelis Karkaletsis, G. Vouros, and P. Stamatopoulos. Summarization system evaluation revisited: N-gram graphs. *ACM Transactions on Speech and Language Processing*, 10 2008.

[HLZ05] E. Hovy, C. Y. Lin, and L. Zhou. Evaluating duc 2005 using basic elements. *Proceedings of DUC-2005*, 2005.

[LH03] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 71–78, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[Lin04] C. Y. Lin. Rouge: A package for automatic evaluation of summaries. *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26, 2004.

[MBF$^+$90] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[QF93] Y. Qiu and H.P. Frei. Concept based query expansion. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–169. ACM Press New York, NY, USA, 1993.

[Tho02] H. C. Thode. *Testing for Normality.* Marcel Dekker, New York, 2002.

[TK03] S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Academic Press, 2003.

[Voo94] E.M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69. Springer-Verlag New York, Inc. New York, NY, USA, 1994.