# Query Focused Document Summarization with LSI

**Eric Bell, Joshua Hou, Glenn Slayden**
University of Washington
Seattle, WA, USA
{ebbell, jshou, gslayden}@u.washington.edu

## Abstract

We present a query-focused multi-document extractive summarization system based on Latent Semantic Indexing (LSI). At the heart of this technique is the linear-algebraic technique of Singular Value Decomposition (SVD), which we apply to a term-by-sentence matrix representing both the corpus and queries, an arrangement which allows us to discard the left singular vectors, decreasing computation time. Post-processing incorporates both linguistically and statistically motivated factors. Our system was developed on the DUC 2006 topicsets and obtains non-jackknifed ROUGE F-scores of 0.8591 and 0.1441 for ROUGE-2 and ROUGE-SU4 respectively, corresponding to putative fifth place amongst the 35 teams from that year. Testing on held-out DUC 2007 topicsets was also strong and our system would have placed 16th in that year's competition. Moreover, in blind human evaluation of those results, we placed 1st among 9 teams in grammaticality, and 3rd in both referential clarity and structure-coherence. We obtained steady improvements throughout our development period which suggest that further improvement is possible with this general approach.

## 1 Introduction

Challenges in automatic document summarization include homography, which leads to poor precision, and synonymy, which leads to poor recall. Widely-known approaches include term-clustering models such as TF/IDF, but these incorporate strong term independence assumptions which can distort results.

An approach which relaxes this assumption by automatically considering transitive term co-occurrence across the input corpus is linear-algebraic Singular Value Decomposition (SVD). Applying this factorization to matrix of term-by-content observations has significant application to natural language processing. First suggested for information retrieval by Deerwester et al. (1990), the approach has since been adapted to indexing (Rosario, 2000), automatic synonymy, word sense disambiguation (Schütze, 1992), clustering, classification, and the instant application, document summarization (Hennig, 2009).

We present an original system which demonstrates the efficacy of the approach by obtaining putative fifth-ranked ROUGE scores amongst DUC 2006 entrants. Test data was provided by holding out DUC 2007 data; on these unseen topicsets, our system places 16th out of the 31 competitors. A feature of our system which contributed to its success was its performance; the ability to run an end-to-end experiment on all fifty topicsets in approximately two minutes allowed us to document 140 different experiments which resulted in 11%, 33%, and 19% increases in ROUGE-1, ROUGE-2, and ROUGE SU-4 scores, respectively, over our initial baseline implementation.

Section 2 presents an overview of the theoretical basis of our model. Section 3 surveys the processing pipeline of our system. We review some of our experiments and quantitative results which supervised the development of our system in Section 4. Section 5 discusses how we evaluated our system, and the final section discusses future research directions and summarizes this work.

## 2 Singular Value Decomposition

### 2.1 Theory

SVD is two-mode factor analysis, allowing it to operate on an $m \times n$ matrix of term and content vectors. By representing these vectors in a space where intersecting transitive co-occurrence relations constitute a cline of choosable dimensionality, the SVD simultaneously

achieves noise attenuation (smoothing) (Schütze, 1992, §5), redundancy detection, and a retrieval metric (Kontostathis and Pottenger, 2002).

The singular value decomposition is defined as

$$A_{m\times n} = U_{m\times d}\Sigma_{d\times d}(V_{n\times d})^T, \quad (1)$$

where $d = min(m, n)$. In our term-by-sentence application, $m$ is the number of the mixed n-grams in the vocabulary and $n$ is number of sentences. According to this configuration, the left singular vectors $U$ indicate the coordinates of each n-gram in the reduced space, the right singular vectors $V$ give the coordinates of each sentence, and $\Sigma$ is formed by arranging the singular values–which are the lengths of the principal semiaxes of the hyperelliptic projection of $A$ (Golub and Loan, 1996, 71)–along the main diagonal of an otherwise empty $d \times d$ matrix.

The power of the decomposition exists in the fact that, as shown in Golub and van Loan (1996, 72-73), for any $k$, the maximum-likelihood 2-norm (least-squares) approximation of rank-deficient $A_{m\times k}$ is given by

$$\hat{A}_{m\times k} = U_{m\times k}\Sigma_{k\times k}(V_{n\times k})^T. \quad (2)$$

This guarantee means that the $k-$dimensional space is usefully rotated so as to align the $k$ axes in the $k$ directions of greatest variation.

Unfortunately, it is not obvious how to determine the optimal $k$ (Madsen et al., 2004), although it is worth noting that only a single SVD operation is needed on a corpus to experiment with different values of $k$. In Section 3.6, we report on experiments with selecting $k$ that corresponds to a fraction of the singular value energy (Madsen et al., 2004), and how ROUGE evaluation instead suggests a simple fixed fraction of $m$.

It is possible to use SVD as a dimensionality-reduction prelude to conventional VSM-style content clustering, and this is an approach we investigated early in our research. We noted however, that the problem of choosing the $k$ parameter is then compounded by having to choose clustering parameters as well. Informed further by Schütze (1992), our extractive approach instead focuses on using cosine measure directly on the right singular vectors $V$ in order to retrieve those sentences that are closest to the query or queries. This approach is discussed in the next section.

### 2.2 $V$-Matrix Queries

In information retrieval (IR) and search applications against a fixed corpus, where queries are unknown when the corpus is given, it is common to build a query-agnostic SVD, and to transform queries into the SVD space later according to

$$\hat{q} = q^T U_{m\times k}{\Sigma^{-1}}_{k\times k}. \quad (3)$$

This is an important consideration for these applications because performing a singular value decomposition for a large term-by-instance matrix can be computationally intensive (Rosario, 2000, 5).

We adopt a different approach which is appropriate for situations where the queries and the corpus are both available at once, such as query-focused document summarization. In our implementation, we append the query sentences to the corpus sentences prior to the decomposition. This allows us to perform the SVD computation without requesting the left singular vectors $U$. Sentences and queries are thus simultaneously transformed into cosine-distance comparable vectors within the $V$ matrix. ALGLIB (Bochkanov and Bystritsky, 2010), the SVD computation library we use, is able to calculate the SVD significantly faster when we indicate that we do not need the left singular vectors, which, in turn, allows us to conclude more experimental trials.

Intuitively, a subtle difference between this approach and the query-agnostic method is that our approach allows the SVD to capitalize on term co-occurrence information that may be present within the queries themselves, and between the queries and the corpus. We construed this methodological consideration as applicable to this query-focused summarization task, while acknowledging that other applications may present different requirements that preclude the approach.

### 2.3 Applicability

In this section, we point out some assumptions and inherent characteristics of SVD modeling which have been discussed in the literature. A theoretical criticism of the application of SVD to document processing is that, like other least-squares methods, this form of matrix decomposition necessarily conditions the maximum-likelihood guarantee mentioned in Section 2.1 upon a normal distribution for the entries in the term-by-content matrix. This assumption is unlikely to be valid, as a Poisson distribution may better characterize the observation count data gathered in this and other related applications (Rosario, 2000). Although this does weaken the theoretical foundations of–and may preclude meaningful extension of–the approach, empirical results

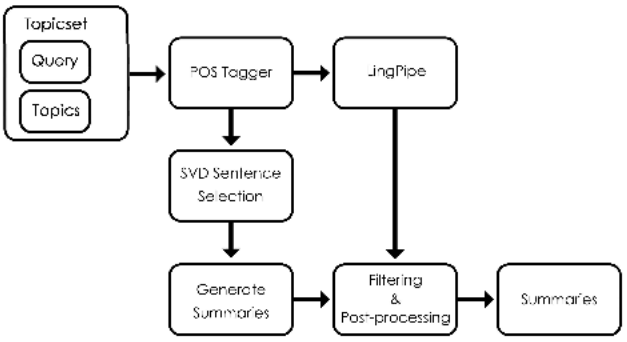


Figure 1: Processing pipeline schematic.

seem to justify the compromise, as they do in many other complex NLP tasks.

Other important considerations for selecting an SVD approach for an NLP task are that: the spatial rotation of the $A$ matrix vastly increases its storage costs, even when taking dimensionality reduction into account, as it is no longer extremely sparse (Rosario, 2000); the "meanings" of the reduced dimensions cannot be discerned (Schütze, 1992); and the SVD computation tends towards $O(n^3)$ (Lee, 2006).

These weaknesses being deemed few and manageable, the objective of our research was to empirically evaluate the suitability of the SVD to query-focused summarization. In particular, most NLP work with SVD has focused on term-by-document models over large numbers of documents and here we explore the particular characteristics of a term-by-sentence model over a smaller number of documents. In such a model, the term vocabulary is expected to be smaller, and the sentence vectors are also expected to be more sparse.

## 3 System Overview

A schematic of our modular pipeline is shown in Figure 1. The independent processing steps shown are described in the following sections. In order to facilitate experimentation, the full, research version of our pipeline stores intermediate results at each stage.

We also built a monolithic version of the system which enables simple end-to-end operation of the processing steps on plaintext inputs, without persisting intermediate results. A sample summary produced by this system is included in Figure 2.

### 3.1 Corpus Cleaning and Sentence Breaking

Our processing pipeline begins by assembling each topicset into a single XML file that is manipulated through subsequent stages. We wrote our own data cleaning and sentence breaking module for several reasons. The first is to allow us greater flexibility in accommodating data cleaning and sentence breaking to the specific requirements of subsequent modules. Secondly, in order to ensure the validity of our divergence statistics, we wished to ensure that our AQUAINT corpus (Graff, 2002) background language model was built using the same process as our topicsets. Finally, we wished to ensure adequate performance of the system to sustain a high throughput of experimental trials. Accordingly, this single-pass component is able to sentence-break the entire 3 gigabyte AQUAINT corpus in approximately 10 minutes.

The sentence breaker annotates every sentence in our system with a number of statistics, such as entropy and various divergence measures. Ready availability of these statistics in the system's internal object model assisted our research inquiries and we intended to use these statistics to inform topicset-dependent parameter selection throughout the pipeline. Our final system, however, does not incorporate statistical parameter selection since, as we discuss in Section 3.5, evaluation guided us towards simpler heuristics. Nevertheless, automatic parameterization is an area of some interest (Kontostathis and Pottenger, 2002) which may benefit from future research inquiry.

### 3.2 Stanford POS Tagging

After cleaning and sentence breaking, the queries and sentences for each topicset are tagged using the Stanford POS tagger (Toutanova et al., 2003). This uses the left three words model trained on the Wall Street Journal corpus. This POS data integrated back into the XML topicset files, as attributes on the ⟨word⟩ XML tags, by a special pipeline step so that this information is accessible to all subsequent steps in processing.

Because the POS tagger strips punctuation and splits some tokens slightly differently than our custom sentence breaker, a resolution step between the POS tagged sentences and original data was necessary. This step involves simulating the punctuation stripping and then resolving the differences in tokenization.

### 3.3 Lingpipe Anaphora Resolution

Next, each query and sentence in the topicset is submitted, as a contiguous text, to the Lingpipe's online

Coreference resolution suite. From their web server, we receive back the documents with named entity marked, coreferent entity assigned the same ID, and the remaining unresolved entities marked with an ID of -1. At this point, there are 1272 unresolved anaphora after coreference resolution, across 49 of the 50 DUC 2006 documents. The remaining document has fully resolved anaphora.

Our code reads through the results to resolve the unresolved entities. As a heuristic for the remaining unresolved anaphora, we resolve each unresolved entity to the previous person, unless there is not a previous person, in which case it is resolved to the previous entity. Said another way, resolution is achieved by marking each unresolved entity with the closest previous entity that is also coreferent with a person entity, even if the closest reference itself is not tagged as a person. In addition to having fully resolved anaphora at this point, gender distinction has also been assigned for most entities.

As before, the information from this stage is incorporated into a the single XML file for the topicset. In this case, multi-word entities are encoded by enclosing one or more word tags with a ⟨metaword⟩ tag pair. This tag has an $entity - id$ attribute which connects it to an entity dictionary that is stored at the bottom of the XML file. Equally important as the XML format, which facilitates unified persistence of all topicset information, is our internal system architecture which provides an efficient in-memory object model for easy experimentation with the topicset. In our final system, anaphora identified in this step are used for term boosting. Moreover, the richness of the representation of this information in our system invites further experimentation and study.

### 3.4 Language Modeling

We construct a language model for each topicset in order to determine the set of term dimensions for the $A$ matrix. The primary function of this step is to prune all terms, including synthetic terms such as entity and n-gram boosting terms, which appear only once across the topicset. Because this analysis is performed as a last step before constructing the $A$ matrix–after our term boosting experiments, we had a measure of protection from generating too many term dimensions, and felt free to experiment liberally with synthetic terms.

### 3.5 Sentence Selection

As discussed in Section 2, sentence selection is based on a $m$-term (rows) by $n$-sentence (columns) matrix which is reduced by SVD. The query sentence or sentences are appended to the corpus as columns of the $A$ matrix prior to the decomposition. This allows us to recover the transformed queries as vectors in the $V$ matrix, and to obtain similarity between corpus sentences and each query by taking the dot product between the appropriate columns in column-normalized $V$.

First we must choose a value for $k$, the number of SVD dimensions to accept. We originally pursued an approach where this parameter would be informed by empirical heuristics based on information-theoretical characteristics of the topicset, but evaluation suggested simply taking $k$ dimensions such that the first $k$ singular values in $\Sigma$ comprise 85% of the total singular value mass. Because the singular values typically describe a power curve, we observed that this figure typically results in $k \approx .6d$. A natural next step was to experiment with the linear factor itself, and this further simplification netted 0.6% and 0.3% gains in ROUGE-2 and ROUGE-SU4. This was in fact the best linear factor that we observed.

Convention for SVD algorithms is to return the singular values in descending order as a vector $w$. After selecting a value for $k$, the matrix $\Sigma_{k\times k}$ is constructed by populating the first $k$ singular values of $w$ to the main diagonal of an otherwise zero-filled $k \times k$ matrix. Accordingly, we include only the first $k$ rows of $V$ in our dot product calculations. These dot product comparisons between normalized columns of $V$ inform our greedy sentence selection heuristic. For each query, the corpus sentences are sorted according to this measure, creating individual sentence picking queues. Until the 250 word limit is reached, each query takes its turn offering its next-closest corpus vector. If its first choice has already been picked by another query, the query does not lose its turn but gets to choose again.

### 3.6 Sentence Ordering

We implemented a simple ordering heuristic for the extracted sentences. When multiple queries are processed in the SVD $A$ matrix, it is logical for our top-level ordering to be determined by which query selected the sentence. As for the ordering of these top-level groups, they are presented in the order in which the query sentences are presented to the system. Within each top level

Challenges in automatic document summarization include homography, which leads to poor precision, and synonymy, which leads to poor recall. The sentence breaker annotates every sentence in our system with a number of statistics, such as entropy and various divergence measures. After cleaning and sentence breaking, the queries and sentences for each topicset are tagged using Stanford's POS tagger. From their web server, we receive back the documents with named entity marked, coreferent entity assigned the same ID, and the remaining unresolved entities marked with an ID of -1. A prerequisite for- and precursor to-shallow abstraction, a synthetic system is thus required to carry, along with the analytically significant content terms, all of the punctuation and typographic information needed to reconstruct the emitted sentence at the end of processing. As mentioned in Section 1, we concluded and documented 140 experiments, and this agile feedback prompted us to reconsider and reconfigure key aspects of our system. Comparing bagged query to our design baseline, where each query was entered in the A matrix separately, gave ROUGE-1 and ROUGE-SU4 improvements of 2.8% and 2.5% respectively. From the baseline system, we experimented with limiting the selection process from the SVD rankings to include only sentences that did not include quotation marks. Manual evaluation primarily includes manual inspection and ad-hoc intuitive evaluation of the generated summaries' relevance to the query. Our system produces ROUGE scores that are competitive with the top DUC 2006 systems and informal human evaluation of our summaries is satisfactory.

Figure 2: 250-word summary of this document produced by our system, given the query "Describe the document summarization system. Include details about major components, system design, and theoretical foundations."

group, we next order each queries' respondent sentences according to the topicset document that it appeared in (again taking this order from the order in which the documents are presented to the system) and, within this ordering, according to the original order of the sentences in the document.

### 3.7 Synthetic Output

Although our system is essentially extractive, our most recent work was to extend our system to support synthetic output. We discuss this work here in order to draw attention to this important system design consideration.

In contrast with a system which emits original extracted sentences as strings (or $regex-$modified strings) from the original source, we define a *synthetic extractive system* as one which reconstitutes its output sentences based on its internal *word* model. A prerequisite for– and precursor to–shallow abstraction, a synthetic system is thus required to carry, along with the analytically significant content terms, all of the punctuation and typographic information needed to reconstruct the emitted sentence at the end of processing.

Our motivation here was to unify the two divergent types of processing that evolved in our post-processing stage, namely, string-based and linguistically-informed modifications. These previously required tedious realignment, but our synthetic system allows us to abandon string-based manipulation of the original source and freely utilize the richness of the word-based programming model during this stage. Final sentences are output from the word model with little or no loss of source fidelity. In some cases, output grammaticality is improved, because typographically invalid source sequences cannot be captured by–or represented in–the internal model.

## 4 Experiments

After our basic system configuration was established, further system modifications and development activities were mainly guided by ROUGE evaluation, with supplemental ad-hoc manual review. As mentioned in Section 1, we concluded and documented 140 experiments, and this agile feedback prompted us to reconsider and reconfigure key aspects of our system. In this section, we highlight some of the interesting experimental results.

### 4.1 Query Bagging

Although our content ordering heuristic depends in part on multiple, independent query sentences selecting their preferred sentences, quantitative evaluation decisively favored query term bagging. Our best results are obtained by taking the set union of the terms from all queries as a single bag, to which the words from the "title" of each topicset are also added. This bag is further subject to entity (Section 4.3) and n-gram boosting (Section 4.2). Comparing bagged query to our design baseline, where each query was entered in the $A$ matrix separately, gave ROUGE-1 and ROUGE-SU4 improvements of 2.8% and 2.5% respectively. This experimental result essentially renders the top-level of our ordering heuristic, which presumes multiple query sentences, vacuous.

| |
|---|
| 1 |
| c (raw counts) |
| sqrt(c) |
| 1 + $\log_{1.5}$(c) |
| 1 + $\log_{2}$(c) |
| 1 + $\log_{3}$(c) |
| 1 + $\log_{10}$(c) |
| 1 + $\log_{50}$(c) |

Table 1: Dampening functions. ROUGE-2 strongly favored sqrt(c)

### 4.2 Query n-Gram Boosting

We observed evaluation gains from boosting, across all sentences in the topicset, n-grams from the query. In this technique, we construct a unique identifier from every possible n-gram in all query sentences independently (n-grams do not span two or more queries). These identifiers are then added once to each query bag or topicset sentence in which that n-gram appears. The component terms of the n-gram are also retained. Any of these identifiers which do not also appear in somewhere in the topicset will be pruned in a later step (Section 3.4), so there is no harm in liberal experimentation here. After experimenting with a number of n-gram ranges, such as 2-5-grams and 4-5-grams, our best result here was obtained by boosting only the 5-grams which appeared in the query.

### 4.3 Entity Boosting

For all sentences and the query bag, we also boost Lingpipe-identified entities by a factor of four (x4). As with n-gram boosting, we generate a unique identifier for the entity, and add it four times to the bag of terms for any query or sentence in which it appears. The original component term or terms of the entity are also retained. Adding the entity multiple times is equivalent to manipulating the term dampening for entities relative to non-entity terms. We evaluated entity boost for factors of x1 through x5. As shown in Table 2, x4 evaluated well.

### 4.4 Term Count Dampening

Because the content vectors in our $A$ matrix represent sentences, instead of full documents, as they would in an IR application, we expected that term count dampening would be a relatively unimportant model parameter. However, we found ROUGE-2 scores to be especially sensitive to the choice of dampening, and this measure strongly preferred dampening term counts by the square root function. Considering this result, it can be explained by the bigram score being most disrupted by changes in the treatment of function words, which are the only words likely to be affected by the choice of dampening function. We evaluated the eight dampening functions shown in Table 1, all of which render the singleton count unchanged. In our final configuration, square root term count dampening improves ROUGE-1, ROUGE-2, and ROUGE-SU4 versus raw counts by 0.53%, 0.26%, and 0.42%, respectively.

| | ROUGE-1 | ROUGE-2 | ROUGE-SU4 |
|---|---|---|---|
| x1 | 0.38448 | 0.08326 | 0.14082 |
| x2 | 0.38602 | 0.08351 | 0.14130 |
| x3 | 0.38689 | 0.08446 | 0.14217 |
| x4 | 0.38737 | 0.08437 | 0.14232 |
| x5 | 0.38703 | 0.08385 | 0.14168 |

Table 2: Entity boost results with bagged query, sqrt dampening, and $k = .6d$

### 4.5 Porter Stemming and Stopwords

Although the seminal work on LSI did not investigate stemming (Deerwester et al., 1990, 15), we found term stemming to be beneficial. We apply Porter stemming (Porter, 1980) to all term inputs to our matrix model. It outperformed our simple baseline stemming by 0.25%, 1.28%, and 0.65% on the respective ROUGE scores. We also experimented with various combinations of stopword application and found that all forms of stopwords were detrimental to ROUGE-2 and ROUGE-SU4. Compared to our baseline system, which enforced a list of 87 stopwords and 79 contractions, removing the stopword lists improved the ROUGE scores by -0.16%, 4.61%, and 1.7%. Having no stopwords was the best configuration we obtained without studying changes to the contents of the two lists.

### 4.6 Word Count Margin

We implement a margin feature which determines how close to the word limit a summary will be considered complete. Naturally, this is a precision-recall trade off, since our sentence picking algorithm skips over sentences that are too long until the word count and margin are satisfied, and this results in selecting sentences with lower cosine scores and possibly detrimental content value. In our experiments with this parameter, we

sought a precision-recall equilibrium, which was unambiguously observed with a margin setting of 12 words.

Related to this feature is a sentence length filter. In this case, aggressive values quickly reduced our ROUGE-2 score, so we elected to allow sentences of any length to be selected.

### 4.7 Sentence Selection Filtering

Many of our experiments studied post-analysis sentence rejection criteria. Post-SVD processing is the preferred stage for categorical rejection, because it allows term co-occurrence information from the rejected sentences to still inform the SVD analysis. From the baseline system, we experimented with limiting the selection process from the SVD rankings to include only sentences that did not include quotation marks. This improved the ROUGE 1 F-score from 0.35246 to 0.35802.

As another experiment, we attempted removing all text outside of the first pair of quotation marks in any sentence that contained at least one pair of quotation marks. This process was applied before the SVD stage in one experiment and after the SVD stage in another experiment. The pre-SVD version supplied the best results, improving the ROUGE 1 F-score from 0.35246 to 0.35203.

### 4.8 Trigram Redundancy

One successful post-processing heuristic was aimed at preventing the selection of nearly identical or highly redundant sentences, relative to those that had already been added to the summary. This heuristic considers the fraction of distinct trigrams that a candidate sentence shares with every picked sentence in turn, and rejects any candidate which exceeds a threshold value. We tested trigram overlap thresholds of 40%, 50%, 60%, 65%, 70%, and 80%, and found 60% to be optimal. As an example of the impact of this parameter, this range of experimentation represents 2.30% change in ROUGE-2, with a peak at 60-65%.

### 4.9 Anaphora

One of our last areas of investigation was anaphora resolution. We planned to rebuild anaphora chains based on Lingpipe entity resolution. We quickly determined, however, that the Lingpipe entity observations were too sparse and imprecise for automatic processing. Simple tests illustrated that grammaticality would be reduced by unsupervised manipulation of the marked entities.

| | Recall | Precision | F-score |
|---|---|---|---|
| ROUGE-1 | 0.39149 | 0.39143 | 0.39142 |
| ROUGE-2 | 0.08590 | 0.08594 | 0.08591 |
| ROUGE-SU4 | 0.14412 | 0.14412 | 0.14411 |

Table 3: DUC 2006 (Development) ROUGE Scores

As part of this investigation, we ran the simple experiment of rejecting all sentences in which the first word is a pronoun. This was our final experiment, and it improved ROUGE-2 by 0.23% and decreased ROUGE-1 and ROUGE-SU4 insignificantly. Since human grammaticality judgement of the summaries was overwhelmingly positive, we retained this heuristic.

## 5 Evaluation

Evaluation includes a combination of manual and automatic evaluation measures. Manual evaluation primarily includes manual inspection and ad-hoc intuitive evaluation of the generated summaries' relevance to the query.

Figure 2 shows the 250-word summary, produced by our system, of this report. The first and last sentences are strong selections, which illustrates that our sentence ordering heuristic, which amounts to presenting the selected sentences in source document order, is effective. Pronouns without antecedents ("From *their* web server...") are a problem, as are embedded discourse connectors ("...system is *thus* required to carry...") and references to out-of-context entities ("Section 1").

Earlier in this project, we used Galvotti-Sebastiani-Simi (GSS) analysis (Galavotti et al., 2000) to evaluate the performance of the clustering of conflated SVD dimensions. In our final system, clustering is not used and we no longer attempt to discern interpretations for particular SVD dimensions, as recommended by most studies (Schütze, 1992).

Automatic evaluation includes the ROUGE metric. Our non-jackknifed average development ROUGE scores over the fifty DUC 2006 topicsets are shown in Table 3. DUC 2007 data was held-out for testing and we ran our system without modification or tuning on these 45 topicsets, obtaining the ROUGE scores shown in Table 4.

Our system was also evaluated in a blind human study which graded 27 of the 45 DUC 2007 topicsets. For each topic, a designated assessor evaluated the grammaticality, referential clarity, and structural coherence by assigning individual scores between 1 ("very poor")

| | Recall | Precision | F-score |
|---|---|---|---|
| ROUGE-1 | 0.40946 | 0.40501 | 0.40706 |
| ROUGE-2 | 0.10042 | 0.09942 | 0.09988 |
| ROUGE-SU4 | 0.15791 | 0.15620 | 0.15699 |

Table 4: DUC 2007 (Held-out) ROUGE Scores

| | mean | median | $\sigma$ | rank |
|---|---|---|---|---|
| Grammaticality | 4.75 | 5 | 0.43 | 1 |
| Referential clarity | 3.71 | 4 | 0.89 | 3 |
| Structure-coherence | 3.25 | 3 | 0.97 | 3 |

Table 5: DUC 2007 Human Evaluation Scores

and 5 ("very good"). Our results, averaged over the 27 randomly selected topics is shown in table 5. Out of nine evaluated systems, our system placed first in grammaticality. As indicated by the standard deviation of 0.43, inter-annotator agreement on our grammaticality score was the best, by a wide margin, in the entire human study. Our referential clarity performance was also strong in human scoring, perhaps owing to the heuristic described in Section 4.9, which excludes sentences that begin with a pronoun. We ranked third in this category, as we did in structural coherence, which was the lowest scoring category in the human evaluation program.

## 6 Summary

Our original query-focused document summarization system demonstrates the elegant efficacy of linear-algebraic singular value decomposition for automatically extracting term co-occurrence patterns from text corpora. Our system produces ROUGE scores that are competitive with the top DUC 2006 systems and 16th out of 31 on held-out DUC 2007 data. Development decisions which favored readability over ROUGE improvement were rewarded with high placement in human evaluation, where our system ranked first in grammaticality and third in both referential clarity and structural coherence. Furthermore, at the end of our ten-week imposed study period, our core approach appears to have potential for continued gains.

## References


Sergey Bochkanov and Vladimir Bystritsky. 2010. ALGLIB numerical analysis and data processing library. http://www.alglib.net/.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Luigi Galavotti, Fabrizio Sebastiani, and Maria Simi. 2000. Experiments on the use of feature selection and negative evidence in automated text categorization. In *ECDL*, pages 59–68.

Gene H. Golub and Charles F. Van Loan. 1996. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

David Graff. 2002. *The AQUAINT Corpus of English News Text*. Linguistic Data Consortium, Philadelphia.

Leonhard Hennig. 2009. Topic-based multi-document summarization with probabilistic latent semantic analysis. In *Recent Advances in Natural Language Processing, RANLP*.

April Kontostathis and William M. Pottenger. 2002. Transitivity and the co-occurrence relation in LSI. Technical Report LU-CSE-02-005, Lehigh University.

Lillian Lee. 2006. Lecture 18: SVD and LSI, CS 630, Human Language Technology, Cornell University, Ithaca NY. http://www.cs.cornell.edu/courses/CS630/2006sp/guides/lec18.hs.corrected.pdf.

Rasmus Elsborg Madsen, Lars Kai Hansen, and Ole Winther. 2004. Singular value decomposition and principal component analysis. February.

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

Barbara Rosario. 2000. Latent semantic indexing: An overview.

Hinrich Schütze. 1992. Dimensions of meaning. In *Proceedings of Supercomputing*.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL 2003*, pages 252–259.