Topic-focused Summarization via Decomposition

Kekoa Riggin, John Greve, Eric Lindberg, Josh Mathias
Department of Linguistics, University of Washington

1 System overview

This paper describes the Ling573 Team 9's D3 system. D3 builds on earlier efforts of our D2 baseline. Our unsupervised learning approach uses pivoted QR matrix decomposition (QRM) weighted by word counts to analyze a document set consisting of 10 articles and generates a purely extractive text summary limited to words for each group of stories, as specified by the Deliverable 3 requirements. We extended a pivoted QR matrix decomposition (QRM) weighted by word counts. Our evaluation uses Rouge scores (Rouge 1-4) against provided gold standard summaries. We compare our D3 results with the D2 baseline and first sentence summarization.

In this paper we will focus on improvements from our previous paper (D2), which include token cleaning and normalization, use of document word counts, and improved sentence parsing.

2 System architecture

Our system is a full sentence extractive summarizer that performs content selection and information ordering on multi-article themed Document Sets drawn from the ACQUAINT and ACQUAINT-2 data sets of news articles and summaries (Graff 2002; Vorhees 2008). Document Sets are defined by a TAC-style Topic Index, and in this paper we focus on a test set consisting of 46 Document Sets presented in GuidedSumm10_test_topics.xml.

Figure 1 shows the Summarizer module which is our main driver. The Summarizer takes a topic index filename and asks Topic Loader to retrieve a collection of Document Sets for that filename. The Summarizer then applies the content selection and information ordering strategies to each Document Set and saves the resulting 100-word max summary in the outputs directory into a single file per summary manner.

Our implementation preserves the arrangement of each Document Set defined in the Topic Index. Each Document Set consists of a collection of Documents (articles) with each Document's text presented as an ordered list of Paragraphs. We use <p> paragraph tags when available to identify paragraphs. When no explicitly tagged paragraphs are found we use the convention that any sentence with a tab character indicates a new paragraph. This is all handled internally by the Topic Loader module which orchestrates the supporting object model. For implementation details see https://github.com/JoshuaMathias/summarizer/.

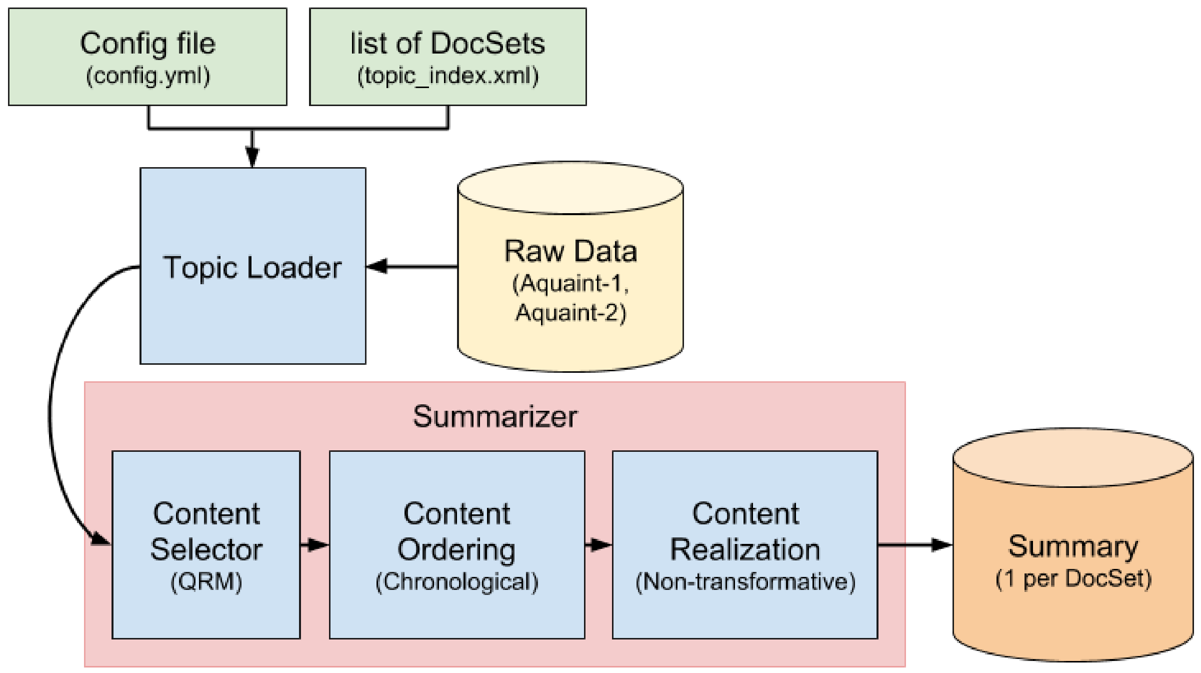

Figure 1: System Architecture, showing data flow

3 Approach

The system is capable of single- and multi-document summarization and can handle batch processing.

For each document group in a batch, we read the data from the specified source, as detailed in a configuration file; select sentences to extract for the summary; compile the sentences into the realized summary; and write the summary to an output file. Scoring is then applied using the reference implementation for this project.

4 Content Selection

Our previous system featured two methods for summarization: the first was a first sentence summarization (FSS) method for content selection and the second is a pivoted QR matrix decomposition (QRM). Our D3 approach builds upon the QRM method of D2.

4.1 Baseline Content Selection Method

Our QRM approach was motivated by Conroy and O'Leary (2001) and uses feature vectors based on the sentences in a document set. Said feature vectors comprise a matrix, where each row represents a sentence and each column represents a feature. In the Conroy paper, these features are meant to capture ideas, which are represented by the words in each sentence.

The sum of each feature vector is weighted, using a weighting function from the paper. This weighted sum is the score for each sentence. The sentence with the highest score is selected. Then, the features from that sentence are removed from the matrix. This reduces redundancy in future sentences that will be selected.

Once all features from the selected sentence have been removed from the remaining feature vectors, the sentence scores are recalculated and this process is repeated until no more sentences will fit in the 100-word summary.

If the best scoring sentence at any recursion of the matrix is too long to add to the summary, the next best scoring sentence that is not too long is used.

4.2 Topic-focused Summarization

Conroy and O'Leary treat each word as equally informative, creating an inherent lack of topic orientation in QRM. The QRM approach admittedly prefers long sentences, which comes as a result of treating each word as an equally valuable idea. To leverage the varying degree of value of words, we first populated the indices of the QR matrix with each word's term frequency, rather than binary values, as outlined by Conroy and O'Leary. We were happy to see a 33% increase in our ROUGE-1 score and similar improvements by the other ROUGE metrics. This, however, was a simple test to determine whether word weighting would have a positive effect on our system performance.

Next, we attempted a more informative measure of word value by populating the indices of the QR matrix with each word's TF*IDF score. Because TF*IDF has produced positive results in other papers (Luhn, 1957), we assumed this weighting metric would produce similar improvements in our performance. We were surprised, however, to see that TF*IDF implementation had a detrimental effect on our ROUGE scores. We found, after seeking to weight the TF*IDF score and the raw term frequency, that the raw term frequency—in conjunction with other features—returned the best results for our system.

Because our system is based on a raw word count weighting system, we found it necessary to remove words that add little value to our summary but that have high frequencies. We utilized the idea of stop words from Hong and Nenkova (2014)—as well as their stop words list—to achieve this purpose. Stop words (such as "the" and "a") appear with high frequency in documents, but contribute little meaningful information. By eliminating these words, we make possible the weighting of valuable words without inflating the sentence scores with valueless words.

4.3 Sentence Tokenization and Content Selection

Our system leverages the NLTK package to tokenize article sentences. We found that this package does a fine job, but some sentences are fragmented in the process. Our solution to this issue was to discard any sentence that did not look like a complete sentence; that is, acceptable sentences begin with an uppercase letter and end with punctuation (period, question mark, or exclamation point). This approach risks excluding valid sentences, but appears to better ensure that only complete sentences are used in summaries. Our data profiling has shown that articles in the test data set only contain 7-bit ASCII letters. Profiling the effect of different boundaries is something we may revisit for D4.

5 Information Ordering

Our system uses chronological ordering, where selected sentences are ordered in the summary by the date their corresponding article was published. We had intended to use both date and time for ordering as explained in the Chronological "Expert" (Bollegala et al., 2004); however, we discovered that the articles in the AQUAINT data did not uniformly provide date/time information. We modified this approach to use the article publication date as the first ordering consideration and the sentence index in its article as the second to serve as a tiebreaker for sentences from articles published on the same date or from the same article.

This ordering decision has no effect on the ROUGE score for our summaries, but we subjectively believe that this ordering improves the flow of content for each summary.

6 Content Realization

Our system currently produces purely extractive summaries. These summaries abide by the specifications provided (100-words, complete sentences), but no transformative content realization is performed.

7 Results

Here we present our results on the "A" document sets of the devtest portion of the ACQUAINT data sets. The two summarization strategies we tried are FSS Scores and QRM Scores. Recall, precision, and F-scores appear correlated in most cases. This indicates we are not suffering anomalies from excellent recall at the expense of precision or vice-versa.

Figure 2 and Table 1 show the results of incremental changes to our QRM system, as described in section 4. "Norm" refers to normalizing words through lowercasing and removing tokens without letters. "Tally" refers to weighting sentence scores using term frequencies within the document set being summarized. "Stop" refers to removing stop words from the system. These each resulted in significant improvement in ROUGE-n scores (compare D2 QRM in Table 2) and combining these changes resulted in the highest scores.

Tf-idf refers to using the tf–idf metric instead of (or in addition to) pure word counts ("tally"). Using tf-idf resulted in lower ROUGE-n scores.

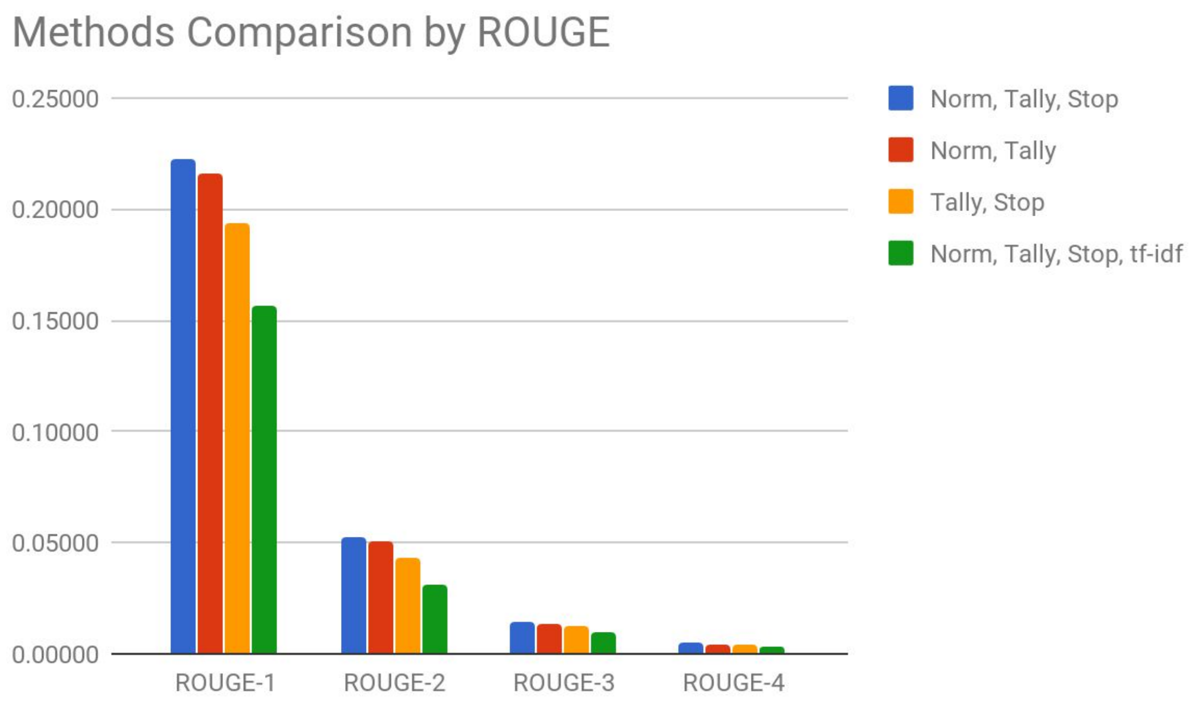

Figure 2. Average ROUGE-n recall values for D3 trials.

	Norm, Tally, Stop	**Norm, Tally**	**Tally, Stop**	**TF*IDF**
ROUGE-1	0.22264	0.21592	0.19384	0.15607
ROUGE-2	0.05216	0.05100	0.04336	0.03121
ROUGE-3	0.01429	0.01373	0.01239	0.00964
ROUGE-4	0.00501	0.00384	0.00396	0.00295

Table 1. Average ROUGE-n recall values for D3 trials.

7.1 D3 vs D2 Rouge Scores

While First Sentence Summarization (FSS) from our D2 results was a high baseline, our final results for D3 QRM were higher than FSS for ROUGE-1 and ROUGE-4, but lower for ROUGE-2 and ROUGE-3. D3 QRM was much higher than D2 QRM for all ROUGE-n.

7.2 Issues and Error Analysis

7.2.a Stopwords

The D3 code has two known bugs related to stopword processing. After our D3 commit we observed that the stopwords weren't being tokenized (via NLTK) the same way.

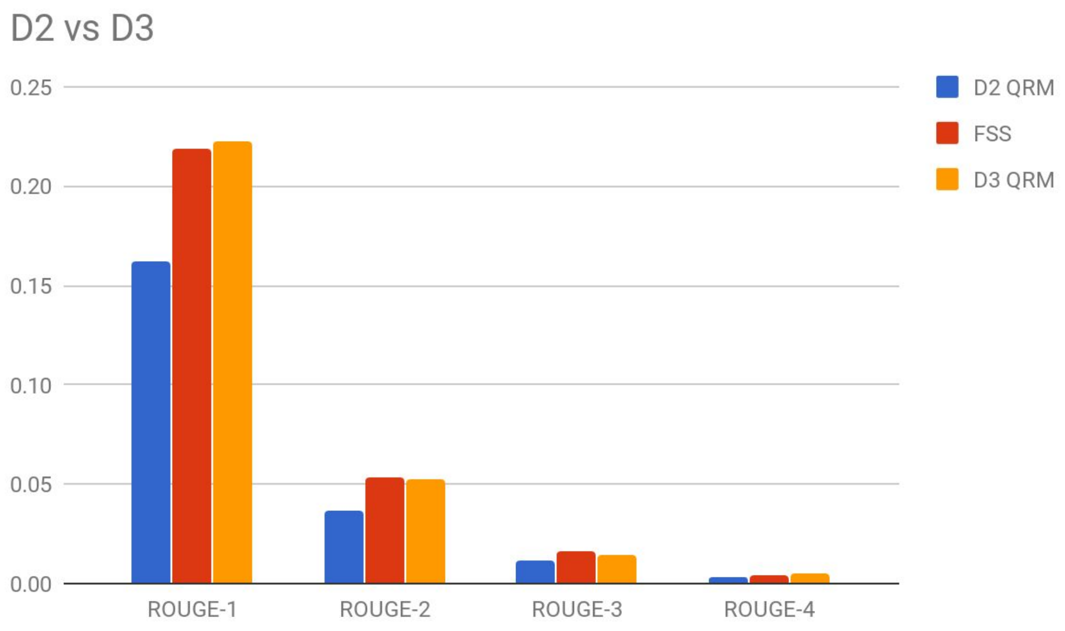

Figure 3. Average ROUGE-n recall values of FSS baseline, D2 QRM, and D3 QRM.

	D2 QRM	FSS	D3 QRM
ROUGE-1	0.16196	0.21842	0.22264
ROUGE-2	0.03677	0.05359	0.05216
ROUGE-3	0.01115	0.01612	0.01429
ROUGE-4	0.00279	0.00455	0.00501

Table 2. Average ROUGE-n recall values of FSS baseline, D2 QRM, and D3 QRM.

Enabling/disabling stopwords wasn’t having the effect we expected. In the process of evaluating this we found a deeper bug that showed we were actually ignoring stopwords. Regrettably the logs showing our initial improvement from adding stopwords were lost and we are unable recreate that data. In order to perform a post-hoc analysis of stopword behavior we added instrumentation to get stop-word usage counts and we verified that while we were loading and filtering stop words we were actually populating the QRMatrix with a different in-memory version of the sentence that retained all of the original words.

Post-hoc analysis on the D3 commit shows the following variations:

Stop Word variations, Rouge-1 recall

	Tokenize	No Tokenize
QRM Yes	0.24838	0.22264
No QRM	-na-	0.24428

QRM Yes/No indicates, respectively, whether the QRMatrix blocked or included stopwords.

Tokenize Yes/No indicates, respectively, whether the stop words were tokenized with NLTK.

Note that the sentences are always processed with NLTK tokenization.

In D3 the effect was No QRM and No Tokenize.

QR.No & Tokenize.Yes is redundant since even if we tokenzied the stop words we'd be ignoring them. Note that since this is post-hoc analysis this code is not in the D3 release.

7.2.b Summary Length

The other issue we noted was our output realization begins to skip highly informative sentences because we are running up against our 100-word limit. When that happens we continue looking for the next most-informative sentence, eventually stopping when the remaining sentences are too long to be added to our summary. This suggests we should look at sentence compression to maximize the benefit from our 100-word limit.

7.2.c Importance of Sentence Position

As the First Sentence Summarization system performs fairly well, we tried giving an exaggeratedly high score (squaring the final score) to sentences that are in the first paragraph and to sentences that are the first sentence of a paragraph. Both of these attempts significantly worsened scores for all ROUGE-n.

7.2.d HMM

We began implementing the HMM implementation as described in Conroy 2001. However, the method described in the paper requires training the transition probabilities on extractive summaries to match summary sentences to positions in the original documents. We found that the training summaries in ACQUAINT were abstractive, hence we were unable to match summary sentences to sentences in raw documents. A possible solution to this is using sentences that are similar to summary sentences using cosine similarity or another similarity metric, and weighting each sentence's contribution to the transition probability based on its similarity to the summary sentence.

8 Discussion

In pondering the cause for our low ROUGE scores when implementing TF*IDF in our content selection method, we came to believe this has to do with the TF*IDF algorithm, as it seeks to measure the information gained by a term rather than that term's salience. Each individual term's frequency is weighted by a value, inversely representing the number of documents that term appeared in--if a term appears in many documents, the weight is reduced. This weighting does well to reduce the effect of prepositions and conjunctions on sentence scores, but it also reduces the weight of words with high document frequencies that we consider to be critical to the topic.

In part due to the nature of newspaper documents, TF*IDF proved to be an underperforming solution to finding topic words in our system. However, by using simple word counts after removing words like prepositions and conjunctions, using a stop words list, we were able to capture the essence of topic words with far greater accuracy than with the TF*IDF weighting system.

The improved results from normalization and removing stop words stresses the importance of preprocessing for the QR matrix method to be effective. Our incremental improvements also suggest that further additions such as stemming and additional sentence features could improve results.

References

Bollegala, D., N. Okazaki, and M. Ishizuka. A preference learning approach to sentence ordering for multi-document summarization. 2004

Conroy, J. M., & O'leary, D. P. (2001, September). Text summarization via hidden markov models. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 406-407). ACM.

Graff, David. The AQUAINT Corpus of English News Text LDC2002T31. Web Download. Philadelphia: Linguistic Data Consortium, 2002.

Hong, K., & Nenkova, A. (2014). Improving the estimation of word importance for news multi-document summarization. In Proceedings of the 14th Conference of the European Chapter of the Association for Computational

Linguistics (pp. 712-721).

Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, *1*(4), 309-317.

Vorhees, Ellen, and David Graff. AQUAINT-2 Information-Retrieval Text Research Collection LDC2008T25. Web Download. Philadelphia: Linguistic Data Consortium, 2008.