Deliverable 2 - Summarization

Joshua Mathias
University of Washington
Seattle, WA
emathias@uw.edu

Eric Lindberg
U. of Washington
Seattle, WA
lindbe2@uw.edu

John Greve
UW Linguistics Dept
Seattle, WA
jgreve@uw.edu

Kekoa Riggin
UW
Seattle, WA
kekoar@uw.edu

1 System Overview

This is the D2 system for Ling573 which establishes an end-to-end baseline summarization system. We read groups of stories and write extractive summaries to the outputs directory as specified by the D2 requirements. There are two methods for summarization: first sentence summarization (FSS) and pivoted QR matrix decomposition method for summarization (QRM). To evaluate we generate Rouge scores (Rouge 1-4) against provided gold standard summaries and discuss our baseline findings.

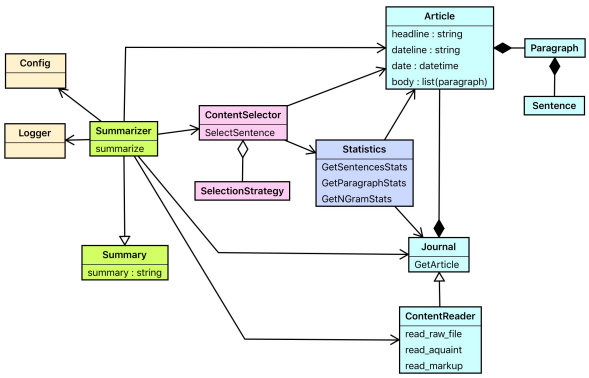
Operational Definitions: The requirements specified well-formed English sentences; for this deliverable we leave this to the level of syntax and punctuation. Semantic composition is outside of our scope.

2 System Architecture

The initial architecture diagram was mapped out as a UML class model (see Figure 1), highlighting the functional areas of the program we wanted to break into different components. In this diagram, the simple arrows mark usage or reference from one class to another. From this, it can be seen that the Summarizer module is the main controller, referring to most of the rest of the package. In the final implementation, the Summarizer logic is contained in summarizer.py. Most of the ContentSelector is still in summarizer.py at this point, with fss.py and qrmatrix.py implementing the SelectionStrategy. The content_provider.py module contains all of the ContentReader, Journal, and Article structures. Statistics are still gathered separately from the main program, for the most part, but will be expanded in the next release.

The unfilled triangular arrows mark generation, showing that the Summarizer creates summaries

Figure 1: UML diagram of summarization system



and that the ContentReader creates the Articles and Document Sets that the Summarizer operates on. Finally, the diamond connectors mark composition or collections, chiefly showing that a Document Set (or Journal) is made up of multiple Articles, which in turn are made up of Paragraphs composed of Sentences. The ContentSelector can refer to multiple selection strategies, though in retrospect this relationship might be one of subclassing rather than collection.

Ultimately, the components are divided into four general areas, Summarization, Data Gathering, Content Selection, and Statistics. These correspond roughly to a standard Model (Data Gathering), Controller (Content Selection and Statistics), View (Summarization) structure.

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 real-

ized summary; and write summary to output files.

Then scoring is applied using the reference implementation for this project.

3.1 Content Selection

Our current system features two methods for summarization. The first is a first sentence summarization and the second is a pivoted QR matrix decomposition.

FSS means that the first sentence of each article in a docset is selected. Given the nature of news articles, this approach produces a fair summary with very little processing.

The pivoted QR matrix decomposition approach, uses feature vectors, based on the sentences in a docset (Conroy and O'leary, 2001). 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.

3.2 Information Ordering

For our baseline FSS method sentences are simply used in the same order that each article is processed.

In our QRM method, sentences that appear earlier in their article appear earlier in our summary. This means that a selected sentence that has a high score may appear lower in the summary while a lower score may appear higher. The goal of this feature is to allow descriptive sentences near the beginning of a news article to appear before sentences that explain or expound later in their own articles.

This ordering decision has no effect on the ROUGE score for our summaries, but we subjectively believe that this ordering affects their readability.

3.3 Content Realization

To generate the summary, we look at each selected sentence sequentially. If the sum of the words in the summary and the words in the current selected sentence is less than 100, the sentence is added to the summary. This realization strategy isnt particularly clever, but it does ensure that the summary adheres to the length requirement and contains only complete sentences. We chose to exclude sentences of less than six words.

Each summary is written to a summary file formatted as described in the Delivery 2 specifications.

4 Results

Here we present our baseline results. The two summarization strategies we tried are FSS Scores and QRM Scores. Since recall and precision appear correlated the F-Scores track with them. This indicates we are not suffering anomalies from excellent recall at the expense of precision or vice-versa. We observed consistently higher FSS Scores over QRM Scores.

Table 1: Average FSS Scores

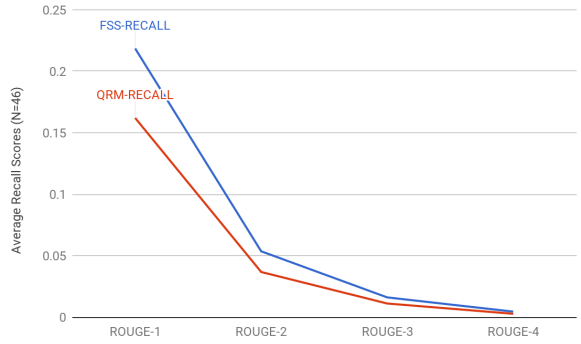
Averages	RECALL	PREC	F-SCORE
ROUGE-1	0.21842	0.23413	0.22481
ROUGE-2	0.05359	0.05758	0.05521
ROUGE-3	0.01612	0.01747	0.01669
ROUGE-4	0.00455	0.00498	0.00473

Table 2: Average QRM Scores

Averages	RECALL	PREC	F-SCORE
ROUGE-1	0.16196	0.20005	0.17844
ROUGE-2	0.03677	0.04512	0.04039
ROUGE-3	0.01115	0.01391	0.01234
ROUGE-4	0.00279	0.00348	0.00309

In Figure 2 you can more easily see the difference between FSS and QRM. It is striking that the ROUGE-1 scores are so much higher than the other rouge scores.

Figure 2: Average ROUGE-n RECALL for QMR and FSS (N=46 doc sets)



5 Discussion

While we were impressed at how well the FSS strategy work we were also surprised at how quickly the rouge scores dropped off. Table 3 shows counts of zero FSS Recall scores for the actual documents. Interestingly everything in ROUGE-1 were able to recall something (e.g. no zero counts). QRM received proportionally similar results and is omitted for brevity.

Table 3: FSS Percentage of Zero Scores

	ROUGE-1	R2	R3	R4
#	0	3	14	33
%	0.00%	6.52%	30.43%	71.74%

To drill into this a little further we are including a heat-map style representation of the observed FSS scores (Figure 3). The reason is that larger N-gram matches become increasingly rare as we go from ROUGE-1 up to ROUGE-4. This will motivate us to look closer into various gold standard N-grams and sample which ones performed well in our baseline and compare them with which ones were absent from our baseline.

This finding shows that we are missing scores, which suggests that the number of empty values will be one of the focuses of our research.

The important thing isn't numeric values but the patterns of zeros (darker red). This pattern will guide the specific doc sets we sample to understand what n-grams matched well in ROUGE scoring for our initial summarization and what matched worse, as in not at all.

We anticipate finding a number of surface features (punctuation, space formatting and so on) that will increase ROUGE scores against the gold

Figure 3: ROUGE heat map for FSS

FSS RECALL			
R-1	R-2	R-3	R-4
0.15638	0.03347	0.01702	0
0.21757	0.0766	0.02597	0.00881
0.18182	0.02643	0	0
0.10744	0.02101	0.00427	0
0.19262	0.04583	0.01695	0
0.16092	0.01556	0	0
0.23556	0.07692	0.00922	0.00469
0.07661	0.0123	0.00417	0
0.20354	0.03604	0.00459	0
0.144	0.06911	0.03719	0.0084
0.08621	0.01754	0	0
0.23171	0.05372	0.01261	0
0.06827	0	0	0
0.12971	0.02128	0.00866	0
0.14717	0.05747	0.02335	0.00791
0.20628	0.03653	0.0093	0
0.15918	0.02905	0.00844	0
0.15481	0.01702	0	0
0.2375	0.05085	0.00862	0.00439
0.13223	0.01261	0.00427	0
0.10593	0.01293	0.00439	0
0.14226	0.04681	0.02165	0.00441
0.09917	0.0042	0	0
0.0625	0.00847	0	0
0.17257	0.0045	0	0
0.09829	0.0087	0.00442	0
0.3373	0.13306	0.02869	0.00833
0.2	0.08051	0.0431	0.02632
0.21667	0.04237	0.00862	0
0.03968	0	0	0
0.10612	0.0249	0.00844	0
0.26718	0.0814	0.03543	0.012
0.15702	0.01681	0.00427	0
0.06584	0	0	0
0.15041	0.03306	0.0042	0
0.09388	0.00415	0	0
0.16204	0.03774	0.00481	0
0.2	0.04382	0.01215	0
0.19184	0.04979	0.02954	0.00858
0.20426	0.06494	0.02203	0.01345
0.08661	0.016	0	0
0.24664	0.08219	0.0186	0
0.28814	0.12069	0.05263	0.01786
0.1308	0.03004	0.01747	0.00444
0.24803	0.024	0	0
0.15748	0.016	0	0

standards. We also expect that the missing longer n-grams will yield direction for where to focus our summarization efforts.

In analyzing our systems results, we realized that we had overlooked the normalization of our feature vectors in terms of punctuation and capitalization. We cannot speak to the significance of the impact this lack of normalization had on our ROUGE scores, but we anticipate improved performance from our system once our QRM method features normalization for punctuation and capitalization.

References

John M. Conroy and Dianne P. O'leary. 2001. Text summarization via hidden markov models. *In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407.