Ling 573 Automatic Text Summarization

Thomas Marsh
University of Washington
ling 573, Spring 2015
`sugarork@uw.edu`

Michael Roylance
University of Washington
ling 573, Spring 2015
`roylance@uw.edu`

Brandon Gahler
University of Washington
ling 573, Spring 2015
`bjg6@uw.edu`

Abstract

This research looks into several different techniques for automatically summarizing documents already grouped by topics. It explores simple sentence similarity techniques such as TF-IDF, sentence distance and sentence length, and then delves into slightly more complicated approaches such as graph similarity, noun phrase clustering, and an extracted "subject predicate object" case frames and entity (from named entity recognition) clustering. Additionally, it examines the application of sentence compression on the overall system, evaluated by readability and ROUGE scores. Finally it attempts to improve the readability of the generated summaries by reordering the sentences with entity transition sequence learning.

1 Introduction

Text summarization has the potential to be a powerful technique for displaying the most relevant information in just a few words. As a form of a question/answering system, it could be able to communicate complex information that would otherwise take a person much longer to read. In this paper, we explore several techniques.

2 System Overview

We used an architecture loosely based on (Jones, 2007), containing modules which perform the following functions:

- Selection (Extraction)
- Redundancy Reduction
- Sentence Compression
- Reordering

2.1 Technologies Used

Python 2.7.9 We coded all development tasks in Python.

Attensity Semantic Annotation Server was used for sentence segmentation, part of speech tagging, named entity recognition, and other extractions.

Natural Language Toolkit was used for sentence segmentation and part of speech tagging.

Stanford Parser was used for sentence trees used with sentence compression.

LibSVM was used for sentence ordering.

TextRazor was used for named entity recognition in the sentence ordering.

2.2 Topic Reader

Topic Reader is a class that reads in an xml file which contains all the topics and their associated document numbers.

2.3 Document Repository

The DocumentRepository class is used in conjunction with the TopicReader class to read the actual documents from the file system. This class looks through an entire XML file and extracts only the document that is referenced. We hand coded the XML reading, as the files we were reading often contained illegal XML characters which caused other XML parsers to break.

Once the Document Repository located and read a document from a file, it then cached it as a pickle file to be used elsewhere.

2.4 Attensity Semantic Annotation Server Parsing and NLTK

The DocModel class is used to store the raw text parsing document annotations from the Attensity

Semantic Annotation Server (ASAS) and Natural Language Toolkit (NLTK). These annotations include sentence segmentation, part of speech tagging, and named entity recognition, among others.

We cached the DocModel for each document from both parsers locally. Parsing can typically be a time-expensive task, so this step saved time as we iterated on our techniques later on.

2.5 Different Sentence Selection Techniques

We initially created several different techniques for scoring the best sentence in a set of documents. These techniques will be discussed further in section 3.

We created a technique summarizer called "Initial Summarizer" to help organize all of the different techniques and select the best sentence. However, we found our extraction clustering technique to be the best all around, so this feature was ultimately turned off.

2.6 KMeans Clustering

We used a form of KMeans clustering to help separate the sentences into different clusters for redundancy reduction. A cluster centroid was defined as a distinct union of all its points (with counts) while removing any common points from other clusters.

This KMeans algorithm was allowed to execute a maximum of one hundred iterations, however there was usually no movement after the forth or fifth iteration.

We explored using different points, such as bigrams, trigrams, noun bigrams, and noun trigrams. Ultimately bigrams proved to generate the highest ROUGE score.

After the all the sentences were scored from the previous techniques, KMeans clustering was used to group similar sentences together. We took the top scoring sentence from each cluster until we reached the maximum word count.

2.7 Sentence Compression

We attempted to train a sequence labeler to identify words to remove, using features derived from the words themselves and from phrase structures and dependency structures produced by the Stanford Parser (Socher et al., 2013).

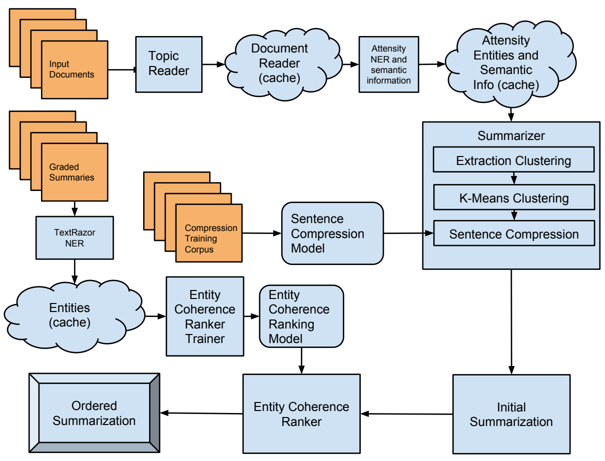
2.8 Sentence Ordering

We implemented an entity-based coherence strategy for sentence ordering, based on Barzilay and Lapata's approach (Barzilay and Lapata, 2008)

2.9 Evaluation

We used ROUGE (Lin, 2004) and pyrouge (Heinzerling and Johannsen, 2014) to evaluate our scores from the realized summaries. For a baseline, we used the standard summarization, inital summarization, and random summarization from the MEAD evaluation package (Radev et al., 2006).

Figure 1: System Architecture



3 Approach

3.1 Content Selection

We implemented a variety of methods initially to help with content selection. Each of these separately implemented content selection systems produces a score for each sentence from 0.0 to 1.0. Empirically determined system weights are used in summing these scores to produce aggregate scores. The top scoring sentences are used to create the summary itself. For a baseline, we used MEAD's main, initial, and random implementations. (Radev et al., 2006)

3.1.1 Trivial Systems

Two of our content selection systems are trivial systems. One system scores sentences based on their length relative to the longest sentence in the document cluster, favoring longer sentences. The second scores sentences based on their position in their document, giving the first sentence in each document a score of 1.0 and the last sentence in each document a score of 0.0.

3.1.2 TF-IDF Scoring

For our first non-trivial content selection system, we calculated the average TF-IDF score of every non-stop word in the sentence. Document-frequency scores were calculated using the Reuters-21578, Distribution 1.0 Corpus of news articles. Sentence scores are scaled such that the sentence with the highest average TF-IDF was given a score of 1.0.

3.1.3 Simple Graph Based Scoring

Next, we implemented a simple graph based metric in which a dense undirected graph of sentences is constructed with edge weights set to the cosine similarity of the sentences. The most connected sentence is iteratively selected, and the weights of the edges of the previously selected sentences are set to be negative to discourage redundancy. The first pulled sentence is given a score of 1.0, with scores incrementally decreasing to 0.0 for the sentence pulled last.

3.1.4 Noun Phrase Clustering

We then implemented a noun phrase clustering metric where we used a custom co-reference resolution system to resolve pronouns to their most likely antecedent, and then compared each sentence's noun phrases to each other sentence. NP-clustering selection was ordered by the number of matches each sentence had to every other sentence. The co-reference system was based on ideas from (Cardie and Wagstaff, 1999).

3.1.5 Extraction Clustering

Finally, we decided to look at external sources for sentence similarity. We used the Attensity Semantic Annotation Server to produce several extractions per document.

The first extraction we used (from ASAS) was entities, which contain semantic information for identified noun phrases (person, location, company, etc). For sentence similarity identification, we compared not only the noun phrase spelling, but also the semantic tag associated with it.

The second extraction used was called "triples". The parser identifies several subject, predicate and object tuples in each sentence. For our similarity comparison, we look at just the subject and object.

The final extraction we used from the parser was called "facts". Facts have two properties associated with them, "element" and "mode". These two properties combine with other extractions (not used for this project, currently) similar to triples for generating case frames (Riloff and Schmelzenbach, 1998).

3.2 Sentence Ordering

For sentence ordering, we implemented an entity-based coherance ordering strategy based on (Barzilay and Lapata, 2008).

Specifically, we used the TextRazor named entity recognizer package (Crayston, 2015) to extract entities to use in transition grids. Although we do have a co-reference resolution system in place, we did not have time to add it to our experiments.

The entity coherence system assigns each entity a type of *Subject*, *Object* or *Oblique* for each sentence. It then extracts the transitions of the entities from sentence to sentence, and builds feature vectors based on the probability of each transition type in the overall document. The main idea here is that by memorizing what the transitions look like in documents that are "good" vs. in those that are "bad" we can try to look at summaries and judge how closely they resemble the ideal pattern.

For Training, we used a set of summarizations which had previously been judged for readability, and used the readability score as the training rank. We do the training ranking on the feature vectors with SVMLight (Jaochims, 2002).

After our content selection and realization modules have selected what they deem to be the optimal sentences, we apply the reordering model to these sentences. To do this, we find all permutations of each sentence, producing feature vectors for each permutation. We send those feature vectors to the ranker, and obtain the optimal ordering by selecting the top-ranked permutation.

For our configuration parameters to the ordering module, we used 2 for the number of transition types, and 1 for the minimum frequency of entities to count as "salient".

3.3 Content Realization

We attempted approaching sentence compression as a sequence labeling problem. Our algorithm used a single chain CRF for labeling words to keep as "B" begin or "I" inner and words to remove as "O" outer.

For training the model we used the Written News Compression Corpus (Clarke and Lapata, 2008). The corpus consists of 82 newswire articles hand annotated with compressed versions of each sentence. Of the 1,433 sentences of the corpus we used the first 90% for training the classifier and the remaining 10% for testing.

We chose to derive large numbers of features and perform automatic feature selection using the chi squared test. The features we chose can be categorised into word level, phrase structure, and dependency structure features. For each word to be classified as “B”, “I”, or “O” the classifier is supplied with the features mentioned below for the current word plus the those for the two previous words, as well as the labels of the two previous words.

The word level features we implemented include the word stem as derived by the NLTK’s implementation of the Snowball stemmer, and the suffix if recoverable. Two features where used to denote the capitalization of the word, whether all the letters where capitalized or only the first letter. Three word classes we considered important to compression were stopwords, punctuation, and negation (“not” and “n’t”). Membership in each of these word classes was denoted by separate features. Finally, the tf-idf score of each word was measured and compared to the highest score in the sentence. If the current word’s score is greater than or equal to some parentage of the highest score, the word was given an appropriate feature. We chose to award these features for thresholds at 20%, 40%, 60%, 80%, and 100%.

To extract phrase structure and dependency structure features, we parsed our sentences with the Stanford Parser (Socher et al., 2013). Phrase structure features include the depth of each word in the phrase structure tree. For each node dominating a given word in the phrase structure tree a feature derived from the node label was awarded, regardless of how much higher it is. Additional features denote the word’s parent and grandparent nodes. Finally a feature marked the words position within its parent node as the number of words from the left.

As with the phrase structure trees, the depth of each word in the dependency structure tree served as a feature, with additional features for going over depth thresholds of 3 and 5. Features were derived from the types of dependency relations each word participated in, with separate features for being the mother or the daughter of the relation.

We tested multiple classifiers, including Nieve Bayes, decision trees, Maximum Entropy models, and linear SVM, along with different chi squared thresholds for feature selection, and our best results were 82.7% accuracy from linear SVM trained on 3% of the total features appearing in the training set. The most valuable features for this task appeared to be the word level features, with our classifier reaching 79.4% accuracy with these alone.

Figure 2: Examples of Compressed Sentences

1. The Three Gorges Dam in central China’s Hubei Province Thursday has opened its floodgates to ease the severe water shortages along the Yangtze River.

2. A co-defendant in the O.J. Simpson armed robbery case told a judge Monday he would plead guilty to a felony and testify against Simpson and four others in the hotel room theft of sports collectibles from two memorabilia dealers .

3. In some ways , the Internet security business has been like the dot-com business .

4. If it were fully loaded , the ship’s deck would be lower to the water , making it easier for pirates to climb aboard with grappling equipment and ladders , as they do in most hijackings .

5. The Internet security industry is in the midst of a major consolidation , with large companies gobbling up little ones as they try to expand their menu of offerings to consumers and businesses .

The readability of the sentences compressed by our system was unfortunately poor. We had hoped that the compressor would remove individual low importance elements such as adjectives, prepositional phrases, adverbs and time expressions, as in sentence 1 in Figure 2. More often however it removed longer, sometimes contentful sections, particularly from the end of sentences. Sentence 2 shows an example of a large prepositional phrase describing a topic of discussion being removed, and sentences 4 and 5 show a seemingly arbitrary early end to the sentence. Even when smaller sections are marked for removal the sentence is sometimes left ungrammatical by the removal, such as in the beginning of sentences 3 and 4.

Unfortunately, the inclusion of this content realization system produced little change in our ROUGE scores. As it failed to improve either human readability or automatic content evaluation metrics, our final system ran with the compressor disabled, and performed content realization by simply outputting the most highly-ranked sentences.

4 Results

Our best results were obtained by using the extraction clustering technique by itself. The results are shown below in Table 4. The addition of new extraction types of entities and case frames have shown to improve in identifying sentences which are most similar, and thus representative of the most important sentences of the document.

We are encouraged by our results from extraction clustering. The ROUGE scores are well above the baselines, indicating that the techniques we employed show promise.

4.1 ROUGE Results

Table 1: NP Clustering

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.23391	0.28553	0.25522
ROUGE2	0.05736	0.07053	0.06272
ROUGE3	0.01612	0.01969	0.01758
ROUGE4	0.00533	0.00657	0.00584

Table 2: Simple Graph

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.19379	0.25550	0.21845
ROUGE2	0.04473	0.05859	0.05033
ROUGE3	0.01170	0.01505	0.01305
ROUGE4	0.00362	0.00453	0.00400

Table 3: tf-idf

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.15341	0.20846	0.17522
ROUGE2	0.03014	0.04037	0.03426
ROUGE3	0.00746	0.01038	0.00863
ROUGE4	0.00242	0.00329	0.00278

Table 4: Final Extraction Clustering

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.23577	0.29921	0.26186
ROUGE2	0.07144	0.09095	0.07949
ROUGE3	0.02821	0.03621	0.03151
ROUGE4	0.01271	0.01624	0.01419

Table 5: Baseline 1 - MEAD Standard

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.22031	0.22465	0.22160
ROUGE2	0.05288	0.05356	0.05301
ROUGE3	0.01412	0.01394	0.01399
ROUGE4	0.00474	0.00454	0.00462

Table 6: Baseline 2 - MEAD Initial

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.22145	0.21050	0.21524
ROUGE2	0.04570	0.04330	0.04435
ROUGE3	0.01093	0.01011	0.01048
ROUGE4	0.00230	0.00210	0.00219

Table 7: Baseline 3 - MEAD Random

Rouge Technique	Recall	Precision	F-Score
ROUGE1	0.15452	0.17465	0.16350
ROUGE2	0.02630	0.02977	0.02784
ROUGE3	0.00671	0.00763	0.00711
ROUGE4	0.00202	0.00225	0.00212

Compared with the best MEAD scores, our results look promising:

Table 8: Improvements from MEAD Standard to Extraction Clustering

Rouge	Recall	Precision	F-Score
R-1	+18.65%	+22.11%	+20.48%
R-2	+29.56%	+33.72%	+31.75%
R-3	+60.62%	+68.01%	+64.26%
R-4	+100.42%	+114.98%	+107.79%

4.2 Readability Results

Our readability scores were generated by sending a random sampling of our summaries to our peers (other linguistics graduate students) for evaluation of readability. The metric ranged from 1 to 5, with the mean being at 3.03. Our average overall score was 2.875, which put us slightly below the mean.

Table 9: Readability Scores by topic

Topic	Our System	Mean	Stddev
D1110B	3.3	3.03	0.706
D1115C	3.3	3.367	0.862
D1119D	2.3	2.53	0.763
D1122D	3.3	2.83	0.582
D1127E	1.7	2.73	0.712
D1130F	2.7	3.2	1.086
D1133F	2.0	3.1	0.746
D1143H	4.3	3.4	0.854

5 Discussion

Overall our readability scores were disappointing due to being below the mean. However, at an individual topic level we did have some that performed well. For example, here is the summarization for D1143H (which scored a 4.3 with a class mean of 3.4):

> A co-defendant in the O.J. Simpson armed robbery case told a judge Monday he would plead guilty to a felony and testify against Simpson and four others in the hotel room theft of sports collectibles from two memorabilia dealers.
>
> A lawyer for one of O.J. Simpson's co-defendants in an armed robbery case said Friday that his client will plead guilty Monday to a reduced charge and testify against Simpson and four others.
>
> Simpson and the men have been charged with seven felony offences: conspiracy to commit kidnapping; conspiracy to commit robbery; first degree kidnapping with use of a deadly weapon; burglary with a deadly weapon; robbery with a deadly weapon; assault with a deadly weapon and coercion with a deadly weapon.

In this case, the systems came together to produce a decently coherent summarization.

Here is a poorly graded summarization for topic D1127E (which scored a 1.7 with a class mean of 2.73):

> The study, published in a recent issue of the American Journal of Geriatric Psychiatry, said despite the high rate of sleep complaints among the patients, a sleep complaint was only reported by the doctor in the patient's chart 19 percent of the time, even when the patient indicated sleep problems in all five sleep questions on the survey.
>
> Mah, a researcher at the Stanford Sleep Disorders Clinic and Research Laboratory, says athletes often aren't counseled on the value of adequate sleep, adding that college students are no strangers to sleep deprivation, sometimes maxing out at five or six hours of sleep a night.
>
> Further, smoking was related to sleeping less than eight hours, bedtime later than midnight, nightmares, difficulty initiating sleep, difficulty maintaining sleep and hypnotic medication use.

In this case, ”The study” was not correctly introduced to the user, which may have contributed to its poor grading. Also, all three sentences are not connected coherently.

Though our readability scores are comparatively low, we are encouraged by our increased ROUGE results. We feel that we are on the right track with our extraction techniques, and we understand what we need to do to increase our readability scores. We feel that our extraction techniques, if given enough time for development, could yield results of very high quality.

In particular, our ROUGE-4 scores seem to be greatly increased from the baseline. We are not sure why these increased so much more than the ROUGE-2 scores, but we are interested to investigate this in the future.

5.1 Error Analysis and Observations

While our scores compare quite favorably to the baselines, there are several improvements which we would like to make, as well as several issues which perplexed us during our experiments.

1. Our sentence selection algorithm appears to be selecting very long sentences most of the time. We believe this is because our strategy for normalizing for the sentence length is insufficiently compensating for the number of

relevant terms it finds in these sentences. Another strategy here would likely improve our results.

2. Our sentence ordering algorithm was not tested with Frequency parameter=2. This may yield better readability results

3. Though our voting/technique-combination strategy did not yield fruit, we are unsure of the reason. We would like to devote more time to investigating and experimenting with this strategy, perhaps in combination with machine learning algorithms to generate the optimal weight sets.

4. Strangely, the MEAD results cited in the TAC results posted in class do not agree with our results obtained from actually running the MEAD summarizer against our data set. We are unsure why this discrepancy exists.

5. All the results cited above are from running against the devtest data set, however, when running our summarizer against the evaltest data, our results were significantly lower. Others participating in the summarization excercize appear to have had higher results with the evaltest data than with the devtest data. We do not understand this difference at this time, and would like to investigate it.

6. Our sentence compression results did not appear to improve readability or ROUGE scores. When we got this working, we had very little time to evaluate and debug it, so it is likely that we simply missed something. We would very much like to have some more time to work on this, potentially adding more features for the classifier.

7. Our final system simply truncated the last sentence at 100 words. While this was useful for ROUGE comparisons, it left our readers hanging, and contributed to our poor readability scores. To fix this, we would like to enable our sentence compression (once it is working: see previous item) and use the shorter sentences in an algorithm that looks ahead and adds shorter sentences if it is about to go over the word limit with the current sentence.

6 Conclusion

We have seen improved ROUGE scores across the board with extraction based clustering, but our readability scores leave much to be desired. We feel that making many of the changes outlined in the Error Analysis section above will increase both our ROUGE scores as well as our readability scores.

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

C. Cardie and K. Wagstaff. 1999. Noun phrase coreference as clustering. *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1001:82–89.

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, pages 399–429.

Toby Crayston. 2015. Textrazor (Version 1.0) [Software]. Available from `https://www.textrazor.com/`

Benjamin Heinzerling and Anders Johannsen. 2014. pyrouge (Version 0.1.2) [Software]. Available from `https://github.com/noutenki/pyrouge`

Thorsten Jaochims. 2002. Svmlight (Version 6.02) [Software]. Available from `http://svmlight.joachims.org/`

Karen Spärck Jones. 2007. Automatic summarising: The state of the art. *Information Processing Management*, 43(6):1449 – 1481. Text Summarization.

Chin-Yew Lin. 2004. Rouge (Version 1.5.5) [Software]. Available from `http://www.berouge.com/Pages/default.aspx`

Dragomir R Radev, Sasha Blair-Goldensohn, and Zhu Zhang. 2006. Mead (Version 3.12) [Software]. Available from `http://www.summarization.com/mead/`

Ellen Riloff and Mark Schmelzenbach. 1998. An empirical approach to conceptual case frame acquisition.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*. Citeseer.