Ling 573 Project: Multi-Document Summarization

Emma Bateman
ebateman@uw.edu

John Dodson
jrdodson@uw.edu

Charlie Guo
qcg@uw.edu

Abstract

We describe an unsupervised multi-document summarization system. Given a corpus, salient sentences are selected through the TextRank algorithm and then re-ordered using a local coherence algorithm for modeling entity distributions across the text. The ordered result set is then compressed using either rule-based or machine learning techniques. The system is divided into three components for ranking salient sentences, logical ordering, and result compression. We evaluate our system on the AQUAINT, AQUAINT-2, and Gigaword corpora.

1 Introduction

Multi-document summarization systems seek to aggregate collections of text documents and condense their content to provide cohesive summaries. Traditional approaches to summarization fall into two distinct categories: abstractive and extractive. Abstractive techniques are categorized based on their ability to generate novel summary content, in contrast with extractive approaches which select significant content directly from the source documents. In this work, we present an unsupervised system for extractive multi-document summarization over a benchmark dataset. The system is centered around the TextRank algorithm, which is a graph-based technique to identify and rank the most salient sentences in the input documents. We evaluate our system on the AQUAINT, AQUAINT-2, and Gigaword corpora, with average ROUGE-1 and ROUGE-2 F-scores of 0.28385 and 0.07365, respectively.

This report is structured as follows: Section 2 provides an overview of our system and its components, including a brief description of the datasets used to train and evaluate the experiment; section 3 details our approach to generating summaries using TextRank, entity-grid representations, and compression techniques; section 4 presents result metrics; section 5 is a discussion and error analysis; and section 6 concludes.

2 System Overview

The system is divided into three primary components: a graph-based content selector, an ordering algorithm, and a compression algorithm. The content selector is responsible for ingesting a schema file which details one or more document clusters in the AQUAINT and Gigaword datasets. The selector identifies which documents belong together in a cluster and then ingests those documents as plain text. The system extracts feature vectors from each sentence by computing word frequencies. The feature vectors are given as input to the TextRank algorithm to sort vectors by salience. Sentences are selected and compressed in order of salience until the summary surpasses the word limit (100 words) at which point the last sentence is discarded. Sentences are ordered by one of of two methods: they are either fed through an entity-grid based algorithm, or ranked by the shortest path between their feature vectors. The logically ordered sentences are further compressed using either a rule-based algorithm or a MaxEnt model trained to compress text.

2.1 Architecture

Figure 1 provides an illustration of the existing system architecture.

2.2 Datasets

We use the AQUAINT (Advanced Question-Answering for Intelligence) and AQUAINT-2 corpora to provide input to our system. Both datasets are composed of English newswire texts. For the multi-document summarization task, we leverage a corresponding schema file which defines one or

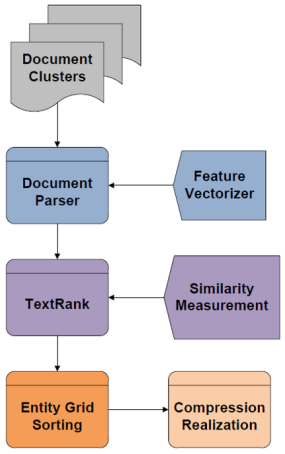

Figure 1: Multi-document summarization system architecture.

more document-topic clusters. For evaluation, we use a subset of the Gigaword dataset.

3 Approach

This section details the main components of the working system.

3.1 Content Selection

The ingest processor for the system consumes one or more documents associated with a cluster identifier. These SGML documents are converted to plain text and represented as a single monolithic text string. After ingest is complete, the system will maintain one text representation per document cluster.

The system uses BeautifulSoup to perform most of its SGML and XML parsing. For a given input file, the ingest processor identifies clusters and their associated documents by finding document IDs which correspond to AQUAINT SGML files. The system then loads the corresponding SGML files and performs traversal using BeautifulSoup to find the appropriate blocks of text.

3.1.1 Feature Vectors

The feature extraction component takes the plain text representation and tokenizes it into individual sentences using Python's Natural Language Toolkit (NLTK). NLTK is used to remove common stopwords from each sentence. Feature vectors are generated in two steps: firstly, the component identifies a comprehensive vocabulary for the cluster; and secondly, a frequency vector is created based on the token frequencies in the given sentence.

The output of this component is an N x V matrix, where N is the total number of sentences in the cluster and V is the size of the vocabulary.

3.1.2 TextRank

Our system uses TextRank to compute saliency scores for each sentence in the cluster and subsequently rank them according to significance. The TextRank algorithm constructs a graphical representation of textual features, where a given textual feature is represented as a vertex in the graph. The algorithm assigns a significance score to each vertex based on inbound edge weighting.

The authors of the TextRank algorithm formally define a graph as a tuple $G = (V, E)$, where $V = \{v_1, v_2, ...v_n\}$ is the set of vertices in the graph and E is a set of edges and a subset of V x V. TextRank is inspired by PageRank and computes the score for some vertex V_i similarly to the original algorithm:

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

$In(V_i)$ represents all inbound connections to vertex V_i, and $Out(V_j)$ represents outbound connections from vertex V_j. The authors use d as a damping factor set between 0 and 1, which effectively biases the calculation. The original PageRank algorithm defaults this factor to 0.85, and in our system we maintain this defaulted value.

In applying TextRank to sentence processing, we compute a pairwise similarity matrix of size N x N, where N is the total number of sentences in the cluster. The similarity between N_i and N_j becomes the edge weight which connects those two vertices in the graph. We use cosine similarity as our distance measurement.

After the TextRank algorithm converges, the algorithm returns the top K sentences as the cluster summary. Each summary contains 100 words maximum. The system maintains a reference to the original sentences, and uses the original text as output.

3.1.3 Redundancy

In our earlier system, we encountered significant amounts of redundancy, including repeated sentences. In one case, a sentence appeared in a summary three times in a row.

> "The nation deserves and I will select a Supreme Court justice that Americans can be proud of," Bush said.
> "The nation deserves and I will select a

Supreme Court Justice that Americans can be proud of," Bush said.
"The nation deserves and I will select a Supreme Court justice that Americans can be proud of," Bush said.
Sandra Day O'Connor, the first woman ever appointed to the US Supreme Court, said Friday that she is retiring, giving US president George W. Bush his first opportunity to appoint a justice.

We fixed this issue by modifying the TextRank algorithm. At each iteration, until enough sentences have been selected to complete the summary, the most salient sentence is extracted. Then, the corresponding feature vector is subtracted from the rest of the matrix, disadvantaging sentences with a high level to similarity to the selected sentence. Finally, the similarity algorithm is rerun before the next sentence is selected.

3.2 Information Ordering

The system uses an entity-based modeling approach to improve summary coherence and provide a more logical ordering of the sentences in each summary. Our approach is inspired by the entity grid mechanism introduced in Barzilay and Lapata 2008, wherein the authors describe a local coherence algorithm centered around entity transitions. Our algorithm models the distribution of entities across the top outputs of TextRank, which means the input data is fundamentally small. We make the assumption that the input sentences naturally contain local coherence after applying TextRank.

3.2.1 Extracting Grammatical Roles

The authors of the original algorithm exploit several factors in their entity representations, namely, coreference resolution, grammatical roles, and saliency. Our system only uses grammatical function in determining entity types. We use the Stanford CoreNLP toolkit to generate dependency parses over each input sentence. Subject and object relations are extracted from the parse tree and used to model entity transitions. We provide weights for entities with subject, object, and alternate relation categories.

3.2.2 Entity Grid Representation

Following the conventions established in Barzilay and Lapata 2008, we use the grammatical roles described in Section 3.2.1 to denote types of discourse entities. After generating dependency parses and compiling the set of unique entities in the input corpus, we compile a table representation of size N x M, where N is the total number of sentences in the input corpus and M is the total number of unique entities extracted from that corpus. In our system N is reasonably small.

The entity grid is populated with weight values corresponding to the grammatical function of an entity in a given sentence, where a table entry $E_{i,j}$ is weighted depending on the grammatical role of entity j in sentence i.

3.2.3 Shortest Path

The shortest path ordering algorithm orders the sentences by solving the traveling salesman problem for the feature vectors of the selected sentences.

The distances between vectors are calculated using cosine similarity.

The sentences are output in the order of the shortest path, starting with the most salient of the two endpoints.

3.3 Compression

This section discusses the sentence compression modules for proper content realization.

3.3.1 Rule-based compression

A rule-based compression is implemented in our system which shortens each sentence by parsing each sentence and then removing less important syntactic component such as PP and DT. Here is an example of a shortened sentence in our system output:

Senior Palestinian official Yasser Abed Rabbo denied on Tuesday reports saying that Palestinian leader Yasser Arafat has died in a French hospital.

The system leverages a named entity recognition capability to identify and simplify individual person mentions when a name is referenced multiple times. For example:

Senior Palestinian official denied reports saying that Palestinian leader **Yasser Arafat** has died.

As **Arafat** had struggled for life, there has been wild guess as to ...

French foreign minister told LCI television that **Arafat** was alive but that ...

Palestinian leader Yasser **Arafat** would be buried at his headquarters in the West Bank town of Ramallah.

We did not exhaust all possible syntactic components. This is discussed further in subsequent sections.

3.3.2 Model-based compression

Our model-based compression algorithm is based on the work of Wang et al (2013).

We began by training a MaxEnt model on hand-annotated compression data from Clarke and Lapata (2008). The training data was downloaded from `https://www.jamesclarke.net/research/resources`. We used 1000 sentences from the Broadcast News Compression Corpus.

The feature vectors used for training were many of the same features used in Wang et al, including:

- all leaves fall within first 3 tokens?
- all leaves fall within last 3 tokens?
- subsumes first 3 tokens?
- subsumes last 3 tokens?
- contains words longer than 10 letters?
- subsumes only one leaf?
- subsumes entire sentence?
- contains capitalization?
- contains all caps word?
- contains negation?
- contains stopwords?
- falls within parenthetical?
- contains lead adverbial?
- contains relative clause?
- contains lead prepositional clause?
- depth of node
- node label
- left sibling node label
- right sibling node label
- left sibling node label of parent
- right sibling node label of parent

We formulated the problem space for each sentence as the possible sequences of "keep" and "delete" labels for each node of the syntax tree. We then performed a heuristic search of the problem space with the trained MaxEnt model as the scoring function. At each node, our algorithm performs search recursively on each child node, then returns the 5 highest scoring sequences found using those results. Of the five sequences discovered by running search on the head node, the highest scoring one is used to compress the sentence.

3.4 Baseline System

The baseline for this experiment is a system with each aforementioned component except the compression algorithms. In theory, this baseline still generates ordered salient results but disregards human readability. The baseline is evaluated on the development corpus comprised of AQUAINT and AQUAINT-2 data.

4 Results

We present result metrics for both ROUGE-1 and ROUGE-2 over the development and evaluation corpora. In addition, we provide human evaluations for six generated summaries.

4.1 ROUGE scores

Table 1 and **Table 2** present ROUGE-1 and ROUGE-2 metrics for development and evaluation data. For the development set, the best reported ROUGE-1 F1 score is given by the baseline algorithm. The reduced scores in the updated system are likely due to the compression component, which focuses on readability. Similarly, for the ROUGE-2 scores, the baseline algorithm gives the best recall, precision, and F1 metrics.

4.2 Human Evaluation

Readability is an important aspect of document summarization that is not represented by ROUGE scores. **Table 3** shows human eval scores for six of our summaries based on ratings given by 4 annotators on a scale from 1 to 5.

5 Discussion & Error Analysis

By visually checking results from both development and evaluation data, we did identify several of issues related to compression and content realization. First, there is loss of grammaticality. We see many cases of incomplete removal of PP,

which is very likely caused by a deficiency in sentence rendering of a truncated syntax tree. The following is an example:

> The death toll **from the** that struck the coastline **near the** of Aitape in the province Friday night, was announced 64 and people were missing.

Secondly, pronouns occasionally appear without an antecedent. This is due to the lack of support for anaphora resolution.

> **He** said that the company did not ...
>
> **He** estimated that ...

Beyond the rule-based module, handcrafting additional features for the MaxEnt implementation is another potential area of improvement. Given that the model-based compression system outperformed every other system configuration for the evaluation data, further feature engineering for the model training data could potentially yield improved metrics.

Other minor issues such as missing and mismatching punctuations were also identified. All these issues affected the readability of the summaries.

6 Conclusion

We have described an unsupervised system for generating single summaries over multiple documents. The system generates feature vectors per sentence by computing word frequencies, and then provides those frequency vectors as input to the TextRank algorithm. The algorithm is able to identify significant lines of text in a single pass. We take the top K sentences as the corresponding summary, up to 100 words max per summary. The system leverages an entity grid algorithm for intelligent sentence ordering, and cleans the output text through one of two compression implementations.

References

Rada Mihalcea and Paul Tarau, *TextRank: Bringing Order into Texts*, Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004.

Regina Barzilay and Mirella Lapata, *Modeling Local Coherence: An Entity-Based Approach* Computational Linguistics, 2008

Lu Wang et al, *A Sentence Compression Based Framework to Query-Focused Multi-Document Summarization*, Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 1384-1394, 2013

James Clarke and Mirella Lapata, *Global Inferece for Sentence Compression: An Integer Linear Programming Approach*, Journal of Artificial Intelligence Research vol 31, pages 399-429, 2008

Development set ROUGE-1 and ROUGE-2 scores						
Algorithm	**ROUGE-1 Recall**	**ROUGE-1 Precision**	**ROUGE-1 F1**	**ROUGE-2 Recall**	**ROUGE-2 Precision**	**ROUGE-2 F1**
Baseline algorithm	0.21358	**0.27155**	**0.23732**	**0.05335**	**0.06814**	**0.05937**
Entity grid, rule-based	0.21643	0.23487	0.22322	0.05036	0.05311	0.05133
Shortest path, rule-based	**0.21865**	0.23280	0.22400	0.05054	0.05282	0.05136
Entity grid, model-based	0.20716	0.27003	0.23264	0.04610	0.05970	0.05169
Shortest path, model-based	0.20730	0.27023	0.23280	0.04586	0.06043	0.05173

Table 1: Evaluation metrics on AQUAINT data

Evaluation set ROUGE-1 and ROUGE-2 scores						
Algorithm	**ROUGE-1 Recall**	**ROUGE-1 Precision**	**ROUGE-1 F1**	**ROUGE-2 Recall**	**ROUGE-2 Precision**	**ROUGE-2 F1**
Entity grid, rule-based	0.25371	0.27168	0.26009	0.06173	0.06545	0.06281
Shortest path, rule-based	0.25628	0.26949	0.26154	0.06235	0.06441	0.06305
Entity grid, model-based	**0.25656**	**0.32682**	**0.28385**	**0.06695**	**0.08390**	**0.07365**
Shortest path, model-based	0.24993	0.31408	0.27512	0.06276	0.07798	0.06882

Table 2: Evaluation metrics on Gigaword data

Topic	**Individual annotator scores**				**mean**
Plane crash Indonesia	2	3	4	3	3.0
Tuna overfishing	2	2	3	4	2.75
China food safety	3	1	2	2	2.0
Cyclone Sidr	2	2	3	3	2.5
Dimona attack	2	2	3	4	2.75
Sichuan earthquake	2	2	3	2	2.25

Table 3: Human evaluation of 6 summaries