Ling 573 - NLP Systems and Applications

Elena Khasanova, Erica Gardner, Saumya Shah, Sophia Chan and **Vikash Kumar**
Department of Linguistics
University of Washington, Seattle
{ekhas1, erstgard, saumyahs, schan2, vikas134}@uw.edu

Abstract

Text summarization is the process of distilling the most important information from a source to produce an abridged version for a particular user and task. In this paper, we discuss a method used for extractive multi-document text summarization. We use topic modelling to find the sentences in a document that are most relevant to a topic. Thus we rank the most relevant words in a topic and uses it to compute a phi score or a word-topic score. The phi score allows us to pick out a sentence that strongly represents the topics in the document. Our end-to-end system produces a ROUGE-2 average F-1 score of 0.05981. Our system performs nearly as well as the baselines, and we find no statistically significant difference.

1 Introduction

Automatic Text Summarization is one of the most challenging and interesting problems in the field of Natural Language Processing (NLP). The demand for automatic text summarization systems is spiking these days thanks to the availability of large amounts of textual data. There are many different paths from a set of documents to a summary.

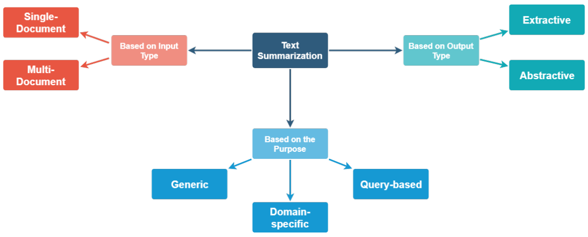

Figure 1: Types of summarization methods

Figure 1 shows the different types of summarization methods based on the type of input, type of output and purpose. We explore the task of extractive multi-document summarization for our project.

For the deliverable 3, we have decided to explore the use of different content selection and information ordering methodologies. We have also made changes to the LDA algorithm to incorporate the TF-IDF scores that we produced in the preprocessing stage. The focus of this deliverable lies on the information ordering, and we have experimented with different methods that will remove redundancy and select the top N sentences for the summary.

2 Literature Review

For the different steps in our implementation, we have relied on certain research areas whose implementations have allowed us to improve our model. In the content selection module, we have made use of TF-IDF that is calculated in the preprocessing stage to aid the topic modelling algorithm to choose the most important topics based on the TF-IDF score.

We follow the implementation of Latent Dirichlet Allocation proposed by (Blei et al., 2003). The goal of LDA is to best estimate the topics that could have generated the observed documents. Each document is described as a distribution over a fixed number of topics and each topic is described as a distribution over the known vocabulary. For our purposes, we build an LDA model with three topics and used the most relevant terms of each topics as a measure of word relevance.

For content selection methods, we have also explored the use of graph-based methods such as TextRank (Barrios et al., 2016), which is based on the PageRank algorithm which was used to compute the rank of web pages. Graph-based ranking algorithms are a way for deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph.

For information ordering, we used the formula in (Barzilay and Lapata, 2008) to calculate coherence scores which is the average cosine similarity across the sentences. To explore semantic coherence we studied the approach proopsed by (Foltz et al., 1998). We also used the BERT (Devlin et al., 2018) sentence embeddings to calculate cosine similarity between the sentences.

For content realization, we followed the heuristics suggested in (Conroy et al., 2006) to trim our sentences.

3 System Overview

Our approach highlights a multi-document generic extractive summarization system.

We first get the document sets from the topics using XML readers and organize them to be fed into the preprocessing module. The preprocessing module reads all the documents by topic. The main steps involve stop word removal, lemmatization, sentence segmentation, and tokenization. At this stage we also produce tf-idf scores for the words in the documents and pass it to the LDA algorithm to choose the top 10 sentences from the topics it generates.

To perform summarization, we use sentence ranking using topic modelling. Topic modelling is an unsupervised method of finding topic clusters within a document and assign a set of words to each topic. Based on the ranking of these words we can find the sentences that contribute most to the topic and hence select content for the summary. Thus the lemmatized sentences for each document provided as input are then organized into topics with a set of vocabulary for each word. Using the word-topic score we find the sentences that are most relevant to summary and is then passed on to the information ordering module.

In the information ordering module, previously, we selected the sentences based on the LDA score from the content selection output and picked the highest ranked sentences to go first. In this deliverable, we have also produced coherence scores using cosine similarity. We also remove the redundant sentence using cosine similarity with sentence embeddings.

For content realization, we experimented with a couple heuristics to get to the 100 word limit such as eliminating sentences shorter than 8 words, removing parenthetical expressions, removing adverbs and gerunds.

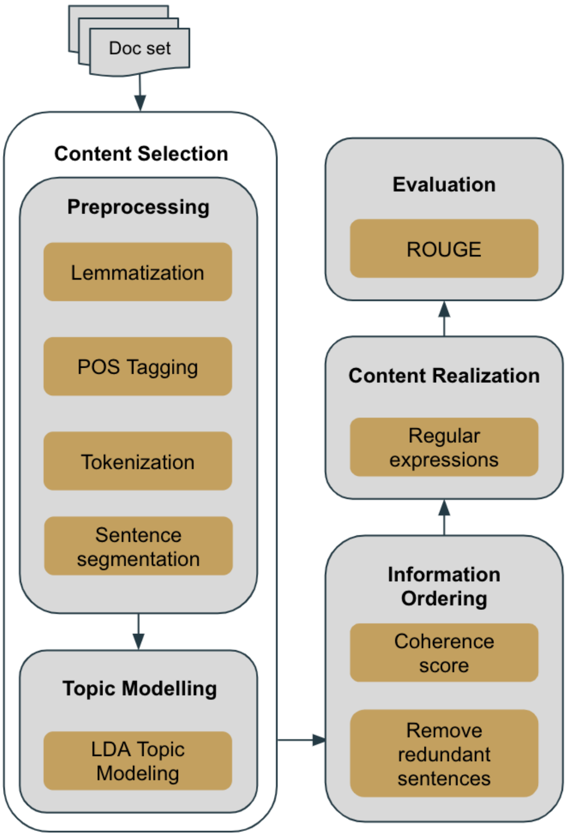

Figure 2: Our proposed system architecture

4 Approach

4.1 Data Extraction

To extract the train, test and development files, we use an XML reader called BeautifulSoup [1], developed in Python. We organized the data into a dictionary with topic_id as the key and the set of documents in the topic as its value. We have also retained parameters like date time, title and narrative, which may provide a good basis for our summarization system.

4.2 Content Selection

The two main components in our selection module includes the preprocessing stage and the topic modelling component using LDA.

4.2.1 Preprocessing

The preprocessing stage was motivated by the main content selection method - topic modeling with Latent Dirichlet Allocation (LDA), which benefits from as little noise as possible in the input data. We used SpaCy[2] processing pipeline to

[1] https://www.crummy.com/software/BeautifulSoup/

[2] https://spacy.io/

obtain various properties of the sentences.

Our preprocessing for topic modeling simply consists of the following steps:

- stopwords removal;
- lowercasing and lemmatization;
- sentence segmentation and tokenization.

This ensures that important topical concepts are not obscured by morphology and increase the performance of LDA.

To retain as much information as possible to further used in downstream tasks, we summarized the properties and various annotations and statistics of the input sentences and grouped them by the *topic (document set) identifier*. We filtered out *interrogative* sentences as those that would be less likely to represent the main content of the news articles. The annotations and statistics include *named entity recognition, part-of-speech tagging, dependency parsing, sentence lengths, sentence index, document index,* and *total number of sentences* in corresponding documents. The indices and lengths are useful in information ordering and content realization, while POS-tagging, dependency parse and named entity recognition are assumed to serve as heuristics and help take advantage of the properties of news genre and weight the candidate sentences accordingly. Figure 3 gives the overview of this component.

4.2.2 Topic Modeling

Latent Dirichlet Allocation (LDA) is a generative statistical modeling technique that allows us to estimate unobserved (latent) topics from an observed distribution of documents (bag of words). We have used the *gensim* package which provides both single-core and multi-core (parallelized) implementation of the LDA model. We have used the default value of {1/number of topics} for the hyper-parameter *alpha* .

One of the drawbacks of LDA is that the latent topics are not always comprehensible in a semantically meaningful or ordinal way, these topics are usually only inferred after manual evaluation. We also observed that the document-topic distribution is not always consistent and is usually skewed towards one of the topics. This makes it restrictive towards extracting sentences from the corpus in accordance with the document-topic probability distribution without under-representation from other topics. We also observed a high overlap of relevant terms(we selected top 50 terms) between topics, these terms/words usually represent the functional words or constant theme across the corpus. To counter these effects, We have used a variation of the *tf-idf* scores to calculate the probability distribution of a sentence over the topic space. In our implementation. The *idf* score is calculated using the standard approach; however, instead of using the normalized *tf*, we use the LDA computed topic-term probability (*phi*) as a measure of importance. This approach has two advantages (i) It demotes common terms/common theme terms which usually overlap among all topics (ii) It elevates unique terms that may have lower *phi* score in a term-topic distribution but the high *idf* score assigns a higher weight to it.

We intend to improve the content selection further more by focusing on span selection instead of selecting entire sentences. Manual observation of the content selection shows that our system is biased towards selecting longer sentences with key topic terms. Our intuition here is that selecting a span around every relevant term will allow us to generate spans which capture the most important information in a sentence worthy to be included in the summary.

4.2.3 Alternatives and Improvements

LDA gave us reliable scores to select the most content rich sentences based on the latent topics distribution over sets of related documents. We considered other metrics to score the sentences that are known to be performant for extractive summarization such as TF-IDF and concreteness scores. Both methods were implemented but kept for the future iterations of our system. TF-IDF (term frequency-inverse document frequency), which reflects the importance of a lexical unit to a document in comparison to other documents. This score was computed using *TfidfVectorizer* from *sklearn* module.

Concreteness scores are obtained from the dataset created by (Brysbaert et al., 2014). This dataset contains the concreteness ratings for 40 thousand English lemma words collected in a behavioral experiment with Amazon Mechanical Turk. The scores were summed for each sentence. This metric can be a good proxy for selecting the candidate sentences specifically from the news articles since we expect the relevant information to be written in accessible and concrete language.

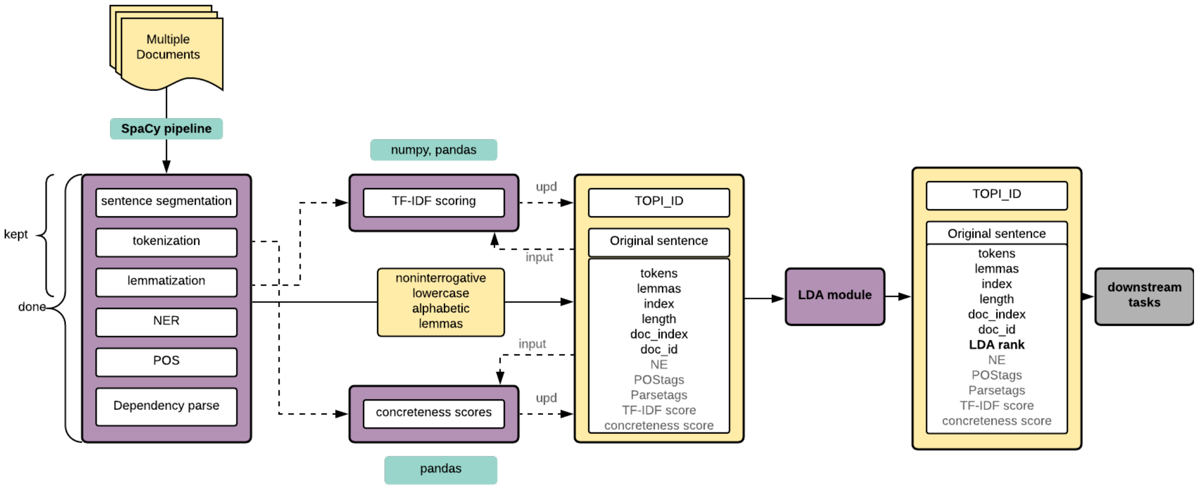

Figure 3: The preprocessing module

We assume that the three metrics - LDA scores that rank the underlying content, TF-IDF scores that rate the representative power of vocabulary items, and concreteness scoring that works in line with the genre - will work better in combination.

4.3 Information Ordering

Our information ordering module takes as input a list of 20 sentences selected for by the content selection module. First, we remove redundant sentences by setting a *redundancy threshold*. Second, we find the most coherent summary by permuting the sentences and computing a *coherence score*. We pick the ordering that gives us the highest coherence score. Both these measures rely on cosine similarity, and thus require a vector representation of the sentence.

4.3.1 Sentence Embeddings

We experimented with 300-dim GloVe embeddings through spaCy (Pennington et al., 2014), and 768-dim BERT embeddings through Hugging Face (Devlin et al., 2018). For the BERT embeddings, we try extracting them from the cased version of pretrained BERT `bert-base-cased`[3] and also from a model that has been fine-tuned on a paraphrase detection task `bert-base-cased-finetuned-mrpc`[4]. For each type, we manually assessed pairwise cosine similarity scores across a set of training sentences. We also observe changes in ROUGE-2 when switching out embeddings.

4.3.2 Redundancy Threshold

To select a redundancy threshold, we annotate a small subset of training sentence pairs with the labels *redundant* and *coherent*. We then embed our sentences using one of the methods mentioned previously, and calculate pairwise similarity across pairs of sentences. We then compute the average cosine similarity score within each set. As a starting point, we select a threshold that maximally separates the two sets. Finally, we refine the threshold by observing changes in ROUGE-2 scores. After removing redundant sentences using this threshold, we calculate a coherence score among different orderings of the remaining sentences.

4.3.3 Coherence Score

To find the most coherent summary among the remaining non-redundant sentences, we permute the sentences and calculate a coherence score for the order (Foltz et al., 1998). The coherence score is the averaged pairwise cosine similarity, as shown in (Barzilay and Lapata, 2008):

$$\text{coherence}(T) = \frac{\sum\limits_{i=1}^{n-1} \cos(S_i, S_{i+1})}{n-1}$$

The information module returns a list of sentences containing the ordering with the highest co-

[3] https://huggingface.co/bert-base-cased
[4] https://huggingface.co/bert-base-cased-finetuned-mrpc

Upper word limit	ROUGE-2 (Average Recall)
60	0.04584
50	0.04741
40	0.0404
30	0.01477

Table 1: Effects of changing the upper word limit for sentences on ROUGE-2 Average Recall.

herence score.

4.4 Content Realization

For our content realization module, we kept the basic heuristics from the previous deliverable, which cut down on unnecessary information and performed sentence compression. Since we used an extractive approach and many of the sentences were long to begin with, cutting out unnecessary information meant we were able to incorporate additional sentences and bring our summary word counts closer to 100.

Our first heuristic was to ignore sentences with fewer than 8 words. We assumed these would by bylines or segments of titles, so would be less likely to contain pertinent information. For this deliverable, we also introduced an upper word count, which eliminated sentences with more than 50 words. We ran experiments with various upper word limits and found we got the best ROUGE scores with 50 words as the upper limit. See Table 1 for the Rouge-2 scores obtained from different upper word limits for sentences.

As with D2, we removed parenthetical expressions, in particular, those enclosed in parentheses, brackets, and hyphens. We also continued to remove adverbs, though we chose to keep sentence-final adverbs, since their removal led to poor readability in some cases.

We modeled our additional heuristics after those set out in CLASSY 2006. (Conroy et al., 2006). We added each heuristic in an iterative manner and tested the resulting ROUGE scores after each subsequent addition, because it is quite easy to go too far with surface level, regular expression-based approaches. The CLASSY-based sentence trimming heuristics that made it into our final system are the following:

- Clauses with unnecessary phrases such as “however”, “also”, “at this point”, “as a matter of fact”
- ages in clauses such as “aged 24”
- gerund phrases in the middle of a sentence
- attributive clauses contained “said”, “stated”, “according”

Our algorithm fills in the summary with sentences in order of LDA rank, but, should the addition of a sentence put us over the 100 word limit, we opt for the next best sentence. This is done to promote readability, since we do not want to have sentences truncated at 100 words.

Since we wanted to maximize the amount of content we included up to the 100 word limit, we evaluated the word count of our summaries before and after incorporating our content realization updates. Prior to incorporating the additional CLASSY heuristics, 20% of our output summaries contained 30 words or fewer. The additional heuristics gave us no summaries under 70 words, which was a satisfactory result.

For the next deliverable, we would like to incorporate additional heuristics that will address some of the issues we found in our output summaries (see Qualitative Analysis section below for additional details). We aim to keep working to bring our summaries as close to 100 words as possible, which will involve storing multiple versions of sentences and picking the shortest version where possible.

For this deliverable, we continued to deal with the surface level of the sentences. For the next submission, we want to incorporate some deep processing methods and look at entity linking and coreference resolution, both for readability and for sentence compression.

5 Results

A comparison of our system to baselines is shown in Table 2.

There is no statistically significant difference between ROUGE-2 Recall achieved by our system and that of the LEAD and MEAD baselines. In addition to computing ROUGE scores, we also qualitatively evaluate our summaries.

We use a standard implementation of ROUGE to evaluate our summaries. The metric measures overlap between automatic and human summaries. Our primary metric is ROUGE-2 Recall, which measures bigram overlap between the system-generated summary and human-generated summary. A higher ROUGE score signals better overlap with the human summary. This metric has

	ROUGE-2 (Average Recall)
LEAD Baseline	0.05376
MEAD Baseline	0.05927
D2 System	0.04340
D3 System	0.05427

Table 2: Comparison of our D3 system to baselines. ROUGE-2 Average Recall is reported.

been shown to correlate well with human assessments of responsiveness and to work well for extractive summaries (Rankel et al., 2011).

The ROUGE-2 Average Recall was 0.05427 (95% CI [0.04508-0.06400]). We compare our performance against that of two standard baselines. The LEAD baseline takes the first 100 words as the summary, while MEAD is an exemplar centroid-based summarization system (Radev et al., 2001). Our system outperforms the LEAD baseline, and performs nearly as well as the MEAD baseline. In fact, we find no statistically significant difference between our system and the two baselines based on our metric.

6 Discussion

6.1 Ordering and Selection

The qualitative analysis of the summaries revealed several problematic areas. First, some summaries contain sentences that are either irrelevant or lacking coherence with the surrounding context. One reason for this is the limitations of LDA, which favors longer and more topic-rich sentences and has no internal mechanism to differentiate between the sentences with the same scores. As a result, content realization may select the sentences of a suitable length but leave out the most relevant sentences that are longer. These issues can be tackled by scoring the LDA subtopics, distributing the sentences across the subtopics and controlling that the selected sentences cover the most highly ranked distinct subtopics.

6.2 Redundant Fragments

Another problem is redundancy of sentence fragments. Being the main focus of this release, sentence-level redundancy was almost completely eliminated (only 6 summaries out of 46 still contain redundant sentences), but cosine similarity is much less efficient with sentence fragments that do not significantly change the total score being negligible proportionally to the sentence length. Although redundant fragments were spotted in less than 22% of the summaries, we expect that using dependency parses or phrase-structure level instead of punctuation as a heuristic would help solve this problem.

6.3 Redundant Attribution, Coreference, Connectors

The issues with redundant attribution, unresolved coreference and sentence connectors were not yet given sufficient attention and expectantly propagated to our summaries. Finally, the interplay between different modules remains a problem to solve. For instance, the content realization is currently centered around redundancy reduction, and thus the improvements to content selection resulting in less repetitive made the former inefficient and harmed the performance. We are yet to discover the best configuration of the different modules in the pipeline with the help of ablation study of different components and hyperparameters. Overall, about 46% of the summaries are of sufficient quality and need only minor improvements, which is a promising result.

7 Conclusion

An extractive multi-document summarization system was successfully implemented using topic modeling with LDA as the main content selection method and surface level filtering of the summaries produced by the system. The end-to-end system achieved the ROUGE-1 and ROUGE-2 scores comparable with the baselines and outperformed the baseline in one out of three cases. The scores show that our approach has the potential to yield good results.

This deliverable is largely focused on experimenting information ordering and content realization methods and improving the concerns in the last deliverable such as redundancy. We also experimented with several content selection methods and built the scaffolding for a more refined versions of this module.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606.

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

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

M. Brysbaert, A.B. Warriner, and V. Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior Research Methods*, 46(1):904–911, August.

John M. Conroy, Judith D. Schlesinger, Dianne P. O'Leary, and Jade Goldstein. 2006. Back to basics: Classy 2006.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Peter W. Foltz, Walter Kintsch, and Thomas K Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25(2-3):285–307.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Dragomir R Radev, Sasha Blair-Goldensohn, and Zhu Zhang. 2001. Experiments in single and multidocument summarization using mead. In *First document understanding conference*, page 1À8. Citeseer.

Peter A Rankel, John Conroy, Eric Slud, and Dianne P O'leary. 2011. Ranking human and machine summarization systems. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 467–473.