LING 573 Project Report

Macklin Blackburn, mblac6@gmail.com
Xi Chen, xch@uw.edu
Yuan Zhang, yuanz80@uw.edu

May 16, 2017

Abstract

We are building a multi-document automatic text summarization system. Our content selection module is based on multi-layer perceptron regression that uses statistical and linguistic features.

1 Introduction

Automatic summarization is an NLP task that turns a set of documents related to the same topic into a summary. This task was first attempted by Luhn (1958). In recent decades, there have been several different approaches to the problem. These approaches fall into two main categories: supervised and unsupervised. Examples of the former category include Schilder(2008), Vanderwende(2007), and Cao(2015). Examples of the latter include Erkan(2004) Otterbacher(2005) and Radev(2004).

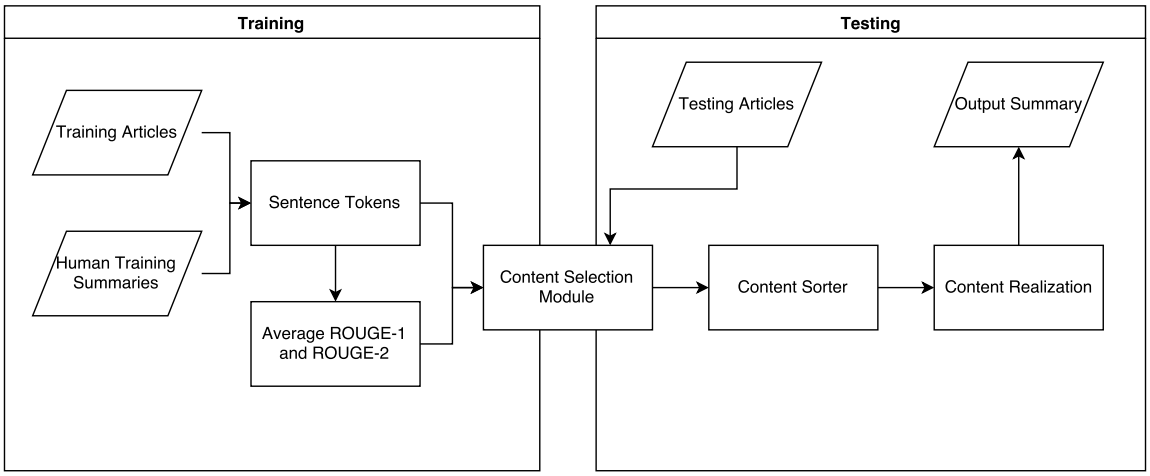
This paper demonstrates a system composed of three sub-systems. First, each sentence is assigned a score to indicate its importance in the final summary. Second, sentences are sorted by scores. Final, a summary is picked out from sorted sentences.

One applicable supervised method is to model the task as sequential labeling, given the input of several features of each sentence. The system will try to predict whether the sentence belongs to the final summary:

$$F(f_{n1},f_{n2},...f_{nk}) \rightarrow Label_n$$

2 System Overview

2.1 System architecture


Training
Training Articles
Human Training Summaries
Sentence Tokens
Average ROUGE-1 and ROUGE-2
Content Selection Module
Testing
Testing Articles
Output Summary
Content Sorter
Content Realization

3 Approach

3.1 Data

We carry out our experiments on four datasets, three from the Document Understanding Conference (DUC) and the other from SummBank 1.0. The DUC 2007, 2009 and SummBank data are used for training and the DUC 2010 data are used for development and testing. In addition, the

There are 45 multi-document clusters in DUC 2007, 44 in DUC 2009, and 46 in DUC 2010. Each cluster contains 10 news articles on the same topic. For each cluster, there are 8 human-written summaries. The SummBank multi-document data consists of 40 document clusters. Each, in turn, contains 10 news articles on a related topic. For each cluster, there are three groups of summaries of different word limits, 50, 100, and 200. For each word-limit group, there are three summaries written by different human judges. Most of these summaries are abstractive.

In addition, we extract from the DUC 2009 peer summaries all the human-ordered extractive summaries, marked with the annotator ID 3, for training and evaluating our information ordering system.[4]

3.2 Preprocessing

For preprocessing, we split each document into sentences using the NLTK sentence tokenizer, lowercase each sentence, remove stop words and non-alphanumeric characters, and lemmatize all verbs and nouns using NLTK WordNetLemmatizer package. For each dataset, we produce two parallel dictionaries, one containing the original sentences and the other processed sentences as lists of word tokens, respectively. Finally, we remove from the dictionaries the sentences, which, after preprocessing, have been reduced to empty lists. We use the processed sentences as the input to our system and the original dictionary for easy lookup.

3.3 Topic Orientation

For topic-guided summarization, we adopt the query-based LexRank approach [2]. We combine a continuous LexRank score and a relevance measure into a stochastic graph-based model and we give the two measures different weights in multiple trials. The topic relevance measure is heavily dependent on word token overlaps between a topic or query and a sentence, balanced by an IDF score. The continuous LexRank aims to get at the salience of each sentence in the document set by a cosine similarity measure applied on the tf-idf score of each word in the sentence. The result of this experiment will be presented in Section 5.1.

3.4 Content Selection

Our content selection system uses multi-layer perceptron regression to identify sentences that are likely to have high ROUGE scores based on a number of features. The ranking of the features is shown below.

Feature	Score
Query-Based Lexrank	0.133033
LLR	0.121160
Sentence Length	0.080823
Earliest First Occurrences	0.065079
LLR Sum	0.064063
Count of NN	0.056965
Is First Sentence	0.054739
Sentence Index	0.039904
Average First Occurrences	0.037592
TF*IDF Average	0.032323
Reverse KL Divergence of Bigrams	0.028661
Lexrank	0.027235
KL Divergence of Bigrams	0.025725
KL Divergence of Unigrams	0.025337
Average Position of words in Documents	0.025042
TF*IDF Sum	0.019876
Count of JJ	0.019386
Reverse KL Divergence of Unigrams	0.01817
Sentiment Intensity Score	0.017199
Probability of Number	0.014041
Count of VBP	0.012650
Count of VB	0.011904
Count of RB	0.011716
Count of NNS	0.010761
Count of IN	0.010759
Count of VBD	0.008942
Count of VBZ	0.006163
Count of NNP	0.005436
Count of VBN	0.004228
Count of PRP	0.003805
Number of Commas	0.003573
Count of DT	0.001987
Count of CC	0.001718

Since our system took inspiration from the RegSum system, it includes some of the same features, such as the earliest first occurrence of a word or the average position of a word in a document. These features are usually averaged over the words of a sentence, but sometimes a sum was a more effective feature. The sentence intensity score was computed using VADER as included in the NLTK package for python, and was intended to approximate several features in RegSum that contained sentiment information. However, there are several features

in this system that are not present in RegSum, such as the KL divergence of bigrams. The content selector uses NLTK's part of speech tagging to count the number of certain parts of speech in the sentence.

During the training stage, the system constructs a vector for each sentence containing each of the features and assigns the sentence a score. The score is computed by averaging all ROUGE scores of the sentence compared to the relevant summaries. Earlier versions of the system assigned each sentence a binary score which represented whether the sentence was in the summary or not, but training with the ROUGE scores yielded better results. After assigning each sentence a score, the sentence vectors are scaled using a function from the SciKit Learn package and the sentence scores are scaled so that the highest score is 1 and the lowest is 0.

3.5 Sentence Ordering

Because the sparcity of ordering learning data, our information ordering procedure is unsupervised one. It is based on sentence cluster adjancency Ji(2008). Given a set of summary $\{S_1,S_2,...S_K\}$, we cluster all sentence of the original documents into K groups $G_1,G_2,...,G_K$ where S_i corresponds to G_i. For each group pairs, the adjacency of G_i and G_j is defined below:

$$C_i, j = \text{pow}(\text{freq}(G_i,G_j),2)/(\text{freq}(G_i)* \text{freq}(G_j))$$

Here f(G_i) denotes the frequency of cluster G_i in source documents and freq(G_i,G_j) denotes the co-occurence frequency between G_i and G_j in a word-window, which current size is 2.

Our current implementation always picks the first sentence in the first published document as the first sentence. After that, a greedy algorithm can be used to order the summary sentences. That is, to find the (i+1)th sentences by maximize the adjacency:

$$S_i + 1 = \arg\max(C_i, j) \text{ where i is the previous sentence in the summary}$$

The implementation uses label spreading routine of scikit-learn to make the clustering and Glove 50D word embedding to measure sentence cosine similarity. Also, we are making a stadnalone evaluation of the ordering procedure. It will be based on the Kendall's tau, which is defined as:

$$\text{tau} = 1 - 4(\text{number of inversions})/\ \text{N(N-1)}$$

Here N is the number of summary sentences. Sentences ordered by the algorithm will be compared to human extracted summaries (AQUAINT and AQUAINT-2) to get the score. A baseline of random ordering will be used as the baseline.

3.6 Surface Realization

The current surface realization is simple: it fetches all sentences with highest possible scores until the final summary length is reached. The only exception is when the TF*IDF cosine similarity between a candidate sentence and a chosen sentence in the summary is greater than a threshold. The current threshold is set to 0.4.

4 Results

4.1 ROUGE Recall Scores

Model	ROUGE-1	ROUGE-2	ROUGE-3	ROUGE-4
D2	0.18687	0.04579	0.01503	0.00558
D3	0.22459	0.05354	0.01287	0.00424

5 Discussion

5.1 Topic Orientation

Our D3 submission contains a query-based lexrank score as a feature in the content selection model. Query-based lexrank was ranked the highest of the features by information gain and improved the ROUGE-2 recall of our overall system by roughly 0.008. An experimental feature that measured the KL divergence between a sentence and the topic of the cluster was ranked as the third best feature by information gain but lowered the overall ROUGE-2 recall by roughly 0.007. It's unclear why topic-focused lexrank is so much more effective than topic-focused KL divergence, especially when the information gain of the KL feature is comparable. Our speculation is that the model may rely too heavily on sentence word overlap with the topic, so that it favor sentences that are too similar to each other and contribute roughly the same information to the end summary. Query-based lexrank might be a good feature because it balances relevance to the topic with relevance to the document cluster as a whole.

5.2 Content Selection

6 Conclusion

This is a work in progress. We have improved our evaluation scores by preprocessing, adding features, adding training data, and by doing information ordering. More will be done in the surface realization module.

References

[1] G. Erkan and D. Radev. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artificial Intelligence Research, 22(1):457-479*, 2004.

[2] J. Otterbacher G. Erkan and D. R. Radev. Using random walks for question-focused sentence retrieval. *Proceedings of Human Languages Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp.915-922*, 2005.

[3] K. Hong and A. Nenkova. Improving the estimation of word importance for news multi- document summarization. *in Proceedings of EACL*, 2014.

[4] P.E. Genest G. Lapalme and M. Yousfi-Monod. Hextac: The creation of a manual extractive run. *Proceedings of the Second Text Analysis Conference (TAC 2009)*, 2009.

[5] A. Nenkova and K. McKeown. Automatic summarization. *Foundations and Trends in Information Retrieval*, 2011.