# CS585 Project Report
# Long Text Summarization using Neural Networks and Rule-Based Approach

Shujian Liu

January 5, 2017

## 1 Abstract

Automatic text summarization is the task for computers to produce a concise and fluent summary conveying the key information in the input. There are generally two types of automatic texts summarization: extraction and abstraction. Many papers has been published on extraction summarization, however, it cannot provide summary close to human language. This project will focus on using neural networks techniques and statistical language models for abstractive summarization on long texts. Results have shown that picking shortest-clause from most important sentence choosen by LexRank Algorhtm appears to have the best performance.

## 2 Introduction

Automatic text summarization is the task for computers to produce a concise and fluent summary conveying the key information in the input [1]. Automatic text summarization technologies are widely used in the industry such as search engines. Developing capable machine learning models can automatically deliver accurate summaries of longer text, such as news stories and social media posts, can be useful for digesting such large amounts of information [2].

There are generally two types of automatic texts summarization: extraction and abstraction. Many papers has been published on extraction summarization, however, it cannot provide summary close to human language.

With the help of neural networks, there are great achievements for abstractive methods by Google, Facebook and IBM in recent years. However, these achievements are only based on very short documents. Facebook [3] used the first sentence of a document while Google [2] and IBM [4] use the first 2 sentences. It is still challenging to summarize longer texts where it is necessary to read the entire document in order to produce good summaries. This project will focus on using neural networks techniques and statistical language models for abstractive summarization on long texts.

# 3 Dataset

Opinosis Opinion Dataset 1.0 [1] is used in preliminary research. This dataset contains 51 articles. Each of them includes sentences extracted from reviews on a given topic. The opinions are obtained from Tripadvisor, Edmunds.com(cars) and Amazon.com. There are approximately 100 sentences per topic. Every topic has five manually written gold summaries for comparison. The total size is 1.0 Mb.

A larger dataset named DeepMind QA [2] in the later part of this project. Hermann et al. (2015) created this two awesome datasets using news articles for QA research. Each dataset contains many documents (90k and 197k each) and the dataset has a total size of 1.3 Gigabyte after compression. This dataset is modified here to extract gold summaries and clean stories from the news documents.

Since all the baseline algorithms are unsupervised, no train/test split of dataset is needed.

# 4 Related work

## 4.1 Baseline/Heuristic Extractive Algorithms

Four baseline extractive text summarization algorithms are used in this project.

### 4.1.1 Luhn

Luhn's algorithm [5] was invented for the purpose of automatically creating abstracts for technical literature, which can save readers' time and effort when finding useful information. It used the word frequency and distribution to calculate the relative significance of word and sentences. The summary is made by extracted the sentences with the highest significance.

### 4.1.2 Edmundson

Unlike Luhn's algorithm, which focused only on the sentence significance, Edmundson [6] used methods with three additional components: pragmatic words, title and heading words and sentence location. Results have shown that adding the last three components can improve extracts generation.

### 4.1.3 Latent Semantic Analysis (LSA)

Gong et al. [7] has applied Latent Semantic Analysis to identify semantically important sentences for summary creations. This approahced was inspired by latent semantic indexing and the singular value decomposition (SVD) is applied.

Steinberger et al. [8] modified singular value decomposition matrices to obtain better results.

### 4.1.4 LexRank

LexRank [9] is a stochastic graph-based algorithm and computs sentence importance based on eigenvector centrality in a graph representation of sentences. A connectivity matrix is used as the

[1] Opinosis Dataset. http://kavita-ganesan.com/opinosis-opinion-dataset

[2] DeepMind QA Dataset: http://cs.nyu.edu/ kcho/DMQA/

adjacency matrix to represent sentences. The connectivity matrix is calculated by intra-sentence cosine similarity.

## 4.2 Graph-based Abstractive Summarization

Kavita Ganesan *etc.* used a graph-based summarization framework named Opinosis, which generates concise abstractive summaries from highly redundant opinions. The main idea of Opinosis is to first construct a textual graph that represents the raw text. Then three unique properties of this graph are used to explore and score various sub-paths that help in generating candidate abstractive summaries [10].

Fei Liu *etc.* also used a graph-based approach for abstractive summarization. In their work, the source text is parsed to a set of Abstract Meaning Representation (AMR) graphs, the graphs are transformed into a summary graph, and then text is generated from the summary graph [11].

## 4.3 Abstractive Summarization with Neural Networks

ve methods by Google, Facebook and IBM in recent years. However, these achievements are only based on very short documents. Facebook [3] used the first sentence of a document while Google [2] and IBM [4] use the first 2 sentences. It is still challenging to summarize longer texts where it is necessary to read the entire document in order to produce good summaries. A detailed explanation of neural network With the help of neural networks, there are great achievements for abstractiapproach by Google Brain Team is given in Section 5.1.

## 4.4 ROUGE Package for Evaluation

ROUGE [12] is short for Recall-Oriented Understudy for Gisting Evaluation. It is a package of several methods which can automatic evaluate similarity between text and summary.

### 4.4.1 ROUGE-N

ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries:

$$\text{ROUGE-N} = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)}$$

### 4.4.2 ROUGE-L

In ROUGE-L, Longest Common Subsqequence (LCS) is used to calculate the similarity. There are sentence-level LCS and summary-level LCS. LCS has two advantages: no need for consecutive matches and no predefined n-gram length is necessary.

### 4.4.3 ROUGE-W

Since the ROUGE-L does not differentiate LCSes of different spatial relations within their embedding sequences, a weighted LCS is proposed. In ROUGE-W, the algorithm remember the length of consecutive matches encountered so far to a regular two dimensional dynamic program table computing LCS.

#### 4.4.4 ROUGE-S

ROUGE-S uses skip-bigram co-occurrence statistics. Skip-bigram co-occurrence statistics measure the overlap of skip- bigrams between a candidate translation and a set of reference translations.

# 5 Methods

## 5.1 Neural Networks Approach

In August 2016, Google Brain Team open-scoured an abstractive text summarization model programmed in TensorFlow. The core model is based on the traditional sequence-to-sequence model with attention.

Sequence-to-sequence model proposed by Ilya Sutskever Etc. is a general end-to-end method to sequence learning that makes minimal assumptions on the sequence structure [13]. They used Long Short-Term Memory (LSTM) architecture to solve these problems. The fundamental idea is to use one LSTM to read the input sequence to obtain large fixed-dimensional vector representation (one timestep at a time), and then to use another LSTM to extract the output sequence from that vector. The structure is shown in Figure 1). The second LSTM is essentially a recurrent neural network language model however, it is conditioned on the input sequence. The LSTM reads the input sentence in reverse, to introduce many short term dependencies in the data that make the optimization problem much easier.

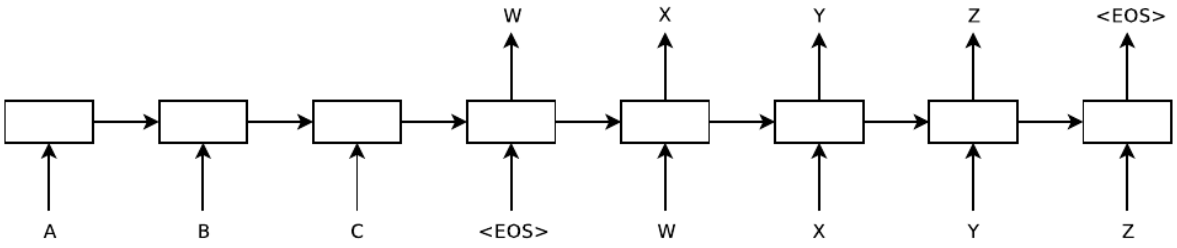


Figure 1: The sequence to sequence model reads an input sentence "ABC" and produces "WXYZ" as the output sentence. The model stops making predictions after outputting the end-of-sentence token [13].

Google Brain team trained the model with Annotated English Gigaword[3] which contains the nearly ten million documents (over four billion words). Due to the nature of news headlines, the model is set to generate headlines from just a few sentences (two to be precise) at beginning of the article [2].

In this project, the neural networks is trained with the provided sample(toy) data. The average loss in training is shown in Figure 2. The training takes approximately one day on desktop. The stopping criteria is set to be 1000 iterations and average loss falls below 1.0 at the end.

[3]Annotated English Gigaword: https://catalog.ldc.upenn.edu/LDC2012T21

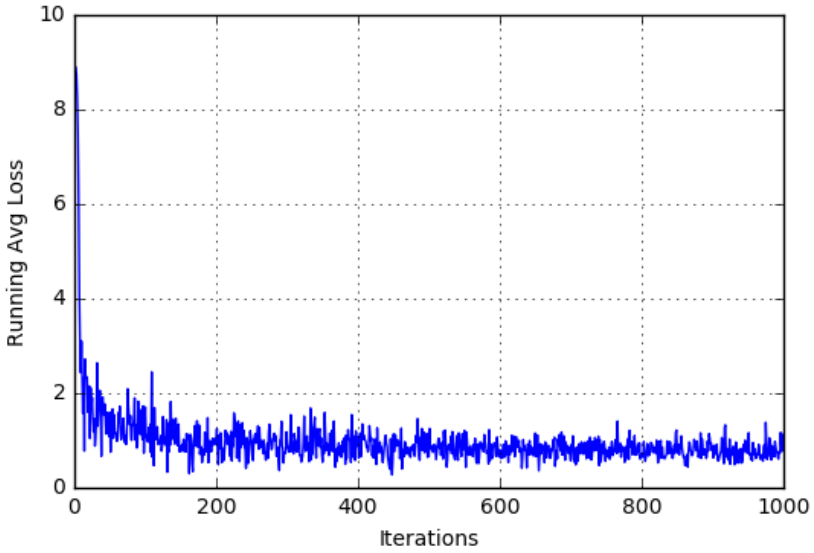


Figure 2: Training Loss of Recurrent Neural Networks

The testing results generated with the sample training data (about 40 articles) are disappointing: every word in the results is <UNK>, which stands for unknown vocabulary. Pavel Surmenok has trained the RNN with a much larger news dataset: CNN and DailyMail [14]. 322k articles exists in the dataset however the results are still unsatisfied. His sample decoding results are:

"your your <UNK>"
"We'll the <UNK>"
"snow hit hit hit <UNK>"

As Daniel Krotov pointed out that training against 1.3 million articles seems to provide descent results[4].

## 5.2 Halfway Discussion

After spent over a month on learning neural networks and TensorFlow, and test Google TextSum locally and on Amazon AWS, I realized that summarization with recurrent neural network is not computationally feasible for my project. On the other hand, the summary generated by Google TextSum are simply based on the first one or two sentence of every article. Is it really necessary to use such complicated neural network and huge training dataset for this task? In the next part, some basic statistical language model methods are used based the materials from CMPSCI 585.

## 5.3 Rules-based Summarization

### 5.3.1 Statistical Language Models

A statistical language model is a probability distribution over sequences of words. A natural language parser is a model that find the grammatical structure of sentences. Among these parsers,

[4]textsum: Format of test data: https://github.com/tensorflow/models/issues/692

Cocke-Kasami-Younger algorithm (CKY) is a parsing algorithm for context-free grammars which uses bottom-up dynamic programming. It find possible nonterminals for short spans of sentence, then possible combinations for higher spans.

PyStatParser[5] is used in this project which uses CKY parser together learning the probabilistic context-free grammars (PCFGs) from the QuestionBank and Penn treebanks. PyStatParser is based on NLTK library, which is a leading platform for building Python programs to work with human language data.

In order to employ statistical language model for summarization, the summary from Google TextSum is reviewed here. Table 2 shows the results from TextSum with colorful marks for the ease of matching. CKY is used to parse the examples in TextSum. The parsing trees are shown in Figure 3-5.

[5]pyStatParser: https://github.com/emilmont/pyStatParser

Table 1: Summarization Results by Google TextSum

| **Input: Article 1st sentence** | **Model-written headline** |
|---|---|
| metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year | mgm reports 16 million net loss on higher revenue |
| starting from july 1, the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products to prevent the possible spread of epidemic diseases | hainan to curb spread of diseases |
| australian wine exports hit a record 52.1 million liters worth 260 million dollars (143 million us) in september, the government statistics office reported on monday | australian wine exports hit record high in september |

Note: Red highlight means matching, green nightlight means rephrasing, and blue highlight means missing information

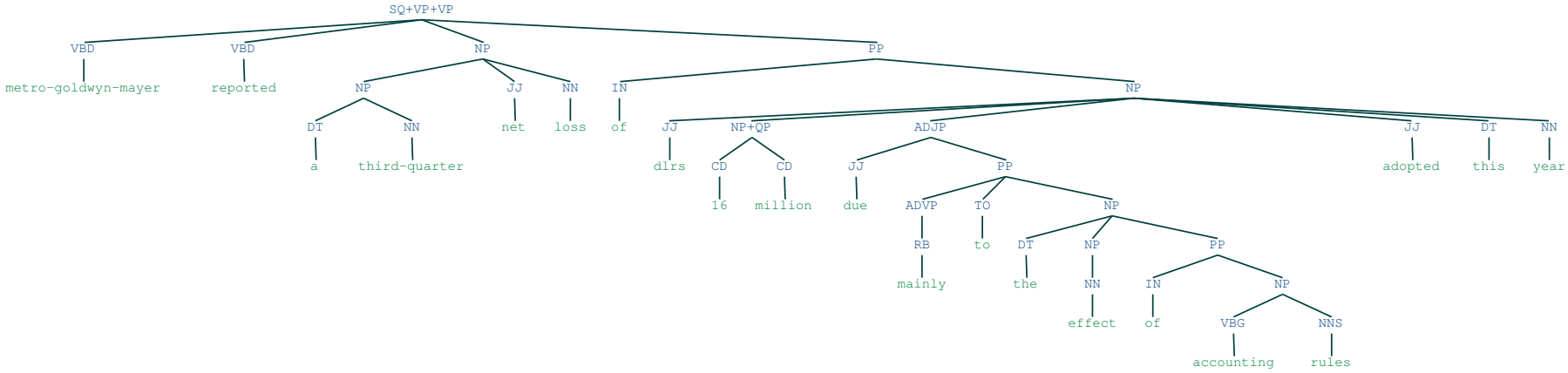


Figure 3: Parsing Tree of Example 1

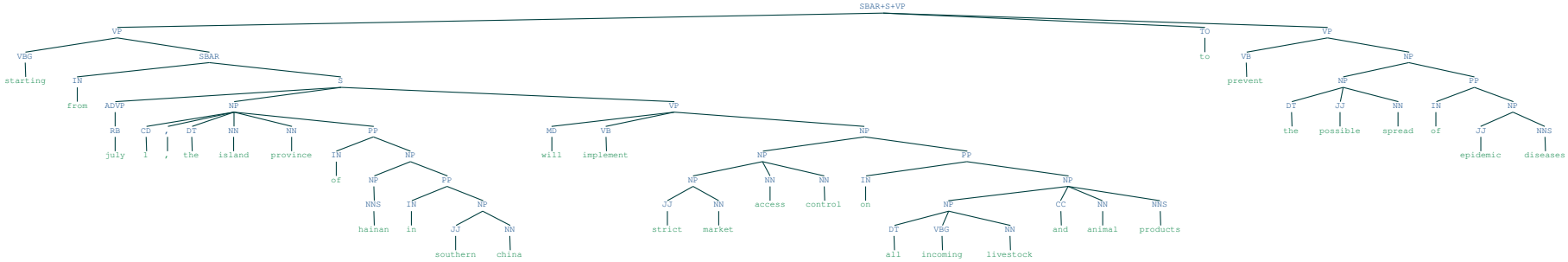


Figure 4: Parsing Tree of Example 2

∞

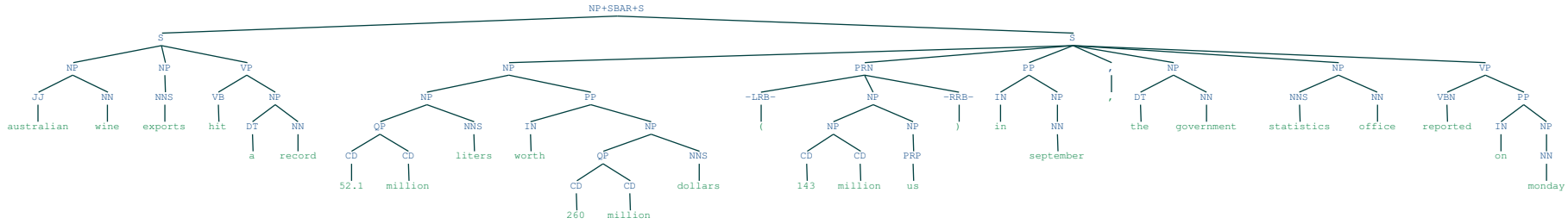


Figure 5: Parsing Tree of Example 3

### 5.3.2 Proposed Rules for Summarization

After comparing the summaries generated from TextSum and observing the parsing tree generated by CKY. Some rules can be concluded: Example 1 suggests using words at top levels in the tree. Example 2 suggests using main clause after comma while example 3 suggests using main clause before the comma. Based on these observations, several empirical rules are proposed here to compress the sentences. Reasonable summaries are marked red in the tables.

**The Longest Sub-sentence (Separated by Comma)**

Table 2: Summary by Longest Sub-sentence

| | Summary | Compression ratio |
|---|---|---|
| Example 1 | metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year | 1.0 |
| Example 2 | the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products to prevent the possible spread of epidemic diseases | 0.88 |
| Example 3 | australian wine exports hit a record 52.1 million liters worth 260 million dollars (143 million us) in september | 0.72 |

**Top Levels Words in the Tree**

Table 3: Summary by Top 3 levels in Parsing Tree

| | Summary | Compression ratio |
|---|---|---|
| Example 1 | metro-goldwyn-mayer reported a third-quarter net loss of dlrs adopted this year | 0.52 |
| Example 2 | starting from to prevent | 0.12 |
| Example 3 | australian wine exports hit ( ) in , the government statistics office reported | 0.52 |

Table 4: Summary by Top 4 levels in Parsing Tree

| | Summary | Compression ratio |
|---|---|---|
| Example 1 | metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due adopted this year | 0.67 |
| Example 2 | starting from to prevent the possible spread of | 0.24 |
| Example 3 | australian wine exports hit a record liters worth ( ) in september , the government statistics office reported on | 0.76 |

**Longest Main Clause**

Table 5: Summary by Longest Main Clause

| | Summary | Compression ratio |
|---|---|---|
| Example 1 | metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year | 1.0 |
| Example 2 | july 1 , the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products | 0.73 |
| Example 3 | 52.1 million liters worth 260 million dollars ( 143 million us ) in september , the government statistics office reported on monday | 0.88 |

**Shortest Main Clause**

Table 6: Summary by Shortest Main Clause

| | Summary | Compression ratio |
|---|---|---|
| Example 1 | metro-goldwyn-mayer reported a third-quarter net loss of dlrs 16 million due mainly to the effect of accounting rules adopted this year | 1.0 |
| Example 2 | july 1 , the island province of hainan in southern china will implement strict market access control on all incoming livestock and animal products | 0.88 |
| Example 3 | australian wine exports hit a record | 0.24 |

## 5.4 Hierarchical Model

Truly abstractive summarization has not been matured yet. Existing abstractive summarizers often depend on an extractive preprocessing component. The output of the extractor is cut and pasted, or compressed to produce the abstract of the text [9].

This project will attempt to use neural networks for abstractive summarization. Google, Facebook and IBM has open-sourced their deep learning models for abstractive summarization for short articles. Their model will be tested in this project. A combination of abstractive and extractive summmarization is chosen to implemented.

The proposed model will first use heurestic extractive methods (with preference of LexRank) to reduce the size of long articles. Then modified abstractive methode with neural networks or rule-based approaches will be used to further reduce the summary size and generate result closer to human language. The structure is illustrated in 6.

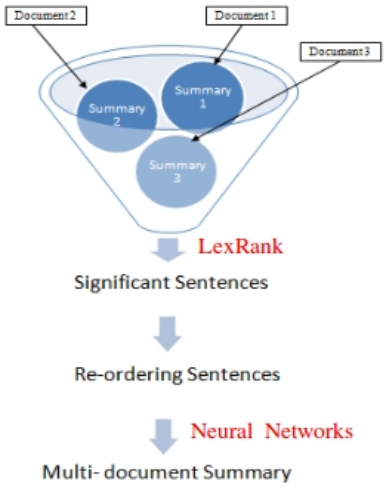


Figure 6: Proposed Hierarchical Model

# 6 Results

## 6.1 Comparing Baseline Extractive Algorithms

The evaluation results by ROUGE package is shown in Figure 7. Algorithms are modified from the SUMY package [6]. Since every article has five gold summaries, the result is the average among all summaries. The output of text summarization algorithms are set to generate two sentences for each article, which is close to the average length of provided gold summaries. LexRank has the best performance for Opinosis Opinion Dataset. Table 7 compares the result from LexRank vs. gold summaries on the topic of accuracy_garmin_nuvi_255W_gps.

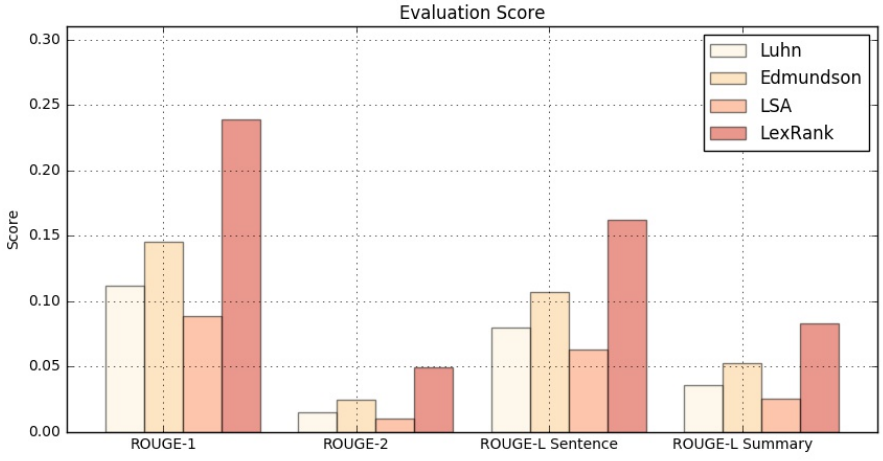


Figure 7: Evaluation Results by Four Baseline Extractive Algorithms

[6] sumy 0.5.0. https://pypi.python.org/pypi/sumy

Table 7: Summarization Results of LexRank

| | |
|---|---|
| Gold 1 | This unit is generally quite accurate. Set-up and usage are considered to be very easy. The maps can be updated, and tend to be reliable. |
| Gold 2 | The Garmin seems to be generally very accurate. It's easy to use with an intuitive interface. |
| Gold 3 | It is very accurate, even in destination time. |
| Gold 4 | Very accurate with travel and destination time. Negatives are not accurate with speed limits and rural roads. |
| Gold 5 | Its accurate, fast and its simple operations make this a for sure buy. |
| LexRank | but after that it is very easy and quite accurate to use . What the 255w does best is find a street address, business, point of interest, hospital or airport and give you turn, by, turn directions with amazing accuracy . |

## 6.2 Using First One or Two Sentence for Summarization

According to Figure 8, using LexRank on the whole article has best performance. Therefore, the assumption of using a few sentences from the beginning of the article may not be valid for this Opinosis dataset.

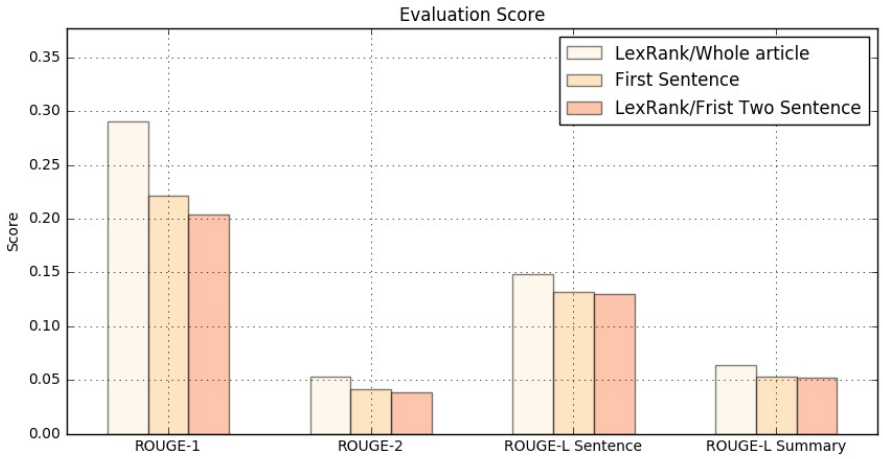


Figure 8: Evaluation Results using First One or Two Sentence for Summarization

## 6.3 Hierarchical Model

The hierarchical model (LexRank picks one sentence and rule-based approaches compress this sentence) is tested on the DeepMind QA Dataset. The results in Figure 9 indicate:

- At unigram level, all five rules improved the summarization. The shortest-clause rule has the best performance.

- At bigram level, longest-sub-sentence, longest-clause and shortest-clause rules have improved the summarization. The shortest-clause rule has the best performance.
- No rule has improved the sentence and summary level of sumarization.

In conclusion, the combination of LexRank and shortest-clause rule has the potential to improve summarization for long text.

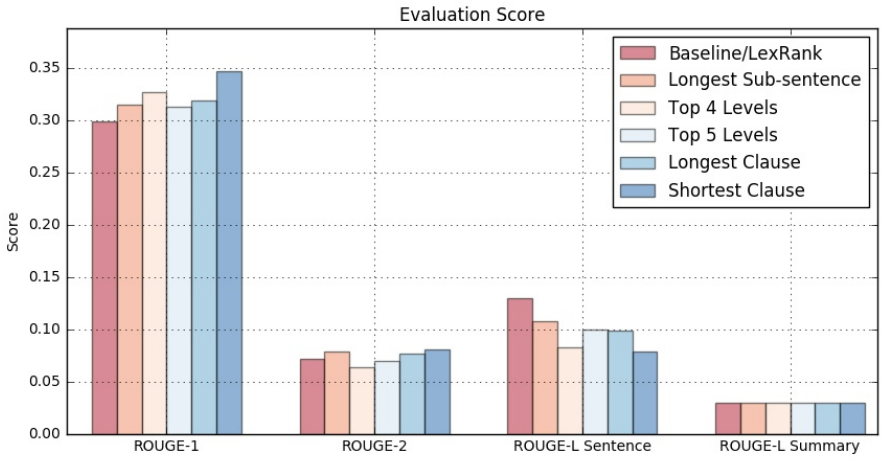


Figure 9: Evaluation Results of Hierarchical Model on the DeepMind QA Dataset

# 7 Discussion

Among four popular baseline extractive sumarization algorithms, the LexRank Algorithm tends to generate the best summary. The summary appears to have the main ideas from the topic. However, comparing with the gold summary, extraction summarization prefers long sentences from the original dataset. The result is not brief enough and also not as fluent as human language. A hierarchical Model is proposed to future compress the summary after pre-processed by LexRank. Two approaches are considered: neural networks and rule-based model. Due to the computation capacity limitation, neural networks are not suitable in this project. Rule-based models are used based on statistical language models. Results have shown that picking shortest-clause from most important sentence appears to have the best performance.

# 8 Future Work

One possible direction for future work can be train TextSum against a dataset larger than 1 million articles, ideally Annotated English Gigaword, which contains nearly 10 million articles. Then implement the hierarchical model with TextSum for sentence compression.

Upon finishing this project, the author noticed a paper under review for ICLR 2017 named *"Unsupervised Pretraining for Sequence to Sequence Learning"* [15], which intializes the encoder

and decoder of the seq2seq model with the trained weights of two language models, and then all weights are jointly fine-tuned with labeled data. This provides a different approach but has similar goal with this project. This approach can be implemented in this project to prove the results in the future research.

# 9 Acknowledgements

The author thanks Prof. Brendan O'Connor for teaching this wonderful course and teaching assitant Haw-Shiuan Chang for invaluable suggestions on the project.

# References


[1] Ani Nenkova and Kathleen McKeown. A survey of text summarization techniques. In *Mining text data*, pages 43–76. Springer, 2012.

[2] Google research blog: Text summarization with tensorflow. https://research.googleblog.com/2016/08/text-summarization-with-tensorflow.html. Accessed: 2016-10-14.

[3] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[4] Ramesh Nallapati, Bowen Zhou, Ça glar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond.

[5] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[6] Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[7] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25. ACM, 2001.

[8] Josef Steinberger and Karel Jezek. Using latent semantic analysis in text summarization and summary evaluation. In *Proc. ISIM'04*, pages 93–100, 2004.

[9] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.

[10] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd international conference on computational linguistics*, pages 340–348. Association for Computational Linguistics, 2010.

[11] Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. Toward abstractive summarization using semantic representations. 2015.

[12] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries.

[13] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[14] How to run text summarization with tensorflow. http://pavel.surmenok.com/2016/10/15/how-to-run-text-summarization-with-tensorflow/. Accessed: 2016-12-20.

[15] Prajit Ramachandran, Peter J Liu, and Quoc V Le. Unsupervised pretraining for sequence to sequence learning. *arXiv preprint arXiv:1611.02683*, 2016.