Computational Linguistics

Automatic Summarizer

Project Update Report

Sulabh Bansal
Himanshu Jindal
Kavita Malani

Objective

To take a document as input and give another document with lesser number of words as output, while retaining the meaning of the original document. The main challenge here is that there is no one single method to evaluate the quality of the summaries. Moreover, there are no gold standard summaries, which can be used for comparison. Hence, our objective is to find summaries by using different features and evaluate them using different techniques and analyze which technique works for which feature and why.

Benchmark Summaries

Because of the aforementioned challenges, we have decided to form 2 categories of benchmark Data

1. Automated summaries

Intellexer Document Summarizer is a desktop application that analyzes the document, extracts its main ideas and puts them into a short summary or creates annotation. This software currently sells for 400$ per copy. [1] However, its trial version can be used for free up to 1 month. We have collected the summaries of 500 different articles. The articles have been selected to be diverse, so that they cover many different topics. Hence, we have used these 500 summaries as the gold standard of automated summaries

2. Manual Summaries

We have used the data collected from the DUC (Document Understanding conferences) and have collected 500 summaries from this data. These form the gold standard of manual summaries

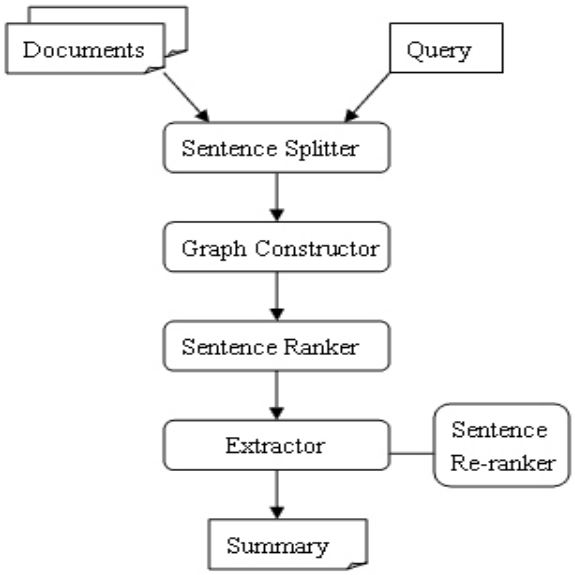
OUR GOLD STANDARD

Summaries generated by a renowned software	500
User Generated summaries	500

For evaluation, we wrote our own program to do an Ngram analysis to calculate precision, recall and FMeasure. Details of which will be explained later.

Methodology

To implement this task, we have split the task into a number of sub tasks. For each sub task, we have employed different features and have tried to see which one performs the best, and analyze why.



The subtasks performed by our summarizer.

Document Preprocessing

- The original document will contain many words, which will not be important in the sentence. It would be obvious to remove the stop words such as "A", "the" etc. from the sentence.
- There are many more sophisticated methods to process the documents, such as retrieving only nouns or nouns and adverbs etc. We will use these features and evaluate which gives the best summaries

Graph Construction

- We then construct a graph from the words that we have filtered from each sentence.
- We assign each sentence a node and calculate the edge weight between each sentence by various similarity functions. There are two methods to calculate similarity between sentences.
    - One is to calculate sentence similarity with the words in sentences.
    - The other is to calculate sentence similarity with stemmed words in sentences.
- We have chosen the original words in the sentences, because the stemmed words would miss the tense and voice information of the sentence.
- Moreover, there are many different Similarity functions which can be used, with each having its own benefits and drawbacks

TextRank

Once, we get the graph with edges, we can apply various techniques, such as HITS or TextRank, to obtain weight for each node.

Following is the formula for TextRank. This weight represents the importance of each node.

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

Summarization

Once we have each sentence and its importance, we can sort the nodes in the order of their weights and display the sentences with most similarity. This will give the summary.

Features used

For evaluation purposes, we have used the following features as baseline

Nouns and adjectives using Rada Mihalcea's similarity algorithm and TextRank to rank the nodes on the graph

In addition, we will be using the following features,

1. Nouns and verbs
2. Nouns, adjectives and Verbs
3. Stop word removal

We will be using the following similarity function to generate different summaries:

1. ***Jaccard Similarity***
   a. It is a statistic used for comparing the similarity and diversity of sample sets.
   b. It uses overlap of words between sentences to calculate similarity

$$Jaccard(s_a, s_b) = \frac{|w(s_a) \cap w(s_b)|}{|w(s_a) \cup w(s_b)|}.$$

2. ***Dice Similarity***
   a. Dice coefficient, is a similarity measure over sets:
   b. It uses overlap of words between sentences to calculate similarity

$$Dice(s_a, s_b) = \frac{2|w(s_a) \cap w(s_b)|}{|w(s_a)| + |w(s_b)|}.$$

*3. **Rada Mihalcea's Similarity Measure***

This uses overlap of words between the two sentences, However, it uses a different normalization factor

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)}$$

We have tried all these features and similarity algorithms till now and have tabulated the results, which will be covered in later sections of this report.

Furthermore, we will be using different ranking algorithms such as *HITS* and *Iterative TextRank*, along with the existing baseline *TextRank* to generate the summaries.

Moreover, our evaluation involves comparison with both human generated summaries and automated summaries.

Our Code and Actual Implementation

We have used java to write the code for this project.
We have used OpenNLP's POS tagger as a syntactic filter to filter the nouns, adjectives and verbs from the document.
To run the code,

- Import the jar file in eclipse.
- Import the folder in which your documents are present.
- Then, run the project and the summaries will be present in the output folder.

Performance Evaluation Script

To validate the summaries, industry wide standard is the Rouge-n measure, which is N-gram analysis of a summary with another benchmark summary. However, since access to ROUGE was not available, we wrote our own N-gram analysis program. This program calculates N-gram precision, recall and F-Measure for n= 1 to 3.
All it needs as input is the path of the reference summaries and path of the summaries, whose accuracy has to be measured

Here is the formula for each -:

Precision	Matching grams in both document / Number of grams in reference document
Recall	Matching grams in both document / Number of grams in our generate summary
FMeasure	2*Precision*Recall/(Precision + Recall)

This program was coded in Java and is attached with this document.

Results

Firstly, we compared our summaries to the gold standard for automated summaries (the summaries from the software Intellexer).

Statistic 1

We evaluated the different features that can be used to preprocess the document. Here are the results of the experiment

Stat	Nouns + Adjective			Nouns + Adj + Verbs			Nouns + Verbs		
N	1	2	3	1	2	3	1	2	3
Precision	**74.4%**	**57.5%**	51.85%	74.1%	56.6%	51%	72%	56%	**52.7%**
Recall	**54.4%**	36.8%	31.9%	54.3%	35.8%	30.9%	48.5%	**37.4%**	**34.1%**
FMeasure	**57.2%**	41.2%	36.5%	57%	40.3%	35.7%	52.7%	**41.3%**	**38.25%**

Table 1: Compares the Precision, Recall and FMeasure for various features. . We have used the average of NA, NV and NAV features for the purpose of this table. So, each value represents the average of 1500 different summaries

Inference from table 1:

- The Nouns and adjectives outperform the other two features for N = 1.
- However, for N = 3, Nouns and verbs perform much better.
- This could be because Nouns and adjectives usually form a main part of the subjects in sentences. Whereas, Verbs focus on the action being done.
- Hence, Nouns and verbs will come together in the summary to give a better summary, with a strong influence of both subject and action, as compared to Nouns and Adjective, which will only focus on the subject.

Statistic 2

We then used various similarity coefficients to generate different summaries. Here are the results of that experiment.

	Rada Mihalcea's			Jaccard			Dice		
N	Precision	Recall	FMeasure	Precision	Recall	FMeasure	Precision	Recall	FMeasure
1	**74.3%**	**54.6%**	**57.3%**	74.1%	51.7%	55.6%	72.1%	50.8%	54.1%
2	**57.1%**	36.3%	40.8%	56.7%	34.5%	39.5%	57%	**39.1%**	**42.4%**
3	51.4%	31.4%	36.1%	51%	29.8%	35.0%	**53.1%**	**35.7%**	**39.3%**

Table 2: Compares the Precision, Recall and FMeasure for various similarity measures. We have used the average of NA, NV and NAV features for the purpose of this table. So, each value represents the average of 1500 different summaries

Inference from table 2

- As we can see, the similarity performance of both Rada Mihalcea and Dice Similarity measures perform very well as compared to Jaccards similarity.

- This is even though, Jaccard and Dice are very similar
  Dice = 2*Jaccard/(1+Jaccard)
- Moreover, we can infer from the table, as the value of N increases, the performance for Dice similarity does not fall down very fast, as compared to other 2 similarity functions.

On the automated summaries, our data performs very well with precision over 50% and FMeasure around 40% for 3 grams.

Statistic 3

We then compared our summaries with the user-generated summaries. Here are the tabulated results

	Rada Mihalcea's			Jaccard			Dice		
N	Precision	Recall	FMeasure	Precision	Recall	FMeasure	Precision	Recall	FMeasure
1	36.5%	**47.9%**	**38.9%**	**37.3%**	44.4%	38%	**37.3%**	44.6%	38.1%
2	**16.9%**	**21.5%**	**17.5%**	16.4%	18.7%	16.2%	16.3%	18.9%	16.2%
3	**11.1%**	**13.9%**	**11.3%**	10.6%	11.8%	10.3%	10.6%	11.9%	10.3%

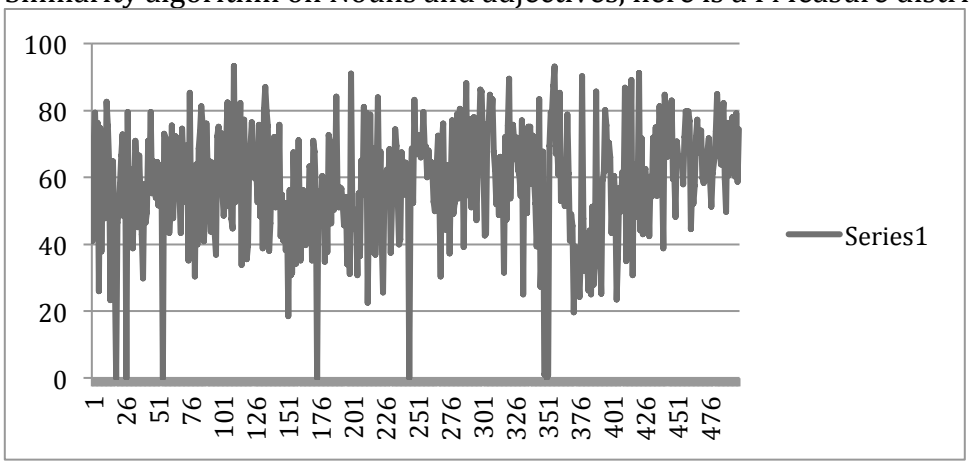
Table 3: Compares the Precision, Recall and FMeasure with human generated summaries. We have used the average of NA, NV and NAV features for the purpose of this table. So, each value represents the average of 1500 different summaries

Inference from table 3:

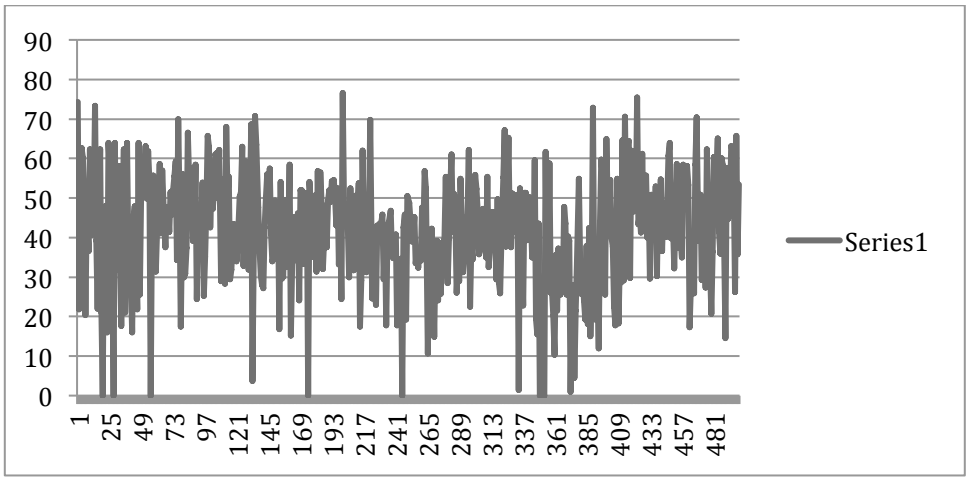
- Firstly, Rada's similarity measure performs better than other two similarity measures
- The FMeasure is significantly lower than the FMeasure obtained from the comparison with Manual Summaries.
- We evaluated the summaries manually and found that the quality of the summaries is not bad. Hence, decrease in FMeasure does not mean that our summaries are not good. So, we further evaluated why the FMeasure is less.
- The evaluation method used by us compares the ngrams in each document. However, if the user generates summary using other words and changing the order of sentences, then any computer-generated summary will score low on FMeasure. For instance, a human can write the summary of a 100-word document by using very less words in common with the base document.
- Hence, we should not compare the FMeasure values obtained with human summaries and computer summaries.
- To obtain a better picture, we will now compute random summaries and show that the FMeasure obtained by our summarizer is higher than other random summaries.

Charts

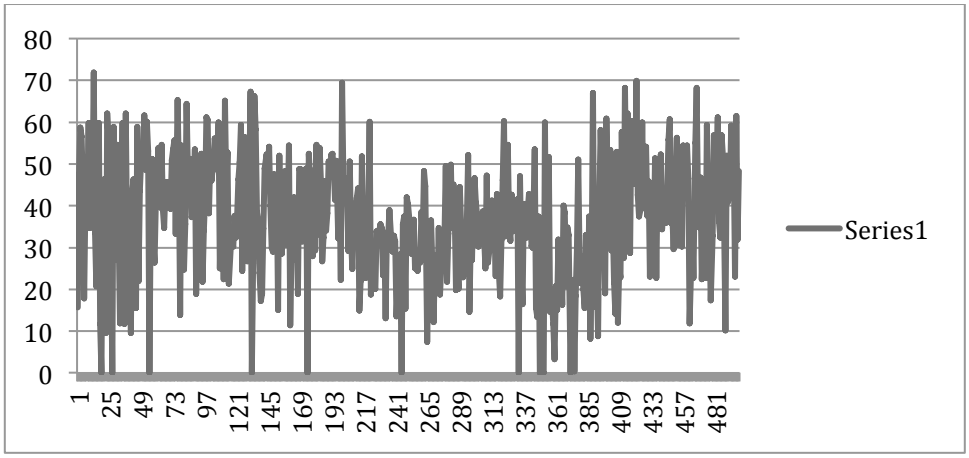
To get an idea of how the average FMeasure varies for 500 documents using Rada's Similarity algorithm on Nouns and adjectives, here is a FMeasure distribution.



Graph 1 FMeasure for 500 docs for Rada Mihalcea with Nouns + Adjectives for N = 1

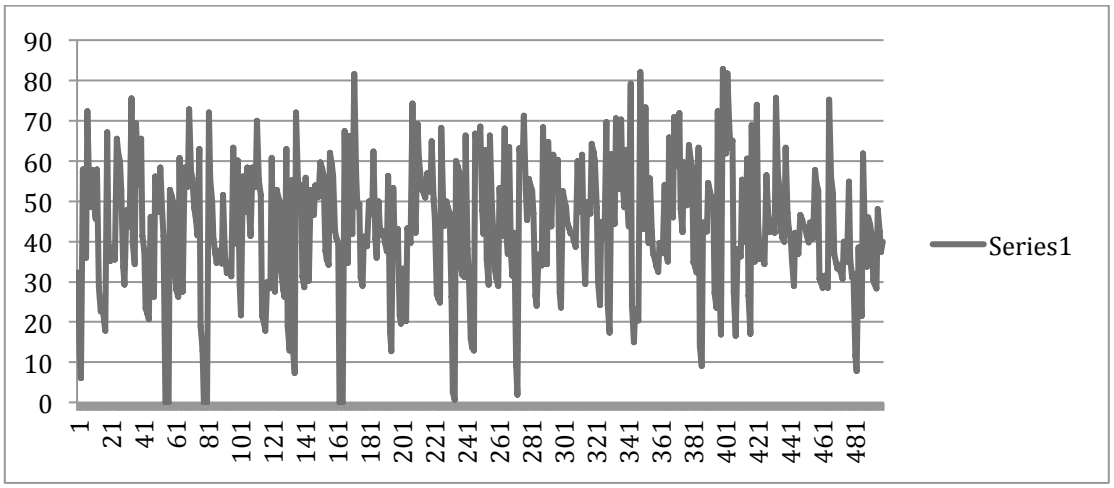


Graph 2 FMeasure for 500 docs for Rada Mihalcea with Nouns + Adjectives for N = 2

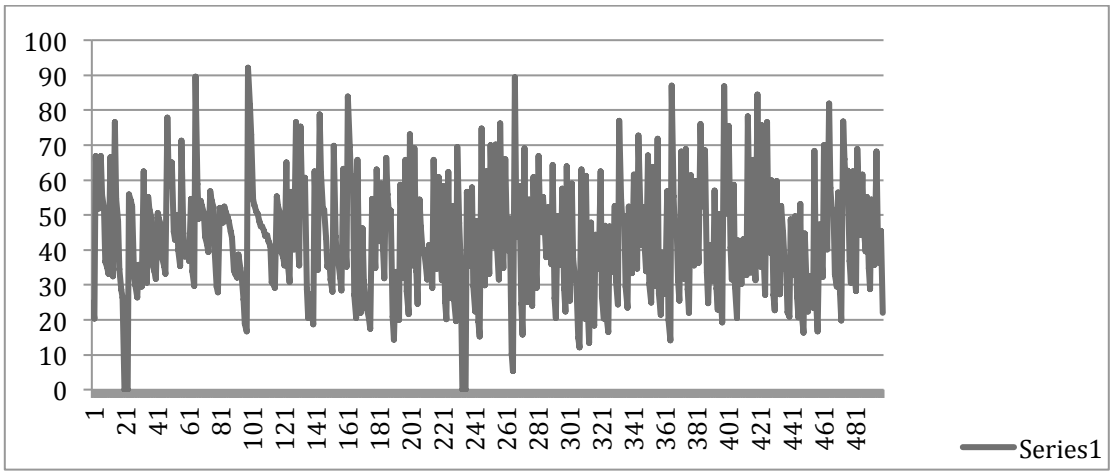


Graph 3 FMeasure for 500 docs for Rada Mihalcea with Nouns + Adjectives for N = 3

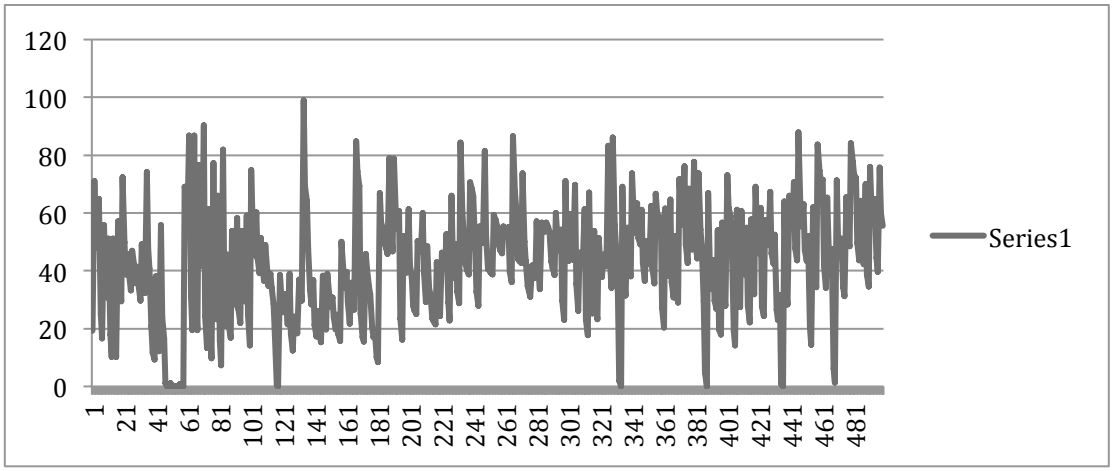
Here is a FMeasure distribution for the Dice Algorithm for Nouns and Verbs. This has shown the best results for N >= 2.



Graph 4 FMeasure for 500 docs for Dice similarity with Nouns + Adjectives for N = 1



Graph 5 FMeasure for 500 docs for Dice similarity with Nouns + Adjectives for N = 2



Graph 6 FMeasure for 500 docs for Dice similarity with Nouns + Adjectives for N = 3

Next Steps

- To add *more similarity measures* such as
  - Cosine similarity
  - Longest Common Subsequence
  - Edit distance
- To use different algorithms for ranking the graphs. There are many existing algorithms, we will use them and modify them to get good results
  - Iterative Text Rank
  - HITS
- Compute random summaries and calculate FMeasures for each and show that the summary generated by us is a good summary.
- Compare the results and analyze which one performs better and why

References

1. Intellexer Summarizer - http://summarizer.intellexer.com
2. A Survey on Automatic Text Summarization, Dipanjan Das and Andre F.T. Martins
3. Scalable Graph-Based Learning Applied to Human Language Technology, Andrei Alexandrescu
4. Calculating Statistical Similarity between Sentences, Junsheng Zhang, Yunchuan Sun, Huilin Wang, Yanqing He
5. Rada Mihalcea and Paul Tarau, TextRank: Bringing Order into Texts,
6. Rada Mihalcea and Hakan Ceylan, Explorations in Automatic Book Summarization
7. Rada Mihalcea Language Independent Extractive Summarization
8. Rada Mihalcea, Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization