TextRank 2.0: Spanish Text Summarization using Keyword Extraction

Aparna Anand
USC ID: 6420332289
aparnaan@usc.edu

Avik Basu
USC ID: 2050987089
avikbasu@usc.edu

Indhu Kamala Kumar
USC ID: 5391099912
kamalaku@usc.edu

Madhusudhan Krishnamachari
USC ID: 7332185833
madhusuk@usc.edu

1. INTRODUCTION

The problem of automatic text summarization is one that has garnered significant interest in recent years. Humans want to simplify their lives as much as possible. They want to read a short gist of a news article before deciding whether or not they want to spend time reading the main article. In particular, presidential speeches can be really long. Furthermore, it could be time consuming to analyze the progress and growth of a country during a president's period from his speech. Hence, there is a need to design algorithms that can give a short summary for presidential speeches. Automatic document summarization can be done in two ways. The first is *abstractive summarization*, which 'understands' the source document and paraphrases it into a more concise one. This task involves language understanding and is significantly harder than *extractive summarization*, which identifies important words/phrases from the source document and creates the summary.

In the past two decades, a lot of algorithms have been proposed for extractive text summarization. Gong and Liu[3] ranked sentences using standard Information Retrieval techniques and used Latent Semantic Analysis to identify semantically important sentences. Erkan and Radev[2] proposed a stochastic graph-based model of sentences to compute the importance of textual units of a document. They used this to calculate the importance of each sentence in a document which helped in summarization. The TextRank method[7] proposed by Mihalcea and Tarau uses a graph-based ranking model which is used for keyword and sentence extraction. More recently, Kageback et al[5] used continuous vector space models for extractive summarization and Xu et al[11] used word features obtained from Wikipedia which helped in summarizing articles.

In this project, we propose an approach to summarize Spanish documents using extractive summarization techniques. We experiment with the TextRank algorithm by incorporating our new ideas and propose a new and improved version of the TextRank algorithm to perform keyword extraction and then use these keywords to extract sentences from the document. The system is evaluated using the standard text summarization measure ROUGE[6], by comparing with various reference summaries generated using existing text summarization algorithms.

2. METHOD

2.1 Materials

The corpus is a transcript of speeches by two Colombian presidents in the Colombian dialect of Spanish, which has been compiled by David Przybilla[9]. The first section contains 250 speeches delivered by Colombian President Alvaro Uribe between 2007 and 2010. The second section contains 391 speeches delivered by President Juan Manuel Santos between 2010 and 2013. Each document contains a title, date, URL of a web source relating to that speech and the transcript of the speech. The corpus contains raw data without any annotation. For this task of text summarization, no annotation was necessary and the raw data is used as it is.

2.2 Procedure

The main task our project performs is language independent text summarization and we achieve this with a core algorithm that performs keyword extraction followed by sentence extraction. Once the keywords are extracted, the sentences in which these keywords primarily occurred are deemed important and thus used in the summary.

The first stage entailed pre-processing of the text to obtain raw words which are used as features, in each document. For this purpose, we removed stop words, performed word stemming and used NLTK's[1] sentence tokenizer of Spanish text. The second stage implemented the core algorithm, for which we used our modified version of the graph based algorithm, TextRank. Typically, for the task of text summarization, TextRank is used to perform sentence extraction directly. It translates a document into a graph with sentences as nodes and similarity between the sentences as edges, calculates a PageRank[8] score for each node and finally picks m nodes, i.e. sentences, with the highest scores to formulate the summary. We introduced a few changes in an attempt to improve the existing TextRank algorithm. First, we used the words in a document as the nodes of the graph, as opposed to sentences, to perform keyword extraction. The edges between the nodes are set using a window which defines the co-occurance relation between the nodes. The window size is set to 2, making it a bigram model. Second, we introduced weights for each edge to make it a weighted graph. The weights were a weighted average over the nodes' positional distribution, calculated using the arc sine probability density function which is given by

$$f(x) = \frac{1}{\pi\sqrt{x(1-x)}} \quad (1)$$

where x is the normalized value of the position of a particular word.

This was based on an intuition that the beginning and end of the text have the important sentences and keywords we require to form the summary.

A sample document is shown below.

Alguien tiene lo que usted busca. Y alguien busca lo que usted tiene.

Somebody has what you are looking for. And someone looking for what you have.

A sample graph for the above document with window size of 2 is shown below.

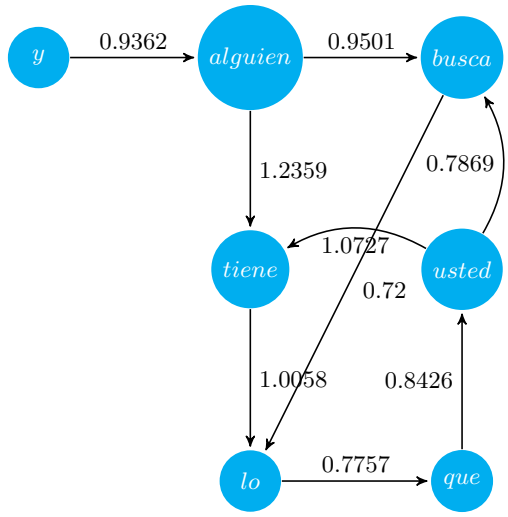

Figure 1: A sample graph used in TextRank v2.0

The calculation of the weight of an edge is explained as follows using the example of the edge between y and *alguien* in the above graph:

- Step 1: The normalized position values are identified for each node. *alguien* occurs at positions 1 and 8. It's normalized position values are $\frac{1}{14}$ and $\frac{8}{14}$.
- Step 2: Each node's f values are calculated for each of its normalized position value using the arc sine probability distribution function and the maximum f value seen is taken as the node's final weight. *alguien* obtains an f value of 1.2359 and y gets an f value of 0.6366.
- Step 3: For each edge, weighted average of the nodes it connects is taken. So the edge between y and *alguien* takes a weight of: $\frac{1.2359+0.6366}{2} = 0.9362$.

After the construction of this graph, we calculated the PageRank score for each node iteratively either till convergence of values or till a maximum iteration count is reached.

The number of keywords extracted depends on the word count of the document and is calculated as follows:

$$n = min(0.1 * size(wc_d), c * ln(size(wc_d)) \quad (2)$$

where n is the number of keywords extracted, wc_d is the word count of the document and c is an empirical constant. For this experiment, $c = 7$ was used. The natural logarithm function was used as it grows slowly and this helps to maintain the value of n under control for very large documents and at the same time, extracts a good amount of keywords for smaller documents.

Last, we introduced our own algorithm to obtain the final sentences from the n extracted keywords. This was done by calculating a sentence score for each sentence, which was based on: a) the positional weight of the sentence - using the arc sine probability density function, b) the sum of PageRank scores of the extracted keywords present in the sentence.

The sentence score is calculated as follows

$$SC_s = P_s * \sum_{k \epsilon K, k \epsilon S} PR_k \quad (3)$$

where SC_s is the sentence score, P_s is the positional probability of sentence S, K is the list of extracted keywords and PR_k is the PageRank score for keyword $k \epsilon K$.

The algorithm picked the m high scoring sentences and ordered them based on their position in the text, to formulate the summary. This m value is experimentally set and for this experiment, $m = 10$. We used this value as we felt that text summaries should not be longer than 10 sentences as we have to maximize the information content of the summary while keeping the summary as short as possible.

2.3 Evaluation

ROUGE is a recall-based evaluation metric for determining the quality of text summarization systems. It stands for *Recall-Oriented Understudy for Gisting Evaluation*. ROUGE has four different types of measures namely ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-S.

We are using the ROUGE-N measure to evaluate our system. It is a recall based N-gram measure between a candidate summary and a set of reference summaries. ROUGE-N is given by:

$$ROUGE-N = \frac{\sum_{S \epsilon R} \sum_{gram_n \epsilon S} Count_{match}(gram_n)}{\sum_{S \epsilon R} \sum_{gram_n \epsilon S} Count(gram_n)} \quad (4)$$

where R is the set of reference summaries, n is the length of the n-gram, $gram_n$ and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries.

Since we are extracting and generating summaries from keywords, an n-gram based statistic for evaluation makes sense.

A text summarization system is good if it agrees with more reference summaries. Therefore, by increasing the number of reference summaries, a candidate summary can be compared with a wider range of summaries. This can potentially increase the ROUGE-N score, thereby indicating that the generated summary is of top quality. Since we don't have access to human generated summaries, our system is compared with reference summaries generated from existing text summarization algorithms namely, Latent Semantic Analysis, LexRank, SumBasic[10], and KL-Sum[4].

3. RESULTS

The summaries generated from our algorithm and the original version of the TextRank algorithm were compared. The ROUGE-N score was computed for the output given by our algorithm, with reference to the reference algorithms mentioned in the previous section. The same was done for the output given by the original TextRank algorithm. The results are shown below.

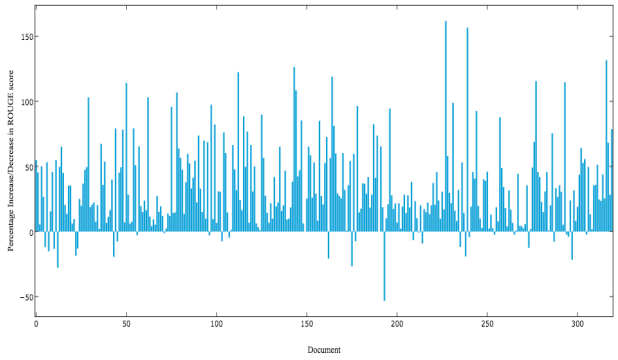

Figure 2: Graph showing % increase/decrease in ROUGE-N score for N=1

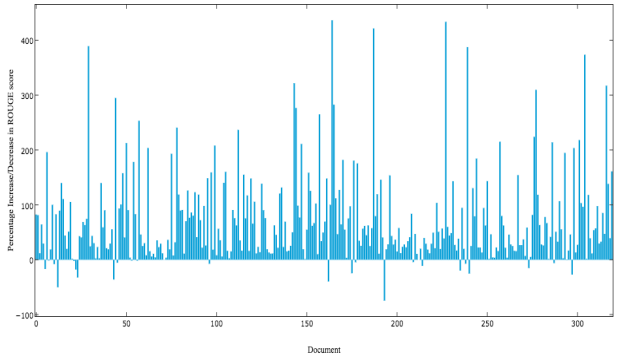

Figure 3: Graph showing % increase/decrease in ROUGE-N score for N=2

The above graphs show that TextRank 2.0 outperforms the original algorithm in most of the cases. This is attributed to using the keywords to extract the sentences from the document rather than using the sentences themselves as nodes in the graph.

	Avg ROUGE-N score		Avg % increase
	TextRank v1.0	TextRank v2.0	
N=1	0.3690	0.4626	32.4%
N=2	0.2619	0.3686	69.92%

Table 1: Average ROUGE-N scores

Table 1 table gives further evidence to prove that TextRank 2.0 works better than the original algorithm.

It was observed that the TextRank 2.0 runs much faster than the original TextRank algorithm. The new algorithm is 40-50% faster than the original algorithm and this shows the efficiency of the proposed algorithm.

4. DISCUSSION

TextRank 2.0 is a language and domain independent algorithm, which improves upon the original TextRank algorithm. Overall, the proposed system performs better on several grounds in comparison to the original system.

The scores obtained from the evaluation show that the ROUGE-N score of TextRank 2.0 has an average percentage increase of 32.4% for N=1 and 69.92% for N=2. This is justified by the inclusion of weights in the graph and our well designed algorithm for sentence extraction from the keywords generated.

We also proposed a heuristic to calculate the value of n, which is the number of keywords to be extracted. This is a function of the document word count. This makes sure that the number of keywords extracted is neither too big nor too small. This also improves the system speed in terms of summarizing the document as we only take into consideration the top n keywords. The usage of words instead of sentences as nodes further augments to the improvement in speed as the original algorithm, unlike the proposed algorithm, operates on a fully connected graph.

The improved algorithm performs keyword extraction as opposed to sentence extraction for the task of text summarization. This identification of the top n keywords of the document as an initial step helps use the keywords to potentially perform other tasks in addition to the summarization task, such as search engine indexing, headline generation for news documents, document categorization, etc.

Text summarization systems have potential to do much more than just generating summaries. With more intelligence and data, they can be used to perform relevant tasks such as generating headlines for news articles and creating abridged versions of books simultaneously. One future direction that we would like to explore is the use of Recurrent Neural Networks and Restricted Boltzmann Machines to generate headlines for news articles using the extracted keywords. Annotating the corpora with Part-of-Speech tags could also help in headline generation wherein the keywords would be used to

create a new headline from scratch and this is an interesting avenue to explore in the future. As another improvement to the algorithm, we would like to experiment with the window size which would modify it from a bi-gram model to an n-gram model.

5. REFERENCES

[1] S. Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[2] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, pages 457–479, 2004.

[3] Y. Gong and X. 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.

[4] A. Haghighi and L. Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics, 2009.

[5] M. Kågebäck, O. Mogren, N. Tahmasebi, and D. Dubhashi. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pages 31–39, 2014.

[6] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. 2004.

[7] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. Association for Computational Linguistics, 2004.

[8] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[9] D. Przybilla. Latin american text resources. `https://github.com/dav009/LatinamericanTextResources`, 2013.

[10] L. Vanderwende, H. Suzuki, C. Brockett, and A. Nenkova. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6):1606–1618, 2007.

[11] S. Xu, S. Yang, and F. C.-M. Lau. Keyword extraction and headline generation using novel word features. In *AAAI*, 2010.

6. DIVISION OF LABOR

- Aparna Anand : Core Algorithm, Data, Theory, Writing
- Avik Basu : Preprocessing and Evaluation, Data, Theory, Writing
- Indhu Kamala Kumar : Core Algorithm, Data, Theory, Writing
- Madhusudhan Krishnamachari : Preprocessing and Evaluation, Data, Theory, Writing

7. WORD COUNT

The word count is 1980.