Nutshell: A Basic Topic-focused Multi-document Summarization System

Shannon Ladymon Haley Lepp Benny Longwill Amina Venton
Department of Linguistics, University of Washington
{sladymon,hlepp,longwill,aventon}@uw.edu

Abstract

We design and implement Nutshell, an automatic topic-focused multi-document summarization system. Using newswire data, we test a modified version of the Biased LexRank graph algorithm to calculate sentence salience using inter-sentential similarity and topic-sentence bias. The ranked sentences are then selected greedily based on this score while avoiding redundancy and remaining within the 100 word limit. We find that our ROUGE scores are slightly more competitive than the LEAD baseline. In future work, we will explore whether stemming, tuning of hyperparameters, variations on weighting formulas, and more complex information ordering and content realization subcomponents can improve performance.

1 Introduction

In this study, we create a base end-to-end automatic summarization system that takes in topic-oriented document clusters from ACQUAINT and ACQUAINT2, and produces 100-word extractive summaries for each topic cluster. Nutshell makes use of a modified version of the Biased LexRank graph approach to determine sentence salience using inter-sentential similarity weighting with bias for cluster topic (Erkan and Radev, 2004; Otterbacher et al., 2005, 2009). Once ranked, sentences are selected greedily with constraints to eschew redundancy and ensure correct summary length.

We evaluate the output summaries against human-produced summaries from the TAC 2010 shared task evaluation using ROUGE-1 and ROUGE-2 metrics. We find that Nutshell outperforms the LEAD baseline, although it is slightly under the MEAD baseline.

2 System Overview

Nutshell follows a straight-forward end-to-end pipeline. First, using the input document clusters, it pre-processes XML files. To select content, Nutshell uses a modified version of Biased LexRank to score sentences by salience and relevance to the cluster. Ranked by score, sentences are then filtered for redundancy and selected iteratively until the summary reaches a maximum of 100 words. This summary is then passed through information ordering, which orders the sentences in the summary reverse chronologically (most recent news articles first). Finally, the summary undergoes content realization, which currently does not make any changes, and then is directly output and evaluated using ROUGE metrics.

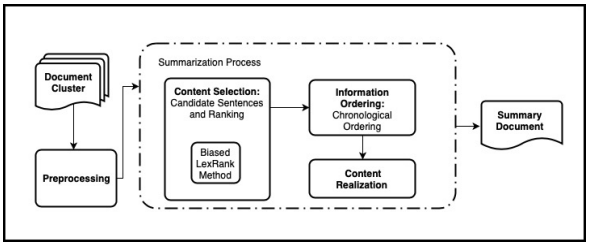

Figure 1: System Architecture

3 Approach

Below, we describe our approach in four subcomponents: data preparation, content selection, information ordering, and content realization.

3.1 Data preparation

We collect data from a set of NIST standard XML files that identify a set of 44 training topics. We parse these using the Beautiful Soup Python library (Nair, 2014). We use NLTK (Loper and Bird, 2002) to instantiate Sentence and Token Class Objects, removing stopwords using the

NLTK.corpus stopwords list.

3.1.1 Document Retrieval

A Document Retriever object extracts raw document HTML or XML. We re-configure the source and publication date from the document IDs to derive a file path to the respective document corpora. Depending on the year of production, we assemble the file path for the ACQUAINT or ACQUAINT2 corpora and make necessary modifications to adhere to file path naming conventions.

3.2 Content Selection

Our system selects content from the corpus via a modified version of the Biased LexRank graph approach to determining sentence salience, which is based on a PageRank style of inter-sentential similarity weighting with an additional bias for the query topic. In the Biased LexRank method, each node in the similarity graph represents a sentence in the topic document cluster (only sentences of 5 or more words are included), and the edges between nodes represent the weight of their inter-sentential similarity. This approach allows sentences to be ranked based on both their relation to the topic and to their centrality in the topic document cluster.

3.2.1 tf-idf Cosine Similarity

We calculate topic-sentence and inter-sentential similarity using tf-idf cosine similarity. In this formula:

$$\text{tf}_{w,s} = \frac{f_{w,s}}{F_s}$$

where $f_{w,s}$ is the count of words w in sentence s, and F_s is the total number of all words in s, and

$$\text{idf}_w = log(\frac{N}{1+n_w})$$

where N is the number of sentences in the topic, and n_w is the number of sentences where word w occurs. As such, td-idf is:

$$\text{tf-idf}_{w,s} = tf_{w,s} \times idf_w$$

We calculate the similarity between two sentences, x and y, using the cosine similarity of these tf-idf values:

$$\text{cos\_sim}(x,y) = \frac{\sum_{w \in x,y} (\text{tf-idf}_{w,x}) \times (\text{tf-idf}_{w,y})}{\sqrt{\sum_{x_i \in x} (\text{tf-idf}_{x_i,x})^2} \times \sqrt{\sum_{y_i \in y} (\text{tf-idf}_{y_i,y})^2}}$$

For inter-sentential cosine similarity, we use a minimum threshold of 0.15 (as in Otterbacher et al. (2005)) to ensure that edge weights are only added when a reasonable amount of similarity is found.

3.2.2 Biased LexRank and Power Iteration

We use cosine similarity to calculate both the topic-sentence similarity bias (note that this differs from the relevance formula used originally in Otterbacher et al. (2005)) and inter-sentential similarity for each sentence. The Modified Biased LexRank score for a sentence s given query topic q is thus:

$$\text{MBLR}(s|q) = d \frac{\text{cos\_sim}(s,q)}{\sum_{z \in C} \text{cos\_sim}(z,q)} + (1-d) \sum_{v \in C} \frac{\text{cos\_sim}(s,v)}{\sum_{z \in C} \text{cos\_sim}(z,v)}$$

where d is the weight to give to the topic bias (higher values of d favor the bias over the inter-sentential similarity) and C is the set of sentences in the topic document cluster.

This Modified Biased LexRank formula can be translated into a Markov Matrix, where A is the bias vector of cosine similarity between sentences and the topic, B is the inter-sentential similarity matrix:

M = [dA + (1-d)B]

With this Markov Matrix, we implement the power iteration method to find the final Modified Biased LexRank values for each sentence. Beginning with p_0 as a uniform probability distribution, we iterate over the following formula until convergence (with an epsilon of 0.1), updating p at each step:

$$p_t = M^T p_{t-1}$$

This process gives a final set of scores, p_t for ranking sentence salience.

3.2.3 Sentence Selection

We use a greedy iterative approach to choose sentences for the summary. Given a set of sentences sorted by their Modified Biased LexRank scores, we iteratively select the highest-ranked sentence

that fits within the 100 word requirement. We filter out sentences with a cosine similarity greater than or equal to 0.5 (as in Otterbacher et al. (2009) with any of the other selected sentences to eliminate potential redundancy.

3.3 Information Ordering

We order the chosen sentences for each topic reverse chronologically by the date associated with their original documents in order to prioritize the most recent news articles. Currently, sentences within the same document do not have a specific order.

3.4 Content Realization

The current implementation of the Nutshell system copies the ordered sentences for each topic from the Information Ordering sub-component.

4 Results

In order to evaluate the performance of the Nutshell system, we ran it on the Text Analysis Conferences (TAC) 2010 shared evaluation task, which is a topic-focused multi-document text summarization task of news article document clusters from ACQUAINT and ACQUAINT2 with human-generated gold model summaries. We used two metrics for evaluation, ROUGE-1 (unigram) average recall and ROUGE-2 (bigram) average recall, and compared against the LEAD and MEAD baselines for ROUGE-2.

	R-1	R-2
LEAD baseline	–	0.05376
MEAD baseline	–	**0.05927**
Nutshell	0.22720	0.05710

Table 1: ROUGE average recall scores on TAC 2010.

Although there are no ROUGE-1 baselines to compare Nutshell's performance to, Table 1 shows that Nutshell's ROUGE-2 performance is somewhere in the middle of the LEAD and MEAD baselines. Nutshell outperforms the LEAD baseline in R-2 by 0.00334, while the MEAD baseline outperforms Nutshell by 0.00217.

5 Discussion

Overall, Nutshell performs topic-focused extractive text summarization well given the simplicity of its approach, outperforming the LEAD baseline. Its use of tf-idf based cosine similarity is a straight-forward approach to determining the surface similarity between sentences, and the similarity graph Modified Biased LexRank model allows Nutshell to account for the relevance of both sentences within a document cluster as well as their connections to the topic itself.

Many aspects of the Nutshell system can be further refined, which will likely lead to an improvement in performance. In the pre-processing stage, stemming words might allow for a better assessment of surface similarity. Many of the hyperparameters used to run Nutshell, such as d and the cosine similarity thresholds, can be tuned to refine their values rather than relying on the settings used in the original LexRank papers (Erkan and Radev, 2004; Otterbacher et al., 2005, 2009). Many aspects of the formulas for Modified Biased LexRank can also be tuned by looking at the effect of using different variations of how to calculate tf-idf, the topic-query bias, and the weighting of inter-sentential edges in the similarity graph. Additionally, implementing more complex versions of the Information Ordering and Content Realization sub-components will likely lead to better summaries and performance.

6 Conclusion

Nutshell is a topic-focused multi-document extractive text summarization system that performs well while using a simple statistical model that can work on even small data sets. Nutshell preprocesses data by tokenizing sentences and removing stopwords and to build a tf-idf language model based on words in sentences in a topic document cluster. It implements content selection using the power iteration method and a modified version of the Biased LexRank similarity graph approach to sentence salience, which calculates weights for both inter-sentential similarity and topic-sentence bias using tf-idf based cosine similarity. Summaries are built using the scores produced by this method in a greedy iteration approach that incorporates the 100 word limit and redundancy constraints, and are additionally organized in reverse chronological order to prioritize more recent news articles.

The summaries that Nutshell produces outperform the LEAD baseline in TAC 2010 ROUGE-2 metrics, although it slightly underperforms the

MEAD baseline. Modifications to the Nutshell system, such as using stemming, tuning hyperparameters, exploring variations on weighting formulas, and adding more complex information ordering and content realization sub-components will likely improve performance and are areas of future work.

References

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Vineeth G. Nair. 2014. *Getting Started with Beautiful Soup*. Packt Publishing.

Jahna Otterbacher, Gunes Erkan, and Dragomir Radev. 2005. Using random walks for question-focused sentence retrieval. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 915–922, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Jahna Otterbacher, Gunes Erkan, and Dragomir R. Radev. 2009. Biased lexrank: Passage retrieval using random walks with question-based priors. *Inf. Process. Manage.*, 45(1):42–54.