Newsjam – Report 2

Shane Kaszefski-Yaschuk **Joseph Keenan** **Adelia Khasanova**
Maxime Méloux **Mahamadi Nikiema**

Institut des Sciences du Digital
Pôle Herbert Simon
13, rue Michel Ney
54000 Nancy

Abstract

In the past month, we have created a minimum viable product for our text summarization project. We implemented two extractive summarization methods, Latent Semantic Indexing (LSI) and K-means clustering. These methods were tested on two different corpora and evaluated using ROUGE and BERTScore metrics. Our scores from these evaluation metrics are variable, giving us a base from which we can compare and improve our project going forward.

1 Introduction

For our project, we are attempting to create a text summarization tool for COVID-19 articles in France. Our thinking behind this topic was that by summarizing all of these articles, people could quickly get an idea of the current situation without having to rummage through the plethora of available information.

2 Literature review

Automatic text summarization refers to the generation of a condensed version of the original text while preserving key information. There are two primary text summarization methods — extractive and abstractive (Saggion and Poibeau, 2013). Extractive summarization centers around identifying a few key sentences in the text and putting them together verbatim, whereas abstractive summarization involves generating novel sentences. Concrete extractive approaches include assigning importance scores to sentences, utilizing topic representation, k-means clustering and latent semantic indexing (Allahyari et al., 2017). Other approaches primarily seen in abstractive text summarization include the use of deep learning, RNN encoder-decoder architecture, gated Recurrent Neural Networks, and Long Short-Term Memory (Suleiman and Awajan, 2020).

Text summarization comes with a couple of key challenges, quality of reference summaries and out-of-vocabulary words (OOV) (Suleiman and Awajan, 2020). During the testing stage, reference summaries are needed for evaluation. However, datasets often contain poor reference summaries or do not contain them at all, making evaluation unreliable or impossible. Another challenge is the occurrence of OOV words that are absent from the training dataset but are central to understanding a document. This is particularly problematic to models that use word embeddings, because without an embeddding, an OOV word cannot be compared to other words in the document.

3 Methodology

3.1 Scraping

We have selected seven French news sites for this project. Since each site has a different underlying structure, a custom scraper is required for each. Our project currently contains two working Python scrapers: one for *Actu*[1], using the *selenium*[2] library, and the other for *L'Est Républicain*[3] using the *beautifulsoup*[4] library.

For each website, we only retrieve articles with the "COVID-19" label or its closest equivalent (such as "Health"). Summaries are also extracted from each article when available.

The output of these scrapers is the same in both cases: a JSON-formatted file containing the exact same fields as the MLSUM corpus (Scialom et al., 2020) which we use to evaluate our model on a large dataset (see 4.1).

[1] https://actu.fr/societe/coronavirus

[2] https://pypi.org/project/selenium/

[3] https://estrepublicain.fr/sante/coronavirus

[4] https://pypi.org/project/beautifulsoup4/

3.2 Summarization

Our project currently implements the following two extractive approaches for summarization:

3.2.1 Latent Semantic Indexing

Latent Semantic Indexing (LSI) or Analysis (LSA) is a technique initially introduced for automatic document classification (Borko and Bernick, 1963) and information retrieval (Deerwester et al., 1990), but which was later found to be efficient for automatic text summarization (Gong and Liu, 2001; Ozsoy et al., 2011) and its evaluation (Steinberger et al., 2004; Steinberger and Jezek, 2009).

Our algorithm replicates that of Gong and Liu (2001), including for sentence selection. Our implementation was initially based on Chakravarthy (2020) as a starting point but has since largely diverged. We compute the optimal number of topics by measuring C_v topic coherence as described in Röder et al. (2015). The final summary is generated by concatenating the chosen sentences by order of decreasing score until we reach a summary length of 280 characters.

We decided to apply LSI not to the raw text, but to a list of keywords generated by removing punctuation and stopwords from the article and then stemming the remaining words. Our intent in doing so is to eliminate noise that could be caused by stopwords and to apply topic modeling to the most relevant words.

3.2.2 Word embeddings and k-means clustering

K-means clustering is an alternative way to model topics within a document, which has successfully been applied to text summarization before (Shetty and Kallimani, 2017).

Our implementation does not use TF-IDF (term frequency-inverse document frequency) as the raw input of k-means clustering, but rather contextual word embeddings (Gupta, 2020; ialifinaritra, 2021). Those word embeddings are generated on a sentence basis using the pre-trained models FlauBERT (Le et al., 2019) and CamemBERT (Martin et al., 2020). We arbitrarily choose $n_{clusters} = 5$ for k-means clustering.

We then use the same algorithm as in 3.2.1 for selecting sentences and building the final summary.

4 Experimental setup

4.1 Dataset

Our own corpus, created via scraping news websites, currently contains 47 articles extracted from *Actu* and 1,753 articles from *L'Est Républicain*. The *L'Est Républicain* dataset does not yet contain summaries for articles and is therefore only used for demonstration purposes.

In addition to our corpus, we also used the MultiLingual SUMmarization corpus (MLSUM) (Scialom et al., 2020). The French version of the MLSUM corpus is made up of over 400,000 news articles from *Le Monde*, split into train, test, and validation sets. The title and content of each article are stored along with a corresponding summary for evaluation, which usually comes from a highlights section found at the beginning of each article.

We chose this dataset to evaluate this early version of our program due to its semantic proximity with ours (recent news articles, in French, extracted from online newspapers) as well as the large amount of data it contains, which we believe will give us more reliable scores than our own, currently smaller dataset.

Our models are therefore currently evaluated on the testset of the French MLSUM, which contains 15,828 articles, as well as our *Actu* dataset.

4.2 Evaluation

To evaluate our models we have chosen to use ROUGE (Lin, 2004) due to its popularity in NLP, as well as the much more recent, state-of-the-art BERTScore (Zhang et al., 2020), which both calculate precision, recall, and F1-score metrics via two different means and are introduced below. For our evaluation, we are focusing on the F1-score, which is calculated on two different forms of our data:

- The "long" score is computed on pairs consisting of the generated summary and the reference summary.
- The "keyword" score is computed on pairs consisting of keyword-only versions of the generated and reference summaries[5]. The way keywords are extracted is described in 3.2.1. In adding this keyword evaluation, our thinking was that it might be more accurate since scores cannot be inflated by stopword similarity between the two summaries.

[5] The keyword version of the BERTScore has yet to be implemented.

Method	Long ROUGE-L F1	Keyword ROUGE-L F1	Long BERTScore F1
MLSUM corpus, testset (15 828 articles)			
LSI	0.1507	0.1147	TBA
FlauBERT + k-means	TBA	TBA	TBA
CamemBERT + k-means	TBA	TBA	TBA
Actu corpus (47 articles)			
LSI	0.5391	0.5309	0.5666
FlauBERT + k-means	0.2911	0.2862	0.2098
CamemBERT + k-means	0.2463	0.2290	0.2941

Table 1: Evaluation results

4.2.1 ROUGE-L

The ROUGE-L score (Lin, 2004) is a specific version of ROUGE using Longest Common Subsequences (LCS). This algorithm tries to find all of the common sequences of words between a generated summary and a reference summary. Scores are then computed based on the length and number of the LCS in a pair of summaries (Steinberger and Jezek, 2009).

4.2.2 BERTScore

BERTScore (Zhang et al., 2020) is a metric originally created for text generation and translation, but due to it being based on comparison of a generated and a reference text, it is also useful for evaluating summarization. We use it as our second evaluation metric since it evaluates the text on a deeper level by using contextual word embeddings, such as BERT (Devlin et al., 2019), to compare the semantic content between the summaries rather than comparing n-grams like in many versions of ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002). The BERTScore measures are computed via cosine similarity.

5 Results

Our results are summarized in Table 1.

We generally observe much higher scores on the *Actu* corpus than the *MLSUM* one. We also find that the LSI model strongly outperforms the k-means clustering implementation.

Due to the small size of the *Actu* corpus, the high scores of our model on it should not be taken too seriously. On the other hand, even though the MLSUM dataset has some issues that will be discussed later in this section, we believe that due to its size, the evaluation on this corpus better reflects the performance of our models.

By checking the output of our LSI model on the *Actu* corpus, we found that the summary selection was quite satisfactory overall, meaning that the summaries chosen by our model outlined the articles well and matched a quality reference summary. In the MLSUM LSI examples, we noticed a recurring issue where we felt that our summary matched the article, but received a low score due to a poor reference summary. This seems like an inevitable issue when working with a corpus of this magnitude, but it is important to note because it reflects a way in which the scores may not always reflect the quality of a generated summary. Finally, one issue that has affected both datasets is poor summary selection, which is due to the performance of the LSI or k-means clustering models themselves.

6 Conclusion

We have implemented two different extractive summarization methods and laid the foundations of a pipeline for scraping, processing and summarizing French online news articles. The results obtained by evaluating those methods on a large and a smaller, custom corpus using two widespread metrics show promising results for the continuation of the project.

One of the next steps for scraping is improving scraper automation (regularly looking for new articles). Additionally, we plan on using document similarity to avoid the inclusion of multiple articles with similar content.

We would also like to improve the overall performance of our current two summarization methods, LSI and K-means clustering. This is an open research question that will involve looking into all of the aspects of our models.

Lastly, we expect to add more evaluation metrics to get a better overall idea of the performance of our models, since the more evaluation metrics we have, the more scores we have to analyze.

References

Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. Text summarization techniques: A brief survey. In *Proceedings of arXiv, USA*, pages 1–9.

Harold Borko and Myrna Bernick. 1963. Automatic document classification. *Journal of the ACM (JACM)*, 10(2):151–162.

Srinivas Chakravarthy. 2020. Document Summarization Using Latent Semantic Indexing.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yihong Gong and Xin Liu. 2001. Creating generic text summaries. In *Proceedings of Sixth International Conference on Document Analysis and Recognition*, pages 903–907.

Akanksha Gupta. 2020. Understanding Text Summarization using K-means Clustering.

ialifinaritra. 2021. French Text Summarization. Original-date: 2020-10-07T21:22:46Z.

Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. 2019. Flaubert: Unsupervised language model pre-training for french. *CoRR*, abs/1912.05372.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Makbule Ozsoy, Ferda Alpaslan, and Ilyas Cicekli. 2011. Text summarization using latent semantic analysis. *J. Information Science*, 37:405–417.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. *WSDM 2015 - Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, pages 399–408.

Horacio Saggion and Thierry Poibeau. 2013. Automatic text summarization: Past, present and future. In *Multi-Source, Multilingual Information Extraction and Summarization*, pages 3–21. Springer, Berlin, Heidelberg.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. Mlsum: The multilingual summarization corpus. *arXiv preprint arXiv:2004.14900*.

Krithi Shetty and Jagadish S. Kallimani. 2017. Automatic extractive text summarization using k-means clustering. In *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pages 1–9.

Josef Steinberger and Karel Jezek. 2009. Evaluation measures for text summarization. *Computing and Informatics*, 28:251–275.

Josef Steinberger, Karel Jezek, et al. 2004. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM*, 4:93–100.

Dima Suleiman and Arafat Awajan. 2020. Deep learning based abstractive text summarization: Approaches, datasets, evaluation measures, and challenges. *Mathematical Problems in Engineering*, 2020:1–29.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert.