Knowledge-based Extractive Text Summarization
-Final Report-

Andrei Fokin Teixeira
Esmaail Albarazi
Affiliations: Engineering and Data Science Department - Coimbra University
afokin@student.uc.dei.pt
esmaail.albarazi@student.uc.pt

1. Introduction

Text summarization is a natural language topic in which the model is given the original text and gives back the output as a reduced version of the original text. It can be done in an extractive-based approach, picking up the important phrases and sentences from the original text and generating the summary, or in an abstractive-based approach, when an overview of the main concepts is generated from zero, looking to imitate human language.

The extractive approach was developed with simpler models such as TF-IDF and TextRank (TR), and with the advance of language models, especially large language models (LLMs), it has been possible to perform abstract summarization more effectively. However, since abstract summarization does not directly use complete extracts from the original text, it is subject to the risk of hallucinations, which is why knowledge injection techniques have been used to reduce them and perform summarization more factually.

Although the introduction of Knowledge Graphs (KGs) information is usually applied in the context of abstractive summarization to help reduce hallucinations, the interest of this project is to test an application of KGs in a simpler extractive summarization model, TR. Given that the abstractive approach has the challenges of rewriting and restructuring, which increases the complexity of the techniques and the processing of the machines and that extractive summarization acts works directly with text information, there is a knowledge gap to be tested in the news area that is KG introduction in extractive summarizations.

In this work, we were interested in summarization of news, since there is a demand from consumers to reduce reading time while maintaining a greater quantity and quality of information. So, the main objective is to perform and justify a new ranking formula that includes information coming from different KGs and check if the new rank of sentences selects some that have the biggest similarity with the original text news. By exploring different ranking schemas, we improve journalistic text extractive summarization and ensure that the extracted summaries are fact-based and more informative compared both to the news highlight and to the summarization without KGs.

2. Related Work

Gupta & Lehal (2010) show the evolution of extractive summarization tasks across the decades. It has been studied since the 1950s, starting with the Luhn (1958) method that assigned sentence weights based on the frequency of significant words while excluding common ones. Additional methods were introduced by Edmundson (1969), that combined sentence weightings based on cue words, title-related content and paragraph location. Kupiec et al (1995) introduced the Trainable Document Summarizer method by adding TF-IDF weights, statistical word selection and sentence scoring based on location and signature word presence.

In the 1990s and 2000s, later advancements explored Hidden Markov Models, offering a probabilistic approach to text summarization, with the potential use of clustering for building links between document terms and phrases and giving the basis for Machine Learning Language Models (LMs). Although classical probabilistic models were widely used until the 2000s, they were gradually replaced by neural models, such as Word2vec (Mikolov, 2013) and Transformers (Vaswani et al., 2017). The first introduced a method to generate dense vector representations of words that encode semantic and syntactic relationships and enable models to interpret the meaning and context more effectively. The second was a big turning point in the field of Natural Language Process (NLP) with the introduction of a mechanism of self-attention that allows a more efficient parallelization and context management.

Transformers allowed the surge of LLMs, which, for summarization tasks, demands at least a decoder component to be efficient.

Given the motivation behind this work, that is the use of techniques to obtain relevant knowledge through summarization of news, these techniques have been applied recently.

The first studies focused on extractive summarization, because they can generate good results and be done with simpler models with a low computational power requirement. Jassem & Pawluczuk (2015) developed a method based in TF-IDF to summarize Polish news by counting, sorting and selecting key sentences based on sentence importance. Li et al. (2015) proposed a system that enhances news summarization by using past articles to provide better context to current ones, a simple kind of knowledge injection. Mirani & Sasi (2017) presented an approach for summarizing news articles using TF-IDF method and using these outputs to provide a sentiment analysis, a combination of extractive techniques with emotional tone analysis. Alwis (2018) also performs extractive summarization with TextRank, advances with the adding of a graph-based approach to obtain a summary from multiple news sources and concludes that normalization by sentence length improves ROUGE values. Kynabay et al. (2021) provided a simple extractive summarization based on counting, sorting and selecting the most important words, but this was the first piece of work based on a text written in the Cyrillic alphabet.

In the last five years, a new kind of approach has been developed, with the application of the abstractive approach together with KGs to improve the similarity indicators of the original text and its summary. Abstractive methods benefit from KGs because, according to Agrawal et al. (2024), their introduction reduces the hallucinations of summarization models. Whether in the inference, the training or the validation phase, it is possible to mitigate hallucinations and improve reasoning accuracy.

The addition of KG information can be done in many different ways. Lakshika et al. (2020) proposed an innovative approach to news summarization by generating abstractive summaries, where the system not only extracts important content but also rephrases it, offering more natural and coherent outputs. Plus, KGs are used together with WordNet, thus adding a graph-based approach to the feature extraction performed with associative rule mining.

Another example of KG addition is Chen et al. (2023), who propose a novel summarization approach combining KGs and Transformers to deal with multiple news sources. The system first constructs knowledge graphs and graph attention networks from input texts to capture key entities and their relationships. The authors incorporate multi-source transformers to decode the main information and generate the final summary. CNN/DailyMail dataset, the same that will be used in this project, were used by the authors with a couple of changes as preprocessing steps, where they: (i) work with 10% of the dataset not to have a heavy model to run; (ii) eliminate the first sentence that, in general, is just an introduction without relevant information; and (iii) they put a limit on the text size to eliminate another source of variability.

In 2024, a brand new paper put together the extractive approach and the use of KGs. Although not in the field of communication: Yu et al. (2024) join a graph-based approach with an extractive summarization method, utilizing knowledge graphs and feature-based techniques such as word segmentation, part-of-speech tagging and named entity recognition. It integrates and segments data from IoT devices, enhancing it with KG data and generating final summaries. All information is imputed in a Graph Convolutional Network encoder and summarized in a Pointer-Generator Network, a new methodology called KG-based Contextualized Pointer-Generator Network.

3. Data & Approach

Given the objective of this work, which is to add KGs to improve journalistic text extractive summarization, the main problem to address is generating concise, extractive summaries of CNN/Daily Mail articles while preserving all important information.

The main approach used to attack the problem is to consider the importance given by some traditional extractive summarization model and add importance, per sentence, to words that are considered important in KGs. The concept of “importance” will be discussed in the next section.

To address the approach: (i) the chosen model is the TextRank (TR), a very fast, well-known and explainable algorithm; and (ii) the chosen KGs are Wikidata (WD), because it is one of the most vast KG to capture entities, its related entities and the properties between them (semantic information), and WordNet (WN), because it captures semantic lexical relations between entities in and out WD. These elements are joined together in an adaptation of the TR ranking formula. Details of the new formula are discussed in the next chapter.

For this work, we chose the CNN/DailyMail dataset, an English-language dataset containing just over 300k unique news articles written by journalists at CNN and the Daily Mail and paired with human-written highlights. The current version supports extractive and abstractive summarization because the dataset is hosted on the Hugging Face platform that converts the data into a dataframe shape helping to have better compatibility and performance in modern workflows (Abisee, 2024).

The original database is composed of three columns: the news id, the news itself and its corresponding human writing highlight. To exploit the new approach, the 30 first news are selected from the full dataset, so the number of final summaries will be equal to 60, half of them made by the traditional TR and half of them by the new approach. 30 observations is the minimum set in which any measure of it can already be considered statistically relevant.

The highlight will act as a "weak label" to the summarization task in the sense that, although the 32 summaries will be compared to the highlight, the main comparison is between each summary with and without KG information.

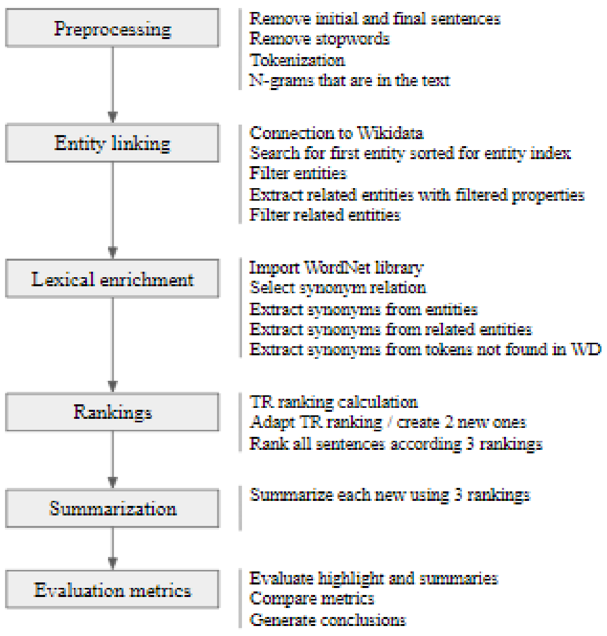
4. Implementation

To enhance text summarization by adding information coming from WD and WN, the goal is to leverage the rich semantic and lexical relationships between words and concepts to improve the importance of the sentences before selecting the top-N rank sentences to generate the summary.

The original dataset receives two adaptations: (i) adding a column that orders the news based on the number of words (from the smallest to the greatest text) for an organizational reason, so all runs could be done in more than one day and collecting the majority of summaries soon; (ii) by adding a column that counts the number of sentences n in the highlight and uses this value when summarizing with and without KG, so when calculating the similarity metrics, both the highlight and summaries are comparable by having the same size.

The main implementation is composed of six main steps: (i) text preprocessing; (ii) identification of the entities, their related entities and their properties in WD and creation of the main triples; (iii) lexical enrichment with WN and creation of the enrichment triples; (iv) sentences rank with TR ranking formula and the new sorter that adapts the TR one to deal with semantic and lexical relationships; (v) summarization itself, with and without KG information; and (vi) metrics calculus. Figure 1 shows a general list of tasks and mini-tasks, that are detailed in continuation.

Figure 1: General schema of the implementation


Preprocessing
Remove initial and final sentences
Remove stopwords
Tokenization
N-grams that are in the text
Entity linking
Connection to Wikidata
Search for first entity sorted for entity index
Filter entities
Extract related entities with filtered properties
Filter related entities
Lexical enrichment
Import WordNet library
Select synonym relation
Extract synonyms from entities
Extract synonyms from related entities
Extract synonyms from tokens not found in WD
Rankings
TR ranking calculation
Adapt TR ranking / create 2 new ones
Rank all sentences according 3 rankings
Summarization
Summarize each new using 3 rankings
Evaluation metrics
Evaluate highlight and summaries
Compare metrics
Generate conclusions

The preprocessing main task is compounded by the following mini-tasks: (i) similarly to Chen et al. (2023), extracting the interest new text, by removing the initial characters like "(CNN) – " or "NEW DELHI, India (Reuters) -- " and, beyond authors cleaning step, by removing the final signature that comes with and after "E-mail to a friend"; (ii) removing stopword, using *stopwords* method from *nltk.corpus* library, and creating tokens with *word_tokenize* method from *nltk.tokenize* library; (iii) indexing sentences, because although the granularity of the text is the word, the sentence level of the organization needs to be taken into account to create all interesting Python objects that will help to identify the most important sentences at the end of the process; and (iv) for each sentence, creation of all possible n-grams, using *ngrams* method from *nltk.util* library, and filter those that are observed in the text. The output of this step is a list of valid words (tokens and their related filtered n-grams) for WD and WN research.

The WD main task consists of the mini-tasks: (i) creating a connection with the platform and setting parameters like "English" for the language of interest and "JSON" for the format to store information; (ii) searching for entities based on the valid words of the previous step and add each word of each sentence in two different dictionaries, found and not found entities. If there is one of the more valid entities in WD, we order the WD identifier and select the first one, because the lower the id number, the more general the concept, a characteristic that we considered important; (iii) filter valid filtered with the help of some discretionary criteria for entities name (for example, if it is a number or if its size is bigger than 10 words) and with the help of another criteria for entity description (for example, demanding that it have any value and that this value is not related to any scientific article, by using a word blacklist); (iv) extract related entities of each valid filtered entity, with the help of a whitelist list of specific predicates in the Wikidata ontology ("Instance Of", "Subclass Of" and "Part of") and a blacklist of the related entity label (again, words related to scientific articles). The execution of these mini-tasks happens with SPARQL, which processes queries to deal with all restrictions. The output is a dictionary with all tuples per sentence, according to the RDF structure "subject-predicate-object", where each subject is a filtered valid entity, each object is a related filtered entity and the predicate is the property in between each filtered valid entity (subject) and each one of its related filtered entities (object).

Enrichment main task is made by using *wordnet* method from *nltk.corpus* library. The chosen lexical enrichment is synonyms, because the hypothesis raised is that the most important words in the text should appear many times, but not written in the same way for reasons of reading cadence, and that they can be captured using synonyms. With the help of *wordnet.synsets.lemma_names()*, synonyms of the previously filtered valid and related entities and the not found entities of the second mini-task of the previous task are extracted. The output is a dictionary with all tuples per sentence, once again according to the RDF structure "subject-predicate-object", where each subject belongs to the set compound by the filtered valid entities and the not found entities, each object is a synonym found in WN and all predicates have value "synonym".

In the fourth main task, for each sentence, if KG information is not added, the traditional TR algorithm ranks all sentences of the text news, but if KG information is added, then a new algorithm is applied to generate two new rankings that can change the importance of the sentences from the standard one. The new algorithm (*new_rank*) is a linear transformation of the TR one (*TR_rank*) by considering the number of main triples generated in the second main task (*m_tr*) and semantic triples in the third one (*s_tr*). Equation 1 and Equation 2 show new formulas, different and not linearly proportional to each other, but complementary for interpreting the final rankings values:

$$new\_rank_1 = TR\_rank \ * \ \left(1 + \frac{s\_tr}{m\_tr}\right) \qquad (1)$$

$$new\_rank_2 = TR\_rank \ * \ \left(1 + \frac{s\_tr}{s\_tr + m\_tr}\right) \qquad (2)$$

Both equations add 1 to each ratio, because in the scenario no semantic triple is generated, and then *TR_rank* is equal to both new sorters (the lower bound). Because of this, both equations will present bigger ranking values and it reflects the gain of importance by adding semantic and lexical information in a sentence.

Equation 1 shows explicitly the proportion between semantic triples and common ones, because it represents the percentage ranking value increase, so it is very easy to understand, but it has no limit for its value because it is sensible and its values inflate if *m_tr* is small, or if there is a big disproportion between *m_tr* and *s_tr*. Equation 2 solves the problem of the upper bound, because in a disproportional scenario, in the limit, *new_rank2* is the double of *TR_rank1*, but its value has a less clear interpretation than new_rank1.

Because of the pros and cons of both sorters, they complement themselves to order a new sentence. And because they are not linearly proportional to each other, some differences in ranking can be seen and will be discussed in the next part of this work. The final output is a list of sentences ranked according to three different ranking equations. Table 1 synthetases this main task:

No KG	Add KG
TR_rank	new_rank_1 new_rank_2

Table 1: All rankings to generate

Finally, one there are two brand new ranking formulas to deal with the addition of KG information and to select the sentences to generate the summary, the last main task to implement are the evaluation metrics between the main text and each one of the four called "short texts": (i) highlight; (ii) standard summary; (iii) KG summary with *new_rank*$_1$; and (iv) KG summary with *new_rank*$_2$). Seven selected metrics are grouped into two groups: (i) ROUGE family, with its metrics ROUGE-1, ROUGE-2, ROUGE-L and the average of them that is manually added; and (ii) BERTScore, with precision (p), recall (r) and F1-Score (f1) values.

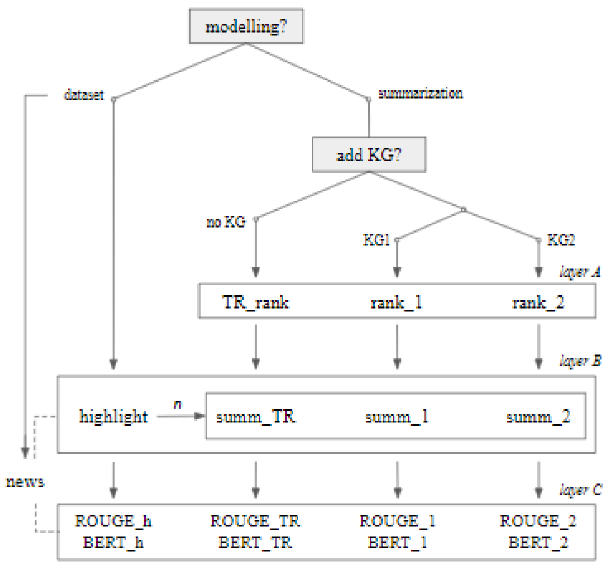
ROUGE family is selected as valid metrics because it is fast to compute and compares n-grams between the summary and the main text, good in an extractive summarization task, but it leads to higher metric levels compared to the highlight one (that written from scratch by humans remembers the abstractive summarization) and it does not capture lexical similarity well like synonyms, exactly the addition of information we did. For this reason, BERTScore, although computationally expensive compared to ROUGE, focuses on how semantically aligned two texts are, rather than exact word matches, which is good in a situation where semantic and lexical information are added from WD and WN. Considered together, Table 2 offers a good basis for comparison by joining extractive and semantic-lexical enrichment tasks. The output of this last main task is a set of 28 metrics per new.

Extractive summarization task	Semantic and lexical enrichment task
ROUGE family (1, 2, L)	BERT Score (p, r, f1)

Table 2: Complementary metrics

Finally, Figure 2 emphasizes the second part of the implementation after collecting all triples and enriched triples for each sentence. From the decision between add KG information or not, three layers (A, B and C) act, respectively, by running a ranking summarization algorithm, by generating the summary matching the *n* number of sentences that the highlight has generating a clear and fair comparison, and by calculating the metrics for each summary and the highlight itself all against the news text.

Figure 2: Ranking, summary and evaluation layers



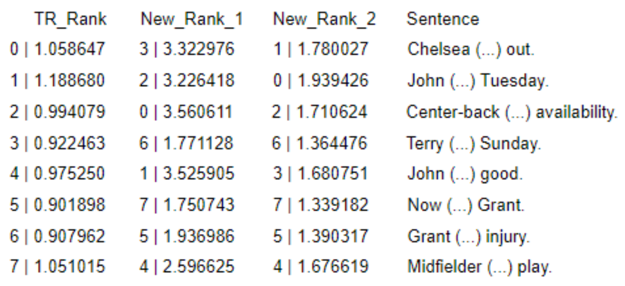
In the next session, we present the results of the ROUGE family and BERTScore metrics, for each ranking algorithm, so it is possible to compare which short text is closer to the content of interest for each new.

5. Evaluation

Before presenting the results of each summarization of the comparison between their metrics, we present an example of how *new_rank1* and *new_rank2* impact the sentence ranking.

Figure 3 shows an example of the first row, composed of the shortest of the 30 selected news, with just 7 sentences. Two things are observed. The first is the difference in the scale of the new ranking, as *new_rank1* has no upper bound and *new_rank2* has it equal to the double of *TR_rank*. The second is how the order of the sentences changes for both new sorting schemas. In the example, the first new schema excluded sentence #1 and in the second just changed the final order of the summary.

Figure 2: Ranking, summary and evaluation layers



TR_Rank	New_Rank_1	New_Rank_2	Sentence
0 | 1.058647	3 | 3.322976	1 | 1.780027	Chelsea (...) out.
1 | 1.188680	2 | 3.226418	0 | 1.939426	John (...) Tuesday.
2 | 0.994079	0 | 3.560611	2 | 1.710624	Center-back (...) availability.
3 | 0.922463	6 | 1.771128	6 | 1.364476	Terry (...) Sunday.
4 | 0.975250	1 | 3.525905	3 | 1.680751	John (...) good.
5 | 0.901898	7 | 1.750743	7 | 1.339182	Now (...) Grant.
6 | 0.907962	5 | 1.936986	5 | 1.390317	Grant (...) injury.
7 | 1.051015	4 | 2.596625	4 | 1.676619	Midfielder (...) play.

After running 30 news for summarization with and without KGs, and for different ranking schemas when adding KG information, Table 3 presents the results of the seven selected similarity metrics for each short text.

Metric	Highlight	Baseline (No KG)	Summary (KG 1)	Summary (KG 2)
ROUGE-1	0.164	**0.358**	0.320	0.352
ROUGE-2	0.088	**0.347**	0.310	0.339
ROUGE-L	0.119	**0.300**	0.268	0.291
Mean ROUGE	*0.124*	***0.335***	*0.300*	*0.330*
BERT p	0.849	0.948	0.948	**0.954**
BERT r	0.808	0.847	0.842	**0.849**
BERT f1	*0.849*	*0.895*	*0.892*	***0.897***

Table 3: Similarity metrics results by short text

As expected, all four ROUGE metrics presented best results for summarization without the addition of semantic and lexical information and BERT metrics had its biggest values with the second new ranking, that one in which there is a clear and easy to understand upper bound and with a reduced interpretability of the number. If we intent to have a global indicator to take both aspects of Table 2, Equation 3 presents the mean value between *Mean Rouge* and *BERT f1*:

$$final\_metric = \frac{Mean\ Rouge + BERT\ f1}{2} \quad (3)$$

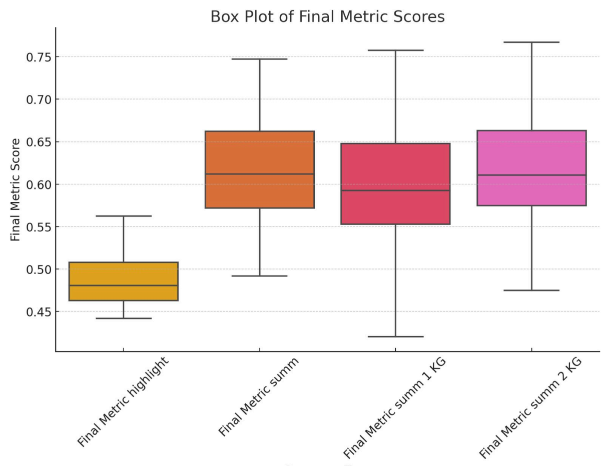
Table 3 shows a final view over the 30 examples with three kinds of summarization, two of them with addition of KG information.

Metric	Highlight	Baseline (No KG)	Summary (KG 1)	Summary (KG 2)
Final Metric	*0.486*	***0.615***	*0.596*	*0.613*

Table 3: Similarity metrics results by short text

All three summaries had a very similar performance. Figure 3 shows a boxplot that brings more information to Table 3 mean data. It is possible to see that even the distribution of the lowest and the highest scores are similar between standard summarization, and the two summaries with KG info.

Figure 3: Final Metric aggregated results boxplot



Compared to the highlight information, summarizing a news article brings more information to the readers. Table 4 presents the percentage difference between each summary and the highlight *Final Metric*.

Metric	**Highlight**	Baseline (No KG)	Summary (KG 1)	Summary (KG 2)
Final Metric	**-**	*+26.5%*	*+22.6%*	*+26.1%*

Table 4: Similarity metrics results by short text

Compared to the standard summarization algorithm, both new ones, *new_rank*$_1$ and *new_rank*$_2$ had similar performance taking into account an error margin of 5% for more and less as seen in Table 5. And between the new algorithms, *new_rank*$_2$ was more successful in adding semantic and lexical information to enrich the sentences than *new_rank*$_1$.

Metric	Highlight	**Baseline (No KG)**	Summary (KG 1)	Summary (KG 2)
Final Metric	*X*	**-**	*-3,08%*	*-0,003%*

Table 5: Similarity metrics results by short text

To conclude, we also observed the behavior of the metrics along the news article. At the beginning of the Implementation session, we discussed that one of the preprocessing made in the original dataset was ordering them by the number of characters. By doing that, it was possible to observe the behavior of each metric along each article news while its text increases (Figure 4). In a linear plot, it is clear that the highlights results were way below the summaries results. In addition, it is noticeable a negative strong spike in the new_rank1 result for new #20, a clear outlier..

Figure 4: Final Metric individual results linear plot

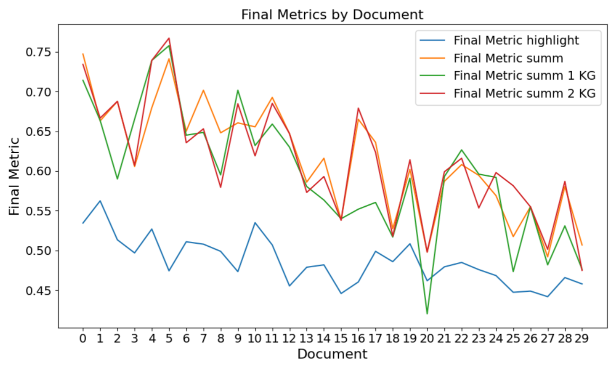
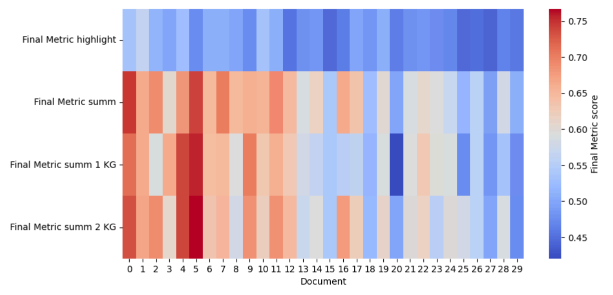


Figure 5 brings another type of visualization, a heatmap with the values of Final Metric per each new article text, where we can see that summaries results overperformed the highlight results for all of the cases. As observed in Figure 4, it is also possible to observe the declining results along the cases when they increase more and more in size, but in Figure 5 is more visible the slightly greater results that *new_rank*$_2$ produced in comparison with *new_rank*$_1$.

Figure 5: Final Metric individual results heatmap



6. Conclusions and Future Work

To conclude, we successfully achieved the goal of this project in tackling work with semantic web technologies (WD and SPARQL) and NLP (WN, TR and extractive summarization). We presented a valuable technique for enriching text summarization using KGs that can be applied to the TR algorithm.

Numerical results suggest that when compared to the standard summarization using ROUGE metrics, the addition of KG info does not add relevant information, so comparing computational processing time, it is fair to say that the standard summarization already helps extract relevant news information for readers (33,5% similarity against 33,0%). When compared to BERT metrics, there is a small gain in performance (89,7% against 87,5%), so the higher time spent to run KG information would need to be compensated with the use of one or more GPUs. And comparing the two new algorithms to rank the sentences with KG, new_rank2 always presents, on average, better similarities compared to new_rank1, so it can be seen as a good benchmark for extractive summarization with KG.

Some critical analyses can be made. The first point is related to the capture of entities and related entities. It would be possible to identify the grammatical class of some tokens and capture specific entities for each grammatical class, which increases the relevance of related entities. The second relates to how main triples and lexical triples were added to TR_rank, as we used equal weight to the properties that related each subject to each object. It would be possible to add a weight for each type of predicate, for example, double the weight for triples whose predicates are "synonyms". Talking about lexical triples, it would be possible to look for other forms of relationship besides "synonym", such as "logical implication" or "derivational relationship", however, the gain from synonyms is high as explained in the Implementation session and it is easy to interpret. A fourth critical analysis can be made of the code because, given its length, it does not have the most optimized processing, especially method *query_multiple_wikidata_entity_relationships()*, which, according to the dependency diagram (Inforestud@nte submission attached image) could be eliminated.

A positive critical analysis is that initially it was only expected to create one new ranking and it was possible to double the goal, so in addition to a comparison with the standard summarization, a comparison between them emerged. In addition, computational costs were encountered given the need for word-by-word connections on WD. Finally, the loss of performance observed in Final Metric (observed in any other metric individually) is relative and due to a methodological choice to make summaries with the same number of sentences as a highlight, so performing a summarization that generates a size equal to X% of the original text would improve the current results.

Given that research in the field of NLP is quite vast, this work goes beyond initial expectations and provides suggestions for future applications based on the lessons learned. From the more theoretical to the more practical, the first relates to different approaches to introducing KG information. For example, it could be included after the standard summarization, aiming to adapt the purely extractive learning by adding relevant text words that have lots of connections with other ones in the computational graphs of KGs.

Regardless of the approach used, a second suggestion is to use other Knowledge Bases such as Word2Vec to add context, or to work directly with the WN to extract semantic relationships that the WD is currently slow to extract.

The third suggestion, connected directly to the code (low level), is to reduce its complexity by: (i) reducing SPARQL query latency with the employment of local caching or batch queries or leveraging a local KG setup; (ii) by limiting the depth of the relationships obtained between entities and related entities in the generation of triples. A last suggestion relates to new low-level changes within the TextRank algorithm, such as pre-computing semantic scores, applying KG-based adjustments selectively or simplifying cosine similarity calculations through approximate methods and optimizing text preprocessing with faster tokenization libraries like spaCy.

7. Bibliographic References

Abisee. (n.d.). CNN/Daily Mail dataset. Hugging Face. Retrieved October 12, 2024, from <https://huggingface.co/datasets/abisee/cnn_dailymail>.

Agrawal, G., Kumarage, T., Alghamdi, Z., & Liu, H. (2024). Can Knowledge Graphs Reduce Hallucinations in LLMs? : A Survey. In K. Duh, H. Gomez, & S. Bethard (Eds.), Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1: Long Papers, pp. 3947-3960). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.219.

Alwis, V. (2018). Intelligent e-news summarization. In 2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer) (pp. 318-324). IEEE. https://doi.org/10.1109/ICTER.2018.8615487.

Chen, T., Wang, X., Yue, T., Bai, X., Le, C. X., & Wang, W. (2023). Enhancing abstractive summarization with extracted knowledge graphs and multi-source transformers. Applied Sciences, 13(13), 7753. https://doi.org/10.3390/app13137753&#8203;:contentReference[oaicite:0]{index=0}&#8203;:contentReference[oaicite:1]{index=1}.

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the ACM*, 16(2), 264–285. https://doi.org/10.1145/321510.321519.

Gupta, V., & Lehal, G. S. (2010). A Survey of Text Summarization Extractive Techniques. Journal of Emerging Technologies in Web Intelligence, 2(3), 258–268. https://doi.org/10.4304/jetwi.2.3.258-268.

Jassem, K., & Pawluczuk, Ł. (2015). Automatic summarization of Polish news articles by sentence selection. In Proceedings of the 2015 Federated Conference on Computer Science and Information Systems (pp. 337–341). Annals of Computer Science and Information Systems.

Kupiec, J., Pedersen, J., & Chen, F. (1995). A trainable document summarizer. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 68–73. https://doi.org/10.1145/215206.215333.

Kynabay, B., Aldabergen, A., & Zhamanov, A. (2021). Automatic summarizing the news from Inform.kz by using natural language processing tools. 2021 IEEE 15th International Conference on Application of Information and Communication Technologies (AICT). IEEE.

Lakshika, N. K., Rasnayaka, A. C., & De Zoysa, K. (2020). Abstractive web news summarization using knowledge graphs. Proceedings of the International Conference on Machine Learning and Computing (ICMLC). Association for Computing Machinery (ACM). https://doi.org/10.1145/3374549.3374573&#8203;:contentReference[oaicite:0]{index=0}&#8203;:contentReference[oaicite:1]{index=1}.

Li, F., Chen, Y., & Li, Z. (2015). Learning from the past: Improving news summarization with past news articles. In Proceedings of the 2015 International Conference on Asian Language Processing (IALP) (pp. 140-143). IEEE. https://doi.org/10.1109/IALP.2015.7451551.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *Presented at IRE National Convention*, New York, 159–165.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.

Mirani, T. B., & Sasi, S. (2017). Two-level text summarization from online news sources with sentiment analysis. 2017 International Conference on Networks & Advances in Computational Technologies (NetACT), Trivandrum, India. IEEE. https://doi.org/10.1109/NETACT.2017.8076735.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*, 5998–6008. https://doi.org/10.48550/arXiv.1706.03762.

Yu, Z., Wu, S., Jiang, J., & Liu, D. (2024). A knowledge-graph based text summarization scheme for mobile edge computing. Journal of Cloud Computing: Advances, Systems and Applications, 13(1), 1–15. https://doi.org/10.1186/s13677-023-00585-6.