TitleQuill: Keywords enrichment and Title refinement using Pre-trained Models

Nicola Aggio
880008@stud.unive.it

Giovanni Costa
880892@stud.unive.it

Martina Novello
880893@stud.unive.it

Sebastiano Quintavalle
878500@stud.unive.it

1 Introduction

Nowadays we are constantly overwhelmed by a huge amount of information and it's becoming increasingly challenging to navigate through the abundance of results in search of the desired one. In this regard, a popular task in the field of NLP comes to help: text summarization. The goal of this task is to create a text of limited size compared to the original one, but still capable of preserving its core meaning and main aspects. The limit case of summarization is Title Generation, that aims to capture the essence of the concept expressed in the text in a few words or in a sentence at most.

The required constraints for the generation of a headline make the problem very challenging: the title is likely to be the only thing the reader will read for most of the considered documents, so it must be catchy but at the same time it has to faithfully represent the original content. Moreover, it must adhere to the stylistic tone of the text, ranging from a more journalistic approach of news to a more formal setting of a scientific publication. Our work will focus primarily on the latter case, which is a particularly intriguing case of study.

Furthermore, scientific publications often carry additional summarizing information beyond the title: the abstract keywords, which are a collection of relevant words that contain the core of the entire discussion. Our work will also attempt to manipulate the data in order to extract this additional information along with headline generation.

The further developments of this project can be found at https://github.com/MrCosta57/TitleQuill

2 Related works

The task of automatic title generation is interesting and widely discussed due to its various applications and it has been a popular research topic over the years. A crucial turning point occurred with the advent of the Machine Learning and especially with the beginning of transformers architecture [16] era. In the literature, we can find two main families of approaches: abstractive and extractive summarization.

Abstractive algorithms focus on the encoder-decoder architecture. The first component basically learns the main information from the original text, while the second one is responsible for generating a concise title. The most remarkable results are obtained using transformers architecture [16], capable of achieving state-of-the-art performance in major benchmarks. One relevant approach [4] combines the universal transformer [2] model, a generalization of the original formulation to solve the non-recursivity problem, along with the byte pairing technique. Another remarkable abstractive approach [12] leverages pre-trained GPT2 [15] technology to perform fine-tuning on the specific dataset, decomposing the task into three stages: generation, selection and refinement. This system currently holds the state-of-the-art in most benchmarks, including the arXiv scientific paper one[1].

On the other hand, the extractive approach produces summaries by copying and concatenating the most important spans (usually sentences) in a document. The state-of-the-art approach is BERTSum [9] which reorganizes the typical BERT [3] structure to work at sentence-level. The embedding of each sentence is passed through an Inter-Sentence-Transformer which determines sentence relevance using binary classification. Finally, relevant sentences are passed through a decoder model that simulates the abstractive part in a two-step training process [10].

[1] https://www.kaggle.com/datasets/Cornell-University/arxiv

3 Our approach

As described above, the scientific literature is very rich in information and methods related to the title generation task. Our approach attempts to innovate these procedures using two pre-trained models, each of which is highly specialized on a specific task. We train the model end-to-end in order to mix the two concepts of extractive and abstractive generation. Moreover, we want our method to automatically extract the most relevant words in the abstract for two main reasons: firstly to return them to the user, secondly to exploit the relation between them and the abstract in order to increase the semantic precision of the title creation.
The entire workflow is represented graphically in figure 1 and can be summarized in three main modules, described in details below: Keywords Extraction, Keywords Enrichment and Title Refinement.

Keywords Extraction Module The process of keyword extraction from the abstract exploits ALBERT [7], a powerful encoder-only model based on BERT [3] that uses parameter-reduction techniques to reduce the memory consumption and increase the overall scalability of the model. Having a specialized module that succeeds in this task turns to be crucial as keywords' information is fundamental for a coherent title generation and a good indexing in the search engines, and it will also affect the process in the rest of the pipeline.

More in details, the extraction is implemented as a sequence labelling task: each token representation obtained from the ALBERT contextual embedding is binary classified as a keyword or not. During the fine-tuning step, when papers' keywords are known, cross entropy for each token is computed using the ground truth information.

Keywords Enrichment Module Once keywords are extracted, the related information is derived from the abstract by peeking all the sentences that contain the such words. This step allows considering only the most relevant information, but at the same time gives to the next module a more wide context rather than the single keyword.

Title Refinement These sentences containing the keywords can be considered as possible title candidates. However, the overall information they convey should be blended and redefined in order to achieve a unique, concise and meaningful result: the paper title. For this purpose we use Mistral [6], a Language Model that is based only on the transformer decoder part, to generate the next word given the previous one.
We append the $< TL; DR :>$ after the output of the previous *Keywords Enrichment Module* and before the special token. Indeed, considering that Mistral was trained on massive internet data, this step is needed to make the model aware to output a summarized version of the important sentence selected before, i.e. the well-formed, concise and semantic meaningful title.

More in details, the fine-tuning of Mistral is performed computing the cross entropy loss between the conditional probability of output a word given the previous one and the original title word. The next token prediction is performed using the Beam Search.

In the end, once both the losses have been obtained, it is finally possible to update the parameters of both the models using the sum of the two losses. This procedure makes the whole workflow trainable end-to-end. Furthermore, the choice to use efficient models was made considering the GPU/TPU limitations that a non-professional setup may have.

Baselines The baseline algorithm for this problem is considered peeking the sentence in the abstract containing the highest number of keywords, assuming to have them in the dataset as in this case. In the event of a tie, select the one containing the first keyword. We assume the keywords to be written in order of importance and relevance within the abstract.
Another baseline algorithm could be taking the first sentence of the abstract, if keywords are not present in the dataset.

3.1 Competitors

We assess how our model compares to the performance achieved by other leading models for the title generation task. We consider the best for any different type of approach. In this sense, the main competitors are TextRank [11], a graph-based approach with a particular modelling of the weighting scheme, Bi-GRU [1], that combines sequence-to-sequence Gated Recurrent Unit with attention, and the yet discussed approach using fine-tuned GPT2 [12] in a three-step pipeline.

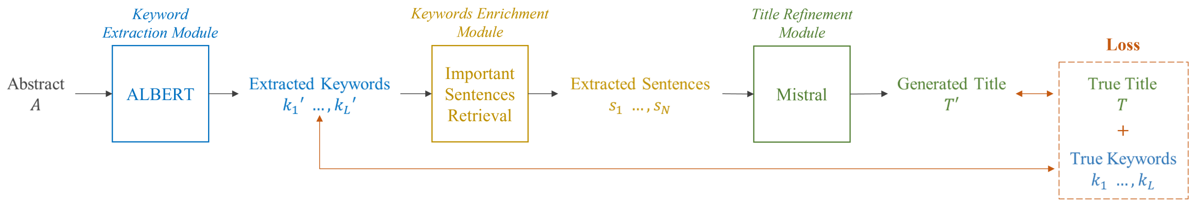


Figure 1: Our approach pipeline

3.2 Schedule

Planning, resource allocation and efficient time management are fundamental for the successful execution of any research project. Our approach involves breaking down the work into distinct phases, each of which has been assigned dedicated time to ensure that every aspect is addressed with the necessary attention. Being aware of the dynamic nature of research projects, we have also added a certain degree of flexibility to address unexpected challenges. The following schedule serves as a roadmap, guiding us through data preparation, model development, analysis and documentation:

1. Data collection and pre-processing (1 week): access, clean and preprocess the existing dataset, conduct exploratory data analysis to understand its characteristics.

2. Model setup (3 weeks): research and select appropriate versions of the chosen models, configure them based on the project requirements and setup the training environment.

3. Train and fine-tune (3 weeks): train the model on the prepared dataset and fine-tune as necessary for optimal performance.

4. Performance evaluation and analysis (2 weeks): evaluate the model on a separate test dataset, analyse and compare results with existing benchmarks or manual evaluations.

5. Documentation and reporting (2 week): drafting of the final reports, summarizing the project's objectives, methodologies, results and conclusions.

Recognizing the intricate nature of our tasks, we have opted for a collaborative approach across all aspects of the project. However, it will be necessary to divide the responsibility for the implementation of the model (point 2). In this phase, individual team members will take on specific aspects of model development, leveraging their expertise. Regular meetings will serve as the backbone of our communication, allowing for continuous exchange of ideas, progress updates and issue of resolutions to ensure that every team member remains actively engaged and contributes to the success of the project.

4 Experiments

In conducting our experiments, we configured our Deep Learning model to optimize its performance according to various metrics. More specifically, we employed a comprehensive set of metrics including ROUGE, BLEU and AES to evaluate the efficacy of our NLP task.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [8] is a metric commonly used for assessing the quality of text summarization. It evaluates the n-gram overlap between the generated summary and reference summaries, emphasizing recall as a measure of content overlap. In this sense, the main drawback of such metric is that it can be sensitive to superficial matching, and may not capture the semantic understanding or fluency of the generated text. This peculiarity intensifies in abstractive summarization, as it mainly tries to synthetize the original text, without generating novel, paraphrased content.

BLEU (Bilingual Evalutation Understudy) [14] is a metric widely used to evaluate the quality of machine-generated text, particularly in machine translation tasks. BLEU computes precision-based n-gram overlap between the generated text and reference text, providing a quantitative measure of the model's linguistic accuracy. Although being a metric mainly used for machine translation, it is also sometimes used for text summarization [5], but it was shown that it performed worse than ROUGE in such task [5].

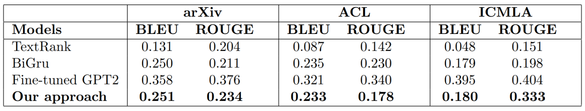
AES (Automatic Evaluation of Summaries)

[13] is a metric specifically designed for evaluating the quality of abstractive text summarization. It leverages pre-existing human-generated summaries to assess the coherence and informativeness of the model-generated summaries, offering a holistic evaluation of content comprehension.

In comparing these metrics, ROUGE emphasizes recall, BLEU focuses on precision, and AES provides a more comprehensive assessment of abstractive summarization. By considering these metrics collectively, we aim to gain a better understanding of our model's performance, ensuring an evaluation of its effectiveness in capturing both content overlap and linguistic quality.

5 Results

To better understand the performance of our model, we compare our ROUGE and BLEU metrics with those of the competitors (AES one was not present in their papers) evaluated on the most popular benchmark datasets in the field, such as arXiv[1], ACL[2], and ICMLA[3].



	arXiv		ACL		ICMLA	
Models	**BLEU**	**ROUGE**	**BLEU**	**ROUGE**	**BLEU**	**ROUGE**
TextRank	0.131	0.204	0.087	0.142	0.048	0.151
BiGru	0.250	0.211	0.235	0.230	0.179	0.198
Fine-tuned GPT2	0.358	0.376	0.321	0.340	0.395	0.404
Our approach	**0.251**	**0.234**	**0.233**	**0.178**	**0.180**	**0.333**

Figure 2: Results

6 Data

Our model heavily relies on keywords for the extractive phase, so it is necessary to find a dataset that includes abstracts, titles and keywords too. Many popular datasets in the field of paper title generation unfortunately do not contain any information regarding keywords, thus cannot be used for training. The dataset that best meets our requirements is OAGKX[4], a keyword extraction/generation dataset consisting of 22,674,436 abstracts, titles and keyword strings from scientific articles. The textual information has already undergone preprocessing, including lowercasing and tokenization. The dataset is released under the CC-BY license[5] and it's freely downloadable by the official dataset page as a JSONL file or though Hugging Face[6] site.

It's important to point out that the constraint of having a dataset with keywords only limits to the training phase, but since our model uses keywords' labels only at an intermediate step it is still possible to evaluate metrics on the final titles testing phase and compare them with competitor models on the main benchmarks in this field.

7 Tools

The foundation of our project will be built upon PyTorch[7], a dynamic and highly extensible deep learning framework, which provided us with the flexibility to design and implement complex neural network architectures. More specifically, to streamline the training process and enhance reproducibility, we will exploit PyTorch Lightning[8], an abstraction layer built on top of PyTorch that automates common training tasks and simplifies the overall workflow.

Moreover, natural language processing tasks, such as tokenization or parsing, will be addressed by SpaCy[9], a leading open-source library for advanced text processing and linguistic analysis, which will allow us to efficiently preprocess and analyze the textual data of our dataset.

We will also use Weights & Biases[10] platform designed to help machine learning developers to track and visualize their ML experiments by versioning and iterating on datasets, evaluating model performance, reproducing models, and managing workflows end-to-end.

We will take advantage of Hugging Face[6], a hub for natural language processing models and pipelines, both for retrieving our dataset and for building our model architecture.

Finally, the experimental setup will be based on Google Cloud Platform[11] and Google Colab[12] infrastructure, that provides powerful hardware for training our system.

The integration of all these tools will not only help us in speeding up the development phase, but will also contribute to the overall success and efficiency of our Deep Learning project.

[2] https://aclanthology.org

[3] https://archive.ics.uci.edu/dataset/434/icmla

[4] https://lindat.mff.cuni.cz/repository/xmlui

[5] https://creativecommons.org

[6] https://huggingface.co/

[7] https://pytorch.org/

[8] https://lightning.ai/docs/pytorch/stable/

[9] https://spacy.io/

[10] https://wandb.ai/site

[11] https://cloud.google.com/

[12] https://colab.research.google.com/

References

[1] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[2] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and L. Kaiser. Universal transformers. *CoRR*, abs/1807.03819, 2018.

[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[4] D. Gavrilov, P. Kalaidin, and V. Malykh. Self-attentive model for headline generation. *CoRR*, abs/1901.07786, 2019.

[5] Y. Graham. Re-evaluating automatic summarization with bleu and 192 shades of rouge. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 128–137, 2015.

[6] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023.

[7] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations, 2020.

[8] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[9] Y. Liu. Fine-tune BERT for extractive summarization. *CoRR*, abs/1903.10318, 2019.

[10] Y. Liu and M. Lapata. Text summarization with pretrained encoders, 2019.

[11] R. Mihalcea and P. Tarau. TextRank: Bringing order into text. In D. Lin and D. Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[12] P. Mishra, C. Diwan, S. Srinivasa, and G. Srinivasaraghavan. Automatic title generation for text with pre-trained transformer language model. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*, pages 17–24, 2021.

[13] A. Moeed, Y. An, G. Hagerer, and G. Groh. Evaluation metrics for headline generation using deep pre-trained embeddings. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1796–1802, 2020.

[14] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[15] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.