News Articles Title Generation

Kaggle group 10.40.11

Paul Vanborre Jérémie Touati

April 12, 2024

Introduction

For this project, we understood the task of *title generation of news articles* as a task of *summarization*. Indeed, the titles from our dataset consist in short versions of their corresponding texts, capturing the essence of their message with only a few sentences. To tackle the challenge of title generation of news articles, we led and combined different approaches.

- This report will first detail how we used transformers-based pre-trained models to achieve abstractive summarization tasks.
- We will then explore the effects of various extractive summarization techniques before applying those pre-trained models.
- Finally, we will delve into how the fine-tuning of these models on our particular dataset has improved the results.

1 Using pre-trained transformers models

1.1 The transformers architecture

The Transformers structure, introduced by the paper Attention is all you need (Vaswani et al. (2017)), has two parts:

- The encoder part, that builds a representation of the input.
- The decoder part, that from this representation of the input builds a target sequence.

Both part rely on attention layers, that basically allow to pay attention to certain words in the sentence when creating the representation of a specific word. In the encoder, we can pay attention to all words in our sentence, before and after the word we are considering. While in the decoder, we can only attend words that are before the word we are considering.

From this overall architecture we can derive three kind of models :

- Encoder-only models, for tasks that require an understanding of the input, such as named entity recognition. Examples of such models are BERT (see Devlin et al. (2018)).
- Decoder-only models, well suited for generative tasks. Examples of such models are GPT-2 (see Radford et al. (2019)).
- Sequence to sequence models, that use both the encoder and the decoder part, when you want to perform generative tasks that take into account the input, like translation and summarization.

We treated this problem of news articles title generation as a summarization problem, so we will focus from now on on this kind of Seq2Seq models. Examples of such models are T5 (Raffel et al. (2020)), BART (Raffel et al. (2020)), Pegasus (Zhang et al. (2020)).

1.2 Using models from HuggingFace: need for French well-suited models

However we faced a challenge, namely the fact that our data was fully in French and that most Seq2Seq models are English specific (for instance Pegasus).

An idea to overcome this issue was to use multilingual summarization models like mT5 (Xue et al. (2020)) or mBART (Liu et al. (2020)), or even better French-specific models like BARThez (Eddine et al. (2020)) or CamemBERT (Martin et al. (2019)).

See the appendix .1 for a description of the mT5, camemBERT-2-camemBERT and BARThez models, that we will use in our experiments below.

1.3 First results on raw models

The table 1 shows the results obtained on our test set when using three raw models from HuggingFace: mt5-small, camembert2camembert and BARThez.

Table 1: ROUGE scores on test set with raw models from HuggingFace

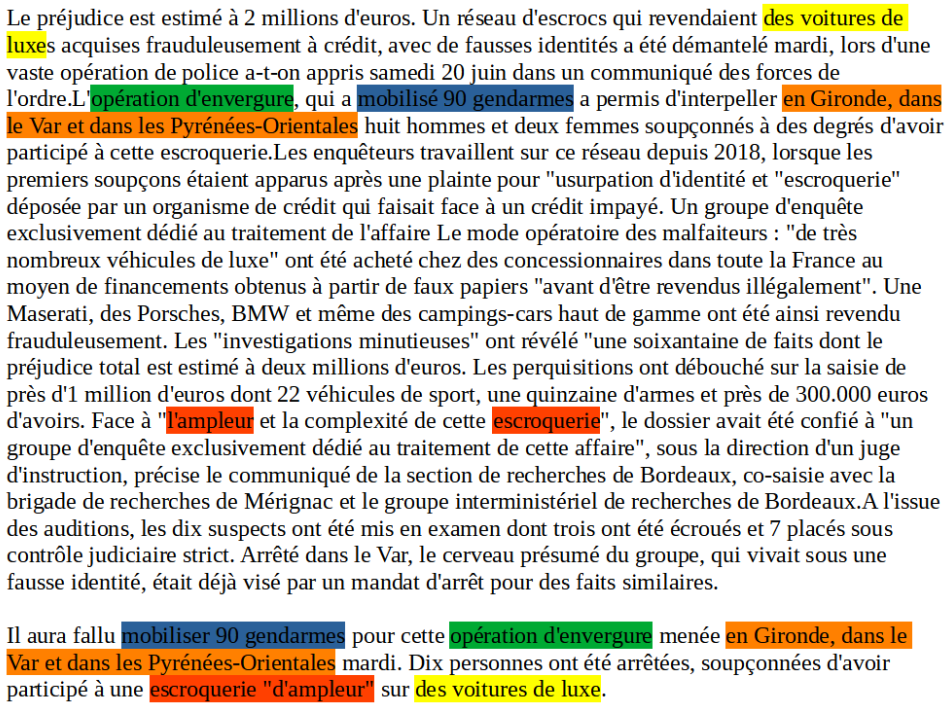
mt5-small	**camembert2camembert**	**BARThez**
3.89%	19.46%	15.73%

The summaries generated by mt5-small are most of the times one-word long. This has to be fixed thanks to fine-tuning as we will do in part 3. On the contrary, the camembert2camembert and BARThez models already do a great job by generating summaries which look like those from our train and validation sets.

2 Combining with extractive summarization

2.1 The motivations behind extractive summarization

Looking at the titles from the train and validation sets, we noticed that most of them were mainly composed of words extracted from their corresponding text. Figure 1 gives an example of a text from the train set with its corresponding title. It shows important phrases of the title directly extracted from the article. This motivated us to achieve an extractive summarization task in order to keep only a few sentences, those containing the important information.


Le préjudice est estimé à 2 millions d'euros. Un réseau d'escrocs qui revendaient des voitures de luxes acquises frauduleusement à crédit, avec de fausses identités a été démantelé mardi, lors d'une vaste opération de police a-t-on appris samedi 20 juin dans un communiqué des forces de l'ordre.L'opération d'envergure, qui a mobilisé 90 gendarmes a permis d'interpeller en Gironde, dans le Var et dans les Pyrénées-Orientales huit hommes et deux femmes soupçonnés à des degrés d'avoir participé à cette escroquerie.Les enquêteurs travaillent sur ce réseau depuis 2018, lorsque les premiers soupçons étaient apparus après une plainte pour "usurpation d'identité et "escroquerie" déposée par un organisme de crédit qui faisait face à un crédit impayé. Un groupe d'enquête exclusivement dédié au traitement de l'affaire Le mode opératoire des malfaiteurs : "de très nombreux véhicules de luxe" ont été acheté chez des concessionnaires dans toute la France au moyen de financements obtenus à partir de faux papiers "avant d'être revendus illégalement". Une Maserati, des Porsches, BMW et même des campings-cars haut de gamme ont été ainsi revendu frauduleusement. Les "investigations minutieuses" ont révélé "une soixantaine de faits dont le préjudice total est estimé à deux millions d'euros. Les perquisitions ont débouché sur la saisie de près d'1 million d'euros dont 22 véhicules de sport, une quinzaine d'armes et près de 300.000 euros d'avoirs. Face à "l'ampleur et la complexité de cette escroquerie", le dossier avait été confié à "un groupe d'enquête exclusivement dédié au traitement de cette affaire", sous la direction d'un juge d'instruction, précise le communiqué de la section de recherches de Bordeaux, co-saisie avec la brigade de recherches de Mérignac et le groupe interministériel de recherches de Bordeaux.A l'issue des auditions, les dix suspects ont été mis en examen dont trois ont été écroués et 7 placés sous contrôle judiciaire strict. Arrêté dans le Var, le cerveau présumé du groupe, qui vivait sous une fausse identité, était déjà visé par un mandat d'arrêt pour des faits similaires.

Il aura fallu mobiliser 90 gendarmes pour cette opération d'envergure menée en Gironde, dans le Var et dans les Pyrénées-Orientales mardi. Dix personnes ont été arrêtées, soupçonnées d'avoir participé à une escroquerie "d'ampleur" sur des voitures de luxe.

Figure 1: A text and its corresponding title from the train set. Main phrases of the title are extracted from the text

However, most of the phrases extracted from the text are set in a different order in the title, the verbs are conjugated and the adjectives made agree differently. Synonyms are also often used instead of the original words from the text. This is why we considered important to use this extractive summarization only as a pre-processing step. The idea would be to feed our abstractive summarization models with short extracted texts.

Doing so, we expect several benefits. First, as the extracted texts are shorter than the original ones, the generation of the titles should be faster. This should also speed-up the fine-tuning of our models (see part 3 where we explain the interest of fine-tuning). Furthermore, as we feed it with only essential information extracted from the texts, we expect the abstractive summarization model to be "helped" by this pre-processing step and to compute better titles. This extractive step indeed does part of the summarization process, which is supposed to

capture the message of the article. Simple reformulation and re-ordering of the remaining ideas will be handled by the abstractive part.

2.2 Techniques implemented

Latent Semantic Analysis

To compute our first extractive summarization method, we got our inspiration in the Machine and deep learning course followed in the first period Vazirgiannis (2023-2024a) which introduced the notion of Latent Semantic Indexing or Analysis (LSA). Indeed, we used the LsaSummarizer module of *Sumy* library to keep the k most meaningful sentences of each of our texts. To do so, it computes the SVD decomposition of the term/sentence matrix of the text (see Padmakumar (2014)). It then keeps the first k rows of the sentence matrix, each one being the representative of a distinct context of the document.

In figure 2, we give the result of LsaSummarizer for the text introduced above. The main phrases of the title are still almost all present, which motivates the choice of this technique to pre-process our texts.

Un réseau d'escrocs qui revendaient des voitures de luxes acquises frauduleusement à crédit, avec de fausses identités a été démantelé mardi, lors d'une vaste opération de police a-t-on appris samedi 20 juin dans un communiqué des forces de l'ordre.L'opération d'envergure, qui a mobilisé 90 gendarmes a permis d'interpeller en Gironde, dans le Var et dans les Pyrénées-Orientales huit hommes et deux femmes soupçonnés à des degrés d'avoir participé à cette escroquerie.Les enquêteurs travaillent sur ce réseau depuis 2018, lorsque les premiers soupçons étaient apparus après une plainte pour "usurpation d'identité et "escroquerie" déposée par un organisme de crédit qui faisait face à un crédit impayé. Les "investigations minutieuses" ont révélé "une soixantaine de faits dont le préjudice total est estimé à deux millions d'euros. Arrêté dans le Var, le cerveau présumé du groupe, qui vivait sous une fausse identité, était déjà visé par un mandat d'arrêt pour des faits similaires.

Figure 2: Extractive summary of the text above with LsaSummarizer (5 sentences kept). Most of the important phrases are still present in the extracted summary

BERT extractive summarizer

This extractive summarizer (see Miller (2019)) was initially designed for lectures. It uses BERT embeddings and then clusters these embedddings using a K-Means algorithm. K is an hyperparameter that corresponds to the final number of sentences in the summary. Hence, we select for each cluster the sentence that is the closest to the centroid for the summary.

2.3 Results

Table 2 shows how the extractive summarization pre-processing step (5 sentences kept in both cases) impacted the ROUGE score of the raw models from HuggingFace on our test set.

Table 2: Impact of the extractive summarization pre-processing step

ROUGE score	**mt5-small**	**camembert2camembert**	**BARThez**
No extractive summarization	**3.89%**	**19.46%**	**15.73%**
LSA extractive summarization	1.65%	15.82%	14.08%
Bert extractive summarization	1.82%	18.47%	14.61%

It appears that mt5 does not succeed any better, which is hard to interpret as the ROUGE score was initially very low (without fine-tuning). However, we witness a drop in the ROUGE score for the camembert2camembert model (especially for the LSA technique), and a slighter drop for BARThez. It may be that the models need more fine-tuning when we feed them with the extracted texts instead of the original ones. The next part will focus on such a fine-tuning.

3 The impact of fine-tuning

The transformers-based pre-trained models from HuggingFace that we introduced in part 1 were trained on large corpus of documents that are not necessarily close to our dataset. We would like to indicate to the model how long a title should be, to what extent it should extract and re-use words from the article, how long its sentences should be... To sum up, we want to teach it how to mimic the stylistic "signature" of the titles from our train and validation sets. This third part thus delves into the fine-tuning of the models we saw previously in Vazirgiannis (2023-2024b) and evaluates the impact on the ROUGE score.

3.1 Transfer learning

Due to the data we have, we do not have enough data to train a whole deep learning architecture, that requires billions of data. Instead, we use ideas from transfer learning (see Tan et al. (2018) for a survey, and more specifically what they denote Network-Based Deep Transfer Learning. Among several transfer learning strategies, we chose to focus on Network-Based Deep Transfer Learning. This means that a network was trained with huge data corpus. Then we take a part of this network (in our applications, often the whole network or only the summarization part) that we plug in our new architecture (in our applications, we did not add any supplementary layers). This new architecture is much easier to train using fine-tuning and few data since it relies on the data that was used during the first training part.

In short, we pre-train the model on generic corpora, and then fine-tune it to specific corpora and task. See the appendix .2 to see how we implemented this transfer learning scheme using HuggingFace libraries (mostly the Transformers library).

3.2 Connecting to the school's servers via SSH

When launching the fine-tuning on our personal computers, whether it is for mt5 model or camembert2camembert, the time needed to compute the training was inconceivable. We hence decided to connect to the school's computers via SSH. The table 3 recaps the training time over one epoch on our personal computers and with SSH. It also tells how our extractive summarization method (LsaSummarizer to keep 5 sentences) impacted the fine-tuning time.

Table 3: Example of fine-tuning time reduction (mt5 model, on train set, one epoch of training)

Training time	**On our personal computers**	**gelinotte.polytechnique.fr's server**
With LsaSummarizer pre-processing	$\approx$ 25h	21min
Without LsaSummarizer	$\approx$ 40h	24min

The results show a considerable gain of time when using the school's computers instead of ours. It thus appears to be an essential step to train our models. All the fine-tuning experiments presented in this report have therefore been achieved thanks to the SSH protocol. The pre-processing step has slighlty reduced the fine-tuning time on the school's servers. We will however keep it in order to compare how well our model generates titles with and without it.

3.3 Results

For those final results, four epochs of fine-tuning were led over the 21041 documents of the training set. The convergence of the optimization is controlled by the stabilization of the loss function on the validation set (1500 documents). The recap chart 4 shows the impact on fine-tuning on the three previous models, with and without the extractive summarization pre-processing steps.

As expected with mt5, the use of fine-tuning had a huge impact on the quality of the generated titles. The titles which were one-word long before tuning are now way closer to the ones of the dataset. For camembert2camembert, fine-tuning the model has surprisingly reduced the ROUGE score, probably because of a mistake in our pipeline for which we lacked time to solve. Finally, BARThez is the model showing the best results as its fine-tuning led to a ROUGE score of 23.44% which corresponds to our best submission.

It is also interesting to notice that fine-tuning the dataset that has been pre-processed with extractive summarizers results in almost the same ROUGE score, for all models, as the dataset without pre-processing. We were doubting that this would occur because of the drop of performance before fine-tuning.

Table 4: Impact of the fine-tuning in the quality of our models

ROUGE score	**mt5-small**	**camembert2camembert**	**BARThez**
No fine-tuning, No extractive summarization	3.89%	**19.46%**	15.73%
No fine-tuning, LSA extractive summarization	1.65%	15.82%	14.08%
No fine-tuning, Bert extractive summarization	1.82%	18.47%	14.61%
Fine-tuning, No extractive summarization	**18.40%**	14.22%	**23.44%**
Fine-tuning, LSA extractive summarization	17.76%	12.18%	22.98%
Fine-tuning, Bert extractive summarization	17.71%	13.70%	22.94%

4 Conclusion

This news articles title generation project has been a way to delve into both extractive and abstractive summarization techniques.

Unfortunately, adding a pre-processing step of extractive summarization, whether it is with LSA or Bert summarizer, did not lead to any improvement of the final ROUGE score. However, it turned out to be a convenient way to work with shorter texts, allowing faster training of our models, without causing any significant loss in the ROUGE score.

The fine-tuning of pre-trained models from HuggingFace, using the school's servers for faster computations, has appeared to be an efficient way to learn the representation of the titles from our datasets and to generate similar summaries on articles from the test set. In particular, fine-tuning the BARThez model led to the most significant gain in the ROUGE score.

Further studies could delve into using sequence2sequence models in order to translate the texts in English. Then English-based models could be used to perform the title generation tasks (exactly as we did with French-based models), before translating the titles back to French. We hope that this could improve the scores as some English-based models may be better (pre-trained with bigger datasets) than the French ones for abstractive and extractive summarization tasks.

Appendix

.1 Description of the models used

.1.1 mt5

First, let us briefly describe the T5 model, introduced by Raffel et al. (2020). The new idea introduced by this model is the text-to-text pipeline from the beginning to the end. It seems fairly natural for summarization where we start from a text and we want the model to output a shorter text. However, for sentence classification for instance, the model is not going to predict 0 or 1 but really the strings "negative" or "positive". This full text-to-text approach explains why we need to add a prefix "summarize :" before our input text, that is add the task as a pre-string to keep this full text approach.

As a consequence, all tasks have the same training objective, which allows for effective fine-tuning among several tasks. T5 uses a basic encoder-decoder Transformer architecture, pre-trained using a masked objective where several tokens are replaced with a masked token.

The pre-training is unsupervised, a very useful approach in NLP regarding the mass of text available on the Internet, that we can just scrape to train the models.

Authors use roughly the same architecture as the original T5 model. The only difference is the use of GLU Variants (see Shazeer (2020)). In this paper, the FFN layers of the Transformers architecture (that is, two linear layers, with a ReLU activation function $Max(xW_1, 0)W2$ are replaced by GLU units. GLU consist of the component-wise product of two linear pro jections, one of which is first passed through a sigmoid function, that is $\sigma(xW) \otimes xV$. The FFN layer is now replaced by a GLU in place of the first linear layer i.e. $(\sigma(xW) \otimes xV)W_2$.

Authors build a multilingual dataset called mC4. During pre-training, they sample texts according to the distribution of languages in the corpus.

.1.2 CamemBERT

CamemBERT uses the RoBERTa architecture (see Liu et al. (2019)), which itself is based on BERT (Devlin et al. (2018)).

Let us recall that BERT is a multi-layer bidirectional Transformer encoder trained with a masked language modeling objective.

RoBERTa improves this original implementation by using dynamic masking, removing the next sentence prediction task, training with larger batches, on more data, and for longer.

The novelty of CamemBERT is to release a French-specific model, trained on open source corpora from the Oscar collection (that itself come from the Common Crawl database). Authors then use SentencePiece (Kudo and Richardson (2018)) as a subword tokenizer.

CamemBERT closely resembles RoBERTa, with its primary divergence lying in the adoption of whole-word masking and SentencePiece tokenization. Consequently, our training approach involves employing a masked language objective, wherein we predict the initial masked tokens through cross-entropy loss.

It takes the RoBERTa idea of dynamically mask tokens, Rather than statically fixing tokens throughout the entire dataset preprocessing stage.

Given our utilization of SentencePiece for corpus tokenization, the input tokens consist of a blend of complete words and subwords. Authors then use whole-word masking as some papers (see Joshi et al. (2020)) demonstrated that it improves performance.

Let us emphasize that for summarization we do not just use camemBERT. Instead, we use of the sequence to sequence model camemBERT2camemBERT (see `https://huggingface.co/mrm8488/camembert2camembert_shared-finetuned-french-summarization`).

.1.3 BARThez

BART (see Lewis et al. (2019)) is well-suited for Seq2Seq tasks since both its encoder and decoder are pretrained. Its encoder consists of a BERT architecture while its decoder is more alike to a GPT architecture.

BARThez (Eddine et al. (2020)) rely on BART and is well suited for generative tasks. It was specifically designed for seq2seq tasks in French.

The model is trained in a self-supervised way, where the input text is perturbated with some masks or some shuffle. The model was pretrained on a new dataset called OrangeSum.

.2 Using pipelines from HuggingFace

To use models described in part 1 and integrate them into a whole pipeline, we make use of the HuggingFace Transformers library (see Wolf et al. (2020) and `https://huggingface.co/docs/transformers/index`). This library enables easy download of models, inference using these models, as well as easy fine-tuning of these models.

The three main components of each model in this library are :

- Tokenizer, that converts input text into an array of integrers encoding tokens. In our applications we make use of the class AutoTokenizer that automatically loads a tokenizer implemented in Rust (see `https://huggingface.co/docs/tokenizers/index` for more details on the Tokenizer library). Of course for our fine tunings this tokenizer must be the one that was used for the pre-training.

- Transformer, that uses these sparse indices to build contextual embeddings

- Head i.e. last layers that are task specific. All Transformers models provided by this library are pretrained with a fixed head and can then be further fine-tuned with alternate heads. In our applications we make use of the class AutoModelForSeq2SeqLM heads that are specific to sequence-to-sequence tasks.

Once our model and tokenizer are loaded using AutoModel and AutoTokenizer, we make use of the Datasets library (see `https://huggingface.co/docs/datasets/index`) and more

specifically its map function to load our dataset and then preprocess it (performing tokenization...) We also add a DataCollatorForSeq2Seq in our pipeline that pads the inputs and the titles to the maximum length in the batch.

Then we make use of the Evaluate library (see `https://huggingface.co/docs/evaluate/index`) to compute the rouge score every 500 steps in our finetuning.

Finally, we finetune using classes Seq2SeqTrainingArguments to define our training arguments, and Seq2SeqTrainer to perform the finetuning.

Another way could have been to directly use a script like

```
python examples/pytorch/summarization/run_summarization.py
```

but we did not follow this approach for better readibility and modularity.

References

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

Eddine, Moussa Kamal, Antoine J-P Tixier, and Michalis Vazirgiannis, "Barthez: a skilled pretrained french sequence-to-sequence model," *arXiv preprint arXiv:2010.12321*, 2020.

Joshi, Mandar, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy, "Spanbert: Improving pre-training by representing and predicting spans," *Transactions of the association for computational linguistics*, 2020, *8*, 64–77.

Kudo, Taku and John Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

Lewis, Mike, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

Liu, Yinhan, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer, "Multilingual denoising pre-training for neural machine translation," *Transactions of the Association for Computational Linguistics*, 2020, *8*, 726–742.

_ , Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

Martin, Louis, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot, "CamemBERT: a tasty French language model," *arXiv preprint arXiv:1911.03894*, 2019.

Miller, Derek, "Leveraging BERT for extractive text summarization on lectures," *arXiv preprint arXiv:1906.04165*, 2019.

Padmakumar, Eswaran, "Extractive Text Summarization using Latent Semantic Analysis," *Indian Institute of Technology*, 2014.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever et al., "Language models are unsupervised multitask learners," *OpenAI blog*, 2019, *1* (8), 9.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, 2020, *21* (140), 1–67.

Shazeer, Noam, "Glu variants improve transformer," *arXiv preprint arXiv:2002.05202*, 2020.

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu, "A survey on deep transfer learning," in "Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27" Springer 2018, pp. 270–279.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, 2017, *30*.

Vazirgiannis, "Deep Learning Applications to Text Mining/NLP," *INF554 Machine and Deep Learning*, 2023-2024.

_ , "Pretrained Large Language Models," *INF582 Introduction to Text Mining and NLP*, 2023-2024.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz et al., "Transformers: State-of-the-art natural language processing," in "Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations" 2020, pp. 38–45.

Xue, Linting, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel, "mT5: A massively multilingual pre-trained text-to-text transformer," *arXiv preprint arXiv:2010.11934*, 2020.

Zhang, Jingqing, Yao Zhao, Mohammad Saleh, and Peter Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in "International conference on machine learning" PMLR 2020, pp. 11328–11339.