Abstractive Text Summarization

Mohammed Zaieda, Omar Ali

Computer Science and Engineering Department, School of Science and Engineering, The American University in Cairo

Introduction

Abstractive text summarization is the task of capturing the most important ideas from a long paragraph and providing a shorter precise summary that delivers the same message. Another summarization technique is extractive text summarization which refers to extracting the most important sentences from a paragraph. Text summarization has various applications including, but not limited to, medical reports summarization for faster analysis and decision making, media monitoring and surveillance tracking, data refinement and memory reduction.

Original Model (PEGASUS)

PEGASUS stands for Pre-training with Extracted Gap-sentences for Abstractive Summarization

PEGASUS masks multiple whole sentences rather than smaller continuous text spans.

Masking strategies:

- Masking random sentences.
- Masking the first m-sentences, also known as Lead.
- Masking the top m-scored sentences based on their importance using Rouge-1.

• Uses a transformer encoder to mask sentences, and a transformer decoder to produce the target text.

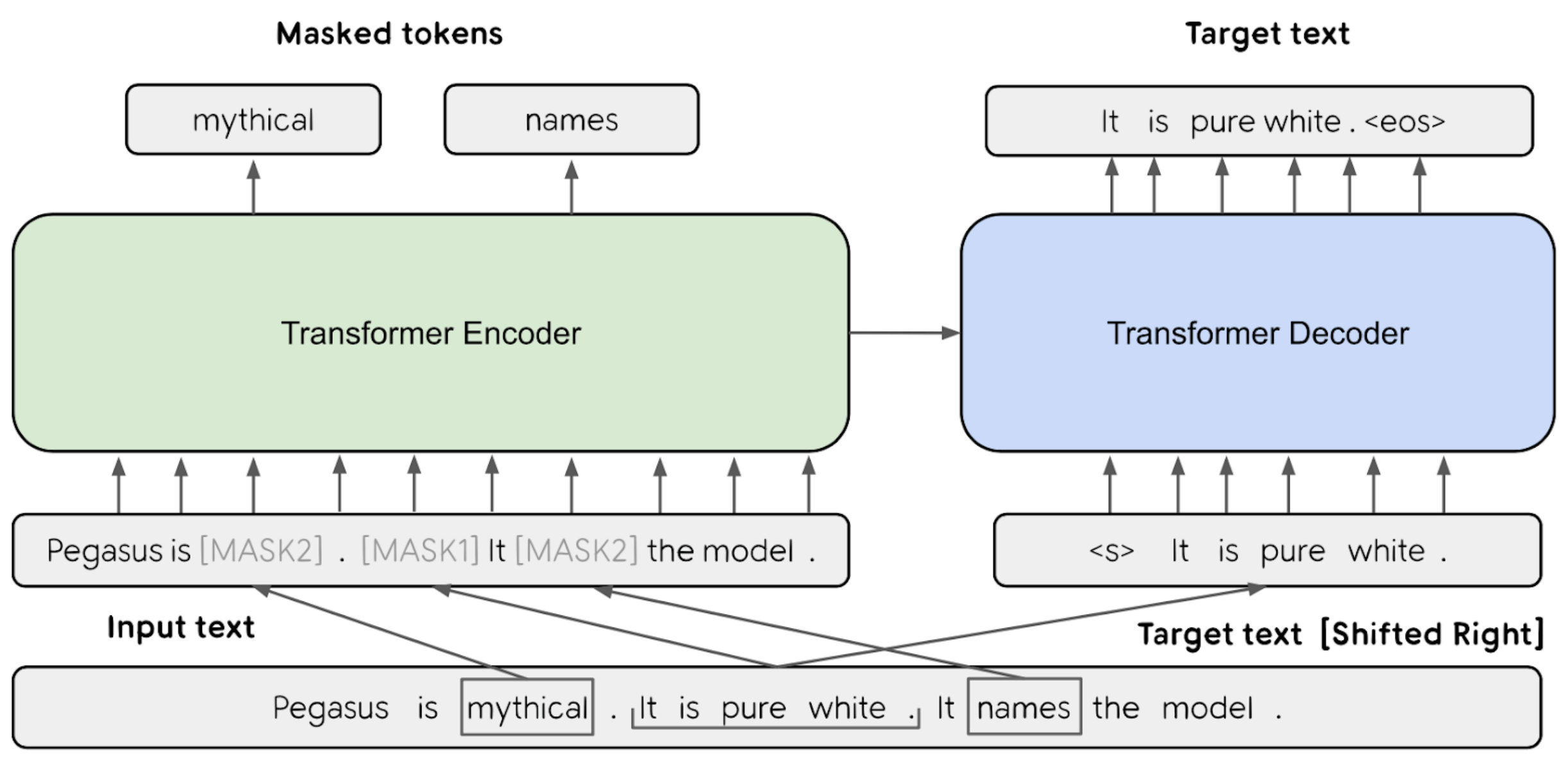
Masked tokens
Target text
mythical
names
It is pure white . <eos>
Transformer Encoder
Transformer Decoder
Pegasus is [MASK2] . [MASK1] It [MASK2] the model .
<s> It is pure white .
Input text
Target text [Shifted Right]
Pegasus is mythical . It is pure white . It names the model .

Figure 1. PEGASUS Model Endoder-Decoder Architecture

Dataset

Many datasets were created for the task of text summarization, such as CNN/DailyMail, BBC XSum, Reddit, WikiHow, and many more. For PEGASUS, two humongous text corpora were used to pre-train the model, namely C4 (750 GB dataset gathered from 350 M website) and Huge News (3.8 TB dataset consisting of 1.5 B article from different news websites). PEGASUS was then trained and tested on other 12 datasets and achieved state-of-the-art results. The dataset that we focused on was BBC Xsum which is composed of 226,711 archived BBC articles ranging from 2010 to 2017 where each article is prefaced with a professionally written introductory sentences serving as the summary of the article.

Our Contribution

We expand the base model (PEGASUS) by implementing data augmentation technique called Back Translation on the dataset. Data augmentation is typically used to reduce overfitting by generating additional training data using some techniques applied on the original training data. Back translation is one of the data augmentation techniques used for NLP tasks which involves translating some samples from the dataset into another language (e.g. from English to French) then translating it back to the original language. This will mostly generate a paraphrased sentence of the original one.

The original model did not use any data augmentation techniques as the training datasets were more than sufficient to prevent any overfitting to occur.

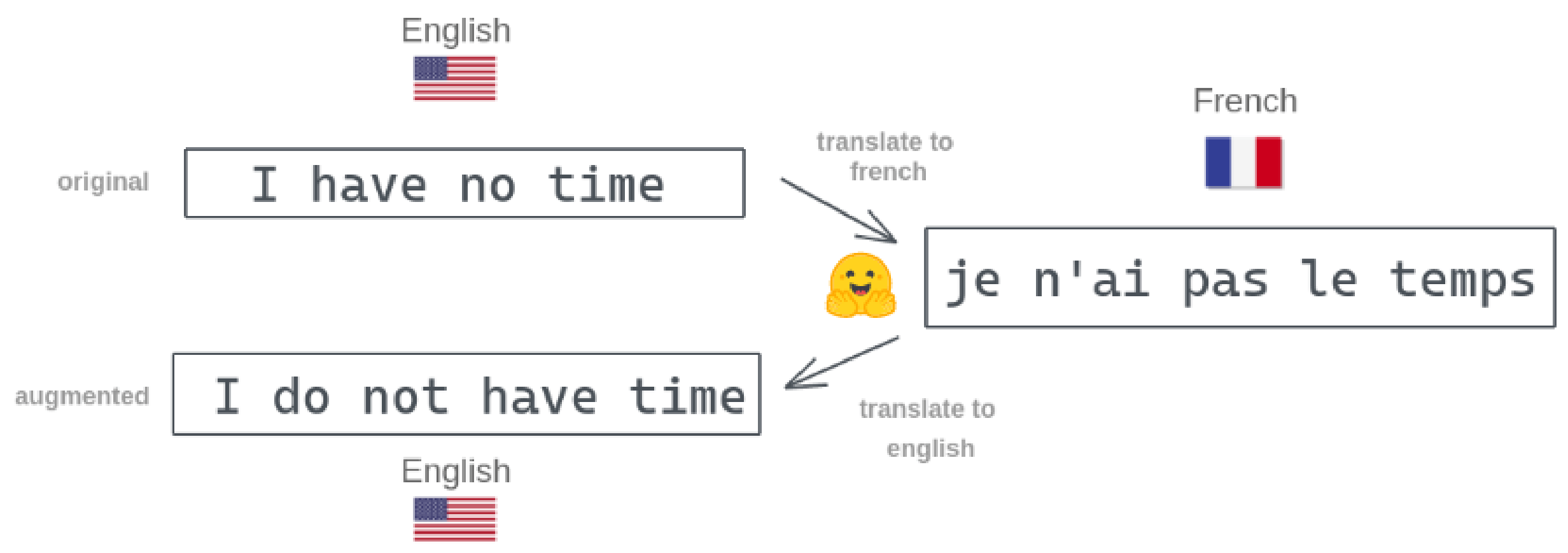

Figure 2. Data Augmentation using Back Translation

We also tested a different word embedding algorithm called Word2Vec. The word2vec algorithm learns word connections from a huge corpus of text using a neural network model. Once trained, the model can detect synonyms and recommend new words for a sentence. As the name suggests, word2vec associates each different word with a specific vector. The semantic similarity between the words can then be approximated via certain mathematical functions applied on their vectors, such as cosine similarity between vectors.

The original model used the positional sinusoidal word embedding which is another way of using words in a semantic way using sin and cos functions.

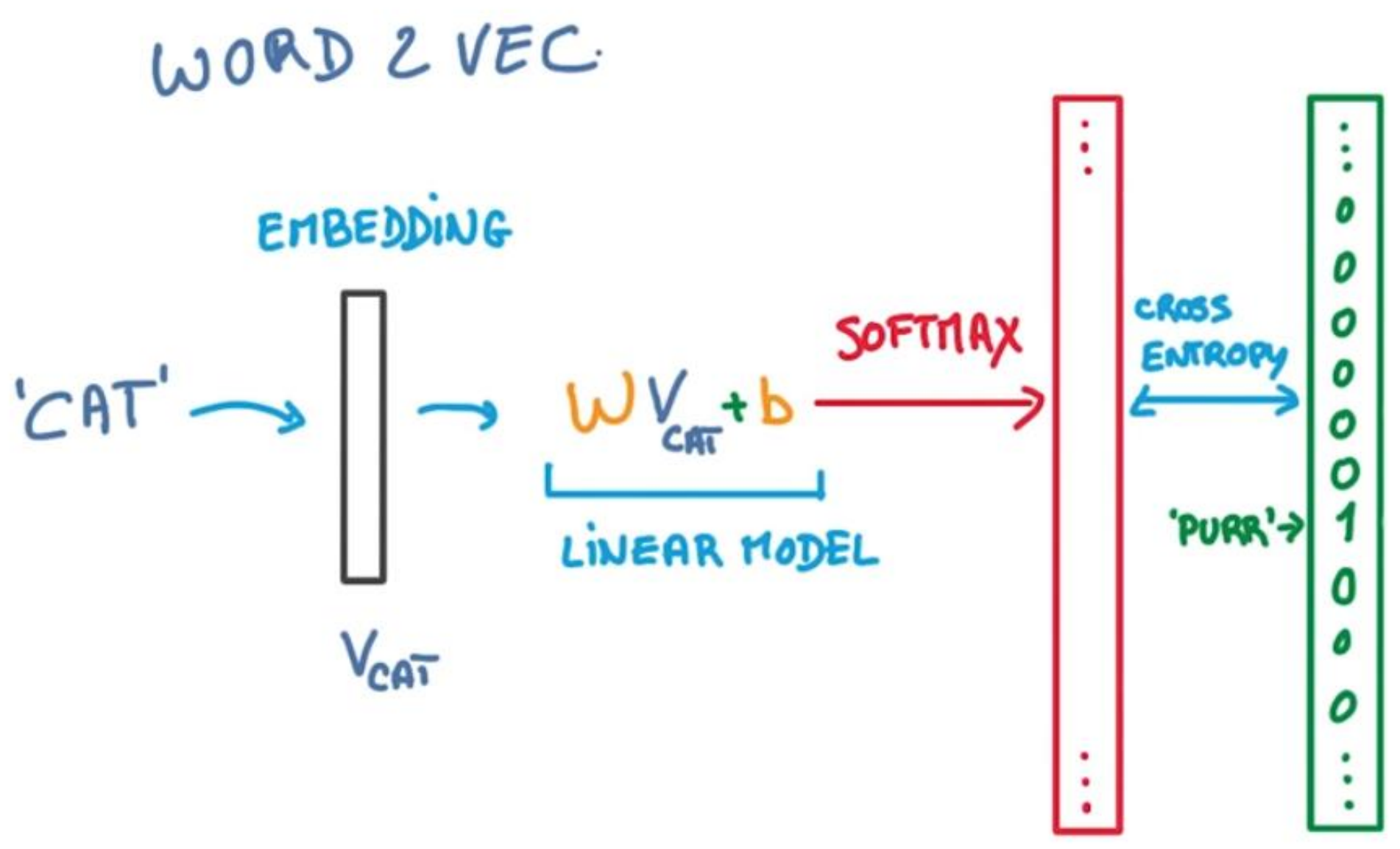

Figure 3. Word2Vec Model Architecture

Results

When it comes to the results, the paper ROUGE-2 score found was 24.56, however, the one generated from the pretrained transformer produced 1.934, this is due to reducing the training dataset size due to resources limitations. The first table shows the word2vec modification which as you can see is much lower. The data augmentation were very similar to the actual model.

Original Model (Paper-HN/Actual)	Modified Model (Actual)
24.56 / 1.934	0.6869
	0.8400
	1.2650

Original Model (Paper-HN/Actual)	Modified Model (Actual)
24.56 / 1.934	1.8761
	1.8902
	1.922

Discussion

As a reflection of the results, the actual used dataset was Xsum only, along with a massive reduction of the training dataset size which has effected the performance as shown in the results section. The ideal number of epochs as well was never reached as we consumed all of the credit used in the google cloud provided. The purpose of using the actual ROUGE-2 value from the pretrained model is to showcase a checkpoint that we can reflect upon in our experimentation. The original word embedding is preferred as the generated summary did not make sense as some generations.

The ROUGE score obtained does not represent the actual performance of our modifications as we limited the dataset size which affected the overall performance.

The data augmentation techniques is a plus, however, the only downside is that it is hard to sense the change as the training dataset size is limited.

We initially proposed performing data augmentation in hopes for preventing overfitting when dealing with model, however, after attempting to train, the resources allocated and the time were not feasible to actually test the data augmentation techniques. Although the score obtained from performing back translation were relatively close to the actual score value without any data augmentation.

Conclusions

In conclusion, we believe that if we were allocated the enough resources, interesting results would have been obtained. As such, For future work, we will attempt to keep the model running longer and perform hyperparameter tuning for the word embedding. We might also want to introduce a different type of dataset.

Contact

[name]
[organization]
[address]
[email]
[phone]

References

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.