Modified PEGASUS Large: Pre-training with Extracted Gap-sentences for Abstractive Summarization

Mohammed Zaieda
Department of Computer Science and Engineering
American University in Cairo
New Cairo, EG 11841
mzaieda@auceegypt.edu

Omar Ali
Department of Computer Science and Engineering
American University in Cairo
New Cairo, EG 11841
o_abdelbaset@aucegypt.edu

***Abstract*— When fine-tuning downstream NLP tasks like text summarization, recent work pre-training Transformers with self-supervised objectives on huge text corpora has demonstrated remarkable results. Pre-training objectives for abstractive text summarization, on the other hand, have not been investigated. In this paper, we suggest using a novel self-supervised model called Pegasus pretrained on Huge News corpus and implement modifications. Important sentences from an input document are removed/masked in PEGASUS and created as one output sequence from the other phrases, comparable to an extractive summary.**

I. Introduction

An The goal of text summarising is to create accurate and succinct summaries from a source document (s). Abstractive summarising, as opposed to extractive summarization, which just replicates informative pieces from the input, can produce new terms. A good abstractive summary is linguistically fluent and includes the most important information in the input. We investigate pre-training objectives for abstractive text summarization and assess them using the BBC XSum dataset. To summarize our contribution:

- Pretrain over BBC XSum dataset and finetune on a Sequence-to-Sequence model.
- Change the original word embedding from positional sinusoidal embedding to word2vec.
- Perform back translation data augmentation over the training dataset.

II. Related Works

T5 extended the text-to-text framework to a range of NLP tasks and demonstrated the benefits of increasing model size (to 11 billion parameters) and pre-training corpus size, introducing C4, a big text corpus obtained from Common Crawl.

BART, to pre-train sequence-to-sequence models, BART included a denoising autoencoder. BART learns to recreate the original text after corrupting it with an arbitrary noising function.

Unlike BART, and T5, PEGASUS masks numerous complete phrases rather than shorter continuous text spans. In order to achieve our end goal, we chose phrases deterministically rather than randomly. PEGASUS, like T5, does not reconstruct whole input sequences, instead generating only the masked sentences as a single output sequence. We do not analyse NLU classification tasks in this study since we are only interested in downstream summarization activities.

III. Datasets

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.

IV. Original Model Architecture

A. GSG(Gap Sentences Generation)

We believe that selecting a pre-training goal that is more similar to the downstream job results in better and faster fine-tuning performance. Our suggested pre-training objective entails creating summary-like language from an input document, which is appropriate given our intended usage for abstractive summarising. In the absence of abstactive summaries, we construct a sequence-to-sequence self-supervised aim to use vast text corpora for pre-training. Pre-training as an extractive summarizer is a basic alternative; however, this technique would only train a model to duplicate phrases, making it unsuitable for abstractive summarising. We choose sentences that appear to be important/principal to the document to get even closer to a summary. The resulting goal combines the experimentally established benefits of masking with a foreshadowing of the downstream task's shape.

B. MLM (Masked Language Model)

Following BERT, we choose 15% of the tokens in the input text to be (1) 80 percent of the time replaced by a mask token [MASK2], (2) 10% of the time replaced by a random token, or (3) 10% of the time left intact. We use MLM to pre-train the Transformer encoder either alone or in conjunction with GSG. When MLM is the only pre-training goal, the Transformer decoder fine-tunes downstream jobs by sharing all parameters with the encoder.

C. Pretraining Corpus

Huge News is a 1.5 billion-article (3.8-terabyte) dataset gathered from news and news-like websites between 2013 and 2019. A whitelist of domains was vetted and utilised to seed a web-crawler, ranging from high-quality news publishers to lower-quality sites like high-school newspapers and blogs. Only the primary article text was retrieved as plain text using heuristics to detect news-like items.

V. Modified Model Architecture

A. Word Embedding

Word placement and order are crucial aspects of any language. They define a sentence's syntax and consequently its real meaning. The order of words is taken into consideration by recurrent neural networks (RNNs), which parse a phrase word by word in a sequential fashion.

The authors' proposed encoding is a simple yet brilliant solution that meets all of those requirements.
To begin with, it is not a single number.
Instead, it's a d-dimensional vector containing data on a single sentence position. Second, this encoding is not included into the model. This vector is instead utilised to provide each word with information about its place in a sentence. To put it another way, we improve the model's input by injecting the word order.

Our contribution to this word embedding is that we will be changing it to the word2vec word embedding. Hierarchical softmax and/or negative sampling can be used to train a word2vec model. The hierarchical softmax approach reduces calculation by using a Huffman tree to represent the conditional log-likelihood a model wants to maximise. By reducing the log-likelihood of sampled negative cases, the negative sampling method addresses the maximising issue.

B. Data Augmentation

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 invloves translating some samples from the dataset into another languauge (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.

VI. Results and Conclusion

When it comes to the results of our contribution:

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

Above there are two tables, where the first indicates the results of implementing the word2vec word embedding. The first value one the left is the value obtained from the original paper while the value after slash is the value obtained from running the model on the pretrained colab interface. The three values on the rights are three runs with different training dataset size and number of epochs. Similarly with the second table but with data augmentation.

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. In conclusion, In conclusion, 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.

References

1. An approach to abstractive text summarization. IEEE Xplore. (n.d.). Retrieved March 19, 2022, from https://ieeexplore.ieee.org/document/7054161
2. Brownlee, J. (2020, August 25). How to improve performance with transfer learning for Deep Learning Neural Networks. Machine Learning Mastery. Retrieved March 19, 2022, from https://machinelearningmastery.com/how-to-improve-performance-with-transfer-learning-for-deep-learning-neural-networks/
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, December 6). Attention is all you need. arXiv.org. Retrieved March 19, 2022, from https://arxiv.org/abs/1706.03762
4. Project, S. N. A. (2017, May 1). Amazon Fine Food Reviews. Kaggle. Retrieved March 19, 2022, from https://www.kaggle.com/snap/amazon-fine-food-reviews
5. Pennington, J. (n.d.). NLP Standard. Glove: Global vectors for word representation. Retrieved March 19, 2022, from https://nlp.stanford.edu/projects/glove/

6. Word2vec ： Tensorflow Core. TensorFlow. (n.d.). Retrieved March 19, 2022, from https://www.tensorflow.org/tutorials/text/word2vec

7. Papers with code - bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension | Papers With Code. (n.d.). Retrieved March 19, 2022, from https://paperswithcode.com/paper/bart-denoising-sequence-to-sequence-pre

8. notebooks/BERT2BERT_for_CNN_Dailymail.ipynb at master · patrickvonplaten/notebooks · GitHub

9. notebooks/RoBERTaShared_for_BBC_XSum.ipynb at master · patrickvonplaten/notebooks · GitHub

10. https://journalofbigdata.springeropen.com/track/pdf/10.1186/s40537-021-00492-0.pdf