Final Report on Project Pollock: Generating Abstracts for Academic Papers

Nicholas Almy
nalmy@email.unc.edu

Edward Baker
eddean@email.unc.edu

Ira Chandramouli
ira01@email.unc.edu

Jackson Davis
jdavis19@email.unc.edu

Zachary Gordon
zgordo@email.unc.edu

April 29, 2023

1 Introduction

The goal of Project Pollock is to develop software that generates abstracts for academic papers utilizing pre-trained transformer models. The primary motivation behind this project is to create a valuable tool that aids in the generation of accurate and concise abstracts while simultaneously exploring the limitations of pre-trained transformer models.

2 Related Work

Previous research in this domain includes the introduction of powerful transfer language models by Element AI, MILA (Montréal Institute for Learning Algorithms), and Université de Montréal, which effectively summarize lengthy scientific articles, surpassing the performance of traditional seq2seq approaches.[4] These models utilize advanced techniques such as self-attention mechanisms and transfer learning. Another notable project is Abstract2Title, a Seq2Seq model derived from T5-base that generates titles for machine learning papers using abstracts as input.[3] This model demonstrates the feasibility of fine-tuning pre-trained models for specific tasks. Moreover, GPT-3, a transformer-based deep learning model developed by OpenAI, demonstrates the ability to produce highly detailed information when given a sentence-format prompt, showcasing the versatility of transformer models.

3 Data

We acquired data from academic papers in the arXiv database, a large repository ($\sim$4TB) of pre-print articles in PDF format.[5] The raw abstracts were extracted from the metadata, and full articles were accessed using a PDFToText method in Python. To prepare the input for the model, we employed techniques to remove abstracts from the raw text. The articles were sourced using a Google Cloud server with a callable API, and additional processing was applied to clean titles, headers, figures, and names as needed.

4 Data Cleaning

We targeted articles in the arXiv database with clearly defined outlines, meaning papers containing labeled abstract, introduction, and references sections. We developed an algorithm to parse the articles, ensuring they contained this structure. The algorithm checks that an article has the words 'abstract', 'introduction', and 'references' in the correct order and discards the article if it does not. This algorithm served a two-fold purpose. It allowed us to separate the body text from the rest of the content in the article and ensured that the majority of the articles were in English since we searched for English keywords. After this, we extracted the body of text and removed all emails, in-text citations, and all characters that were not alphanumeric, whitespace, or punctuation.

We noticed that we were getting weird artifacts as a consequence of converting the pdfs to text and were seeing many words squished together or split apart. In order to solve this, we employed the use of wordninja- a python package for probabilistically splitting combined text into individual words. We combined all text into a single string and then let wordninja split the text again to determine the correct words without them being squished or separated. This produced greatly improved results. However, this advanced cleaning slowed down the training time for the model significantly and ultimately was left out for the purposes of the project. If we had more time to train our model, we would hope to use a more advanced cleaning method like this for better results.

5 Model and Architecture

We chose the Longformer Encoder-Decoder (LED) model, specifically the led-base-1638 variant developed by AllenAI,[1] as our baseline model. The LED model extends the transformer architecture by implementing a sliding window mechanism that increases the context window size to approximately 8000 tokens. This feature enables the model to process and generate longer text sequences, making it well-suited for generating academic paper abstracts. The model employs a combination of fixed-size sliding windows, dilated windows, and global attention.

6 Experiments and Results

To enhance the computational power and efficiency of the training process, we utilized the UNC LongLeaf Cluster. This resource facilitated rapid iteration and refinement of our experiments and enabled us to train our models more efficiently.

For user interaction and demonstration purposes, we established a Gradio front-end interface, which allowed users to input academic papers and receive generated abstracts as output.

To evaluate the quality of the generated abstracts and compare the performance of the models, we used standard evaluation metrics including BLEU, Rouge1, Rouge2, Rougel, and hand-grading. BLEU measures precision by comparing the number of n-grams in the machine-generated text that also appear in the original text, while Rouge calculates recall by examining the number of n-grams in the original text that appear in the generated text. Rouge1, Rouge2, and Rougel consider unigrams, bigrams, and the longest common subsequence, respectively. We used Rouge2 to calculate the loss in our model.

Our baseline model's performance was evaluated using a combination of BLEU and Rouge metrics, as well as hand-grading, and obtained Rouge scores of Rouge1: 0.356, Rouge2: 0.100, and RougeL: 0.194, and a BLEU score of 0.027. These scores suggest that the model is capable of generating abstracts that closely resemble the reference abstracts in terms of content and structure.

We conducted additional experiments to enhance the model's performance, building upon the baseline model. These experiments involved varying the training set sizes and extending the training time. We trained two additional models, one using a sample of 50,000 articles over 11 epochs and another using all training samples over 3 epochs. Both models showed improved accuracy compared to the baseline. While neither model finished training before the SLURM time limit expired, their higher training efficiencies (better optimized evaluation steps) allowed them to complete more epochs and achieve higher results. The new scores for Rouge1, Rouge2, RougeL, and BLEU are 0.380, 0.117, 0.210, and 0.0352, respectively.

We also employed a hand-grading process to evaluate the quality of the generated abstracts. The process involved scoring the abstracts on a scale of 1 to 5 across five criteria: coherence, relevance, accuracy, results/findings, and conclusions/implications, with 1 being poor and 5 being excellent. The average hand-grading score was 2.85 out of 5, indicating that there is still room for improvement in the quality of the generated abstracts. However, our evaluation took into consideration not only content but also writing quality and the extent to which the abstracts accurately and coherently captured the original papers' essence.

7 Conclusion

In summary, our work on Project Pollock has shown that pre-trained transformer models have the potential to generate high-quality abstracts for academic papers. We started by cleaning the arXiv text data, then trained a baseline model and evaluated its performance using BLEU, Rouge, and hand-grading metrics. The results were promising,

with the model generating abstracts that closely resembled the reference abstracts in terms of content and structure.

We further conducted experiments to enhance the model's performance, which involved varying the training set sizes and extending the training time. The experiments showed that by increasing the training set sizes and training time, we can improve the accuracy of the model. The models that we trained using a sample of 50,000 articles over 11 epochs and all training samples over 3 epochs both exhibited improved accuracy compared to the baseline model.

While there is still room for improvement in data cleaning, training larger models with more examples, and optimizing the training process, our work has shown that pre-trained transformer models can be effective in generating accurate and concise abstracts for academic papers. Our ongoing experiments aim to further improve the model's performance by using higher quality training data, scaling up the training process, and optimizing the training process.

Overall, our work on Project Pollock provides valuable insights into the capabilities of pre-trained transformer models for generating abstracts and opens up opportunities for future research in this area. We hope that our findings will inspire further work on developing more sophisticated models that can generate abstracts with even higher levels of accuracy and coherence.

8 Contributions

Each member of the team contributed to the development and execution of Project Pollock. Nicholas helped setup the longleaf cluster and monitored the training of the model, he also helped with evaluation and Gradio. Edward implemented the cleaning and pre-processing algorithms, helped with the Gradio front-end, and evaluation. Ira contributed to the presentations, evaluation and final report. Jackson also worked with Edward to implement the cleaning and pre-processing algorithms, he also assisted with evaluation and experimentation. Zach was responsible for data collection, evaluation, and model fine-tuning.

References

1. Longformer Encoder-Decoder: https://huggingface.co/allenai/led-base-16384
2. BLEU and Rogue: https://clementbm.github.io/theory/2021/12/23/rouge-bleu-scores.html
3. Abstract2Title: https://github.com/nerdimite/abstract-to-title-generator
4. https://syncedreview.com/2019/09/23/transformer-based-language-model-writes-abstracts-for-scientific-papers/
5. arXiv: https://www.kaggle.com/datasets/Cornell-University/arxiv
6. Longformer paper: https://arxiv.org/abs/2004.05150