THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

Final Project Report

Chirag Lakhanpal

The George Washington University

DATS 6312: Natural Language Processing

Professor Amir Jafari

Table of Contents

Introduction

This report covers a concentrated exploration in the topic of legal text analysis, especially in the domain of text production. It describes the application and fine-tuning of the GPT-2 model for producing legal texts using a subset of the legal_contracts dataset that comprises 10% (65083 records) and 1% (6508 records). This technique tries to capitalize on GPT-2's sophisticated capabilities while adapting it to the complex and formal structure of legal language.

This project is significant because it has the potential to contribute legal document drafting and advisory creation through automated text generation. The project explores new possibilities in the efficiency and accuracy of legal text processing by fine-tuning GPT-2 using a properly selected legal dataset.

Dataset

Link - https://huggingface.co/datasets/albertvillanova/legal_contracts

For training the GPT-2 model in legal text generation, this project utilized a carefully selected subset of the albertvillanova/legal_contracts dataset. From the original 650,833 records (33 GB), a 10% subset (65,083 records, 2.87 GB) was chosen for the main training phase, balancing data diversity with computational efficiency. Additionally, a 1% subset (6,508 records, 260 MB) was used specifically for hyperparameter tuning, optimizing model performance without excessive computational demand.

Figure 1: First record for the dataset depicting Employment Non-Compete Agreement.

Individual Work (My contribution and Work Description)

My contribution in this project was varied, covering the building of a Streamlit application to illustrate the model's capabilities in a real-world setting as well as the fine-tuning of the GPT-2 model for legal text synthesis.

1. Fine Tuning GPT2 Model

- Model and Tokenizer Setup: The project started with setting up the GPT-2 model and tokenizer. GPT-2, which is well-known for its powerful language production skills, was chosen for its ability to handle complicated language structures, such as those found in legal writings. I also tried BERT and DistilBERT but was unable to get them to function.

2. Data Preparation

- Dataset Processing: Normalization of text from the albertvillanova/legal_contracts dataset was a vital step in the preprocessing procedure. To ensure text quality and uniformity, non-ASCII characters were removed throughout this procedure. To guarantee consistency, the text was transformed to lowercase.
- Regular Expression Usage: Regular expressions were extensively used for cleaning the data, including.
    i. Replacing sequences of dashes with spaces
    ii. Removing extraneous newline, carriage return, and tab characters
    iii. Stripping excessive whitespace.
    iv. Spell checking the words. Due to computation limitation this could not work.
    v. NER on names, person, and organizations. Again, due to the sheer amount of the data, it got computationally expensive.

```python
preprocess_function(examples):

    # Convert to lowercase and remove non-ASCII characters
    processed_text = [re.sub(r'[^\x00-\x7F]+', ' ', text).lower() for text in examples['text']]

    # Remove extra "---" characters
    processed_text = [re.sub(r'---+', ' ', text) for text in processed_text]

    # Remove newlines, tabs, and carriage returns
    processed_text = [text.replace('\n', ' ').replace('\r', ' ').replace('\t', ' ') for text in processed_text]

    # Remove extra whitespace
    processed_text = [re.sub(r'\s+', ' ', text).strip() for text in processed_text]

    # Remove numbers
    processed_text = [re.sub(r'\b\d+\b', '', text) for text in processed_text]

    # anonymized_text = []
    # for text in processed_text:
    #     doc = nlp(text)
    #     for ent in doc.ents:
    #         if ent.label_ in ["PERSON", "ORG", "GPE"]:
    #             text = text.replace(ent.text, f'[{ent.label_}]')
    #     anonymized_text.append(text)

    # corrected_text = []
    # for text in anonymized_text:
    #     blob = TextBlob(text)
    #     corrected_text.append(blob.correct().string)

    eturn tokenizer(processed_text, truncation=True, padding='max_length', max_length=512)
```

Figure 2: Python function for preprocessing text data, showing various steps including normalization, cleaning, and tokenization.

3. Training Process and Hyper Parameter tuning

- **Epoch Configuration**: For the smaller dataset, epochs were set at **1, 10, and 50**. For the larger dataset, **25 epochs were used,** which took around ~45 mins per epoch, balancing depth of learning with computational efficiency.
- **Batch Size Variation**: Two different batch sizes, 8 and 16, were experimented with. I tried with bigger batch sizes starting from 128 and reducing it to 32 in the power of 2, however I encountered "CUDA out of memory" error.
- **Optimizer Selection**: For the training used two different optimizers- **AdamW and SGD (Stochastic Gradient Descent). AdamW.**
- **Learning Rate Experimentation**: Multiple learning rates were tested: **5e-5, 1e-5, 5e-4, 5e-3, and 5e-2**. This range allowed for observing how the model responded to both subtle and significant changes in learning rate, influencing the speed and stability of the learning process.
- **Gradient Clipping**: To address the potential issue of exploding gradients, gradient clipping was incorporated. This technique involved capping the gradients during backpropagation to a predefined range, ensuring they did not exceed manageable levels.

4. Streamlit Application Development

- The Streamlit application for this project is structured into two primary scripts: one for the chat functionality and the other for the main application, featuring a landing page for task selection. Here's a brief overview of each aspect:
- Landing Page and Navigation: Users are first directed to a landing page offering task selections such as document summarization, classification, or initiating a legal chat. This is facilitated through a tabbed layout for easy navigation.
- Custom Styling: I applied custom CSS styling, enhancing the visual appeal and user experience.
- Input Sanitization: A crucial aspect of the app is the sanitization of user inputs using regular expressions, ensuring the inputs are clean and concise for optimal model response.

5. Things That Did Not Work

- In this project, an initial attempt to create a Q&A style chatbot using the nguha/legalbench dataset, which contains 162 specialized tasks, faced significant challenges.
- Computational Power Limitations: Initially, I tried to process the entire dataset in large batches for model training. This approach quickly led to out-of-memory errors, indicating that the dataset's size and complexity were too demanding for the available computational resources.
- Modular Approach Issues: Shifting to a modular strategy, where the dataset was fed one task at a time, did not resolve these memory issues. It became clear that both the batch size and the individual task complexity contributed to the computational challenges.
- Dataset Specificity: 'Yes' or 'no' responses to specific legal questions made up most of the sample. Because of its narrow focus, the chatbot was less useful to a broader audience because it was unable to adequately respond to a larger variety of legal inquiries.

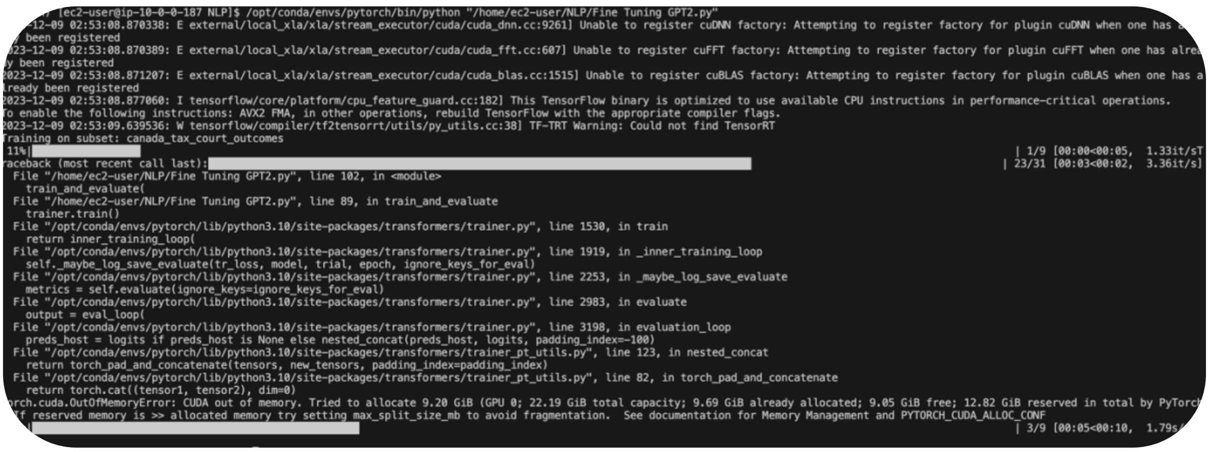


Figure 3: Screenshot of a CUDA out-of-memory error encountered during model training.

Results and Summary

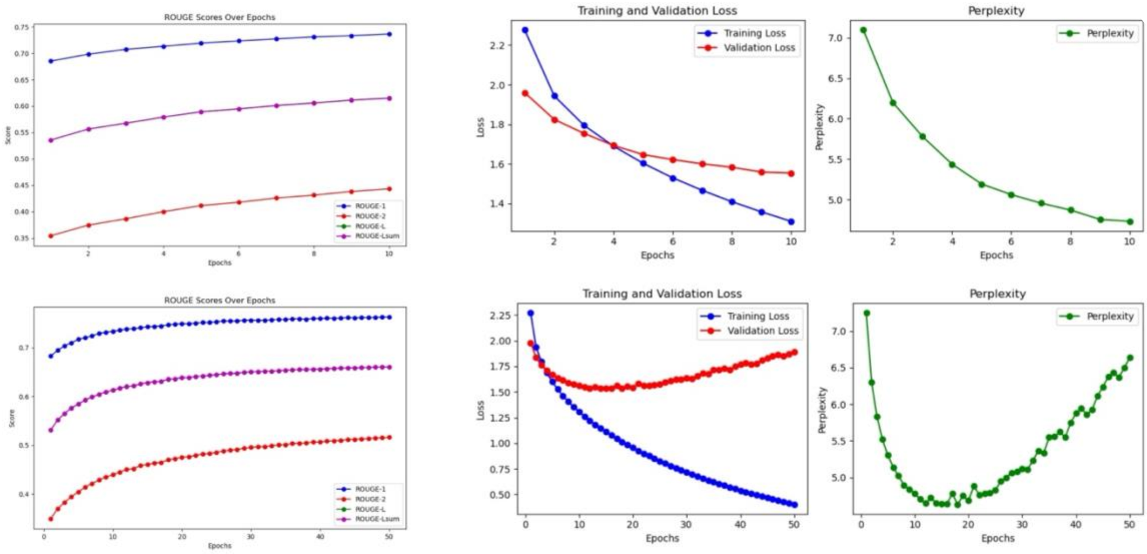


Figure 4: Comparative analysis of ROUGE scores (left), training and validation losses, and perplexity (right) over different numbers of training epochs on a small dataset (~300 MB).

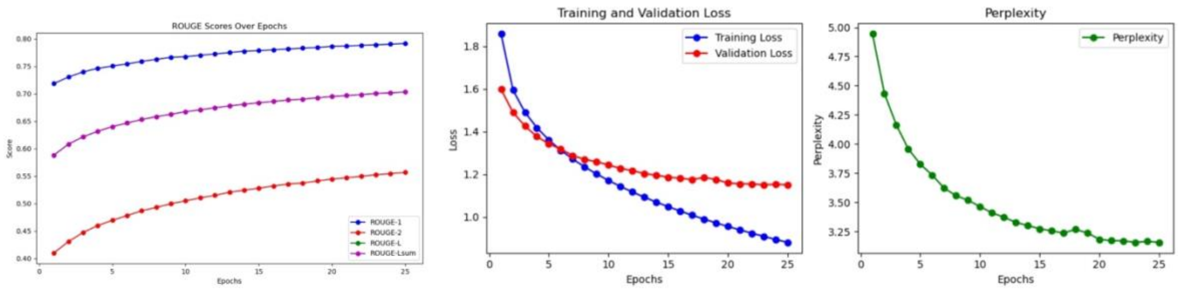


Figure 5: Comparative analysis of ROUGE scores (left), training and validation losses, and perplexity (right) over different numbers of training epochs on a large dataset (~3 GB).

The figures displayed represent the results of training a text generation model across different epochs, as evidenced by ROUGE scores, training and validation losses, and perplexity metrics.

1. ROUGE Scores

- ROUGE (Recall-Oriented Understudy for Gisting Evaluation) scores are a set of metrics for evaluating automatic summarization and machine translation software. They compare the overlap of n-grams, word sequences, and word pairs between the computer-generated output and a set of reference texts. Higher scores indicate better performance, with ROUGE-1, ROUGE-2, and ROUGE-L representing the overlap of unigrams, bigrams, and the longest common subsequence, respectively.
- Top Graphs: The top left graph illustrates an improvement in ROUGE scores across all three metrics over 10 epochs. The consistent upward trend across ROUGE-1, ROUGE-2, and ROUGE-L sum indicates the model's increasing proficiency in generating text closely matching the reference texts in terms of content and structure.

2. Training and Validation Loss

- Loss is a measure of how far the model's predictions are from the actual outcomes. Lower loss values correspond to better model performance.
- Top Middle Graphs: The training loss (blue) and validation loss (red) both show a downward trend over 10 epochs, suggesting that the model is learning effectively and generalizing well to unseen data.

3. Perplexity

- Perplexity is a measurement of how well a probability distribution or probability model predicts a sample. In the context of language models, lower perplexity indicates a better predictive performance.
- Top Right Graph: A decreasing perplexity trend over 10 epochs is observed, implying the model's improving ability to predict the next word in the sequence.

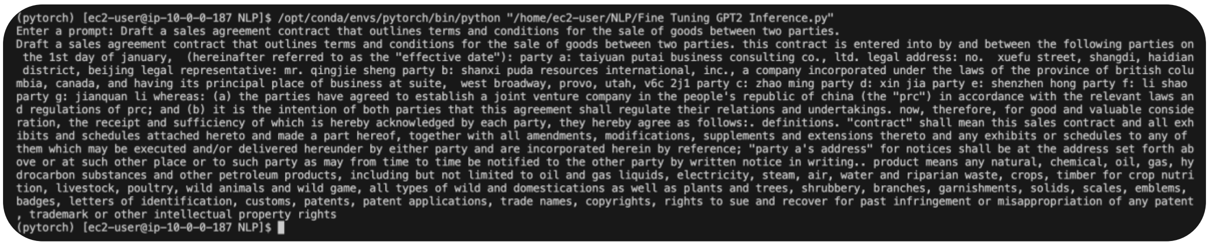
4. Observations from Extended Epoch Training

- When extending the training to 50 epochs, the graphs show distinct trends.
- Bottom Graphs: The ROUGE scores continue to improve, although they begin to plateau, indicating that the model might be approaching its optimal performance. The training and validation loss graphs depict a notable divergence after around 20 epochs, potentially suggesting overfitting where the model learns the training data too

well and may not perform as effectively on new, unseen data. The perplexity initially decreases, reflecting improved model predictions, but then increases dramatically, further implying that the model's performance on the validation set is worsening, possibly due to overfitting.

Best model Parameters

Parameter	Best Value	Experimental Values
Learning rate	5e-5	5e-5, 1e-5, 5e-4, 5e-3, and 5e-2
Optimizer	AdamW	AdamW and SGD
Batch Size	8	8, and 16 (Working)
Momentum (For SGD)	0.9	0.9
Epochs	10	1,10,25,50
Number of Workers (Data Loading)	8	4, 8, and 16
Dataset Size	-	10% (65083 records) and 1% (6508 records)



```
(pytorch) [ec2-user@ip-10-0-0-187 NLP]$ /opt/conda/envs/pytorch/bin/python "/home/ec2-user/NLP/Fine Tuning GPT2 Inference.py"
Enter a prompt: Draft a sales agreement contract that outlines terms and conditions for the sale of goods between two parties.
Draft a sales agreement contract that outlines terms and conditions for the sale of goods between two parties. this contract is entered into by and between the following parties on
 the 1st day of january,  (hereinafter referred to as the "effective date"): party a: taiyuan putai business consulting co., ltd. legal address: no.  xuefu street, shangdi, haidian
 district, beijing legal representative: mr. qingjie sheng party b: shanxi puda resources international, inc., a company incorporated under the laws of the province of british colu
mbia, canada, and having its principal place of business at suite,  west broadway, provo, utah, v6c 2j1 party c: zhao ming party d: xin jia party e: shenzhen hong party f: li shao
party g: jianquan li whereas: (a) the parties have agreed to establish a joint venture company in the people's republic of china (the "prc") in accordance with the relevant laws an
d regulations of prc; and (b) it is the intention of both parties that this agreement shall regulate their relations and undertakings. now, therefore, for good and valuable conside
ration, the receipt and sufficiency of which is hereby acknowledged by each party, they hereby agree as follows:. definitions. "contract" shall mean this sales contract and all exh
ibits and schedules attached hereto and made a part hereof, together with all amendments, modifications, supplements and extensions thereto and any exhibits or schedules to any of
them which may be executed and/or delivered hereunder by either party and are incorporated herein by reference; "party a's address" for notices shall be at the address set forth ab
ove or at such other place or to such party as may from time to time be notified to the other party by written notice in writing.. product means any natural, chemical, oil, gas, hy
drocarbon substances and other petroleum products, including but not limited to oil and gas liquids, electricity, steam, air, water and riparian waste, crops, timber for crop nutri
tion, livestock, poultry, wild animals and wild game, all types of wild and domestications as well as plants and trees, shrubbery, branches, garnishments, solids, scales, emblems,
badges, letters of identification, customs, patents, patent applications, trade names, copyrights, rights to sue and recover for past infringement or misappropriation of any patent
, trademark or other intellectual property rights
(pytorch) [ec2-user@ip-10-0-0-187 NLP]$
```

Figure 6: Inference on custom model trained on large dataset.

Conclusions

- ROUGE scores improved consistently across 10 epochs, indicating better alignment with reference texts in the model's output.
- Both training and validation loss decreased over 10 epochs, suggesting effective learning and generalization to unseen data.
- Perplexity metrics showed a downward trend over 10 epochs, reflecting the model's improved predictive accuracy.
- In extended training up to 50 epochs, ROUGE scores began to plateau, suggesting a limit to the model's improvement in text generation.

- Divergence between training and validation loss after 20 epochs indicated potential overfitting, where the model excessively learned from the training data at the expense of generalization.
- A dramatic increase in perplexity after an initial decrease during extended training suggested diminishing returns in predictive performance, possibly due to overfitting.

Code Contribution

Total lines of code = 427 (Streamlit + train). Modified lines = ~80. Added lines = ~40. Percentage = 89.66%

References

1. Alammar, J. (n.d.). The Illustrated GPT-2 (Visualizing Transformer Language Models). Retrieved from https://jalammar.github.io/illustrated-gpt2/
2. Streamlit Docs. (n.d.). Build a ChatGPT-like app. Retrieved from https://docs.streamlit.io/knowledge-base/tutorials/build-conversational-apps#build-a-chatgpt-like-app
3. OpenAI. (n.d.). ChatGPT.
4. Papers with Code. (n.d.). GPT-2. Retrieved from https://paperswithcode.com/method/gpt-2

5. Image References:

    a. ResearchGate. (n.d.). Structure of the applied GPT-2 medium architecture. Retrieved from https://www.researchgate.net/figure/Structure-of-the-applied-GPT-2-medium-architecture_fig2_365625866
    b. Alammar, J. (n.d.). The Illustrated GPT-2 (Visualizing Transformer Language Models). Retrieved from https://jalammar.github.io/illustrated-gpt2/