Anna Kopec
5/9/2025
Stefan Kaufmann
LING 3000Q

LING 3000Q Final Project

For my final project, I decided to implement an instance of the BERT model using the HuggingFace Transformers library. BERT stands for *Bidirectional Encoder Representations from Transformers*. It is a model for NLP tasks developed by Google, and it was trained on a dataset of 3.3 billion words.

Transformer models are neural networks that are based around 'self-attention' mechanisms. Inputs are encoded into embeddings, and processed through many stacked transformer blocks. Each block contains multi-head self-attention layers that allow the model to weigh the importance of every other token in the sequence when encoding each input token. Tokens are weighed based on their similarity to the current token, which is calculated with a dot product. The results from these layers are then passed through feedforward layers and then through a softmax output layer.

Attention in transformers comes in the form of attention heads, which continuously weigh the importance of previous tokens relative to the current input token. In a model like BERT, there exist multiple attention heads operating in parallel, each one focusing on identifying different types of linguistic relationships, such as syntax or semantics. BERT's bidirectional structure also allows it to attend to both previous and future tokens, unlike causal transformers models that can only look towards the past.

BERT's bidirectional structure makes it ideal for handling NLP tasks that are very context-dependent. One common task handled by models like BERT is masked language modelling (MLM). I decided to train BERT for an MLM, where the model is given sentences with certain words hidden under a special MASK token. The model must predict what words go in the hidden positions.

To begin, I started a notebook in Google Colab. I imported a special instance of BERT from HuggingFace specifically pretrained for MLM. I also used a tokenizer to encode all the text from the novel, 'Moby Dick' into a numerical format suitable for BERT to process. I imported the text from Project Gutenberg and then cleaned it using regular expressions. I then split it into a large list of sentences, where words were represented numerically in arrays, and batched together with other sentences.

I implemented a custom MLM Dataset class to group together all of these samples, before passing all my data into a dataloader to be used during model training. Since the tokens I had generated were simply coded text, I had to generate another version of the tokens that contained masking. To do this, I manually processed each token and randomly assigned it to be masked, random, or remain the same. For my implementation, I followed the procedure described in Jurafsky and Martin, which involved choosing 15% of tokens, and then either applying a MASK token (80% of the time), replacing it with a random token (10% of the time), or leaving it as is (10% of the time). Alongside this, I generated 'attention mask' and 'labels' arrays. The attention mask indicates which tokens are actual input (not padding), and the labels array contains the correct word values only for the masked tokens, replacing all other positions with -100.

For training, I used the Adam optimizer and implemented a learning rate of 0.00005. My dataset was shuffled and presented in batches of 8 sentences. After training for 5 epochs, BERT produced the loss plot shown in figure 1.

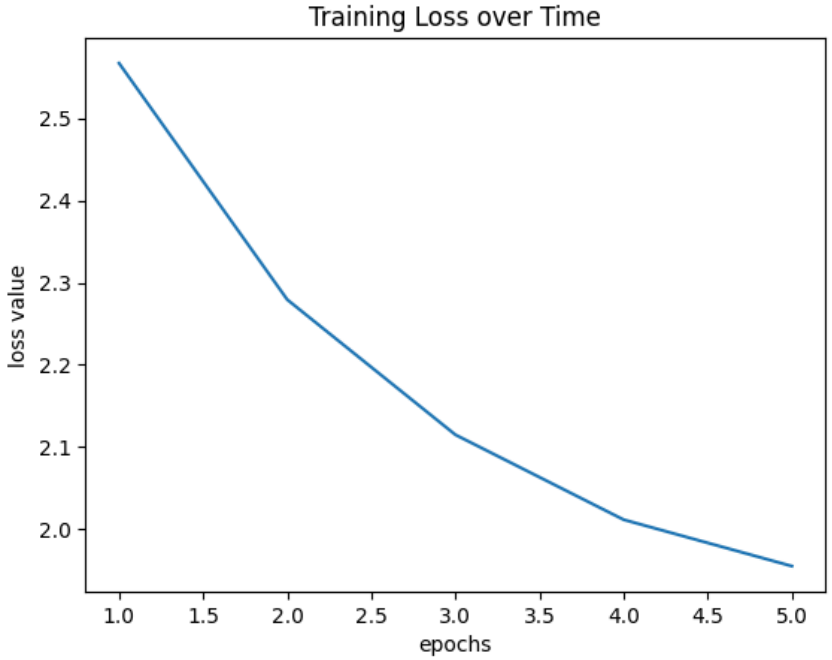

Figure 1. BERT MLM Loss Plot

The power of transformers in NLP tasks has been made clear with the rise of LLMs like BERT who can complete incredibly complex language tasks with ease. Even a task such as sentence completion, which may be difficult for humans, can be completed with high accuracy by a network like BERT. This project allowed me the opportunity to implement such a model for myself, and experiment with it. I found it amazing how easy it was to get a live model running on my own laptop hardware, even while training on a large corpus such as an entire novel. Despite the amount of processing going on, each epoch took less than a minute to run. This project showed me firsthand the power of transformers in difficult NLP tasks, which makes me

want to experiment with them more. If I were to have more time, I would like to have BERT perform additional NLP tasks, such as coming up with word embeddings, or sentiment analysis. I would also like to compare it to decoder models like GPT to see if its bidirectional attention will lead to a significant difference in performance or learning speed.

Sources:

Jurafsky, D., & Martin, J. H. (2025). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition with language models (3rd ed.). Retrieved from https://web.stanford.edu/~jurafsky/slp3/