Transformer Architectures for Speech Classification and Language Modeling

Zhicheng Wang

Abstract

In this report, we implement transformer-based encoder and decoder architectures for two NLP tasks: speech-segment classification and language modeling. Specifically, the encoder is trained to classify speech segments among three American politicians, and the decoder is trained in an autoregressive fashion to predict the next word in a sequence. This document summarizes the model design, training procedures, and empirical results, including attention map analyses and performance metrics.

1 Introduction

In this assignment, we implemented a transformer encoder and decoder from scratch, leveraging these architectures for two different NLP tasks: text classification and language modeling. Specifically, the encoder was trained to predict which American politician (Barack Obama, George W. Bush, or George H.W. Bush) delivered a given speech segment, while the decoder was trained on a language modeling task to predict the next word in a sequence. By building the transformer blocks ourselves, we gained a deeper understanding of how transformers use attention mechanisms to capture linguistic structure in text. This report outlines the model implementations, training processes, results, and analysis of the attention mechanisms used in both parts of the assignment.

2 Part 1: Transformer Encoder with Classifier

2.1 Encoder Implementation

The transformer encoder consists of multiple transformer blocks, each containing a multi-head self-attention layer and a feedforward network. The self-attention layer allows the model to learn contextualized embeddings by attending to various parts of the input sequence, while the feedforward network further refines these embeddings.

To address the lack of inherent sequentiality in transformers, we add positional embeddings to the token embeddings. The encoder processes a sequence of tokenized words through its layers to produce high-dimensional, context-rich embeddings.

2.2 Feedforward Classifier

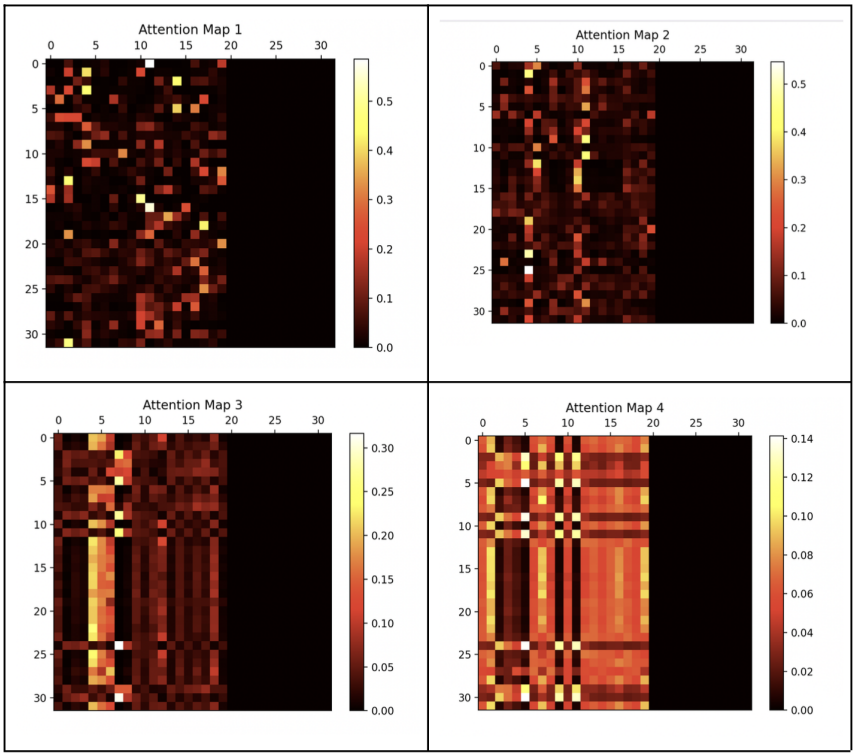
For the classification task, we implement a simple feedforward neural network (one hidden layer with ReLU activation) on top of the encoder. The output layer has three units, corresponding to the three politicians. We use mean-pooling on the encoder's final output embeddings to obtain a single feature vector for each input segment, which is passed to the classifier.

2.3 Joint Training

The encoder and classifier are trained end-to-end. Each speech segment is passed through the encoder, and the mean-pooled encoding is fed into the classifier to produce a label prediction. We use the cross-entropy loss and backpropagate errors through both the encoder and the classifier.

2.4 Attention Map Analysis

During training, attention maps from each layer were visualized to verify the model's attention distribution. Representative examples of these maps include:



- **Attention Map 1 and Map 2:** Show diverse focus patterns, with some tokens attending heavily to specific nearby tokens.

- **Attention Map 3 and Map 4:** Show higher levels of self-attention, focusing more on individual tokens and their immediate neighbors.

These patterns indicate that the encoder learns to contextualize each word based on its surroundings — a capability crucial for distinguishing stylistic differences in speech segments.

2.5 Training and Accuracy Results

The model was trained for 15 epochs, and the classification accuracy was evaluated on a held-out test set. Table 1 summarizes the training loss and test accuracy:

Table 1: Encoder Classifier Training: Loss and Test Accuracy per Epoch

Epoch	Loss	Test Accuracy
1	1.0779	40.53%
2	1.0363	37.73%
3	0.9573	45.07%
4	0.8833	59.07%
5	0.7639	60.93%
6	0.6584	68.00%
7	0.5701	69.87%
8	0.4948	73.73%
9	0.4043	78.13%
10	0.3002	80.40%
11	0.2603	81.20%
12	0.1890	84.67%
13	0.1376	83.60%
14	0.1106	85.87%
15	0.1293	86.53%

By the 15th epoch, the model achieves a final test accuracy of 86.53%, indicating successful learning of stylistic features in the speech segments.

3 Part 2: Transformer Decoder for Language Modeling

3.1 Decoder Implementation

The decoder architecture follows a GPT-like transformer structure, designed for autoregressive language modeling. Each block contains:

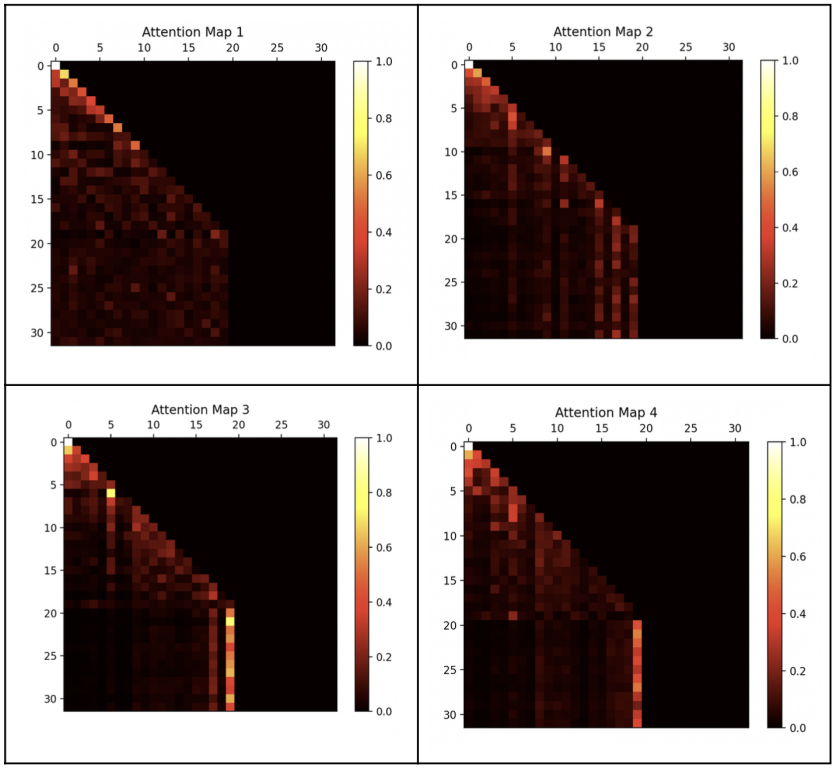
- **Masked Multi-head Self-Attention:** Prevents the model from attending to future tokens.
- **Feedforward Network:** Processes and refines embeddings after attention.

Positional embeddings are also included to represent token order. The masking enforces the autoregressive nature of language modeling, ensuring the model sees only past or current tokens when predicting the next word.

3.2 Pretraining on Language Modeling Task

For language modeling, we train the decoder to predict the next word given the previous tokens. We use a cross-entropy loss and update model parameters via backpropagation. Due to resource constraints, training was limited to 500 iterations.

3.3 Attention Map Analysis



The attention maps in the decoder exhibit a triangular pattern due to masking. In the lower layers, attention is strongly localized to immediate history; in higher layers, the model broadens its attention over a wider context. This approach is consistent with the autoregressive objective, focusing primarily on preceding tokens when generating the next output.

3.4 Perplexity Results

We track perplexity throughout training to measure the model's capability to predict the next token accurately. Table 2 shows the loss and corresponding training perplexity at various points during the 500 iterations:

Table 2: Decoder Training Loss and Perplexity (selected iterations)

Iteration	Loss	Training Perplexity
100	6.3976	589.98
200	6.0102	435.86
300	5.7639	296.77
400	5.4180	212.66
500	4.9491	160.99

We evaluate the final model on separate test sets from each politician's speeches. The final perplexities are listed in Table 3:

Table 3: Test Set Perplexities by Politician

Test Set	Perplexity
speechesdataset/test_LM_obama.txt	353.51
speechesdataset/test_LM_wbush.txt	440.90
speechesdataset/test_LM_hbush.txt	395.93

Higher perplexities in certain sets suggest that the model found some speech styles more challenging to predict, possibly due to unique vocabulary or structure.

4 Conclusion

Through implementing the encoder and decoder portions of a transformer from scratch, we developed deeper insights into how attention-based models capture context in text. The encoder-based classifier accurately distinguished speech segments from three different politicians, while the decoder-based language model achieved moderate perplexity given limited training. Attention map analyses revealed how the model learns to focus on relevant contexts for both classification and generation tasks.