Improving Speech-to-Text Brain-Computer Interface Performance with Neural Decoders and Large Language Models

Stanford CS224N Custom Project

Laywood Fayne
Department of Computer Science
Stanford University
lfayne@stanford.edu

Mohammad Rehan Ghori
SCPD Student - Google
Stanford University
rghori7@gmail.com

Abstract

A speech neuroprosthesis has been developed by Willett et al. [1] to convert neural activity to text and audible speech for those who have lost the ability to speak due to paralysis or other causes. They have created a pipeline consisting of a gated recurrent unit (GRU) and an n-gram language model. Our work aims to improve this pipeline by switching out model architectures and adding conversational context to decrease the word error rate of the speech output.

1 Key Information to include

- Mentor: Chaofei Fan

2 Introduction

Brain-computer interfaces (BCIs) offer tremendous potential for improving quality of life for those with a range of neurological disabilities. They record the brain's electric activity and translate it into concrete forms for applications in various tasks. Inherently, some form of computer processing is necessary in this translation, and specifically, deep learning methods have shown promise. Past research has looked into technology for restoring movement and improving motor capabilities for those with disabilities, and interfacing with virtual devices to interact with the outside world and help in day-to-day tasks [2].

In addition, there have been efforts to produce devices for those that do not have the ability to speak. In this work, natural language processing (NLP) is crucial to producing understandable language output. This project builds off of research by Willett et al. [1], who have developed a pipeline that "records spiking activity from intracortical microelectrode arrays" and translates it into speech. Research in this area is critical to allowing people to regain the ability to communicate effectively and fully enjoy a piece of the human experience that they may have lost.

2.1 Data

Our dataset consists of information from 10,000 spoken sentences from a single participant over the course of 3 weeks. For each sentence, we have a written text transcript, a translation of the transcript into spoken phonemes, or units of sound, and a corresponding time series of processed (digital filtering, thresholding, feature extraction, etc.) neural data.

2.2 Existing Framework

The speech neuroprosthesis produced by Willett et al. [1] is made up of two main components. First, the neural data, which is generally speaking represented as real-valued vectors $v \in \mathbb{R}^{256}$ for 20

ms time bins, is passed into a recurrent neural network (RNN), specifically a gated recurrent unit (GRU) model. For every time series of neural data, this piece outputs another time series of decoded phonemes. The English language contains 40 distinct phonemes. The electrode array used to record brain activity can be placed in an area of the brain that is designed to produce/process spoken sounds, leading to this choice. Attempting to translate neural activity directly to text/speech is likely too difficult and/or invasive.

These phonemes are passed into an n-gram language model, which is trained on the OpenText corpus and a large vocabularly. This model directly outputs decoded sentences, combining the auditory information from the phoneme sequence with the English language context it is trained on.

3 Replacing GRU

We first looked into improving the GRU model within the pipeline. Due to compute constraints, we were unable to feasibly modify the n-gram language model, as it would require training on >70GB of data. Additionally, the existing architecture requires $\approx$ 60GB of main memory for the small version of the decoder (3-gram) and $\approx$ 1TB of main memory for the larger version (5-gram).

Gated recurrent units are a type of RNN similar to Long Short-Term Memory (LSTM), which uses gating signals to help maintain signals over the processing of a time series. It uses 2 gates instead of the 3 used in LSTMs and it has been shown to offer better performance in many cases, being able to retain more long-term information [3].

3.1 Transformer Model

Our first attempt involved a transformer based model that utilized multi-head self attention blocks as introduced in Attention Is All You Need [4]. We hypothesized that the the attention mechanism would be able to effectively learn relationships between neural inputs at different time points and potentially improve phoneme decoding performance.

3.1.1 Model Architecture

Our model takes some elements from the standard multi-head attention framework and modifies portions to be compatible with the existing setup. A diagram of the model can be seen in Figure 1.

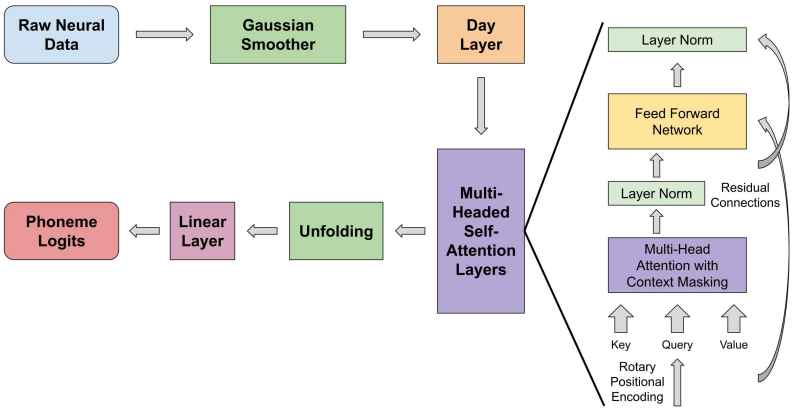

Figure 1: Transformer model architecture

The raw neural data is first smoothed to enhance the signal-to-noise ratio, before being passed into a fully connected layer for each day of speech. These daily layers are required for recalibration due to natural fluctuations in brain activity. The data then moves through several multi-head attention layers as outlined. For the self-attention mechanism, we masked inputs more than $\pm x$ seconds away. Transformers typically require a lot of training data, which we do not have access to, so we mask these inputs to effectively reduce the data size. In addition, there is a certain range (generally around ± 1 seconds) where phonemes are correlated with one another, so there is limited use for using the

complete context available. For example, sounds are generally relevant for the word they are being used for and possibly adjacent words in the sentence. We then pass the data through an unfolding layer and a final fully connected layer to get phoneme logits.

3.1.2 Parameter Finetuning and Training

Because of limited computing resources, we were unable to do a full hyperparameter sweep to optimize our implementation. However, we did experiment with learning rate scheduling, hidden dimension sizing, number of layers, and context masking. We found the best results with a linear learning rate schedule from 10^{-6} to 10^{-3} for the first half of training before remaining constant, hidden dimension of size 256, 10 multi-head attention layers and a context mask of ± 1 second. We trained 40000 batches with a batch size of 16. A single training pass took around 8 hours on an NVIDIA T4 GPU.

3.2 EEGNet

The EEGNet architecture is a computationally efficient model specifically designed to capture the unique characteristics of electroencephalogram (EEG) data, which represents electrical activity in the brain. EEG signals offer a window into neural processes, and the EEGNet architecture effectively extracts spectral, spatial, and temporal features from these signals. These features are key to understanding the dynamic patterns underlying brain function, particularly in the context of speech production.

In our implementation, we adapted the EEGNet model from [5] to better suit the task of phoneme probability prediction. Lawhern et al. use EEGNet to classify various motor movements and validate the model on existing BCI datasets. We modified the model by incorporating a "day layer" and Gaussian smoothing as in our transformer design, taking from the GRU-based approach in [1].

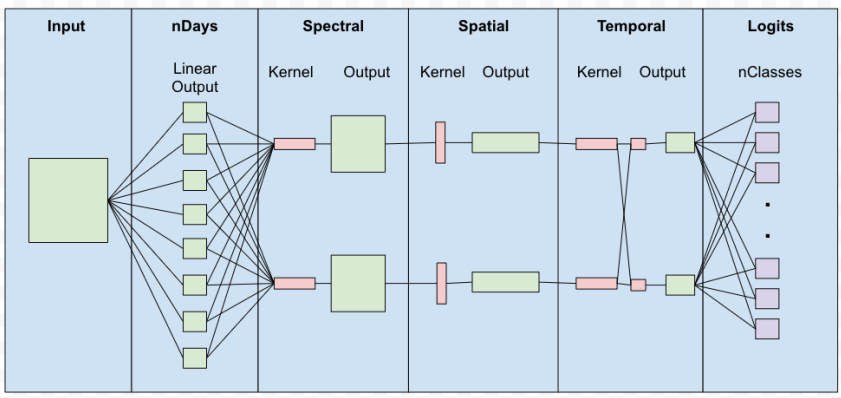

Figure 2: High level EEGNet architecture

3.2.1 Model Architecture

Day Layer: Same as GRU and transformer models.

Spectral Convolution (1D): This layer applies a 1D convolution to the input EEG data along the temporal dimension. It extracts frequency-specific features from each EEG channel, similar to applying a bandpass filter.

Spatial Convolution (Depthwise 1D): This layer applies a depthwise convolution, where each filter operates on only one channel of the EEG data at a time. This allows the model to capture local spatial patterns within each channel independently. Average pooling is used to reduce the dimensionality of the features in both the spatial and temporal dimensions. This helps to downsample the data and make the model more computationally efficient. Dropout is applied after pooling to reduce overfitting by randomly dropping some neurons during training.

Temporal Convolution (1D): This layer applies another 1D convolution, this time without grouping. This allows the model to capture more complex temporal relationships between the features extracted

in the spatial block. The kernel size controls the temporal extent of the patterns captured. Batch normalization and ELU activation are applied for improved training stability and non-linearity.

Final Linear Layer: This layer takes the flattened output of the temporal block and projects it onto the phoneme probability space. It has an output dimension of n_classes + 1 to account for the blank token in CTC decoding.

3.3 Hybrid EEGNet+LSTM: Enhanced Temporal Feature Extraction

This model builds upon the foundational EEGNet architecture by integrating a Long Short-Term Memory (LSTM) layer within the spatial processing block. LSTMs are excellent at modeling sequential data and capturing long-term dependencies. By incorporating an LSTM into the EEGNet, the model can better identify complex temporal patterns across multiple EEG channels that might be missed by standard convolutional layers.

3.3.1 Model Architecture

In our exploration of optimal architectures for EEG-based phoneme prediction, we hypothesized that a hybrid model, combining the strengths of EEGNet and LSTM, would outperform individual architectures.

EEGNet is well-suited to EEG data due to its efficient spatial filtering capabilities, while LSTMs excel at capturing temporal dependencies. We integrated an LSTM layer between the spectral and spatial blocks of the EEGNet to harness the strengths of both.

3.4 Results

The results in Table 1 are on the test data for all of the models we experimented with. For our custom models, the best performer is displayed. We evaluate using the phoneme error rate, which is the edit distance between an output phoneme sequence from the model divided by the length of the true phoneme sequence. The output phoneme sequence is computed by taking the argmax over the phoneme logits, which indicates the most likely phoneme. The sequence is then generated from unique consecutive phonemes. Edit distance is generally used to measure how (dis)similar two strings are. It is calculated as the minimum number of edit operations (insertion, deletion, substitution) to make the two strings equal to one another. Here, phonemes effectively replace characters in the strings.

Model	Final Test Phoneme Error Rate
GRU	20.7%
Transformer	39.3%
EEGNet	41.7%
EEGNet+LSTM	63%

Table 1: Comparison of neural data to phoneme models. Best phoneme error rate over the test dataset.

Unfortunately, we were unable to create a custom model that outperforms the original implementation by Willett et al. [1]. All of our custom models performed significantly worse than the baseline at around double the error rate. For our attempt with the transformer, one possible reason is due to our relatively small training set. Transformers often need to be quite large and deep and trained on a lot of data to be effective. The transformer-based architectures used in the state of the art LLMs are so successful because they use massive models trained on immense corpora in parallel. In addition, research has been done that shows transformers can struggle with time series forecasting [6]. While our problem is not exactly equivalent, it is somewhat similar in taking in sequential time data and producing some output. One can even argue that our problem is more complex, having to predict sequence-to-sequence. Compared to RNN-style models, transformers are often preferred since they are able to capture long-range context and use it to update their predictions effectively. However, in this case, we do not necessarily need a lot of lookback in order to translate brain activity to sounds, as discussed previously for the context masking.

The inferior performance of the EEGNet and EEGNet+LSTM models compared to GRU and Transformer models could be attributed to several factors. EEGNet and EEGNet+LSTM are simpler architectures compared to GRU and Transformer models. Although they are designed to efficiently capture EEG-specific features, they may lack the capacity to model complex temporal dependencies and interactions crucial for accurate phoneme prediction. Also, the hyperparameters (e.g., learning rate, regularization, number of filters) for EEGNet and EEGNet+LSTM are not optimally fine tuned for our task and limited dataset. While EEGNet is designed for EEG signals, it might not be the best architecture for all EEG-related tasks. The specific features that EEGNet is designed to extract might not be the most relevant for phoneme prediction compared to the features learned by the GRU or transformer models. EEGNet is used more for tasks like making a distinction between left or right motor skills or simpler classification.

4 LLM Rescoring

Given that we did not have the resources to modify the n-gram decoder, we next tried to figure out how to improve the final decoded text output using large language models (LLMs) trained on comprehensive text corpora. The existing pipeline is able to output an n-best list of decoded sentences from the n-gram decoder. As a baseline, one can use the top-1 output as the final decoded sentence. Alternatively, one can pass in multiple options to an LLM and use it to rerank these options and select a potentially different top-1 output.

Due to memory limitations, we used OpenAI's GPT-2 XL model [7], the 1.5B parameter version of GPT-2, and Meta AI's Open Pre-trained Transformer (OPT) Language Model [8], which has 2.7B parameters. We used the HuggingFace API to access these models and perform our experiments [9] [10]. Additionally, we were only able to use the smaller version of the n-gram decoder.

Based on the results from attempting to modify the GRU portion of the pipeline in section 3, we used the final phoneme logits from the existing GRU model to pass into the n-gram decoder in order to achieve the best results.

4.1 Simple Rescoring

From the n-gram decoder, we received a list of selected sentence decodings as well as corresponding log probabilities from the language model itself and an "acoustic score" corresponding to the confidence from phoneme output from the first part of the pipeline. We took the selected sentences, tokenized them and passed them into our LLM of choice, which returned a tensor of logits. We then softmaxed these logits to get a probability distribution at every token position. We summed the log of these probabilities across the tokens for a given sentence to get an overall log probability for the sentence as a whole. I.e. at the position of token 1, we looked at how likely token 2 in the sentence is according to the LLM logits.

After the above process, there are 3 log probabilities for each sentence, one corresponding to the n-gram language model, one corresponding to the LLM, and one corresponding to the initial phoneme decoded logits. From here, we used different linear combinations of these values to get a final probability score for each sentence. The resulting decoded sentence choice was simply the option with the highest score.

4.2 Rescoring with Context

The approach in the previous section allows us to harness the more complex language understanding present in LLMs as compared to a more simple n-gram decoder. However, it does not utilize the LLMs capabilities to process long form context and use this context to better inform its interpretation of the following inputs/outputs.

Our test dataset is taken from SWITCHBOARD, which is "a large multispeaker corpus of conversational speech and text" [11] (the participant was asked to repeat sentences from this dataset). The test examples were selected randomly from the recorded telephone conversations. In order to feed relevant context to the model, we used the true labels to find the conversation from which the example originates. We then took the preceding k pieces of the dialogue. This text was prepended

to the selected outputs from the n-gram decoder before being given to the LLM for rescoring as demonstrated above.

4.3 Results

The results below show the character error rate (CER) and word error rate (WER) of the decoded sentence outputs for different LLMs and parameter values. The final probability scores were calculated as:

$$w_{\text{LLM}} \cdot \text{LLM score} + (1 - w_{\text{LLM}}) \cdot \text{n-gram score} + w_a \cdot \text{acoustic score},$$

where w_{LLM} represents a changeable language model weighting and w_a represents a changeable acoustic score weighting. The best parameter values across a sweep were selected.

LLM	w_{LLM}	w_a	**CER (95% CI)**	**WER (95% CI)**
None	N/A	N/A	21.7% (20.5%, 23.0%)	32.0% (30.3%, 33.9%)
GPT-2 XL	1	1.2	21.5% (20.4%, 22.6%)	33.6% (32.0%, 35.3%)
OPT 2.7B	1	1.2	21.3% (20.2%, 22.3%)	33.2% (31.5%, 34.9%)
GPT-2 XL (w/ Context)	1	0.6	19.8% (18.7%, 20.9%)	29.8% (28.1%, 31.4%)
OPT 2.7B (w/ Context)	1	0.6	19.6% (18.6%, 20.6%)	29.5% (27.9%, 31.1%)

Table 2: Comparison of LLM rescoring methods

We can see in Table 2 that simply using the LLM to rescore did not noticeably affect the CER and even makes the WER worse. However, adding conversational context reduced both metrics by around 2%. For the data above, we used 3 prior pieces of dialogue as context. Interestingly, the best decoding performance across the board was for weights that entirely focused on the LLM scoring as opposed to the n-gram model's ranking. This might simply be because the LLMs are trained on larger and more robust copora of text, meaning they have a better understanding of the English language.

5 Conclusions and Future Work

Our attempts to modify the first part of the pipeline and replace the existing GRU model were largely unsuccessful. None of the models we tried were able to come close to matching the phoneme decoding performance. For the transformer model, there are several fundamental design features that may have caused this as discussed in subsection 3.4. We were also limited in computation resources and were not able to effectively finetune the hyperparameters of the models. A simple modification the hyperparameters of the EEGNet model, we were able to increase the phoneme error rate from 73% to 41.7%. Similarly running the hybrid EEGNet model turned out to be lot more challenging due to increased memory requirements which prevented us from fine tuning the hyperparameters effectively.

Despite this lack of success, some improvements could possibly be made upon the GRU model itself. For example, there are different kinds of architectures that use various kinds of gating mechanisms [3].

We did find some success, however, with applying LLM rescoring given conversational context. We were able to improve the final sentence decoding accuracy, which, in the end, is the main metric of speech quality. We still fall behind the 23.8% word error rate that Willett et al. [1] achieved using the 5-gram decoder model. This suggests that the approach we use is limited by the selected outputs from the 3-gram decoder model. However, the success we have indicates that LLMs might be able to improve outputs from the 5-gram decoder as well.

Further work could be done to experiment with different context lengths and the use of larger models. In addition, one could consider feeding the n-gram outputs into a state of the art model, such as GPT-4, and having it rescore and potentially modify the outputs if none of them seem to make much sense. However, this implementation would likely be impractical in a real world setting for real time speech decoding since these kinds of models are so large.

6 Ethics Statement

Our project involves the use of invasive brain-computer interfaces, which raises several ethical concerns and potential societal risks. These include:

- **Privacy and security of neural data:** Neural data is highly sensitive and personal. There is a risk of unauthorized access, misuse, or discrimination based on this data.
- **Informed consent and autonomy:** Participants in BCI research must be fully informed about the risks and benefits of the technology and must provide voluntary and informed consent. There is a risk of coercion or undue influence, especially for individuals with limited communication abilities.
- **Equity and access:** BCIs are expensive and may not be accessible to everyone. This could exacerbate existing health disparities and create new forms of inequality.
- **Unintended consequences and misuse:** BCIs could be used for purposes other than restoring communication, such as surveillance or manipulation. There is also a risk of unintended consequences, such as changes in personality or cognition.

Mitigation Strategies:

To mitigate these risks, we propose the following strategies:

- **Robust data security and privacy protocols:** Implement strict data encryption, access controls, and anonymization techniques to protect neural data.
- **Thorough informed consent process:** Ensure that participants fully understand the risks and benefits of BCI technology and provide voluntary and informed consent.
- **Equitable access and distribution:** Advocate for policies and funding mechanisms that promote equitable access to BCI technology.
- **Responsible research and development:** Engage in ongoing ethical discussions and assessments of BCI technology, and develop guidelines for responsible use and development.

References

[1] Chaofei Fan Donald T. Avansino Guy H. Wilson Eun Young Choi Foram Kamdar Matthew F. Glasser Leigh R. Hochberg Shaul Druckmann Krishna V. Shenoy Francis R. Willett, Erin M. Kunz and Jaimie M. Henderson. A high-performance speech neuroprosthesis. In *Nature 620, 1031–1036*, 2023.

[2] Homayoon Zarshenas Mahdi Bamdad and Mohammad A. Auais. Application of bci systems in neurorehabilitation: a scoping review. *Disability and Rehabilitation: Assistive Technology*, 10(5):355–364, 2015. PMID: 25560222.

[3] Rahul Dey and Fathi M. Salem. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1597–1600, 2017.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[5] Vernon J Lawhern, Amelia J Solon, Nicholas R Waytowich, Stephen M Gordon, Chou P Hung, and Brent J Lance. Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces. *Journal of Neural Engineering*, 15(5):056013, July 2018.

[6] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting?, 2022.

[7] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[8] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

[9] openai-community/gpt2-xl. https://huggingface.co/openai-community/gpt2-xl. Accessed: 2024-06-05.

[10] facebook/opt-2.7b. https://huggingface.co/facebook/opt-2.7b. Accessed: 2024-06-05.

[11] J.J. Godfrey, E.C. Holliman, and J. McDaniel. Switchboard: telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1, 1992.

A Appendix

The code used in this report is contained in a private repository: https://github.com/lfayne/cs224n_fp. Access can be granted upon request.

We would like to thank Chaofei Fan as our mentor for this project. The code base we built our work of of is located in the following repositories: https://github.com/fwillett/speechBCI/tree/main, https://github.com/cffan/neural_seq_decoder/tree/master.