Reading Between the Minds: Context-Aware Brain-to-Text Decoding

Stanford CS224N Custom Project
TA mentor: Chaofei Fan

Ellie Tanimura
Department of Computer Science
etanim@stanford.edu

Sarosh Khan
Department of Electrical Engineering
skhan44@stanford.edu

Abstract

Speech brain–computer interfaces (BCIs) hold promise for restoring communication in individuals who are unable to speak intelligibly by decoding neural activity into text. However, current state-of-the-art brain-to-text systems produce word error rates (WER) that are too high for practical use. This study mitigates common sources of error by integrating an additional rescoring step in the pipeline that considers the conversational context of the decoded transcriptions. From there, we explore three lightweight, context-inclusive interventions, including a self-improvement approach to prompt engineering inspired by iterative prompt tuning. Using the model ensemble proposed by Willett et al. (2023) as our baseline, our low-resource interventions achieve a 5% reduction in WER, highlighting the potential of context-aware decoding to enhance the accuracy of speech BCIs and improve their usability in real-world applications.

1 Introduction

Speech BCIs have the potential to restore rapid communication to individuals who cannot physically speak intelligibly by decoding neural activity into text Willett et al. (2023). These systems are typically activated when BCI users think about or attempt to vocalize the words they wish to communicate. This mental activity generates neural spikes in target regions of the brain, which the system decodes into corresponding sentences or phrases.

Willet et al. recently demonstrated a groundbreaking speech-to-text inference pipeline that decodes spiking activity into sentences with a 23.8% word error rate (WER) on a 125,000-word vocabulary, which is not yet sufficiently low for everyday use Xiong et al. (2018)He et al. (2019).

In their analysis, they noted that phonemes articulated similarly were more likely to be confused by the RNN decoder. Additionally, after examining the outputs from their test set, we observed that many of the errors were homonymous in nature—for example, if the ground truth value is 'know,' their pipeline outputs 'no.' Given these observations, our project aims to disambiguate similar-sounding outputs by incorporating conversational context into the decoding process with an additional rescoring step. Borrowing from existing large language models (OPT-350M, GPT-3.5), we study the effects of three context-inclusive approaches to rescoring—using a general-purpose transformer model, exploring a parameter efficient fine-tuning (PEFT) method to improve the transformer's accuracy, and developing a self-tuning prompt engineering technique.

Quantitative and qualitative analysis of our interventions suggests that incorporating context-awareness into the brain-to-text pipeline is a highly-effective intervention for improving the quality and accuracy of the decoded transcriptions. While there is still much work to be done to ensure the usability of speech BCIs, we hope our findings are a positive step towards their practical deployment.

2 Related Work

2.1 Recent Advancements in Brain-to-Text Systems:

Most of the work on speech BCIs explores one of two areas of improvement: disambiguation of similar-sounding phonemes, or latency reduction. This project concentrates on the former. In particular, it builds upon the recent work by Willett et al. Their system employs intracortical electrode arrays to capture neural activity, which is processed through a five-layer gated recurrent unit architecture. This architecture decodes a time series of neural spikes into a sequence of phonemes by selecting the highest probability phoneme at each index. The resulting phoneme sequences are then inputted into a Weighted Finite-State Transducer, which maps the phonemes to likely words using weights generated by a trigram language model (LM). The output from the trigram LM is a list of decoded hypotheses, ranked according to the probability that each one accurately represents what was actually thought or voiced by the user. In their supplementary work, Willett et al. found that applying a transformer LM to rescore these outputs significantly reduced the word error rate from 23.8% to 17.4%. These findings were corroborated by in a subsequent study.

Additionally, tangential work by Benster et al. (2024) on noninvasive silent-speech interfaces introduced a technique called LiSA. This technique employs few-shot learning on transformer language models to select the most accurate output from the decoding step, prioritizing contextual and grammatical correctness while avoiding repetitive or nonsensical phrases. Although their work focuses on noninvasive BCIs, they evaluated their method with the same vocabulary used by Willett et al. and achieved a 8.9% error rate. This marked the first instance where noninvasive silent speech recognition on an open vocabulary achieved a WER of less than 15%.

2.2 Low-resource computing:

Given the size of our dataset and the limited computational resources at our disposal, we also explored several approaches to low-resource, low-data computing.

First, prompt engineering is the process of optimizing the task descriptions passed through to large language models for better performance on the prompted task. Compared to fine-tuning, it has a lower computational cost. Shin et al. (2020) introduced a prompt engineering technique called Autoprompt, which employs search algorithms to select prompts that elicit knowledge from pre-trained language models. Related work by Li and Liang (2021) added continuous prompts to the input text and used gradient descent to adjust these prefixes.

3 Approach

With the model architecture proposed by Willet et al., our translation pipeline initially passes RNN-generated phoneme probabilities to a trigram model created with Kaldi, which is then used to score all of the possible sentences. WeNet is employed to simplify the implementation of this real-time LM decoder Willett (2022).

Unlike Willett et al., instead of outputting the highest probability sentence after the trigram-weighted beam search step, we extracted the top n possible outputs for each sequence, as well as its ground truth sentence. To each of the outputs, we prepended additional conversational context, which we defined as the m utterances that precedeed the ground truth sentence in its corresponding transcript from the Switchboard corpus.

Then, we rescored *n* best context-prepended hypotheses (fewer in cases where the RNN provided less than n possible hypotheses). We ran three experiments to determine the best approach to rescoring.

3.1 OPT-350M

First, based off work by Card et al. (2024), we experimented with rescoring our hypotheses using a general purpose `facebook/opt-350m` model and a byte-level Byte-Pair Encoding tokenizer. This model contains 350 million parameters, and belongs to Facebook's Open Pre-trained Transformer (OPT) family of models, which are trained on large-scale datasets and are optimized for both efficiency and accessibility.

We selected this model for several reasons. Firstly, its open-source nature and lightweight architecture make it ideal for environments with limited computational resources. Secondly, we conducted a hyperparameter sweep for both `facebook/opt-350m` and `facebook/opt-6.7b` and found that the larger model's marginal increase in performance (a 0.1% decrease in accuracy) did not justify the substantial increase in computational and memory demands. This insight underscored the efficiency of `facebook/opt-350m`, allowing us to achieve nearly equivalent accuracy while significantly reducing resource usage.

Each block in `facebook/opt-350m` is composed of a multi-head attention mechanism and a feed-forward layer. Multi-head attention can be represented as a linear transformation of concatenated individual heads, where a single head h_i is equal to

$$h_i = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})V_i.$$

For each tokenized input, the model stacks several of these blocks to output a matrix of logits L_i, where each entry L_{ijk} represents the logit for the k-th possible token at the j-th position in the i-th hypothesis Zhang et al. (2022) Vaswani et al. (2017).

We use this series of logits to compute the conditional log probabilities for each token. To obtain the rescores for each hypothesis, we applied a length penalty to these probabilities.

$$S_i = \sum_{j=\text{context\_length}}^{n_{\text{tokens}}} P_{ij}(\text{hypothesis} \mid \text{context}) - n_{i,\text{real\_tokens}} \times \text{lengthPenalty}$$

To calculate a total score for each hypothesis, we linearly combined S_i with the acoustic scores from the RNN and the LM scores from the trigram model, using weighting hyperparameters α and a:

$$S_{total} = \alpha * S_i + (1 - \alpha) * \text{oldLMScores} + a * \text{acousticScores}.$$

Finally, we reranked n best hypotheses using S_{total} , and reevaluated the WER with this new ranking.

3.2 OPT-350M-LoRA

In this study, we utilized Low-Rank Adaptation (LoRA) Hu et al. (2021) with the OPT-350M model. LoRA is a parameter-efficient fine-tuning technique (PEFT) designed for fine-tuning pre-trained language models with limited computational resources. LoRA operates by leaving the original weight matrix W_0 of the model as is during fine-tuning instead learning two low-rank matrices A and B such that $W \approx W_0 + BA$.

Given an original weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA decomposes the fine-tuning process into learning $A \in \mathbb{R}^{r \times k}$ and $B \in \mathbb{R}^{d \times r}$, where r is the rank of the adaptation. The final weight matrix used during inference is $W = W_0 + BA$. An illustration of LoRA is available in Appendix A Figure 2

3.3 GPT-3.5

We used GPT-3.5-turbo-0125, a GPT-3 variant designed to provide faster responses and more efficient processing compared to the standard GPT-3.5 models. Rather than asking GPT-3.5 to rescore the n-best hypotheses, we asked it to choose and return the best option.

3.3.1. Prompt Engineering: We employed prompt engineering and few-shot learning to direct GPT-3.5 on this task. Distinct from the previous methods we examined, we used iterative self-feedback loops to tune our prompts in real-time. Specifically, for a small part of our dataset, we prompted GPT-3.5 to compare its response to the ground truth transcription for that set of hypotheses, and apply its insights to rewrite the task description, which we then fed back into it.

This approach enables the model to identify and guard against common decoding errors specific to the dataset it evaluates, effectively tuning its task to the nuances of this particular dataset. This is especially important for this task as the candidate transcriptions are notably noisy, often failing to closely match the ground truth. The model adapted to this challenge, learning to recognize the phonetic and stylistic similarities even among inaccurate candidates. It even began to autonomously generate its own hypotheses, mirroring the style of the candidate descriptions but aligning more closely with the actual context.

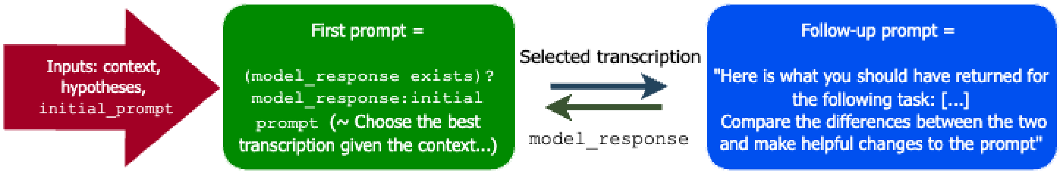
Inputs: context, hypotheses, initial_prompt
First prompt =
(model_response exists)? model_response:initial prompt (~ Choose the best transcription given the context...)
Selected transcription
model_response
Follow-up prompt =
"Here is what you should have returned for the following task: [...]
Compare the differences between the two and make helpful changes to the prompt"

Figure 1: Iterative prompt tuning for a single set of hypotheses.

3.4 Baselines

Prior to applying LLM rescoring, we recalculated the WER for our dataset and established a preliminary score of approximately 16.2%. After rescoring with the LLM on hypotheses without context, we observed a WER of 14.0%.

4 Experiments

4.1 Data

To evaluate the effects of our intervention, we started with Willett et al.'s dataset. To generate this dataset, a speech-impaired research participant with looked at the sentences on a screen and thought about speaking a total of 12,100 sentences sampled from the Switchboard Dialog Act CorpusJurafsky et al. (1997)Shriberg et al. (1998)Stolcke et al. (2000) and OpenWebText2Gao et al. (2020) datasets. The entire dataset is publicly available on Willett et al.'s GitHub.

As mentioned, Willett et al. prompted their research participant to speak sentences from the following corpora:

- **Switchboard Dialog Act Corpus**: Comprises over 1,000 short telephone conversations in English between two individuals. The 440 participants were provided with specific prompts to guide their discussions. The conversations are transcribed into text, with disfluencies and other irregularities removed through post-processing methods.
- **OpenWebText2**: Consists of web pages scraped from Reddit URLs with 40GB across more than 8 million pages.

The Switchboard Corpus comes with several utility functions for the retreival of conversational context associated with each of the ground truth sentences. Since OpenWebText2 consists of individual web pages scraped from various sources without any inherent conversational structure or continuity, the sentences from OpenWebText2 were omitted from our dataset. After this omission, we ended up with 284 sentences with conversational context.

4.2 Evaluation method

We evaluated the performance of our model using word error rate and character error rate, measuring the Levenshtein distance between the ground truth transcription and the post-rescoring highest-ranked hypothesis. WER provides a straightforward measure of the overall accuracy of the predicted transcriptions at the word level. CER, on the other hand, evaluates the performance by comparing individual characters. They are computed as follows:

$$\text{WER, CER} = \frac{\text{Substitutions} + \text{Deletions} + \text{Insertions}}{\text{Total words, characters in ground truth}}.$$

4.3 Experimental details

4.3.1. Context Types: We explored several methods for incorporating context, using information from the Switchboard Corpus. For our first approach, we prepended previous utterances from the conversation onto the decoded phrase. After rescoring with `facebook/opt-350m`, this intervention resulted in a 12.8% Word Error Rate (WER). Next, we added punctuation back into the context before prepending it. This adjustment decreased the WER by 0.6%, bringing it to 12.2%. Finally, we included speaker turns by listing the current speaker throughout the conversation. This further reduced the WER to 11.8%.

An example of this processing on a single contextual passage can be found in Appendix A, Table 5

4.3.2. OPT-350M Configuration and Hyperparameters: In developing our general-purpose OPT model for rescoring, we enabled 8-bit loading to reduce memory usage. Our preliminary experiments indicated that 8-bit precision did not significantly impact performance, while substantially decreasing compute time and memory requirements. We initialized the tokenizer with the standard end-of-sequence (`eos_token`) token, ensuring it was appended to the end of each sequence.

We performed a comprehensive hyperparameter sweep to determine the optimal values for our rescoring calculations. Specifically, we swept the following hyperparameters over the corresponding ranges: α and a: [0, 1]; lengthPenalty: [-10, 10]; n (number of best hypotheses to include in rescoring): [10, 100]; m (context length for each hypothesis): [10, 25]. The context length (m) sweep was capped at 25 utterances, as larger contexts necessitated memory capacities exceeding those available. We determined that the optimal hyperparameters were $\alpha = 0.8$, $a = 0.5$, lengthPenalty = -0.5, $n = 30$, and $m = 25$, yielding a WER of 11.5%.

4.3.3. LoRA Hyperparameters and Performance We chose r to be 16 to encourage expressiveness and set the LoRA α parameter to 32 to enhance adaptability without causing high variance. The alpha parameter is a scaling factor used to adjust the learning rate of the low-rank matrices A and B. Specifically, the adapted weight matrix during fine-tuning is calculated as $W = W_0 + \frac{1}{\alpha}BA$. A higher α scales down the updates, helping to prevent overfitting and maintain stability, while a lower α increases the risk of overfitting. A dropout rate of 0.3 was used to prevent overfitting and the learning rate was set to 3e-5. We trained conducted over 6 epochs. Given the size of our dataset, we restricted the size of the training set (only 100) for fine-tuning.

4.3.4. GPT-3.5 Hyperparameters: Before prompt engineering, we determined the optimal number of examples for few-shot learning by conducting a sweep from one to five examples. This range was chosen based on the constraints of our small dataset and the findings from previous studies on few-shot learning. Our experiments determined that two examples yielded the most accurate predictions from GPT-3.5.

4.3.5. Prompt Engineering: Using few-shot learning, we asked GPT-3.5 to choose and return the best option from a list of hypotheses given the context for that set of hypotheses. The specific prompt we sent to ChatGPT, modified from Benster et al, can be found in Appendix A Section 8.4.1.

We also developed a self-feedback approach to prompt engineering. Specifically, for a small subset of our dataset—60 examples—we executed a second task that prompted GPT-3.5 to compare its output against the ground truth transcription, and rewrite the first task description to provide better instructions. Each iteration aimed to address the general shortcomings identified in the previous output, refining the prompts to enhance performance. The specific script we used to prompt GPT-3.5 to rewrite the task description can be found in Appendix A Section 8.4.2. Below are several modifications that GPT-3.5 made to the first task description. The table also includes the ground truth and output transcriptions that prompted each change with bolded differences:

Table 1: Dynamic Prompt Engineering

GPT-3.5 Output	Ground Truth	Additions to the Task Description
`they didn't hook up the wiring wrong`	they didn't **fit the car**	“...Always choose the best candidate transcription, even if none seem to perfectly fit the context..."
`anything like that we provided`	anything like that we **participated**	“... If needed, you can change a word in the candidate sentence to make it fit better, but never add or delete words..."
`call and find out`	“**i** call and find out"	“... Some of the correct answers may be grammatically incorrect..."

Each time GPT-3.5 returned a modified prompt, we used it as the task description for the subsequent query into task 1.

After tuning the first prompt on the small subset of datapoints, we queried GPT-3.5 with the tuned prompt on 150 more datapoints. This number was chosen with consideration for our credit limit and the runtime for a dataset of this size—approximately an hour with a 25-utterance context. We used the same dataset to determine two baselines. First, we queried GPT-3.5 with our original prompt, omitting context from each set of hypotheses. Then, we reincorporated context and determined our second baseline. Compared to these methods, our dynamic prompt engineering approach significantly reduced the WER.

4.4 Results

The best performance we achieved with each of the following methods is listed below:

Table 2: Comparison of Models Based on WER and CER

Method	**WER**	**CER**
Trigram *(Baseline)*	16.2%	11.8%
Trigram + OPT-350M *(Baseline with OPT)*	14.0%	10.0%
Trigram + OPT-350 + context	**11.5%**	**8.4%**
Trigram + OPT-350M-LoRA	18.35%	15.7%
GPT-3.5 + few-shot learning - context *(Baseline)*	21.9%	14.6%
GPT-3.5 + few-shot learning + context *(Baseline with context)*	16.9%	11.9%
GPT-3.5 dynamic prompt engineering	**12.3%**	**9.5%**
Trigram + OPT-6.7B + context	11.1%	8.3%
GPT-4 + few-shot learning + context	11.3%	8.4%

The marked reductions in the WER and CER exceed initial expectations. However, the analysis also highlighted some challenges to tackle with further research.

Notably, the performance of OPT-350M-LoRA, which recorded a WER of 18.35%, did not meet our expectations, as it performed worse than both of its baseline measures. This was unexpected since LoRA is designed to mitigate overfitting. A potential explanation for this apparent overfitting might involve poor tuning of relevant hyperparameters, or a small-sample dataset that skewed the learning process. Delving deeper into the proper training method for LoRA could uncover more about why the model's expected generalization didn't occur as anticipated.

Moreover, the minimal performance gap between the OPT-6.7B and OPT-350M models raises questions about the diminishing returns of scaling up model sizes within specific transcription tasks. That being said, we are curious to see how a more recent, much larger model like LLaMA would perform.

That being said, we are very encouraged by the 5% reduction in the WER following OPT-350M rescoring. A visual review of the outputs from this step revealed that a significant number of errors were due to the absence of accurate candidates in the initial lists. This suggests that the accuracy of the system is somewhat bottlenecked by the performance of the RNN and trigram LM. To address this, future work should apply adversarial training on input phoneme sequences to increase the robustness of the RNN against errors in phoneme capture.

Crucially, the dynamic prompt engineering approach with GPT-3.5 demonstrated significant potential in circumventing this bottleneck, as it was able to generate accurate hypotheses that were not included in the initial candidate lists. In Table 3, we illustrate several examples where GPT-3.5 generated responses that matched the ground truth transcriptions much better than any of the provided candidate transcriptions could.

Table 3: Dynamic Prompt Engineering Mitigates RNN Decoding Errors

Best Choice from Static Prompting or LLM Rescoring	Dynamic Prompt Engineering	Ground Truth
`they put him in the nursery office`	`they put him in the nurse's office`	they put him in the nurse's office
`i just feel valuable in time that every year`	`i just file federal and joint income tax every year`	i just file federal income tax every year
`that's not that you like your settings also`	`that's neat that you like cross-stitching also`	that's neat that you like cross-stitching also
`i do personally find it on`	`i do personally find it annoying`	i do particularly find it annoying
`it has so been nice here`	`it has mostly been nice here`	i do particularly find it annoying
`i take it she was an icon`	`i take it she was an accountant`	i take it she was an accountant
`i has so been nice here`	`it has mostly been nice here`	it has really been nice here
`i know the us`	`i know there is`	i know there is

Although the last output from dynamic prompt engineering shares the same WER as the best option from the candidate transcriptions, it still has a more gramatically correct structure.

That being said, there remains a need to refine the extent to which GPT-3.5 can alter its outputs. Occasionally, if the model decided that none of the provided hypotheses were sufficiently accurate, it generated a substantially different and inaccurate response that closely matches the context. To address this, tuning the second prompt is essential. We present an example of this below.

Table 4: Comparison of Prompt Engineering, Dynamic Prompt Engineering, and Ground Truth

Context	Candidate Hypotheses	Dynamic Prompt Engineering	Ground Truth
`speaker 1370: do you want to hear about my other animals i've had? i've had a skunk--speaker 1362:, do the skunk, was it kind of like a cat to have around the house?`	`1. this to say it's either all through the house; 2. here to see each other all through the house; 3. these two see each other all through the house; ...`	`did the skunk act like a cat to have around the house`	used to chase each other all through the house

While we didn't have the credits to run all of our GPT tests with it, our initial experiments with GPT-4 using static, context-inclusive few-shot learning yielded an 11.3% WER. The significantly improved results with dynamic prompt engineering on GPT-3.5 lead us to be optimistic about the potential performance enhancements that could be achieved with GPT-4.

5 Analysis

5.1 OPT-350M Rescoring

A review of outputs following the OPT-LLM rescoring step revealed that a significant number of errors were due to either the absence of the correct transcription in the set of decoded hypotheses, the lack of appropriate context, or ambiguous hypotheses. We expand on this now.

Due to the limited quality of the RNN and trigram-decoded outputs, the set of n best candidate hypotheses doesn't always contain the ground truth transcription. In these cases, the hypotheses will phonetically resemble the ground truth transcription, but often contain none of the correct content or words. See Table 5 for an example of such a set.

These errors in RNN decoding are often compounded by poor contexts. Contextual passages are often quite noisy, and contain people stuttering or abruptly moving on to a different topic. For example, consider the following subset of transcriptions: `[one in college away, run in college away, runs in college away, done in college away, and in college way...]`. The context corresponding to this set of transcriptions covers a wide range of topics, including the government, taxes, and "exotic weapon systems." The utterance right before the decoded transcription—which is supposed to say, "one's in college already"—cuts off, so the local context isn't adequately captured. "One's in college already" is not in the list of decoded hypotheses, but even GPT-3.5 directed with dynamic prompting was unable to generate the correct hypothesis because of this poor context.

The last class of errors that the LLM we encountered is the presence of ambiguous hypotheses. For example, one set of candidate transcriptions contains both `‘‘i can relate to that"` and `‘‘i can't relate to that"`, in response to the other speaker's specific circumstances. In this case, without specific knowledge about the person, it is difficult to choose the correct transcription. This is the smallest class of errors, appearing only a handful of times throughout the dataset.

We believe that the main bottleneck is still primarily the accuracy of the RNN and trigram transcriptions, as it worsens everything else.

5.2 Prompt Engineering

We identified several limitations in our dynamic prompting approach. Firstly, GPT-3.5 consistently failed to remove elements from previous task descriptions, opting instead to continue to append additional stipulations. This accumulation led to overly complex prompts, which we observed to be less effective because they increased the model's cognitive load, leading it to be less focused.

Secondly, our current strategy does not provide a particularly effective method for GPT-3.5 to identify and integrate corrective feedback directly into the task descriptions. Many times, GPT-3.5 returned the unedited task description, even if there were remarkable errors between its output and the ground truth. To address this, we propose splitting the second prompt into two distinct parts. The first part would identify discrepancies between the model's outputs and the ground truth, and the second would rewrite the task description based on these discrepancies. We tested this method using the ChatGPT UI, where it demonstrated a significant improvement in recognizing and integrating corrective feedback.

6 Conclusion

In this project, we investigated an innovative approach to brain-to-text decoding by integrating context awareness into the decoding pipeline. Our initial findings indicated that context awareness significantly reduced the word error rate. To capitalize on this improvement, we explored several context-based interventions, including a lightweight, dynamic prompt engineering scheme that mitigates decoding errors from the RNN and trigram LM.

We hope that our project will contribute to the usability of speech BCIs. In practical applications, conversational context can be recorded in real-time with an ensemble of automatic speech-to-text models and our brain-to-text pipeline. Additionally, with access to more computational power and memory, we expect even more significant improvements in transcription accuracy.

7 Ethics Statement

Given that brain-computer interfaces possess intimate access to a user's neurodata, these devices have the potential to be extremely dangerous. If, for example, outside parties obtained data from the neuroprosthesis, such as individualized model weights, they would be able to obtain deep insights into users' cognitive processes and emotional states, and perhaps capitalize on them. Additionally, there is the risk that users will lose autonomy if people find ways to control them through the devices. To address these risks, these systems should incorporate robust security measures, such as encryption to safeguard data during transmission, and anti-virus and anti-malware technology to defend against cyberattacks that could target or hijack the devices. Additionally, there must be strict access control protocols—rigorous authentication processes, for example—in place to ensure that only authorized personnel have access to sensitive data. To ensure that BCI companies are adhering to these strict privacy standards and regulations, there should be regular audits of these companies conducted by independent bodies.

Additionally, obtaining informed consent is all the more crucial, both because BCIs interact with a person's neurological processes and because they are a new, relatively unknown, and usually invasive device whose long term impacts are unknown. That being said, it is very difficult to explain enough about how they work to obtain consent, since understanding the technology requires a high level of technical knowledge and the brain is very opaque.

Another ethical concern is the potential for BCIs to exacerbate social inequalities. If these technologies are accessible only to a select few, it could deepen the divide between socioeconomic groups, and worsen the stigmatization of disability. As promising as this burgeoning technology is, it is crucial to develop policies in tandem with the development of these devices to promote both safe and equitable access to BCIs. Policymakers could subsidize the cost of these technologies, or align innovation incentives with with public health equity objectives to encourage companies to cater to a larger consumer base.

If only a select few individuals can afford and access these emerging technologies, BCIs may exacerbate existing social inequalities. In order to ensure this does not become a reality, policies should subsidize access to BCIs, especially for people with disabilities who stand to benefit the most from such technology. Through government subsidies, companies might also be encouraged to foster innovation in areas that meet broad societal needs.

8 Appendix A:

8.1 LoRA Illustration

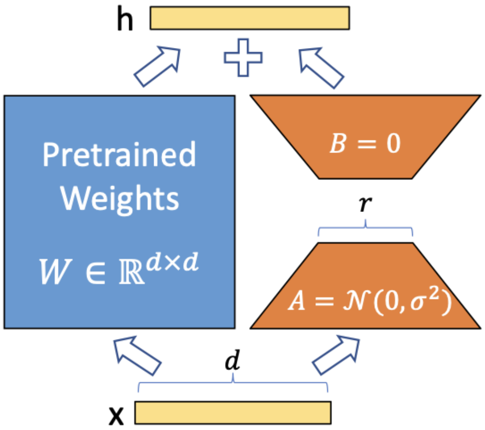
h
Pretrained Weights
$W \in \mathbb{R}^{d\times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 2: Visualization of Low-Rank Adaptation (LoRA). The original weight matrix $W \in \mathbb{R}^{d\times d}$ is frozen, and two low-rank matrices A and B are learned.

8.2 Sentence Preprocessing

To ensure consistent formatting, we preprocess the corpus similarly to Willett et al., but with additional cleanup steps. Specifically, we remove filler words (e.g., "uh", "huh", "um"), unnecessary characters such as parentheses, hashtags, and asterisks, normalize spaces, handle contractions, correct punctuation issues (e.g., double punctuation, misplaced commas), capitalize the first letter of each sentence, split utterances into sentences based on punctuation, and add speaker information to each utterance to maintain conversational context.

In the Switchboard corpus, an utterance is equivalent to a persons turn in a conversation. When one of the conversation participants begins to say something, a new utterance is added to the conversation. These utterances can be multiple sentences long, or just filler words such as "uhm", or "hmm".

The preprocessing function is designed to clean and normalize the text to ensure consistency across all sentences, integrating speaker information for better contextual understanding.

8.3 Different Context Interventions

Here is an example of this processing on a single contextual passage:

Table 5: Comparison of Different Transcription Formats

Stripped	**Punctuation**	**Punctuation and Speakers**
well got any problems on mockingbird with crime or is that a crime free zone there no i dont think there is any such thing as a crime free zone any longer im afraid youre right one evening i decided to retire early and heard sirens and noises and thought oh well somethings happened on mockingbird and then heard yells and screams and the next thing i know there are policemen all around my house and they had stopped a b	well, got any problems on mockingbird with crime or is that a crime free zone there? no, i don't think there is any such thing, as a crime free zone any longer. i'm afraid you're right. one evening i decided to retire early and heard sirens and noises and thought, oh, well, something's happened on mockingbird and then heard yells and screams and the next thing i know there are policemen all around my house. and they had stopped a, b-,	**speaker 1397:** well, got any problems on mockingbird with crime or is that a crime free zone there? **speaker 1378:** no, i don't think there is any such thing, as a crime free zone any longer. **speaker 1397:** i'm afraid you're right. **speaker 1378:** one evening i decided to retire early and heard sirens and noises and thought, oh, well, something's happened on mockingbird and then heard yells and screams and the next thing i know there are policemen all around my house. and they had stopped a, b-,

8.4 Prompt Engineering Prompts

8.4.1 Task Description 1: Modified Prompt from Benster et al.

> Your task is to choose or generate the most accurate and contextually appropriate transcription from a list of candidate transcriptions. The candidate transcriptions you will receive were ranked by an RNN and are in order from most likely to least likely.
>
> You will be provided with five examples, each containing a list of candidate transcriptions, the corresponding ground truth transcription, and the context before which the transcription occurs. You must use these examples to learn and guide your decisions when selecting or generating transcriptions for new lists. Focus on key differences in the candidates that change the meaning or correctness. Avoid selections with repetitive or nonsensical phrases. In cases of ambiguity, select the option that is most coherent and contextually sound.
>
> Always only respond with either a candidate transcription or a generated transcription. Never provide any introductory or error text, only the transcription you chose. Don't give up. If there are less than two candidate transcriptions, DO NOT generate your own; choose one of them.
>
> Here are the two examples for you to learn from: `[Examples here]`
> And here is the context for your current set of candidate transcriptions: `[Context here]`

8.4.2 Task Description 2: Self-Feedback-Driven Tuning:

This is the task description we used to prompt ChatGPT to assess its errors and return a modified version of Task Description 1 that we could use in our next query.

> Here is the ground truth transcription that you should have returned: `[Ground truth here]` for the following task: `[Original task description here]`. Follow these steps:
> 1. Read your response–`[Model output here]`– and the ground truth.
> 2. Compare your response to the ground truth to identify any differences.
> 3. If your response and the ground truth are not the same, generate a new task description that clarifies the original task or adds a stipulation to improve accuracy. Otherwise, return the original task description.
> 4. Focus on general features that would have been helpful to pay attention to, rather than specific details of the error/pair.
> 5. Do not make edits specific to the error/pair, but rather ensure the new task description addresses the general issues.
> 6. Include the two examples I gave you in the previous task at the end of your task description. Do not modify or add examples to your task description.
> 7. Make sure GPT will always choose the best option, even if none of them seem to fit the context. Never return messages that they don't fit the context. Never generate your own option. If it needs to, GPT can change a word in the candidate sentence to make it fit better, but never add words or delete them.
> 8. Never return anything other than the best candidate transcription.
> Feel free to change the original task description as much as you need.

References

Tyler Benster, Guy Wilson, Reshef Elisha, Francis R Willett, and Shaul Druckmann. 2024. A cross-modal approach to silent speech with llm-enhanced recognition.

Nicholas S. Card, Maitreyee Wairagkar, Carrina Iacobacci, Xianda Hou, Tyler Singer-Clark, Francis R. Willett, Erin M. Kunz, Chaofei Fan, Maryam Vahdatinia, Darrel R. Deo, Eun Young Choi, Matthew F. Glasser, Leigh R. Hochberg, Jaimie M. Henderson, Kiarash Shahlai, David M. Brandman, and Sergey D. Stavisky. 2024. An accurate and rapidly calibrating speech neuroprosthesis. *TBD*, TBD(TBD):TBD. Equal contribution by Maitreyee Wairagkar, Carrina Iacobacci, Tyler Singer-Clark, Francis R. Willett, and Erin M. Kunz.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Yanzhang He, Tara N. Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, Qiao Liang, Deepti Bhatia, Yuan Shangguan, Bo Li, Golan Pundak, Khe Chai Sim, Tom Bagby, Shuo-yiin Chang, Kanishka Rao, and Alexander Gruenstein. 2019. Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Daniel Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual, draft 13. Technical Report 97-02, University of Colorado, Boulder Institute of Cognitive Science, Boulder, CO.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV au2, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts.

Elizabeth Shriberg, Rebecca Bates, Paul Taylor, Andreas Stolcke, and Daniel Jurafsky. 1998. Can prosody aid the automatic classification of dialog acts in conversational speech? *Language and Speech*, 41(3–4):439–487.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, and Daniel Jurafsky. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–371.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30. Curran Associates, Inc.

Frank Willett. 2022. speechbci: A software repository for speech brain-computer interface research. `https://github.com/fwillett/speechBCI`. Accessed: 2024-05-20.

Frank R. Willett, Emily M. Kunz, C. Fan, et al. 2023. A high-performance speech neuroprosthesis. *Nature*, 620:1031–1036.

W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. 2018. The microsoft 2017 conversational speech recognition system. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5934–5938.

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. 2022. Opt: Open pre-trained transformer language models.