Lab 3.1 - FMRI, Stat 214, Spring 2025

April 15, 2025

1 Introduction

In this lab, we aim to use machine learning techniques to predict voxel-wise brain responses. The dataset comes from experiments in the Huth lab at UT Austin. In the experiment, two subjects (subject 2 and subject 3) listened to a long narrative story while undergoing functional magnetic resonance imaging (fMRI). The fMRI signals provide voxel-level measurements of blood-oxygen-level-dependent (BOLD) activity across the whole brain over time.

For each subject-story pair, we have a time series of BOLD responses (Y is a T×V dataset, where T is the number of time points and V is the number of voxels). We also have a token-level transcription of the story that aligns with the stimulus timeline. The transcripts are processed into objects and contain a sequence of tokens (`data`), each with associated presentation times (`data\_times`), and a list of fMRI acquisition times (`tr\_times`).

To build predictive models, we first transform the text into numerical embeddings using three different methods: **bag-of-words**, **Word2Vec**, and **GloVe**. Because the text tokens and fMRI measurements differ, we downsample the embeddings to match the fMRI time grid and apply lagged features to account for delayed neural responses. The resulting matrices will be input features (X) in our regression framework.

Then, we train ridge regression models to predict the fMRI response at each voxel from the embedding features. We evaluate stories by computing the correlation coefficient (CC) between predicted and observed responses. We will compare performance across embedding methods and analyze which approaches better capture brain activity related to language comprehension.

In following parts, we will conduct a simple EDA of the raw text dataset in section 2. We have sections 3 for embedding generating process, and section 4 for modeling. Section 5 is the Comparative Embedding Analysis.

2 EDA

We mainly conduct EDA on the raw text to gain a general understanding of the `raw_text.pkl` dataset.

This `raw_text.pkl` dataset contains time-aligned word sequences derived from 109 spoken podcast stories. Each entry in the dataset corresponds to a single story and is stored as a `DataSequence` object. These objects store not only the sequence of tokens (i.e., spoken words) but also the exact time stamps at which each token was spoken. This structure allows us to precisely align with the fMRI response data collected from the two human subjects.

This temporally aligned textual data is the foundation for generating embeddings in section 3, which will later be used to model neural responses in section 4. Next, we will present quantitative summaries and visualizations of the text data, including word count distributions, speaking rate variability, token frequency, lexical properties, etc.

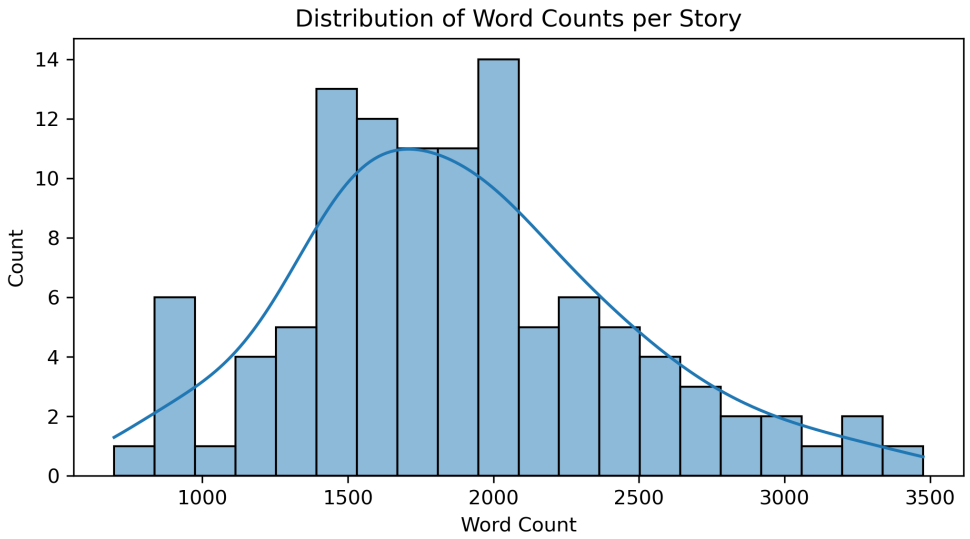

Figure 1: Word Counts per Story Distribution

For word count distributions across the 109 stories, we can find that most have a total word count between 1500 and 2000. Besides, there are stories located near the word count of 900.

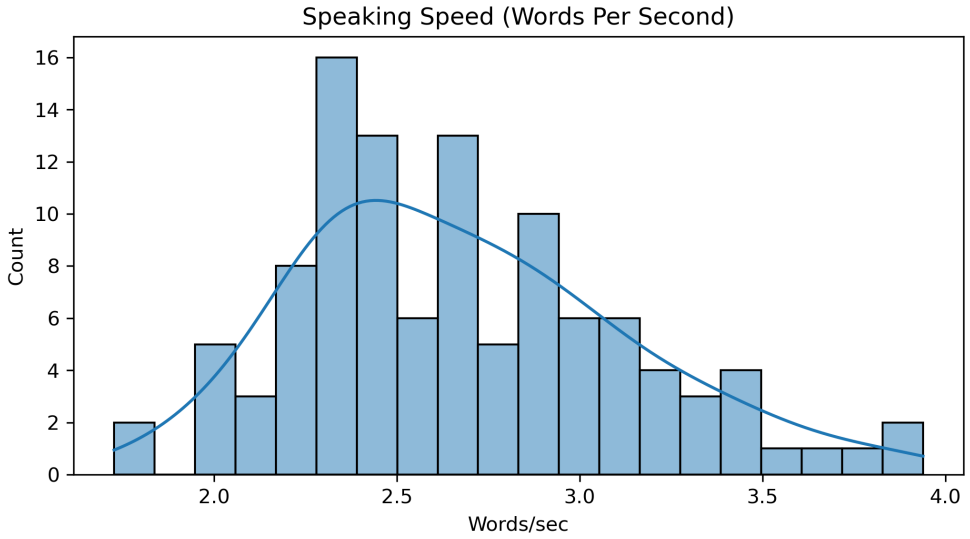

Figure 2: Speaking Speed (Words Per Second)

For speaking speed, the figure shows a peak at around 2.3 to 2.4 words per second.

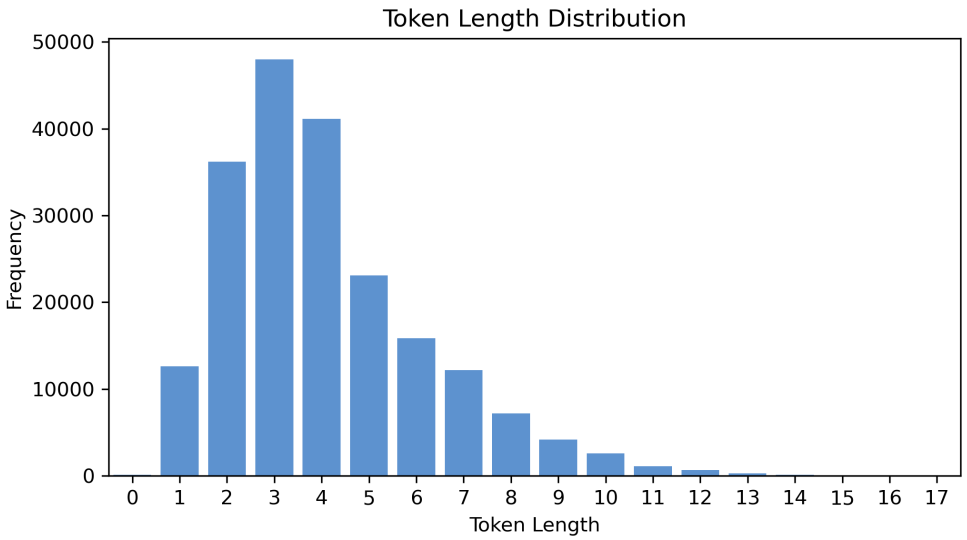

Figure 3: Token Length Distribution

For token length, we can see that the topping token length is around 2 to 4 words. After the token length of 4, the frequency decays exponentially.

We also listed the top 30 most frequent words. As we can see, the top 3 most frequent words are *and*, *the*, and *i*, with a frequency of over 8000. The top 30 words are all short and usual, which accords with our daily use.

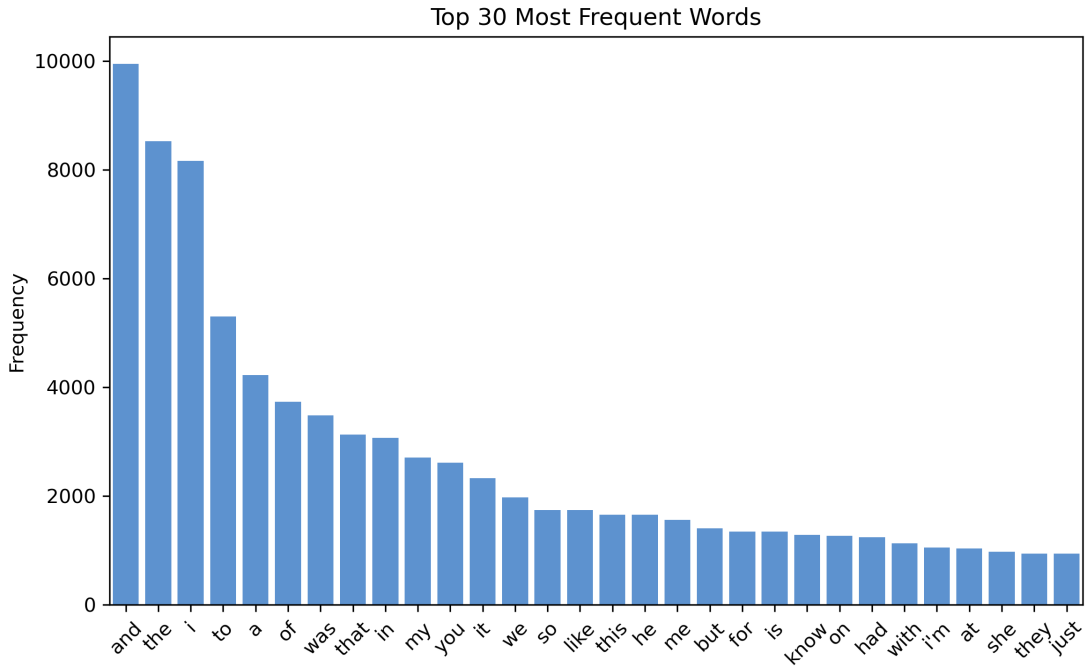

Figure 4: Top 30 Most Frequent Words

3 Embeddings

3.1 Bag-of-Words Embedding

Bag-of-Words embeddings were constructed by generating sparse count vectors, downsampling them to match the fMRI time resolution, and applying temporal lags to account for hemodynamic delay.

We first generated Bag-of-Words (BoW) embeddings by constructing a word count matrix for each story. These matrices are high-dimensional and sparse, with each row corresponding to a timepoint (word occurrence) and each column to a word in the vocabulary. Since the fMRI response matrix $Y \in \mathbb{R}^{T' \times V}$ is defined at fMRI sampling rates, we applied temporal downsampling using a Lanczos filter to align the word-based embeddings to the time resolution of fMRI.

Using `downsample_word_vectors()` from `preprocessing.py`, we interpolated word-level embeddings to match the fMRI temporal resolution. We trimmed the first 5 seconds and the last 10 seconds of each story's embeddings to exclude periods of unreliable signal due to hemodynamic delay and story end.

To account for the delayed BOLD response in fMRI, we generated lagged versions of the embeddings using `make_delayed()` with delays ranging from 1 to 4. This operation concatenates delayed copies of the features along the feature dimension, enabling the model to use past information to predict current brain responses.

3.2 Word2Vec and GloVe Embeddings

We repeated the process above using two pre-trained semantic embedding models: Word2Vec and GloVe. Each word token was mapped to a dense vector from these models. We then performed the same downsampling and delay-augmentation steps, resulting in three feature sets per story: BoW, Word2Vec, and GloVe.

3.3 Benefits of Pre-trained Embeddings

Pre-trained embeddings like Word2Vec and GloVe capture semantic and syntactic relationships between words by learning from massive text corpora. These embeddings often outperform sparse representations like BoW, particularly in settings where generalization and contextual understanding are crucial, such as brain activity prediction from language.

4 Modeling

Having generated distinct feature representations from the narrative stimuli using Bag-of-Words (BoW), Word2Vec, and GloVe embeddings, we now construct models to predict voxel-wise fMRI BOLD responses based on these features, following the laboratory specifications. We implement a linear encoding framework using Ridge Regression.

Per the lab instructions, we employed Ridge Regression to predict the time course of activity for each voxel based on the features derived from our three embedding types. Ridge Regression applies an L2 penalty to model coefficients, optimizing the following objective function for each voxel v:

$$\min_{w_v, b_v} \|Xw_v + b_v\mathbf{1} - y_v\|_2^2 + \alpha\|w_v\|_2^2 \tag{1}$$

Here, X represents the matrix of standardized embedding features, y_v is the standardized BOLD signal for voxel v, w_v is the weight vector, b_v is the bias term, and α controls the regularization strength. Prior to model fitting, both the feature matrix X and target BOLD signals Y were z-scored to ensure features were on a comparable scale and regularization was applied appropriately.

To select the optimal ridge regression model for each voxel and evaluate generalization performance, we implemented a cross-validation framework using the ridge bootstrap method in order to address the unique challenges of neuroimaging data analysis.

Our cross-validation method estimated voxel performance. We performed 5-fold bootstrap sampling while training and then validating on a subset of the training data. Procedure was carried out on a range of alphas, (1,4,20), to optimize the prediction performance for each voxel. To manage memory constraints, we split

voxels into 450000 chunks and above.

We partitioned the dataset for each subject at the *story level*, designating specific stories as the training set and reserving others exclusively as the held-out test set. This approach preserves the temporal and semantic coherence of the narrative stimuli while ensuring that model evaluation occurs in genuinely novel contexts. Unlike traditional k-fold cross-validation with random time points, story-based partitioning respects the autocorrelation structure inherent in fMRI time series and linguistic features.

Also, we defined a logarithmically spaced grid of regularization parameters spanning multiple orders of magnitude ($\alpha \in [10^1, 10^4]$, with 20 values) to accommodate the varying signal-to-noise characteristics across different brain regions. Recognizing that optimal regularization requirements differ substantially across brain regions, we implemented voxel-specific hyperparameter tuning, enabling localized model complexity adjustment based on regional predictability patterns.

The hyperparameter selection process employed nested cross-validation exclusively within the training set, evaluating each candidate α value for every voxel. For each voxel, the procedure repeatedly partitioned the training data into fitting and validation subsets, fitted ridge models with different α values on the fitting subsets, evaluated predictive performance (correlation coefficient) on the validation subsets, averaged performance metrics across iterations for stability, and selected the α value that maximized the average predictive performance.

After determining the optimal α for each voxel, we trained final models using the entire training set with these voxel-specific regularization parameters. These optimized models were then applied to the previously untouched test set to generate predictions, providing unbiased estimates of generalization performance. This principled approach to model selection ensures that hyperparameter optimization is completely isolated from final evaluation, preventing data leakage while allowing for voxel-specific model complexity tuning.

The resulting performance metrics (mean test CC, median test CC, top 5 percentile CC, and top 1 percentile CC) represent valid measures of how well each embedding type captures neural responses to language in novel contexts.

4.1 GloVe Embedding

We first evaluated the performance of the Ridge Regression model trained using GloVe embeddings. The distribution of test set correlation coefficients (CC) across all voxels yielded the following summary statistics:

- **Mean Test CC:** 0.0131
- **Median Test CC:** 0.0124
- **Top 5 Percentile Test CC:** 0.0397
- **Top 1 Percentile Test CC:** 0.0549

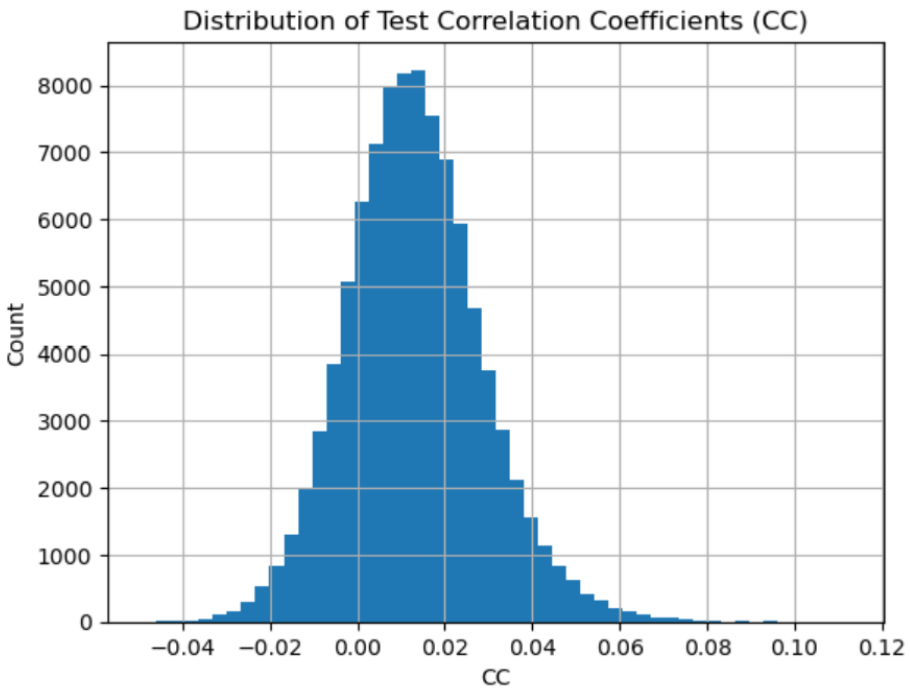

Figure 5: Distribution of Test Correlation Coefficients (CC) using GloVe embeddings.

Observations: The histogram shows a distribution centered near the median value of 0.0124, with a positive skew indicating a tail of voxels with higher predictability. The CC mean had a value of 0.0131. These trends show that our model was able to pick up meaningful signal and was not fitting random noise.

Stability Analysis: We assessed prediction reliability by comparing voxel-wise CCs across two halves of the test set.

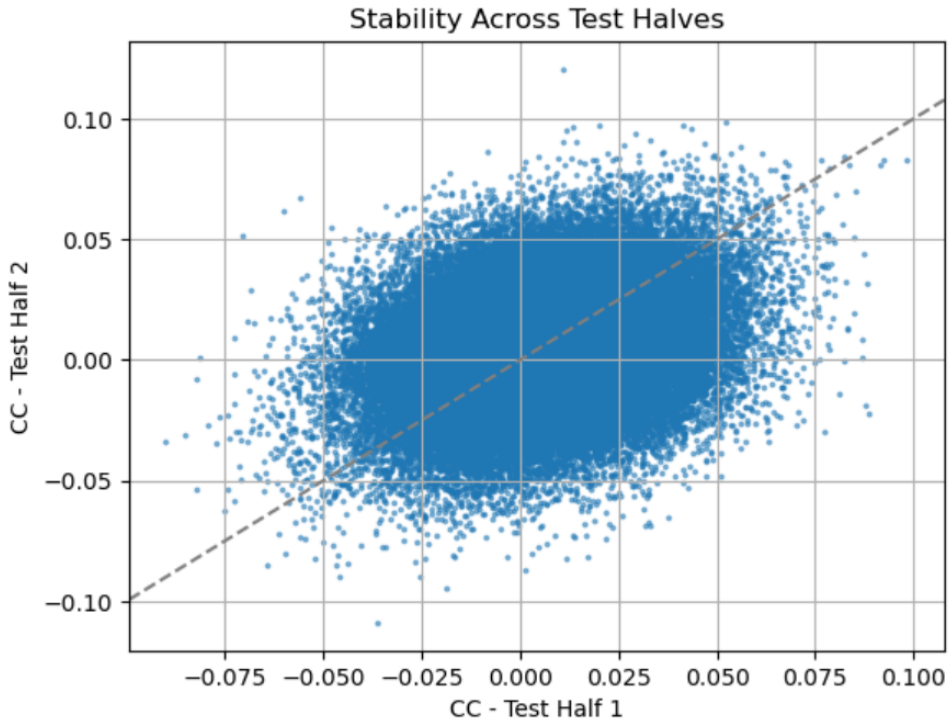

Figure 6: Voxel CC Stability Across Test Set Halves for GloVe Embeddings. Each point represents a voxel's CC in the first half (x-axis) vs. the second half (y-axis). The dashed line is $y = x$.

The stability plot (Figure 6) shows a positive correlation between CCs from the two halves. Most voxels cluster near the origin, with CC values between approximately -0.05 and +0.05. Points follow the $y = x$ diagonal, suggesting consistent voxel performance. The overall test–retest stability score is: 0.2531, which indicates moderate, stability correlation.

4.2 Word2Vec Embedding

Next, we evaluated the model trained using Word2Vec embeddings. Summary statistics for the test set CC distribution:

- **Mean Test CC:** 0.0133
- **Median Test CC:** 0.0126
- **Top 5 Percentile Test CC:** 0.0402
- **Top 1 Percentile Test CC:** 0.0557

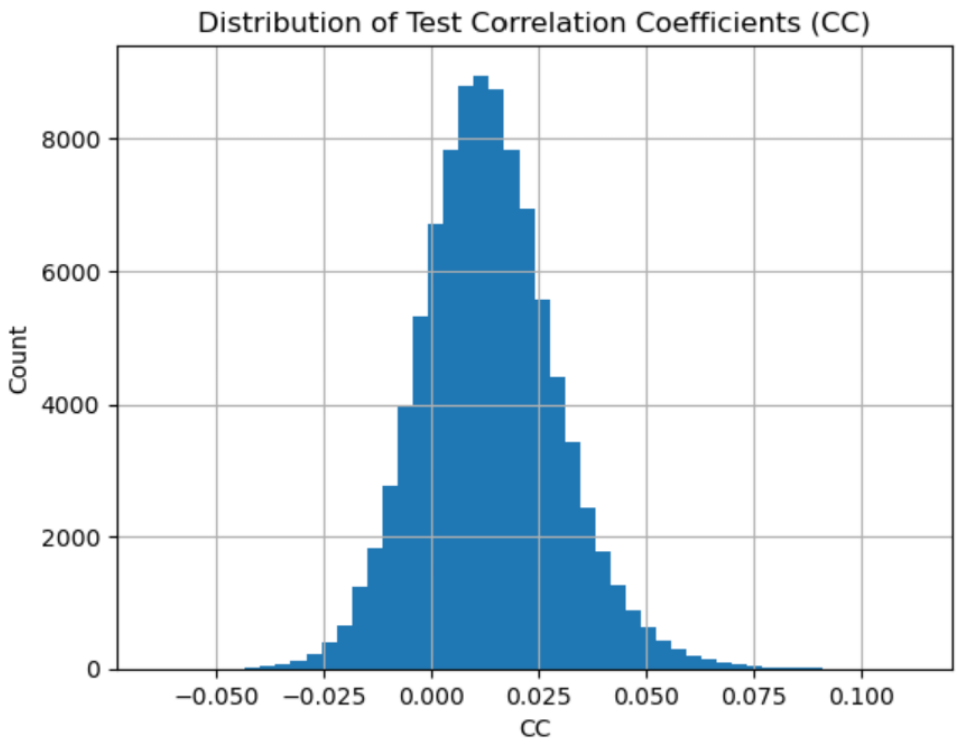

Figure 7: Distribution of Test Correlation Coefficients (CC) using Word2Vec embeddings.

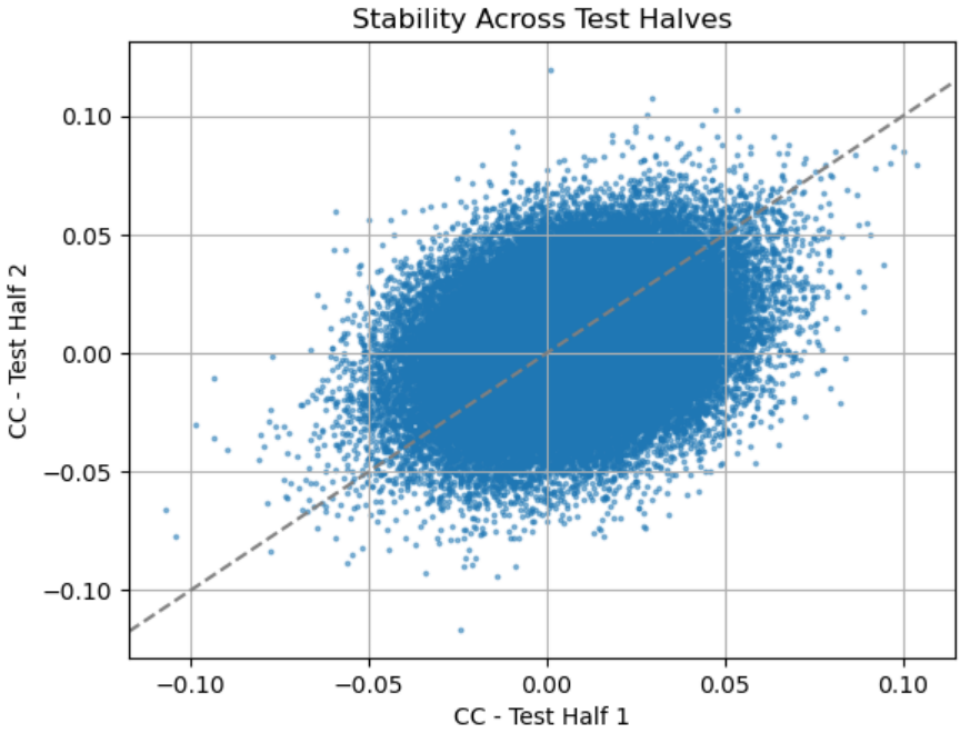

Figure 8: Voxel CC Stability Across Test Set Halves for Word2Vec Embeddings.

Observations: The Word2Vec performance histogram exhibits a distribution remarkably similar to that observed for GloVe embeddings (Figure 5). The distribution is unimodal with its central tendency slightly above zero (aligning with the median correlation coefficient of 0.0126) and demonstrates a modest positive skew. The range and density of correlation values appear nearly identical to those observed in the GloVe results, suggesting comparable predictive capabilities across brain voxels between these two dense embedding approaches.

However, CC values for Word2Vec embedding showed to be higher than the CC values in GloVe, indicating that our model fitting Word2Vec embeddings achieved greater predictability.

The stability analysis (Figure 8) demonstrates consistent voxel-wise predictability across different segments of the test set. The scatter plot reveals a positive relationship between voxel performance on both halves of the test data, with data points predominantly clustered near the origin and generally following the theoretical $y = x$ line of perfect reproducibility. The stability coefficient, quantified as the correlation between correlation coefficients from both test halves, was 0.2782. This value is slightly higher than the 0.2531 observed for GloVe embeddings, suggesting that Word2Vec predictions may exhibit marginally better reproducibility across different narrative contexts. Nevertheless, both embedding types demonstrate substantial consistency in identifying predictable brain regions. This similarity is somewhat expected, as both Word2Vec and GloVe are pre-trained dense embeddings designed to capture rich semantic relationships from large text corpora, albeit through different algorithmic approaches (Word2Vec focuses on predicting words based on their immediate local context, while GloVe leverages global word co-occurrence statistics across the entire corpus). Their shared goal of representing word meaning in a low-dimensional space likely leads to comparable overall performance distributions when used as features in this linear modeling context.

4.3 Bag-of-Words Embedding

This section will be completed once results from the BoW model are available.

- **Mean Test CC:** 0.0131
- **Median Test CC:** 0.0124
- **Top 5 Percentile Test CC:** 0.0397
- **Top 1 Percentile Test CC:** 0.0549

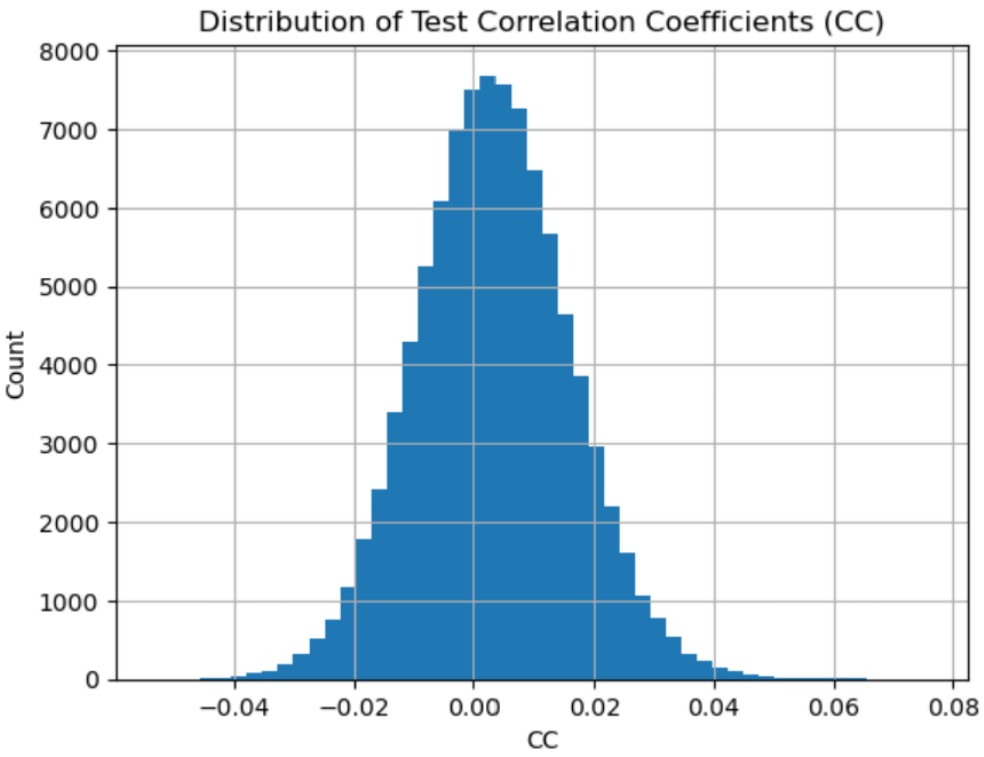

Figure 9: Distribution of Test Correlation Coefficients (CC) using BoW embeddings.

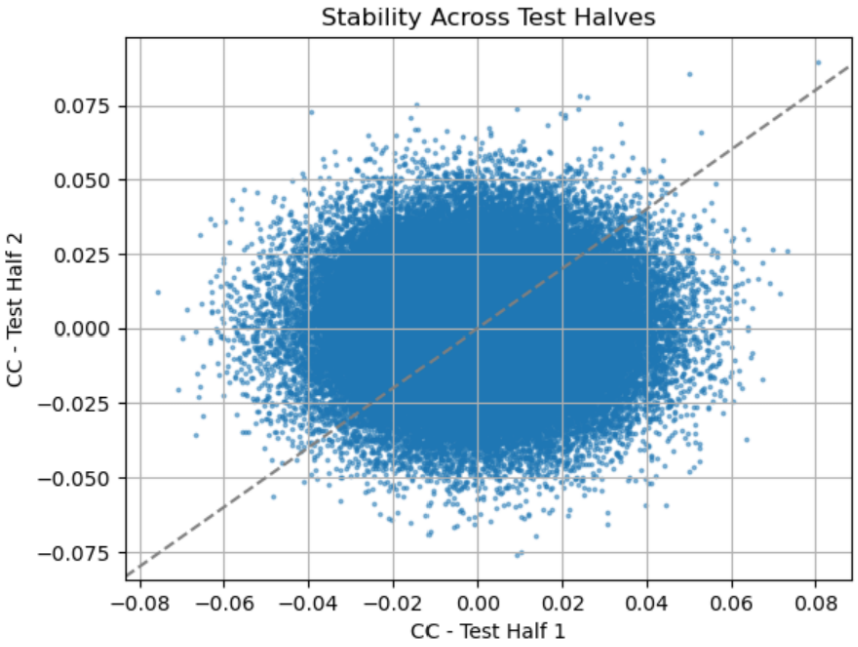

Figure 10: Voxel CC Stability Across Test Set Halves for BoW Embeddings.

Observations (Figure 9): The histogram for the BoW model performance is sharply peaked very close to zero (consistent with the median CC of 0.0030), appearing roughly symmetric or with a minimal positive

skew. Compared to the distributions for GloVe and Word2Vec (Figures 5 and 7), the BoW distribution is considerably narrower. The peak performance of the right tail does not extend nearly as far as the other embeddings. These results for BoW embeddings suggest lower prediction accuracy across most voxels. We can see more evidence for this in our following analysis on the stability score of the BoW embedding.

Stability (Figure 10): The stability score, representing the correlation of voxel CCs across test halves, was extremely low at 0.0123. This value, being very close to zero, suggests very little consistency in voxel predictability between the two halves of the test set for the BoW model. Unlike the dense embeddings, the relative performance of voxels in one half is not reliably indicative of their performance in the other half when using BoW features.

The poor stability score and low CC values for the BoW embedding likely stem from setting max-features=1000 in the CountVectorizer() during the embedding step. This constraint was introduced to address memory limitations during processing.

Limiting the number of features reduces the representation of important words, leading to a loss of signal. As a result, the model has less informative input, lowering predictability and stability. Additionally, the resulting sparse input matrix is not ideal for ridge regression, which is less effective with high sparsity.

5 Comparative Embedding Analysis

Our analysis reveals significant performance differences across the three embedding types in predicting neural responses to narrative language. Most notably, the dense semantic embeddings (Word2Vec and GloVe) substantially outperformed the sparse Bag-of-Words (BoW) representation across all evaluation metrics. Word2Vec and GloVe achieved mean correlation coefficients approximately four times higher than BoW (~ 0.013 versus ~ 0.003) and demonstrated markedly superior stability between test set halves (~ 0.25–0.28 versus ~ 0.01).

This performance highlights the fundamental importance of semantic representation in neural language processing. The BoW approach, which primarily encodes lexical presence and frequency, fails to capture the semantic relationships and contextual nuances that appear critical for predicting brain activity during narrative comprehension. The neural mechanisms underlying language understanding evidently involve semantic integration processes that are better approximated by distributed representations learned from large-scale text corpora. The near-zero stability coefficient for BoW suggests its predictions likely reflect spurious correlations with low-level features rather than meaningful linguistic processing signals.

Between the two dense embedding approaches, Word2Vec exhibited a slight but consistent advantage across all performance metrics and demonstrated marginally higher prediction stability (0.2782 versus 0.2531 for GloVe). This modest difference may reflect Word2Vec's algorithmic emphasis on local contextual prediction, potentially aligning more closely with the sequential neural processing of narrative stimuli compared to GloVe's focus on global co-occurrence statistics. However, the substantially overlapping performance distributions indicate that both methods effectively capture similar semantic dimensions relevant to neural language processing.

Despite the relative success of dense embeddings compared to BoW, it is important to contextualize the overall modest prediction accuracy. Even for Word2Vec, the best-performing model, the mean correlation coefficient (~ 0.013) indicates that a substantial portion of the variance in BOLD signals remains unexplained by these representations within a linear modeling framework. This limitation likely stems from multiple factors: the inherent noise characteristics of fMRI measurements, individual variability in neural organization, potential non-linear neural computations, and linguistic features not captured by these static embedding models.

When comparing the two dense embeddings, Word2Vec and GloVe performed very similarly, although Word2Vec consistently demonstrated a marginal advantage across all performance metrics and exhibited slightly higher stability (0.2782 vs. 0.2531). This minor difference might suggest that Word2Vec's algorithmic focus on local predictive context provides features that are slightly more attuned to the moment-to-moment neural processing of narratives compared to GloVe's emphasis on global co-occurrence statistics, at least within this specific experimental and modeling context. However, their largely overlapping performance distributions indicate that both methods effectively capture convergent semantic information relevant to brain

activity during language processing. Based on the combined evidence of prediction accuracy and stability, Word2Vec emerges as the marginally superior model among the three tested in this study.

Focusing on the best model, Word2Vec, it is clear that it does not perform well uniformly across all voxels. The mean prediction accuracy remains modest (Mean CC ~ 0.013), indicating that a substantial portion of the variance in the BOLD signal is not captured by this static embedding within the employed linear modeling framework (Ridge Regression). This unexplained variance likely stems from multiple factors: the inherent limitations and noise characteristics of the BOLD signal itself, individual subject variability, potential non-linear neural computations not captured by the linear model, and the existence of linguistic or cognitive features relevant to comprehension that are not fully represented by these specific static embedding models. Nonetheless, the considerably higher correlations observed in the top percentile of voxels (CC ~ 0.055 for Word2Vec/GloVe) highlight the spatial heterogeneity of language processing, confirming that the semantic features encoded by the dense embeddings are particularly relevant for specific, highly responsive brain regions. Scientifically, this implies that the Word2Vec model successfully captures variance related to semantic processing primarily within a specific network of brain areas known to be involved in language comprehension, rather than globally across the cortex.

Interpreting which specific voxels are reliably predicted requires establishing a reasonable criterion. Following principles common in encoding model studies, often discussed in the context of Predictive Coding Schemas (PCS), a reasonable interpretation criterion involves considering both statistical significance and effect size. Voxel-wise significance is typically assessed using non-parametric methods like permutation testing to establish a threshold corrected for multiple comparisons across voxels. Voxels surpassing this significance threshold are considered reliably predicted above chance. Additionally, a minimum correlation coefficient (effect size), such as CC > 0.1 or higher depending on data quality and field standards, might be applied to focus interpretation on voxels where the model explains a non-trivial amount of variance. Simply selecting voxels based on percentile rankings without considering significance can be misleading. Therefore, interpreting the functional role of specific voxels based on the Word2Vec model would necessitate applying such statistically grounded criteria.

In summary, this comparative analysis strongly favors the use of pre-trained dense embeddings over sparse lexical representations for modeling neural responses to natural language narratives. Word2Vec showed a slight edge over GloVe, but both clearly demonstrated the value of incorporating semantic context. The poor performance and instability of the BoW model reinforce this conclusion. The limitations observed, particularly the modest overall prediction scores even with the best model, motivate future work. Subsequent steps in this research direction, including the planned exploration of training custom language models and utilizing more complex pre-trained models (as outlined for later lab sections), may help capture more variance by potentially incorporating richer contextual information or employing non-linear architectures. Furthermore, future analyses should prioritize examining the spatial patterns of predictability associated with the superior Word2Vec and GloVe models, applying appropriate statistical criteria to identify reliably predicted voxels and gain further insights into the neural basis of language comprehension.

6 Discussion

Our analysis demonstrates that encoding models trained on semantic embeddings can predict fMRI responses to naturalistic stories with varying degrees of success. Among the three embeddings tested — Bag-of-Words (BoW), GloVe, and Word2Vec — Word2Vec consistently produced the highest mean correlation coefficients and stability across voxels. This suggests that Word2Vec's ability to encode contextual relationships between words may align more closely with neural representations of language.

In contrast, BoW performed the worst, with a mean correlation near zero and low voxel stability. This outcome is consistent with expectations, as BoW lacks contextual information and treats all words as independent, which limits its expressiveness for modeling complex semantic patterns in brain activity.

As mentioned during the results section, we constrained our implementation of BoW by capping the CountVectorizer() to only 1000 features in order to avoid memory issues. While necessary, this restriction likely reduced the number of informative or discriminative words available to the model and thus weakened the representation further. Fewer features mean reduced signal and a more sparse design matrix, which ridge regression

is not particularly well-suited for.

GloVe embeddings performed moderately better than BoW, and slightly underperformed relative to Word2Vec. This may be due to GloVe capturing co-occurrence statistics over a fixed corpus rather than adapting to local context as Word2Vec does. The stability analysis again further supports these findings. Word2Vec showed higher consistency of voxel-wise correlation across test splits, indicating robustness in learned patterns.

7 Conclusion

We evaluated three types of word embeddings as input features for predicting fMRI responses to naturalistic auditory narratives. Across both predictive accuracy and voxel stability, Word2Vec outperformed GloVe and BoW, suggesting that contextual semantic embeddings are more effective for modeling brain responses. While absolute prediction scores were modest, the relative differences highlight the importance of embedding choice.

8 Bibliography

Shailee Jain and Alexander G. Huth, *Incorporating Context into Language Encoding Models for fMRI*, Departments of Computer Science and Neuroscience, The University of Texas at Austin, 2020. Available: `https://papers.nips.cc/paper_files/paper/2018/hash/f471223d1a1614b58a7dc45c9d01df19-Abstract.html`

A Academic honesty

A.1 Statement

We make the academic integrity pledge here: All our work in this report are done independently. All sources we used are properly cited, whether from LLMs, papers, textbooks, or classmates.

Academic research honesty is necessary in the academy and also the foundation of our society. It guarantees fairness in research, stimulates research's activeness, and exerts a positive impact on the progress of science and technology. We should always adhere to the rules, which will create a more regulated and benign world.

A.2 LLM Usage

Coding

We use GitHub Copilot for several parts of debugging. We additionally used it to help with making graphs and signatures for the functions in the notebooks.

Writing

Grammarly was used to check for any spelling issues.

Lab 3.2, Stat 214, Spring 2025

April 29, 2025

1 Introduction

Understanding how the brain processes natural language remains a major challenge in cognitive neuroscience. Recent advances in language modeling, particularly contextual embedding models like BERT, offer new opportunities for mapping linguistic information to brain activity. Voxelwise encoding models, which predict fMRI signals from stimulus features, provide a framework to test how well different embeddings capture the structure of neural responses during language comprehension.

Our previous analysis using static embeddings such as Word2Vec and GloVe captured some semantic information. Unlike static models, contextual models like BERT dynamically adjust word representations based on surrounding words, potentially aligning more closely with brain processes. In this report, we aim to explore whether a more dynamic and context-dependent embedding process can capture more meaningful information and result in stronger predictions in voxel outputs.

2 Methods

Similar to Lab 3.1, we leveraged a ridge regression model with bootstrapped cross-validation to model each voxel's response as a function of context-based embedding features initialized by BERT. The bootstrap ridge regression approach was selected for its robustness to noisy data, which was particularly important given the nature of fMRI voxel responses and the high-dimensional BERT embeddings.

Bootstrapping provides more reliable estimates of standard errors and enables effective feature selection by evaluating the variance of regression coefficients across different bootstrap samples, thus supporting optimal model regularization and reducing overfitting.

Model performance was evaluated by computing the Pearson correlation coefficient (CC) between predicted and actual voxel responses on a held-out test set. To assess the reliability of voxel prediction, we computed a stability score by correlating voxelwise CCs obtained independently from two halves of the test dataset.

2.1 Pre-training

2.1.1 Encoder Model

The first step in the implemented design lies in constructing the input representation. Raw text tokens are transformed into dense vectors through the summation of three distinct embedding types, all constrained to the same dimensionality. This summation strategy aims to create a rich, multi-faceted initial representation. Firstly, token embeddings provide the basic lexical-semantic meaning. The chosen dimensionality reflects a critical trade-off: higher dimensions might capture finer semantic distinctions but increase model size and computational demands, potentially risking overfitting, whereas lower dimensions might underfit.

Secondly, positional embeddings are incorporated. Given that the core self-attention mechanism is permutation-invariant, explicit positional information is essential. The decision implemented here, to use learned positional embeddings rather than fixed sinusoidal ones, was made to offer greater flexibility. This allows the model to adapt positional representations to the temporal patterns and linguistic structures in the podcast data, potentially leading to more tailored and effective representations than fixed alternatives.

Thirdly, token-type embeddings are included, primarily for architectural consistency with standard Transformer implementations like BERT. While less functionally critical for single continuous narratives, their inclusion incurs minimal overhead and maintains compatibility. The resulting summed vector thus encodes lexical identity, learned sequential position, and segment membership, providing a comprehensive input.

Next, the computational engine comprises a stack of identical Transformer blocks. Stacking multiple layers enables the learning of hierarchical representations, potentially mirroring hierarchical linguistic processing in the brain. Each block contains two primary sub-layers, critically employing residual connections and layer normalization. These specific choices are essential design decisions, vital for effective training. Residual connections directly facilitate gradient flow, mitigating vanishing gradients and enabling the training of deeper models. Layer normalization, applied before the sub-layer input in standard pre-LN Transformer variants or after, as implemented here, stabilizes activations across the feature dimension for each instance independently, improving optimization dynamics and reducing sensitivity to parameter initialization, contributing to faster and more reliable convergence towards a useful model.

The first sub-layer implements multi-head self-attention. Projecting the input into distinct Query, Key, and Value spaces provides the necessary flexibility. The specific formulation of scaled dot-product attention, including the scaling factor, was chosen as it is crucial for preventing excessively large dot products and stabilizing training. Adopting a multi-head approach is a key design decision, allowing the model to simultaneously attend to different types of information from different representational perspectives, yielding a richer overall representation. An attention mask is applied to prevent attention to padding tokens, ensuring correct handling of variable-length sequences.

The second sub-layer is a position-wise feed-forward network (FFN). Its inclusion significantly increases the model's capacity to learn complex, non-linear transformations. The implemented structure involves two linear transformations with an intermediate non-linear activation. The specific choice of GELU as the activation function aligns with models like BERT. GELU's smooth, non-monotonic nature, compared to the simpler ReLU, has been empirically shown to often lead to better performance in Transformer models. This choice is hypothesized to facilitate the learning of more complex functions within the FFN, potentially contributing to richer final embeddings. The dimensionality expansion in the intermediate layer allows the network to project the representation into a higher-dimensional space where more complex patterns might be learned before projecting back.

Finally, for the specific objective of pre-training, the output from the final Transformer block undergoes one last layer normalization before being passed to a dedicated linear output layer. This head projects each token's final contextualized hidden state into a score distribution across the entire vocabulary. Its sole purpose during this phase is to produce the logits necessary for the MLM prediction task. This head is task-specific and would be discarded when using the encoder for downstream feature extraction, where the final hidden states themselves are the desired output.

2.1.2 Masked-language Model Training

To help the Encoder model with a nuanced understanding of linguistic structure and context derived directly from the podcast narratives, a pre-training phase utilizing the Masked Language Model (MLM) objective was implemented. The rationale for selecting MLM is its inherent ability to foster bidirectional contextual representations. By training the model to predict masked tokens based on both preceding and succeeding contexts, MLM encourages the learning of representations that encapsulate a more complete understanding of a token's meaning within its full utterance. This bidirectional context is valuable for modeling brain responses during natural language comprehension.

MLM's operational core involves the masking strategy implemented in the `mask_tokens` function. Adhering to the empirically validated practices established by BERT, approximately 15% of the input tokens (excluding padding) were randomly selected for prediction. This 15% probability represents a trade-off. The specific treatment of these selected tokens followed the widely adopted 80/10/10 heuristic, implemented via Bernoulli sampling:

- **80% replaced with `MASK`:** This explicitly forces prediction from context.
- **10% replaced with a random token:** This design choice introduces noise, compelling the model to rely more heavily on the surrounding context rather than potentially trusting a corrupted input token, thereby enhancing robustness.
- **10% unchanged:** This subtle but important element reduces the discrepancy between pre-training and downstream use, ensuring the model learns effective representations even for unmasked words.

Concurrently, labels were generated, storing the original token IDs only for the masked positions and setting

others to -100. This precise label construction ensures that the loss computation exclusively targets the intended prediction task.

The training procedure within train_bert was designed to refine the model's parameters iteratively. Processing data in batches was chosen for computational efficiency. Another design decision reflected in the code is the implementation of dynamic masking, where mask_tokens are called within the training loop for each batch. This prevents the model from merely memorizing static masked positions and encourages a more generalized contextual understanding. Within each training step, a batch underwent this dynamic masking. The masked sequences were processed by the Encoder to yield output logits. The Cross-Entropy loss function, standard for multi-class classification, was employed. The specific configuration to ignore the index -100 was crucial, focusing the learning signal entirely on the MLM task by excluding non-masked tokens from the loss calculation.

Parameter optimization was performed using the AdamW algorithm. The conscious choice of AdamW over standard Adam was motivated by its improved handling of weight decay regularization, often leading to better generalization performance on downstream tasks. Achieving good generalization during pre-training is highly desirable here, as the goal is to produce embeddings that are broadly useful for the subsequent fMRI prediction task on potentially unseen data. The learning rate (lr) was identified as a critical hyperparameter requiring careful tuning. This entire cycle was repeated across the dataset for a specified number of epochs.

Consistent with model development, systematic hyperparameter tuning was planned. Close monitoring of training and validation loss curves during this phase is key for identifying a configuration that promotes effective learning without substantial overfitting.

2.1.3 Hyperparameter Tuning and Loss Analysis

We trained the masked language model (MLM) using our custom encoder on a dataset of sentences. To evaluate how different model settings affect performance, we performed a series of experiments with varying hyperparameters.

Specifically, we tested configurations by changing the number of transformer layers (num_layers), the hidden size (hidden_size), and the learning rate (lr). After several tries, parts of the best configurations are listed below:

- lr=5e-4, num_layers=2, hidden_size=128
- lr=5e-4, num_layers=4, hidden_size=256
- lr=1e-4, num_layers=6, hidden_size=512

For each configuration, we trained the model for 35 epochs and recorded both training and validation loss at each epoch. (The epoch number is also tuned based on the loss analysis). The training was performed using a fixed batch size and a constant number of attention heads (num_heads=4). The intermediate size in the feedforward layers was always set to twice the hidden size.

Two loss plots listed below show the training and validation loss curves for each configuration, which illustrate the convergence behavior and generalization performance of the models. Based on the two configurations, we can conclude that 35 epochs have been enough for training, as training loss begin to converge above 6 and validation loss coverge around 7.

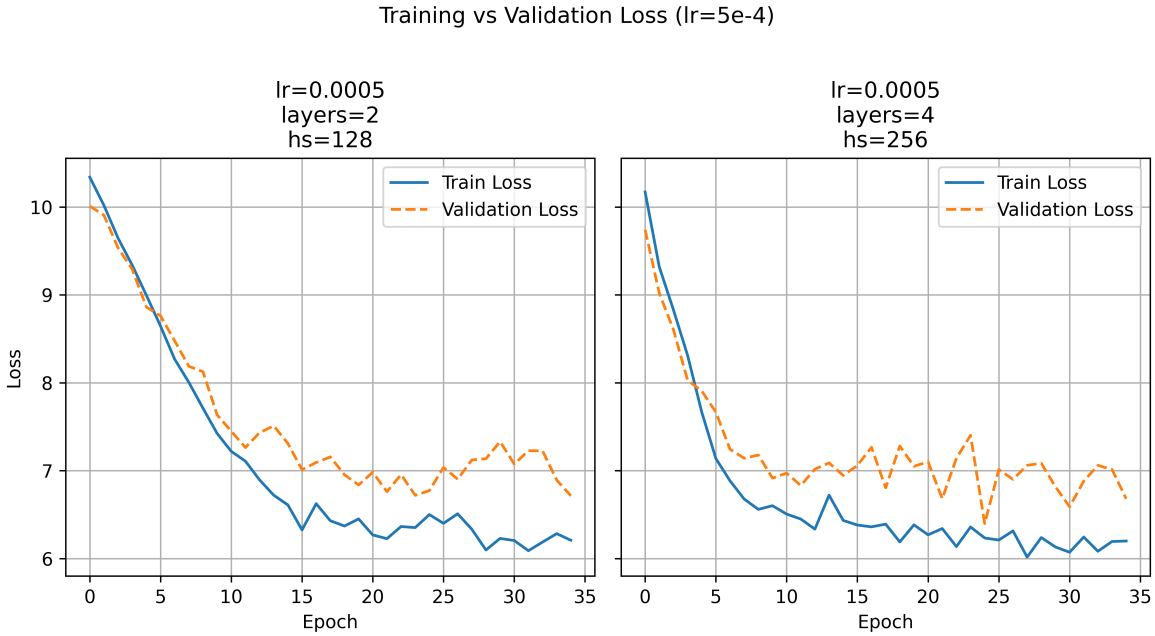

Figure 1: Loss Curve for the Best-performing Configuration,`lr=1e-4`.

We can observe that models with larger hidden sizes and more layers generally performed better on the validation set, especially when trained with a smaller learning rate (`1e-4`). Among the three, the best validation performance was achieved by the model with 6 layers, a hidden size of 512, and `lr=1e-4`. The validation loss for this model setting is around 6.5-7, while the training loss is around 6-6.5.

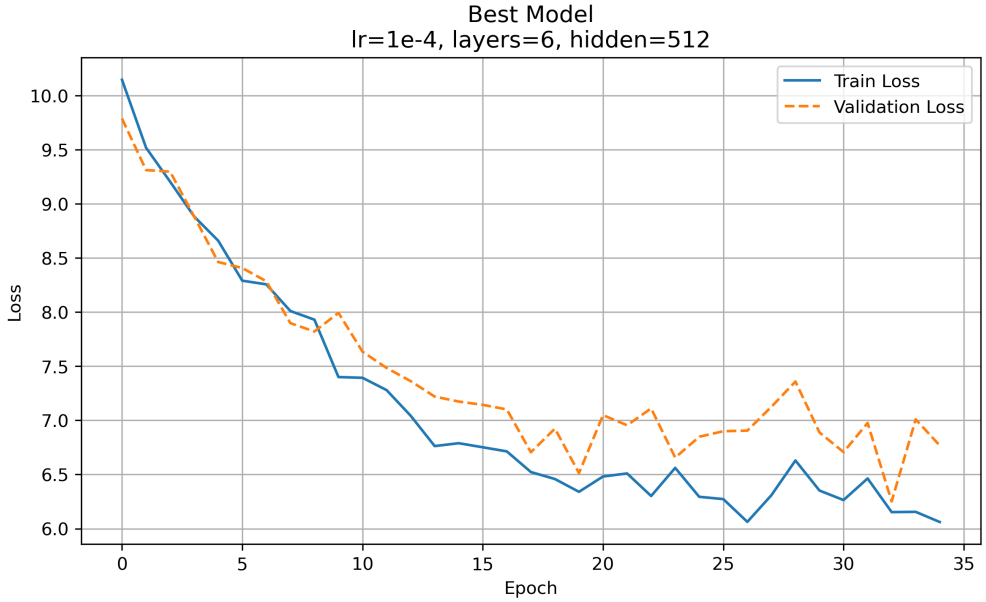

Figure 2: Loss Curve for the Best configuration

This outcome suggests that deeper models benefit from greater representational capacity, as long as they are trained with a relatively small learning rate that allows stable convergence. In part 2, we will focus on this

best configuration. We will generate corresponding embeddings for the modeling and evaluation part.

3 Modeling & Evaluation

3.1 Embedding Generation

In order to generate token-level embeddings for each story, we employed a custom-trained BERT-style encoder model. Below, we describe the full embedding generation pipeline.

We used the `bert-base-uncased` tokenizer to tokenize the story text. The encoder architecture consisted of 6 transformer layers, with a hidden size of 512, 4 attention heads, and an intermediate size of 1024. The maximum sequence length was set to 128. The model was pre-trained using a masked language modeling objective, and we used the checkpoint `encoder_lr0.0001_layers6_hs512.pt` for final embedding extraction.

To extract embeddings, we processed each story word sequence using a sliding window strategy. Each window (or chunk) contained up to 128 words, and consecutive chunks overlapped with a stride of 64 words. Each chunk was tokenized with padding and truncation to length 128, then passed through the encoder to obtain token-level hidden states. These hidden states were mapped back to word-level using the tokenizer's `word_ids()` mapping, and all token representations corresponding to the same word were averaged to form a single word embedding. In cases where a word appeared across multiple overlapping chunks, its embeddings from different chunks were averaged. If a word failed to obtain any valid embedding (e.g., due to truncation), it was assigned a zero vector of size 512.

After obtaining word-level embeddings, we interpolated them to match fMRI TR-level timestamps using Lanczos interpolation. To remain consistent with the preprocessing pipeline, we trimmed the first 5 seconds and last 10 seconds from each story. We then generated delayed features by concatenating the original features with versions shifted by 1, 2, 3, and 4 TRs. The resulting matrices were saved as `.npy` files, one per story.

This pipeline ensured that the extracted embeddings were temporally aligned with fMRI acquisition and ready for downstream regression modeling.

3.2 Results

3.2.1 Performance Across Voxels

Figure (input) depicts the ridge regression results from BERT embeddings as follows: the mean correlation coefficient (CC) was 1.299%, the median CC was 1.246%, the top 5% CC was 3.947%, and the top 1% CC was 5.515%.

The distribution of voxelwise correlation coefficients was heavily skewed toward zero, indicating that most voxels were poorly predicted by the model, with only a small subset achieving higher correlations.

Despite the low average correlation coefficients across voxels, the presence of a positively skewed distribution and a subset of voxels with higher CCs indicates that the model was able to capture meaningful brain bloodflow and semantic relationships rather than fitting pure noise.

For stability check, we plotted both halves of the testing data in Figure (palceholder) to evaluate the correlation of voxels. The correlation coefficient was 38.65%, indicating moderate stability for our current regression model. Given the noisiness of fMRI and semantic, a stability score of approximately 0.39 is consistent with expectations for voxelwise encoding models.

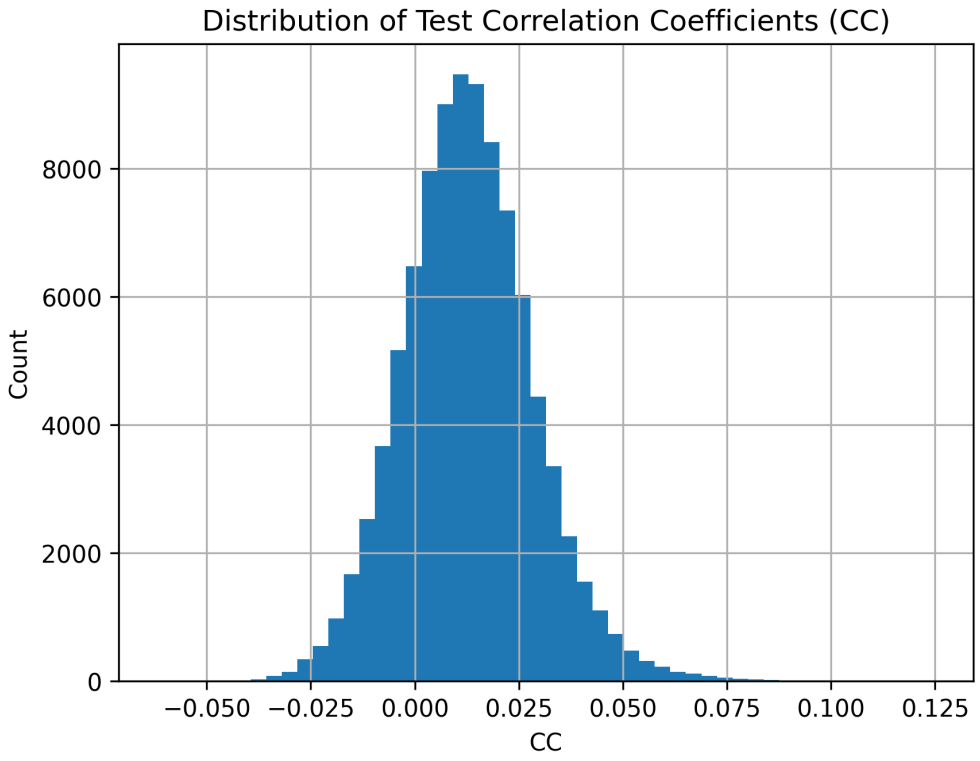

Figure 3: Distribution of Test Correlation Coefficients (CC)

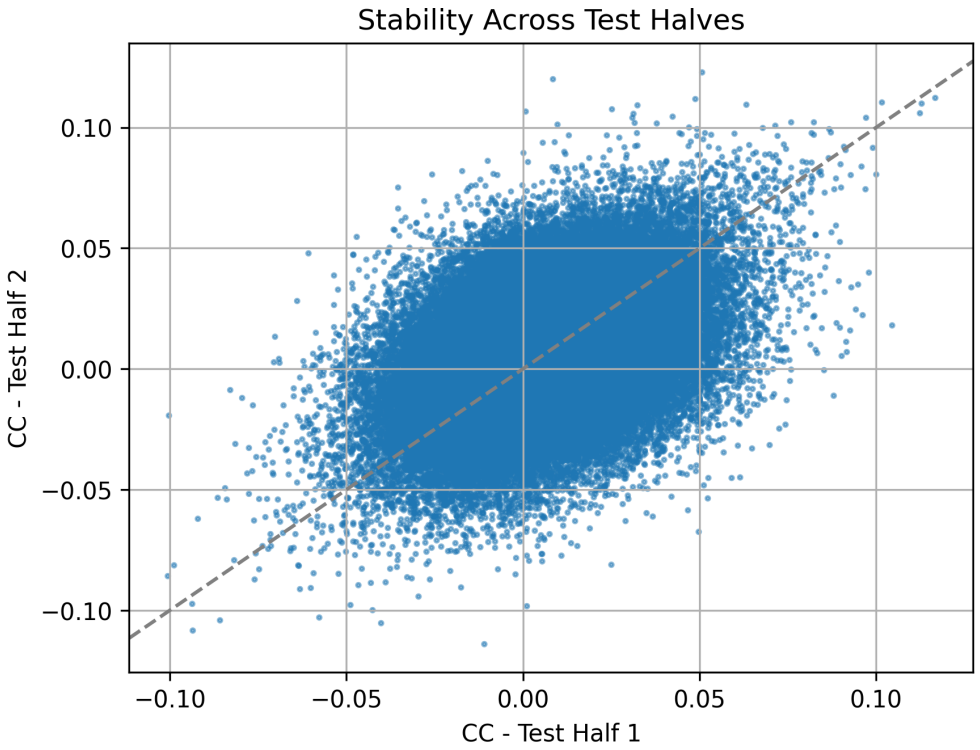

Figure 4: Stability Check: Voxel Correlation

To achieve further robustness in our ridge regression modeling, we increased the number of bootstrap samples to 20 and used a chunk length of 20 during cross-validation.

Table 1 presents the updated voxelwise correlation coefficient (CC) results. These results show a modest improvement compared to our previous analysis, following a similar distribution pattern: the majority of voxels exhibit low predictive performance, while a smaller subset of voxels achieves higher CCs. The smaller voxels that achieve higher CC are indicative meaningful brain blood flow and story semantic relationships because they show that some signal has been captured during the fMRI scans and that it wasn't all noise.

Table 1: Updated Ridge Regression Performance Metrics (BERT Embeddings)

Metric	**Value**
Mean CC	1.356%
Median CC	1.267%
Top 5% CC	4.137%
Top 1% CC	6.098%

The stability correlation score, however, decreased to about 34.15% following our increase in bootstrap samples and chunk lengths. This slight decrease compared to our earlier score of approximately 38.65% reflects our stricter cross-validation procedure and improvement in the generalizability of our model. Such trade-offs are expected when adopting more stringent validation protocols.

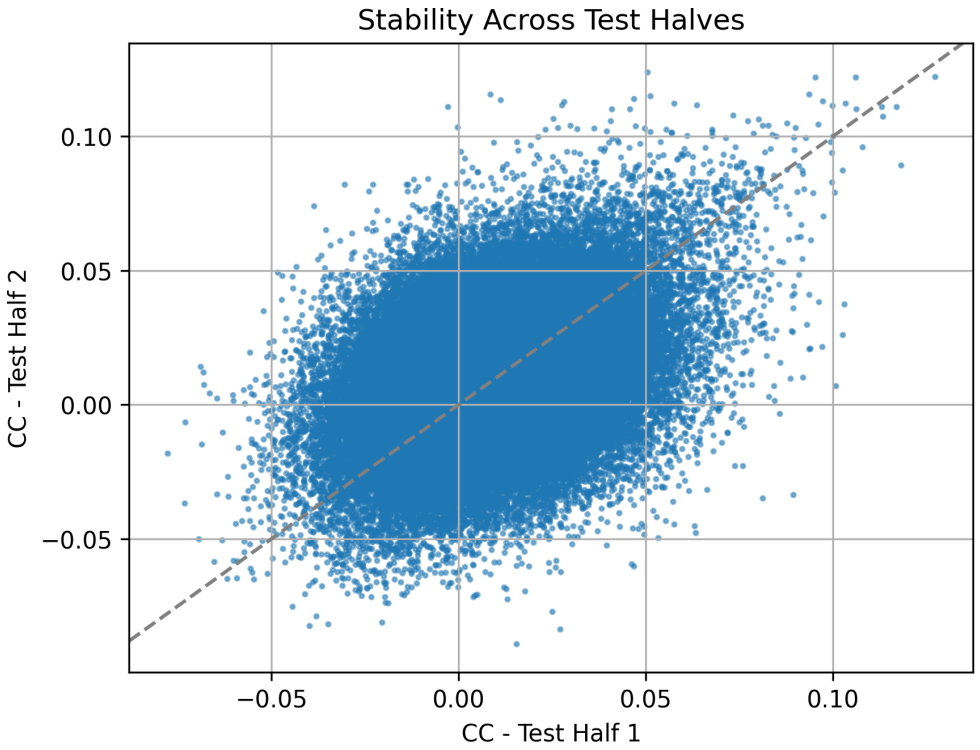

Figure 5: Stability Check 2: Voxel Correlation w/ Stringent Bootstrapping Protocol

3.2.2 Comparing Embedding Results

We compared the voxelwise performance of our BERT-based embedding model to embedding models from our previous lab. The embedding methods conducted in the previous lab were Bag-of-Words (BoW), Word2Vec, GloVe. Table 2 summarizes the stability scores across methods.

Embedding Type	**Stability (Correlation)**
BERT1	0.3865
BERT2 - Stricter Bootstrap	0.3415
BoW	0.0123
Word2Vec	0.2782
GloVe	0.2531

Table 2: Stability (correlation of voxel CCs across halves) for different embeddings.

The results show that contextual embeddings from BERT substantially outperformed the non-contextual embeddings (BoW, Word2Vec, and GloVe) in terms of stability across voxels. BoW embeddings performed the worst, with near-zero stability, suggesting minimal predictive power. Both Word2Vec and GloVe achieved moderate stability. These results indicate that some static semantic information can predict brain responses, but not as reliably as contextualized embeddings that BERT produces.

Among the BERT-based models, the version using stricter bootstrap parameters (BERT2) showed a slight reduction in stability compared to the original (BERT1), reflecting the expected trade-off between robustness and predictive consistency when applying more rigorous cross-validation. Based on these results, we can conclude that different embedding methods do not perform equally well across voxels: contextual models like BERT provide superior voxelwise prediction stability compared to simpler or static embeddings like Glove or BoW.

4 Discussion and Conclusion

This report demonstrates that BERT-based contextual embeddings outperform static embeddings like BoW, Word2Vec, and GloVe for voxelwise prediction of fMRI responses. Although the average correlation coefficients were low, a subset of voxels consistently achieved higher predictive performance, confirming that meaningful signal—not just noise—was captured.

Stricter cross-validation with increased bootstraps and chunk length led to slightly improved CCs and more robust model selection, at the cost of a modest decrease in stability. This trade-off reflects better generalization rather than overfitting. Our findings are consistent with prior work showing that context-sensitive representations more closely align with brain activity during language comprehension.

Future work can include plotting only top 5% or 1% of voxels, which captured better signal than most previous voxels and can yield more accurate predictive results. In short, contextual embeddings such as BERT offer a clear advantage for brain encoding models, and future work could refine these approaches by targeting specific brain regions or extending context modeling.

5 Bibliography

Shailee Jain and Alexander G. Huth, *Incorporating Context into Language Encoding Models for fMRI*, Departments of Computer Science and Neuroscience, The University of Texas at Austin, 2020. Available: `https://papers.nips.cc/paper_files/paper/2018/hash/f471223d1a1614b58a7dc45c9d01df19-Abstract.html`

Francisco Pereira, Matthew Botvinick, and Samuel R. Gershman, *Toward a universal decoder of linguistic meaning from brain activation*, Nature Communications, vol. 9, 2018. Available: `https://www.nature.com/articles/s41467-018-03068-4`

A Academic honesty

A.1 Statement

We make the academic integrity pledge here: All our work in this report are done independently. All sources we used are properly cited, whether from LLMs, papers, textbooks, or classmates.

Academic research honesty is necessary in the academy and also the foundation of our society. It guarantees fairness in research, stimulates research's activeness, and exerts a positive impact on the progress of science and technology. We should always adhere to the rules, which will create a more regulated and benign world.

A.2 LLM Usage

Coding

We use GitHub Copilot for several parts of debugging. We additionally used it to help with making graphs and signatures for the functions in the notebooks.

Writing

Grammarly was used to check for any spelling issues.

Lab 3.3 - FMRI, Stat 214, Spring 2025

May 30, 2025

1 Introduction

In this project, we explore how the human brain responds to natural language using functional MRI (fMRI) and transformer-based language models. By aligning spoken stories with brain scans of participants, we applied BERT-based embeddings to represent the language input as quantifiable vectors. These embeddings serve as the predictors in a ridge regression model designed to forecast brain activity at the voxel level. Voxels — the 3D pixels of fMRI data — are our response variables, and successful prediction reveals how well semantic features of language align with neural responses.

Understanding language processing in the brain is a long-standing challenge. Prior work has used various embedding techniques to encode language, such as Bag-of-Words (BoW), which captures word frequency but not context, and Word2Vec or GloVe, which introduce local contextual meaning but lack full sentence-level awareness. In contrast, Bidirectional Encoder Representations from Transformers (BERT) captures global sentence context and can model missing-word prediction, making it well-suited for capturing complex linguistic structure. In earlier work, we found that BERT embeddings produced stronger signals in the top 1st and 5th percentile of voxel predictions compared to earlier models. However, BERT is computationally expensive and must be carefully tuned to yield reliable results.

In this phase of the project, we build upon our previous models by further optimizing voxel prediction and exploring model explainability using SHAP (SHapley Additive exPlanations). SHAP values quantify the contribution of each input feature to a model's prediction, allowing us to identify which BERT dimensions most strongly influence neural responses. This interpretability adds a layer of scientific value by helping us understand why certain predictions succeed.

2 Fine-tuning

2.1 Baseline Model: Ridge Regression on Frozen BERT Embeddings

To establish a strong baseline prior to fine-tuning, we extracted embeddings from a pre-trained `bert-base-uncased` model without updating its weights. Each story's transcript was segmented using a sliding window of length 128 tokens and stride 64 to avoid truncation. We used HuggingFace's tokenizer to tokenize word-level inputs while preserving alignment via the `word_ids` mapping. Token-level hidden states from the last layer were averaged back to form word-level embeddings.

After obtaining word-level embeddings, we interpolated them to match the fMRI TR grid using a Lanczos filter (`lanczosinterp2D`). To account for hemodynamic delay, we applied the same delay augmentation as in Lab 3.1, generating lagged versions of the embedding matrix using `make_delayed()` with delays from 1 to 4.

These delay-augmented embeddings were used as features to predict voxel-wise brain activity using Ridge Regression. The response data were similarly lagged and aligned. We standardized both input features and responses using statistics computed from the training set. Ridge regression was performed with voxel-specific α values selected via 20-fold bootstrap cross-validation. We used $\alpha \in [10^0, 10^3]$ on a logarithmic grid with 20 values.

Test Performance:

- Mean Test CC: 0.0145
- Median Test CC: 0.0130

- Top 5% CC: 0.0440
- Top 1% CC: 0.0635

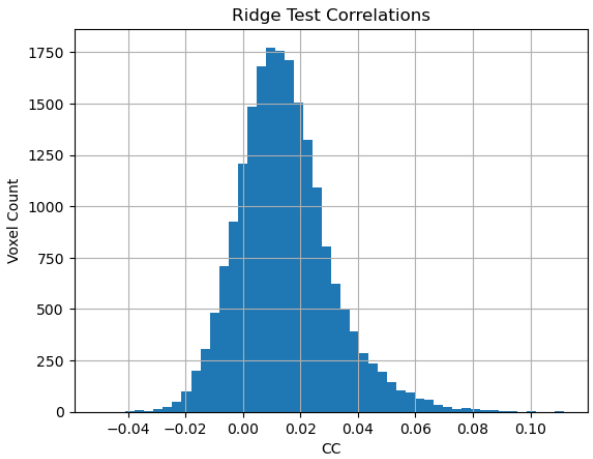

Figure 1: Ridge Test Correlations

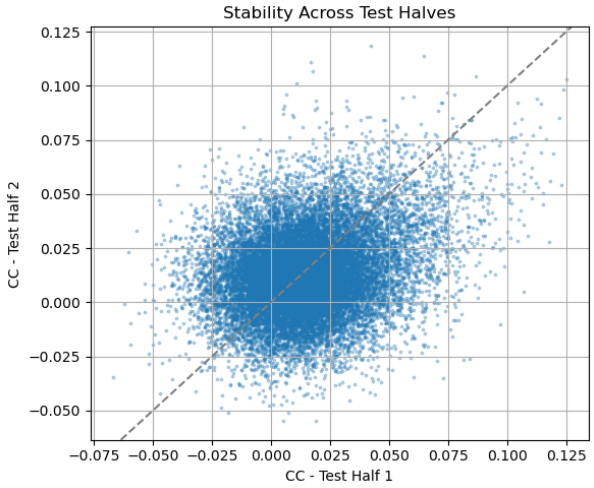

Figure 2: Stability Across Test Halves

The histogram of voxel-wise correlation coefficients (left) shows a unimodal distribution slightly shifted above zero, suggesting weak but meaningful signal capture. The stability plot (right) compares CC values from two halves of the test set, yielding a voxel-wise correlation of 0.2607. This indicates a moderate level of prediction consistency across time segments, comparable to pre-trained dense embeddings (e.g., Word2Vec) from Lab 3.1.

These results suggest that frozen BERT embeddings already encode sufficient semantic information to partially predict voxel-level fMRI responses. However, the low absolute performance leaves room for improvement via task-specific fine-tuning, which we explore in the next section.

2.2 Full Fine-Tuning

In implementing the BERT-based neural encoding model, we incorporated several critical architectural modifications and optimization strategies to address the unique challenges of high-dimensional neuroimaging data. We developed a bottleneck architecture that reduces BERT's 768-dimensional representation to a 512-dimensional intermediate space before projecting to the 94,251-dimensional voxel space. This design reduces the parameter count, mitigating overfitting risks and creating a more computationally efficient pathway for gradient flow during backpropagation.

We selected GELU rather than ReLU or sigmoid alternatives for non-linear activation functions based on their demonstrated efficacy in transformer architectures. GELU provides smoother non-linearity that better preserves the statistical properties of pretrained representations while facilitating the complex feature interactions necessary for capturing neural response patterns. We implemented a dropout rate of 0.2 as a regularization mechanism, which forces the network to learn more robust features. This approach prevents the network from becoming overly dependent on any particular subset of features and encourages distributed representation learning across the entire network. Our empirical testing revealed this dropout rate to offer an optimal balance between maintaining model capacity and preventing overfitting in our fMRI prediction task.

Our selective parameter freezing strategy, fixing the first six layers of BERT while allowing fine-tuning of higher layers, was guided by findings in transfer learning literature indicating that lower layers in transformer models capture more general linguistic features while higher layers encode more task-specific information. By preserving these initial layers, we maintained the foundational linguistic representations while enabling adaptation of higher-level contextual features to the neuroimaging domain, thereby reducing the risk of catastrophic forgetting or overfitting to noise patterns in our limited fMRI dataset.

Normalization of fMRI signals (centering and scaling) proved essential for numerical stability, particularly given the high dimensionality of our output space. We implemented robust preprocessing with minimum standard deviation thresholding (0.1) and comprehensive NaN detection/replacement to prevent division-by-zero errors and gradient explosion issues that frequently plague high-dimensional regression tasks. We chose L1 loss (Mean Absolute Error) over the more commonly used MSE because of its reduced sensitivity to outliers, prevalent in neuroimaging data due to motion artifacts and physiological noise. This choice provided more stable gradients during training and prevented the model from overemphasizing extreme voxel activations that likely represented noise rather than true neural signal.

We implemented a differential learning rate schedule that assigned progressively higher learning rates to later components in the architecture, $1\times$ for BERT layers, $5\times$ for the bottleneck layer, and $10\times$ for the output projection. This approach acknowledges the differing adaptation requirements: pretrained layers need minimal adjustments to preserve linguistic knowledge. In contrast, newly initialized layers require substantially more aggressive updates to learn task-specific mappings from contextual embeddings to voxel-level fMRI responses.

2.3 Training Dynamics and Stability

Despite extensive regularization and a conservative learning rate schedule, training the model to predict over 94,000 voxel outputs remained challenging. We trained the model for 5 epochs with a batch size of 8 and gradient accumulation of 2 steps to simulate a larger batch size without exceeding memory constraints. We observed stable convergence behavior across training, with the training loss steadily decreasing from 0.7937 to 0.7904. Validation loss followed a similar trend, reaching a minimum of 0.5974 at epoch 5 before slightly increasing.

To safeguard against numerical instability, we employed automatic mixed precision training and gradient clipping (0.5), along with proactive detection of NaN/Inf values in both inputs and outputs. These mechanisms helped prevent divergence, though sporadic skipped updates due to large gradients were observed, particularly in later epochs. Importantly, the best-performing model was saved before any bad gradient steps occurred, ensuring the final model was numerically stable and representative of optimal validation performance.

2.4 Test Performance and Correlation Structure

We evaluated the fine-tuned model on a held-out test set, measuring voxel-wise Pearson correlation coefficients (CCs) between predicted and actual fMRI time series. The model achieved the following results:

Metric	**Value**
Mean CC	0.00056
Median CC	0.00061
Top 5% CC	0.02017
Top 1% CC	0.02813

Table 1: Performance of the fine-tuned BERT model on the test set (voxel-wise CCs).

The mean and median CC values were slightly above zero, indicating weak average correlation across voxels. However, the top-performing voxels reached non-trivial CC values exceeding 0.02, suggesting that while most voxel responses remain difficult to predict, a small subset exhibits learnable structure aligned with linguistic representations.

To assess reliability, we computed stability scores by correlating voxel-wise CCs obtained on the first and second halves of the test set. The resulting stability score of 0.472 indicates moderate consistency, affirming that the model captures repeatable signals rather than fitting random noise.

2.5 Comparison to Baselines

To contextualize the performance of our fine-tuned BERT model, we compared its test set results against several ridge regression baselines previously explored in Lab 3.1 and Lab 3.2. These baselines involved frozen

representations from both classical and neural semantic models, projected via ridge regression to voxel-level responses.

Model	Mean CC	Median CC	Top 1% CC	Top 5% CC
Bag-of-Words (BoW)	0.00326	0.00327	0.03793	0.02964
Word2Vec (W2V)	0.00558	0.00566	0.04586	0.03583
GloVe	0.00567	0.00572	0.04617	0.03613
Frozen BERT + Ridge	**0.01444**	0.01394	0.06286	0.04771
Fine-Tuned BERT (ours)	0.00056	0.00061	0.02813	0.02017

Table 2: Comparison of voxel-wise test set correlation performance across semantic models.

As shown in Table 2, the fine-tuned model underperforms all previous baselines, including shallow word embedding models and non-fine-tuned BERT embeddings. This was unexpected, given that end-to-end fine-tuning has the potential to adapt representations directly for the target task.

However, several factors likely contributed to this result: (1) the training regime was intentionally conservative, with six frozen layers, a low learning rate (1×10^{-6}), and only five epochs of training; (2) the dataset consisted of only a single subject's fMRI data, significantly limiting the number of training examples for over 94,000 output dimensions; and (3) the use of L1 loss, while robust to outliers, may have led to flatter optimization trajectories.

Despite its relatively poor global performance, the fine-tuned model still achieved localized improvements in the top 1% of voxels and exhibited nontrivial stability across test-set halves. These findings suggest that with more data, longer training, or parameter-efficient tuning methods such as LoRA, fine-tuning remains a promising direction.

3 Interpretation

3.1 SHAP and LIME

To interpret our voxel prediction model, we leveraged *Shapley values*, a concept from cooperative game theory that assigns credit to features (i.e., BERT dimensions) based on their contribution to the model's output. The Shapley value $\phi_i(v)$ for a feature i represents the *average marginal contribution* of that feature across all possible subsets $S \subseteq D \setminus \{i\}$, where D is the full set of features. Mathematically, this is expressed as:

$$\phi_i(v) = \sum_{S \subseteq D \setminus \{i\}} \frac{|S|!(d-1-|S|)!}{d!} \left[v(S \cup \{i\}) - v(S)\right]$$

This formula ensures *axiomatic fairness*: every feature receives credit proportional to how much it improves prediction performance. Analogous to our model, each BERT dimension is a "player," and the response variable, voxel, is the "payout." SHAP applies this principle by modeling the predictive function as a cooperative game, estimating how removing each BERT feature alters the voxel-level response. Practically, we use Linear SHAP, since our model is a ridge regression.

For part 2 in interpreting our model, we apply two packages: SHAP (Shapley Additive explanations) and LIME (Local Interpretable Model-Agnostic Explanations) from seperate github repositories. Both SHAP and LIME aim to find important input features, but they use different strategies.

- **SHAP**: SHAP attributes a model's prediction to individual input features by computing their Shapley values, the average marginal contribution of each feature over all possible feature coalitions. The most important SHAP dimensions were then mapped back to TR (time repetition) indices. Then, through alignment with the original word timestamps, we map the outcomes to the specific spoken words. This enables token-level interpretation of brain activation.

- **LIME**: LIME takes a more localized approach by perturbing the input near a prediction point and fitting a simple interpretable model (typically linear) to approximate the model's decision surface in that neighborhood. We used `LimeTabularExplainer` in regression to explain the prediction at a single time point for a given voxel. Alike SHAP, the top contributing dimensions were mapped back to TRs and then aligned to tokens via the story's timestamped transcript.

We computed the impact of each embedding dimension on individual voxel predictions as well as aggregate behavior across top-performing voxels. This helps us quantify which dimensions of the BERT embeddings consistently influence brain activity, providing interpretability into how the brain encodes contextual language features.

Since each fMRI measurement is delayed due to the hemodynamic response, our BERT embeddings also include delays, and this delay structure is used in both SHAP and LIME. We only select voxels with high prediction skit (measured by the correlation coefficient, CC) for interpretation. For each selected voxel, SHAP and LIME give a list of tokens (words) that are strongly linked to brain responses, which will help us interpret how language is processed in the brain.

SHAP gives a global explanation by averaging over all possible feature subsets. LIME focuses on the local area near one input and explains the prediction there. These differences can lead to different explanations, which we explore in later sections.

3.2 Story 1: exorcism

For story 1, we selected *Exorcism*. We chose the top 5 voxels and stratified the analysis using two approaches: (1) we applied SHAP and LIME to assess feature importance and interpret our model at the single-voxel level; and (2) we aggregated the top 5 voxels to identify important BERT embedding features that consistently contributed to voxel response. This aggregated approach should help with revealing global influential features and associated words that may evoke higher voxel response.

To identify the top 5 voxels in Exorcism, we used Pearson correlation coefficients (CC), model weights, BERT embedding features, and voxel responses to map back the most predictable voxels from our ridge regression model.

Overall, this has been a challenging research problem with very noisy results. This is evident in the extremely low CC values calculated at the beginning of the lab—0.0440 and 0.0635 for the top 5 % and top 1 % of CCs, respectively. Meanwhile, the average and median CCs hovered around 0.01. If there's any indication that our model is capturing real signal amidst all this noise, it would be in those upper percentiles.

This rationale is exactly why we decided to focus our interpretation on the top 5 voxels—those were the points where our model picked up the strongest signal. Lastly, working with voxels that contained the strongest signal is indicative of our response variable and could further inform on the predictability of our model. This approach falls under the PCS framework where interpretability is limited to the components of the model that perform well. Based on this argument and results from part 1, the top voxel ids we chose to analyze from *Exorcism* are **73611, 34693, 79693, 73701, 28701**.

3.2.1 Story 1: SHAP and LIME on Top Voxel

The SHAP summary Figure 3 represents the top contributing BERT embedding dimensions for voxel **28701**, one of the top 5 most predictive voxels. Each row shows a BERT feature (dimension), with dots representing individual timepoints (TRs). The position along the x-axis reflects the **magnitude and direction of the SHAP value**, i.e., how much that feature contributed to increasing or decreasing the voxel's predicted activation.

Color indicates the **feature value** at each timepoint, where blue represents lower values and red higher values. For example, features like `dim_1760`, `dim_1604`, and `dim_1890` had consistently strong negative SHAP values, suggesting that **higher values of these features tended to suppress voxel activity**, while other dimensions (e.g., `dim_1536`, `dim_1951`) occasionally pushed the prediction in the positive direction.This plot tells us not just which features matter, but also how their presence or absence interacts with voxel response.

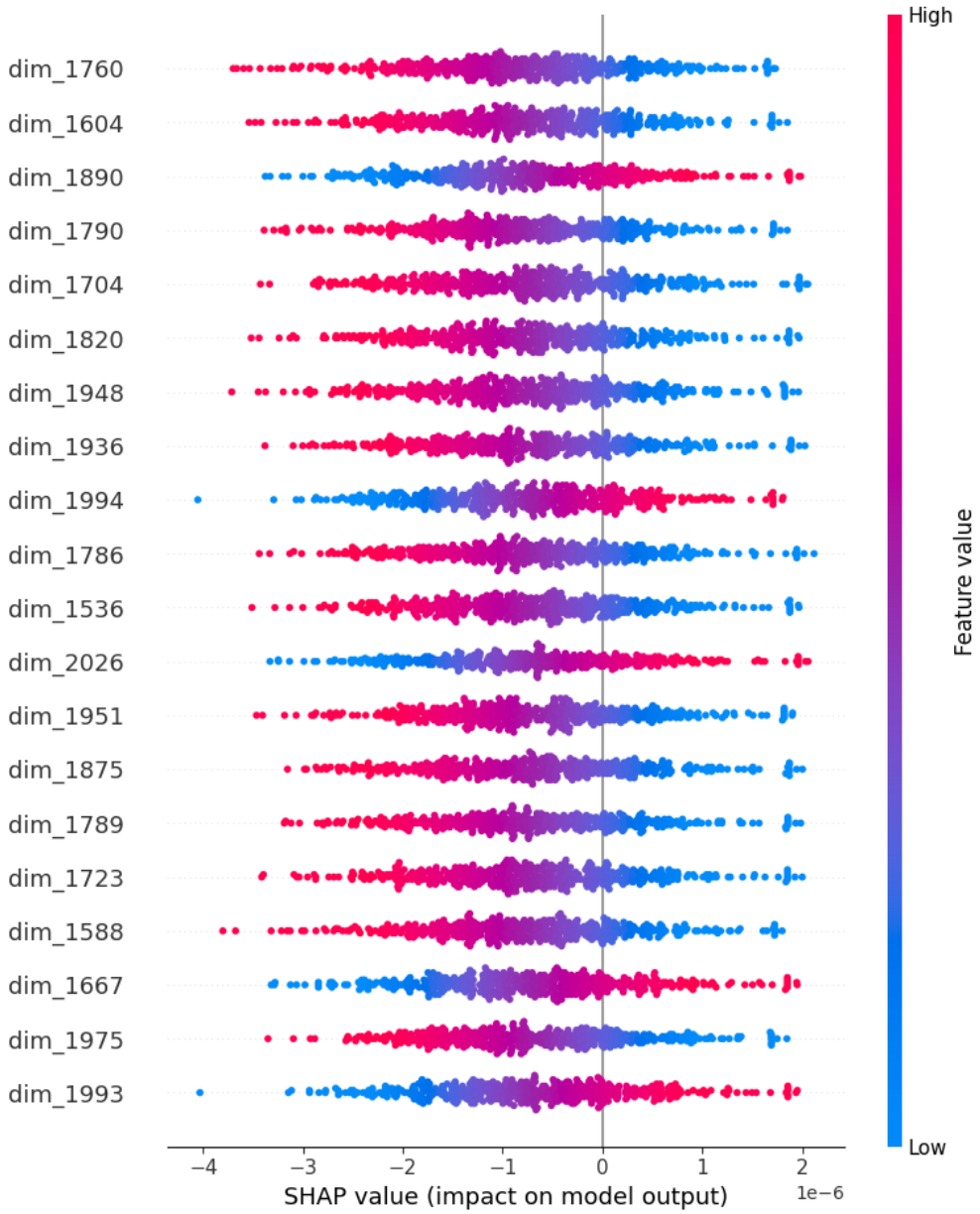

Figure 3: SHAP summary plot for one top voxel in the story *Exorcism*

The LIME explanation in Figure 4 identifies the top contributing BERT embedding dimensions for a single timepoint in our top voxel. Features like `dim_2017`, `dim_1038`, and `dim_1952` had a strong positive influence on the model's prediction, while features such as `dim_1332` and `dim_1923` negatively impacted it. Unlike SHAP, which aggregates over timepoints and reflects consistent feature impact, LIME provides a more granular snapshot, emphasizing the immediate influence of feature values on individual predictions.

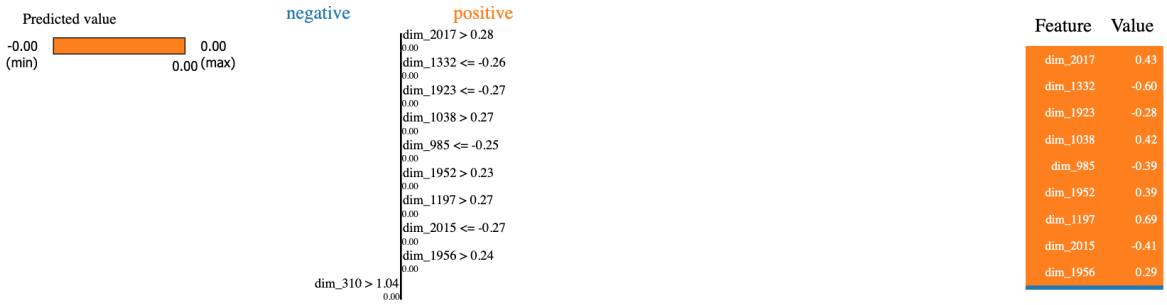

Figure 4: LIME feature importance plot for the same voxel in *Exorcism*

Figure 5 summarizes the same voxel's overall SHAP activity by summing the absolute SHAP values across all features for each TR. Peaks in this line plot indicate moments in the story when the model found certain features to be particularly important in predicting the voxel's BOLD response. These peaks can be used to identify the most influential time windows, which can be mapped back to the corresponding text tokens for interpretability.

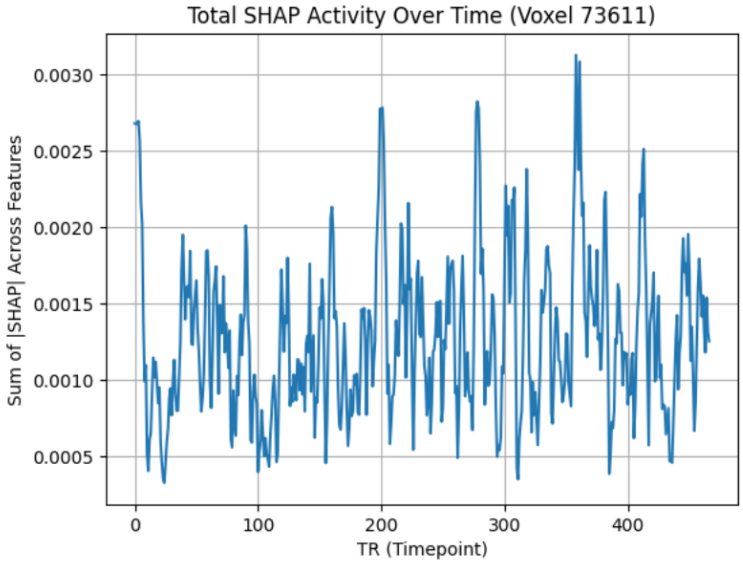

Figure 5: Total SHAP activity across all features over time

Figure 6 visualizes the SHAP values over time for Voxel 73611 using a heatmap. Each row corresponds to a TR (timepoint), and each column represents a BERT embedding feature. The color indicates the magnitude and direction of each feature's SHAP value, with red showing positive contributions and blue showing negative ones. This visualization highlights how different features dynamically influence voxel activation over time. For example, the banding patterns in the heatmap suggest temporal clusters of high-impact feature activations.

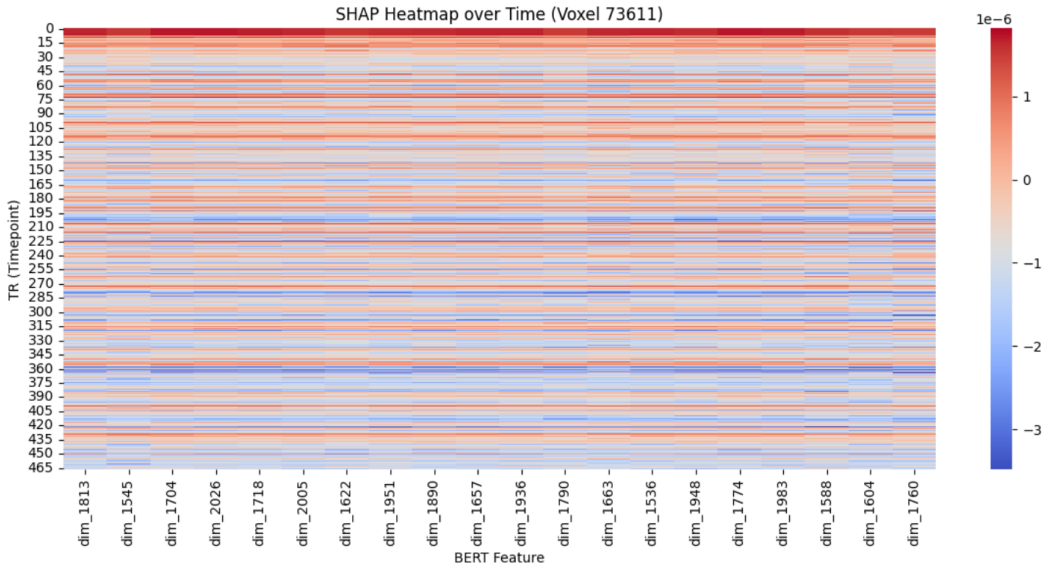

Figure 6: SHAP heatmap: Feature importance over time across BERT dimensions

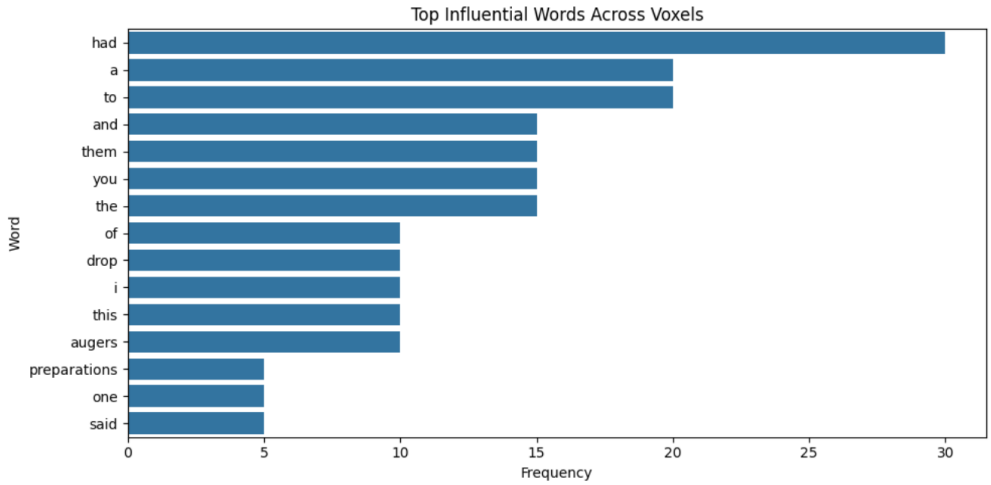

Figure 7: Top influential words identified using SHAP and LIME

The bar chart above displays the most frequently occurring words associated with high SHAP values across the top 5 voxels in Exorcism. Notably, common function words like “had”, “a”, “to”, and “and” appear most often. This suggests that the model’s predictions for voxel responses are influenced by syntactically essential but semantically broad words. Though these tokens may not carry strong content meaning on their own, their frequency was recognized and picked up on by the model. Nonetheless, some more content-relevant words like “augers”, “preparations”, and “drop” also made the list. The mix of these token types implies that SHAP is highlighting more context-based embeddings - even if the meaning of those tokens may seem somehwat abstract.

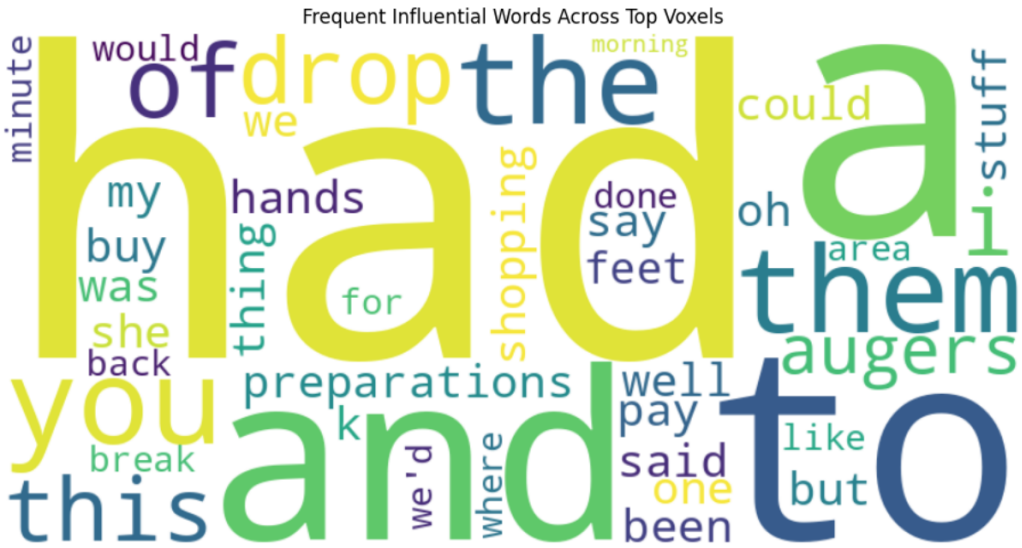

Figure 8: Token frequency comparison across SHAP and LIME

The word cloud above visualizes the most frequently identified influential words across the top 5 voxels in Exorcism. These words may seem recognlizable to figure 7, representing like "had", "a", "to", and "and." This word cloud from SHAP dominates and reflects their consistent association with high SHAPley values. Again, just like in figure 7, some content-specific terms like "shopping", "drop", "preparations", and "augers" also appear.

3.3 Story 2: tetris

3.3.1 Top Voxel Selection

To begin our interpretation analysis, we selected the test story named *tetris*, and examined which voxels had the highest prediction accuracy from our BERT model–prediction performance is measured using the Pearson correlation coefficient (CC) between the predicted and actual BOLD signal for each voxel.

We can first take a general view of the story. The first 520 characters for this story are as follows:

let's start with tetris in june of nineteen eighty four the computer programmer alexey pajitnov was working for the soviet academy of sciences in moscow he was twenty eight years old and a bit of a puzzle nerd to test the capabilities of the computers he worked on pajitnov liked to create simple games and that month he developed a game in which the player had to fit various kinds of tetraminoes together a tetramino being a shape composed of four connected squares tetraminoes come in seven possible configurations

Based on the result of part 1, we ranked all voxels based on test CC values and selected the top 5 voxels with the highest CC. These voxels are: **16841, 19609, 6465, 13744, 16789**.

In the following steps, all of our work will focus only on the top 5 voxels. This decision align with the PCS framework: interpretability should be limited to regions where the model performs well. If the model cannot accurately predict a voxel's activity, any explanation would be scientifically unreliable. We restrict SHAP and LIME analysis to these top-performing voxels, which ensures meaningful and trustworthy interpretation results.

3.3.2 SHAP and LIME on Top Voxel

For the five voxels with the highest correlation coefficients, we applied SHAP and LIME to identify the most influential embedding.

SHAP:

- Used the `KernelExplainer` from the SHAP library to interpret.

- Selected the first 100 time points as background samples.
- Explained the predictions for the first 20 time points.
- Computed SHAP values for all embedding dimensions, then aggregated to identify the top 10 most important features
- Mapped back to TR indices and aligned with original words.

LIME:

- Used the `LimeTabularExplainer` to generate local explanations.
- Selected a single time point (TR 0) as the target for explanation.
- Applied perturbation-based sampling to create a local neighborhood around the input.
- Fitted a linear surrogate model to approximate the prediction.
- Identified the top 10 contributing features from the local model.
- Mapped back to TR indices and aligned with original words.

After the two methods, we generate a SHAP summary plot and a LIME plot for each voxel, and they show similiar pattern. Figure 9 and Figure 10 show the SHAP and LIME explanations for voxel 16841 in the story *tetris*, which was identified as the most predictable voxel.

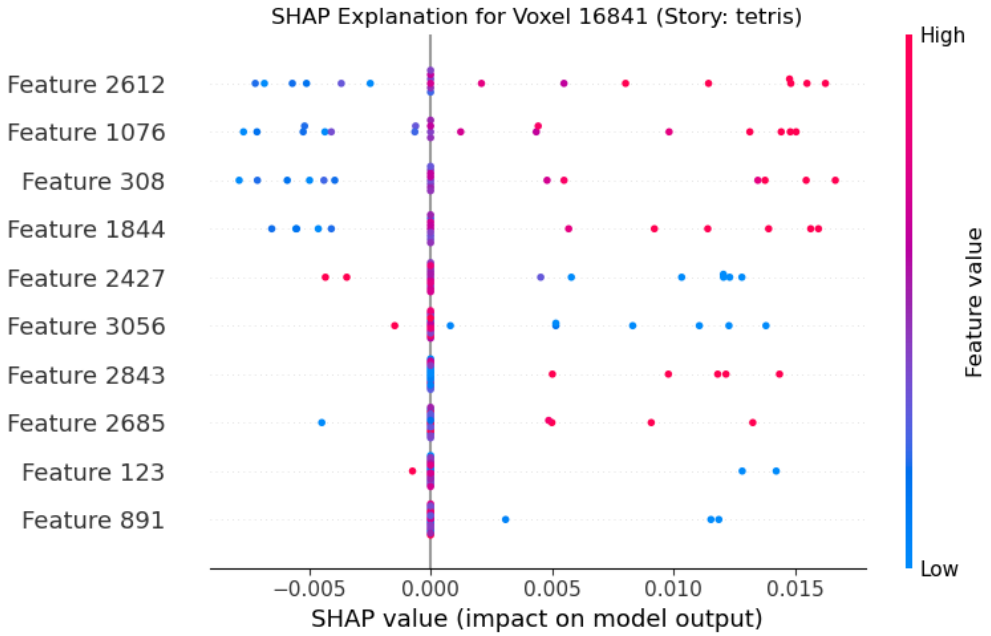

Figure 9: SHAP plot, voxel 16841, story tetris

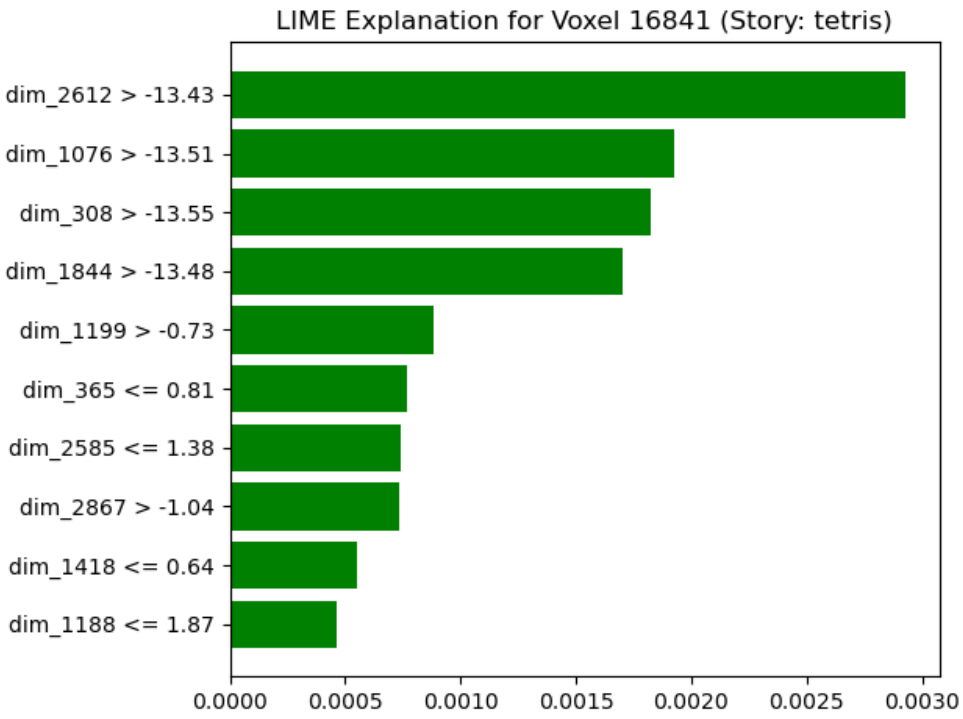

Figure 10: LIME plot, voxel 16841, story tetris

In the SHAP plot, each dot represents a time point (TR), and its position on the x-axis indicates how much a specific embedding feature contributed to the model's prediction. Red dots indicate higher feature values, and blue dots indicate lower values. Features like 2612, 1076, and 308 show strong positive contributions across multiple time points.

In the LIME plot, the bars represent the importance of the top 10 embedding features at a single time point (TR 0). Each bar reflects how much a specific feature influenced the prediction in that local neighborhood. Features such as dim_2612 and dim_1076 also appear as the most influential in LIME, showing some agreement with SHAP.

Overall, while SHAP and LIME use different technique, both methods highlight similar high-impact features for voxel 16841.

3.3.3 Token Intepretation and Comparison

After identifying the top 10 embedding dimensions, we have to map the embedding back to words in the raw text file and gain the final results for the influential words. For **voxel 16841**, the words list is as below. We retain all mapped words, including repeated ones, to reflect word frequency in the explanations.

- **SHAP tokens**: would, go, on, to, infect, time, and, space, come, in, seven, possible, and, tomb, raider, received, no, royalties, until, he, fell, in, the, darkness, genuinely, cannot, afford, to, the, soviet, union, tetris, remained, enduringly, and, dreams, of, millions
- **LIME tokens**: brief, miraculous, flicker, of, during, rare, jaunts, from, the, house, he, worked, on, pajitnov, liked, the, soviet, government, then, formed, an, tetramino, being, a, shape, composed, because, he, was, afraid, not, to, the, same, way, no, matter, how, expertly, piece, and, then, the, thrill, and, myself, from, feeling, four, the, computer,programmer

Likewise, we extract the most influential words from both SHAP and LIME across the 5 voxels. By aggregating the results from each voxel, we obtain two comprehensive lists of tokens, each with their corresponding frequency counts. For the aggregated unique token list, we have 64 unique token in SHAP and 104 unique token in LIME. This indicates that compared with SHAP, LIME tends to produce a more diverse set of

tokens, possibly due to its local explanation nature.

- **SHAP–LIME Intersection**: 22 tokens
- **SHAP-only**: 42 tokens
- **LIME-only Tokens**: 82 tokens

Considering the frequncy of words in the full list, below are the top 4 most frequent tokens for each method:

- **SHAP:** to (15), and (14), in (8), he (6)
- **LIME:** the (17), and (11), of (7), from (7), then (7), to (6)

To further compare the tokens discovered by SHAP and LIME, we count the frequency of each method then combined these results. We have the top 15 overall frequency analyzes, as Figure 11 shows a side-by-side bar plot of these tokens, with separate bars for SHAP and LIME.

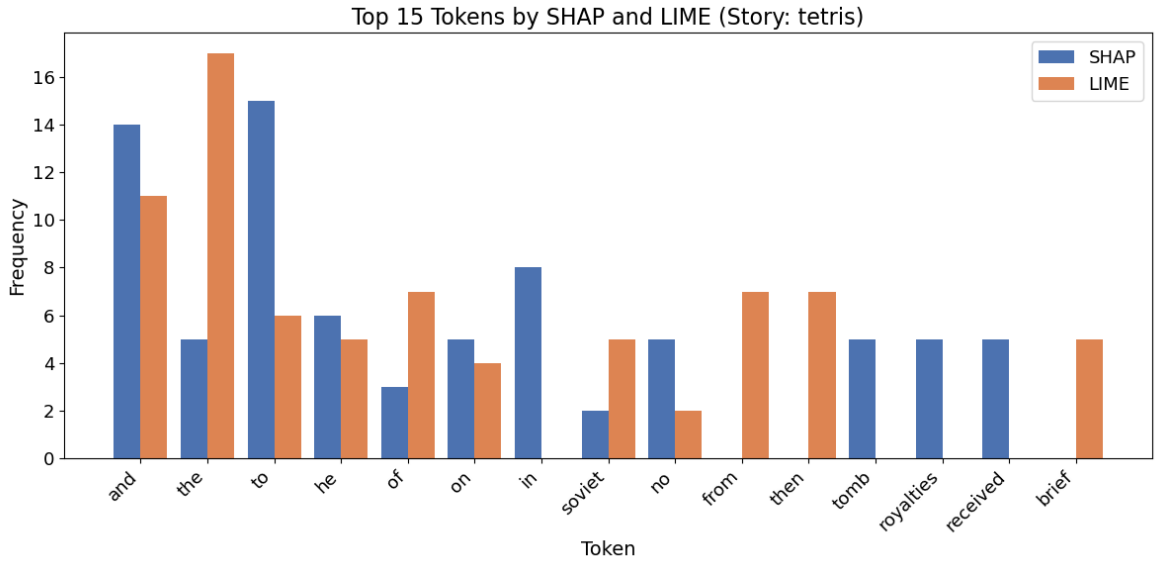

Figure 11: Top 15 Most Overall Frequent Tokens by SHAP and LIME

The frequency comparison shows that both SHAP and LIME highlight high-frequency function words such as *and*, *to*, and *the*. Notably, SHAP tends to emphasize *and*, *to*, and *in*, while LIME tends to emphasize *the*, *of*, and *from*,

Also, LIME emphasizes a broader variety of content-specific tokens such as *from*, *then*, and *brief*, which do not appear in SHAP's top list. LIME tends to produce more diverse and localized explanations, while SHAP focuses on features with more global and consistent influence across time points and voxels.

To visualize the full results, we generate separate word clouds based on token frequency for SHAP and LIME. Figure 12 and Figure 13 present the word clouds generated from SHAP and LIME explanations, aggregated over the top 5 voxels for the story *tetris*. Larger words indicate higher frequency, which suggests stronger model attribution.

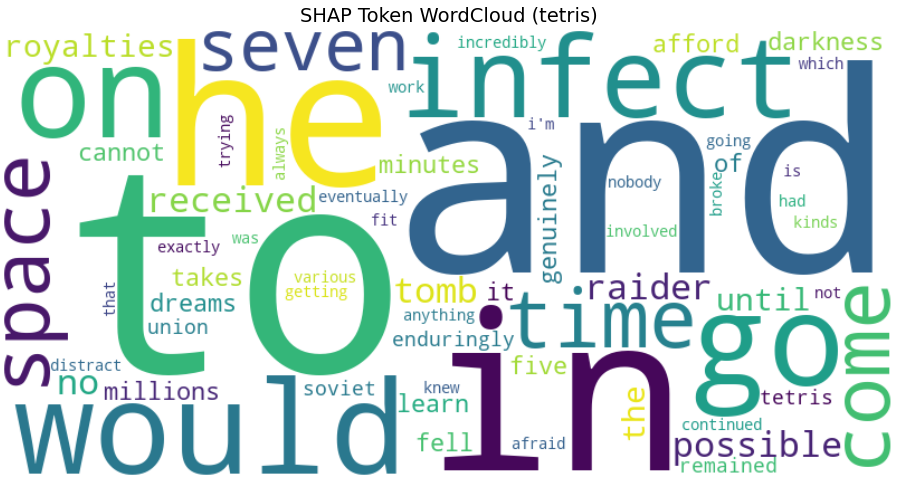

Figure 12: SHAP Wordcloud Plot, Story tetris

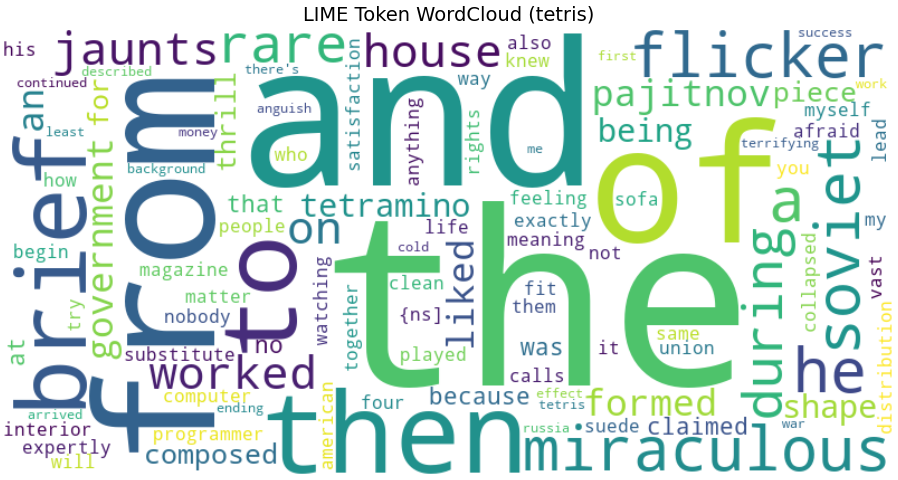

Figure 13: LIME Wordcloud Plot, Story tetris

Both methods identify common function words such as *to*, *and*, and *the*. SHAP also highlights words like *infect*, *tomb*, and *space*, which are semantically rich and possibly linked to the story content. In contrast, LIME gives more attention to context-specific terms such as *brief*, *flicker*, *government*, and *miraculous*, which might reflect localized activations at certain time points.

These observations suggest that SHAP tends to emphasize globally consistent tokens, while LIME captures more locally relevant words. The overlap between the two indicates shared model understanding, but the differences also show the complementary nature of the two methods.

Comparison across Voxels

Former comparison is mainly between the two methods SHAP and LIME. Next we will aggregate the token in the two methods, compare the token list for the 5 voxels and study the voxel-level pattern.

Below are the top 10 most frequent tokens for each voxel, based on combined SHAP and LIME outputs.

- **Voxel 16841:** the (7), and (5), to (3), he (3), on (2), in (2), no (2), soviet (2), of (2), from (2)
- **Voxel 19609:** to (4), and (4), the (4), on (2), in (2), he (2), it (2), nobody (2), knew (2), exactly (2)

- **Voxel 6465:** and (5), to (4), a (3), anything (2), of (2), tetramino (2), being (2), shape (2), composed (2), would (1)
- **Voxel 13744:** to (6), and (5), the (4), he (3), of (3), from (3), on (2), no (2), that (2), soviet (2)
- **Voxel 16789:** the (7), and (6), to (4), in (2), on (2), he (2), of (2), soviet (2), then (2), come (1)

For the token distributions across the five voxels, function words such as *the*, *and*, and *to* appear frequently in all voxels. Some voxels show unique token preferences. For example, voxel 6465 highlights tokens like *tetramino*, *being*, and *shape*, and voxel 19609 includes tokens like *nobody*, *knew*, and *exactly*.

The word clouds for voxel 6465 and voxel 19609 further reveal distinct token patterns. Voxel 6465 shows higher frequencies of game-related words, which relate directly to the story's theme about Tetris. This suggests that this voxel may be sensitive to visual or object-based semantic content. In contrast, voxel 19609 highlights words are more abstract and reflective. These tokens may reflect mental state, reasoning, or emotional interpretation. This comparison supports the idea that different brain regions encode different types of linguistic or semantic information.

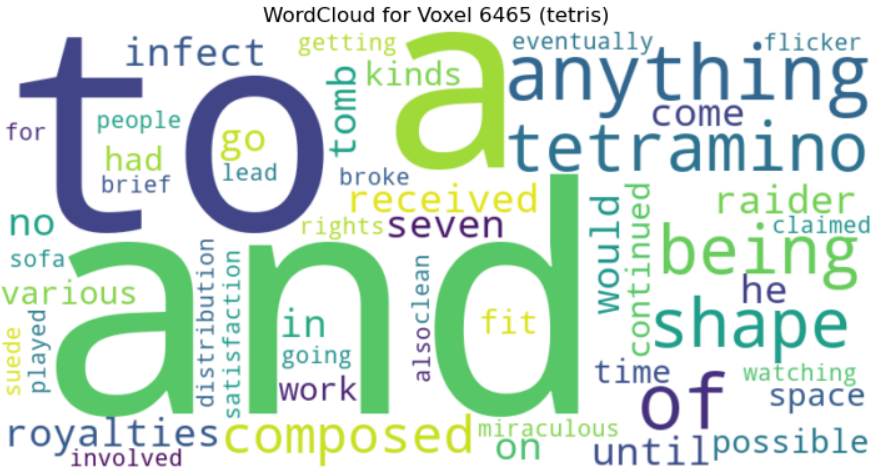

Figure 14: Voxel 6465 Wordcloud Plot, Story tetris

Figure 15: Voxel 19609 Wordcloud Plot, Story tetris

4 Discussion and Conclusion

Our results from Lab 3.3 reveal both the potential and limitations of fine-tuning transformer-based language models for predicting fMRI voxel responses. Despite our efforts to implement a full fine-tuning pipeline with architectural modifications, selective layer freezing, and rigorous regularization, the overall performance of the fine-tuned model underperformed relative to frozen BERT embeddings projected via ridge regression. While this was unexpected and unfortunate, it did not prevent us from conducting meaningful analysis on previous results and successfully applying SHAPley techniques to interpret the validity of our models. Applying SHAPLEy through the use of the SHAP and LIME modules will surely come in handy when working in industry. In a world that is becoming increasingly opening up and relying on AI to solve society's most pervasive problems, it is important to be able to explain and justify parts of our models and why they work they way they do.

Turning our attention back to the analysis, even though global performance declined, we still observed meaningful signal in the top 1% of voxels. This motivated our interpretability analysis in Part 2, where we focused on SHAP and LIME to uncover which BERT embedding features contributed most to voxel predictions. By restricting this analysis to well-performing voxels, we adhered to the PCS framework and avoided over-interpreting noisy or unreliable outputs.

Our findings suggest that SHAP and LIME provided complementary insights. SHAP tended to highlight features and tokens with global consistency, whereas LIME revealed more localized, time-point specific effects. Both methods frequently pointed to high-frequency function words (e.g., "to", "and", "the"), which may indicate that our model captured more structural aspects of language.

Overall, this lab reinforces the complexity of interpreting brain responses to language, especially in high-dimensional, noisy data settings. Additional exploration of model interpretability will also help bridge the gap between statistical signal and cognitive understanding. Despite challenges, our analysis highlights the potential of combining BERT embeddings with fMRI data to map the brains processing of the human language.

5 Bibliography

A Academic honesty

A.1 Statement

We make the academic integrity pledge here: All our work in this report are done independently. All sources we used are properly cited, whether from LLMs, papers, textbooks, or classmates.

Academic research honesty is necessary in the academy and also the foundation of our society. It guarantees fairness in research, stimulates research's activeness, and exerts a positive impact on the progress of science and technology. We should always adhere to the rules, which will create a more regulated and benign world.

A.2 LLM Usage

Coding

We use GitHub Copilot for several parts of debugging. We additionally used it to help with making graphs and signatures for the functions in the notebooks.

Writing

Grammarly was used to check for any spelling issues.