Blood-Oxygen-Level-Dependent Prediction with FMRI Data, Spring 2025

Calvin Carter, Yuki Asahara, Aditya Vunnum

May 15, 2025

1 Introduction

Understanding how the brain processes natural language is a fundamental question in computational linguistics. Predicting brain activity in response to language stimuli is informative to our understanding of language representation and processing in the cortex, but is also difficult to accurately capture due to the complexities of language.

Language encoding models help explain language processing in the human brain by learning functions that predict brain responses from the language stimuli that elicited them. However, many word embedding-based approaches treat stimulus words independently and ignore the context of language that are essential to predicting brain response. In this report, we investigate the relationship between natural language and brain activity by predicting blood-oxygen-level-dependent (BOLD) responses measured through functional Magnetic Resonance Imaging (fMRI) as subjects listen to spoken stories. We aim to develop and evaluate encoding models that map textual features of the stories onto voxel-wise brain responses. This work builds upon research conducted by Huth et al. and Jain & Huth, which demonstrate that contextual word representations perform better in prediction tasks than static word embeddings [Jain and Huth, 2018].

This work is important because improving the accuracy of encoding models allows us to better localize language-related brain activity. Second, comparing different embedding strategies provides insight into how linguistic information is organized and processed in the human brain, such as the roles of context, temporal structure, and hierarchical representations. Improved models mapping brain response to language could lead to applications in brain-computer interfaces, and language disorder diagnostics in the future.

2 EDA

The dataset used in this report consists of 108 time-aligned story transcripts picked by the Huth Lab at UT Austin and fMRI recordings collected from two subjects as they listen to spoken narratives. Each story is a sequence of words associated with a timestamp for when it was spoken to the subject. Figure 1 shows a histogram of the number of words and the length in seconds for each story. For each story the subject listens to, we constructed a response matrix with dimensions T x V, where T is the number of fMRI time points, and V is the number of voxels in the subject's brain scan.

A voxel, or volumetric pixel, is a 3D unit of brain tissue. Each voxel captures the blood-oxygen-level dependent (BOLD) signal within that section of the brain. This signal is a measure of neural activity, and changes in blood oxygenation can be indicative that a particular section of the brain is involved in language comprehension.

The datasets were divided into a training and test set using a random split of the stories. So, the 20% of the stories were withheld for the test set while remaining 80% of the stories were used for cross-validation and training the final model. The story lengths were examined to make sure that the training and testing sets end up with an appropriate amount of data using Figure 1.

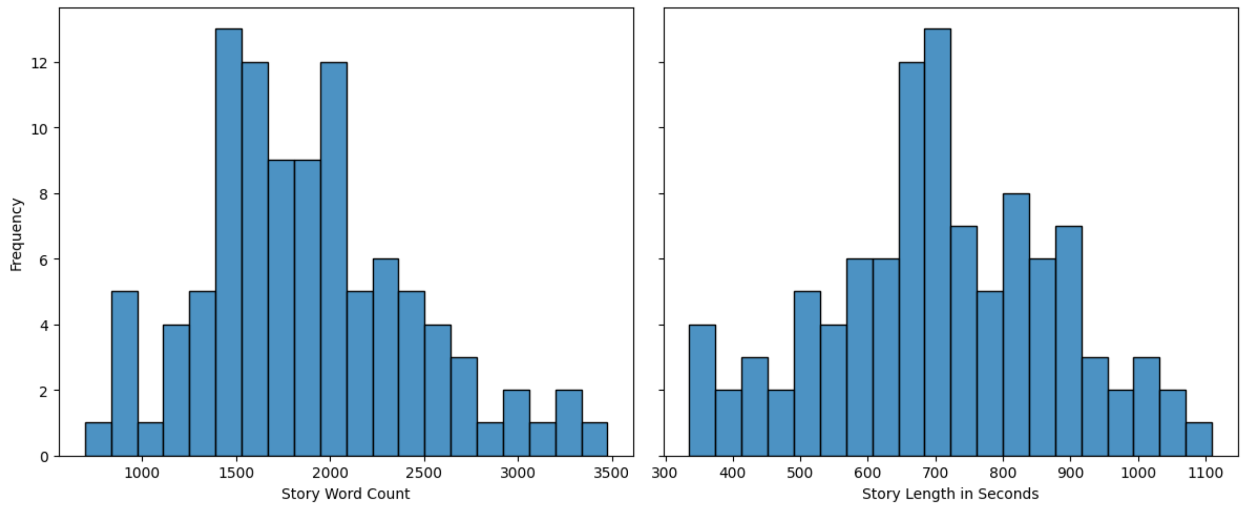

Figure 1: Distribution of story lengths by word count (left) and time in seconds (right)

3 Embeddings

3.1 One-Hot Encoding, Word2Vec, GloVe

In the first part of this lab, we explore embedding techniques such as bag-of-words, and employ pretrained models including Word2Vec and GloVe to transform the text into numerical features. These embeddings are the input features to the encoding model and are aligned in time with the fMRI responses for each voxel.

To generate one-hot-encoded embeddings, we construct a vocabulary built on the training stories. We then use this vocabulary to construct a one-hot-encoded ”response” matrix for every story to create the embeddings. To do this, we have each row represent a word from the current story and every column represent the words from the vocabulary, and set the proper row-column to 1 if the word was in the vocabulary (resulting in each row containing a sum of at most 1, but sometimes 0 if the word didn't exist in the vocabulary). Like Word2Vec and GloVe, we had to down-sample these response matrices using Lanczos resampling so that the fine-grained timing of language processing and the coarse-grained fMRI measurements would align. This process ensures that the encoding models accurately predict brain responses without misalignment.

To construct Word2Vec and GloVe embeddings, we utilized publicly available pretrained models. The model ”word2vec-google-news-300” which is trained using part of the Google News dataset and contains 300-dimensional vectors for 3 million words, was used to get the Word2Vec embeddings for the story words. To get the GloVe embeddings for the words in the stories, the model ”glove-wiki-gigaword-300”, which is trained using Wikipedia 2014 and Gigaword 5 and contains 300-dimensional vectors for 400,000 words, was used. If the word a word from a story is not contained in the model, the embedding for that word was set to a vector of all zeros.

Like mentioned before, a key challenge in this process is the temporal mismatch between spoken language and the slower temporal resolution of fMRI. When listening to a story, there is a constant influx of words, but the fMRI data does not match this temporal frequency. Therefore, we incorporate techniques like down sampling and temporal lagging to synchronize the two signals.

3.2 BERT-style Masked Language Model

Another method we used to generate embeddings was to train a BERT-style masked language model (MLM) encoder using the text from the stories provided. The motivation behind using a masked language model was to generate contextualized embeddings that better capture intra- and inter-word dependencies, compared to static embeddings like Word2Vec. To implement this, we used transformer blocks to process the inputs. This included using multi-head self-attention to allow a larger amount of different input sequences to be processed at once, giving us insight on inner-token information. We then processed this in combination with a feed forward neural network to build the forward pass of our transformer blocks so that we can also capture information from individual tokens.

Hidden Size	**# Heads**	**# Layers**	**Intermediate Size**
128	2	2	256
256	4	4	512
512	8	8	1024

Table 1: Hyperparameters tested for encoder optimization

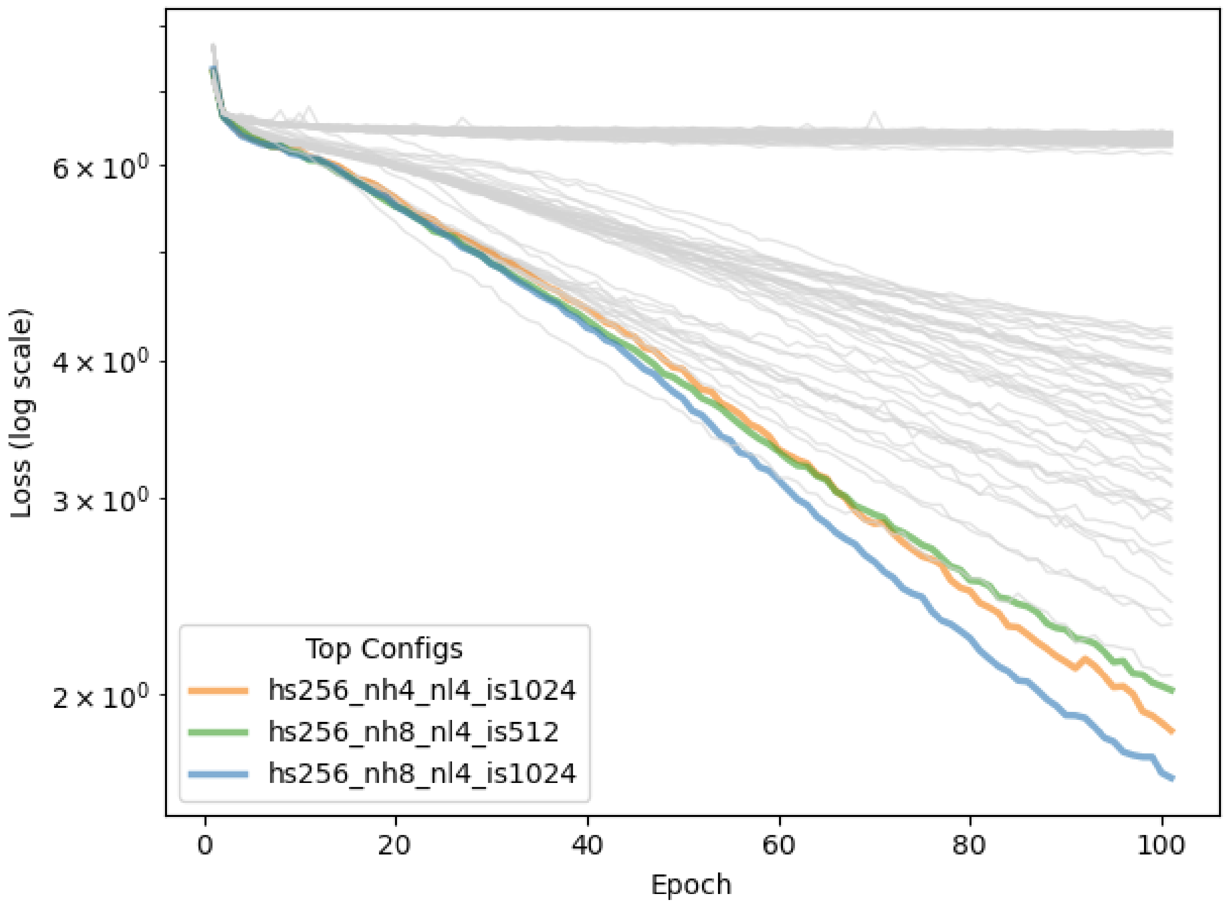

Figure 2: Cross Entropy Loss at each epoch for the 81 different encoder models trained.

Lastly, we passed the embedded data into n layers of these transformer blocks to get the final resulting embeddings. During pretraining, a fixed percentage of input tokens were randomly masked and the model was trained to predict the original tokens, forcing the model to embed both local and long-range context. We then adjusted the hyperparameters for the hidden layer size, the number of heads, the number of layers, and the intermediate size shown in Figure 2. More specifically, Table 1 shows the hyperparameters that were tested, which resulted in a total of 81 different encoders trained. Figure 2 shows the cross-entropy loss across training epochs for all models. We selected the three encoder models with the lowest validation loss, listed in Table 2, assuming that stronger token reconstruction would lead to richer embeddings for downstream fMRI prediction tasks. As with other embedding methods, the final encoder-derived embeddings were downsampled and lagged to match the temporal characteristics of the fMRI signal.

Name	**Loss**	**Hidden Size**	**# Heads**	**# Layers**	**Intermediate Size**
Encoder 1	1.68	256	8	4	1024
Encoder 2	1.85	256	4	4	1024
Encoder 3	2.02	256	8	4	512

Table 2: Cross entropy loss of the different encoders and their hyperparameters.

The three encoders with the lowest loss are listed in Table 2. These three encoders were chosen to generate the embeddings that will be used in the modeling section. These embeddings were then down sampled and lagging was added to match the signals between fMRI data and the word embeddings, as done with the other embeddings.

3.3 Pretrained and Fine-tuned BERT-style Masked Language Model

We also obtained embeddings using a pretrained base BERT model from Hugging Face. This model is trained on a large corpus of English data without any human labeling. More specifically, the model is trained for masked language modeling and next sentence prediction. Since the input of the model is limited to 512 tokens, the stories were chunked to pieces with 512 tokens each. Then the embeddings were obtained for each token. Lastly, some words were split into multiple tokens, so the mean of the embeddings for the tokens belonging to the same word were calculated to obtain the embeddings the words.

Embeddings for a fine-tuned version of this model were also obtained. The BERT model was fine tuned using the text from the stories provided and a technique called Low-Rank Adaptation of Large Language Models (LoRA). Using the stories allows the model to be fine tuned to our specific problem. LoRA reduces the number of trainable parameters, making the fine-tuning process fast and efficient. For our model, the query and value modules were fine tuned for a feature extraction task, since the features are what will be used for modeling. For the training, the stories were broken into lists of 512 tokens, with an overlap of 256 tokens between the chunks originating from the same story, to preserve the context of the tokens as much as possible. The losses from the training loop are shown in Figure 3. Since the loss was bottoming out, the training lasted for 10 epochs.

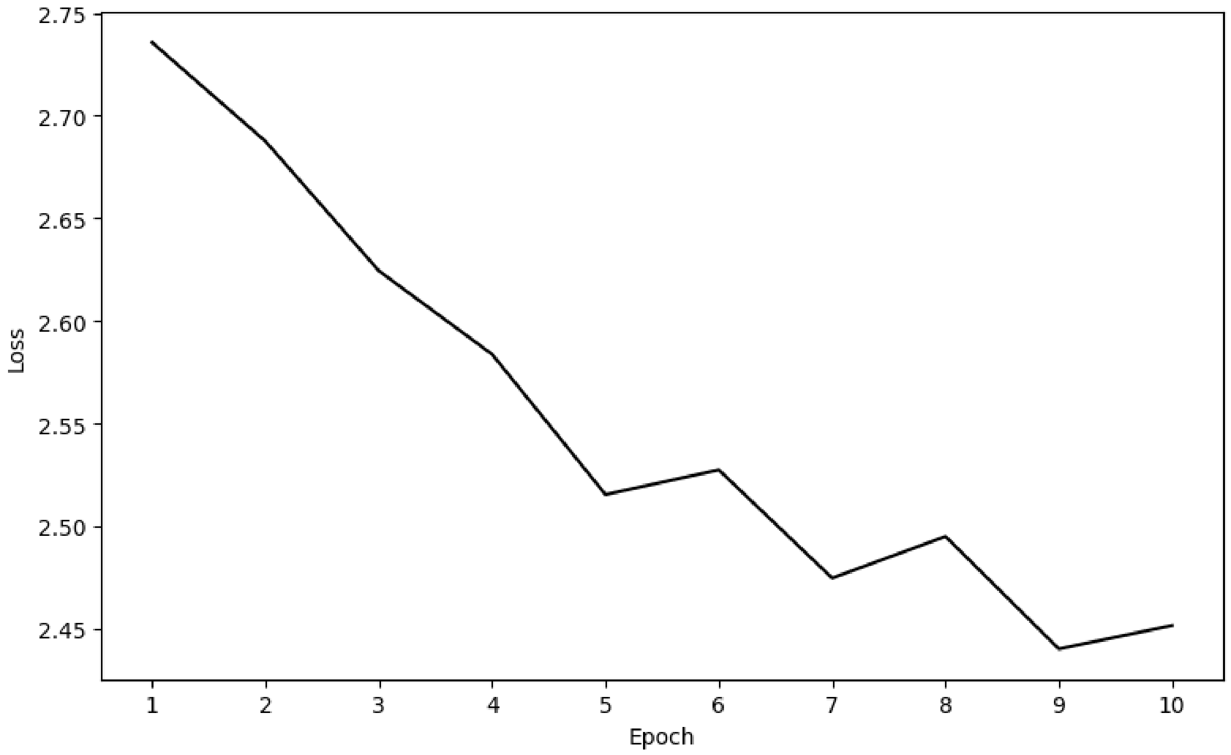

Figure 3: Cross entropy loss at each training epoch for fine-tuning.

After fine-tuning, the word embeddings using this model were obtained in the same fashion that was used to obtain the embeddings for the base BERT model.

4 Modeling

4.1 One-Hot Encoding, Word2Vec, GloVe

4.1.1 Ridge Regression

To model how the brain reacts to auditory inputs from stories, ridge regression was used. For each subject and word embedding technique used, the embeddings were used as features and the corresponding BOLD signals for each voxel were used as the response. The regularization strength for each voxel was tuned through cross-validation. First, the stories were split into a training and testing set using a 80%-20% split. Then cross-validation was performed by dividing the training stories further stories in the training set into 5 separate validation sets, with each validation set containing 20% of the stories. New one-hot-encoding embeddings were also obtained at each fold to ensure that there is no data leakage. With the 5-fold cross-validation, the cross-validation correlation coefficients (CV CC) were obtained by taking the mean of the correlation coefficients across folds. The CV CC's were obtained for each embedding type, subject, voxel, and 10 different values of α, ranging from 10^0 to 10^3. Then, for each embedding type, subject, and voxel, the α that resulted in the largest absolute value of the CC were chosen as the optimal regularization strength. We took the absolute values for the CC's since a strong negative CC may be as informative as a strong positive CC for our investigation. Table 3 displays the mean CV CC data for each subject-embedding type pair when using the optimal α values for each voxel.

Metric	**Subject 2**	**Subject 3**
Mean CV CC (One-Hot Encoding)	0.009052	0.016591
Mean absolute CV CC (One-Hot Encoding)	0.015455	0.020750
Mean CV CC (Word2Vec)	0.014903	0.026026
Mean absolute CV CC (Word2Vec)	0.019603	0.028617
Mean CV CC (GloVe)	0.015612	0.025246
Mean absolute CV CC (GloVe)	0.020171	0.028185

Table 3: Mean cross-validated correlation coefficients (CC) and absolute CCs for One-Hot-Encoding, Word2Vec and GloVe embeddings with the best alphas.

Based on this data, it is clear that the one-hot-coding embedding technique performs the worst when trying to predict BOLD signals in the brain. The two pretrained word embedding models, Word2Vec and GloVe, perform identically with the optimal alphas. However, since the Word2Vec model used for this contains vectors for 3 million words as opposed to the GloVe model, which only has 400,000 words, the Word2Vec embeddings were used to create the final model. This makes the model more robust, as it is more unlikely to encounter future data with a word that cannot be embedded. Interestingly, the trend of subject 2 having lower correlation coefficients compared to subject 3 persists even when using the optimal hyperparameters.

Then, all of the training data using the Word2Vec embeddings was used to fit a final regression models with the optimal α values found through cross-validation. Summary statistics of the resulting correlation coefficients obtained using the test set are shown in Table 4.

Metric	**Subject 2**	**Subject 3**
Mean CC	0.015265	0.027443
Mean absolute CC	0.019511	0.029954
Median CC	0.011636	0.019570
Top 5% CC	0.058230	0.095278
Top 1% CC	0.085571	0.141896

Table 4: Summary statistics of voxel-wise correlation coefficients (CC) for each subject using the best alphas and Word2Vec embeddings.

The mean CC and the mean absolute CC for the final model are in line with what was found during cross-validation, which ensures that we are not overfitting to the training data. To dive deeper into the distribution of the correlation coefficients, they were plotted in Figure 4. In general, the CC's for subject 3 skewed more toward higher CC's compared to subject 2, which is reflected in the mean and the quantiles of the CC's. In addition, it can be observed that the distribution of the CC's are centered fairly closed to 0, meaning that the model does not work well across all voxels. This is not surprising, since not every part of the brain is expected to react to auditory cues. Conversely, it is likely that the parts of the brain that correspond to voxels that had high CC's are involved in processing auditory signals. This relationship can be investigated more to leverage the predictability component of the PCS framework further.

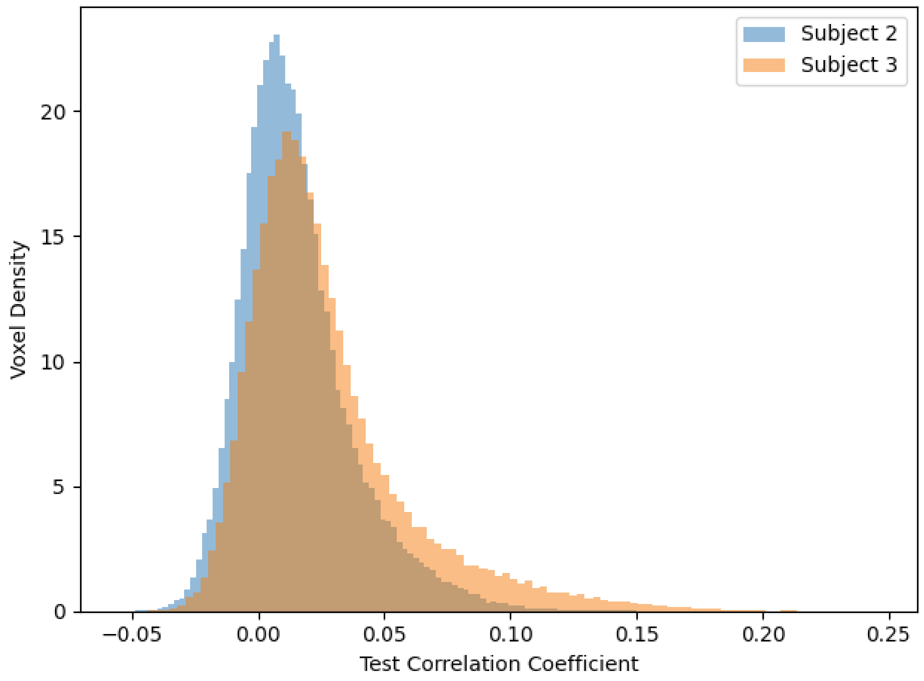

Figure 4: Distribution of ridge regression correlation coefficients on the test set using the optimal alphas for each subject and Word2Vec embeddings.

4.1.2 Stability Analysis: Word2Vec

Lastly, a stability analysis was performed to validate our findings. For this, the ridge regression was fitted with the same optimal alphas for Word2Vec embeddings, but with 30 different train-test splits. The mean CC's for the different train-test splits are plotted in Figure 5. This figure shows that the mean CC's for the different train-test are consistent with the mean CC obtained from the original train-test split.

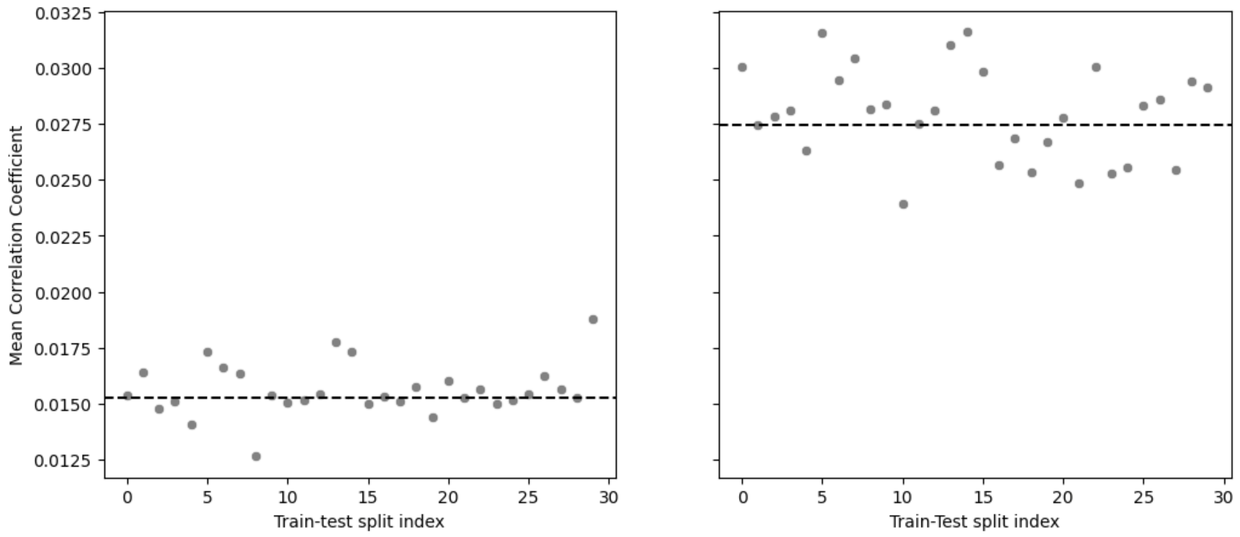

Figure 5: Scatter plots for the mean correlation coefficient using Word2Vec for 30 different train-test splits and the original train-test split (dashed line) for subject 2 (left) and subject 3 (right).

The CC's for the new train-test splits were aggregated and their distribution were plotted against the distribution of the original test set CC's in Figure 6. The distributions from the original train-test split and the new train-test splits have a significant overlap, further validating our results.

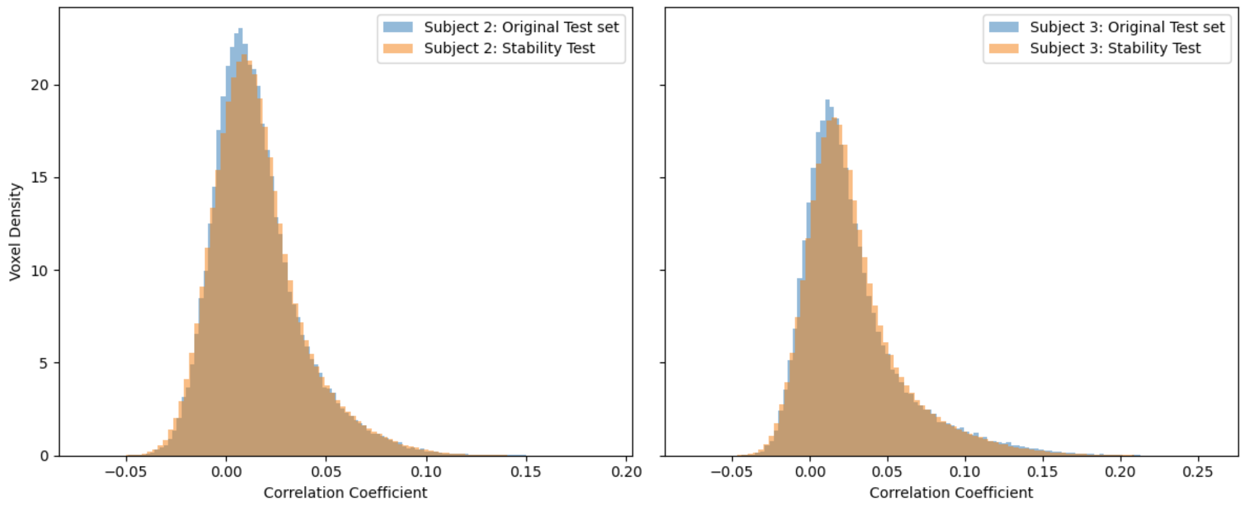

Figure 6: Distribution of ridge regression correlation coefficients using Word2Vec on different train-test splits.

4.2 BERT-style Masked Language Model

4.2.1 Ridge Regression

Using the three encoders chosen, word embeddings for the stories were generated. Then, a ridge regression was fit using these embeddings as the features and the corresponding BOLD signals in the same way as done for One-Hot Encoding, Word2Vec, and GloVe. The regularization strength for each voxel was tuned through cross-validation, using the same procedure as done for the other embedding methods with the 5-fold cross-validation scheme. Since the encoders are pretrained, there was no need to generate new embeddings for each fold. The CV CC's were obtained for each embedding type, subject, voxel, and 10 different values of α, ranging from 10^0 to 10^3. Then, for each embedding type, subject, and voxel, the α that resulted in the largest absolute value of the CC were chosen as the optimal regularization strength. Table 5 displays the mean CV CC data for each subject-embedding type pair when using the optimal α value for each voxel.

Metric	**Subject 2**	**Subject 3**
Mean CV CC (Encoder 1)	0.004836	0.010408
Mean absolute CV CC (Encoder 1)	0.010178	0.013663
Mean CV CC (Encoder 2)	0.005488	0.009847
Mean absolute CV CC (Encoder 2)	0.010144	0.012962
Mean CV CC (Encoder 3)	0.008078	0.012609
Mean absolute CV CC (Encoder 3)	0.011943	0.015302

Table 5: Mean cross-validated correlation coefficients (CC) and absolute CCs for the three encoder embeddings with the best alphas.

Based on this data, it seems that Encoder 3, which has a hidden size of 256, 8 heads, 4 layers, and an intermediate size of 512, outperforms the other encoders for this specific task. So, all of the training data using Encoder 3 embeddings were used to fit a final ridge regression model with the optimal regularization strengths. Then, the test set was used to evaluate the model. The results obtained from this model is compared with the results obtained from the ridge regression model using Word2Vec embeddings in Table 6.

Metric	**Subject 2**	**Subject 2**	**Subject 3**	**Subject 3**
Embedding	*Encoder 3*	*Word2Vec*	*Encoder 3*	*Word2Vec*
Mean CC	0.009194	0.015265	0.012620	0.027443
Mean absolute CC	0.013704	0.019511	0.016786	0.029954
Median CC	0.008374	0.011636	0.010417	0.019570
Top 5% CC	0.034435	0.058230	0.044207	0.095278
Top 1% CC	0.053904	0.085571	0.078554	0.141896

Table 6: Summary statistics of voxel-wise correlation coefficients (CC) for each subject using the best alphas with Encoder 3 and Word2Vec embeddings.

Overall, the performance of this model is consistent with the cross validation performance so the model did not overfit to the training data. However, it seems that model using embeddings obtained from Encoder 3 underperformed in many metrics compared to model using Word2Vec embeddings when it comes to predicting fMRI signals using word embeddings. Across both subjects, Word2Vec embeddings consistently outperform Encoder 3 embeddings across all summary statistics, with higher mean, median, and top correlation coefficients. The improvement is especially pronounced in the top 1 percent of voxels, where Word2Vec nearly doubles the correlation compared to Encoder 3 for both subjects. This may be due to the fact that there is limited training data for the encoder and that the encoder was trained to minimize the loss when predicting masked words instead of training to optimized the performance on this ridge regression task. For both embeddings, the CC's for subject 3 skewed more toward higher CC's compared to subject 2, which is reflected in the mean and the quantiles of the CC's. In addition, it can be observed in Figure 7 that the distribution of the CC's using Encoder 3 embeddings are centered fairly closed to 0, meaning that the model does not work well across all voxels. This is similar to what was observed with CC's when using Word2Vec embeddings. As mentioned before, we hypothesize that the parts of the brain that correspond to voxels that had high CC's are involved in processing auditory signals.

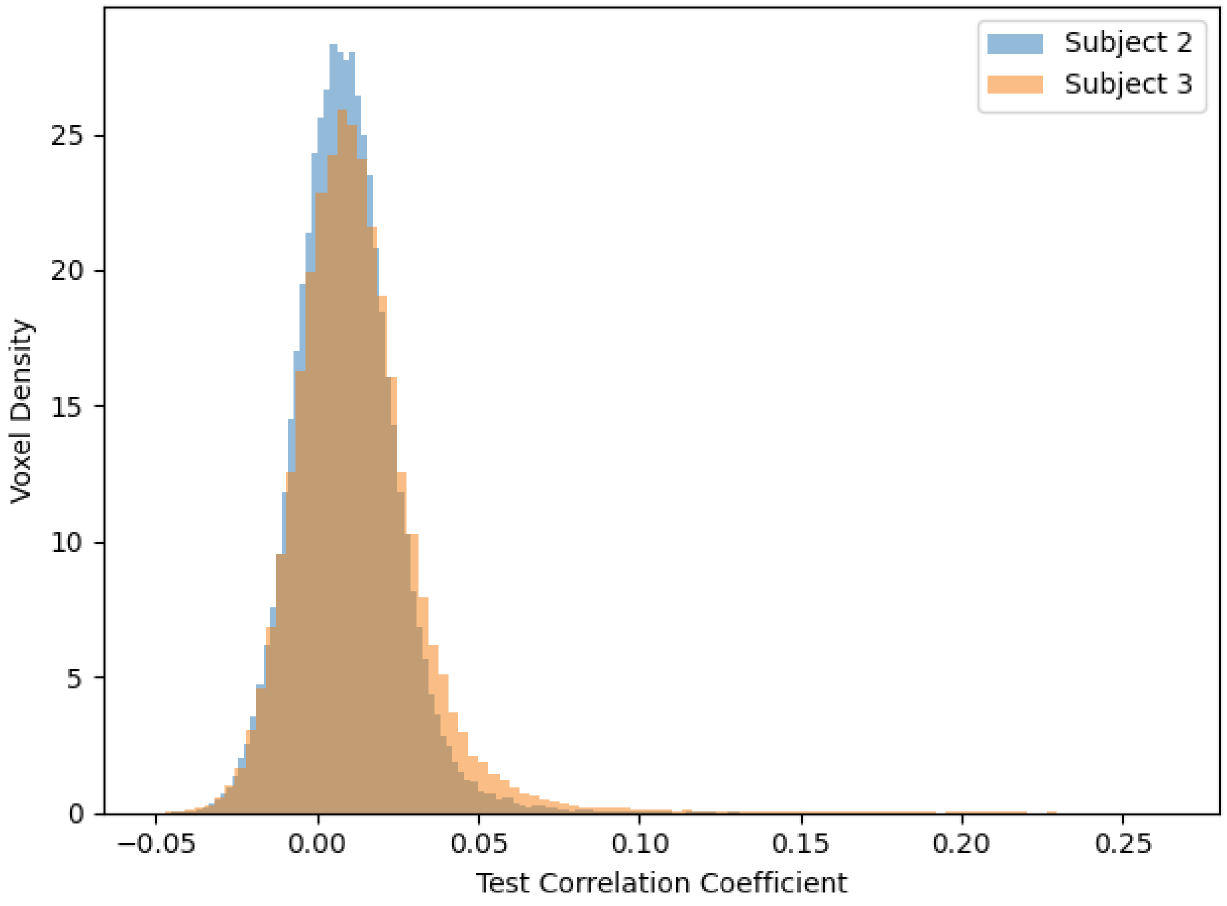

Figure 7: Distribution of ridge regression correlation coefficients on the test set using the optimal alphas for each subject and Encoder 3 embeddings.

In Figure 8, the distribution of the CC's for the ridge models using Encoder 3 and Word2Vec embeddings are directly compared for each subject. In general, the distribution of the correlation coefficients for the model using Word2Vec skews to the right more. This indicates that the Word2Vec embeddings are better for predicting the fMRI signals and potentially for identifying voxels that correspond to the parts of the brain that processes sounds.

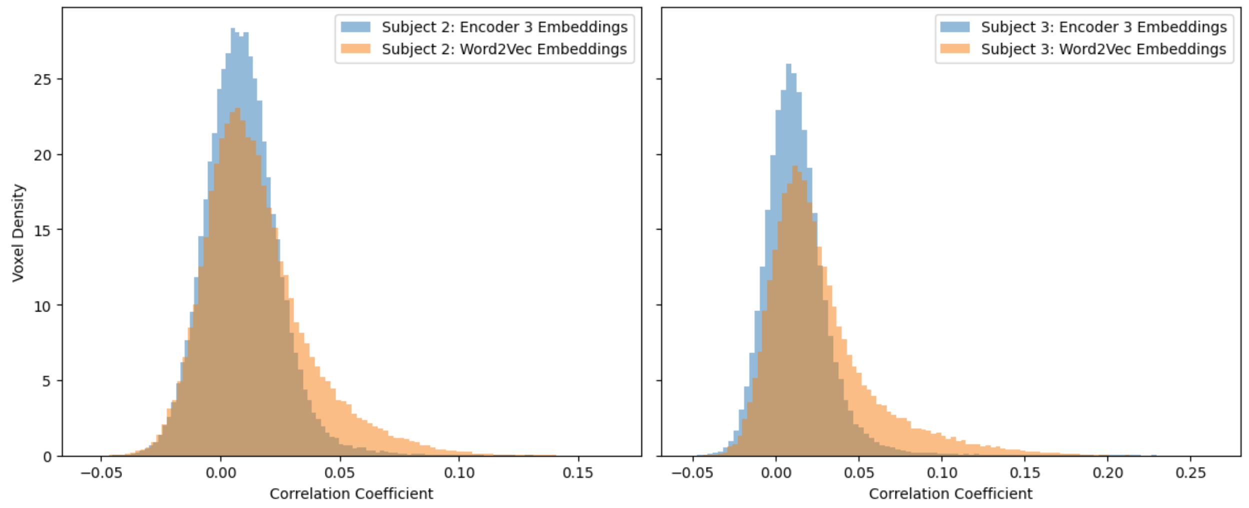

Figure 8: Comparison of the distribution of ridge regression correlation coefficients between the Encoder 3 and Word2Vec embeddings.

4.2.2 Stability Analysis: BERT-Style Masked Language Model

A similar stability analysis was performed using the embeddings obtained from Encoder 3. Based on Figure 9, it is clear that the results obtained from the original train-test split are consistent with results obtained from different train-test splits.

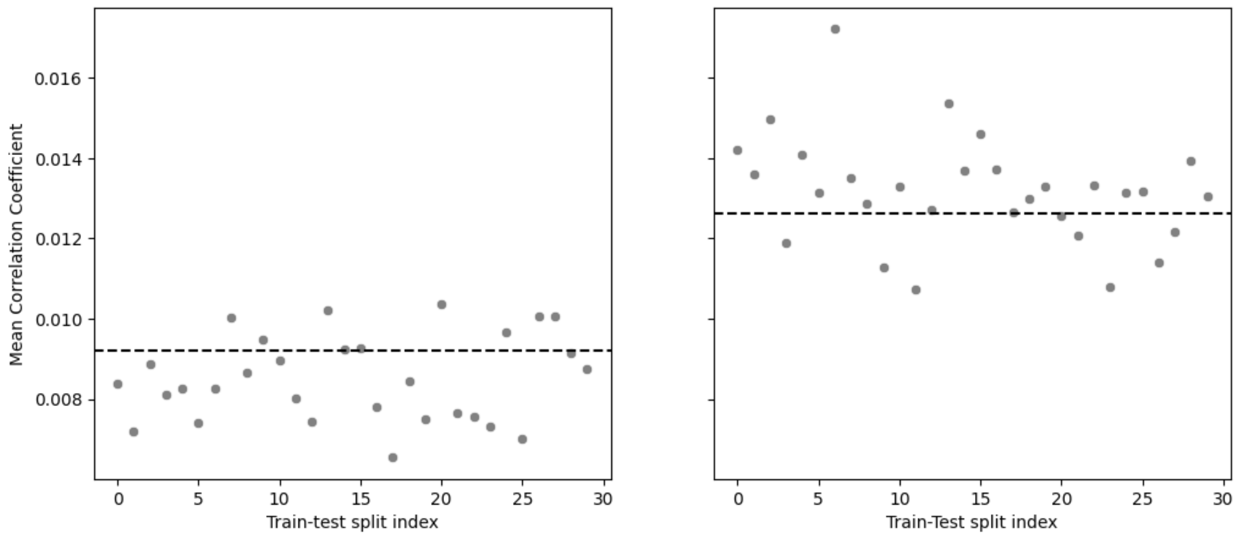

Figure 9: Scatter plots for the mean correlation coefficient using Encoder 3 for 30 different train-test splits and the original train-test split (dashed line) for subject 2 (left) and subject 3 (right).

Figure 10 shows that the distribution of the CC's from the original train-test split are very close to the distribution of the CC's obtained from 30 other train-test splits when using the embeddings from Encoder 3. These findings validate the stability of our analysis.

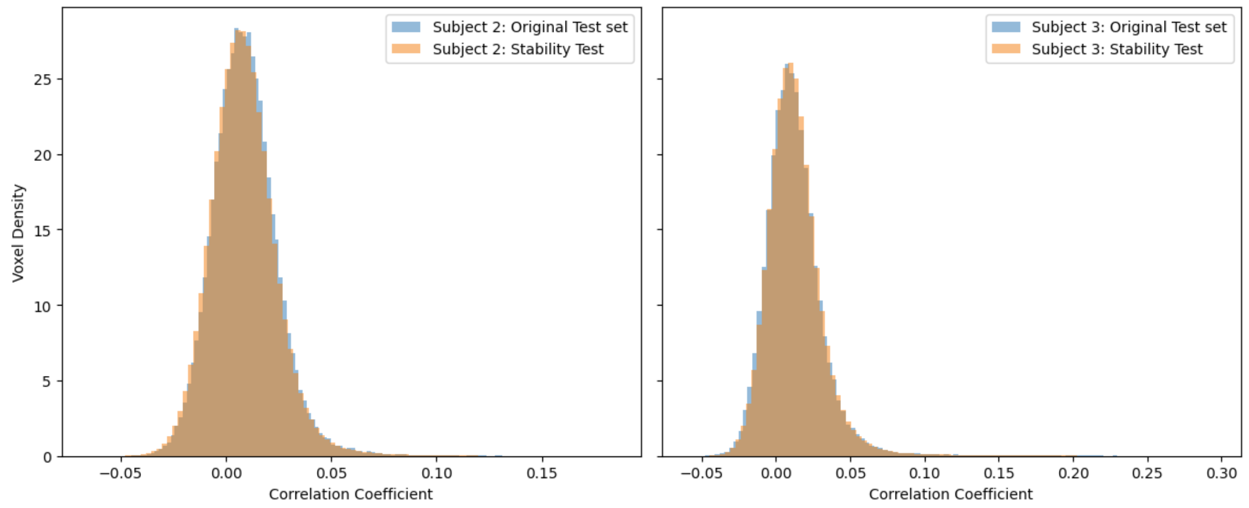

Figure 10: Distribution of ridge regression correlation coefficients using Encoder 3 on different train-test splits.

4.3 Pretrained and Fine-tuned BERT-style Masked Language Model: Hugging Face

4.3.1 Ridge Regression

Similarly with the other encodings, word embeddings for the stories were generated, then a ridge regression was fit using these embeddings as the features and the corresponding BOLD signals. The regularization strength for each voxel was tuned through cross-validation, using the same procedure as done for the other embedding methods with the 5-fold cross-validation scheme. Since the encoders are pretrained, there was no need to generate new embeddings for each fold. The CV CC's were obtained for each embedding type, subject, voxel, and 10 different values of α, ranging from 10^0 to 10^3. Then, for each embedding type, subject, and voxel, the α that resulted in the largest absolute value of the CC were chosen as the optimal regularization strength. Then, all of the training data was used to fit the ridge regression with the optimal values of α. The correlation coefficients were then obtained using the test set. Table 7 and Table 8 display the correlation coefficient data for different embedding types when using the optimal α value for each voxel.

Embedding	**Mean CC**	**Mean Abs. CC**	**Median CC**	**Top 5% CC**	**Top 1% CC**
Fine-Tuned BERT	0.009324	0.015195	0.008366	0.039075	0.059760
Pretrained BERT	0.009726	0.015487	0.008802	0.039588	0.060577
Encoder 3	0.009194	0.013704	0.008374	0.034435	0.053904
Word2Vec	0.015265	0.019511	0.011636	0.058230	0.085571

Table 7: Summary statistics of voxel-wise correlation coefficients (CC) for subject 2 using the best alphas with various embeddings.

Embedding	Mean CC	Mean Abs. CC	Median CC	Top 5% CC	Top 1% CC
Fine-Tuned BERT	0.019695	0.022946	0.016299	0.061935	0.097427
Pretrained BERT	0.020512	0.023574	0.017044	0.063311	0.098996
Encoder 3	0.012620	0.016786	0.010417	0.044207	0.078554
Word2Vec	0.027443	0.029954	0.019570	0.095278	0.141896

Table 8: Summary statistics of voxel-wise correlation coefficients (CC) for subject 3 using the best alphas with various embeddings.

Based on the correlation coefficient data, it seems that the pretrained base BERT model from Hugging Face performs better than the fine-tuned model. However, this difference is minimal and could be due to the choice of stories in the train-test split. In addition, all of the embeddings tested show that the CC's for model for subject 3 are higher compared to the CC's for the model for subject 2. Overall, the Word2Vec embeddings still outperform any of the other embeddings tested. The distribution of these correlation coefficients are sown in Figure 11.

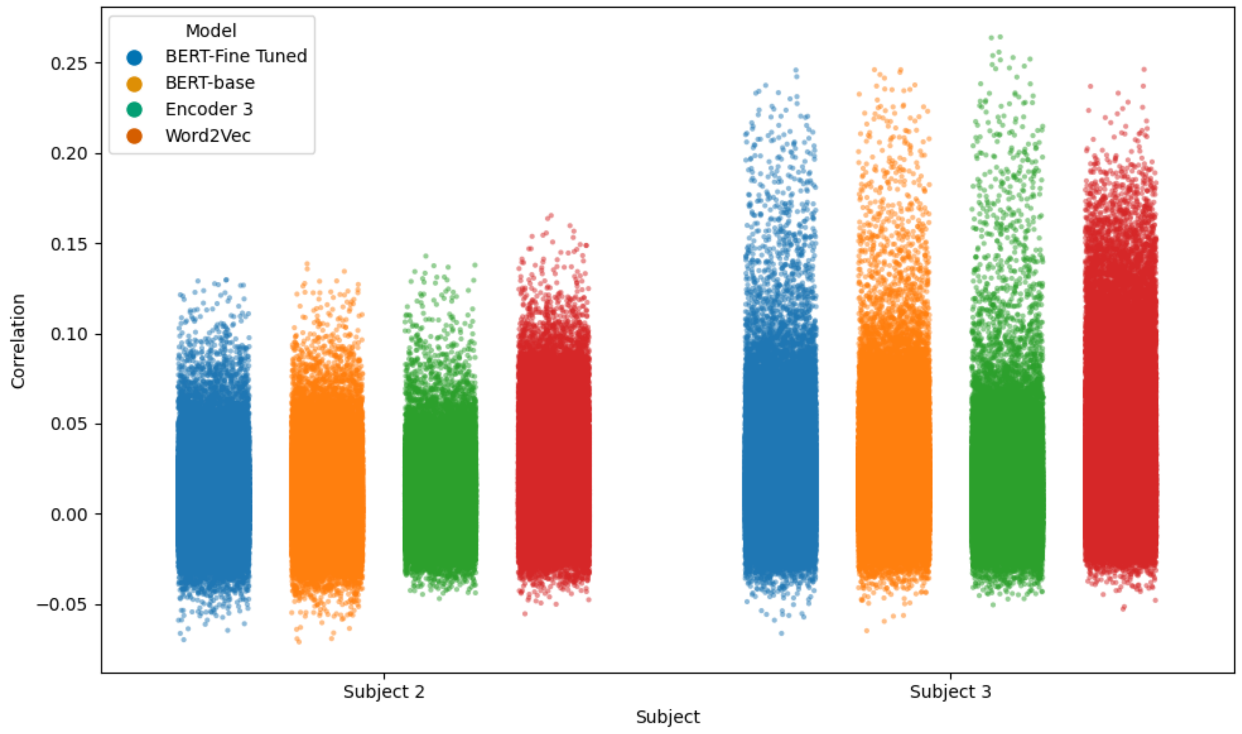

Figure 11: Distribution of the correlation coefficients for the ridge regression model using different embeddings and optimal regularization strengths.

Similarly to before, the CC's for the fine-tuned and base BERT models from Hugging Face indicate that the model does not perform uniformly well across all voxels. As mentioned previously, this could be due to the fact that certain parts of the brain are more involved with sensing sound and interpreting words.

4.3.2 Stability Analysis: Fine-tuned BERT-style Masked Language Model

The ridge regression model with the optimal regularization strengths were fit with 30 additional train-test splits for a stability analysis. The mean correlation coefficients for each train-test split are shown in Figure 12.

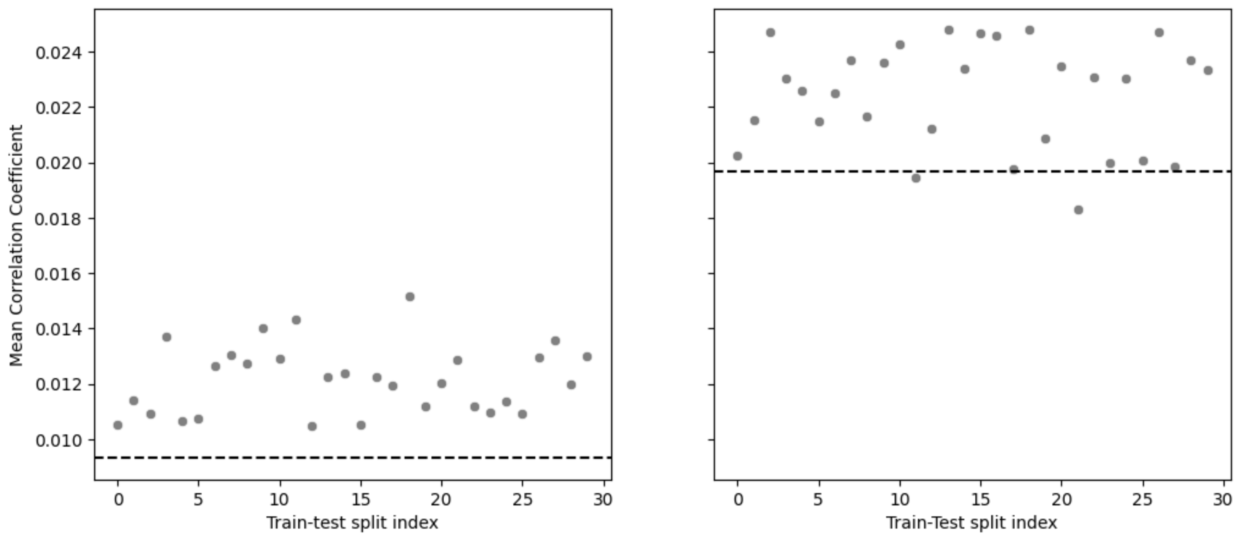

Figure 12: Scatter plots for the mean correlation coefficient using the fine-tuned BERT model for 30 different train-test splits and the original train-test split (dashed line) for subject 2 (left) and subject 3 (right).

Based on this scatter plot, it seems like the original train-test split resulted in a relatively low performance compared to the other splits for both subjects 2 and 3. However, the mean correlation coefficients from the different splits are not extremely far off from each other, meaning that the conclusions we derive from this result are still reasonable. The distribution of the correlation coefficients from the original train-test split is compared with the distribution correlation coefficients from all of the 30 new train-test splits in Figure 13. This shows that the distribution of the correlation coefficients from the original split largely overlaps with the distribution from the other 30 train-test splits, establishing stability.

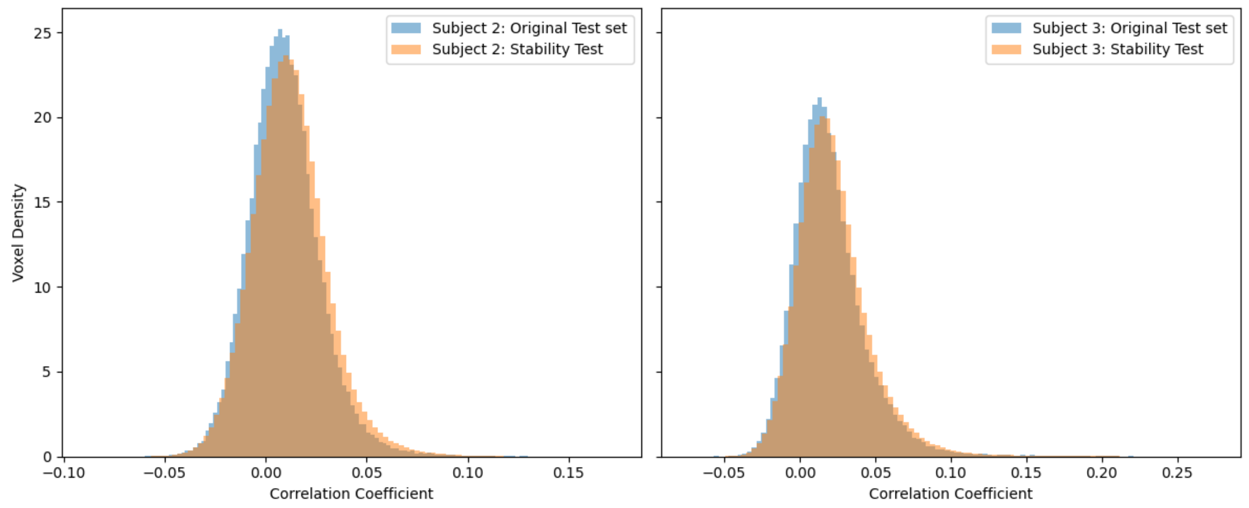

Figure 13: Distribution of ridge regression correlation coefficients using the fine-tuned BERT model on different train-test splits.

4.3.3 Influential Words

To measure influential words, we ran SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) across the voxels where our model performed the best, to the first and second stories in our test story set: ”cocoonoflove” and ”escapefromadirediagnosis”. To do this, we created a function that uses the Python ”shap” package’s ”LinearExplainer” function on our Ridge model to get the SHAP importances for each word. We made a similar function for LIME using the ”lime” package’s ”LimeTabularExplainer” function also for the Ridge model.

Table 9: Top-3 words (with voxel-frequency) and # unique words identified by SHAP vs LIME for stories ’cocoonoflove’ and ’escapingfromadirediagnosis’

Story	Subject	Method	Top-3 Words (freq)	# Unique Words
`cocoonoflove`	2	LIME	`one`(3), `they`(2), `sat`(2)	23
	2	SHAP	`appearance`(10), `because`(9), `was`(9)	5
	3	LIME	`i`(4), `the`(4), `over`(4)	17
	3	SHAP	`appearance`(9), `because`(9), `was`(8)	7
`escapingfromadirediagnosis`	2	LIME	`when`(2), `since`(2), `nance’s`(2)	26
	2	SHAP	`’` (7), `right`(7), `t`(5)	8
	3	LIME	`my`(6), `the`(2), `from`(2)	21
	3	SHAP	`’` (8), `t`(7), `right`(7)	7

More specifically, for each of the two subjects, we ran both algorithms on the top 10 best performing voxels, then extracted the top 3 most influential words from calculating the L2 norm across the word dimension of the SHAP and LIME values. We ran these algorithms for only the most stimulated voxels because only specific regions of the brain are stimulated for auditory processing; therefore, we concluded that the results would be the most interesting or telling in these voxels. Running these algorithms on additional voxels would be too time intensive and likely wouldn't reveal relevant insights because of these voxels were not activated while listening to the stories. Table 9 shows the words that appear in the top 3 words of the top 10 voxels at the highest frequency.

The first story that we analyzed was 'cocoonoflove'. Using SHAP, we found that the most influential words were fairly similar between the two subjects, including the words "body", "because", "was", "appearance", "thanks", "talk", and "like". The words "appearance", "was", and "because" appeared in nearly every voxel for both subject 2 and subject 3. However, using LIME yielded very different results. Not only did the voxels have more differences for a given subject, but also both subjects had strong reactions to different words when measuring for importance. Some of the new words found to be the most important in LIME include "village", "war", "labor", "nazis", "invaded", "bribe" and "extermination".

This did not come to a surprise to our group because of the critical differences on how SHAP and LIME are calculated. By construction, SHAP accounts for feature interactions due to its extensive coalition-based approach, while LIME uses a linear surrogate model that often misses complex interactions. In action, this meant that SHAP was more likely to emphasize to transition words and thematically important words to the broader context of the text. By contrast, LIME highlighted words that stand out as important or "eye-catching" on their own. This also explains why LIME had such different results between the two subjects, as their individual experience could identify different highlighted words, while with SHAP we had similar words for each subject because their importance to the broader story rendered the same for both of their brains.

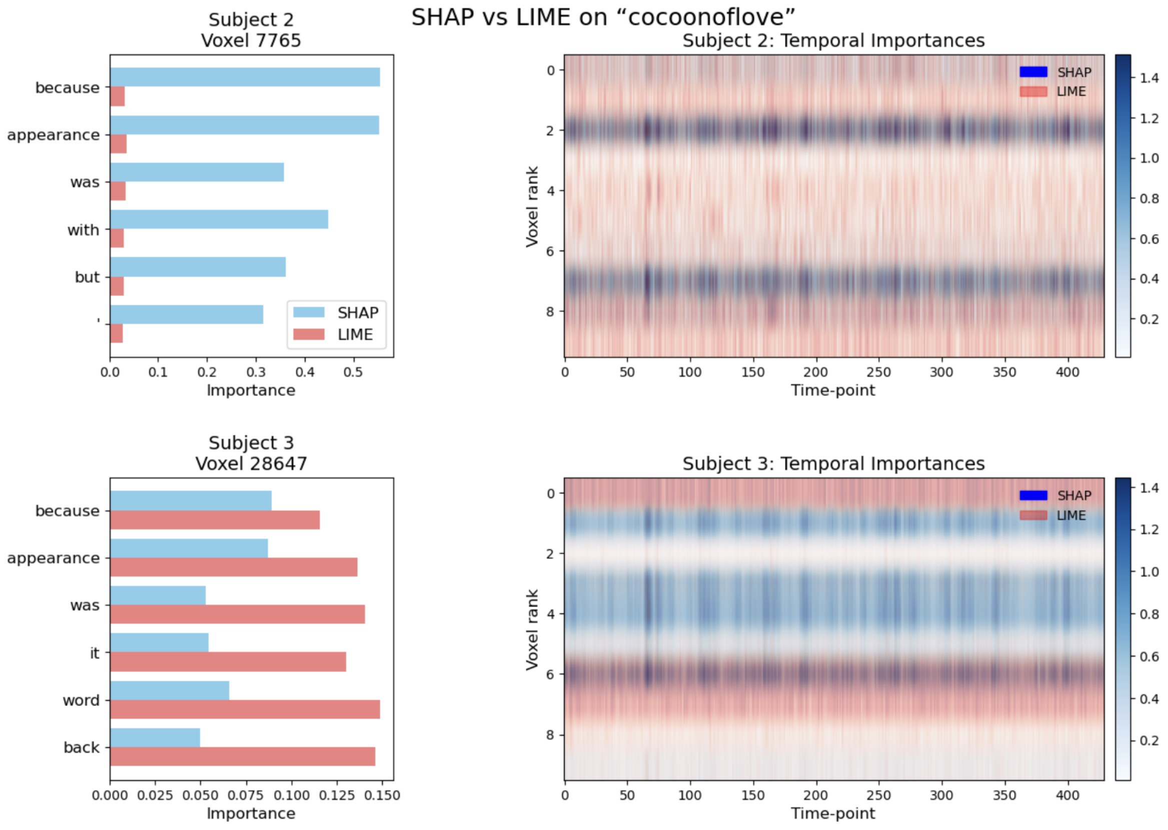

Figure 14: Comparison of SHAP vs LIME explanations for the story "cocoonoflove", across Subject 2 (top row) and Subject 3 (bottom row).

Figure 14 shows how the word importances compared when using LIME vs SHAP. The left panels show the union-bar-chart of the top 3 words for each method on the single highest-correlated voxel. The right panels show overlaid temporal heatmaps of the top 10 voxels (SHAP in blue, LIME in transparent red). LIME and SHAP show very different sensitivity for subjects 2 and 3. For subject 2, LIME displays nearly uniform importance across all timepoints in the important voxels, whereas SHAP shows much higher peaks at voxels ranked 2 and 7 - indicating that a few regions of the brain are especially important for Subject 2's response. On the other hand, subject 3 shows more agreement between LIME and SHAP. The temporal overlay shows co-localized red (LIME) and blue (SHAP) bands at several of the top 10 voxels, suggesting that for Subject 3 the narrative produces more consistent activations across multiple regions that both methods capture.

We repeated our analysis on the story "escapingfromadirediagnossi" to check the stability of our results. We decided to use the same process to gather the most important words based on SHAP/LIME L2 norms. Important words identified by SHAP included "don't", "blonde", "what", "right", "about", "right" and "can". Again, we saw much overlap in both subjects from SHAP for this method. Important words identified by LIME in this story included "medical", "dating", "matchdotcom", "glasses", "friendship", "get", "profile", "straightforward", and "meaning". Again we saw with LIME that the words stood out on their own from the story, and we did not see too much overlap between the most LIME important words between the two subjects.

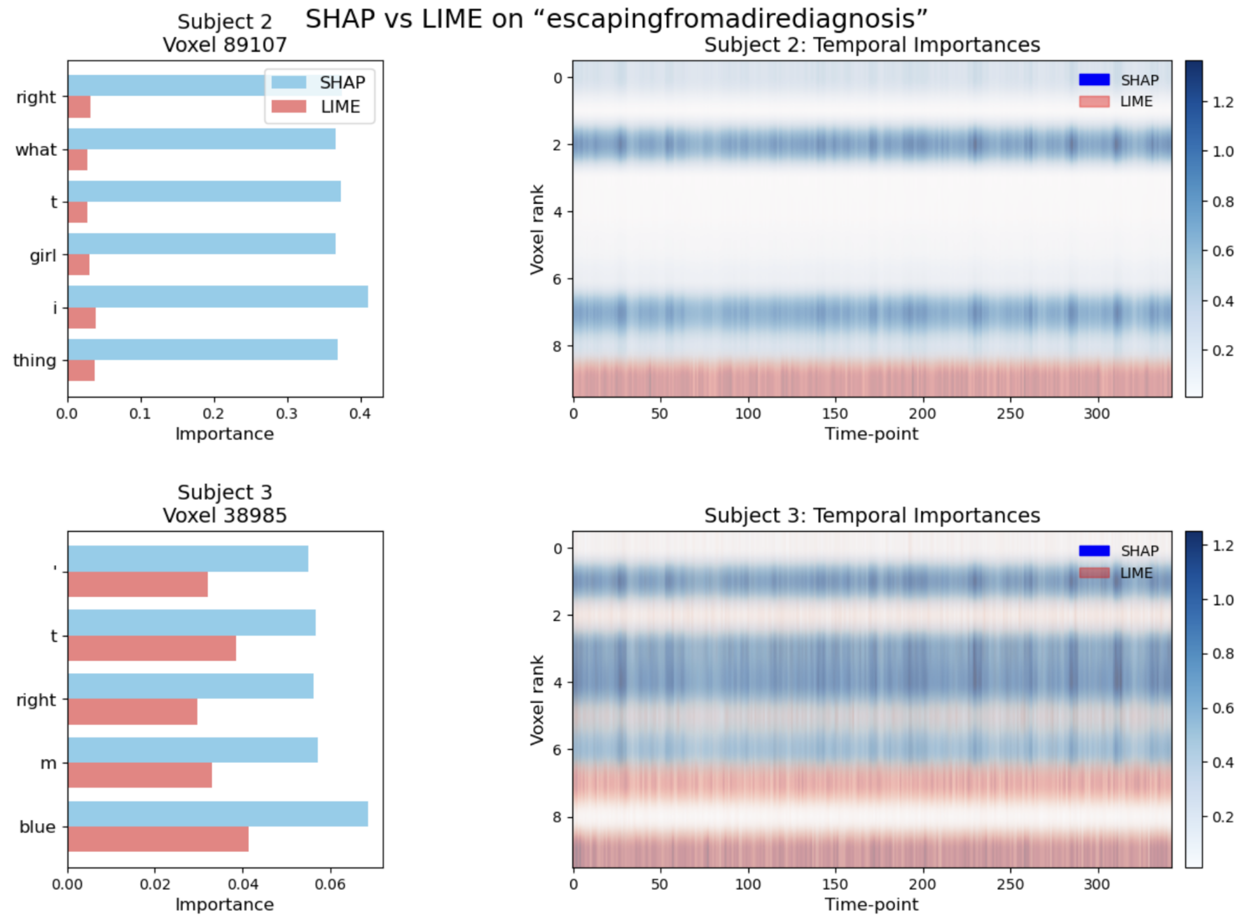

Figure 15: Comparison of SHAP vs LIME explanations for the story "escapingfromadirediagnosis", across Subject 2 (top row) and Subject 3 (bottom row).

Our results from running this analysis on the second test story corroborated with what we found with "cocoonoflove". SHAP did a better job isolating individual brain regions that seemed to have a higher importance to the story, while LIME focused more on context independent words that might be relevant such as "matchdotcom" and "dating". We also saw a similar trend in the results between subjects 2 and 3. SHAP and LIME showed more agreement in subject 3 than in subject 2, where there was nearly no overlap in the words found.

5 Conclusion

Our findings demonstrate that with One-Hot Encoded, GloVe, and especially Word2Vec embeddings, a ridge regression model can explain the variability BOLD signals in some voxels in the brain. We were able to match dimensionality using Lanczos resampling between the fMRI data and our embeddings, and use concatenated delayed versions of our embeddings to account for the mismatching of temporal frequency in fMRI readings. Additionally, the performance improvement from the one-hot encoded method to the Word2Vec method demonstrates the importance of context when modeling how the brain processes language.

With an aim to achieve higher performance, BERT-style masked language model encoders were trained using the stories provided. However, using a ridge regression model with word embedding generated from these encoders performed worse than the model using Word2Vec embeddings. This could be due to the fact that there is limited data available to use to train the encoder and the way that this model was trained.

So, pretrained BERT model trained on a much larger dataset was next chosen and fine tuned using the text from the stories provided. Even the, the embeddings generated from the encoders performed worse than the model using Word2Vec embeddings.

Despite our efforts, our performance on the surface appears fairly underwhelming, with the best mean absolute correlation coefficient demonstrated was about 0.02 using Word2Vec embeddings. This is likely due to the fact that only some regions of the brain process audio, and further analysis of voxels within some top $\alpha < 0.05$ percentile could confirm this using their cluster proximity in combination with more domain knowledge of neuroscience.

So, for the top 10 voxels, we analyzed what words influence the BOLD signals using SHAP and LIME. However, no conclusive statement can be made on what words influence the signals the most, likely due to the fact that this is a noisy problem and cannot be modeled well with ridge regression.

References

Shailee Jain and Alexander Huth. Incorporating context into language encoding models for fmri. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper_files/paper/2018/file/f471223d1a1614b58a7dc45c9d01df19-Paper.pdf`.