Reconstructing Sound From Brain Responses To Music Stimuli

Cagan Bakirci
USC
cbakirci@usc.edu

Neval Cam
USC
ncam@usc.edu

Asmaa Hassan
USC
hassanas@usc.edu

Joel Mohammed-Paige
USC
joelmoha@usc.edu

Abstract

This study explores reconstructing music stimuli from EEG signals, building on the EEG2Mel paper. Using the NMED-T dataset, we preprocessed EEG data into power spectral density representations and paired it with mel-spectrogram outputs of the music stimuli. We implemented different models architectures to map EEG signals to audio features, optimizing performance through rigorous preprocessing and model tuning. Our best model outperformed the baseline EEG2Mel study with accuracy of 82.86% (cosine similarity: 0.80). This research highlights the potential of deep learning in decoding brain responses to music and provides insights for future advancements in neural signal processing.

1 Introduction

In this paper, we focus on reconstructing music from electroencephalogram (EEG) signals. EEG provides temporal data that reflects the brain's response to auditory stimuli, but its noisy and low spatial resolution nature presents significant challenges for decoding complex stimuli like music. Despite these obstacles, recent research, such as the EEG2Mel study[1], has demonstrated the feasibility of reconstructing music stimuli from EEG signals. Building on this foundation, our goal is to replicate and extend the findings of the EEG2Mel study by implementing various models to decode EEG signals and reconstruct the corresponding music stimuli. We replicate the baseline CNN approach introduced in the EEG2Mel study, validating its efficacy in reconstructing music from EEG signals. Then, we explore the use of more advanced techniques not used in EEG2MEL study such as recurrent convolutional neural networks (RCNNs), encoder-decoder, and Transformer architectures to map EEG power spectrum density (PSD) representations to mel-spectrogram outputs.

2 Methods

2.1 Dataset

This paper utilized the Naturalistic Music EEG Dataset - Tempo (NMED-T)[2], an open dataset designed to facilitate research on brain responses to music stimuli. The dataset is comprised of EEG recordings from 20 adult participants listening to music. Each participant listened to 10 songs with electronically produced beats. The songs were presented in full-length (4:30–5:00 minutes) covering a range of tempos and genres. Data was collected using a 128-electrode dense-array system at a sampling rate of 1 kHz, later downsampled to 125 Hz during preprocessing. Preprocessing involved high-pass and notch filtering, artifact removal using Independent Component Analysis, and aggregation across participants into song-specific matrices. The dataset includes both raw and preprocessed EEG data.

For this project, we exclusively used the preprocessed EEG data. In the EEG2MEL paper, which served as a reference for our work, the authors compared results obtained using raw versus imputed EEG data. Their findings indicated that models trained on raw data exhibited significantly lower accuracy compared to those trained on imputed EEG data. This suggests that cleaning the data reduces noise and provides better results. Additionally, processing raw data proved computationally expensive as resources available to us, including personal computers and Google Colab Pro, were insufficient. Based on these two insights, we made the decision to focus exclusively on imputed EEG data. This allowed us to direct our efforts toward optimizing other aspects of the data processing pipeline, such as transforming the EEG data into different PSD representations and designing a multitude of model architectures.

2.2 Data Preprocessing

For this study, each EEG recording (X) was trimmed to 270 seconds and segmented into 1-second chunks, resulting in 270 chunks per song for each participant. This process yielded a total of 54,000 data points across all participants and songs. Each data point captures the activity of all 125 electrodes over a 1-second window, represented as a 2-D array of 125 × 125 (electrodes × samples).

Similarly, for the corresponding stimulus (Y), the 10 songs were trimmed to 270 chunks and repeated for all 20 participants. This results in 54,000 data points that correspond to the EEG data points ensuring a one-to-one mapping between X and Y. Each data point is a 1-D array of 22050 elements, representing the sampling rate used to process the audio files.

2.2.1 Power Spectral Density (PSD)

To process the EEG signals for our models, we employed PSD analysis, a critical step in transforming EEG values into a frequency-domain representation. PSD provides insight into how power is distributed across various frequency components, reflecting the neural activity stimulated by music. This transformation was essential for understanding the underlying neural responses and enhancing the interpretability of our models. Specifically, we utilized Welch's method and the Periodogram approach to compute PSD.

Welch Welch's Method splits the EEG data into overlapping segments, computing the Fourier Transform for each, and averaging the spectral estimates. This approach reduces variance and produces a smoothed PSD. This makes it highly suitable for identifying consistent frequency components amidst noisy data. However, it might be less optimal for signals with minimal noise as smoothing can obscure subtle variations in the frequency.

Periodogram In contrast to Welch, the Periodogram method calculates the PSD directly from the Fourier Transform of the entire signal, offering higher frequency resolution. While more sensitive to noise compared to Welch's method, this sensitivity proves advantageous in scenarios demanding high precision, such as analyzing rapid neural responses to music stimuli.

After applying PSD transformation, the final dimensions of our X data is (54000, 63, 125, 1).

2.2.2 Mel-spectrogram

We utilized mel-spectrograms as a feature representation to model music stimuli. A mel-spectrogram is a time-frequency representation of an audio signal, with the frequency axis mapped onto the Mel scale; a scale that closely aligns with human auditory perception by emphasizing the frequencies most relevant to human hearing. Consequently, it captures the intricate auditory features that are likely to be encoded in the brain during music perception. This transformation preserves the temporal dynamics of the music while focusing on meaningful frequency bands. It provides a detailed yet compact representation, reducing data dimensionality and enabling computationally efficient processing.

To evaluate the impact of different mel-spectrogram representations on modeling performance, we explored two approaches.

Raw Mel-spectrogram Representation The first approach used the raw mel-spectrogram directly, representing the energy levels of the audio signal across the mel scale. This representation maintains the original linear scale of power values, capturing the energy distribution without further transformations.

Decibel (dB) Scale Representation The second approach transformed the mel-spectrogram into the decibel (dB) scale, compressing the dynamic range of the signal to reflect how humans perceive sound intensity on a logarithmic scale. By doing so, the dB scale emphasizes smaller variations in quieter sounds while reducing the influence of louder sounds, potentially highlighting subtler auditory features.

After applying Mel-spectrogram transformation, the final dimensions of our Y data is (54000, 128, 44).

We experimented with var data splits for training, validation, and testing, including 80/10/10 and 70/10/20 to evaluate model performance under different partitioning strategies.

2.3 Models & training

2.3.1 Baseline CNN

We implemented a baseline CNN similar to the architecture used in the original EEG2MEL study. The goal of this baseline model was to validate the original study and provide a comparison point for evaluating the performance of more advanced architectures. The CNN was designed to map PSD inputs to their corresponding mel-spectrogram representations.

2.3.2 RCNN

Music and its corresponding EEG signals exhibit complex temporal patterns. While CNNs are effective for extracting local spatial features, RCNNs allow modeling sequential dependencies in data. With this approach in mind, we used the RCNN architecture to leverage temporal dependencies in the data while the convolutional layers maintained the integrity of local spatial features. These extracted features were then reshaped and passed through recurrent layers, modeling the temporal dynamics necessary for reconstructing musical stimuli.

2.3.3 Encoder-decoder

As a natural progression from RCNNs, we explored models with encoder-decoder architectures. The encoder-decoder framework was designed to explicitly separate feature extraction from sequential decoding, aiming to enable more flexible and powerful representations of the EEG-to-music mapping task.

The encoder consisted of two convolutional layers, similar in design to those used in the RCNN. These layers were followed by a Long Short-Term Memory (LSTM) layer to model the temporal dependencies, compressing the input sequence into the decoder. The decoder was designed to reconstruct mel-spectrogram sequences from the latent representation. It begins with a Repeat Latent Vector layer, which expands the latent vector across the temporal dimension to match the desired output sequence length. This is followed by an LSTM layer to model the mel-spectrogram and a fully connected dense layer to project the output into the mel frequency space.

2.3.4 Transformers

Aiming to overcome the limitations of sequential processing inherent in recurrent architectures, we explore implementing various Transformer-based architectures. Transformers are renowned for their ability to efficiently model long-range dependencies using self-attention mechanisms, which process entire sequences simultaneously rather than step-by-step[7]. Therefore, by leveraging attention weights, the model dynamically focuses on relevant time steps.

In our experiments, we implemented and evaluated various Transformer-based architectures and techniques:

- Positional Encodings: We used sinusoidal positional encodings to inject sequence order information into the model.
- Masking Techniques: Applied causal masks during decoding to ensure that predictions at a given time step depended only on past and current inputs.
- Teacher Forcing: During training, ground truth outputs were provided as inputs to the decoder to improve convergence.

- Scheduled Sampling: Gradually transitioned from teacher forcing to autoregressive decoding to simulate real inference conditions.
- Autoregressive Decoding: Implemented autoregressive generation for inference, predicting one time step at a time while feeding the output back into the decoder.
- Output Sequence Interpolation: Interpolated the encoder outputs to match the required sequence length for the decoder.
- Zero Initialization: During inference, initialized the decoder input with zeros to simulate a fully autoregressive generation scenario.

3 Implementation

This project was implemented using Python, leveraging a combination of different deep learning frameworks. Below, we provide a detailed breakdown of the implementation process.

3.1 Model development

For all model architectures described in the methods section:

CNN Baseline: Constructed and implemented to replicate the benchmark model in the baseline EEG2Mel study using a similar CNN architecture and experimenting with layers and hyperparameters.

RCNN, Encoder-Decoder, Transformers: We designed and implemented these models independently to explore different architectures for modeling the sequential nature of EEG and audio data.

3.2 Frameworks and Libraries

PyTorch: Used extensively for flexible and efficient implementation of custom architecture: Transformer, RCNN, and encoder-decoder models.

TensorFlow/Keras: Used for CNN-based models and to evaluate results across frameworks.

Librosa: Employed for audio processing, including mel-spectrogram transformations and feature extraction.

3.3 Preprocessing and Utilities

Key preprocessing and utility functions were implemented from scratch, including:

Data Preprocessing: Trimming, splitting, padding, and transforming EEG and corresponding audio stimuli into appropriate training formats.

Cosine Similarity Metric: Implemented to evaluate alignment between generated and target mel-spectrograms.

3.4 Reuse of Existing Code

While the primary model architectures and utilities were independently implemented, the EEG Song Classifier to measure accuracy was repurposed from an EEG classification paper[5]. The EEG2Mel reference paper also repurposed the same classifier. This classifier played a key role in evaluating the quality of reconstructed mel-spectrograms by mapping them to song labels. Using the same classifier architecture ensured consistency and facilitated direct comparisons with the baseline study.

4 Experiments

4.1 Hyperparameters

In our experiments, selecting and tuning hyperparameters played a critical role in optimizing the performance of all models. We systematically tested a range of hyperparameters, including learning rates, batch sizes, and optimizer settings to identify the best configuration for CNN, RCNN and encoder-decoder architectures. For the Transformer, we tuned parameters such as the number of

Table 1: Accuracy accross Different CNN Models

Inputs(1secs)	Targets(1secs)	Classifier Accuracy	Cosine Similarity (-1.00,1.00)
Periodogram	**Mel-dB**	**82.86%**	**0.80**
Periodogram	Mel	62.79%	0.43
Welch	Mel-dB	73.84%	0.79
Welch	Mel	57.56%	0.44

Table 2: Comparison of Classifier Accuracy Between EEG2MEL and Our CNN Model

Model	Classifier Accuracy
CNN (Periodogram+Mel-dB)	**82.86%**
EEG2MEL CNN	80.80%

encoder and decoder layers, the hidden dimension size (d-model), the number of attention heads, the feed-forward layer dimensions, and the dropout rate. We varied the number of convolutional filters, kernel sizes, and pooling strategies in the encoder, as well as the number of LSTM layers and hidden units in both the encoder and decoder.

4.2 Accuracy Metrics

EEG Song Classifier Reconstructed Mel outputs from our models were passed through a classifier that classifies the name of the song from each reconstruction. This type of validation has been used by the EEG2Mel paper we are replicating. We implemented this approach to objectively compare our baseline results to the reference paper. Table 1. shows a summary of our results from classified outputs across different modeling approaches. While Table 2 demonstrates comparison between EEG2Mel's best model compared to our best baseline model.

The reference paper does not specify which PSD method was used. During our experiments, we trained different models using both the Welch and Periodogram methods and evaluated the results using the EEG Song Classifier. As seen in Table 1, with the Welch method, our best model achieved an accuracy of 73.84%, while with the periodogram method, it achieved **82.86%**, surpassing the baseline paper's reported accuracy of **80.80%**. These results highlight the significant influence of the PSD estimation method on model performance.

Cosine Similarity To enhance our evaluation framework, we explored additional methods to assess the reconstructed mel-spectrograms. While classification evaluates whether the model successfully identifies the song, cosine similarity evaluates the fidelity of the spectral reconstruction itself. It effectively measures the alignment between the predicted and target mel-spectrograms in a high-dimensional space. It allows us to focus on the relative patterns of energy distribution, which are crucial for assessing perceptual similarity in audio. Capturing the angular similarity between the vectors, cosine similarity values range from -1 to 1, where 1 indicates perfect alignment between the vectors, 0 indicates orthogonality, and -1 indicates complete dissimilarity.

We calculated the cosine similarity between the reconstructed and target mel-spectrograms for each data point and then averaged these values to produce a single metric for model performance. As shown in Table 3, our best-performing model achieved a mean cosine similarity of **0.80**, which translates to **90%** accuracy when normalized. This result demonstrates the robustness of our approach and highlights cosine similarity as a more nuanced and informative metric compared to discrete classification.

The highest-performing RCNN model achieved a cosine similarity of 0.7392, slightly outperforming the best encoder-decoder model, which reached a cosine similarity of 0.7302. Both models fall short of the performance seen with simpler architectures like our baseline CNN, suggesting that either the added complexity does not necessarily translate to better outcomes for this task, or that the level of complexity explored so far may still be insufficient, leaving room to investigate more powerful architectures that could better capture the intricate patterns in the data.

Table 3: Cosine Similarity accross Different Architectures

Model	Cosine Similarity (-1.00,1.00)
CNN	**0.80**
RCNN	0.74
Encoder-decoder	0.73

The Transformer approach, while powerful for many sequential tasks, did not yield optimal results in our case due to the strong reliance on teacher forcing during training, which hindered the model's ability to generalize in real-world autoregressive scenarios. While the model achieved high cosine similarity in reconstructing songs when ground truth outputs were provided at each decoding step, this does not count as success because it does not translate to scenarios without such inputs. The discrepancy between training and inference conditions led to error accumulation and degraded performance during autoregressive decoding, especially over longer sequences. Additionally, the lack of explicit temporal dependencies in EEG data and the relatively small dataset size likely limited the transformer's ability to fully leverage its self-attention mechanism. These factors, combined with the challenge of predicting fine-grained Mel spectrograms, contributed to the model not performing well enough.

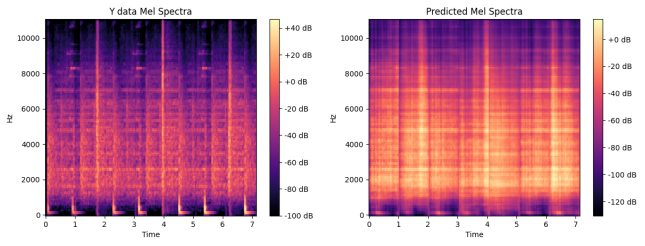

Figure 1: Target vs. Reconstructed Mel-Spectrograms from EEG Data.

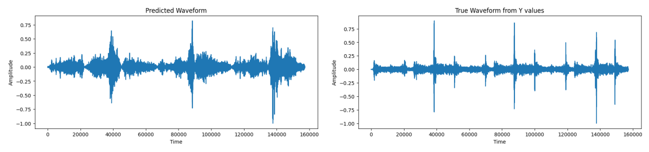

Figure 2: Target vs. Reconstructed Audio Waves

4.3 Computational resources

3 team members utilized 3 Google Collab Pro accounts with NVIDIA T4 GPU - High RAM with a significantly increased amount of RAM to process the large dataset and complex models that required more memory. 3 Google Collab Pro accounts combines used over 425 computational units and trained over 215 hours total. 4th team member utilized their personal laptop with Apple M3 Max Chip and 32GB of RAM where they trained over 60 hours.

5 Conclusion

In this paper, we explored the reconstruction of musical stimuli from EEG responses by building upon the methods introduced in the EEG2Mel study and extending them with advanced models[1][4]. Although we were able to improve the baseline accuracy slightly and produce comparable accuracies with more complex models and techniques, these more advanced model's performances did not exceed our best CNN performance.

References

[1] Ramirez-Aristizabal, Adolfo G. and Christopher T. Kello. "EEG2Mel: Reconstructing Sound from Brain Responses to Music." ArXiv abs/2207.13845 (2022): n. pag.

[2] Steven Losorelli, Duc T. Nguyen, Jacek P. Dmochowski, and Blair Kaneshiro (2017). Naturalistic Music EEG Dataset - Tempo (NMED-T). Stanford Digital Repository.

[3] Steven Losorelli, Duc T. Nguyen, Jacek P. Dmochowski, and Blair Kaneshiro (to appear). NMED-T: A Tempo-Focused Dataset of Cortical and Behavioral Responses to Naturalistic Music. In Proceedings of the 18th International Society for Music Information Retrieval Conference, Suzhou, China.

[4] M.-A. Moinnereau, T. Brienne, S. Brodeur, J. Rouat, K. Whittingstall, and E. Plourde, "Classification of auditory stimuli from eeg signals with a regulated recurrent neural network reservoir," arXiv preprint arXiv:1804.10322, 2018

[5] A. G. Ramirez-Aristizabal, M. K. Ebrahimpour, and C. T. Kello, "Image-based eeg classification of brain responses to song recordings," arXiv preprint arXiv:2202.03265, 2022.

[6] Zhang, B., Leitner, J., Thornton, S. (2019). Audio Recognition using Mel Spectrograms and Convolution Neural Networks.

[7] Vaswani, Ashish et al. "Attention is All you Need." Neural Information Processing Systems (2017).