ECG Classification using CNN and Transformers

Mike Koscak
Georgia Institute of Technology
mkoscak3@gatech.edu

James Scanlon
Georgia Institute of Technology
jscanlon8@gatech.edu

Kyle Seaman
Georgia Institute of Technology
kyle.seaman@gatech.edu

Steven Sesterhenn
Georgia Institute of Technology
ssesterhenn3@gatech.edu

Abstract

Electroencephalograms (EEG) is a method for recording electrical activity of the brain. Using deep machine learning for the interpretation of EEG signals to determine specific brain activity has shown promise in previous studies. This project uses deep machine learning for the interpretation of EEG signals by combining several methods of preprocessing with the CNN and Transformer neural network models that has shown success in other EEG applications.

Using two MindBigData 2022 datasets, a Transformer and two custom CNN models showed strong learning in the Cap64 data set but achieved only 10% accuracy in validation on a difficult classification task. However, the Transformer model was able to achieve the same accuracy results as the CNN model in a small fraction of the training time. Using the Muse dataset with a revised CNN model, training on an easier classification task, the project was able to achieve a higher 84% accuracy.

1. Introduction/Background/Motivation

1.1. Introduction

It is said that the brain, with its 100 billion neurons and trillions of connections between them, is a "final inner frontier" of understanding of the how the brain learns and creates human consciousness. Our current tools for modeling and exploration are rudimentary both due to the complexity of the brain's network, but also due to the limits of existing technology to "read" different brain activities in real-time, like emotions, seizures, sight, speech, imagination and sleep stages.[1]

Electroencephalogram (EEG) data offers a unique glimpse into the neural activities of the brain. The actual electric potential of the brain's billions of neurons, known as EEG signals, can be detected non-invasively and with variable numbers of sensors located physically around a subject's head (spatial resolution) and varying snapshot rates (temporal resolution). By developing deep learning models that can predict classifications from EEG signals, we can facilitate a better understanding of brainwaves. With this better understanding of EEG signals, there exists an opportunity to develop Brain-Computer Interfaces (BCIs) that can translate these signals into actionable information. [6]

The MindBigData project provides a substantial dataset of EEG signals to help further the development of decoding brain activities. In the scope of this project, we worked with MindBigData's Visual MNIST dataset, which captures brain activity as subjects view handwritten digits. Our project aims to explore various deep learning methods to analyze and classify these EEG signals. This exploration includes implementing both established Convolutional Neural Network (CNN) models and more recent Transformer architectures to compare their effectiveness in this domain.

1.2. Background and Motivation

Our primary goal was to determine if EEG reading data could be used to classify thoughts of subjects based on a number they were shown. Previous research has demonstrated significant advancements in the classification of EEG signals using deep learning techniques, particularly CNNs. For instance, the study titled "EEG Signals to Digit Classification Using Deep Learning-Based One-Dimensional Convolutional Neural Network" successfully utilized a one-dimensional CNN model to classify EEG signals representing digits, achieving high accuracy rates. This success suggests that deep learning models can effectively interpret EEG signals to identify specific thoughts or stimuli viewed by the subject. Extending this capability could provide significant benefits, such as enabling quadriplegic patients to control devices through thought alone, thus vastly improving their quality of life.

Today there are good results however most approaches

require heavy pre-processing and are for very specific classification tasks. This results in non real-time use cases or very specific applications that require extensive training. We hope to add research to this field and continue to move the topic towards implementations that can be trained to a specific subject faster and create models that evaluate in near real-time.

1.2.1 Why CNNs Have Been Successful

CNNs are well-suited for this problem due to their ability to capture spatial hierarchies in data. In the context of EEG signals, CNNs can efficiently learn local patterns and temporal dependencies through convolutional filters. The hierarchical feature extraction mechanism of CNNs helps in identifying intricate patterns in EEG signals that are essential for accurate classification. Additionally, CNNs have been successfully applied in numerous EEG classification tasks, proving their effectiveness in handling the complex and noisy nature of EEG data.

1.2.2 Benefits of Transformers

Transformers, known for their self-attention mechanisms, can bring several advantages over CNNs. The self-attention mechanism allows Transformers to consider the entire sequence of EEG signals simultaneously, capturing long-range dependencies more effectively than CNNs. This can be particularly beneficial for EEG data, where important features might be spread across the entire signal.

1.3. State of the Art

Interestingly enough, the best performing model in EEG classification uses a fusion model of a transformer and a CNN, using multi-head self-attention and temporal convolutions. This model yielded an accuracy of 94.20% in EEG classification tasks. [3] Without the use of transformers, the best performing model used a four layer CNN along with an Infinite Impulse Response Butterworth Filter to suppress higher frequency elements in the EEG data.[6] These models demonstrated that CNNs and transformers were the best place to start for EEG data and influenced our decisions in development.

1.3.1 Dataset Choice

The team selected a primary dataset and after experimentation used a second dataset for additional exploration. The reason for multiple datasets is detailed in section 2 Approach. The first dataset, "MindBigData2022" Cap64 data set, available on Hugging Face was used to start. The existing research leveraged similar datasets that were collected with different visuals or devices but followed the same structure. Each of these datasets collects channels of voltage readings of the brain for a period of time. We selected this dataset because it was different from previous explorations and we believed applying the approach to a new dataset could help move the field forward.

The Cap 64 dataset collected 2 seconds of data at 200 hz which gives 400 voltage readings represented as integers. These 400 samples were collected across 64 channels. This results in one record having 25,600 readings total. For the Cap64 dataset there were 1,860 training records and 471 test records. There was additional metadata included in the dataset such as information about the pixels of the image the subject was shown however this was removed because our analysis only required the EEG reading and the label. The labels were the 0-9 digit the user was shown or a -1 representing a blank screen.

In addition to the Cap64 dataset the team further explored EEG data by using the MNIST MU dataset. This used a Muse headset for collection. In contrast to the Cap64 dataset this took readings at 220 hz for 2 seconds giving 440 voltage readings per channel. The Muse headset only had 4 input channels so each record consisted of 1,760 readings. This dataset had 32,800 training rows and 8,200 test rows.

1.3.2 Motivation and Benefits

We believe that successful classification of EEG data to specific thought patterns can create exciting and beneficial outcomes. Extending this capability could yield significant benefits, such as enabling quadriplegic patients to control devices through thought alone, greatly enhancing their quality of life. Beyond medical applications, this technology holds the potential to transform various fields. In assistive technology, it could empower individuals with severe motor impairments to interact with computers and smart devices, increasing their independence and communication capabilities. In the gaming and entertainment industries, EEG-based control systems could create more immersive and intuitive experiences. Additionally, in industrial environments, thought-controlled interfaces could improve efficiency and safety by enabling hands-free operation of machinery. These advancements could have widespread societal impacts, promoting inclusivity and accessibility across numerous sectors.

2. Approach

Our approach involved several key steps, starting with data pre-processing, followed by model implementation, and ending with training and evaluation. The team found that both CNN and Transformer models did not show learning or generalization on the originally chosen dataset, which is detailed in section 3 Experiments and Results. In the interest of learning and exploration the team then leveraged a second dataset to compare approaches and results. This

approach section is broken into two main sections, one for each dataset. All model development and code execution was done leveraging Python and Pytorch.

All code is contained in the following GaTech GitHub repository: Link to Repository.

2.1. MindBigData2022 VisMNIST Cap64

Our first dataset was the Visual MNIST Cap 64 dataset available at HuggingFace. The Visual MNIST dataset from the MindBigData archive contains EEG signals captured while subjects viewed the handwritten digits (0-9) from the original MNIST dataset. This dataset was collected using a custom built EEG cap that had 64 channels.[1]

2.1.1 Data Pre-processing

The dataset included raw EEG signals from the 64 channels, with a significant portion labeled as -1, indicating no digit was shown. As mentioned in section 1.2.3 each record in the dataset had 64 channels of readings with exactly 400 readings each. To improve class balance and focus on digit recognition, we removed these -1 labels, which constituted about 50 percent of the data. This step ensured that our dataset was balanced and focused on the task of digit classification.

The raw EEG signals pick up a lot of noise that has nothing to do with the classification task. Things like your heartbeat, muscle movement, and eye movement all contribute to the signal being measured. These innate activities makes it important to apply noise reduction and feature extraction techniques in order to accurately interpret the brain activity related to the classification task.

For noise reduction, we applied a technique called "filtering", which splits the data into frequency bands. This allows us to interpret specific frequencies of the brain signals that we may be interested in. In our experiments, we applied an alpha (8-12Hz) and beta (12-30Hz) filter on our data. These bands are usually associated with active thinking. Low frequencies add noise to the data, making it harder to generalize.

For the feature extraction, we applied a continuous wavelet transform (CWT). This transformation captures the time and frequency of the data, unlike the Fourier transform. The CWT extracts features very well, especially when applied to specific frequency bands. Another advantage to CWT is that it outputs a scalogram, making it a great feature extraction tool to use on a CNN, given CNN's known success with image classification.[6] Once pre-processing was completed we then implemented and compared both CNN and Transformer models.

For the Cap64 dataset, the training data was separated into 90% for training and 10% for validation.

2.1.2 1-D CNN Model Implementation

The architecture of our 1D CNN model, shown in Table 1, was chosen based on the success reported by Tiwari et al. on another dataset[5]. A few other architectures with less convolution layers were also attempted however the resulting output was similar and so for brevity those are left out from this discussion.

Layer Type	**Parameters**
Input	*EEG signal*
Convolution 1D	256 filters, kernel size 7, stride 1, padding 3
Batch Norm	Applied after Conv1D
Activation	ReLU
Max Pooling	Kernel size 2, stride 2
Convolution 1D	128 filters, kernel size 7, stride 1, padding 3
Batch Norm	Applied after Conv1D
Activation	ReLU
Max Pooling	Kernel size 2, stride 2
Convolution 1D	64 filters, kernel size 7, stride 1, padding 3
Batch Norm	Applied after Conv1D
Activation	ReLU
Max Pooling	Kernel size 2, stride 2
Convolution 1D	32 filters, kernel size 7, stride 1, padding 3
Batch Norm	Applied after Conv1D
Activation	ReLU
Max Pooling	Kernel size 2, stride 2
Dropout	60 percent
Fully Connected	128 neurons
Activation	ReLU
Fully Connected	64 neurons
Activation	ReLU
Fully Connected	Number of classes
Activation	Softmax

Table 1. 1-D CNN Model

2.1.3 Transformer Model Implementation

The architecture of the Transformer model is detailed in Table 3. This model is a straightforward implementation of a Transformer Encoder, designed to classify a sequence of EEG sensor inputs. It is based directly on the design by Vaswani et al. [2]. Each sensor's value is used as the input feature, with each snapshot of the sensors utilized as a separate entry in the sequence, in the order they were collected. The EEG data, shaped as (batch size, num sensors, sequence length), is fed directly into the model, where positional encoding is added as detailed by Vaswani et al. [2]. The data then passes through a single self-attention

layer with two heads, followed by two fully-connected linear layers with a ReLU activation function and dropout in between. This output is processed through another dropout layer and then batch-normalized, with the mean of the output sequence taken to produce raw predictions for a single classifier value for the input sequence. The softmax function is then applied to obtain the output probabilities for the different classes. This final output is used for loss calculation during training via back-propagation and for making classifier predictions from the model.

Layer Type	**Parameters**
Input	*EEG signal*
Pos Embedding	Sin/Cos
Self Attention	256 neurons, 2 Heads, 0.5 Dropout
Fully Connected	input 256, output 2048 neurons
Activation	ReLU
Drop Out	0.5
Fully Connected	input 2048, output 256 neurons
Drop Out	0.5
Batch Norm	2048 neurons
Mean	Applied to Sequence Dimension
Fully Connected	input 2048, output 10 neurons
Activation	Softmax

Table 2. Transformer Model

2.2. MindBigData2022 MNIST MU

The results from training and testing the model described in Section 2 did not meet the accuracy metrics anticipated, nor did they align with those observed in prior research. An in-depth analysis of these results, along with potential causes for the discrepancies, is presented in Section 3. To further investigate, we tested another dataset, MindBigData2022 MNIST MU, available from HuggingFace.

2.2.1 Data Pre-processing

As outlined in Section 1.2.3, the MNIST MU dataset contains significantly more records than the Cap64 dataset, with 32,800 training rows and 8,200 testing rows. This dataset features consistent readings taken over 2-second intervals at a frequency of 220 Hz, using a 4-channel cap. Similar to the Cap64 dataset, any extraneous fields, such as pixel information, were removed, leaving only the EEG readings and labels.

However, unlike the Cap64 dataset, we introduced an additional pre-processing step: grouping the data into 'digit' and 'non-digit' categories. This decision was motivated by the poor digit classification results observed with the Cap64 dataset and the fewer channels available in the MNIST MU dataset. We hypothesized that identifying whether the subject had been shown a digit or a blank screen might be more feasible than classifying each digit individually. This approach leverages the notion that different parts of the brain might activate when processing integers versus non-integers, potentially allowing for easier classification between the two categories.

After preparing the data, we ran two experiments using the model detailed in Section 2.2.2. One experiment was conducted without pre-filtering the data, while the other included a pre-filtering step. In both scenarios, a wavelet transformation was applied to the data.

Figure 1 illustrates the results of the wavelet transformation for both the unfiltered (left) and filtered (right) EEG reading records. The unfiltered data (left image) exhibits less distinct patterns and a generally lower intensity across the spectrum, indicating more noise and less clarity in the signal. In contrast, the filtered data (right image) shows more defined patterns with higher intensity areas, suggesting that the pre-filtering step successfully reduced noise and enhanced the signal's features. This enhancement in the filtered data is crucial for improving the accuracy of subsequent analysis and classification tasks. The comparison highlights the importance of pre-filtering in obtaining clearer and more informative wavelet transformations of EEG data.

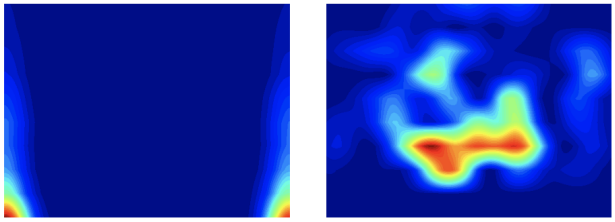

Figure 1. Sample Wavelet Transformation Unfiltered (Left) and Filtered (Right) of EEG Reading Record

2.2.2 2D CNN Model Implementation

For this experiment a 2D CNN model was used, the model's architecture is outlined in Table 3. The biggest difference here is a two-dimensional CNN was employed versus the one-dimensional CNN model that used with the Cap64 dataset, in order to possibly take advantage of relationships between the fewer number of EEG sensors used in the Muse dataset.

3. Experiments and Results.

3.1. Cap64 Dataset with 1D CNN and Transformer

3.1.1 Cap64 1D CNN Expirments and Results

Our experiment testing the Cap64 dataset using the 1D CNN outlined in Sections 2.1.1 and 2.1.2 yielded the results shown in Figure 2. It is apparent from the loss and

Layer Type	Parameters
Input	*EEG Signal*
Convolution 2D	64 filters, kernel size 3, stride 1, padding 1
Activation	ReLU
Batch Norm	Applied after Conv2D
Max Pooling	Kernel size 2, stride 2
Convolution 2D	32 filters, kernel size 3, stride 1, padding 1
Activation	ReLU
Batch Norm	Applied after Conv2D
Max Pooling	Kernel size 2, stride 2
Convolution 2D	32 filters, kernel size 3, stride 1, padding 1
Activation	ReLU
Batch Norm	Applied after Conv2D
Max Pooling	Kernel size 2, stride 2
Fully Connected	Flattened size to 64 neurons
Activation	ReLU
Fully Connected	64 neurons to 32 neurons
Activation	ReLU
Fully Connected	32 neurons to 2 neurons

Table 3. 2D CNN Model

accuracy curves that the model is overfitting and failing to generalize.

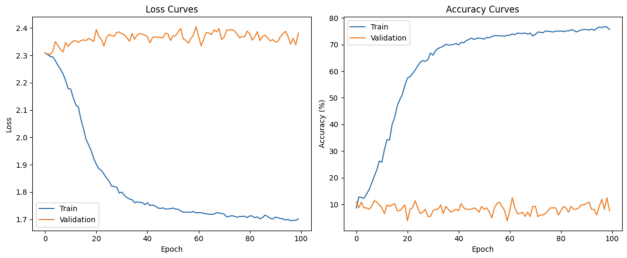

Figure 2. CNN 1D Results

The loss curves indicate that the training loss consistently decreases over the epochs, suggesting that the model is learning from the training data. However, the validation loss does not follow the same trend; instead, it remains relatively high and even increases after an initial decline. This divergence between training and validation loss is a clear sign of overfitting, where the model performs well on the training data but fails to do so on unseen data.

Similarly, the accuracy curves show a significant gap between training and validation accuracy. While the training accuracy steadily improves, reaching around 75 percent, the validation accuracy fluctuates and remains low, around 10 percent. This further confirms that the model is memorizing the training data rather than learning generalizable features.

The 1D CNN model took about 16.5 minutes to run the full 100 training epochs, with the validation score never having a clear upward trend.

3.1.2 Cap64 Transformer Experiments and Results

The experiment using the Transformer model detailed in section 2.1.3 against the Cap64 dataset yielded the results shown in Figure 3.

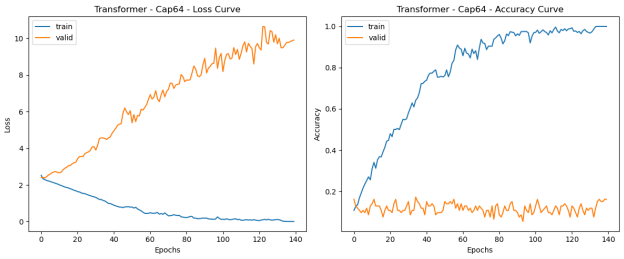

Figure 3. Transformer Cap64 Results

Similar to the CNN 1D model against the Cap64 dataset, the loss and accuracy curves show steady improvement over the 140 training epochs, while the loss for the validation continues to increase and the accuracy remains roughly the same. The model is able to fit to nearly 100 percent of the training data, showing that the Transformer model was complex enough to memorize the entire training set.

Various self-attention hyper-parameters (dropout of 0.1 to 0.5, neurons of 256 to 1024, heads of 2 to 4, and layers of 1 to 3) were experimented with, but the results were no better with the more complex models, so the simplest was chosen. As the simplest model was able to fit to 100% accuracy of the training data, this makes sense.

The transformer took just 53 seconds to run the full 140 training epochs, but the best validation loss of 2.3868, achieved at only epoch 3, was a tiny fraction of that total time. Overall, the model took just a few seconds to produce it's best training result.

3.1.3 Comparing Transformer and 1D CNN

Through prior research, 1D CNN models were shown to be successful in the task of classifying EEG signals recording visuals of numeric digits. While attempting to duplicate those previous efforts, this project sought to accomplish the same or better result using a Transformer model. While we weren't able to duplicate the classification results of other work, compared to the 1D CNN, the Transformer model was able to match the validation accuracy of the 1D CNN, learn significantly faster (53 seconds for maximum overfit versus 16.5 minutes) and effectively memorize the entire training set better (100% versus 80%) in a smaller model (2.67M versus 3.17M trainable parameters). This validates the potential of the Transformer model for this task, if we can get around the problems discovered with the data sets.

3.2. Muse Dataset with 2D CNN Experiments and Results

The experiment using the 2D CNN model detailed in section 2.2.2 against the Muse dataset yielded the results shown in Figure 4.

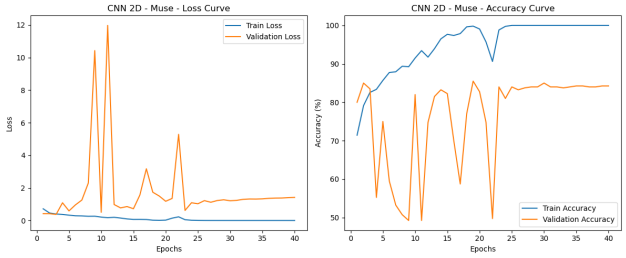

Figure 4. CNN 2D Muse Results

The CNN 2D model took 362 seconds to train the full 40 epochs shown in the graph. However, the training data achieved near 100% accuracy along with the converged best validation accuracy of 84% at training epoch 23.

3.3. Dataset and Results Comparison

The team believes the success of the second dataset is more due to the dataset itself versus the model structure. Because the Cap64 dataset was measured with 64 channels, it captures much more activity than just the classification task we want to generalize. Measuring signals with more channels gives richer and noisier data, which can both enhance and challenge the classification task. With more channels, there are richer features that can be extracted from the data. However, this makes it much more difficult generalize a specific task due to the complexity of brain signals. In order to take advantage of the large number of channels, more complex feature extraction techniques are needed, and possibly in combination with dimension reduction techniques. Prior work has shown success using a VAE architecture to reduce the dimensional space while preserving the meaningful features of the EEG signals [4]. We believe our model overfits on the Cap64 dataset because it suffers from the curse of dimensionality. A more complex data pre-processing technique is needed to handle the higher dimensionality.

The Muse dataset on the other hand only has 4 channels. This makes the data much more manageable to pre-process. While this data still has some noise, it can easily be removed by applying filtering on specific frequency bands. The results in Figure 4 demonstrate that CWT is able to extract meaningful features from this dataset. Although this dataset is much simpler than Cap64, it allows for more success on simpler classification tasks. The difference in results between the two datasets also demonstrate Occam's Razor. There is value in simpler datasets when it is sufficient to solve the classification task.

4. Summary and Future Work

The project was able to create 1D CNN and Transformer models to process the Cap64 dataset, which were able to train well but did not generalize in validation data. A 2D CNN model was able to train with the Muse dataset, which had fewer data dimensions, and was able to very successfully classify a simple goal with 84% accuracy in validation.

While this project initially attempted to use the Cap64 dataset, which has 64 channels and possibly suffered from too many dimensions as a function of the number of samples in the training data, attempting to use earlier datasets with fewer features would be a first place to begin again.

This project attempted several ways to overcome overfitting through pre-processing of the Cap64 data, but there are additional techniques that could be tried, such as Principle Components Analysis (PCA), to reduce the number of dimensions. In addition, augmenting the existing Cap64 data using standard and novel techniques for EEG would be another exploration area that would add training data and help with learning.

Finally, there was not time to try the Muse dataset test with reduced classification complexity goal using the Transformer model. Demonstrating the Transformer advantages against this working classification task would be another task for future work.

5. Work Division

The delegation of work by the team members is detailed at the end of the report in Table 4.

References

[1] Felix Cuesta David Vivancos. Mindbigdata 2022 a large dataset of brain signals. 27 Dec 2022. 1, 3

[2] Ashish Vaswani et al. Attention is all you need. 2 August 2023. 3

[3] Hadi Seyedarabi Mahsa and Reza Afrouzian. Classification of eeg signals using transformer based deep learning and ensemble models. *Biomedical Signal Processing and Control*, 86A(105130). 2

[4] Samuele Russo Christian Napoli Salvatore Falciglia, Filippo Betello. Learning visual stimulus-evoked eeg manifold for neural image classification. *Science Direct*, 2024. 6

[5] Andrzej Cichocki Shu Gong, Kaibo Xing and Junhua Li. Deep learning in eeg: Advance of the last ten-year critical period. 20 May 2021. 3

[6] Shivani Goel Smita Tiwari and Arpit Bhardwaj. Eeg signals to digit classification using deep learning-based one-dimensional convolutional neural network. *Arabian Journal for Science and Engineering*, 48:9675–9691, 2023. 1, 2, 3

Student Name	Contributed Aspects	Details
Mike Koscak	Data Creation and Implementation	Assisted with dataset downloading code and preprocessing. Developed code within Mike Branch of repository. Key files include updates to main.py, and the various model classes such as eeg-cnn-model.py, eeg-preprocessing.py, model.py, and others. Code included data preprocessing and creation of 1D CNN (included in report) and additional testing of 2D CNN and transformer (unsuccessful and not included in report). Tuned hyperparameters for each of these models and created artifacts outlining the performance of the 1D CNN. Mike assisted in writing the introduction, approach and experimental results section. Specifically from the report sections 2.1.2 and 3.1.1 are directly related to Mike's model implementations.
James Scanlon	Implementation and Analysis	Researched and experimented with existing models from papers. Built on and did additional parameter testing for the CNN model in Section 2.1.2. Researched and contributed to development of CNN models and conducted research on exisiting successful models, found in main-Notebook.ipynb. Assisted in the final report introduction and reviewed final report.
Kyle Seaman	Data collection, pre-processing, and feature extraction	Implemented all the data collection, data pre-processing and feature extraction code (see preprocessing-kyle.py), see section 2.1.1. Produced results for the spectrogram shown in Figure 1. Researched all prior work and different pre-processing and feature extraction techniques done in prior work. Implemented the 2D CNN and training loop (see cnn.py), which produces the results found in Section 2.2.2 and section 3.2. Contributed to the analysis between datasets found in section 3.3.
Steven Sesterhenn	Model Implementation and Analysis, Final Report	Researched project background. Created and trained Transformer model using Cap64 data, per section 2.1.3, contained in main code branch as main-steve-transformer.py and preprocessing-steve.py. Tuned hyperparams of Transformer, analyzed results, and created report artifacts, per section 3.1.2. Created a 2D CNN model for Cap64 and Transformer model for Muse, both unsuccessful and not included in report. Contributed to final report introduction, approach, experimental results and conclusion sections.

Table 4. Contributions of team members.