Reading the Mind with Artificial Intelligence: Towards a Practical, Versatile Brain-Computer-Interface for Imagined Speech Decoding

AssistEEG V0

Research Paper V0

Nithin Naikar
Olentangy Liberty High School
Powell (O.H.), U.S.
nithin.naikar@gmail.com

***Abstract*— Around 2 million Americans suffer from neuromuscular disorders. These disorders interfere with communication and control of the external environment by disrupting neuromuscular pathways. In severe cases, patients may experience complete "locked-in" syndrome, which makes natural communication impossible. Brain-computer interfaces (BCIs) have emerged as a potential solution for communication. However, traditional BCIs such as P300 speller BCIs or point-and-click typing BCIs rely on residual muscle control and are not feasible for completely "locked-in" patients, while also possessing a low word-per-minute rate and unintuitive control strategy. This project aimed to progress towards a practical and effective BCI for communication by creating a classification system for imagined speech decoding and experimenting with a variety of classification modes. To achieve this, neural electroencephalography (EEG) recordings from the publicly available KARA ONE dataset were preprocessed and denoised. Relevant features were extracted, and the dimensionality of the feature vector was reduced through the Analysis-Of-Variance algorithm based on their correlation to output classes. T-distributed-stochastic-neighbor-embeddings were computed to visualize the data in a low-dimensional embedding and examine inter-class discriminability for each classification task. Two classification models, K-Nearest Neighbors (KNN) and Multilayer Perceptron (MLP), were implemented and optimized through grid search. All mean testing accuracies reported for both the KNN and MLP were above chance level, with some cases achieving significantly above chance level. The results demonstrated that EEG signals may encode some correlation with imagined speech, paving the way for future studies in imagined speech decoding to be implemented and the development of a practical BCI for communication.**

Keywords—Brain-Computer Interface, Electroencephalography, Imagined Speech, Artificial Intelligence

I. INTRODUCTION

Approximately two million people in the U.S, and far more globally, suffer from neuromuscular disorders, including Amyotrophic Lateral Sclerosis (ALS), brain stem strokes, brain or spinal cord injuries, cerebral palsy, muscular dystrophies, and multiple sclerosis [1]. These disorders disrupt the neuromuscular pathways through which the brain communicates and controls its

external environment. Those severely affected by these disorders may lose nearly all voluntary muscle control, including those responsible for communication. This loss of control typically leads to a “locked-in” state, where natural communication is difficult or outright impossible. This state is also called “locked-in” syndrome, or LIS, of which the famous physicist Stephen Hawking suffered from due to ALS [2]. For patients with mild to moderate LIS, eye or hand movement-based systems that can be used to answer questions, give simple commands, or even operate word processing programs have been built [3]. These systems rely on the precondition that patients have some residual (left-over) muscle control remaining, namely the hands or eyes. However, in the case of severe, or complete LIS, where there is virtually no muscle control remaining, this precondition is not met. In fact, to the best of my knowledge, there is no device able to provide both practical and effective communication for a completely “locked-in” ALS patient.

Brain-computer interfaces (BCI's) have gained significant attention as a potential solution for individuals who have limited or no control over their neuromuscular pathways [1]. BCIs establish a direct connection between a person's brain and a computer, providing a way to interact with the environment without relying on traditional physical movements [1]. This technology is based on the idea that the brain generates distinct electrical signals that correspond to different mental states, which can be recorded and interpreted. Various recording modalities such as Electroencephalography (EEG), which will be the main focus in this study due to its relative practicality, Magnetoencephalography (MEG), functional Magnetic Resonance Imaging (fMRI), among others, can capture these neural signals [4]. The collected data is then preprocessed to eliminate noise and fed into a feature extraction algorithm, which identifies the relevant features in the signal. A classification algorithm is then used to classify the mental states based on the extracted features, providing the desired output. BCIs are essentially an extension of the user's intention and control, where the user must encode their intention into neural signals, and the BCI must decode those signals to produce the desired output [1]. These assistive devices have tremendous potential for those with disabilities, including completely "locked-in" LIS patients, and are rapidly evolving as an alternative communication method.

Traditionally speaking, BCI’s have been used for restoring lost function to individuals with severe motor disabilities. BCIs allow individuals to control external effectors, such as a computer cursor, robotic limb, or functional electrical stimulation device, through the capture of their brain activity. This is achieved by analyzing and decoding the user's neural activity into commands that control the external effector. Traditional BCI applications have focused on the control of robotic and prosthetic arms, which is one of the most challenging tasks for BCI researchers [5]. In this application, the user must decode high-dimensional and preferably real-time control commands from their neural activity. Achieving speed, accuracy, and reliability for real-world applications remains the major challenge for BCI-based robotic control [6] However, researchers continue to develop new solutions that may improve the BCI user experience with robotic effectors. Additionally, BCI technology has been used to allow control of an electric device for patients who are completely paralyzed, such as ALS patients, through the real-time analysis of oscillatory EEG components during hand movement imagination [7] These developments highlight the potential of BCIs as a tool for restoring lost function and improving the quality of life for individuals with severe motor disabilities.

In terms of communication, BCI’s are rapidly evolving as an alternative communication method for individuals who are unable to communicate effectively due to various neurodegenerative diseases, such ALS and LIS, and other related conditions. BCI-based communication methods rely on non-invasive or invasive techniques to record neural signals from the brain and decode them into corresponding computer-generated commands. Traditional BCI communication methods, such as the P300 speller BCI [8], require visual attention and gaze control, which makes them impractical for patients with impaired vision or lack of gaze

control. However, recent advances in BCI technology have enabled the development of gaze-independent BCI communication methods, such as the steady-state visually evoked potential (SSVEP) BCI [9], which relies on covert attention and has shown improved classification rates. The SSVEP-based BCI has been tested on healthy volunteers and patients with LIS, and demonstrated the feasibility of online communication with a covert SSVEP paradigm that is truly independent of all neuromuscular functions. Another BCI communication method is the neural point-and-click typing BCI [10], which allows individuals to control a computer cursor with neural signals from their brain and type using an on-screen virtual keyboard. This BCI has been successfully tested on an individual with incomplete LIS, demonstrating the first use of an intracortical BCI for neural point-and-click communication by an individual with incomplete LIS. However, traditional BCI communication systems are still limited by their unintuitive control methodology and limited upper-bound on their word-per-minute (WPM) rate.

Imagined speech is a growing field within the realm of brain-computer interfaces (BCI), which leverages the natural process of speech production in the brain to provide a more intuitive means of communication for individuals with motor disabilities. Compared to other traditional BCI systems, such as the P300 speller and point-and-click typing BCI, imagined speech has several key advantages, including a more seamless interaction between the user and the BCI system, and the potential for higher WPM rates. While EEG-based BCI systems for imagined speech recognition have faced challenges due to low signal-to-noise ratio [11], recent advancements in preprocessing, feature extraction, and classification techniques have led to more accurate and efficient systems. One of the primary benefits of imagined speech is its ability to provide individuals with motor disabilities, namely those with complete LIS, a new means of communication and independence. As such, there has been a significant amount of research focused on designing practical BCI systems for imagined speech recognition, including the selection of words to be imagined, the number of electrodes to be recorded, and the use of temporal and spatial filtering [12]. Although there is still room for improvement in terms of speed and accuracy, imagined speech BCI systems hold great promise for the future. Continued development and refinement may lead to even more practical applications beyond laboratory experiments, ultimately improving the quality of life for individuals with motor disabilities and other neurological conditions.

II. Methods

a. Dataset Collection

While open-sourced BCI datasets involving motor imagery and other traditional applications are relatively established, the same cannot be said for datasets involving newer paradigms, such as imagined speech. In this study, all possible online, pre-existing, and open-sourced datasets for imagined speech classification were considered. The KARA ONE dataset, published by researchers Shunan Zhao and Frank Rudicz at the University of Toronto's Computational Linguistics department, was chosen due to the dataset's relatively heavy involvement in imagined/inner speech, use of a wide range of phonemic/syllabic English prompts, and large number of participants. In particular, the dataset consists of EEG data corresponding to the imagined speech of 7 similar/dissimilar phonemes ('/diy/', '/iy/', '/m/', '/n/', '/piy/', '/tiy/', '/uw/') and 4 words ('gnaw', 'knew', 'pat', 'pot') collected from 12 participants, 4 females and 8 males, with a mean age of 27.4 years, recruited from the University of Toronto campus [13]. All participants were right-handed, had at least some post-secondary education, had no visual, hearing, or motor impairments, and had no history of neurological conditions or drug abuse [13]. While the data of many participants were contained in the dataset, I only chose to use the data of one randomly-selected participant in order to avoid issues arising from various cross-subject complications [14]. The dataset includes both EEG and facial information recorded by a Microsoft Kinect camera, however just EEG data was utilized. The EEG data were

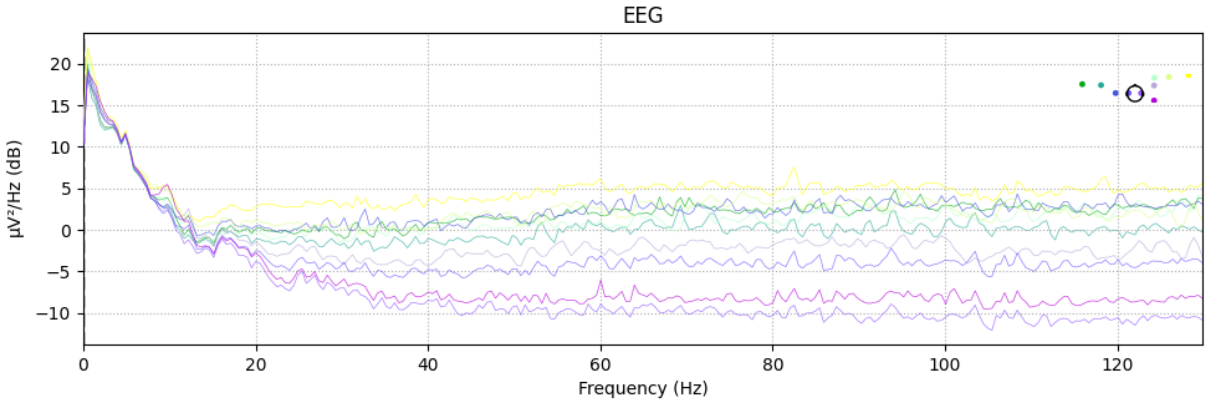
recorded using a 64-channel Neuroscan Quick-cap and sampled at 1 kHz. The study was conducted in an office environment at the Toronto Rehabilitation Institute, where the participants were presented with the 7 phonemic/syllabic prompts and 4 words derived from Kent's list of phonetically similar pairs. Each prompt was presented 12 times for a total of 132 trials, and the trials were randomly permuted within each of the two sections. The data were preprocessed to remove excessive noise in the speaking state EEG. Ethical approval was obtained from both the University of Toronto and the University Health Network, of which Toronto Rehab is a member. Overall, the KARA ONE dataset provides a valuable resource for studying speech production and its neural correlates.

b. Classification Tasks

In the original literature that proposed the KARA ONE dataset, several classification tasks were presented. These included different binary classifications such as vowel-only vs. consonant (C/V), presence of nasal (± Nasal), presence of bilabial (± Bilab.), presence of high-front vowel (±/iy/), and presence of high-back vowel (±/uw/), using six different modalities (including EEG, facial, and audio data) [13]. While these tasks provided valuable insights into the potential of decoding imagined speech from EEG, they also had some limitations, such as being restricted to binary classifications and involving multiple modalities, which is not necessarily scalable, computationally-efficient, or practical. To address these limitations and introduce some points of novelty, I decided to focus solely on EEG data for my classification tasks and use a variety of multiclass and binary classifications. I defined seven different modes of experimental variation (classification tasks), three of which were multi-class classifications while the rest were binary classifications. The first classification task I performed was a phoneme free for all, where I assigned each of the phonemes in the dataset to their respective class labels (resultant classes: '/diy/', '/iy/', '/m/', '/n/', '/piy/', '/tiy/', '/uw/'). The second task was a word free for all, where I assigned four words from the dataset to their respective class labels (resultant classes: ‘gnaw’, ‘knew’, ‘pat’, ‘pot’). The third task involved classifying between words and phonemes, where I assigned each word and phoneme to their respective class labels [resultant classes: phoneme ('/diy/', '/iy/', '/m/', '/n/', '/piy/', '/tiy/', '/uw/'), word (‘gnaw’, ‘knew’, ‘pat’, ‘pot’)]. The fourth task was a comprehensive free for all, where I assigned all the phonemes and words in the dataset to their respective class labels (resultant classes: '/diy/', '/iy/', '/m/', '/n/', '/piy/', '/tiy/', '/uw/', ‘gnaw’, ‘knew’, ‘pat’, ‘pot’). The fifth task classified between subsets of words, where I assigned two words to one subset and the other two words to the other subset [resultant classes: subset 1 (‘gnaw’, ‘knew’), subset 2 (‘pat’, ‘pot’)]. The sixth task classified again between subsets of words, where I assigned two words to one subset and the other two words to the other subset [resultant classes: subset 1 (‘gnaw’, ‘pot’), subset 2 (‘pat’, ‘knew’)]. The seventh task classified between subsets of phonemes, where I assigned two phonemes to one subset and two other phonemes to another subset [resultant classes: subset 1 (/diy/, /iy/), subset 2 (/m/, /n/)]. Overall, my approach represents a useful contribution to the field, as it allows for a more nuanced analysis of the data and provides new insights into the potential of decoding imagined speech from EEG. My multiclass and binary classifications provide a more fine-grained analysis of the data, and my focus solely on using only EEG data makes the results more practically applicable. The seven modes of experimental variation I defined also add a level of complexity and comprehensiveness to the analysis that was not present in the original literature.

c. Data Preprocessing

In this stage, EEG recordings from the KARA ONE dataset were processed to enhance the quality of the EEG data and remove any unwanted noise and artifacts present in the data. In other words, I aim to maximize the signal-to-noise (SNR) ratio in this step. EEG data is often contaminated with various sources of noise, including electrical interference from the environment, eye movements, and surrounding muscle activity. Such artifacts do not encode any of the user's intent and are irrelevant towards classification. Thus, removing such artifacts from the dataset signals is crucial to achieving high performance. A particular focus on light and computationally inexpensive preprocessing steps were employed in this section, beginning with the dropping of the following EEG channels due to their irrelevancy, as stated by the original authors of the dataset: 'CB1', 'CB2', 'VEO', 'HEO', 'EKG', 'EMG', 'Trigger'. Following this, out of the resultant set of channels, only a select few (ten) channels were selected for further analysis: "FC6", "FT8", "T7", "C5", "C3", "C4", "CP5", "CP3", "CP1", "P3, based on their involvement (correlation) with the speech task, as stated by the original paper [13]. These channels typically reside towards central locations and are generally situated around the auditory cortex. Following channel management, a Butterworth 4th order high pass filter with a lower bound of 1 Hz and a low pass filter with an upper bound of 50Hz was applied to the data to allow frequencies within the specified range to be present in the output signal, while diminishing frequencies outside of this range. While the range of the filter may seem broad, there does not currently exist a researcher consensus on the optimal frequency band of interest in imagined speech classification, unlike other modalities, such as motor imagery [12]. Utilizing this frequency band also acts as a notch filter at 60Hz (U.S.), which is useful in removing powerline noise. The raw signal vs the filtered signal is displayed in figure 1. For explicit removal of ocular artifacts, Independent Component Analysis (ICA) was initially proposed. ICA is a mathematical algorithm to separate a mixture of signals into their respective components. By using ICA, it is possible to reconstruct a denoised EEG signal by separating it from ocular noise components [15]. However, the removal of unwanted components involves manual iteration, which is far too time-consuming and beyond the scope of practicality. Automated ICA methods that utilize algorithms to remove unwanted components without human intervention have been proposed, but I decided to defer such advanced processing methods to a later study. Following denoising, data was epoched, or split into segments of interest, labeled with the appropriate mental state, and organized for further analysis.



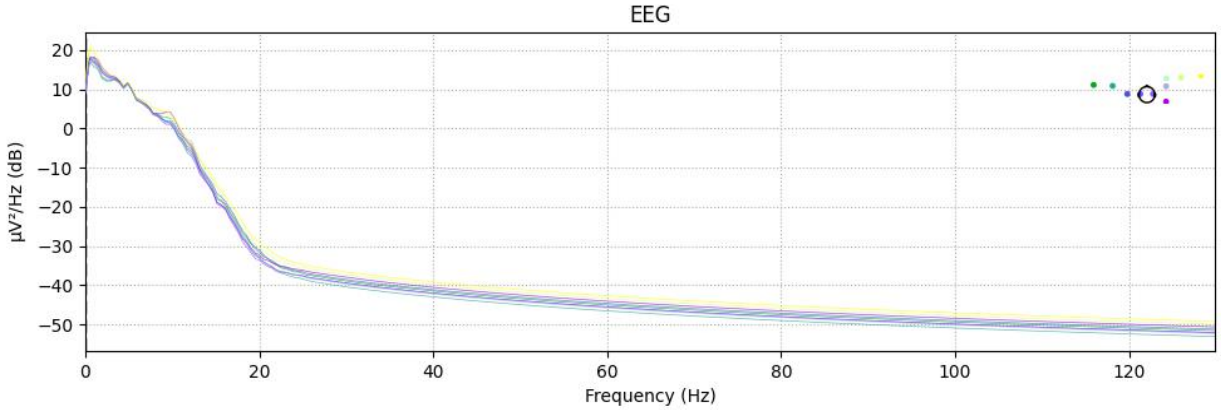


FIGURE 1: Raw EEG signal vs filtered EEG signal.

d. Feature Extraction

This stage mainly consists of extracting relevant and important features and patterns present in the EEG data that can distinguish between different brain states, or, in this case, different linguistical units. For the initial V0 prototype, a discrete set of features were computed, as mentioned in the initial KARA ONE dataset paper. In particular, following [25]'s feature extraction method, 24 scalar-valued, numerical features were extracted from the EEG recordings, including positive area, negative area, total area, total absolute area, amplitude (maximum signal value), latency, latency/amplitude ratio, absolute amplitude, absolute latency/amplitude ratio, peak-to-peak, peak-to-peak time window, zero crossings, peak-to-peak slope, zero crossings density, standard deviation, variance, mean signal value, mode signal value (rounded to 1 decimal), mode signal value (rounded to 2 decimals), mode signal value (rounded to 3 decimals), mean frequency, mode frequency, and median frequency, and referred to in a later part of the paper as 'par', 'tar', 'nar', 'taar', 'amp', 'lat', 'lar', 'aamp', 'alar', 'pp', 'ppt', 'zc', 'pps', 'zcd', 'std', 'var', 'mns', 'mes', 'md1s', 'md2s', 'md3s', 'mnf', 'mef', 'mdf', respectively. These features were computed for each of the ten classification channels, yielding a 240-sized feature vector for each recording.

However, such a feature vector can cause efficiency complications later on in the processing pipeline due to its size. To reduce the dimension of the feature vector, the Analysis of Variance (ANOVA) algorithm, as proposed by [16], was applied. ANOVA (Analysis of Variance) is a statistical technique that can be used to identify the most important features for the classification task. In this context, ANOVA is often used as a feature selection method, where a large number of features are initially extracted from the input data, and then only the most relevant features are retained for use in the classification model. ANOVA can be used to measure the statistical significance of each feature with respect to the target variable, which allows the user to rank the features in order of importance. By using ANOVA to select only the most relevant features, the dimensionality of the feature vector can be reduced, which can help to improve the performance of the classification model by reducing the risk of overfitting and improving the speed and efficiency of the learning algorithm. In particular, In ANOVA (Analysis of Variance), statistical tests are used to determine whether the variance between groups or samples is statistically significant or if it could have occurred by chance. The basic idea is to partition the total variability in a data set into different sources of variation, and then compare the size of these sources of variation

using statistical tests. In the context of feature selection for machine learning, ANOVA is used to determine which features are most strongly related to the outcome variable. The ANOVA F-test is applied to each feature separately to test whether there is a significant difference in the mean value of the feature between the different classes or groups in the outcome variable. The F-test produces a score for each feature, which reflects how much variance in the outcome variable is explained by that feature. After computing the F-scores for each feature, a threshold is set to determine how many of the top-scoring features should be retained. This threshold is usually determined based on the desired number of features or a desired percentage of the total variance explained. In this case, a threshold of $n = 30$ features was chosen to be retained in the output feature vector. The resulting feature set contains only those features that are most strongly related to the outcome variable and can be used for model training and prediction. The resultant $n = 30$ features for the first classification task are listed in the following matrix. The rows are ordered in descending order of F-values.

F-Value	P-Value	Channel Name	Feature Name
2.974	0.010	C5	mes
2.516	0.026	CP1	mnf
2.516	0.026	CP1	mdf
2.478	0.028	CP1	nar
2.466	0.029	C5	par
2.428	0.031	CP3	taar
2.428	0.031	C5	alar
2.416	0.032	CP5	mnf
2.416	0.032	CP5	mdf
2.409	0.033	C5	lat
2.396	0.033	CP3	std
2.394	0.034	CP1	std
2.364	0.036	CP1	pp
2.345	0.037	P3	taar
2.320	0.039	P3	std
2.300	0.040	CP3	pp
2.299	0.041	CP3	nar
2.298	0.041	CP1	var
2.216	0.048	C3	zcd
2.202	0.049	C5	tar
2.201	0.049	C5	mns
2.194	0.050	CP1	taar
2.172	0.052	CP3	var
2.163	0.053	P3	nar
2.162	0.053	P3	var

F-Value	P-Value	Channel Name	Feature Name
2.161	0.053	P3	par
2.118	0.058	CP3	par
2.104	0.060	FT8	amp
2.069	0.064	T7	mes
2.027	0.069	FC6	mns

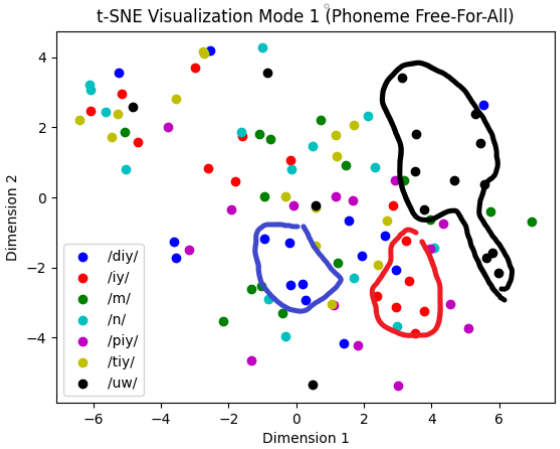
TABLE 1: ANOVA-produced top *n = 30* features for first classification task.

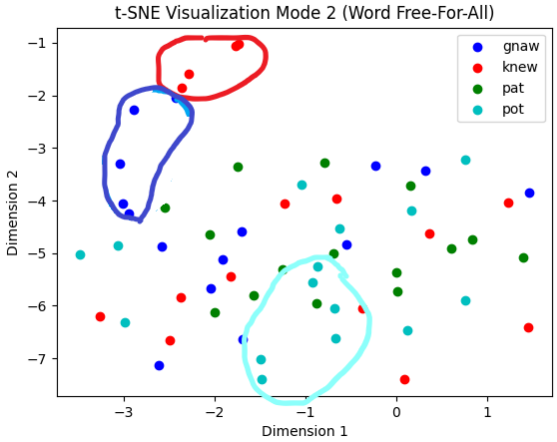
e. Data Visualization with t-SNE

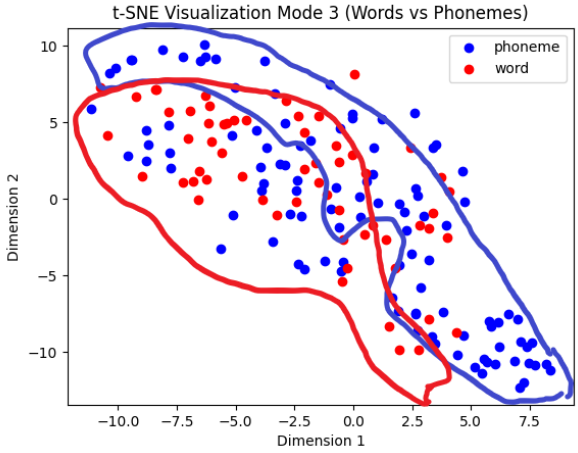
In the last step, feature extraction, my main objective was to obtain the most relevant and significant information that will aid in correctly classifying the neural signals. Once I had extracted these features, I used them to discriminate between the various linguistic categories. In order to get an idea of inter-class discriminability, I created a visualization using t-distributed stochastic neighbor embedding (t-SNE) [17]. t-SNE is a technique that visualizes high-dimensional data by giving each data point a location in a two or three-dimensional map. t-SNE is a relatively new, encouraging, and powerful technique that allows us to visualize high-dimensional data in a way that reveals the underlying structure and helps us to evaluate the effectiveness of our feature extraction process in discriminating between different linguistic categories. It is a variation of stochastic neighbor embedding (SNE) that is much easier to optimize and produces significantly better visualizations by reducing the tendency to crowd points together in the center of the map. t-SNE is better than existing techniques at creating a single map that reveals structure at many different scales. This is particularly important for high-dimensional data that lie on several different, but related, low-dimensional manifolds, such as images of objects from multiple classes seen from multiple viewpoints.

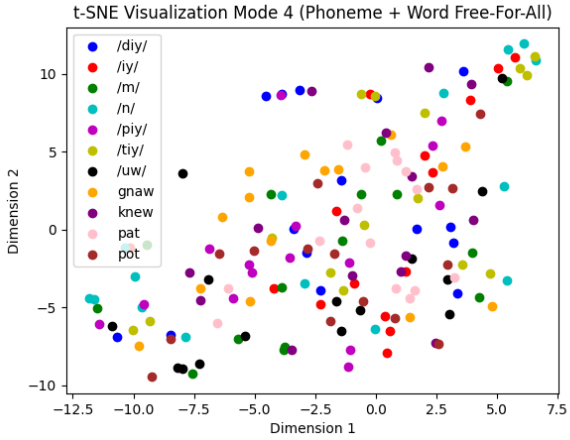
For my project, I created a 2D t-SNE visualization for every classification task. In particular, for every seven classification tasks presented in tables 1-7, I generated a 2D t-SNE visualization corresponding to that class. To use TSNE to generate a 2D visualization of a dataset, I started with a matrix of size n_samples x n_features, where n_samples is the number of data points, and n_features is the number of features (or dimensions) of each data point. TSNE is then used to reduce the dimensionality of the feature vector of each data point, while preserving its similarity to other data points. To achieve this, TSNE first constructs a probability distribution over pairs of high-dimensional points, such that similar points have a higher probability of being chosen than dissimilar points. This probability distribution is defined as a Gaussian distribution with variance proportional to the distance between the points in the high-dimensional space. This means that points that are close to each other in the high-dimensional space will have a higher probability of being chosen to form a pair. Next, TSNE constructs a similar probability distribution over pairs of points in the low-dimensional space (in this case, a 2D space), and minimizes the Kullback-Leibler divergence between these two probability distributions. This minimization process causes the low-dimensional map to represent the high-dimensional data in a way that preserves the local structure of the data, while also reducing the dimensionality. In other words, data points that are close to each other in the high-dimensional space will also be close to each other in the 2D space, while preserving the relative distances between data points. By using TSNE to create a 2D visualization, and then assigning sample points with colors corresponding to their class, we can gain insights into the inter-class discriminability of the data. This technique has been

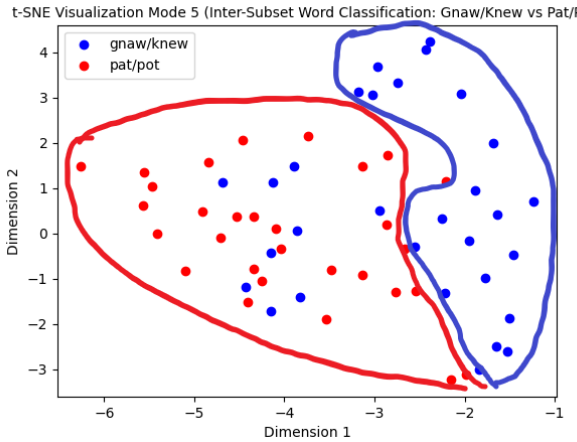
used in various applications such as image recognition, text mining, and neuroscience research, where high-dimensional data needs to be visualized in a way that preserves the local structure of the data. For example, Willett et al. (2021) used TSNE to visualize the neural activity patterns associated with imagined handwriting and demonstrated its effectiveness in examining inter-class discriminability. In the following figures, the produced 2D t-SNE visualizations along with annotated clusters are shown.

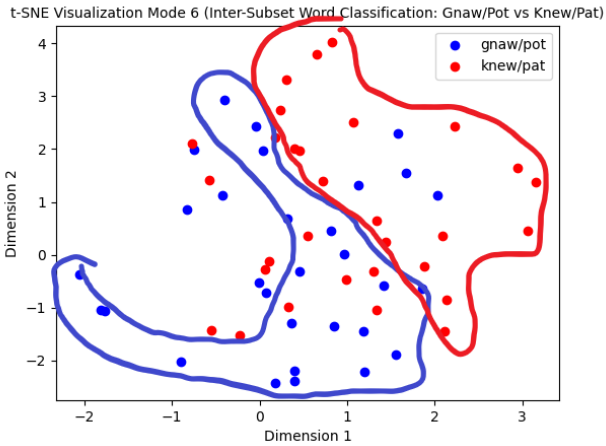












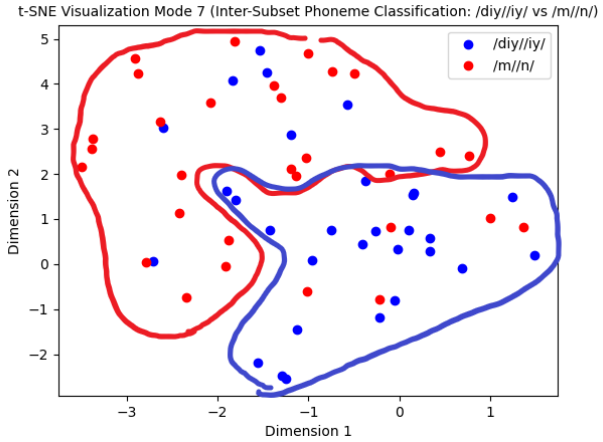


FIGURE 2: 2D t-SNE visualizations for each of the 7 classification tasks. Annotated "back-of-the-envelope" (approximate) clusters also shown.

Generally speaking, the t-SNE visualizations for binary classification showed strong inter-class discriminability, as shown by the approximate clusters drawn. However, for multi-class classification, the t-SNE visualizations were more convoluted and overlapping, and only a few approximate clusters were able to be identified.

f. Classification

Classification is an essential part of decoding imagined speech from EEG signals and consists of mapping the extracted features to their respective mental states. Various techniques have been used in this field. Traditional classification methods, including Support Vector Machine, K-Nearest-Neighbor, Decision Trees, and Logistic Regression, have been used in the past to classify EEG signals [11]. As a baseline analysis, I employ the use of the K-Nearest-Neighbor machine learning algorithm to classify the processed signals [18]. KNN is a simple and intuitive classification algorithm that works by finding the k closest training examples to a given test example in a feature space, and then predicting the label of the test example based on the labels of its k-nearest neighbors. KNN has various hyperparameters, which can be modified to improve performance, such as n_neighbors (K), which determines the number of nearest neighbors to consider for each point in the dataset, weights, which determines how the neighbors are weighted when making predictions, algorithm, which determines the algorithm used to compute nearest neighbors, leaf_nodes, which controls the size of the leaf nodes in tree-based algorithms, and p, which determines the distance metric used for computing nearest neighbors. Due to its simplicity, KNN is used as a baseline algorithm from which to compare future algorithms to.

While k-NN is a simple and effective algorithm for classification and regression tasks, it has some limitations that can hinder its performance in certain scenarios. For example, k-NN can be computationally expensive for large and high-dimensional datasets, and it can be sensitive to irrelevant and noisy features [19]. To address these limitations, I explored the use of artificial neural networks, specifically multilayer perceptron's (MLPs), which are a class of feedforward Artificial Neural Networks that can model complex and non-linear relationships between inputs and outputs. In particular, deep learning techniques, such as artificial neural networks or multilayer perceptron's (MLP), have gained attention due

to their ability to extract complex features from high-dimensional data [20]. MLP's are inspired by the biological neurons in the human brain and are composed of several layers of interconnected nodes. Each node is a simple processing unit that performs a mathematical function on its inputs, which can be thought of as the outputs of other nodes. MLP's also have various hyperparameters, including hidden_layer_size, which indicates the size (number of neurons) in the hidden layer (tuple indicates multiple hidden layers), activation, which indicates the activation function used for neurons in the network, solver, which indicates the gradient-based optimization algorithm used to train the network, alpha, which indicates the value of the L2 regularization penalty (used to prevent overfitting), learning rate, which indicates how the learning rate (step size) is to be changed throughout the use of the gradient-based optimization algorithm, max_iter, which indicates the number of training iterations to perform, early_stopping, which indicates whether to stop network training once the performance metric has stopped improving, validation_fraction, which indicates the fraction of data is to be used for validation set, beta_1 and beta_2, which indicate exponential decay rates for the Adam optimizer, and epsilon (a small number added to the denominator for numerical stability in Adam optimizer. MLPs have shown promising results in various domains, such as computer vision, speech recognition, and natural language processing, and they can overcome some of the limitations of k-NN by automatically learning relevant features and performing efficient parallel computations. MLP's have become increasingly popular in the field of EEG-based speech decoding due to their ability to capture complex relationships in the data. I believe that it is worth incorporating MLPs into our analysis in order to determine if it is a more accurate and robust solution for our problem, and I use it as an experimental algorithm from which to compare results to the baseline algorithm.

Despite the potential benefits of MLP's, they are not without challenges. One of the main challenges in using MLP's, as well as the previously mentioned KNN, is optimizing the hyperparameters. One such hyperparameter optimization algorithm, Grid Search, involves exhaustively searching over a pre-defined set of hyperparameters to find the optimal combination [21]. That is, the algorithm exhaustively iterates over each possible combination of hyperparameters and chooses the combination that yields the highest performance, which, in this case, is quantified by the classifier's cross-validation accuracy. This method, while exhaustive and rather time-consuming, can provide a robust estimate of the optimal hyperparameters. In my research on decoding imagined speech from EEG signals, I used grid search to optimize the hyperparameters of the KNN classifier and the MLP classifier. In particular, I iterated over two sets of parameters for tree-based KNN and brute-force KNN, shown in table 9 and table 10, respectively, and iterated over one set of parameters for the MLP, shown in table 11. By using grid search, I was able to identify the optimal hyperparameters for the KNN and MLP, resulting in improved classification performance. In the following tables, optimal parameter choices for both the KNN and MLP for the first classification task are shown.

Parameter	Values
n_neighbors	3, 4, **5**, 6, 7, 8, 9, 10, 11, 15, 20, 21, 25, 30, 31

Parameter	Values
weights	**'uniform**', 'distance'
algorithm	**'ball_tree'**, 'kd_tree'
leaf_size	**5**, 10, 15, 30, 50
p	**1**, 2

TABLE 2: Grid-search parameter list for KNN (tree-based algorithms). Optimal parameter choices for classification task 1 are bolded.

Parameter	Options
n_neighbors	3, 4, 5, 6, 7, 8, 9, 10, 11, 15, 20, 21, 25, 30, 31
weights	'uniform', 'distance'
algorithm	'brute'
p	1, 2

TABLE 3: Grid-search parameter list for KNN (brute-force algorithm). Classification task 1 did not choose brute force as an optimal algorithm, and therefore no parameters are bolded here.

Parameter	Values
hidden_layer_sizes	(10,), (50,), (100,), (10, 10), (50, 50), **(100, 100)**
activation	identity, logistic, **tanh**, relu
solver	**adam**

Parameter	Values
alpha	**1e-5**, 1e-4, 1e-3, 1e-2, 1e-1
learning_rate	**constant**, invscaling, adaptive
max_iter	100, **200**, 300
early_stopping	**True**
validation_fraction	0.1, **0.2**
beta_1	**0.9**
beta_2	**0.999**
epsilon	**1e-8**

TABLE 4: Grid-search parameter list for MLP. Optimal parameter choices for classification task 1 are bolded.

III. Results

In this section, the testing accuracies and F1-scores (reported as an interval; mean value +/- standard deviation) for both the KNN and MLP on all classification tasks are reported in the following table.

Classification Mode	Algorithm	Accuracy (mean +/- std.)	F1 Score (mean +/- std.)
1	KNN	24.99% (+/- 8.27%)	21.18% (+/- 7.67%)
1	MLP	17.97% (+/- 9.70%)	13.81% (+/- 9.38%)
2	KNN	36.17% (+/- 11.12%)	31.18% (+/- 11.97%)
2	MLP	42.83% (+/- 15.61%)	37.79% (+/- 15.81%)
3	KNN	65.55% (+/- 6.74%)	54.10% (+/- 9.88%)
3	MLP	58.68% (+/- 10.34%)	54.01% (+/- 9.79%)
4	KNN	14.11% (+/- 7.59%)	12.26% (+/- 6.96%)
4	MLP	14.54% (+/- 6.79%)	11.85% (+/- 6.30%)
5	KNN	62.17% (+/- 10.97%)	56.68% (+/- 14.81%)
5	MLP	63.00% (+/- 15.74%)	60.72% (+/- 17.70%)
6	KNN	61.33% (+/- 14.31%)	60.21% (+/- 14.99%)
6	MLP	59.67% (+/- 13.29%)	57.49% (+/- 15.17%)
7	KNN	62.83% (+/- 10.50%)	61.70% (+/- 11.21%)

7	MLP	60.33% (+/- 17.79%)	57.72% (+/- 19.48%)

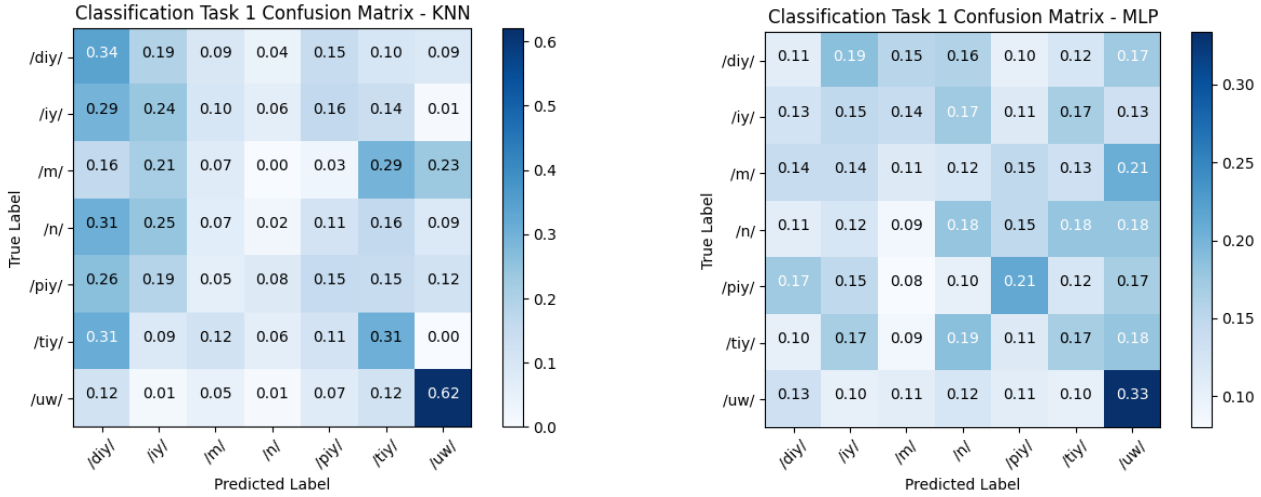
TABLE 5: Testing accuracies and F1 scores for both KNN and MLP on all classification modes.

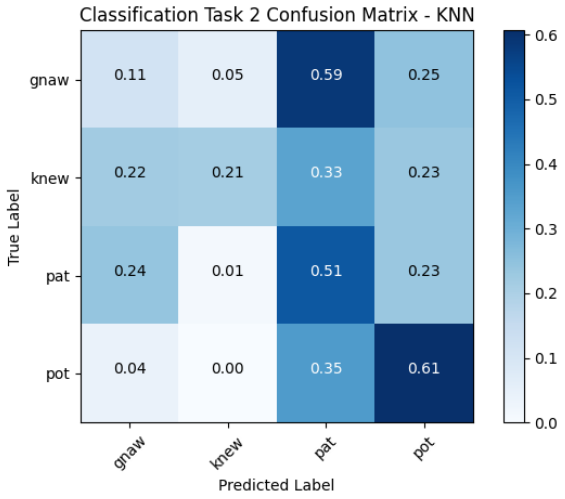
Normalized confusion matrices were also derived and graphed for each of the models on all classification tasks. In classification problems, a confusion matrix is a table that summarizes the performance of a model by comparing its predicted labels against the true labels of a dataset [22]. The confusion matrix displays the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) produced by the model. A metric named precision can be derived from the confusion matrix and measures the proportion of predicted positive cases that are truly positive, while another derived metric called recall measures the proportion of true positive cases that are correctly predicted. F1 score, reported above in the table, is the harmonic mean of precision and recall and is a useful metric in classification. The F1 score is a commonly used metric in classification tasks that combines precision and recall into a single value. The F1 score ranges between 0 and 1, where a value of 1 indicates perfect precision and recall. The formula for calculating F1 score is:

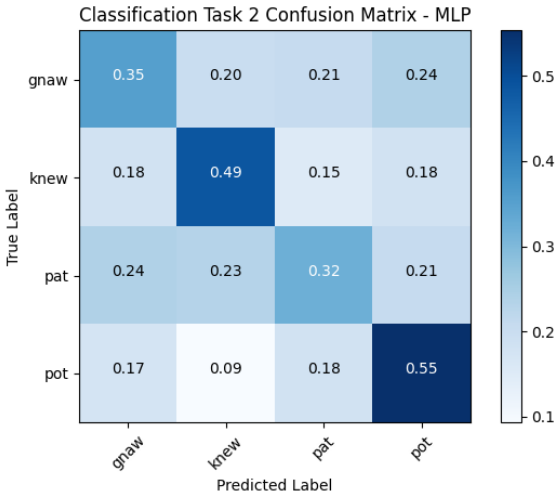
$$F1 = \left(\frac{2 * Precision * Recall}{Precision + Recall}\right)$$

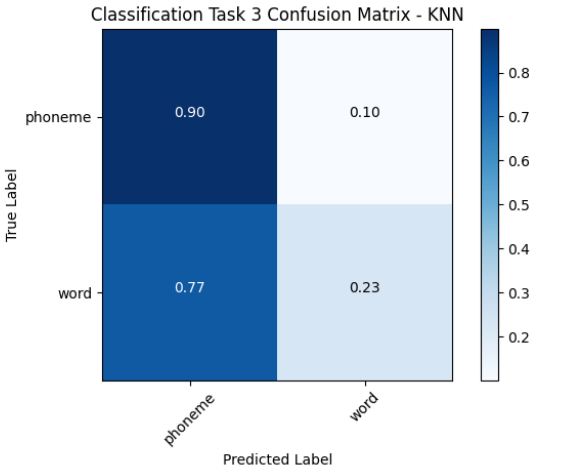
EQUATION 1: Formula for F1-score.

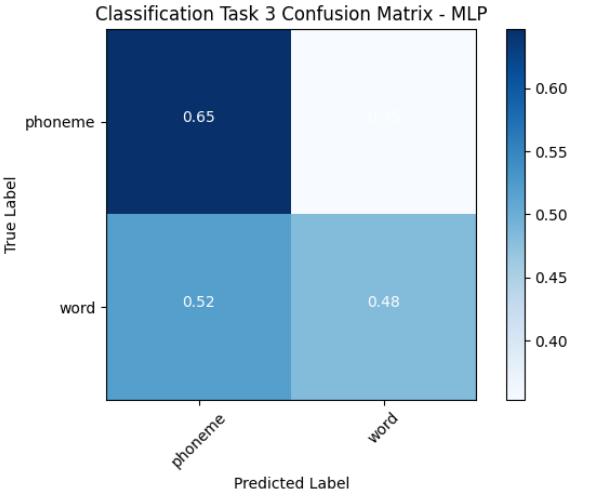
The confusion matrices for both the KNN and MLP for each classification task are shown below.

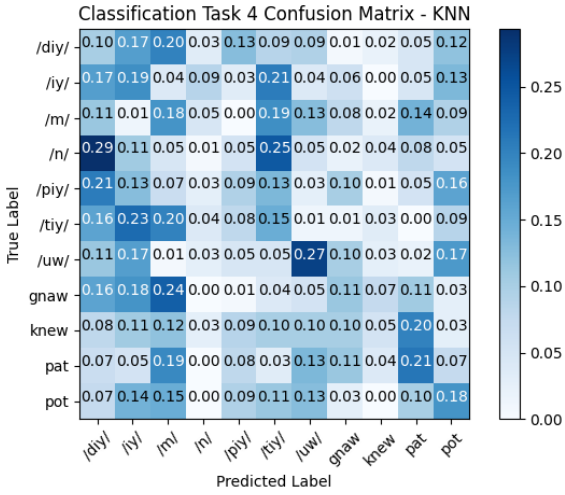


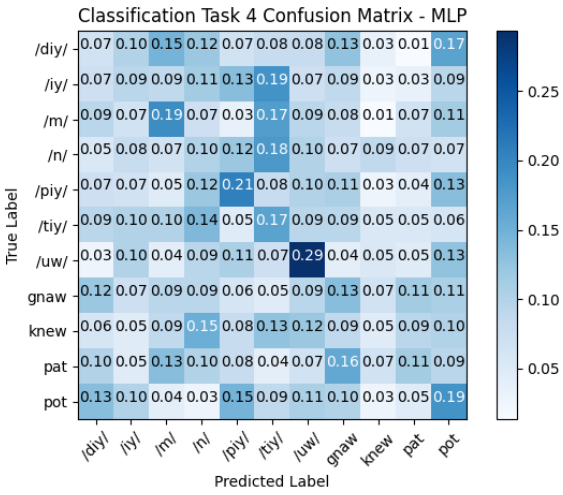

Classification Task 2 Confusion Matrix - KNN
gnaw 0.11 0.05 0.59 0.25
knew 0.22 0.21 0.33 0.23
pat 0.24 0.01 0.51 0.23
pot 0.04 0.00 0.35 0.61
True Label
gnaw knew pat pot
Predicted Label
0.6 0.5 0.4 0.3 0.2 0.1 0.0


Classification Task 2 Confusion Matrix - MLP
gnaw 0.35 0.20 0.21 0.24
knew 0.18 0.49 0.15 0.18
pat 0.24 0.23 0.32 0.21
pot 0.17 0.09 0.18 0.55
True Label
gnaw knew pat pot
Predicted Label
0.5 0.4 0.3 0.2 0.1


Classification Task 3 Confusion Matrix - KNN
phoneme 0.90 0.10
word 0.77 0.23
True Label
phoneme word
Predicted Label
0.8 0.7 0.6 0.5 0.4 0.3 0.2


Classification Task 3 Confusion Matrix - MLP
phoneme 0.65
word 0.52 0.48
True Label
phoneme word
Predicted Label
0.60 0.55 0.50 0.45 0.40


Classification Task 4 Confusion Matrix - KNN
/diy/ 0.10 0.17 0.20 0.03 0.13 0.09 0.09 0.01 0.02 0.05 0.12
/iy/ 0.17 0.19 0.04 0.09 0.03 0.21 0.04 0.06 0.00 0.05 0.13
/m/ 0.11 0.01 0.18 0.05 0.00 0.19 0.13 0.08 0.02 0.14 0.09
/n/ 0.29 0.11 0.05 0.01 0.05 0.25 0.05 0.02 0.04 0.08 0.05
/piy/ 0.21 0.13 0.07 0.03 0.09 0.13 0.03 0.10 0.01 0.05 0.16
/tiy/ 0.16 0.23 0.20 0.04 0.08 0.15 0.01 0.01 0.03 0.00 0.09
/uw/ 0.11 0.17 0.01 0.03 0.05 0.05 0.27 0.10 0.03 0.02 0.17
gnaw 0.16 0.18 0.24 0.00 0.01 0.04 0.05 0.11 0.07 0.11 0.03
knew 0.08 0.11 0.12 0.03 0.09 0.10 0.10 0.10 0.05 0.20 0.03
pat 0.07 0.05 0.19 0.00 0.08 0.03 0.13 0.11 0.04 0.21 0.07
pot 0.07 0.14 0.15 0.00 0.09 0.11 0.13 0.03 0.00 0.10 0.18
True Label
/diy/ /iy/ /m/ /n/ /piy/ /tiy/ /uw/ gnaw knew pat pot
Predicted Label
0.25 0.20 0.15 0.10 0.05 0.00


Classification Task 4 Confusion Matrix - MLP
/diy/ 0.07 0.10 0.15 0.12 0.07 0.08 0.08 0.13 0.03 0.01 0.17
/iy/ 0.07 0.09 0.09 0.11 0.13 0.19 0.07 0.09 0.03 0.03 0.09
/m/ 0.09 0.07 0.19 0.07 0.03 0.17 0.09 0.08 0.01 0.07 0.11
/n/ 0.05 0.08 0.07 0.10 0.12 0.18 0.10 0.07 0.09 0.07 0.07
/piy/ 0.07 0.07 0.05 0.12 0.21 0.08 0.10 0.11 0.03 0.04 0.13
/tiy/ 0.09 0.10 0.10 0.14 0.05 0.17 0.09 0.09 0.05 0.05 0.06
/uw/ 0.03 0.10 0.04 0.09 0.11 0.07 0.29 0.04 0.05 0.05 0.13
gnaw 0.12 0.07 0.09 0.09 0.06 0.05 0.09 0.13 0.07 0.11 0.11
knew 0.06 0.05 0.09 0.15 0.08 0.13 0.12 0.09 0.05 0.09 0.10
pat 0.10 0.05 0.13 0.10 0.08 0.04 0.07 0.16 0.07 0.11 0.09
pot 0.13 0.10 0.04 0.03 0.15 0.09 0.11 0.10 0.03 0.05 0.19
True Label
/diy/ /iy/ /m/ /n/ /piy/ /tiy/ /uw/ gnaw knew pat pot
Predicted Label
0.25 0.20 0.15 0.10 0.05

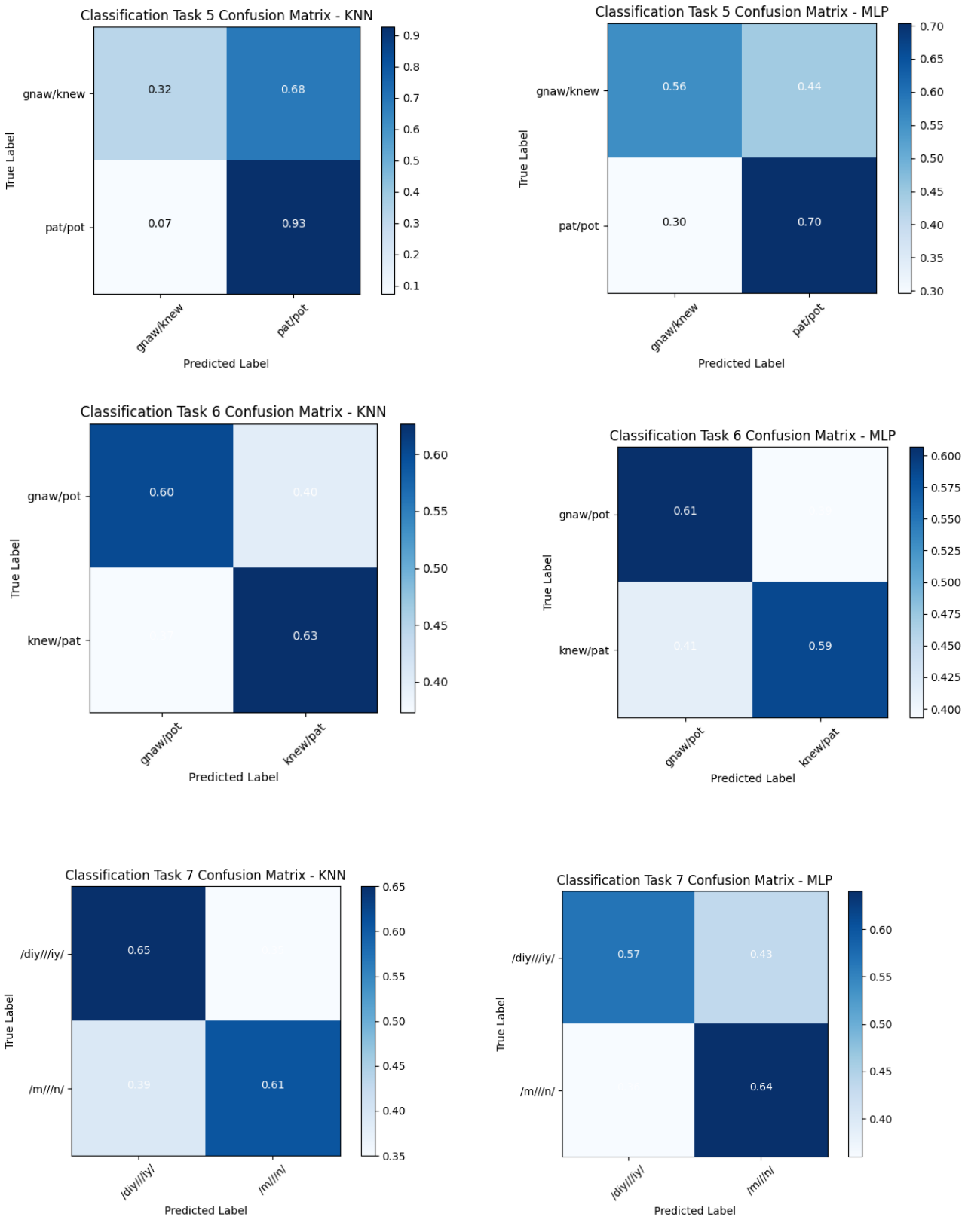


FIGURE 3: Confusion matrices for both the KNN and MLP for all 7 classification tasks.

IV. Discussion

In this section, discussion regarding the results for each of the classification tasks is provided, as well as limitations of the study and future directions. Generally speaking, I noticed testing accuracies tended to trend downwards as the number of classes in the classification task increased. For example, the mean KNN testing accuracy for a relatively large multi-class classification task, like task 4 (comprehensive free-for-all) was 14.11%, while the mean KNN testing accuracy for a binary classification task, such as task 5, was significantly higher at 62.17%. This is intuitive, as the more classes, or degrees of freedom, a classifier has to predict, the lower its accuracy can be expected to be- there simply aren't that many possible options with a lower DOF task, like binary classification. This is further reinforced by examining confusion matrices. Generally, confusion matrices corresponding to binary classification tasks (ex. Tasks 5, 6, 7) have stronger and more apparent diagonal patterns, which indicates a high degree of correct predictions across all classes. Another such trend was that, generally speaking, the mean testing accuracy of KNN was observed to be larger than that of the MLP, except for classifications tasks 2 and 4. At any rate, the testing accuracy differences between the two were not significant in any task. This result is fairly expected, as KNN's are expected to outperform MLP's on classification problems with a relatively small dataset size. For massive high-dimensional datasets, however, MLP's are indeed expected to outperform KNN's. Perhaps the most encouraging and interest detail is the fact that *all* mean testing accuracies reported for both the MLP and KNN for all classification tasks – task 1 (chance = 14.3%), task 2 (chance = 25%), task 3 (chance = 50%), task 4 (chance = 9.1%), task 5 (chance = 50%), task 6 (chance = 50%), and task 7 (chance = 50%), were *above* chance level, with some tasks and algorithms being significantly above chance level. Decoding imagined speech from EEG signals is an incredible challenge across a wide-variety of disciplines due to factors including the limited spatial resolution of EEG, limited training data, etc, and the reporting of mean testing accuracies that are above chance level indicates that EEG may contain information needed to discriminate between imagined linguistical units, which is quite encouraging considering the difficulties involved. In the future, additional analysis and experimentation with different algorithms in the areas of data preprocessing/denoising, feature extraction, and classification will be performed. Additionally, different datasets may be experimented with, that may not be regarding imagined speech tasks, but also imagined handwriting [23] or imagining images [24]. The reasoning behind such experimentation is that different paradigms may offer a higher quality of signal data, at least in terms of higher cortical activation, more pronounced features and inter-class discriminability, than imagined speech. The end goal for AssistEEG is to produce a practical BCI system capable of decoding imagined communication from speech-impaired users, at both a relatively high WPM rate and high accuracy. This project, AssistEEG V0, represents the first steps taken in building such a system, and demonstrates encouraging first steps in using EEG signals to decode imagined speech. The results obtained from this initial version of AssistEEG provide a foundation for future improvements and developments towards achieving the end goal of a practical and reliable BCI system for speech-impaired individuals. The potential impact of such a system could significantly enhance the quality of life and communication capabilities of those with speech disabilities, allowing them to interact with their surroundings in new and meaningful ways.

V. References

[1] Wolpaw, J. R., Birbaumer, N., McFarland, D. J., Pfurtscheller, G., & Vaughan, T. M. (2002). Brain-computer interfaces for communication and control. Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology, 113(6), 767–791. doi:10.1016/s1388-2457(02)00057-3

[2] Fenton, A., Alpert, S. Extending Our View on Using BCIs for Locked-in Syndrome. Neuroethics 1, 119–132 (2008). https://doi.org/10.1007/s12152-008-9014-8

[3] Dr. Stephen Hawking: A case study on using technology to communicate with the world. Dr. Stephen Hawking: A Case Study on Using Technology to Communicate with the World | DO-IT. (n.d.). Retrieved February 27, 2023, from https://www.washington.edu/doit/dr-stephen-hawking-case-study-using-technology-communicate-world

[4] Haynes, JD., Rees, G. Decoding mental states from brain activity in humans. Nat Rev Neurosci 7, 523–534 (2006). https://doi.org/10.1038/nrn1931

[5] Vilela, M., & Hochberg, L. R. (2020). Applications of brain-computer interfaces to the control of robotic and prosthetic arms. In Brain-Computer Interfaces (pp. 87–99). Elsevier. https://doi.org/10.1016/b978-0-444-63934-9.00008-1

[6] Mcfarland, D. J., & Wolpaw, J. R. (2010). Brain–Computer Interfaces for the Operation of Robotic and Prosthetic Devices. In Advances in Computers (pp. 169–187). Elsevier. https://doi.org/10.1016/s0065-2458(10)79004-5

[7] Guger, C., Harkam, W., Hertnaes, C., & Pfurtscheller, G. (2001). Prosthetic Control by an EEG-based Brain-Computer Interface (BCI).

[8] Kübler, A., Furdea, A., Halder, S., Hammer, E. M., Nijboer, F., & Kotchoubey, B. (2009). A Brain-Computer Interface Controlled Auditory Event-Related Potential (P300) Spelling System for Locked-In Patients. In Annals of the New York Academy of Sciences (Vol. 1157, Issue 1, pp. 90–100). Wiley. https://doi.org/10.1111/j.1749-6632.2008.04122.x

[9] Lesenfants, D., Habbal, D., Lugo, Z., Lebeau, M., Horki, P., Amico, E., Pokorny, C., Gómez, F., Soddu, A., Müller-Putz, G., Laureys, S., & Noirhomme, Q. (2014). An independent SSVEP-based brain-computer interface in locked-in syndrome. Journal of neural engineering, 11(3), 035002. https://doi.org/10.1088/1741-2560/11/3/035002

[10] Bacher, D., Jarosiewicz, B., Masse, N. Y., Stavisky, S. D., Simeral, J. D., Newell, K., Oakley, E. M., Cash, S. S., Friehs, G., & Hochberg, L. R. (2015). Neural Point-and-Click Communication by a Person With Incomplete Locked-In Syndrome. Neurorehabilitation and neural repair, 29(5), 462–471. https://doi.org/10.1177/1545968314554624

[11] Lopez-Bernal D, Balderas D, Ponce P and Molina A (2022) A State-of-the-Art Review of EEG-Based Imagined Speech Decoding. Front. Hum. Neurosci. 16:867281. doi: 10.3389/fnhum.2022.867281

[12] Panachakel JT and Ramakrishnan AG (2021) Decoding Covert Speech From EEG-A Comprehensive Review. Front. Neurosci. 15:642251. doi: 10.3389/fnins.2021.642251

[13] Zhao, S., & Rudzicz, F. (2015). Classifying phonological categories in imagined and articulated speech. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). ICASSP 2015 - 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE. https://doi.org/10.1109/icassp.2015.7178118

[14] Wan, Z., Yang, R., Huang, M., Zeng, N., & Liu, X. (2021). A review on transfer learning in EEG signal analysis. In Neurocomputing (Vol. 421, pp. 1–14). Elsevier BV. https://doi.org/10.1016/j.neucom.2020.09.017

[15] Yosrita, E., Nur Aziza, R., Farah Ningrum, R., & Muhammad, G. (2021). Denoising of EEG signal based on word imagination using ICA for artifact and noise removal on unspoken speech. In Indonesian Journal of Electrical Engineering and

Computer Science (Vol. 22, Issue 1, p. 83). Institute of Advanced Engineering and Science. https://doi.org/10.11591/ijeecs.v22.i1.pp83-88

[16] St›hle, L., & Wold, S. (1989). Analysis of variance (ANOVA). In Chemometrics and Intelligent Laboratory Systems (Vol. 6, Issue 4, pp. 259–272). Elsevier BV. https://doi.org/10.1016/0169-7439(89)80095-4

[17] Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).

[18] Peterson, L. E. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883.

[19] Uddin, S., Haque, I., Lu, H., Moni, M. A., & Gide, E. (2022). Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction. In Scientific Reports (Vol. 12, Issue 1). Springer Science and Business Media LLC. https://doi.org/10.1038/s41598-022-10358-x

[20] Murtagh, F. (1991). Multilayer perceptrons for classification and regression. In Neurocomputing (Vol. 2, Issues 5–6, pp. 183–197). Elsevier BV. https://doi.org/10.1016/0925-2312(91)90023-5

[21] Fayed, H. A., & Atiya, A. F. (2019). Speed up grid-search for parameter selection of support vector machines. In Applied Soft Computing (Vol. 80, pp. 202–210). Elsevier BV. https://doi.org/10.1016/j.asoc.2019.03.037

[22] Susmaga, R. (2004). Confusion Matrix Visualization. In Intelligent Information Processing and Web Mining (pp. 107–116). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-39985-8_12

[23] Willett, F.R., Avansino, D.T., Hochberg, L.R. et al. High-performance brain-to-text communication via handwriting. Nature 593, 249–254 (2021). https://doi.org/10.1038/s41586-021-03506-2

[24] Kavasidis, I., Palazzo, S., Spampinato, C., Giordano, D., & Shah, M. (2017). Brain2Image. In Proceedings of the 25th ACM international conference on Multimedia. MM ’17: ACM Multimedia Conference. ACM. https://doi.org/10.1145/3123266.3127907

[25] Hovagim Bakardjian "Optimization of steady-state visual responses for robust brain-computer interfaces" 2010 Abootalebi et al. "A new approach for EEG feature extraction in P300-based lie detection" 2009