Towards Real-Time Classification of Human Imagined Language

Joseph Zonghi
Rochester Institute of Technology & Kanazawa Institute of Technology
Department of Computer Engineering
Kanazawa, Japan
jaz8207@rit.edu

***Abstract*—The primary goal of this research was to develop a system capable of predicting a given user's imagined language in real time using their brainwave data. This research analyzed both FPGA-based and software based approaches to real-time classification of imagined language. The classification was binary between English and Japanese, and a dataset containing imagined speech from both languages was also created. Another goal of this research was to consider the effects of quantization of the network weights in order to examine the resulting utilization of the FPGA to allow for other applications to run in conjunction with our proposed system. With test accuracies over 95% but real-time accuracies only barely approaching 60%, it can be considered partly successful. Real-time approaches to predicting imagined EEG words are rare, and attempts at predicting imagined language are even rarer. Such a system could be beneficial in helping multi-lingual environments that standard natural language processing systems have difficulty in noticing changes in language, especially those that occur in real time. Further iterations on this proposed system could also assist those who have difficulty articulating speech and would benefit from having a brainwave-based system that is portable and works in real-time. It is hopeful that this work can lead to future iterations and advancements in the realm of real-time imagined speech classification both through their own attempts or perhaps with the help of the English/Japanese imagined speech dataset created through this research.**

***Index Terms*—EEG, brainwave, classification, imagined speech, neural network, FPGA**

I. Introduction

A common problem in the realm of neuroscience is with regards to patients who have anarthria, or the loss of the ability to articulate speech. There are many ways to assist those with anarthria in order to communicate, but properly classifying electroencephalography (EEG) data obtained from the user's brain wave signals was thought to be an eventual possibility at best. However, as time and technology progress, whole word and phrase decoding using EEG data has been done to varying degrees of success by researchers in multiple studies [6] [12]. They discovered, through the use of invasive measures implanted inside the user's head to record their EEG signals, that classification of imagined words was indeed possible. However, one issue encountered was that post-hoc accuracy was higher than real time accuracy. Furthermore, an invasive device used to perform these measurements incurs surgery and maintenance costs that might not be easily attainable for the average person. Fortunately, non-invasive EEG measuring technologies exist in the form of a headset the user can place on their head. Were these to become cheaper and more commonplace, it stands to reason that many people would consider it a more agreeable solution. However, the major drawback with non-invasive methods is their generally lower accuracy rates compared to that of intra-cranial options typically in the range of 20 to 100 times worse signal quality when using signal to noise ratio (SNR) as a metric [2].

When analyzing EEG data, a common problem encountered is its heavy dependence on multiple time points scattered across multiple channels. This high level of dimensionality results in standard linear regression systems lacking sufficient accuracy in properly classifying or predicting said data. Therefore, more advanced data classification techniques such as an artificial neural network (ANN) or perhaps one with multiple layers, a deep neural network (DNN), could be beneficial in this case. An ANN's ability to better classify a wide range of input data than that of standard linear regression algorithms could be a suitable choice for this type of problem. An important consideration regarding EEG data though is its time dependence, as data from two nearby time points do have some level of correlation with each other. As such, it is important to consider that a new input data point does have relation with previous or future data points.

II. Related Work

As a result of all of these reasons, analyzing and classifying EEG data is not necessarily a simple task. Through this we propose a system wherein a neural network is loaded onto and trained on an FPGA. This FPGA is also able to handle input data, save it across a given window, perform feature extraction on said window, and utilize the aforementioned neural network to classify the input data as either English or Japanese. The target of this being English and Japanese is to have a dataset with its two states having comparatively high variance between each other. Japanese and English have very different syntax and sentence structures [8], potentially leading to similarly different EEG signals due to the inherent temporal property sentences have (with a beginning, middle, and end most of the time). Furthermore, there is a measured difference in a native Japanese speaker's ability to speak English [7], and it might be the case that these difficulties also carry over into imagined

speech. Therefore, such a dataset should theoretically be a good fit for having two distinctly differentiable classes to classify. However, the ability to classify these on their own may not provide much tangible benefit. Instead, it might be beneficial to use this EEG data to assist with speech recognition systems that have difficulty differentiating between a change in language in real time. Some solutions to such issues are using video in conjunction with audio in busy environments, but the accuracies from such pursuits were still below 30% [10]. Perhaps a low cost EEG-based system could assist in environments, particularly learning environments, where multiple languages are present but hard to differentiate through audio alone. Of course, this would incur costs with regards to supplying headsets to users as long distance EEG is not known to be feasible currently.

Decoding speech from EEG has been showing steady improvements in accuracy in the past few years [5] [12] both in terms of audible and imagined speech. One attempt [4] with k-nearest neighbor approaches at classification resulted in 58% accuracy for binary classification, with a big takeaway being that syllable classification is easier than word-based classification. A similar approach [11] using Naive Bayes, Support Vector Machines, and Random Forests tried to expand beyond binary and reached accuracies marginally above random chance for Support Vector Machines (20-35% for 5 classes) but over 40% accuracies when using Random Forests-based approach. Echo state network (ESN) based methods have found success in extracting features from EEG data even through unsupervised methods [9], so it might be useful to incorporate similar feature extraction methods if the current setup is found to be unsatisfactory in terms of accuracy. An ESN is a recurrent neural network that specializes in recognizing relationships or extracting features from temporal data, which while they could be done using functions like mean or median, they might not provide enough of a difference between points to allow the network to successfully converge on a solution. When it comes to bilingual classification, there is a previous study that had attempted to do so [1]. They managed to have high accuracies (about 92%) for language classification, but their dataset only contained responses to yes or no questions.

III. Quantization-Aware Training

The network was designed and tested using Tensorflow in Python. However, MATLAB was still used to prepare any training data for its ease of use for quickly adjusting various features, window sizes, and other general meta-level components of the training data. This would then be saved to a MATLAB .mat file for the Python program to import. Many tests were performed in Python to determine the optimal fixed point size with regards to assumed utilization. There were three methodologies performed, standard model quantization, i-bit quantization, and 1-bit quantization. Standard model quantization means using Tensorflow's standard training methods and then quantizing the network's weights after training is complete. The other two, named i-bit and 1-bit quantization,

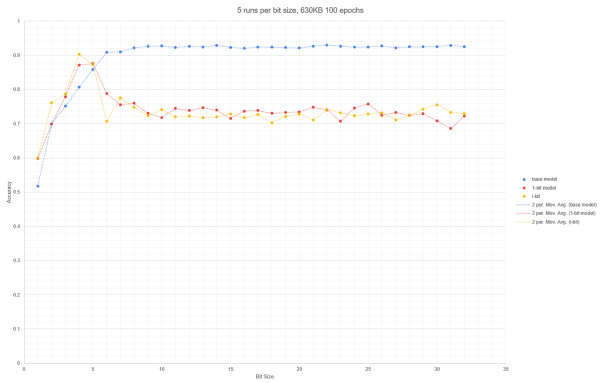

Fig. 1. Results of the 3 training methodologies per post-training bit rounding.

use Tensorflow's quantization-aware training functionality to train the network. Quantization-aware training is the practice of training a model to be able to better handle incoming data that might not be a standard floating point number. For example, 8-bit quantization aware training trains the model to handle incoming data that is 8 bits in precision. i-bit training, in this case, would be training the model to handle data of any given i-bit resolution and comparing it against the 1-bit quantization aware and standard training methods. Figure 1 shows the results of training on prompt-based speech confined to an approximate network size at various fixed point sizes. The accuracy values shown are also the average of multiple runs.

The x axis refers to the resolution of the weights after training by manually changing them to their respective quantized values. This is true for all three methodologies, where the only difference between them is the level of awareness of quantization. The base model takes no quantization awareness into account when training, the 1-bit model always trains under the awareness it will function with 1-bit resolutions regardless of the final rounding resolution, and i-bit quantization trains under the awareness that it is training on bit resolutions equal to the resolution that its weights will also be rounded to after training. From Figure 1, it can be seen that there are few differences between 1-bit and i-bit quantization aware training.

Another takeaway from these tests is that resolution does not necessarily have a noticeable effect on the accuracy of the network after a certain point. Moreover, a sufficiently large network can have relatively low numbers of hidden neurons without compromising accuracy. As per Table III, network sizes above 100 do not necessarily result in noticeably large increases in accuracy, and small networks with only 20 neurons are capable of adapting to training data belonging to a single user.

With regards to quantization, it depends on the network with regards to what training methodology and bit resolution should be used. For a full 32 bit fixed point implementation, using Tensorflow's standard training procedure is recommended. For models that need to use smaller number sizes, such as those below 6 bits, using either single bit or i-bit training would

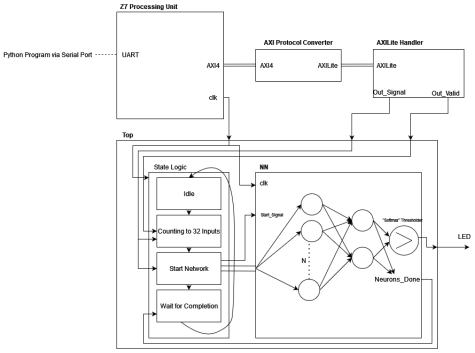

Fig. 2. RTL Diagram of the FPGA design.

be recommended. For the purposes of this research, speed is important, and as such minimizing the number of weights is also important. i-bit and 1-bit trained models use higher numbers of weights to achieve the same results as a normal model. With more weights present, the network has to spend more time computing the addition of each weight and input combination. Thus, Tensorflow's standard training was used.

IV. FPGA Design

By using a field-programmable gate array (FPGA), the final design would be able to perform multiple different tasks simultaneously in real time. An FPGA was chosen due to the ability to easily adjust specific parameters of the design in order to accommodate various rapid prototyping configurations. For example, the neural network could potentially go through various iterations, so it would be beneficial to have a generically scalable design to assist with rapid prototyping.

Since one of the goals of this research was to examine the trade-offs between accuracy, number quantization, and FPGA utilization, relatively small-size FPGAs are the target for this. Specifically, the Zybo Z7 Zynq-7020 SoC board was used [3]. The Emotiv series of devices require the use of their proprietary software in order to obtain raw EEG data from a network data stream, so most of the data handling is done outside of the digital logic. Figure 2 shows the FPGA design when using a primarily software-based approach to get the raw EEG data from the EmotivPRO Lab Streaming Layer and pre-process it before sending to the FPGA's behavioral neural network.

The FPGA design was ultimately performed using RTL simulations to estimate and observe the effects of changing the network size and quantization levels and the resulting utilizations of the device. A Python program is used to communicate with and interact with the EmotivPRO data stream, and it saves the data to a single 1x32 array. For training, each newly obtained array (at a rate of 128 Hz) is saved as a new line in a text file including the current language's corresponding label, 0 for English and 1 for Japanese. The labels are determined by the Python program itself when it displays the language to the user over command prompt. For real-time purposes, the data is not saved to a file but instead sent over the serial port to the FPGA itself. First, the Python program must convert each number to a two's complement string of binary and then separate the string into separate bytes and send as a packet. This alone currently takes more than 1/128 of a second to process, meaning the input is delayed and the Python program must clear the queue from the data stream to get the next real value, taking even more time. For this reason, the RTL simulation-based approach will be discussed instead with the hardware approach saved for future work.

A. Pre-Processing

Based on the success found in [12] with their choice of features, a similar feature set was used in this case. Those features were computed over the given window and then treated as one data point per feature per channel. Specifically, the features are: sum, mean, absolute mean, median, sixth power mean, standard deviation, variance, maximum, absolute maximum, minimum, absolute minimum, delta of maximum and minimum, sum of maximum and minimum, skewness, and kurtosis.

While these features were considered for implementation in the FPGA, software attempts at training on the data found that the presence of more features resulted in decreases in accuracy as can be seen in Table II similar to the findings in [12] when using simpler SVM methods with variable numbers of features present. Assuming the data coming from Python is raw EEG data, the processing unit on the FPGA uses a C program to compute the moving mean. It uses two arrays to do so. The first array (1x32) contains the sums of the incoming raw data per channel while the second array is a 2D array (Nx32) containing the raw EEG values to subtract from the corresponding channel sum with N being the desired window size. The subtractions do not occur until the first N number of values come in. At this point, the program adds the next incoming values to their corresponding sums while subtracting the first values in the subtraction 2D array. The memory usage of this program can be estimated to be approximately the size of an integer (4 bytes) multiplied by 32 and then by N, the size of the desired window. Those values are then replaced with the newly received values and the pointer moves to the second values to subtract up until N where they loop back to the first values (time points with a multiple of N). At this point the program also sends a start signal to the FPGA to begin accepting incoming data over AXILite. The FPGA then uses a simple state machine to accept the next 32 signals as input to the neural network portion.

B. Utilization

The neural network region generates a generic number of neurons for the input layer and 2 for the output layer. Weights for the network are created ahead of time using the Tensorflow model's weights, converting them to the desired fixed point value, and writing them out as special arrays of sfixed_bus_array (named WeightsL1 and WeightsL2 for a network with one hidden layer). The general size in bits of this synthesizable ROM is given by Equation 1 where L is the size of the hidden layer, I is the number of inputs to the

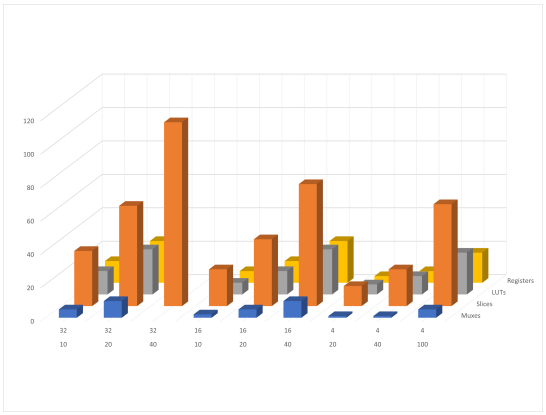

Fig. 3. Percentage of utilization for each component type with the X-axis showing total bit usage per number above and total number of neurons below.

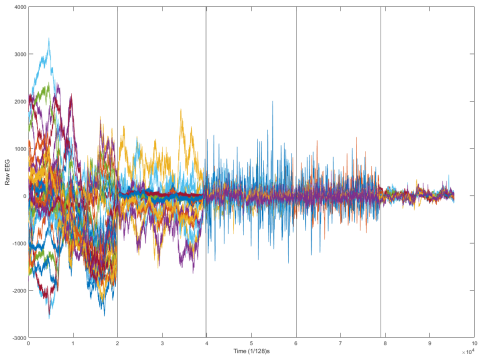

Fig. 4. EEG Data of all users using the 32 channel Emotiv Flex.

network, X is the length of the integer portion of the fixed point number, and Y is the fractional length of the fixed point number. The additions of 1 are to account for the bias weight.

$$S = ((X+Y)*(I+1)*L) + ((X+Y)*(L+1)*2) \quad (1)$$

It should be noted, however, that the FPGA may not necessarily synthesize this as ROM with an S amount of bits. When using Xilinx Vivado, the utilization is reported as LUTs, registers, multiplexers, and logic slices. Figure 3 shows a 3D bar graph of the utilization percentages.

Interestingly, the utilization appears to follow a linear trend with regards to the size of the network, with only a variation of 1-3%, likely due to larger networks using the same amount of control lines as a previous size with the only difference then being more components used to handle the extra weights present. Utilization appears to be consistent with total number of bits in the system, as the utilizations are very similar for a 20 neuron network with 32 bit weights compared to a 40 neuron network with 16 bit weights. However, as bit sizes become low, it seems to be the case that the utilization increases at a larger rate. In this case, there may be too many weights present to properly partition the logic slices, the consistently largest part of all of the configurations. Though it is likely to differ to some extent between FPGAs and synthesization software, we believe this methodology to be effective in predicting the utilization of any desired network size due to its comparatively linearly comparable nature.

V. Dataset Creation and Results

A. Prompt-Based Imagined Speech

The dataset contains 5 subjects, ages 20-25, with 4 male subjects and 1 female subject. 4 of the subjects are native Japanese speakers while one of the male subjects is a native English speaker. Each user was instructed to read random combinations of phrases for a total of approximately 8 seconds per combination. The prompts were displayed via Windows command prompt without any other words presented on the screen. The prompts presented would be given by switching between English and Japanese and having the user read one language at a time. For example, a prompt could be ”Today is very hot, but it seems that it will rain next week.” combined with ”The supermarket sells bananas, but they do not have blueberries.” with the Japanese version being a similarly translated version. Prompts contained an array of topics in order to more closely mimic natural speech. By using random combinations of prompts, the user would be more likely to focus if the given prompts were somewhat randomized. Each user would be given 60 combinations in total, split up as 30 English and 30 Japanese. With a 128 Hz sampling rate from the Emotiv Flex, this results in approximately 61,440 data points per user. The decision was made to keep the dataset balanced between English and Japanese as they are both considered to be stimuli-based and inherently do not have balance-related differences between them in the way that the [12] dataset did with regards to the resting and speaking states, with speaking being more varied between users than resting. The raw EEG data obtained from the each of the 32 channels from all of the 5 subjects can be seen in the MATLAB plot in Figure 4. The vertical lines demarcate changes in subject, and these transitions did not happen in real time; the plot was created after the experiments were complete. Furthermore, the downward trend in amplitude is just coincidence, as the 5th subject, the English native subject, was actually recorded first. Due to them being a potential outlier due to the language difference, they were chosen to be number 5.

Initially the Emotiv EPOC X was used on the native English speaker subject for its lightweight practicality and comparatively low setup time. However, when attempting on the same user with the Emotiv Flex, a 32 channel head cap, the increase in accuracy ranges from 10-25% depending on methodology over that of the EPOC X. No other parameters of the training, pre-processing, or data collection methods were changed, just the device used. Taking this into consideration, it was decided that the sacrifice in ease of use with the EPOC X was worth it in exchange for the large increase in accuracy provided by the EPOC Flex. Use of the EPOC Flex requires more time calibrating and adjusting saline levels of the individual

TABLE I
TABLE COMPARING THE AVERAGE TEST ACCURACIES OF IMAGINED SPEECH BETWEEN EPOC X (15 FEATURES) AND EPOC FLEX (1 FEATURE) FOR USER 5.

	EPOC X	EPOC Flex
random + no regularization	0.7094 ±0.033	0.9538 ±0.021
random + L1 & L2 reg. of 0.0001	0.7583 ±0.054	0.9962±0.001
prompts + L1 & L2 reg. of 0.0001	0.5940 ±0.051	0.8750±0.043

TABLE II
TABLE SHOWING THE RESULTING ACCURACIES OF VARIOUS FEATURE COMBINATIONS FOR USER 5.

	EPOC X	EPOC Flex
Mean Only	0.7137 ±0.039	0.9846 ±0.010
Above + Max + Min	0.6368 ±0.020	0.9538 ±0.018
Above + Std. Dev. + Var.	0.5897 ±0.022	0.9077 ±0.014
Skewness & Kurtosis	0.5214 ±0.027	0.3692 ±0.076
All 14	0.5940 ±0.019	0.8923 ±0.034
Raw EEG	0.5024 ±0.041	0.9940 ±0.003

sensors as well as a higher price for purchasing. Despite this, the increased accuracy was ultimately considered to be worth more, so subsequent tests were done using the EPOC Flex. Table I shows the preliminary results when comparing the two for a single user.

Random refers to data obtained from having the user imagined completely random speech whereas prompts refer to speech generated by reading prompts. Based on the results in Table I, the remaining users would have their data taken using the Flex for the markedly higher accuracy. Another important result from these early findings was the feature usage. In both cases, it appears that reducing the amount of features available generally increases the accuracy incrementally. This is likely due to the other features introducing or amplifying any noise present in the data whereas the mean should work to better neutralize it by nature of acting as a low pass filter. A various assortment of feature combinations and their resulting accuracies for the EPOC X and EPOC Flex can be seen in Table II for random sentences for one subject.

Since the mean becomes such an integral part in having an effect on the classification compared to any other single feature, some further analysis was performed. Figure 5 shows the moving averages for each channel over time for three users with the black vertical lines representing changes from one subject to another. Do note that the English and Japanese plots do no occur simultaneously, but they were recorded separately from each other and comprise the same amount of time as each other, so it would be reasonable to show them as if they occurred simultaneously for the sake of visual comparison.

The first visually distinct part of the plots is the large variation in structure for each subject. Despite each subject having wildly different patterns, their average value is still as expected, 0.

B. Inter-Personal Uniqueness

Based on the results from Table III, it appears to be the case that an individual user presents a new level of variance

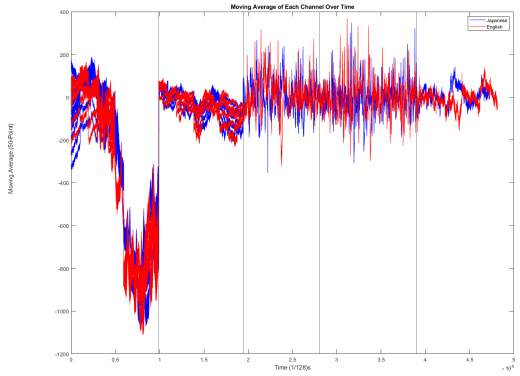

Fig. 5. Comparison of moving average per channel.

TABLE III
COMBINATIONS OF SUBJECTS AND THE RESULTING ACCURACIES AMONGST THEMSELVES AND COMPARED TO A NEW SUBJECT.

100 Neuron Network	Test Accuracy	Accuracy on New Subject
Subjects 1-4	0.7842 ±0.010	0.5111 ±0.047
Subjects 1-3	0.7984 ±0.011	0.5130 ±0.032
Subjects 1-2	0.8570 ±0.017	0.4983 ±0.045
Subjects 2-3	0.8443 ±0.013	0.5083 ±0.032
Subject 2	0.8646 ±0.013	0.5101 ±0.039
Subject 2 (20 neurons)	0.7467 ±0.006	0.5264 ±0.028
Subject 2 (1000 neurons)	0.8919 ±0.005	0.4969 ±0.030

that the network is not capable of classifying in any reasonable manner given that the accuracies all stay very close to random guessing, 0.5. Though this does not align with the goal of this research, that is not to say it is an entirely unexpected outcome. As the number of users present in the training set increases, the accuracy seems to decrease, but it decreases at a rate that also declines with respect to the number of users present. Perhaps the way to rectify this issue is the addition of more subjects to the dataset as a whole in order to potentially smooth out the overall variance through the introduction of more data points. In Table III, the most potentially promising combinations were those of subjects 2 and 3 tested on subject 4. They reached a reasonably high test (and train) accuracy of 0.8443, but the accuracy when testing on subject 4 resulted in 0.5083. The subject 1 and 2 combination was done to see if combining two separate subjects with highly contrasting levels of variance would be more effective when introducing a new subject, but that was not the case as the final accuracy for said subject was 0.4983. Based on these results alone, especially with how low the accuracies are for a new subject, the current methodology and dataset are not sufficient in creating a system capable of working immediately for new users. It should also be noted that the test accuracies for all of these tests were very close to the training accuracies. This could potentially mean the data itself can not effectively be classified by any manner, and accuracies above 90% for prompt-based methodologies are out of reach currently. Despite this, accuracies above 60% are certainly possible when using training and testing data

TABLE IV
VARIOUS RESULTS OF THE "REAL-TIME" APPROACH TO EVALUATING THE MODEL.

Users	Target	Test Accuracy	"Real-Time" Accuracy	Method
1-5	1	0.9095 ±0.029	0.5540 ±0.034	Raw EEG
1-5	2	0.8817 ±0.036	0.5502 ±0.019	Raw EEG
1-5	3	0.8792 ±0.031	0.5347 ±0.013	Raw EEG
1-5	4	0.9063 ±0.030	0.5477 ±0.022	Raw EEG
1-5	5	0.9418 ±0.038	0.5049 ±0.027	Raw EEG
1-5	1	0.9136 ±0.024	0.4915 ±0.036	Mean
1-5	2	0.9546 ±0.024	0.5071 ±0.038	Mean
1-5	3	0.9008 ±0.026	0.4998 ±0.048	Mean
1-5	4	0.9126 ±0.028	0.5447 ±0.017	Mean
1-5	5	0.9465 ±0.021	0.4958 ±0.066	Mean
1	1	0.9520 ±0.009	0.5118 ±0.034	Raw EEG
2	2	0.8279 ±0.020	0.5241 ±0.047	Raw EEG
3	3	0.8549 ±0.008	0.5256 ±0.033	Raw EEG
4	4	0.8516 ±0.009	0.5379 ±0.023	Raw EEG
5	5	0.9316 ±0.018	0.4615 ±0.038	Raw EEG
1	1	0.9411 ±0.019	0.4861 ±0.027	Mean
2	2	0.9896 ±0.003	0.6042 ±0.025	Mean
3	3	0.9893 ±0.006	0.5421 ±0.044	Mean
4	4	0.8970 ±0.011	0.5122 ±0.033	Mean
5	5	0.9980 ±0.002	0.4648 ±0.032	Mean

belonging to all users. In this regard, the dataset can be considered successful, as the introduction of new subjects to the training set does not necessarily preclude the network from accurately classifying the output from any of the members in the dataset, regardless of native language.

C. Real-Time Results

In order to best replicate real-time performance, a file containing raw EEG data taken at a time separately from when the training data was used and given to the network along with its corresponding labels through the Tensorflow evaluate() function.

Test Accuracy refers to the accuracy obtained from data removed from the training data but still part of the same experiments. Target refers to the number of the user whose data will be used as the evaluation data. The "Real-Time" column refers to accuracies obtained from said user's data. For each iteration, the training data contained data from two separate recording sets, with the third recording set (20 prompts per set) used as the "real-time" evaluation data. Despite being able to successfully adapt to the two sets for training and testing purposes, the model does not currently meet successfully accuracy levels for a third set except for barely passing in one case for user 2 by reaching above 60% accuracy. There are multiple reasons as to why this may happen. Perhaps EEG data is truly so unique that it is very difficult to generalize it for a large sized amount of entirely new data, even if similar stimuli are being presented to the user. Furthermore, when conducting the experiments, there is also potential that the users' moods or level of focus fluctuated throughout. For training points close to each other temporally, this may not be an issue, but when considering points temporally far from each other, this evidently becomes a much larger issue. While training used L1 and L2 regularization values of 0.001, it still does not seem to be the case that regularization could assist in this case. Visually looking at Figure 5 further shows that even when constrained to a single user, there tends to be a large fluctuation in the values as time progresses, especially for User 1.

VI. CONCLUSION

While the desired real-time classification results were not as hoped, there is still much that can be learned from this research. The first is that EEG data is inherently differentiable and classifiable, but it is not a simple task. Despite having very successful accuracies when training and testing the network on a given dataset, the network is unable to adapt to any new user or data obtained with from an existing user at a different time period (even if only approximately 30 seconds separated between recording sessions). The best way to rectify such an issue would be to record and include more EEG data as part of the training data. While a network has more success in classifying data belonging to the same user used in training data compared to a training dataset containing all users, theoretically a large enough set of users balances out this issue to allow for potentially completely generalized models that do not need to train on a new user. However, unfortunately, this is not yet the case with this research and this dataset.

VII. REFERENCES

REFERENCES

[1] A. Balaji, A. Haldar, K. Patil, T. S. Ruthvik, V. CA, M. Jartarkar, and V. Baths. Eeg-based classification of bilingual unspoken speech using ann. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1022–1025, 2017.

[2] T. Ball, M. Kern, I. Mutschler, A. Aertsen, and A. Schulze-Bonhage. Signal quality of simultaneously recorded invasive and non-invasive eeg. *NeuroImage*, 46(3):708–716, 2009.

[3] Zybo z7 - digilent reference, 2020.

[4] N. Hashim, A. Ali, and W.-N. Mohd-Isa. *Word-Based Classification of Imagined Speech Using EEG*, pages 195–204. Springer, 02 2018.

[5] G. Krishna, Y. Han, C. Tran, M. Carnahan, and A. H. Tewfik. State-of-the-art speech recognition using eeg and towards decoding of speech spectrum from eeg, 2020.

[6] D. A. Moses, S. L. Metzger, J. R. Liu, G. K. Anumanchipalli, J. G. Makin, P. F. Sun, J. Chartier, M. E. Dougherty, P. M. Liu, G. M. Abrams, A. Tu-Chan, K. Ganguly, and E. F. Chang. Neuroprosthesis for decoding speech in a paralyzed person with anarthria. *New England Journal of Medicine*, 385(3):217–227, 2021.

[7] K. Ohata. Phonological differences between japanese and english: Several potentially problematic areas of pronunciation for japanese esl/efl learners. *Asian EFL Journal*, 2000.

[8] D. Smith. An analysis of similarities between japanese and english. *The Classic Journal*, 2020.

[9] L. Sun, B. Jin, H. Yang, J. Tong, C. Liu, and H. Xiong. Unsupervised eeg feature extraction based on echo state network. *Information Sciences*, 475, 09 2018.

[10] L. S. Tapia, A. Gomez, M. Esparza, V. Jatla, M. Pattichis, S. Celedón-Pattichis, and C. LópezLeiva. Bilingual speech recognition by estimating speaker geometry from video data, 2021.

[11] A. Torres-García, C. A. Reyes-Garcia, and L. Villaseñor-Pineda. Toward a silent speech interface based on unspoken speech. In *BIOSIGNALS 2012 - Proceedings of the International Conference on Bio-Inspired Systems and Signal Processing*, 02 2012.

[12] S. Zhao and F. Rudzicz. Classifying phonological categories in imagined and articulated speech. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 992–996. IEEE, 2015.