ENS 491 – Graduation Project (Design)

Progress Report I

Project Title: Unsupervised SSVEP Signal Classification

Group Members:

Deniz Küçükahmetler

Pelinsu Çiftçioğlu

Giray Coşkun

Supervisor(s): Hüseyin Özkan

Date: 30.05.2021

1. PROJECT SUMMARY

Spellers via brain-computer interfaces (BCI) aim to establish a direct communication channel between the brain and a computer for individuals who need complete assistance to communicate (Güney et al., 2020). By using a visual presentation of a matrix of characters and numbers through a computer interface, it is possible to exploit the brain signals with electroencephalogram (EEG) and allow subjects to type solely based on their brain signals.

Visual presentation flickers with a unique frequency for each character generating Steady-State Visually Evoked Potentials (SSVEP) which are recorded by EEG. Target character identification is a multi-channel multi-class classification problem (Güney et al., 2020). State-of-art solutions include training with supervised data with both deep learning architectures and statistical methods. An unsupervised training approach is motivated to develop a model which is able to utilize unlabeled data. A deep learning architecture is developed with both unsupervised and supervised learning methodologies to improve information transfer rate (ITR) for target identification of spellers which utilizes Steady State Visually Evoked Potentials (SSVEP) based brain-computer interfaces (BCI). Model is developed with the inspiration from the properties of Canonical Correlation Analysis (CCA). The main objective is to maximize information transfer rate (ITR) which is a function of model accuracy and visual simulation duration (Güney et al., 2020). The model trained via unsupervised learning is expected to outperform training-free methods.

$P = accuracy\ of\ model,\ T = stimulation\ duration,\ M = number\ of\ classes$

$$I(P,T) = (log_2 M + P * log_2 P + (1 - P)log_2[\frac{1-P}{M-1}]) * \frac{60}{T}$$

Project utilizes 2 datasets Benchmark (Wang et al., 2017) and BETA (Liu et al., 2020). Benchmark includes 35 subjects each with 6 time blocks with a duration of 5 seconds stimulation for each character. Beta includes 105 subjects with 40 characters with 6 time blocks with a duration of 3 seconds for each character.

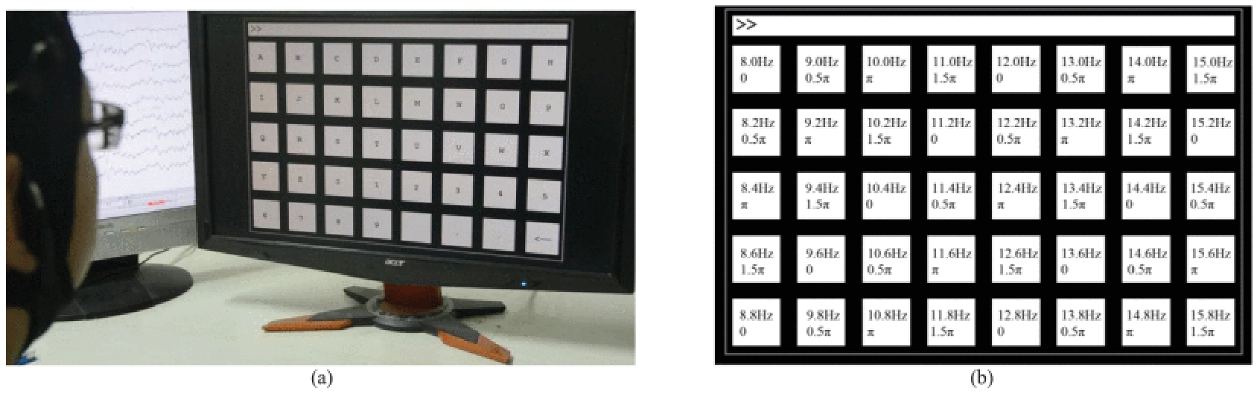


Figure 1: Visual Representation of the Characters Matrix

2. SCIENTIFIC/TECHNICAL DEVELOPMENTS

Spellers through brain computer interfaces established for people that are in need of complete assistance for communication (Güney et al., 2020). The main working principle of the spellers is that for a subject whose brain signals are tracked by EEG, the visual stimuli matrix as in the Figure 1, consisting of characters is presented and the signals collected are described as Steady State Visually Evoked Potentials (SSVEP)(Güney et al., 2020). They are processed to identify the target character and output through the interface. For identification of the SSVEP spellers, there are many methods from rule based models to various deep learning models that utilize different approaches (Kwak et al., 2017 ; Attia et al., 2018). This system has shown to have great promise as an alternative communication way for patients but yet the accuracy still remains an aspect that can be improved. Therefore, this project focuses on creating deep learning architectures for solving the target identification problem as accurately as possible by seeking high ITR.

Another important aspect to mention is that for the sake of practicality, sensitivity, accuracy and security, instead of collecting data from real subjects that requires a subtle EEG set up procedure and adequate participation, an already existing trusted data source has been used. The source provides various data-sets that are already used in the literature for similar research purposes and they can be downloaded from http://bci.med.tsinghua.edu.cn/download.html. This project will be utilizing two of the data-sets: BETA and Benchmark. They both consist of

SSVEP via EEG data collected from subjects while they are performing the target identification on the matrix by BCI.

First stage of the project has been completed by implementing a Fast Fourier Transform (FFT) and a Canonical Correlation Analysis (CCA) which is a training-free multivariable statistical method (Lin et al., 2007) to realize the motivation of target identification through SSVEP multi-channel signals and to have a deeper understanding towards application. This step also has established a base to compare with developed models. As the CCA method to predict the target character is a training-free method, the developed approach is expected to outperform this method as a base comparison.

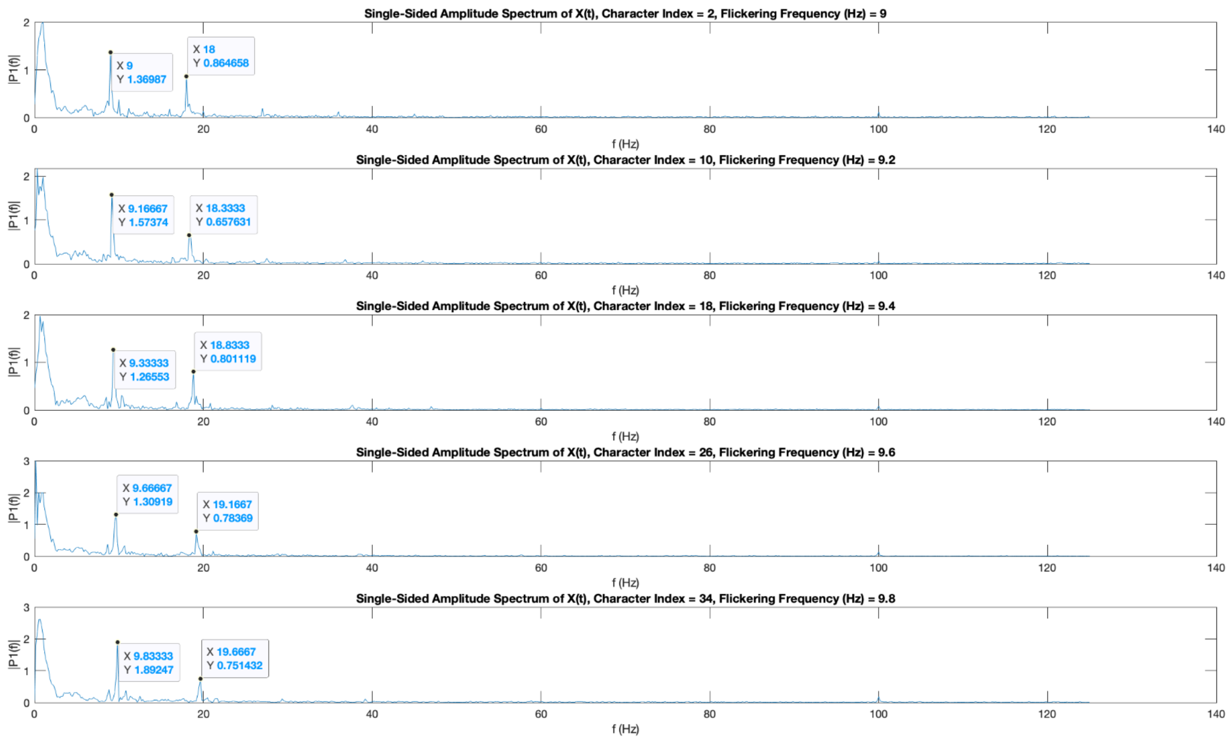
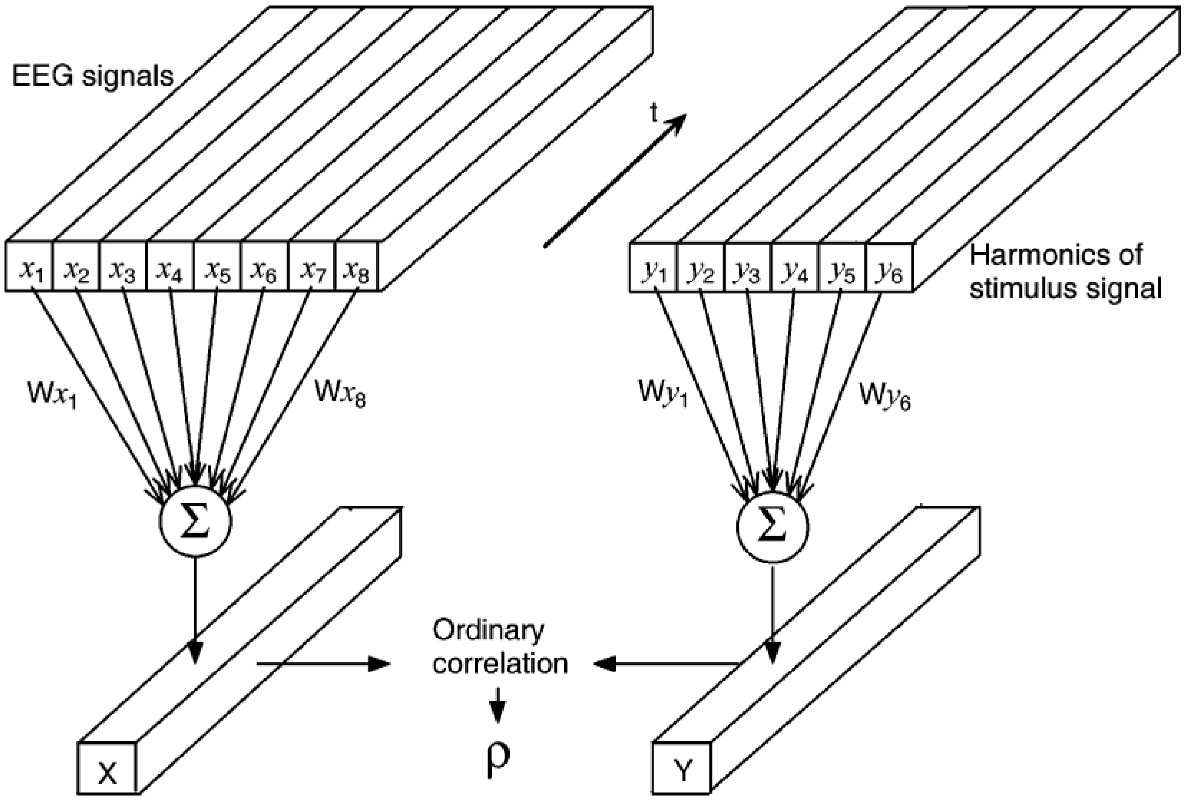


Figure 2: Single-Sided Amplitude Spectrum of Some Characters



EEG signals
t
Harmonics of stimulus signal
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
y_1 y_2 y_3 y_4 y_5 y_6
Wx_1
Wx_8
Wy_1
Wy_6
Σ
Σ
Ordinary correlation
ρ
X
Y

Figure 3: Canonical Correlation

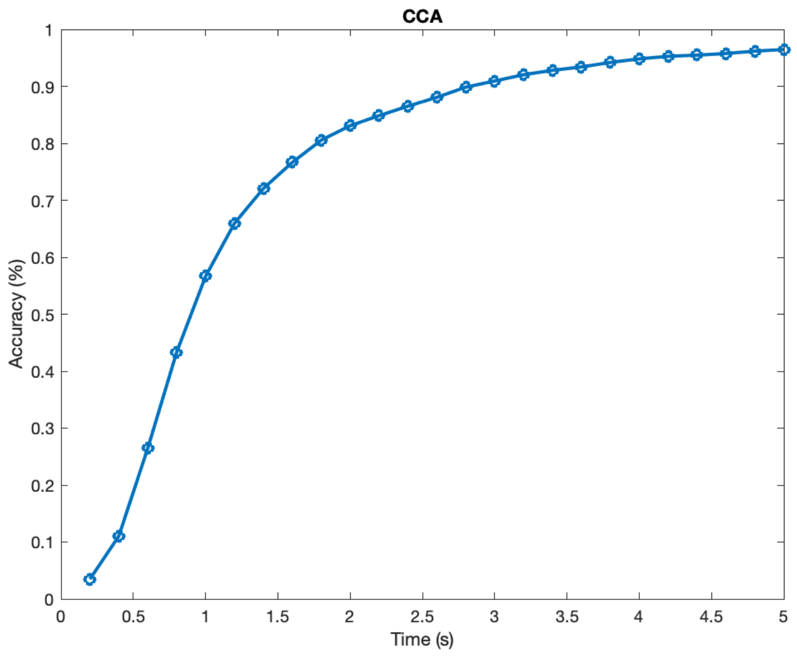


Figure 4: CCA Accuracy for 5 seconds

Second stage is to design the neural network architecture. Architecture is designed in M-branches, where the M is the number of classes. The input data is preprocessed which generates 3 subbands to capture different harmonic advantages. A filter is applied among subbands to choose the most beneficial harmonic combinations. Each branch is utilized to differentiate one target from others. At each branch filters are applied first through channels and then through signal length. The Relu activation function is applied to output. The output is carried over channel filter, signal filter and activation function procedures 3 times. The result of M branches are concatenated to be an input to the softmax layer. The designed architecture was first optimized by supervised learning.

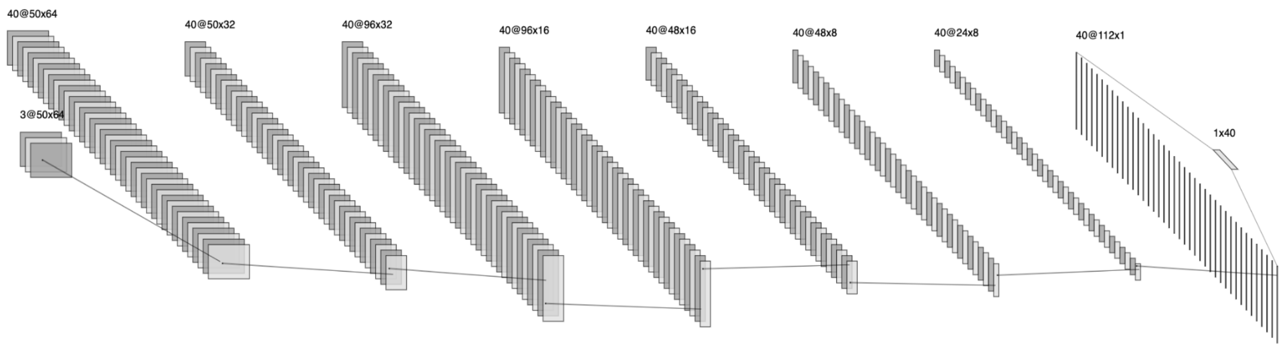


Figure 5: Network Architecture

a. First Layer for Harmonics (Sub-Band Combination):

As a data-driven approach; our model is expected to be trained to learn the most beneficial subband combination from data. For this reason, the preprocessing step utilizes a filter designed as zero-phase Chebyshev-Type 1 with filter order 2 and 1 dB pass band ripple. (MATLAB designfilt function). And the layer chooses a linear combination of input so the model can choose which harmonics to include by choosing appropriate weights.

b. **Second Layer for Channel Combination:**

   In literature; it is common to choose 9 channels (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, and O2) which include the most information regarding SSVEP; however each channel carries information which might be helpful to identify the target. Therefore our model is aimed to learn from all 64 channels to be able to choose a linear combination of channels. For this reason a filter is applied through channels to create d/2 number of combinations from d channels.

c. **Third Layer for Down-Sampling:**

   Over signal data a filter of size alpha*d/2 is applied to down-sample. alpha left to set as a hyper-parameter.

d. **Activation and Iteration over Filters:**

   A ReLu activation function with a higher slope is applied to output to capture non-linearity. From layer b to d is repeated N times to down-sample. Slope of the activation function is searched as a hyper-parameter.

e. **Concatenation and Softmax Layer:**

   Output of each branch is concatenated to apply a softmax layer to solve the multi-class classification problem.

The main architecture of the model is conserved. And the optimization procedure is exercised with two main aspects. Hyper-parameter selection and layer alteration for normalization, drop-out and activation layers. Hyper-parameter search is focused on alpha. By various alpha selections in the range of 1 to 15 best performance is acquired when alpha equals 3. Due to combat overfitting, a drop-out layer is introduced to different points of the model. However, a significant enhancement has not been acquired via drop-out layers. Batch normalization layer is applied at different points of the model to increase the accuracy of the model. And input is normalized to zero mean to eliminate signal dependent strengths of the input vector. The CCA method is also utilized to initialize the weights.

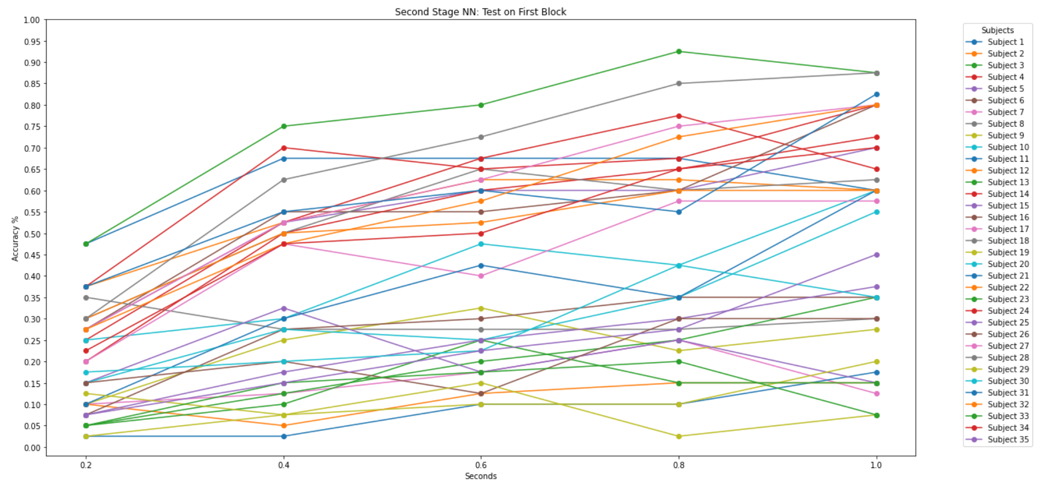
The results show that the network still needs to be improved. Even though the proposed architecture gives a better solution compared to CCA, the accuracies are not sufficient enough to compete with previously implemented networks. Nevertheless, the network is still open to improvements with minor modifications in the architecture and hyperparameter tuning. Following, the results will be interpreted. It is important to mention that the implementations are done on the Benchmark data set for the time being, however, the trials will continue for BETA in as the project progresses.

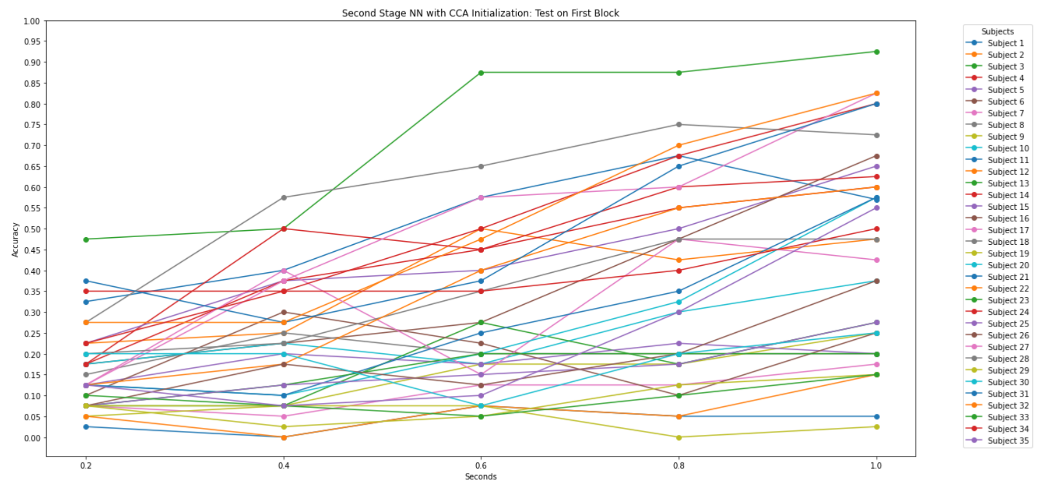
Following Figure 5 indicates the results that are derived from the implementation of the Neural Network both with CCA initialization and without. In the first stage, they both have been trained for 0.2, 0.4, 0.6, 0.8 and 1 seconds and tested on the first block which gave the accuracies below. In the second stage, the network is tested for each subject and the accuracy on each subject is also calculated and inserted into the graph to see the individual scores of each.

As can be seen below from the Figure 5, there is a huge difference between the accuracies of the trained network on different subjects on both initialization with CCA and without which is also called as second stage.

For the second stages, the average accuracy is calculated for both initialization with CCA and without. And overall results showed that at 1 second CCA results showed still a relatively higher accuracy. However all the accuracies at the end (at 1 second training) ranged between 40 and 60%. This means that the network is still open for improvements but started getting closer to the CCA results as the training interval increased. That can be seen in all of the graphs, as the training interval increased, the accuracy increased gradually and significantly.

Since the code takes too much time to run, only the test on first block results are included in the progress report. However, code still runs and after the testing on all blocks have been done, all the accuracies will be averaged for both first and second stage and the results will be discussed accordingly. Additionally, the ITR is calculated at the end of the implementation therefore to be informed for that, the code needs to be finished and the results of ITR matrix will be checked and included accordingly on the second progress report. Furthermore, following steps will be taken according to the results gathered in this stage and will be decided to take action either to continue improving the current network or take a different approach, specifically for the unsupervised learning.





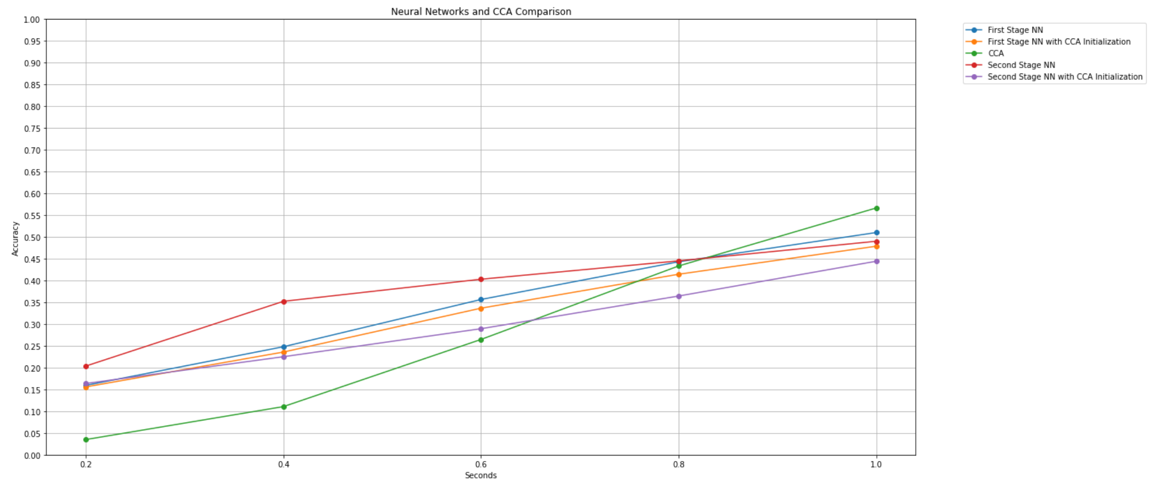


Figure 6: Result Graphs of Different Types of Neural Network and CCA. Results represent the testing on the first block.

3. ENCOUNTERED PROBLEMS

As mentioned before in the proposal, time is a very crucial factor in terms of limiting aspects of the project. The reason behind is that because of the nature of the deep learning architecture and their implementations, the training process is highly time consuming. For instance, for a network training in a single branch (for a single character) for 0.2 seconds, took approximately 6 to 9 hours on a personal computer with an average computing speed. As the computations got more expensive, such as training on 40 branches or training for more than 0.2 seconds, the personal computer's power and speed eventually became insufficient. At this stage in the project, a high performance computer (HPC) from the campus has been reached remotely and codes started to run on that. The problems did not finish at that point, since the team was lacking the knowledge of LINUX and how to handle the old version MATLAB for deep learning (since only the older version of MATLAB was available in HPC). However, with the technical support acquired from the Master's student as a great support in most of the aspects in the project, finally the codes started to run remotely without invading space on the personal computers.

At this stage, the trials for the neural network started which covered a lot of different parameters and architecture experimentation. Even though the HPC was a great help to run the codes, the process was still time consuming and the trials remained limited each week.
The optimization process was conducted in 0.2 seconds to represent a baseline accuracy. After the optimization process is nearly finalized, the architecture is also tested on bigger time intervals. However, based on the literature review, the expected accuracy could not be achieved from the network. For instance, the network accuracy was highest around 18% which gave the impression that it could reach up to 90% when the training captures 1 second which was clearly not correct.

As another option, it has also utilized another network where the weights of the first layer in the network initialized with the matrices derived from the CCA implementation of the data set. This idea was based on the inspiration of the current neural network architecture which was

influenced by the CCA approach of predicting the characters. However, this architecture with the current fine- tuning process did not result in any better results to a greater extent.
For the time being, the optimization process needed to be finalized and the network is finalized with the 2 different approaches: with CCA initialization and without CCA initialization. Both versions are trained for 0.2, 0.4, 0.6, 0.8 and 1 seconds. That caused another problem for time since it took too long to acquire the results again.

At this point, there are two different approaches that can be followed for the rest of the project. First is to try to have better combinations for the supervised neural network architecture with continuing the parameter exploration and fine-tuning. However, as discussed with the supervisor, this approach can be dangerous in terms of being unnecessarily time consuming. Second approach is to continue with the unsupervised learning and start optimizing an unsupervised neural network architecture.

In both of the cases, since these probabilities are being considered from the beginning and both of the learning types were the aim of the project, the progress of the project is still relevant to the original goals. The aim is to reach the best accuracy and speed as much as possible and the exploration and trials will take place as long as needed with the appropriate research. For this purpose, summer break will be used as an opportunity to work extra on the material and improve the project.

Finally, comparing where the progress is to the proposed time table in the beginning, there is a consistency to a great extent and most of the predicted tasks have been completed. However, instead of implementing optimization on the unsupervised learning model, the optimization continued for the supervised model which was a decision made voluntarily with accordance to the supervisor based on the outcomes received during the progress.

4. TASKS TO BE COMPLETED UNTIL PROGRESS REPORT II

TASK	MONTH																			
	JUNE				JULY				AUGUST				SEPTEMBER				NOVEMBER			
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
FINALIZING THE CURRENT NETWORKS	🟪	🟪	🟪	🟪																
LITERATURE REVIEW ON UNSUPERVISED LEARNING					🟦	🟦	🟦	🟦												
CREATING THE NEW ARCHITECTURE									🟦	🟦	🟦	🟦	🟦	🟦						
PARAMETER OPTIMIZATION / FINE-TUNING													🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

5. REFERENCES

Fast fourier transform - MATLAB fft. (n.d.). MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink. https://www.mathworks.com/help/matlab/ref/fft.html

Design digital filters - MATLAB fft. (n.d.). MathWorks - Makers of MATLAB and Simulink - MATLAB & Simulink. https://www.mathworks.com/help/signal/ref/designfilt.html#bt7qiz0

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. https://www.deeplearningbook.org/

Güney, O. B., Oblokulov, M., & Özkan, H. (2020). A Deep Neural Network for SSVEP-based Brain-Computer Interfaces. arXiv:2011.08562v2 [cs.LG]. https://arxiv.org/abs/2011.08562

Lin, Z., Zhang, C., Wu, W., & Gao, X. (2007). Frequency recognition based on canonical correlation analysis for SSVEP-based BCIs. IEEE Transactions on Biomedical Engineering, 54(6), 1172-1176. https://doi.org/10.1109/tbme.2006.889197

Liu, B., Huang, X., Wang, Y., Chen, X., & Gao, X. (2020). BETA: A large benchmark database toward SSVEP-BCI application. Frontiers in Neuroscience, 14. https://doi.org/10.3389/fnins.2020.00627

M. Attia, I. Hettiarachchi, M. Hossny, and S. Nahavandi, "A time domain classification of steady-state visual evoked potentials using deep recurrent-convolutional neural networks," IEEE International Symposium on Biomedical Imaging, pp. 766–769, 2018.

N.-S. Kwak, K.-R. Müller, and S.-W. Lee, "A convolutional neural network for steady state visual evoked potential classification under ambulatory environment," PLOS ONE, vol. 12, no. 2, pp. 1– 20, 2017.

Wang, Y., Chen, X., Gao, X., & Gao, S. (2017). A Benchmark Dataset for SSVEP-Based Brain–Computer Interfaces. IEEE Transactions On Neural Systems and Rehabilitation Engineering, 25(10), 1746-1752. https://doi.org/10.3389/fnins.2020.00627