Saccade Detection with Multi-Task Learning on EEG

Abstract

Eye-tracking technologies are of great importance in various fields including diagnosis and behavioral sciences. As a non-invasive and low-cost measure of brain dynamics, electroencephalography (EEG) provides both research and application value in eye-tracking recognition. The objective of our project is to create one unified model to get all the information of the saccade from datasets which only have partial information. To the greatest extent possible, we have achieved comparable results of accuracy on all tasks as implemented in the literature. We have extended the work to implement multitask classification using pseudo-label generation. We obtained $\mathbf{92.88}\%$ accuracy for left/right classification, mean squared error(RMSE) of **27.29 mm** for Amplitude, **0.98 rad** for Angle, and Euclidean distance of **54.45 mm** for the absolute position. Some of the results beat the baseline performance using one single model. The next step is to update the pseudo labels and quantify the weights based on probability. Future work will focus on hyperparameters tuning, reducing noisy labels, and evaluating options for multitasking paradigms.

1 Introduction

Tracking eye position is important in various behavioral sciences, including assistive technology and user experience. It also aids in the diagnosis of neurological disorders such as Autism, Obsessive Compulsive Disorder, Schizophrenia, Parkinson disease, and others. Tracking eye movement has also paved the way for eye-tracking technology with EEG — a non-invasive, minimally restrictive, and relatively low-cost measure of mesoscale brain dynamics with high temporal resolution — as a result of technological advancements.

Researchers can study perceptual, attentional, and cognitive processes in naturalistic situations by combining behavioral data from eye tracking with neurophysiological markers provided by EEG. This may hasten scientific discovery about human behavior, neurological, and psychiatric diseases, particularly during free viewing of complex stimuli (i.e. naturalistic paradigms) and in clinical settings where installing an eye tracker is impractical. The EEGEyeNet dataset was gathered to further the investigation.

In the classic machine learning approach such as emotion detection using EEG signals, which mostly depends on human experience; the feature extraction procedure is often time-consuming and labor-intensive. Zhang et al. [2020] has presented that end-to-end deep learning algorithms were created as an effective means of resolving this issue for the use of time-frequency spectrum and raw signal characteristics. Therefore, to enhance eye-tracking performance using only EEG signals, we have decided to implement:

1. Pseudo Label Generation
2. One Unified Multi-Task Learning Model

2 Literature Review

EEG saccade prediction is an active research topic with applications in advertising Okada et al. [2018], human behavior analysis Chaaraoui et al. [2012], and human-computer interaction Majaranta and Bulling [2014], to name a few. Additionally, Bulling et al. [2010] showed that activity recognition from eye movements is possible. Some models including work from Nakashima et al. [2015], Koch and Ullman [1987] use saliency maps to predict gaze location, while others use machine learning to infer gaze position from oblique data. Son et al. [2020] and LaConte et al. [2006] used functional magnetic resonance imaging (fMRI), while Krafka et al. [2016] used webcam images. O'Connell and Chun [2018] did the same by reconstructing fixations maps from fMRI data, which can be used to forecast eye movement patterns. A fascinating finding from Krafka et al. [2016] suggests that gaze direction may be directly deduced from EEG data. While studies have shown that combining EEG and Eye Tracking (ET) can improve performance in a variety of tasks when compared to using either modality alone, to our knowledge this work was the first one that used deep learning to estimate eye position from EEG.

Kastrati et al. [2021] worked on Gaze prediction from EEG measurements using the dataset EEGEyeNet. The benchmark consists of three increasingly challenging tasks: left-right, angle-amplitude, and absolute location. Extensive trials on this benchmark were conducted to provide robust baselines based on classic machine learning models (SVM, KNN, Linear Regression et.c.) and huge neural networks (CNN, Inception Time, EEGNet et.c.). Despite that they achieved excellent accuracy in decoding the left-right task, the performances of the other two tasks remained unsatisfying. In this reference they are trying to reproduce and improve this research.

Ruder [2017] They presented a review of the literature, describing the two most popular Multitask Learning techniques used in deep learning, and talk about current developments. Specifically aims to assist ML practitioners in applying Multitask Learning and its applications.They also focus on how Multitask Learning functions and provide advice for picking suitable auxiliary tasks.

Zhang et al. [2022], hypothesized that when viewing video content, brain connection activation aspects are highly connected to eye movement characteristics. They looked at the connection between brain connectivity (strength and directionality) and eye movement characteristics (left and right pupils, saccades, and fixations). They used Russell's two-dimensional model which is the emotion circumplex model Blunt et al. [2008]. According to this model, emotions are distributed in a circular, two-dimensional space with arousal and valence dimensions. To investigate the association between brain activity and eye movement, they employed supervised learning (1D-CNN) to identify emotional and non-emotional states and extract eye movement feature values associated with distinct emotional states.

Wolf et al. [2022] implemented DETRtime, a new framework for time-series segmentation that generates ocular event detectors that do not require extra recorded eye-tracking modalities and depend purely on EEG data. DETRtime delivers cutting-edge ocular event detection performance across many eye-tracking experiment paradigms. Furthermore, they show that their model generalizes effectively in the task of EEG sleep stage segmentation.

Görnitz et al. [2014] showcased that the oversimplified assumption of independently distributed noise fails in many real-world situations, because labels can have structured, systematic noise. For instance, training data for brain-computer interface applications is frequently the product of protracted experimental sessions in which participants' levels of attention can fluctuate. Because most machine learning techniques assume independent and identically distributed label noise in such application scenarios, structured label noise will be problematic. In this article, they described a brand-new learning and assessment methodology that takes systematic label noise into account. The main component of which is a brand-new, latent variable-aware version of the one-class SVM with support vector data description. Based on the literature we are planning to implement multitask model to predict all labels of EEG data with the limited known labels.

3 Dataset Description

Given the numerous benefits that EEG-based eye tracking can bring to various domains, Kastrati et al. [2021] have presented a benchmark for evaluating gaze estimation from EEGEyeNet. The dataset we have chosen for this project, EEGEyeNet, is made up of 356 different subjects' simultaneous

electroencephalography (EEG) and eye-tracking (ET) recordings from three different experimental paradigms. It contains a dataset of high-density 129-channel (including 1 reference channel) EEG data synchronized with video-infrared eye tracking making up more than 47 hours of recording. EEGEyeNet is a massive dataset of EEG data that has been synchronized with precise eye-tracking recordings, which is suitable to investigate deep learning models.

This benchmark consists of three tasks of increasing difficulty and is intended to be a tool for conducting comparable and reproducible research on gaze estimation from EEG data. A cleaned and raw dataset has been provided for EEGEyeNet. Figure 1 shows the experiment setup and an example of the data used for classification.

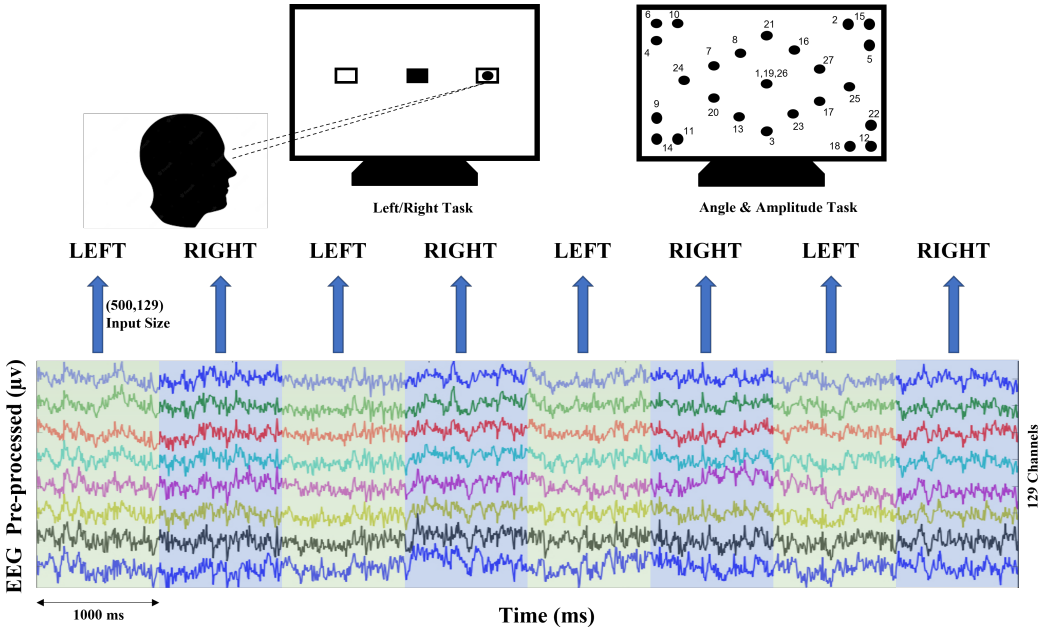

Figure 1: Data and task Description

The first task is the left-right binary classification task. Based on the instruction on the screen in front of the subjects, they performed either a left or a right saccade. A total of 30, 842 samples are collected for classification. The performance of a classifier is measured by classification accuracy.

The second task is to determine the angle (radian) and amplitude (millimeters) of a saccade. In this task, subjects were asked to fixate on one target out of 25 discrete points that were sequentially showing up in the screen. The continuous angle and amplitude of the relative change of the gaze position are predicted by the model. For this task, the evaluation metric is the square root of the mean squared error (RMSE).

The third task is absolute position estimation. The aim is to continuously decode the absolute gaze position as subjects scan across the screen. This task is also the most challenging one. The x and y position is estimated by the model. The evaluation metric is the euclidean distance between the actual and the estimated gaze position.

The input to the model is scalp EEG signals with a dimension of 500x129, which contains 500 time points of EEG recordings from 129 electrodes. Given that the sampling rate is 500 Hz, each input contains 1s long data.

4 Baseline Selection

Previous work implemented 5 deep learning architectures on the eye movement EEG dataset, which are Vanilla CNN, Pyramidal CNN, EEGNet, InceptionTime, and Xception. Since Vanilla CNN, Pyramidal CNN, and Xception lead to the best results, we will provide details for these three architectures. The baseline performance is listed in Table B.1, B.2, and B.3 shown in appendix.

4.1 12 - Layer Vanilla Convolution Neural Network (CNN)

A standard 12 - layer 1D CNN model was used with additive residual connections around blocks of three layers. Each layer consisted of convolution, batch normalization, ReLU activation, and max pooling. In the convolution layers, 16 filters of kernel size 64 were applied. For the pooling operation, a kernel of size 2 and stride 1 was used. Each residual connection performed a convolution followed by batch normalization.

The block structure of three layers with residual connection is shown in Figure A.1 in appendix. Each layer can be represented as:

$$Layer(x) = Pool_{Max}(\delta(BN(Conv_{64}(x))))$$

Where,
$Conv_n = W_n * x + b$ refers to 1D convolution with a kernel size of n
$*$ refers to convolution
BN refers to 1D batchnorm operation
$\delta()$ refers to ReLU activation
$Pool_{Max}$ refers to Max Pooling

The shortcut operation can be represented as:

$$Shortcut(x) = BN(Conv_{64}(x))$$

Then a single residual block containing three layers can be represented as:

$$output = \delta(Shortcut(x) + Layer(Layer(Layer(x))))$$

4.2 Pyramidal CNN

A 6-layer CNN model with an inverse pyramidal shape was applied. The number of filters was 16 in the first convolution layer and increased by 16 in each following layer. The kernel size was 16 for all the convolution layers and there were no residual connections. The batch normalization, ReLU activation, and max pooling settings were the same as the CNN model described above. In appendix Figure A.2 shows the overall structure of the model.

Each layer in the PyramidalCNN can be represented as:

$$Layer(x) = BN(Conv_{16}(x))$$

Where,
$Conv_n = W_n * x + b$ refers to 1D convolution with a kernel size of n
$*$ refers to convolution
BN refers to 1D batchnorm operation.

Then PyramidalCNN model containing six layers can be represented as: Where $Layer_o$ refers to each layer with the number of output channels o.

$$output = Layer_{96}(\delta(Layer_{80}(\delta(Layer_{64}(\delta(Layer_{48}(\delta(Layer_{32}(\delta(Layer_{16}(x))))))))))$$

Where,
$\delta()$ refers to ReLU activation
The weights in each layer are increasing as shown below, where the first dimension represents the filter size and the second dimension represents the kernel size.

$$W_1 = (16, 16), W_2 = (32, 16), W_3 = (48, 16), W_4 = (64, 16), W_5 = (80, 16), W_6 = (96, 16),$$

4.3 Xception

Chollet [2017] proposed Xception. This model shared the same general structure as the CNN model introduced above, with 18 layers and residual connections after every three layers. Each layer had a 1D depth-wise separable convolution, with 64 filters and a kernel size of 40, followed by batch normalization and ReLU activation. After the convolution layers, 1D average pooling with a kernel size of 2 was applied, then followed by a fully connected linear layer to get one output for each variable.

The block structure of three layers with residual connection is shown in Figure A.3 in appendix. Each layer can be represented as:

$$Layer(x) = BN(Conv_1(Conv_{40}(x)))$$

Where
$Conv_n = W_n * x + b$ refers to 1D convolution with a kernel size of n
$*$ refers to convolution
BN refers to 1D batchnorm operation.

The shortcut operation can be represented as:

$$Shortcut(x) = BN(Conv_{40}(x))$$

Then a single residual block containing three layers can be represented as:

$$output = \delta(Shortcut(x) + Layer(\delta(Layer(\delta(Layer(x))))))$$

Where,
$\delta()$ refers to ReLU activation

4.4 Hyperparameters

According to Kastrati et al. [2021], hyperparameters including learning rate were tuned based on the Left-Right task, and the same parameter values were used in all other tasks. Binary cross entropy (BCE)loss was used to train the models for the Left-Right task, and mean square error (MSE) loss was used for the other two tasks. There were 50 training epochs for every model. Adam optimizer with a learning rate of 0.0001 and early stopping with patience of 20 epochs on the validation sets was applied to all model training.

5 Model Description

The goal of our project is to propose one unified model that is able to describe the saccade, the information including Left/Right, angle, amplitude, and absolute position. There are two main steps in this work. The first one would be generating complete labels for all datasets. The second one is then to achieve multi-task learning with the generated dataset.

5.1 Label Generation

The EEGEyeNet dataset provides different labels for different tasks. To create labels for all the samples, we passed three datasets to our pre-trained models described in Section 4 and 5 to generated all the labels (Left/Right, angle, amplitude, and absolute position) for every dataset. For each task, the model with best performance was chosen: CNN was used for Left/Right, Angle, and Abosulte Position label generation; Xception was used for Amplitude label generation. If the labels to be predicted are not available in that dataset, the predicted labels are stored with a flag indicating that it is generated; otherwise, the original labels are stored with a flag indicating that it is the true label. The general label generation process is shown in Figure A.4 in appendix.

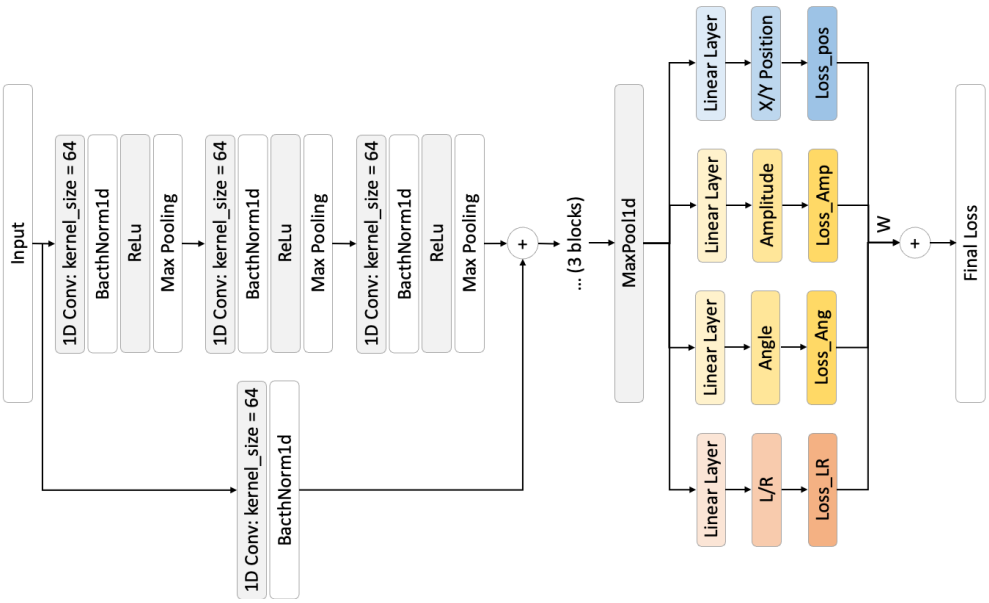

Figure 2: Multi-Task Model Structure

5.2 Multi-Task Learning

The multi-task model is based on one of the baseline models. We use 12-layer standard CNN as a feature extractor since it gives the best performance overall, after which 4 separate classification layers are applied in parallel to get the output of 4 labels (Left/Right, Angle, Amplitude and Absolute position). The overall architecture is shown in Figure 2.

Distinct loss functions are used for each label. For Left/Right label, binary cross entropy (BCE) loss is used, for Angle label, mean squared error (MSE) loss is applied, and for other two labels, Amplitude and X/Y position, L1 loss is applied. The reason for using L1 loss instead of MSE loss is because of the wide range of these two labels. MSE loss would dominate in the final loss function and cause weak training on other variables. With L1 loss and proper scaling factors, the losses can be controlled at comparable levels with the other ones. The final loss of this model is a weighted average of these four losses. True labels have higher weight since they are the ground truth and therefore are more reliable, while the pseudo labels are assigned with lower weights. The ranges of different labels vary a lot, a scaling factor is applied on the losses to keep losses from different tasks at a similar level. For example, the range of Amplitude is 0 to 1200, while the LR label is either 0 or 1. Therefore the loss of Amplitude would be dominant without scaling. The scaling factor is defined by the inverse of the label range, as shown in Equation 1.

$$\mu_i = 1/(max(l_i) - min(l_i)) \tag{1}$$

, where μ_i refers to the scaling factor for i$^{\text{th}}$ label, and l_i refers to the corresponding label values.i here ranges from 1 to 4, referring to 4 different labels.

The final loss can then be represented as Euqation 2

$$FinalLoss = \sum_{i=1}^{4} \mu_i (Loss(l_i^{True}) W^{True} + Loss(l_i^{Pseudo}) W^{Pseudo}) \tag{2}$$

, where the superscript $True$ refers to true labels, and $Pseudo$ refers to pseudo labels. W^{True} and W^{Pseudo} are the weights for true loss and pseudo loss.

The learning rate of the model is 0.0001 and is kept constant across the training process. Adam optimizer and mixed precision training is applied. The model is trained for 10 epochs. The weights

applied to true and pseudo loss are 0.8 and 0.2 respectively. Multiple combinations of hyper parameters and other model specifications including weights, scaling factors, loss functions and model architectures have been applied, and the results will be discussed in the next section.

5.3 Further Extension

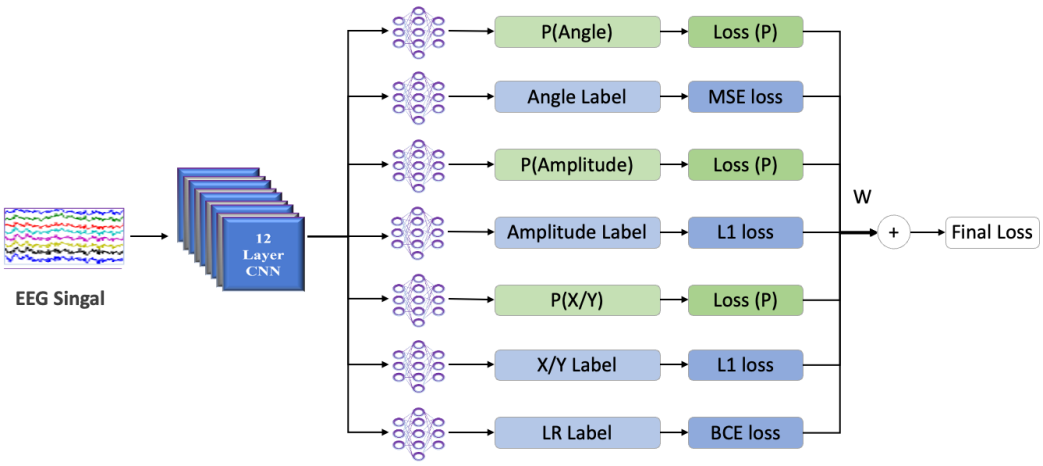

Figure 3: Multi-Task Model with Pseudo Labels Updated and P generated

Based on our current multi-task model, we have had a in-depth discussion about further extension of this work, and we have started working on it. As shown in Figure 3, the next step is to update pseudo labels during training and weight the loss according to the probability. To get an estimated probability to measure how reliable the predicted label is, the idea is to add three linear layers outputting the p value for three regression tasks (Angle, Amplitude, and Absolute position), and the loss of the probabilities is then computed by comparing the output probability and the corresponding loss value. After proper training, the probability is expected to match the loss value. High loss value corresponds to a higher probability, referring to a less accurate pseudo label generation. For the classification task (Left/Right), the probability is simply 0.5, so there is no need to add an additional layer for Left/Right probability estimation.

During the training process, the pseudo labels keep updated based on the model output, and the weights of the pseudo loss values are determined according to the estimated probability. We expect to further improve the model performance with more reliable pseudo labels, also with quantified weights for pseudo and true labels.

However, there still exist several challenges, including a proper way to include the loss for probabilities in the final loss. Besides, as the number of outputs of the multi-task model increases, the difficulty of training model to better predict every label also increases.

6 Results and Discussion

6.1 Baseline Implementation

Table 1 gives a description of the data size we used as training, validation and test sets. 70% of the entire data is used for training, 15% for validation, and the left 15% is used for test as described in the paper.

Table B.1, B.2, and B.3 in appendix show the comparison between reproduction results when we implement the same models as described above and their baseline performance.

According to the table results and Figure 4, we get comparable performance as the results provided in the EEGEyeNet paper with respecting to all three tasks.

Table 1: Benchmark Data

	# Participants				# Samples			
Dataset	Total	Train	Validation	Test	Total	Train	Validation	Test
Left-Right	329	229	50	50	30842	21042	4980	4820
Angle/Amplitude	27	19	4	4	17830	12275	2836	2719
Abs. Position	27	19	4	4	21464	14706	3277	3481

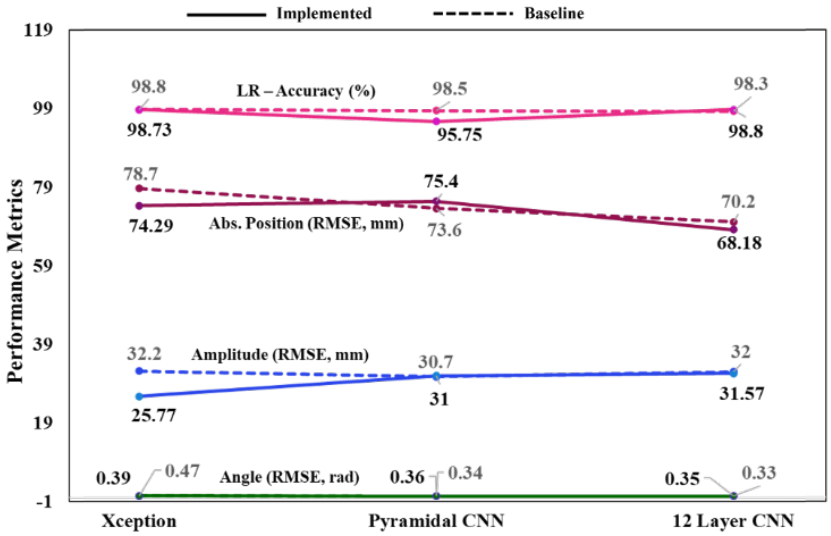

Figure 4: Baseline Implementation Results

6.2 Pseudo Label Generation

The pseudo label was generated for all the tasks using the implemented Xception model. Then those labels were used in training of the multi task model. We cross verified the accuracy of the generated pseudo label with that of the predicted label logically. In appendix, figure A.5 shows that the generated labels well represent the real condition and is reliable for later training.

6.3 Multi-Task Model Performance

The multi-task model performance was measured for all the generated pseudo labels and the true labels. The performance was shown as the bar chart. We did seven experiments with different combinations of feature extractor model and the Linear classification layer, and the results are shown in Table 2. We tried different parameters of weights for true and pseudo labels, scaling of the loss function, and different loss functions for different tasks.

Figure 5 shows the performance comparison between our proposed model and the baseline models. With one single model, we achieve comparable results as reported in the reference paper in all tasks, and we are able to get better than the baseline performance in absolute position prediction.

7 Conclusion and Future Work

The research work presented has a modern approach for saccade detection. The saccade detection technique used here had 129 electrode to capture EEG signals. The work was improved based on the concepts of multi-task learning and the existing eye movement detection techniques as described in literature. We have implemented the one unified multi-task model which is capable of detecting pseudo labels without the actual information. We have achieved better results with our multi-task model by detecting all the pseudo labels for the three different tasks Left/right, Position, Angle, and Amplitude. With the comprehension of the idea of pseudo-label generation and multi-task learning, we try to get more information from EEG signals with limited resources. Multiple output labels from

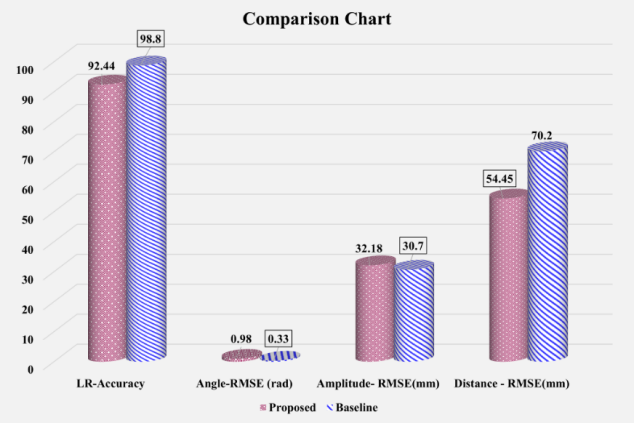

Figure 5: Comparison between Baseline and Proposed Multi-Task Model

a single input label can help us reduce training time as otherwise, have to be trained independently for all tasks. The generation of pseudo labels also increases the sample size, as its a combination of three datasets.

Table 2: Multitask Model Performance

Exp.	**FE Model**	**Weights**		**Loss Scaling**	**Loss Function**				**Performance Metrix**			
		True Label	Pseudo Label		LR	Amplitude	Angle	Distance	LR Accuracy (%)	Amplitude RMSE (mm)	Angle RMSE (rad)	Euclidean distance (mm)
1	CNN	0.8	0.2	No	BCE	MSE	MSE	MSE	91.46	**27.29**	1.81	54.71
2	CNN	0.8	0.2	Yes	BCE	MSE	MSE	MSE	58.76	148	1.7	268
3	Xception	0.8	0.2	Yes	BCE	L1	MSE	L1	58.10	148.75	1.7	268
4	**CNN**	**0.8**	**0.2**	**No**	**BCE**	**L1**	**MSE**	**L1**	**92.44**	**32.18**	**0.98**	**54.45**
5	CNN	0.8	0.2	Yes	BCE	L1	MSE	L1	92.88	42.82	1.01	68.71
6	CNN	0.1	0.9	Yes	BCE	L1	MSE	L1	88.33	360	1.7	112
7	CNN	0.1	0.9	Yes	BCE	L1	MSE	L1	91.58	45.84	1.2	71.35

However, our current work has some limitations, which prevents the proposed model from outperforming the baseline results. This multi-task model utilizes a common feature extractor in which there may be some task-specific information missing. This can be overcome by changing the feature extractor with that of the task-specific Experts like MMOE (Multi gate Mixture of experts) or PLE (Progressive Layer Extraction). The next limitation of our work is the presence of pseudo-label noise. This can be reduced using the method introduced in the paper Görnitz et al. [2014] as described in the literature.

Another limitation of this work includes simple 1D-CNN based feature extractor. As described in the literature, existing EEG-based eye movement detection techniques are mainly based on temporal dynamics of EEG signal by applying a 1-dimensional CNN network on time series EEG data. However, it has been confirmed that changes in the frequency component of EEG also contribute to saccadic activities according to Plöchl et al. [2012] and Mammone et al. [2020]. These pieces of evidence indicate the potential value of investigating saccade-related brain activity in the time-frequency domain.

Our future plan is to reduce the noise generated by pseudo labels and to try quantify the confidence level of generated labels. Also we plan to enhance the multi-task model by trying more combination of feature extractors, different task-specific weight scaling, and classification layers with wider or deeper structures.

References

M. O. Blunt, J. C. Russell, M. d. C. Giménez-López, J. P. Garrahan, X. Lin, M. Schroder, N. R. Champness, and P. H. Beton. Random tiling and topological defects in a two-dimensional molecular network. *Science*, 322(5904):1077–1081, 2008.

A. Bulling, J. A. Ward, H. Gellersen, and G. Tröster. Eye movement analysis for activity recognition using electrooculography. *IEEE transactions on pattern analysis and machine intelligence*, 33(4): 741–753, 2010.

A. A. Chaaraoui, P. Climent-Pérez, and F. Flórez-Revuelta. A review on vision techniques applied to human behaviour analysis for ambient-assisted living. *Expert Systems with Applications*, 39(12): 10873–10888, 2012.

F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

N. Görnitz, A. Porbadnigk, A. Binder, C. Sannelli, M. Braun, K.-R. Mueller, and M. Kloft. Learning and Evaluation in Presence of Non-i.i.d. Label Noise. In S. Kaski and J. Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 293–302, Reykjavik, Iceland, 22–25 Apr 2014. PMLR. URL `https://proceedings.mlr.press/v33/gornitz14.html`.

A. Kastrati, M. M. B. Płomecka, D. Pascual, L. Wolf, V. Gillioz, R. Wattenhofer, and N. Langer. Eegeyenet: a simultaneous electroencephalography and eye-tracking dataset and benchmark for eye movement prediction. *arXiv preprint arXiv:2111.05100*, 2021.

C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Matters of intelligence*, pages 115–141. Springer, 1987.

K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba. Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2176–2184, 2016.

S. LaConte, S. Peltier, K. Heberlein, and X. Hu. Predictive eye estimation regression (peer) for simultaneous eye tracking and fmri. In *Proc. Intl. Soc. Magn. Reson. Med*, volume 2808, page 13, 2006.

P. Majaranta and A. Bulling. Eye tracking and eye-based human–computer interaction. In *Advances in physiological computing*, pages 39–65. Springer, 2014.

N. Mammone, C. Ieracitano, and F. C. Morabito. A deep cnn approach to decode motor preparation of upper limbs from time–frequency maps of eeg signals at source level. *Neural Networks*, 124: 357–372, 2020.

R. Nakashima, Y. Fang, Y. Hatori, A. Hiratani, K. Matsumiya, I. Kuriki, and S. Shioiri. Saliency-based gaze prediction based on head direction. *Vision research*, 117:59–66, 2015.

G. Okada, K. Masui, and N. Tsumura. Advertisement effectiveness estimation based on crowdsourced multimodal affective responses. In *Proceedings of the IEEE Conference on computer vision and pattern recognition workshops*, pages 1263–1271, 2018.

T. P. O'Connell and M. M. Chun. Predicting eye movement patterns from fmri responses to natural scenes. *Nature communications*, 9(1):1–15, 2018.

M. Plöchl, J. P. Ossandón, and P. König. Combining eeg and eye tracking: identification, characterization, and correction of eye movement artifacts in electroencephalographic data. *Frontiers in human neuroscience*, 6:278, 2012.

S. Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

J. Son, L. Ai, R. Lim, T. Xu, S. Colcombe, A. R. Franco, J. Cloud, S. LaConte, J. Lisinski, A. Klein, et al. Evaluating fmri-based estimation of eye gaze during naturalistic viewing. *Cerebral Cortex*, 30(3):1171–1184, 2020.

L. Wolf, A. Kastrati, M. B. Płomecka, J.-M. Li, D. Klebe, A. Veicht, R. Wattenhofer, and N. Langer. A deep learning approach for the segmentation of electroencephalography data in eye tracking applications. *arXiv preprint arXiv:2206.08672*, 2022.

J. Zhang, S. Park, A. Cho, and M. Whang. Recognition of emotion by brain connectivity and eye movement. *Sensors*, 22(18):6736, 2022.

Y. Zhang, J. Chen, J. H. Tan, Y. Chen, Y. Chen, D. Li, L. Yang, J. Su, X. Huang, and W. Che. An investigation of deep learning models for eeg-based emotion recognition. *Frontiers in Neuroscience*, 14:622759, 2020.

A Figures

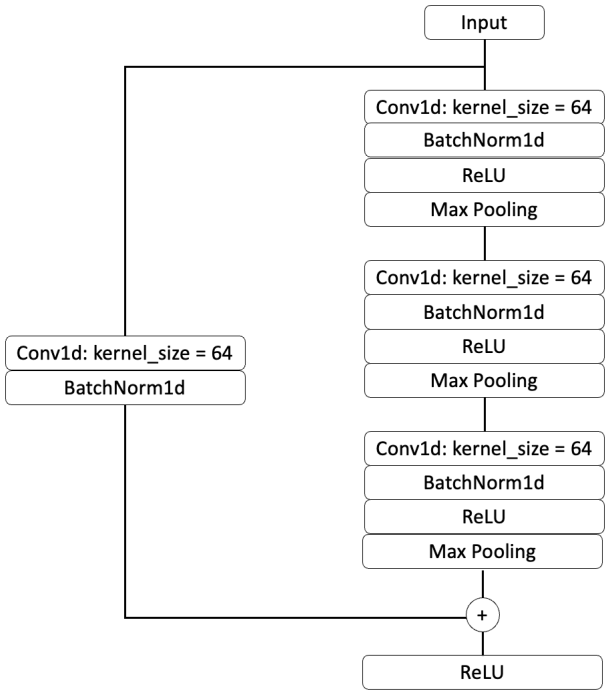

Figure A.1: CNN Single Block

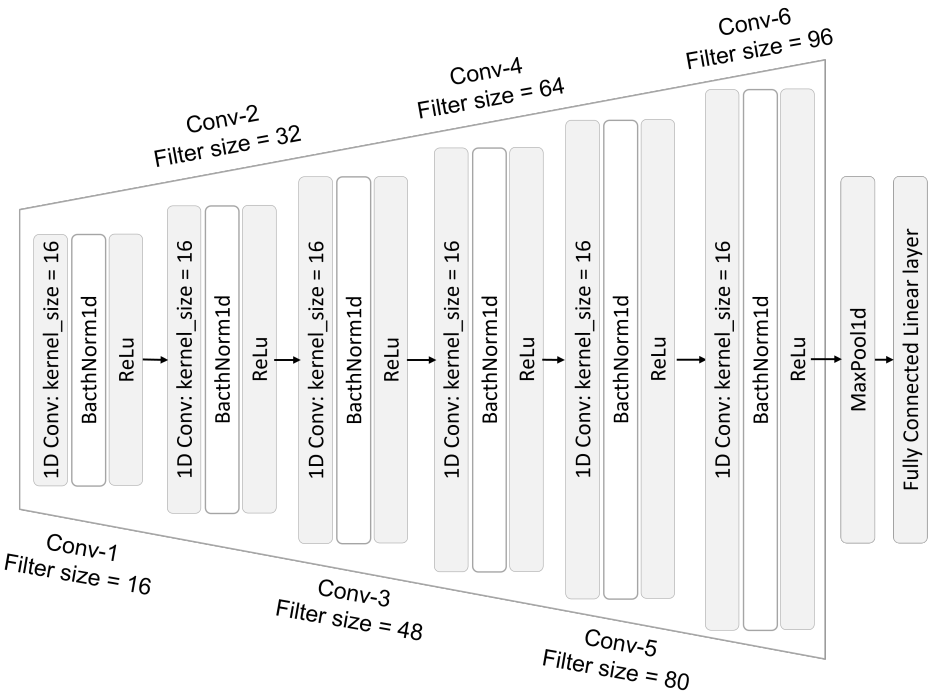

Figure A.2: PyramidalCNN Structure

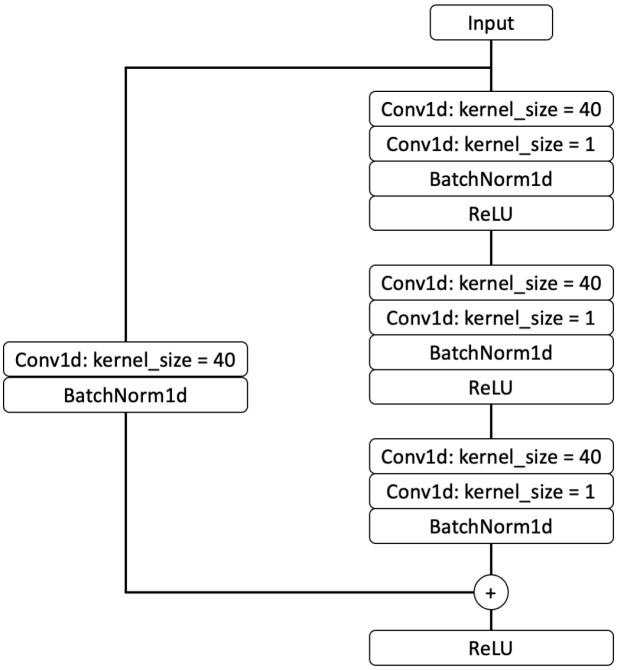

Figure A.3: Xception Single Block

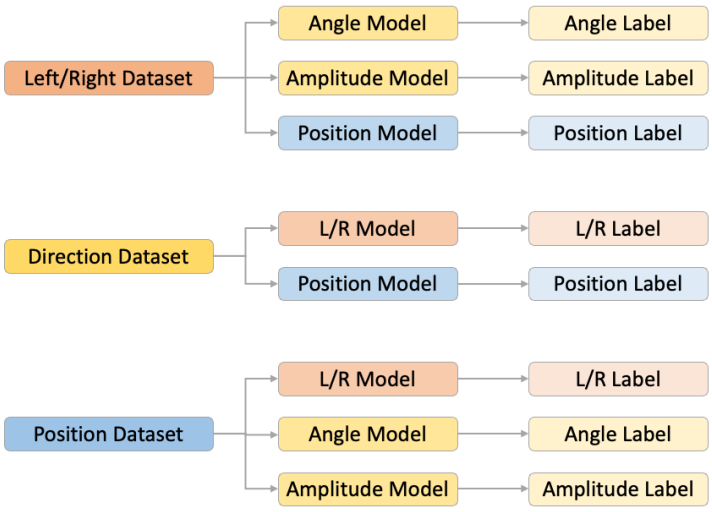

Figure A.4: Pseudo Label Generation

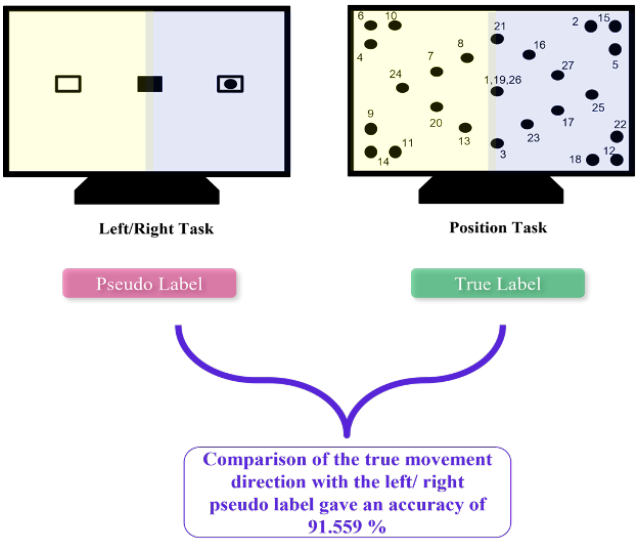

Figure A.5: Pseudo Label Accuracy

B Tables

Table B.1: Performance Comparison of Left/Right Task

	Left/Right-Accuracy %	
Model	Baseline	Obtained
CNN	98.3	98.80
PyramidalCNN	98.5	95.75
Xception	**98.8**	98.73

Table B.2: Performance Comparison of Angle/Amplitude Task

	Angle-RMSE (rad)		Amplitude-RMSE (mm)	
Model	Baseline	Obtained	Baseline	Obtained
CNN	**0.33**	0.35	32	31.57
PyramidalCNN	0.34	0.36	**30.7**	31
Xception	0.47	0.39	32.2	25.77

Table B.3: Performance Comparison of Absolute Position Task

	Abs.Position-RMSE (mm)	
Model	Baseline	Obtained
CNN	**70.2**	68.18
PyramidalCNN	73.6	75.4
Xception	78.7	74.29