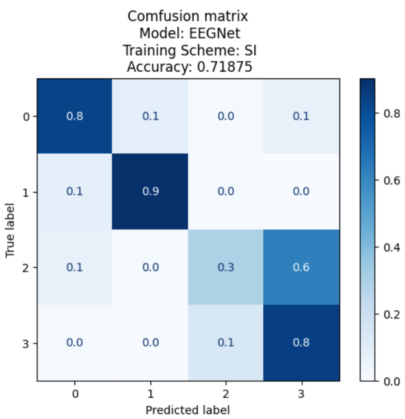
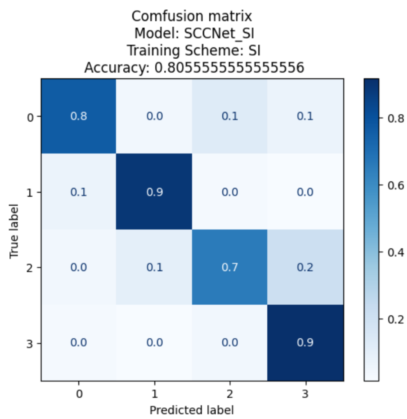
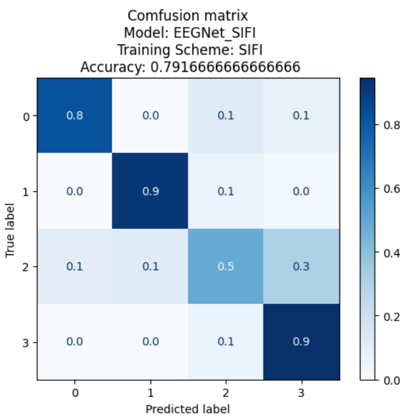
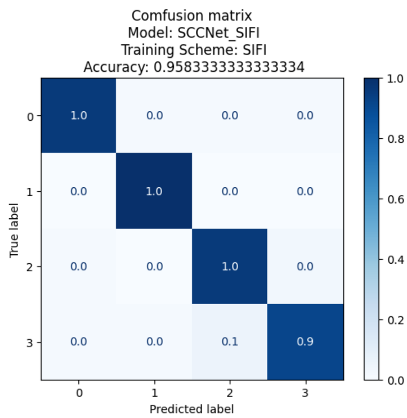
CSIC30100: Brain Computer Interface **(Due: 12/15/2023)**

HW3: Deep Learning for BCI

Instructor: Chun-Shu Wei
TA: 11055090 王昱力

1.1. **Requirements** (60%)

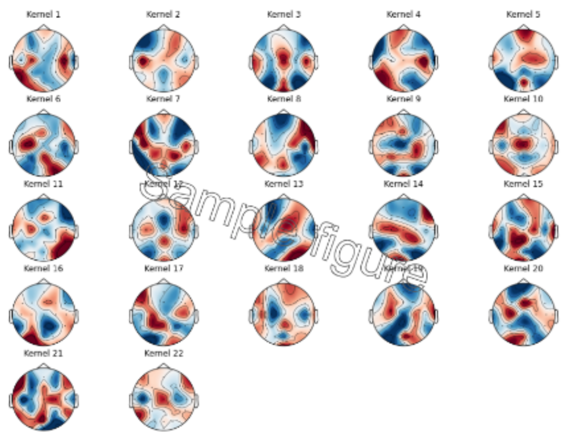
1.1.1. (45%) Please go through the referenced literature[2] carefully and complete the "#TODO" parts. For SCCNet implementation, you have to use the predefined arguments to construct each layer. If your SCCNet structure is constructed with fixed constants instead of the provided arguments, the penalty is HW3 score -5. For training scheme implementation, make sure your dataset contents meet the scheme definitions.
Plot the confusion matrix of your best performed model-scheme-curriculum combination.
※ [confusion matrix wikipedia]()
※ [matplotlib.cm (optional)]()

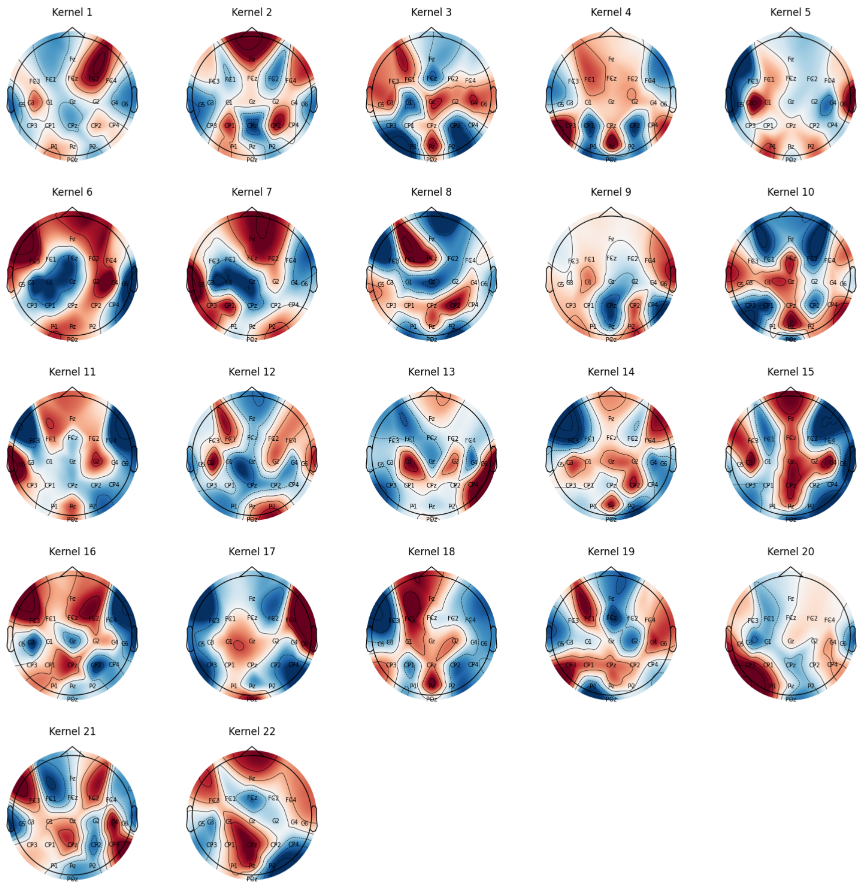
<table>
<tr><th></th><th>EEGNet</th><th>SCCNet (or SCCNet_v2)</th></tr>
<tr><td>SI</td><td>
</td><td>
</td></tr>
<tr><td>SI+FT</td><td>
</td><td>
</td></tr>
</table>

1.1.2. (5%) Fill in your hyper-parameter settings (Batch size, learning rate, epochs, optimizer, etc) of your best performed model-scheme-curriculum combination.
※ if you didn't modify the sample code, fill in the hyper-parameters specified in the sample code.

	EEGNet	SCCNet (or SCCNet_v2)
SI	Batch size: 16 learning rate: 1e-4 epochs: 250 optimizer: Adam scheduler: CosineAnnealingLR loss_fn: CrossEntropyLoss	Batch size: 16 learning rate: 1e-4 epochs: 600 optimizer: Adam scheduler: StepLR loss_fn: CrossEntropyLoss
SI+FT	Batch size: 16 learning rate: 1e-4 epochs: 50 optimizer: Adam scheduler: CosineAnnealingLR loss_fn: CrossEntropyLoss	Batch size: 16 learning rate: 1e-4 epochs: 50 optimizer: Adam scheduler: StepLR loss_fn: CrossEntropyLoss

1.1.3. (10%) Obtain spatial kernel weights from the first convolutional layer of your best performed SCCNet model, visualize the weights as topographic maps using the MNE package.
※ [PyTorch Conv2D attributes]
※ [mne.channels.make_standard_montage] (Hint: electrode position)
or [mne.info] (Hint: EEG recording metadata structure)
※ [mne.viz.plot_topomap]
(Hint: There should be 22 topo maps for one model in your figure, each corresponding to a channel. Model weights and their meanings [article1], [article2])





1.1.4. (0%) Make predictions to the 5 test subjects without true labels (unlabeled_test) with the one with best performance from your trained model, export the result as a .csv file using the sample code (section `Generate Submission csv File`) and **submit it to kaggle**. This part is to make sure your code is executable and your models are trainable, which should beat the 0.6 accuracy baseline (the baseline will be adjusted dynamically according to your submissions). **Otherwise you will only get half of the score in your implementation (1.4.1)**.
(Hint1: You may need to tune your hyper-parameter setting to get a higher accuracy)

1.2. Discussion (40%)

1.2.1. Strengths and weaknesses of the 2 CNN models and your observations regarding the models (structures, parameter size .etc)

- EEGNet
  - **Strengths**: EEGNet captures temporal patterns through convolutional operations and spatial patterns through the depthwise separable convolution. This allows the model to learn hierarchical representations, which can be beneficial for EEG data where both temporal and spatial characteristics are important.
  - **Weakness**: The model's receptive field may be limited, especially with shorter temporal convolutions. This limitation may impact its ability to capture long-range dependencies in the input data.

- **Observations**: The model has a relatively compact architecture compared to deeper neural networks, which could be advantageous for applications with limited computational resources.

- SCCNet
  - **Strengths:** SCCNet introduces the concept of spatial-temporal kernels, allowing the model to capture both spatial and temporal dependencies in EEG data. This can be advantageous for tasks where the interplay between spatial and temporal features is crucial.
  - **Weakness:** The introduction of spatial-temporal kernels and non-linear activations might make the model more complex, potentially making it harder to interpret the learned features.
  - **Observations:** The performance of SCCNet with a motor imagery dataset does outperform EEGNet.

1.2.2. Strengths and weaknesses of the training schemes and curriculums

- **Strengths**: Compared to the traditional training process, as exemplified by curriculum 2, the performance of curriculum 1 and 3 often outperforms that of curriculum 2. I believe the reason behind this is that the model may learn more from the training data.
- **Weakness**: Since we take more training steps, the model may learn more from the training dataset, resulting in high accuracy. However, when testing on the testing dataset, the performance drops significantly. In the homework, I can achieve an accuracy of more than 70% without using the fine-tuning method; however, when uploaded to Kaggle, it drops to nearly 55%.

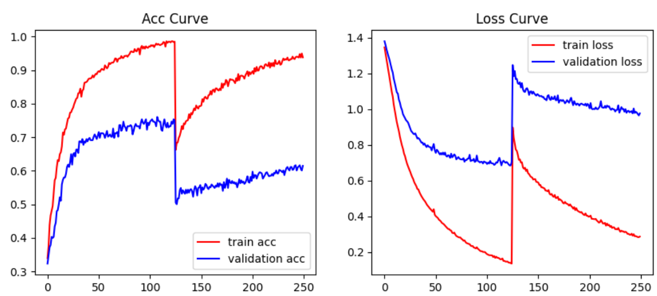
1.2.3. (Optional) For models trained with different subject sets, what are the possible reasons for the difference in model performance.

- I think choosing a different easy subject list might influence the performance to some extent. Taking subject 3 as an example, since its performance shows the best accuracy, the model performs well when subject 3 is included in an easy list after training.

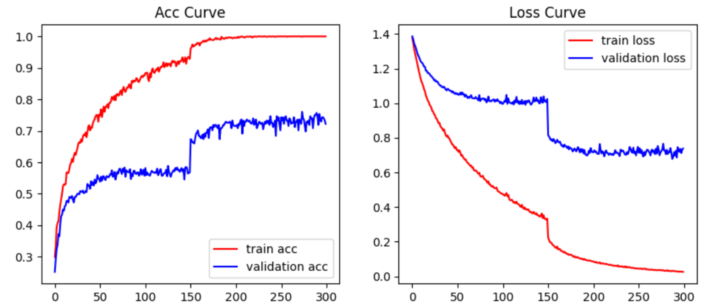
1.2.4. Other topics you find worthy to discuss.

- It's quite strange to me that, in this homework, when implementing a fine-tuning method, I can achieve an accuracy of more than 80%. However, when uploaded to Kaggle, the accuracy drops to 50%, and even worse, to 45%. Since the data used for fine-tuning is the target test subject, the model should learn more about it from the dataset. And thus, I'm guessing that it might be happening because of overfitting conditions.
- After working on a different curriculum, I'm not sure whether this method will surely evaluate the performance of the model. By taking a look on the curve graph:

  Curriculum 1:



Curriculum 3:



Observing the charts, we can clearly see a significant steep change at half the number of epochs. This is because the model encounters new datasets. For curriculum 1, the model faces new datasets, but for curriculum 3, these datasets have been encountered before during the training process. Even though a high accuracy is presented during the training stage, I believe it does not contribute significantly to the model's training. To improve performance, I think designing a better model is more crucial than refining the training curriculum.