CSIC30100: Brain Computer Interface **(Due: 05/05/2023)**

HW3: Deep Learning for BCI

Instructor: Chun-Shu Wei
TA: 311553009

Introduction

In this homework, you are asked to utilize Convolutional Neural Network (CNN) for motor imagery EEG classification tasks, with experiments on 4 training schemes and 3 model architectures.
Please follow the provided code template for CNN framework implementation, model training, and dataset loading to fulfill the requirements. Try to make observations through the progress and discuss the results.

Kaggle link
Code template

**Make sure that you download the sample code before starting your implementation.

Submission Policy

Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- **PLAGIARISM IS STRICTLY PROHIBITED. (0 point for Plagiarism)**
- The code will be graded on code completeness, model structure and algorithmic correctness.
- Submission deadline: **2023.05.05 10:00:00 AM**.
- Late submission penalty formula:

$$\text{original score} \times (0.7)^{\#(\text{days late})}$$

Submission Format

- Each student submits 1 zip file including **1 report** (.pdf file) and your **code** (.ipynb file). Paper submission or programming languages other than Python/deep learning framework other than PyTorch is not allowed.
- Remember to upload the results for unlabeled test data to Kaggle.
- The source code must contain **comments**.
- The report must contain **observations, results, and explanations**. Please name your zip file as hw3_studentID_Name.zip.
- Illegal format penalty: **`-5 points`** for violating each rule of submission format.

Materials

- PyTorch tutorials:
  https://pytorch.org/tutorials/
  https://github.com/utkuozbulak/pytorch-custom-dataset-examples
- MNE: EEG processing & analysis Python package

  documents
- (for your own reference: EEGNet and ShallowConvNet implementation)

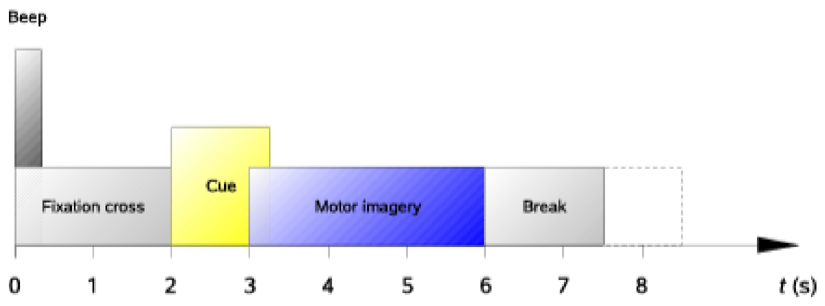
---

1. Problem

1.1. EEG Dataset

Dataset description: (Official document)

BCI competition IV 2a is a well-studied motor imagery dataset composed of **22** channel EEG data from **9** subjects. Four motor imagery tasks are involved, namely the imagination of movement of the **left hand** (class 1), **right hand** (class 2), **both feet** (class 3), and **tongue** (class 4). The experiment paradigm is shown as follows.



In a trial, the subject is presented with a visual fixation cross. The instruction cue to imagine specific body part movement is given at 2s and lasts for 1.25s, the subject is prompted to perform the desired motor imagery task until 6s. Each subject has one training and one evaluation session, each session includes **72** trials for the 4 classes, yielding a total of 72*4 trials per session. The given dataset is filtered by FIR with band [4, 38] Hz and resampled to 125.0 Hz. The event trials are epoched using the same procedure described in referenced literature[2] .

Dataset is provided through Kaggle. You should follow the instructions in the code template to download this dataset. The dataset will be organized according to the following structure.

Kaggle data structure
- BCI_hw3_dataset/   - train/   - BCIC_S0x_T.mat (x from 1~9, keys=x_train, y_train)   - labeled_test/   - BCIC_S0x_E.mat (x from 1~4, keys=x_test, y_test)   - unlabeled_test/   - BCIC_S0x_E.mat (x: 5, 6, A, B, C, keys=x_test)

The dataset filename is under the format "BCIC_S{subject_ID}_{T(train)/E(test)}.mat", e.g., "BCIC_S01_T.mat" is the training set for subject one and "BCIC_S03_E.mat" is the test set for subject three. For the unlabeled test data, three of the subject IDs are hidden and the trials were shuffled.

1.2. CNNs

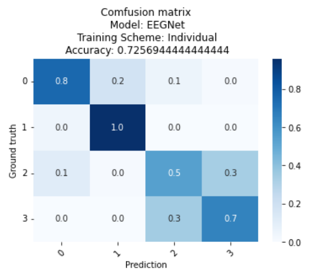
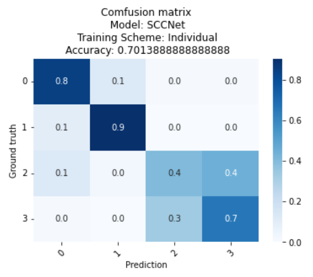
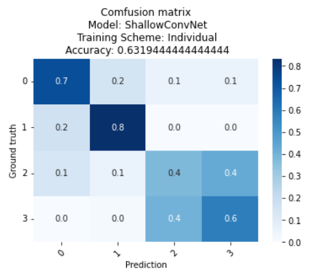
Three baseline EEG models: **EEGNet**[1], Spatial Component-wise Convolutional Network (**SCCNet**)[2] and **ShallowConvNet**[3] are included in this homework. The EEGNet and ShallowConvNet implementations are provided in the sample code, for SCCNet you will have to finish the implementation on your own. The model design ideas and more details could be found in the BCI_ML2 course material and referenced literature[1] [2] [3].

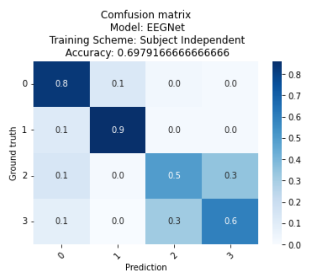
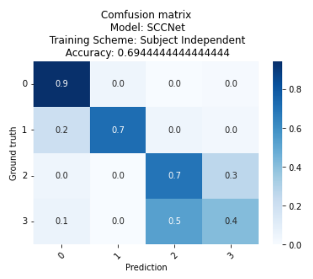
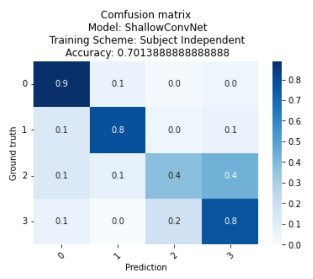
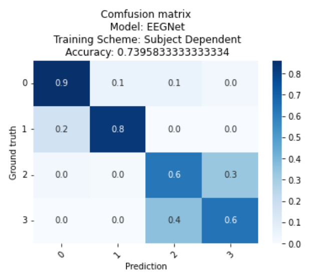
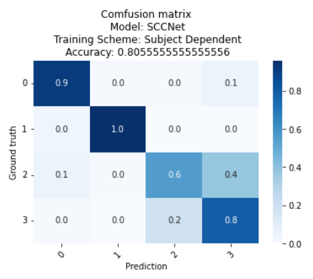
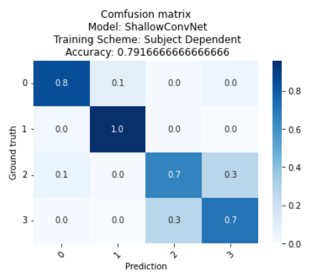
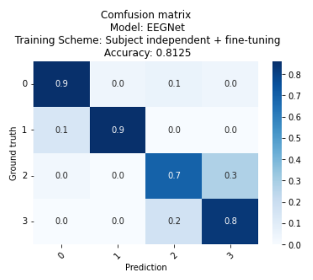
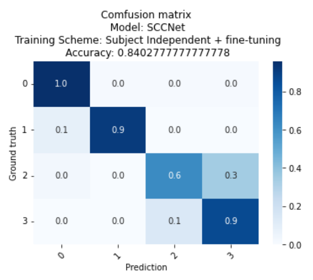
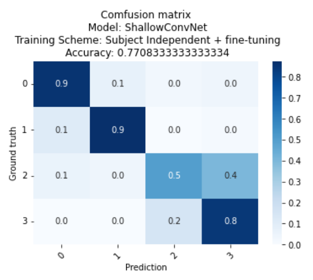
1.3. Training Schemes

Four training schemes: Individual(**Ind**), Subject Independent(**SI)**, Subject Dependent(**SD**) and Subject Independent + Fine Tuning (**SI+FT**) are included in this homework. The visualized explanation could be found in the BCI_ML2 course material and the text explanation could be found in the referenced literature[2] . The training schemes refer to different training data / test data combinations to simulate different kinds of real life BCI application scenarios.

1.4. **Requirements** (60%)

1.4.1. (40%) 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 not changed along with the provided arguments, the penalty is HW3 score -5. For training scheme implementation, make sure your dataset contents meet the scheme definitions.
After finishing the code, train 3 models with 4 training schemes and test on **subject 1 (BCIC_S01_E.mat)**.
Show testing accuracies and plot confusion matrices for your 12 models.
confusion matrix wikipedia
matplotlib.cm (optional)

	EEGNet	SCCNet (or SCCNet_v2)	ShallowConvNet
Ind			

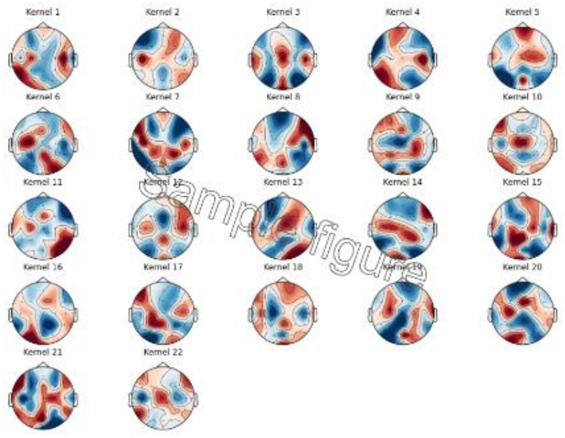
SI			
SD			
SI+FT			

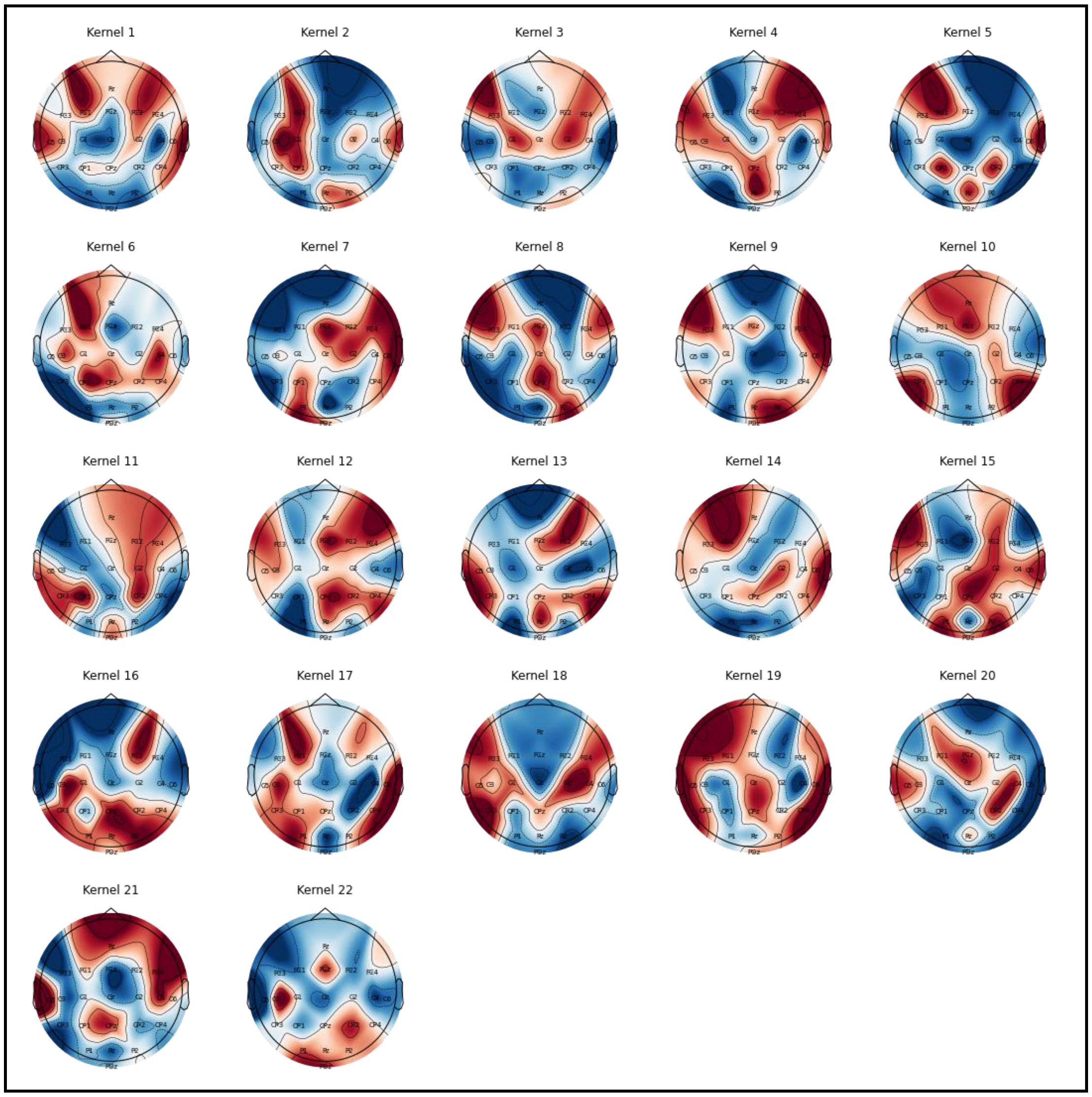
1.4.2. (10%) Fill in your hyper-parameter settings (Batch size, learning rate, epochs, optimizer, etc).
※ if you didn't modify the sample code, fill in the hyper-parameters specified in the sample code
※ if you have different settings for each subjects, fill in the hyper-parameters for your subject 1 model.

	EEGNet	SCCNet (or SCCNet_v2)	ShallowConvNet
Ind	batch size: 16 learning rate: 1e-03 epochs: 120 optimizer: Adam	batch size: 16 learning rate: 1e-04 epochs: 120 optimizer: Adam	batch size: 64 learning rate: 1e-04 epochs: 50 optimizer: Adam
SI	batch size: 16 learning rate: 1e-03 epochs: 200 optimizer: Adam	batch size: 16 learning rate: 1e-04 epochs: 250 optimizer: Adam	batch size: 64 learning rate: 1e-04 epochs: 150 optimizer: Adam
SD	batch size:16 learning rate: 1e-03	batch size: 16 learning rate: 1e-04	batch size: 64 learning rate: 1e-04

	epochs: 200 optimizer: Adam	epochs: 200 optimizer: Adam	epochs: 150 optimizer: Adam
SI+FT	batch size: 8 learning rate: 1e-04 epochs: 100 optimizer: Adam	batch size: 8 learning rate: 1e-05 epochs: 150 optimizer: Adam	batch size: 32 learning rate: 1e-05 epochs: 50 optimizer: Adam

1.4.3. (10%) Obtain spatial kernel weights from the first convolutional layer of your SCCNet model trained with Ind scheme on subject 1, 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)





1.4.4. (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 submission). 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)
(Hint2: Generally, subjects 5 and 6 result in lower test accuracy)

Discussion (40%)

1.4.5. Pros and cons of the 3 CNN models and 4 training schemes

Models

1. EEGNet

- Pros: EEGNet uses depthwise separable convolutions, which makes it computationally efficient and reduces overfitting. It also includes a spatial attention mechanism that helps the model focus on important features in the EEG signals.
- Cons: EEGNet may not be able to capture as much complex information in the EEG signals as SCCNet, and may not perform as well on more challenging classification tasks.

2. SCCNet
    - Pros: SCCNet is also a shallow neural network that has the smaller model size. The design of SCCNet focuses on leveraging the benefits from applying spatial kernel to multi-channel EEG data in the first convolutional layer for purposes such as feature extraction and noise reduction
    - Cons: Because the feature of small model size, SCCNet can only be adapted to relatively small EEG datasets, it may get the worse result on the large dataset classification task.
3. ShallowConvNet
    - Pros: ShallowConvNet was proposed simultaneously with DeepConvNet for decoding motor-imagery EEG data, but ShallowConvNet has been more popular than DeepConvNet because of its smaller model size. Therefore, it can learn more complicated features using the less parameters in the training phase.
    - Cons: Compare with SCCNet and EEGNet, ShallowConvNet has larger model size because it is a deep neural network. As a result, on the easy classification task, the model complexity of ShallowConvNet is too large to learn well.

Training schemes

1. Individual: The model is trained on a dataset that includes EEG signals from a single subject, and is tested on EEG signals from the same subject. This can help to evaluate the performance of the model on a specific individual, and may be useful for personalized medicine applications.
2. Subject dependent: The model is trained on a dataset that includes EEG signals from multiple subjects, and is tested on EEG signals which includes the data from one of training subject, but this data still be unseen in the training phase. Because we will split the subject data to two sessions, the first session for training and the second session for testing. Therefore, this training scheme usually get the better results than subject independent strategy.
3. Subject independent: The model is trained on a dataset that includes EEG signals from multiple subjects, and is tested on EEG signals from subjects that were not included in the training set. This can help to evaluate the generalization performance of the model. But it is a challenge for this scheme to achieve the good performance.
4. Subject independent + fine tuning: The model is pre-trained on a large dataset of EEG signals from multiple subjects, and then fine-tuned on a smaller dataset that includes EEG signals from a specific subject. This can help to improve the

performance of the model on a specific individual by fine-tuning on the test subject while still taking advantage of the knowledge gained from the larger dataset.

1.4.6. Your observations regarding the models (structures, size .etc)

In terms of model size, ShallowConvNet is the largest with about 50000 parameters, although it is a shallow version of DeepConvNet, it still belongs to the deep neural network. On the other hand, EEGNet has fewer parameters than SCCNet because it has a more streamlined architecture, with fewer convolutional layers and a depthwise separable convolutional layer that reduces the number of parameters needed. However, the size of the model is larger than SCCNet because it may require more memory to store the weights and biases with higher precision.

1.4.7. Difficulties you encountered in this homework.

There are about two challenges for me in this assignment.

1. Implement SCCNet from scratch

   Implementing the general architecture of the model is not difficult, but like the padding parameters in the convolutional layer, it is only mentioned using zero-padding in the original paper. When I set the padding parameter depending on the description of the paper, the timepoint of output shape in the second convolutional layer is not the same as the value given by the sample code. In my understanding, the second convolution uses convolutional kernels with a size of (Nu, 12), where the number 12 corresponds to 0.1 seconds along time domain, so my code is `kernel_size=(Nu, math.floor(fs*0.1))` and the setting of the padding parameter I used is `padding=(0, int(math.floor(fs*0.1)/2))`. Maybe there is something wrong with my understanding?

2. How to split the data to training and validation set to get the expected results

   Another challenge is how to design the good data splitting or maybe we need to use all data as training set to train our model instead of splitting out validation set to evaluate. We only know that this experimental data has two sessions, but we don't know which session the T and E data correspond to, this thing isn’t seem to mention in the spec. Because the experimental conditions could vary between sessions, if we can know in advance how the session is differentiated, it will help us to evaluate the unlabeled test data in E in final part.

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

There are several reasons for the difference in model performance when trained with different subjects:

1. Variability in EEG signals

   EEG signals can vary significantly between individuals, even among healthy individuals. The differences in EEG signals could lead to variations in the performance of the model, it may not be able to generalize well across different subjects.

2. Differences in the mental states of the subjects

The mental states of the subjects could affect the EEG signals, leading to differences in the performance of the model. For example, if some subjects are more anxious or distracted than others, this could affect the EEG signals and, in turn, the result of the model.

3. Differences in the electrode placements
   The electrode placements could vary between subjects, leading to differences in the spatial information available in the EEG signals. This could affect the performance of the model as it may not be able to capture the spatial information in the same way across different subjects.
4. Variations in the experimental conditions
   The experimental conditions could vary between subjects, leading to differences in the EEG signals. For example, if the EEG signals are recorded under different lighting conditions, this could affect the performance of the model. Also, if the subject moves during the recording, it can cause artifacts and noise in the EEG signal.

1.4.9. Other topics you find worthy to discuss.

Before doing this homework assignment, I originally thought that for the three models under different training schemes, the classification performance from good to bad would be subject independent + fine-tuning, subject dependent, individual, subject independent. Because the testing data of model with subject dependent include the same subject for training phase, the EEG signals of the same subject have similar characteristics and even wave patterns. In contrast, if we use subject independent scheme to train our model, the subject for testing data will not appear in the training phase, so it is difficult to achieve a higher accuracy, the advantage is that the prediction results are more generalized. After using fine-tuning data to train the pretrained model again can potentially improve its performance.

However, experiments have shown that this idea is not quite right. Especially, the performance of ShallowConvNet with individual scheme is worse than subject independent, I think the possible reason is the model size of ShallowConvNet is large, it needs more complex and the larger number of data to train well, and there is too few data for a single subject.

1.5. Bonus Challenge

1.5.1. SCCNet_v2

In the SCCNet architecture there is a “permutation layer”, whose functionality can be achieved simply through some modification of the kernels in the convolutional layer(s). To get bonus points, try to implement SCCNet without the permutation layer. (Hint: go through the referenced literature[2] and try to understand the kernel direction, their corresponding data dimensions, and the kernels’ operation objectives.)

Correct implementation: HW3 +10 points (+10(0.45/3) = +1.5 for semester score)

1.5.2. Kaggle leaderboard

The top **3** submissions in the Kaggle leaderboard will get bonus points.

Top 1: HW3 +30 points (+ 4.5 for the semester score)

Top 2: HW3 +20 points (+ 3 for semester score)
Top 3: HW3 +10 points (+ 1.5 for semester score)

1.6. References

[1] Lawhern, Vernon J., et al. "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces." Journal of neural engineering 15.5 (2018): 056013.
[2] Wei, Chun-Shu, Toshiaki Koike-Akino, and Ye Wang. "Spatial component-wise convolutional network(SCCNet) for motor-imagery EEG classification." 2019 9th International IEEE/EMBS Conference on Neural Engineering (NER). IEEE, 2019.
[3] Schirrmeister, Robin Tibor, et al. "Deep learning with convolutional neural networks for EEG decoding and visualization." Human brain mapping 38.11 (2017): 5391-5420.