

**Name: Varshitha Akula**

**Netid: Va2437**

## **DETECTION OF SSVEP AND PREDICTION OF WEAK SPOTS IN THE VISION**

### **ABSTRACT**

In this work an efficient system is presented which can detect the weak spots in the vision. Eyesight is an important aspect of human life. The main causes of blindness and vision loss are age-related eye conditions such as age-related macular degeneration, cataracts, glaucoma, and diabetic retinopathy. Other common eye diseases are amblyopia and strabismus. Glaucoma Early and late diagnosis onset of symptoms with irreversible degeneration of retinal ganglion cells. The default automatic visual field test is the gold standard for assessing glaucoma. However, the exam is subjective and the answer is, the interpretation of the exam is highly biased as it varies from test run to test run. This study presents an approach to rapid point-of-care diagnostics, For patients with glaucoma Available visual field assessment. Unlike the existing method used, we mainly report the accuracy of foveal target detection. The Model Architecture of a Combination of Convolutions and Multi-Task Model efficiently captures signals Peripheral vision simultaneously creates a visual response map from the fovea. Efficiently learn several tasks at the same time. We evaluated using model classification on 40 class records and achieved a yield of 93% on accuracy and 96% on F1 score.

### **INTRODUCTION**

The leading causes of blindness and low vision are age-related eye diseases such as age-related macular degeneration, cataract, diabetic retinopathy, and glaucoma. The vital symptomatic sign for a glaucoma patient is fringe vision misfortune, alluding to the most extreme point field of vision from the focal point of obsession for each eye. Glaucoma suspects are encouraged to evaluate their visual capacities early and consistently.

This evaluation is finished by standard computerized perimetry (SAP); it is the highest quality level performed to evaluate and analyse the illness' seriousness. The evaluation of visually impaired regions in the visual field is utilized to screen visual capacity in glaucoma.

The technique length is around 10-minutes per eye, where the patient is needed to take a gander at a huge semi-round the bowl that covers their whole field of view. We present a perform multiple tasks learning system for creating a visual reaction map possibly appropriate for giving glaucoma diagnostics. The patient needs to keep up with obsession at a focal objective during the whole system. The patient reacts by clicking a button at the point when an improvement is seen, and this interaction yields a guide of the seeing pieces of the patient's field of view.

SAP is an emotional assessment where changing reactions might be acquired each time the test is performed or in any event, during a similar test. These vacillations as a rule increment with the seriousness of the illness. It has been the greatest downside of this visual field appraisal, as it might essentially frustrate the test's translation. These clashing outcomes might impede the doctor's choice to make a conclusion and request different resulting tests. To lessen results of inconsistency, patients should initially be taught to utilize the framework; this would take various meetings, contingent upon the patient's capacity to convey the method. Roughly ten visual field tests are expected to accomplish an exact expectation point-wise and mean responsiveness for an average glaucoma patient in the centre, which thus can prompt a postponement in analysis. This features the current limits of visual field testing. A review examines patients' encounters in regards to their glaucoma follow-up demonstrates that patients track down visual field assessment by SAP testing or are awkward. A few patients who went through various tests depict their sensations of uneasiness. As a patient's intellectual capacity is engaged with the appraisal, it can create wrong test results because of the patient being unpractised with the framework and assessment technique. A patient could lose obsession to the focal objective because of weariness furthermore interruption, which could likewise influence the experimental outcomes.

These elements feature a requirement for an innovative arrangement that gives a genuine appraisal that is exceptionally attractive to further develop visual field test viability. Consistent state visual-evoked potential (SSVEP) is an oscillatory improvement reaction evoked by specific monotonous boosts with a steady recurrence. These are recorded in the electroencephalogram (EEG) and can be identified from the essential visual cortex. The principle thought of our review is represented in the work with the expectation to carry out a framework that produces visual field tests results that are more solid as we dispense with the intellectual perspective in the current visual field evaluation.

Thus, patients don't need to figure out how to utilize the framework, and test results are not impacted by patients being diverted or feeling awkward. Since our MTL model is fit for identifying

numerous upgrades, we can decrease visual field appraisal time and produce dependable experimental outcomes. This could be conceivably reasonable for giving fast place-of-care diagnostics for glaucoma patients.

## **Objectives**

The following are the objectives of this project:

- The Main objective is to Detect the SSVEP - Steady-state visual-evoked potential (SSVEP)
- Detect the weak spot in the vision, using a combination of convolutions and Multitasking.
- To productively catch signals all the while from the fovea and the adjoining targets in the vision, producing a visual reaction map.

## **Background and Literature Survey**

Most examinations have been committed to distinguishing a solitary glinting objective where the attention is on conveying solid SSVEP reactions are recognized on fovea vision; thus boosts at the fringe vision are viewed as commotion signals. In this study,

we centre around distinguishing the fringe field to analyse glaucoma patients by catching signs from the fovea and the adjoining focuses in the fringe vision. An in a general sense diverse method for distinguishing SSVEP was proposed in this review, utilizing the utilization of Multi-task Learning (MTL), which is to prepare a neural organization with numerous connected goals (or then again assignments) while sharing a typical organizational structure. MTL can decide how undertakings are connected without being given a piece of explicit information for task relatedness. Preparing an organization that figures out how to foresee various undertakings in a single organization performs better than preparing numerous different organizations to foresee numerous undertakings. In SSVEP identification, we use MTL to further develop SSVEP arrangement execution. The common layers empowered the model to interpret SSVEP from crude EEG; what's more, assignment explicit layers will figure out how to isolate diverse objective frequencies. The quantity of results relates to the number of target frequencies in the dataset, where each result is the likelihood of the SSVEP recurrence present in the EEG. MTL will permit us to recognize a patient's visual field by distinguishing numerous SSVEP focuses immediately, in this manner chopping down the patient's appraisal time

## **Proposed System**

Our model is comprised of 3 sections:




































1) SSVEP highlight extraction 2) include learning, and 3) perform various tasks learning blocks. A delineation of the proposed network is displayed beneath. Our neural organization is made out of four convolution blocks, answerable for learning EEG portrayals across all objective frequencies. The fifth convolution layer is a performance learning block, where it figures out how to separate different SSVEP target frequencies.

## **Dataset**

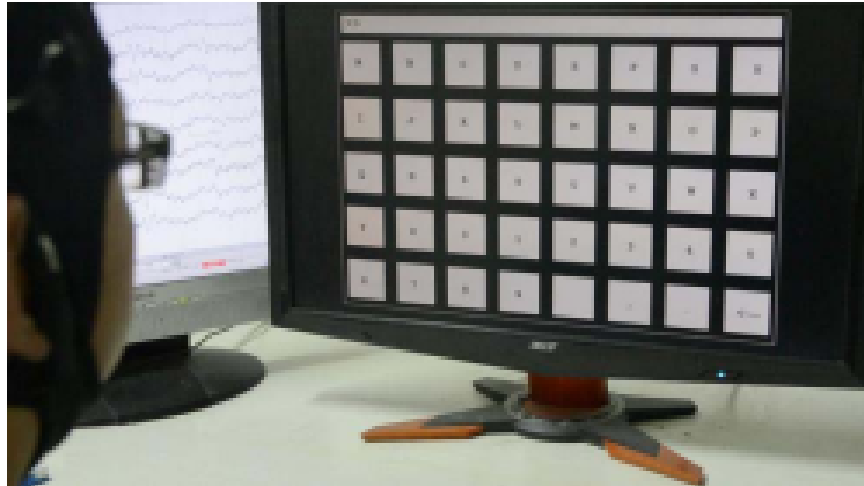
A Benchmark Dataset for SSVEP-Based Brain–Computer Interfaces Theoretical—This paper presents a benchmark consistent state visual evoked potential (SSVEP) dataset gained with a 40-target mind PC interface (BCI) speller. The dataset comprises

64-channel Electroencephalogram (EEG) information from 35 sound subjects (8 experienced and 27 guileless) while they played out a promptly directed objective choosing the errand. The virtual console of the speller was made out of 40 visual flashes, which were coded utilizing a joint recurrence and stage balance (JFPM) approach. The excitement frequencies went from 8 Hz to 15.8 Hz with a time frame Hz. The stage contrast between two contiguous frequencies was  $0.5\pi$ . For each subject, the information included six squares of 40 preliminaries compared to every one of the 40 flashes shown by a viewable prompt in irregular requests. The feeling term in every preliminary was five seconds. The dataset can be utilized as a benchmark dataset to think about the strategies for upgrading coding and target recognizable proof in SSVEP-based BCIs. Through disconnected reproduction, the dataset can be utilized to plan new framework graphs and assess their BCI execution without gathering any new information. The dataset likewise gives great information to computational displays of SSVEPs. The dataset is unreservedly accessible from

<http://bci.med.tsinghua.edu.cn/download.html>.

 S1 Microsoft Access Table Shortcut 100 MB	 S2 Microsoft Access Table Shortcut 100 MB	 S3 Microsoft Access Table Shortcut 101 MB	 S4 Microsoft Access Table Shortcut 100 MB	 S5 Microsoft Access Table Shortcut 101 MB
 S6 Microsoft Access Table Shortcut 101 MB	 S7 Microsoft Access Table Shortcut 100 MB	 S8 Microsoft Access Table Shortcut 101 MB	 S9 Microsoft Access Table Shortcut 100 MB	 S10 Microsoft Access Table Shortcut 101 MB
 S11 Microsoft Access Table Shortcut 102 MB	 S12 Microsoft Access Table Shortcut 100 MB	 S13 Microsoft Access Table Shortcut 101 MB	 S14 Microsoft Access Table Shortcut 101 MB	 S15 Microsoft Access Table Shortcut 100 MB
 S16 Microsoft Access Table Shortcut 100 MB	 S17 Microsoft Access Table Shortcut 101 MB	 S18 Microsoft Access Table Shortcut 101 MB	 S19 Microsoft Access Table Shortcut 100 MB	 S20 Microsoft Access Table Shortcut 101 MB
 S21 Microsoft Access Table Shortcut 100 MB	 S22 Microsoft Access Table Shortcut 100 MB	 S23 Microsoft Access Table Shortcut 100 MB	 S24 Microsoft Access Table Shortcut 101 MB	 S25 Microsoft Access Table Shortcut 101 MB
 S26 Microsoft Access Table Shortcut 101 MB	 S27 Microsoft Access Table Shortcut 100 MB	 S28 Microsoft Access Table Shortcut 101 MB	 S29 Microsoft Access Table Shortcut 101 MB	 S30 Microsoft Access Table Shortcut 101 MB
 S31 Microsoft Access Table Shortcut 101 MB	 S32 Microsoft Access Table Shortcut 100 MB	 S33 Microsoft Access Table Shortcut 101 MB	 S34 Microsoft Access Table Shortcut 101 MB	 S35 Microsoft Access Table Shortcut 101 MB

This dataset assembled SSVEP-BCI accounts of 35 subjects (17 females, matured 17-34 years, mean age: 22 years) zeroing in on 40 characters glimmering at various frequencies (8-15.8 Hz with a time frame Hz).



For each subject, the investigation consisted of 6 squares. Each square contained 40 preliminaries relating to every one of the 40 characters shown in the irregular request. Every preliminary began with an obvious prompt (a red square) demonstrating an objective upgrade. The prompt showed up for 0.5 s on the screen. Subjects were approached to move their look to the objective at the earliest opportunity inside the signal term. Following the prompt offset, all improvements began to flash on the screen simultaneously and kept going 5 s. After improvement offset, the screen was clear for 0.5 s before the following preliminary started, which permitted the subjects to have brief breaks between continuous preliminaries. Every preliminary endured an aggregate of 6 s. To work with visual obsession, a red triangle showed up beneath the glimmering objective during the excitement time frame. In each square, subjects were approached to keep away from eye flickers during the excitement time frame. To stay away from visual weakness, there was a rest for quite some time between two continuous squares.

## **BCI-Terminology**

### **Understanding of EEG**



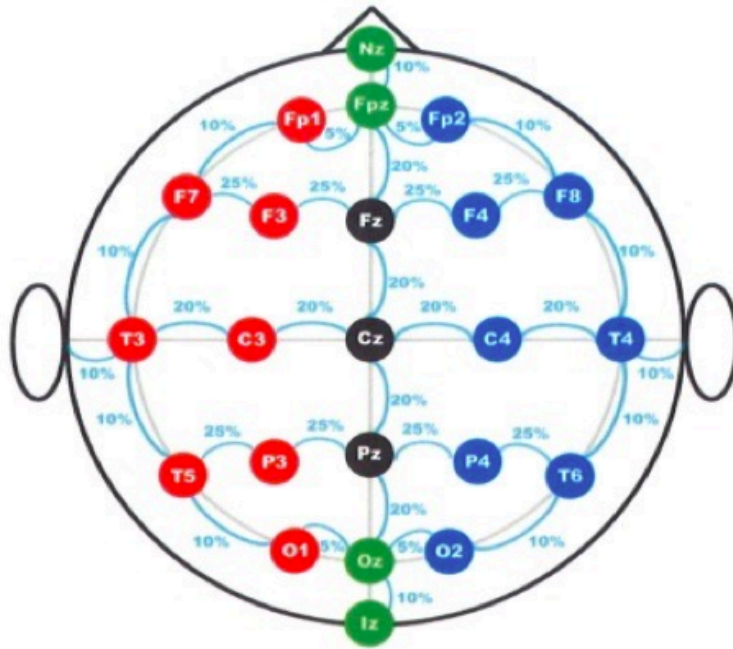
The electroencephalogram (EEG) is a recording of the electrical activity of the brain from the scalp. The recorded waveforms reflect the cortical electrical activity. Signal intensity: EEG activity is quite small, measured in microvolts (mV).

### **EEG - Sampling Rate**

The sampling rate is usually expressed in Hz, for example 240 Hz is 240 times per second. The minimum acceptable sampling rate is 2.5 times greater than the highest frequency of interest but most digital EEG systems will sample at 240 Hz.

The sampling frequency or sampling rate,  $f_s$ , is the average number of samples obtained in one second, thus  $f_s = 1/T$ . Its units are samples per second or hertz e.g. 48 kHz is 48,000 samples per second.

### **EEG-Channels**



An electrode capturing brainwave activity is called an EEG channel. Typical EEG systems can have as few as a single channel to as many as 256 channels.

## SSVEP

Small amplitude stable VEP were generated which were entitled “steady-state” visually evoked potentials (SSVEPs) of the human visionary system. There hence, steady-state visual evoked potentials (SSVEPs) are defined as the potential elicited by the change in the visual field with a frequency higher than 6 Hz.

## Model

Our model consists of 3 sections, 1) SSVEP highlight extraction, 2) include learning, and 3) perform multiple tasks learning block. A delineation of the proposed network is displayed beneath.

Our neural organization is made out of four convolution blocks, liable for learning EEG portrayals across all objective frequencies. The fifth convolution layer is a perform



various tasks learning block, where it figures out how to separate different SSVEP target frequencies. [Image by Author]

### SSVEP highlight extraction

The motivation behind this part is to interpret crude EEG signals into input highlights for our model. There are two convolution blocks (C1 and C2); every convolution block comprises a convolution layer, a cluster standardization, and a dramatic direct unit.

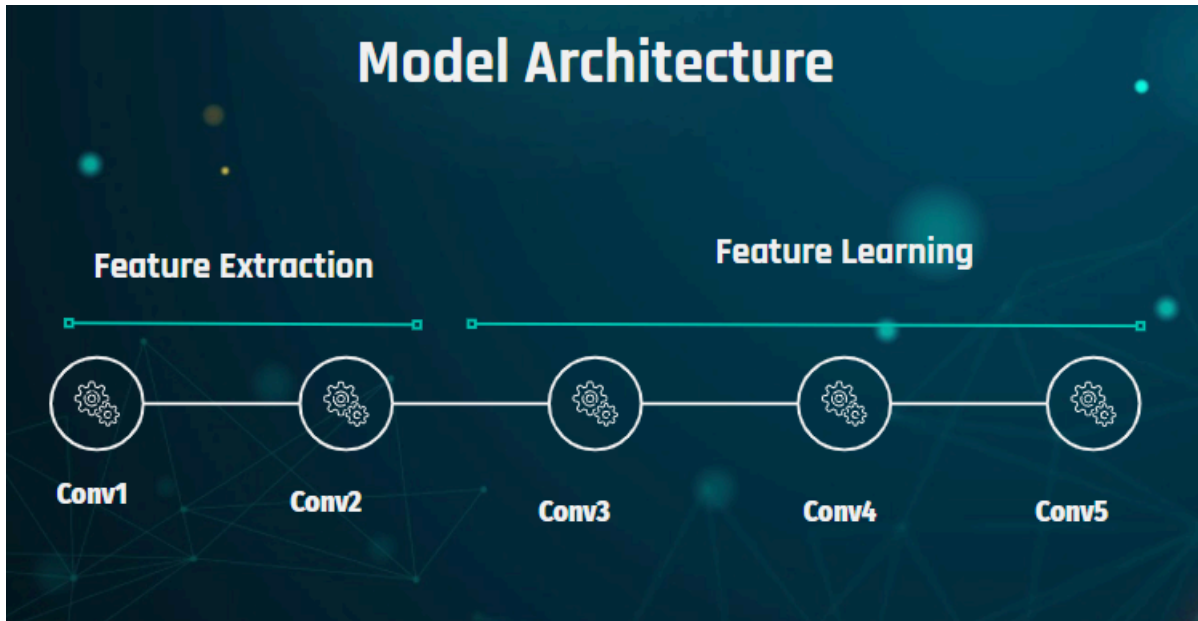
C1 block was intended to separate the phantom portrayal of the EEG input, as it performs convolution across the time aspect, catching highlights from every EEG channel freely from the others.

C2 block was intended for leading spatial sifting, as it performs convolutions across the channel aspect. The goal of this layer is to become familiar with loads of all channels at each time test.

### Highlight learning

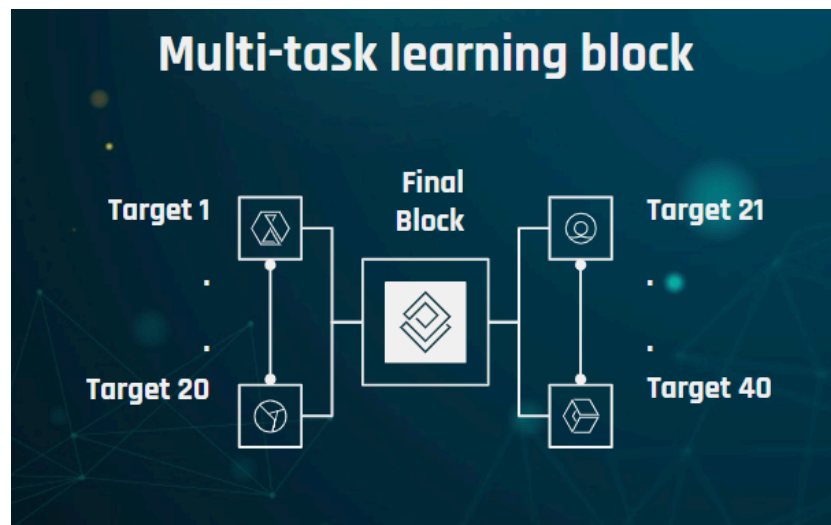
Here, there are two convolution blocks (C3 and C4), the motivation behind these squares is to catch the fleeting examples in each extricated highlight map.

We additionally investigated diverse expansion setups on C3 and C4 blocks. As the portion size required for signals is a lot bigger than for pictures, widening convolutions permit us to grow the open field, perform highlight learning with a more modest bit size, produce a more modest model, and possibly increment execution.



Perform multiple tasks learning block

The fifth and last convolution block figures out how to separate numerous SSVEP target frequencies. We utilized strategies from performing various tasks learning and planned this square to create different results.



In exemplary perform various tasks learning designs, each assignment is isolated into task-explicit layers, where each layer is answerable for figuring out how to distinguish

each errand. In any case, rather than copying a convolution layer for each result, we perform convolutions by gatherings. We are basically performing separate convolutions inside a solitary convolution layer. This permitted us to prepare different assignments in equal proficiency on a solitary GPU, and we can progressively change our model scale to quite a few undertakings successfully, which is possibly reasonable for others to perform multiple tasks learning applications.

As it is a Regression problem. I applied 3 different machine learning algorithms for training and testing the data for choosing the better performing algorithm in this case.

First, I trained the model using Multiple linear regression and applied mean absolute error for accuracy check. Multiple linear regression plots the available data and tries to form different lines which will pass through as many possible numbers of points plotted and forms an equation and predicts the data with the equation.

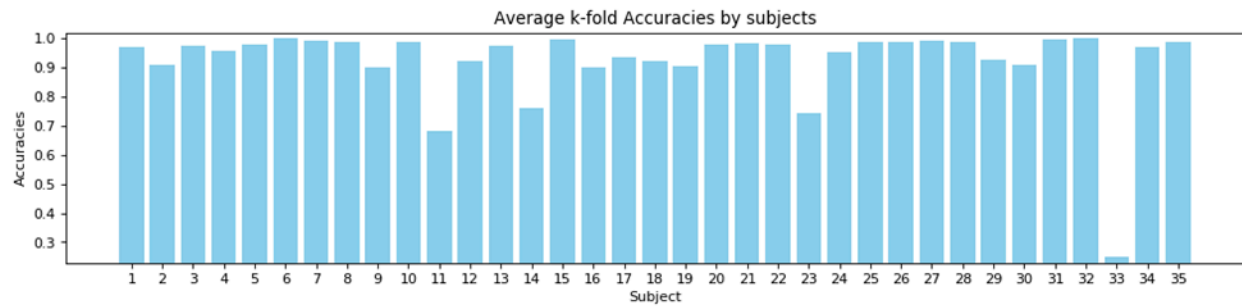
Second, I used a decision tree regression algorithm.

Third, I used K Nearest neighbours' algorithm, in this it plots the data into a graph and depending on the data it groups the nearest point into a cluster and predicts values using these clusters.

Then for predicting future total energy consumption I used K nearest neighbours' and decision tree algorithms. As it is related to time and we only use time and main consumption data the models like linear regression and sv regression does not work for time data types.

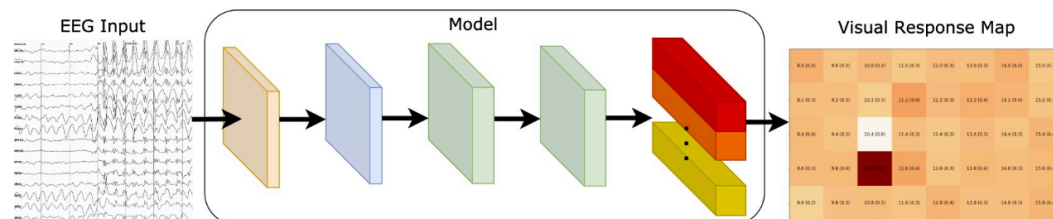
## **RESULTS**

We evaluated using model classification on 40 class records and achieved a yield of 93% on accuracy and 96% on F1 score.



Individual Accuracies of 35 Subjects

**CODE :**



▼ Init and Config

```
[ ] cd "C:\Users\varsha\Downloads\review3-code\torchsignal"

[ ] from torchsignal.datasets.hsssvpep import HSSSVPEP
    from torchsignal.datasets.multiplesubjects import MultipleSubjects
    from torchsignal.trainer.multitask import Multitask_Trainer

    import numpy as np
    import torch
    from torch import nn
    import matplotlib.pyplot as plt
    from matplotlib.pyplot import figure

[ ] config = {
    "exp_name": "model",
    "seed": 12,
    "segment_config": {
        "window_len": 4,
```

```

        "window_len": 4,
        "shift_len": 250,
        "sample_rate": 250,
        "add_segment_axis": True
    },
    "bandpass_config": {
        "sample_rate": 250,
        "lowcut": 1,
        "highcut": 40,
        "order": 6
    },
    "train_subject_ids": {
        "low": 1,
        "high": 35
    },
    "test_subject_ids": {
        "low": 1,
        "high": 35
    },
    "root": "../data/hsssvpep",
    "selected_channels": ['P2', 'P05', 'P03', 'P0z', 'P04', 'P06', 'O1', 'Oz', 'O2', 'P07', 'P08'],
    "num_classes": 40,
    "num_channel": 11,
    "batchsize": 64,
    "learning_rate": 0.001,
    "epochs": 100,
    "patience": 5,
    "early_stopping": 10,
    "model": {
        "n1": 4,
        "kernel_window_ssvep": 59,
        "kernel_window": 19,
        "conv_3_dilation": 4,
        "conv_4_dilation": 4
    },
    "gpu": 0,
    "multitask": True

```

#### Load Subjects Data

```

[ ] subject_ids = list(np.arange(config['train_subject_ids']['low'], config['train_subject_ids']['high']+1, dtype=int))

data = MultipleSubjects(
    dataset=HSSSVEP,
    root=config['root'],
    subject_ids=subject_ids,
    selected_channels=config['selected_channels'],
    segment_config=config['segment_config'],
    bandpass_config=config['bandpass_config'],
    one_hot_labels=True,
)

```

```

Load subject: 1
Load subject: 2
Load subject: 3
Load subject: 4
Load subject: 5
Load subject: 6
Load subject: 7
Load subject: 8
Load subject: 9
Load subject: 10
Load subject: 11
Load subject: 12
Load subject: 13
Load subject: 14
Load subject: 15
Load subject: 16
Load subject: 17
Load subject: 18
Load subject: 19
Load subject: 20
Load subject: 21
Load subject: 22

```

```
[ ] Load subject: 33
Load subject: 34
Load subject: 35
```

```
[ ] print("Input data shape:", data.data_by_subjects[1].data.shape)
print("Target shape:", data.data_by_subjects[1].targets.shape)
```

```
Input data shape: (240, 11, 1000)
Target shape: (240,)
```

```
[ ] class Multitask_Model(nn.Module):
    def __init__(self, num_channel=10, num_classes=4, signal_length=1000, filters_n1=4, kernel_window_ssvep=59, kernel_window=19, conv_3_dilation=4, conv_4_dilation=4):
        super().__init__()

        filters = [filters_n1, filters_n1 * 2]

        self.conv_1 = conv_block(in_ch=1, out_ch=filters[0], kernel_size=(1, kernel_window_ssvep), w_in=signal_length)
        self.conv_2 = conv_block(in_ch=filters[0], out_ch=filters[0], kernel_size=(num_channel, 1))
        self.conv_3 = conv_block(in_ch=filters[0], out_ch=filters[1], kernel_size=(1, kernel_window), padding=(0, conv_3_dilation-1), dilation=(1, conv_3_dilation), w_in=self.conv_1.w_out)
        self.conv_4 = conv_block(in_ch=filters[1], out_ch=filters[1], kernel_size=(1, kernel_window), padding=(0, conv_4_dilation-1), dilation=(1, conv_4_dilation), w_in=self.conv_3.w_out)
        self.conv_mtl = multitask_block(filters[1]*num_classes, num_classes, kernel_size=(1, self.conv_4.w_out))

        self.dropout = nn.Dropout(p=0.5)

    def forward(self, x):
        x = torch.unsqueeze(x, 1)

        x = self.conv_1(x)
        x = self.conv_2(x)
        x = self.dropout(x)

        x = self.conv_3(x)
        x = self.conv_4(x)
        x = self.dropout(x)

        x = self.conv_mtl(x)
```

```
]
class conv_block(nn.Module):
    def __init__(self, in_ch, out_ch, kernel_size, padding=(0,0), dilation=(1,1), groups=1, w_in=None):
        super(conv_block, self).__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(in_ch, out_ch, kernel_size=kernel_size, padding=padding, dilation=dilation, groups=groups),
            nn.BatchNorm2d(out_ch),
            nn.ELU(inplace=True)
        )

        if w_in is not None:
            self.w_out = int( ((w_in + 2 * padding[1] - dilation[1] * (kernel_size[1]-1)-1) / 1) + 1 )

    def forward(self, x):
        return self.conv(x)

class multitask_block(nn.Module):
    def __init__(self, in_ch, num_classes, kernel_size):
        super(multitask_block, self).__init__()
        self.num_classes = num_classes
        self.conv_mtl = nn.Conv2d(in_ch, num_classes*2, kernel_size=kernel_size, groups=num_classes)

    def forward(self, x):
        x = torch.cat(self.num_classes*[x], 1)
        x = self.conv_mtl(x)
        x = x.squeeze()
        x = x.view(-1, self.num_classes, 2)
        return x
```

```
] model = Multitask_Model(num_channel=config['num_channel'],
    num_classes=config['num_classes'],
    signal_length=config['segment_config']['window_len'] * config['bandpass_config']['sample_rate'],
    filters_n1=config['model']['n1'],
```


```

x = torch.ones((40, config['num_channel'], config['segment_config']['window_len'] * config['bandpass_config']['sample_rate'])).to(device)
print("Input shape:", x.shape)
y = model(x)
print("Output shape:", y.shape)

def count_params(model):
    return sum(p.numel() for p in model.parameters() if p.requires_grad)
print("Model size:", count_params(model))

del model

```

 Input shape: torch.Size([40, 11, 1000])  
Output shape: torch.Size([40, 40, 2])  
Model size: 520788

#### ✓ K-fold Train Test

```

[ ] subject_kfold_acc = {}
subject_kfold_f1 = {}

test_subject_ids = list(np.arange(config['test_subject_ids']['low'], config['test_subject_ids']['high']+1, dtype=int))

for subject_id in test_subject_ids:
    print('Subject', subject_id)
    kfold_acc = []
    kfold_f1 = []

    for k in range(config['runkfolds']):
        data.split_by_kfold(kfold_k=k, kfold_split=config['runkfolds'])
        train_loader, val_loader, test_loader = data.leave_one_subject_out(selected_subject_id=subject_id, dataloader_batchsize=config['batchsize'])
        dataloaders_dict = {
            'train': train_loader,
            'val': val_loader
        }.to(device)

        epochs=config['epochs'] if 'epochs' in config else 50
        patience=config['patience'] if 'patience' in config else 20
        early_stopping=config['early_stopping'] if 'early_stopping' in config else 40

        trainer = Multitask_Trainer(model, model_name="model", device=device, num_classes=config['num_classes'], multitask_learning=True, patience=patience, verbose=False)

        trainer.fit(dataloaders_dict, num_epochs=epochs, early_stopping=early_stopping, topk_accuracy=1, save_model=True)

        test_loss, test_acc, test_metric = trainer.validate(test_loader, 1)
        print('Testset (k={}) -> loss:{:.5f}, acc:{:.5f}, f1:{:.5f}'.format(k+1, test_loss, test_acc, test_metric))
        kfold_acc.append(test_acc)
        kfold_f1.append(test_metric)

    subject_kfold_acc[subject_id] = kfold_acc
    subject_kfold_f1[subject_id] = kfold_f1
    print()

print('Results')
print('subject_kfold_acc', subject_kfold_acc)
print('subject_kfold_f1', subject_kfold_f1)

```

Subject 1  
Testset (k=1) -> loss:188.74998, acc:0.97500, f1:0.98700  
Testset (k=2) -> loss:193.91165, acc:0.95800, f1:0.97900  
Testset (k=3) -> loss:187.81735, acc:0.97100, f1:0.98500

Subject 2  
Testset (k=1) -> loss:331.29451, acc:0.87100, f1:0.93100  
Testset (k=2) -> loss:291.48574, acc:0.92500, f1:0.96100  
Testset (k=3) -> loss:256.26734, acc:0.93300, f1:0.96600

Subject 3  
Testset (k=1) -> loss:206.96071, acc:0.97900, f1:0.98900  
Testset (k=2) -> loss:162.33227, acc:0.97500, f1:0.98700  
Testset (k=3) -> loss:180.66558, acc:0.97100, f1:0.98500

## ✓ Plot Results

```
[ ] # acc
subjects = []
acc = []
acc_min = 1.0
acc_max = 0.0

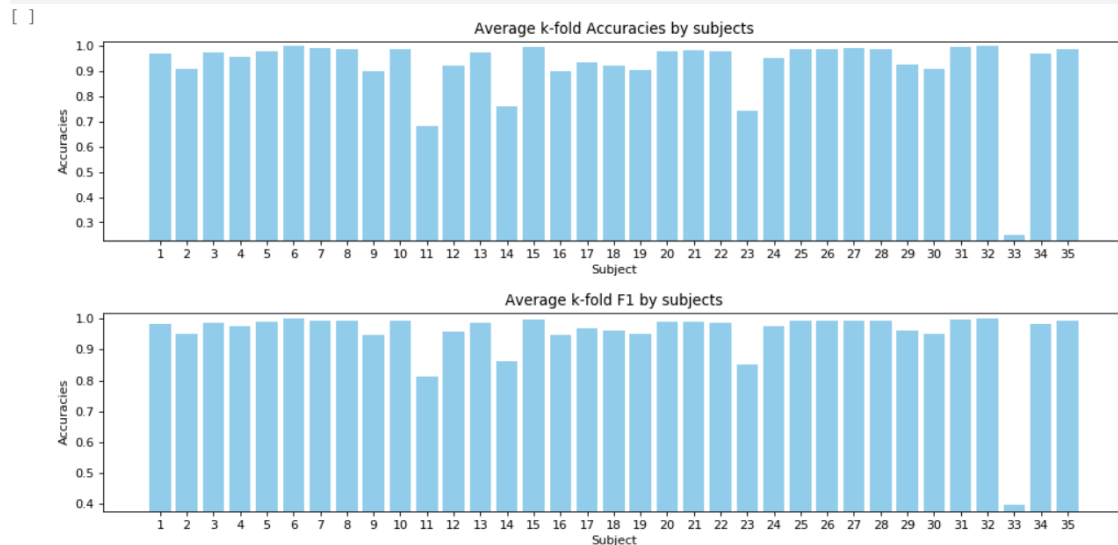
for subject_id in subject_kfold_acc:
    subjects.append(subject_id)
    avg_acc = np.mean(subject_kfold_acc[subject_id])
    if avg_acc < acc_min:
        acc_min = avg_acc
    if avg_acc > acc_max:
        acc_max = avg_acc
    acc.append(avg_acc)

x_pos = [i for i, _ in enumerate(subjects)]
figure(num=None, figsize=(15, 3), dpi=80, facecolor='w', edgecolor='k')
plt.bar(x_pos, acc, color='skyblue')
plt.xlabel("Subject")
plt.ylabel("Accuracies")
plt.title("Average k-fold Accuracies by subjects")
plt.xticks(x_pos, subjects)
plt.ylim([acc_min-0.02, acc_max+0.02])
plt.show()

# f1
subjects = []
f1 = []
f1_min = 1.0
f1_max = 0.0
```

```
plt.xlabel("Subject")
plt.ylabel("Accuracies")
plt.title("Average k-fold F1 by subjects")
plt.xticks(x_pos, subjects)
plt.ylim([f1_min-0.02, f1_max+0.02])
plt.show()
```

```
print('Average acc:', np.mean(acc))
print('Average f1:', np.mean(f1))
```



```
Average acc: 0.9224380952380953
Average f1: 0.9521142857142857
```

## CONCLUSION

This review introduced a profound learning technique that possibly empowers us to recognize different SSVEP improvements at the same time, along these lines planning



a visual guide of glaucoma patients, lessening visual field appraisal time, and producing solid experimental outcomes.

Taking into account late occasions during sickness episodes and pandemics where insignificant medical clinic arrangements are prescribed to be kept to a base, this appraisal technique can diminish the quantity of tests required, accordingly limiting any pointless or extra tests. Generally, this review empowers our future work to possibly evaluate glaucoma patients' visual field to distinguish fringe vision misfortune. To work on the unwavering quality of the appraisal results, using SSVEP could kill a patient's capacity to do the technique and the inconsistency of the patient's psychological state. Appraisal time could be chopped somewhere near distinguishing numerous SSVEP focuses on the double and producing a visual reaction map

## **REFERENCES**

- [1] S. Gao et al., "Visual and auditory brain-computer interfaces," IEEE Trans. Biomed. Eng., vol.61, no.5, pp.1436-1447, 2014.
- [2] X. Chen et al., "High-speed spelling with a noninvasive brain-computer interface," Proc. Natl. Acad. Sci. USA, vol.112, no.44, pp.E6058-E6067, 2015