THE AMERICAN UNIVERSITY IN CAIRO

الجامعة الأمريكية بالقاهرة

School of Sciences and Engineering Department of Computer Science and Engineering

Spring 2025

CSCE 3611: Digital Signals Processing

Project: EEG- Attention Classifier

GITHUB REPO: https://github.com/FaridaBey/EEG-Attention-classifier.git

Dr. Seif Eldawlatly

BY:

Farida Bey - 900212071

Arwa Abdelkarim - 900221451

Rana Taher - 900221430

Introduction:

In this project, we worked on classifying attention states—whether someone is focused or drowsy—using EEG brain signals. We were given data from five different people, and for each person, we had training and testing files that included EEG readings from seven channels over 15-second trials. Each trial was labeled as either focused (0) or drowsy (1).

Our goal was to build a model that can tell the difference between these two states. To do that, we first cleaned the data using a filter to remove noise, then we extracted features based on brainwave frequencies like alpha, theta, and delta. After that, we used the K-Nearest Neighbors (KNN) algorithm to classify the data and find the best settings for each person—like the best frequency band, EEG channel, and K value that gave the highest accuracy.

We also tried combining different features in a few ways (like using all channels or all frequency bands together) to see if it made the predictions better. In the end, we compared the results across all five subjects and explained what worked best for each one.

Project Description and approach used:

Project Description:

This project addresses the classification of human attention states—focused vs. drowsy—based on EEG signals. Data is provided for five subjects, each represented by two .mat files (train and test), for each file the following was provided:

1. Channels: Names of 7 EEG channels
2. Data: A 3D matrix (trials × samples × channels), where each trial contains 15 seconds of EEG data
3. Sampling rate (fs): Provided in the metadata
4. Labels: Binary values indicating the attention state (0 for focused, 1 for drowsy)

Approach Taken:

1.Common Average Reference filter:

To mitigate the common noise, we applied the CAR filter. For each time point across each trial and time sample the mean of the channels is computed, taken as a reference, and this reference is then subtracted from each channel. This enhances the uniqueness of each channel while suppressing any repetitions across the channels.

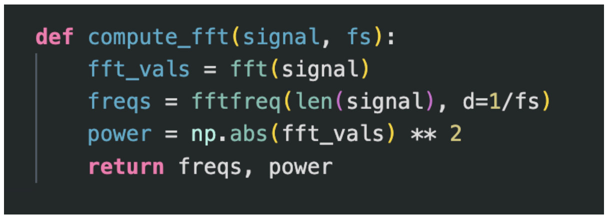
```python
def apply_car_filter(data):
    # Compute mean across channels for each trial and time sample
    car_data = data - data.mean(axis=2, keepdims=True)
    return car_data
```

The function takes the data array as an input, computes the average across the channels **`axis=2`** since in the shape of data is **`Shape: (trials, samples, channels)`**, and **`keepdims=True`** for compatibility with the subtraction, where we subtract the average from the original data. This function is then called for all test and train files of all five subjects to preprocess the data fields.

2. Feature Extraction:

Goal: Turn each trial into a **feature vector** to then use in the KNN classification.

2.1. Fourier Transform:



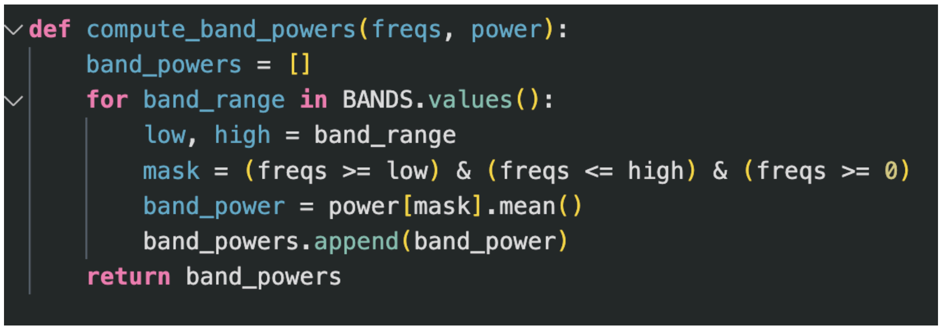
```python
def compute_fft(signal, fs):
    fft_vals = fft(signal)
    freqs = fftfreq(len(signal), d=1/fs)
    power = np.abs(fft_vals) ** 2
    return freqs, power
```

First, we apply the Fourier Transform to the EEG signal using the fft() function. This converts the signal from the time domain into the frequency domain, which shows what frequencies are present in the signal and how strong they are.

We use the fftfreq() function to determine the actual frequency values that correspond to each FFT output. This allows us to know, for example, how much power is in 1 Hz, 2 Hz, all the way up to the Nyquist frequency. The power spectrum is essential for identifying which brainwave bands (like delta, theta, etc.) are most active in the EEG signal. The function takes the number of samples, as well as the sampling interval.

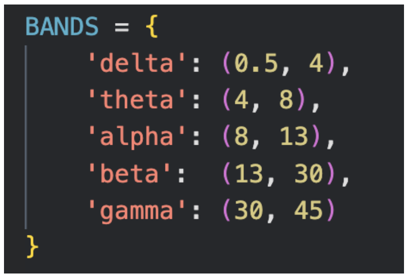
To understand how much of each frequency is present, we take the magnitude of each complex number and then square it to get the power at that frequency. This gives us a power spectrum, a representation of how the energy of the signal is distributed across different frequencies.

2.2. Band Powers:



```python
def compute_band_powers(freqs, power):
    band_powers = []
    for band_range in BANDS.values():
        low, high = band_range
        mask = (freqs >= low) & (freqs <= high) & (freqs >= 0)
        band_power = power[mask].mean()
        band_powers.append(band_power)
    return band_powers
```

We loop over the bands, representing the different frequencies of consciousness:

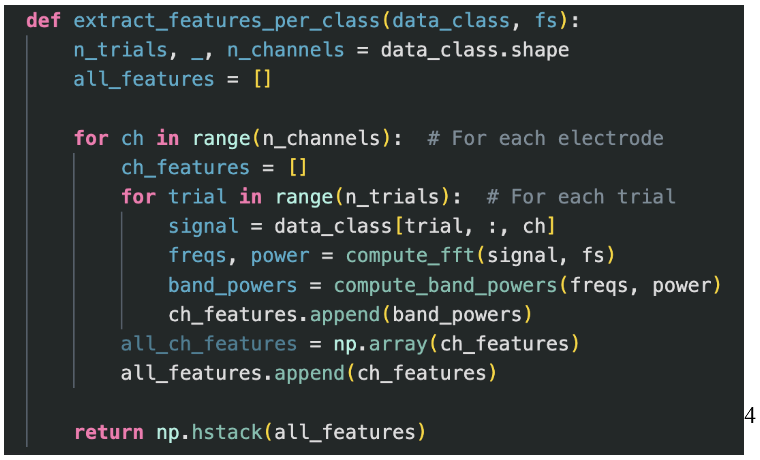


```python
BANDS = {
    'delta': (0.5, 4),
    'theta': (4, 8),
    'alpha': (8, 13),
    'beta':  (13, 30),
    'gamma': (30, 45)
}
```

Then we apply a mask to pick the frequencies in that band, keeping only the frequencies above the band's lower limit and below its upper limit, also we exclude negative frequencies as they are aliases.
Then we calculate the average of the power in that band.

2.3. Extract the features by combining 2.1 & 2.2:

Finally, we combine both functions in extract_features_per_class() to have the final feature vector needed.



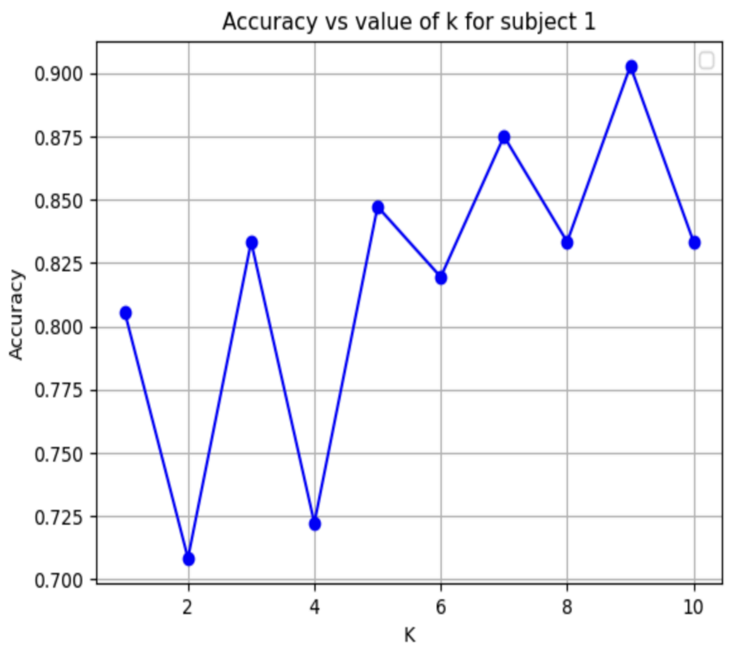
```python
def extract_features_per_class(data_class, fs):
    n_trials, _, n_channels = data_class.shape
    all_features = []

    for ch in range(n_channels):  # For each electrode
        ch_features = []
        for trial in range(n_trials):  # For each trial
            signal = data_class[trial, :, ch]
            freqs, power = compute_fft(signal, fs)
            band_powers = compute_band_powers(freqs, power)
            ch_features.append(band_powers)
        all_ch_features = np.array(ch_features)
        all_features.append(ch_features)

    return np.hstack(all_features)
```

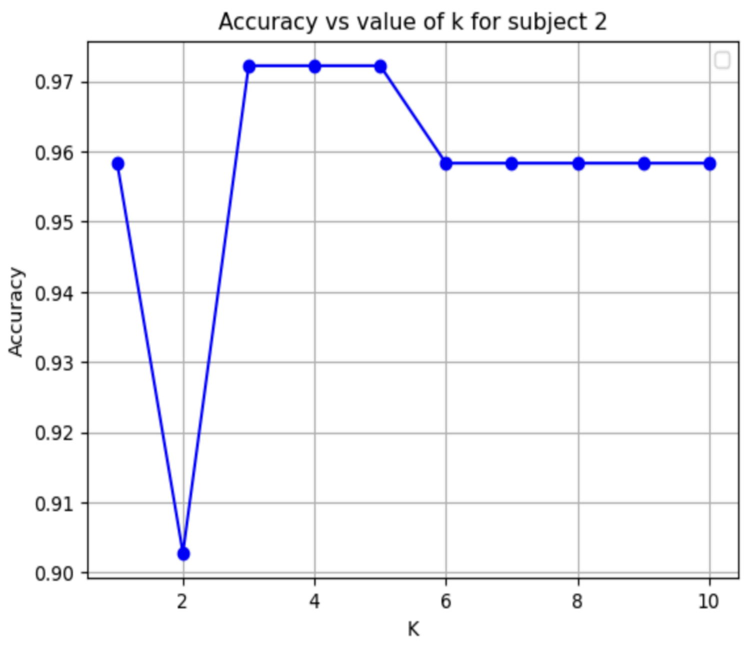
3. Classification: K-nearest neighbour:

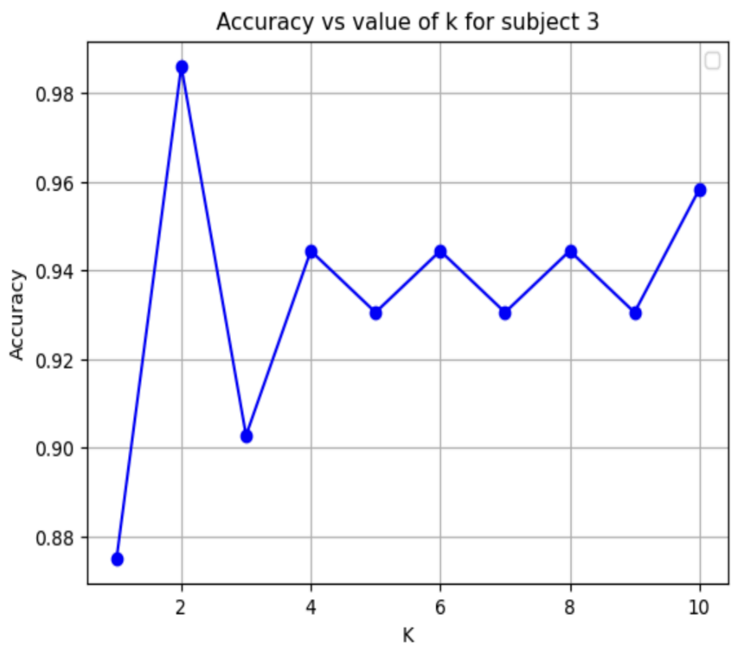
The main objective was to distinguish between focused and drowsy attention states based on the EEG-derived feature vectors. For each subject, we trained and evaluated the KNN classifier separately, ensuring that the model could adapt to individual differences in EEG patterns.
To determine the optimal value of K, we systematically varied K from 1 to 10. For each value of K, a KNN classifier was trained on the feature vectors from the training set and then tested on the corresponding test set for each subject. The classifier predicts the label for each test trial by examining the K closest feature vectors from the training set, using Euclidean distance as the metric. The predicted label is determined by majority vote among these neighbors.
The performance of the classifier was measured using accuracy, defined as the proportion of correctly classified trials in the test set. For each subject, we recorded the accuracy achieved for each value of K, ultimately selecting the K that yielded the highest accuracy as the optimal parameter for that subject. This subject-specific approach accounts for inter-individual variability in EEG signals and ensures robust classification across all five subjects.
Throughout this process, we maintained a consistent pipeline: preprocessing with the CAR filter, feature extraction via band power computation, and classification with KNN. By evaluating the classifier on held-out test data for each subject, we ensured that our results reflected genuine predictive performance rather than overfitting to the training data. This methodology provides a comprehensive and reliable assessment of the feasibility of EEG-based attention state classification using band power features and KNN.

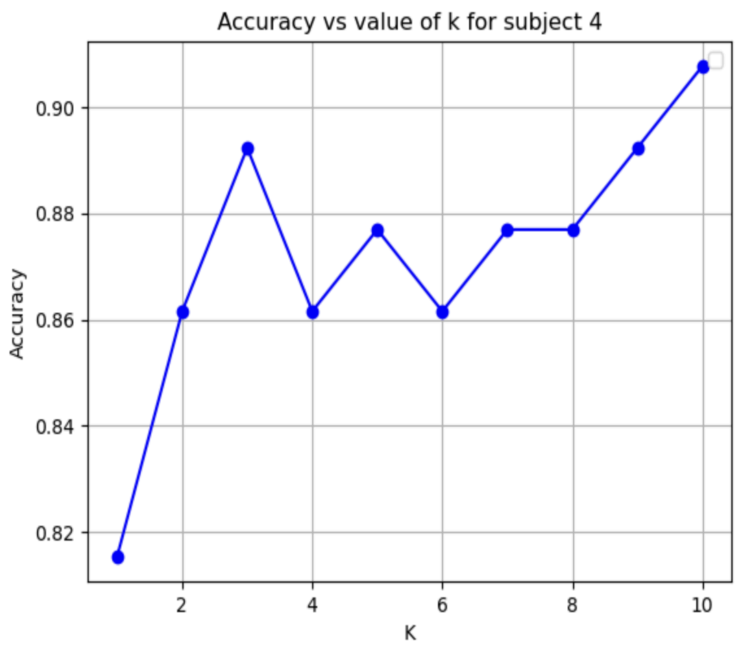
Results and Observations:

The following are the results obtained from using the data in its original format:







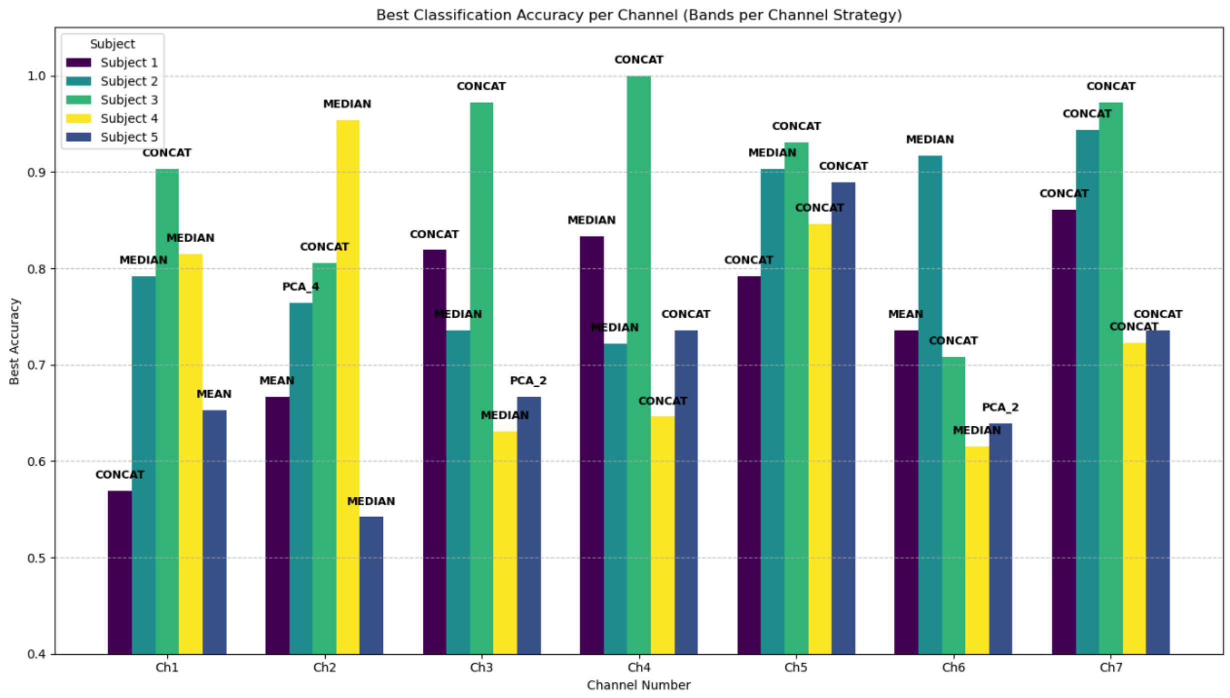


The above figures show how the accuracy of the classification for all five subjects is affected by the change in the value of K. As you can see in the first subject graph the value of k that yields the best accuracy of approximately 0.903 with an error of 0.0972 is 9 at the band alpha of channel 4 while the second subject best accuracy of approximately 0.972 with an error of 0.027 is yielded at K of values 3, 4, and 5 at band alpha of channel 7. For subject 3 the highest accuracy obtained is 0.9861with an error of 0.0139 at K of 2 at band theta of channel 4 and this is the highest accuracy of classification achieved from all subjects. Then for subject 4 the highest accuracy achieved is 0.9077 with an error of 0.0923 at K of 10 at band alpha of channel 2 . Lastly, the highest accuracy achieved for subject 5 is 0.833 with an error of 0.167 at K of 3 at band delta of channel 5.

All subjects yield their highest accuracy at different values of K except subjects 2 and 5 the best value of K for both of them is 3. Moreover, the alpha band was the most common band among the subjects for yielding the highest accuracy as three subjects had alpha as the best band. For the channels only channel 4 was found to get the best accuracy for more than one subject which are 1 and 3 while the channels for the remaining three subjects were different.

Combining Features:

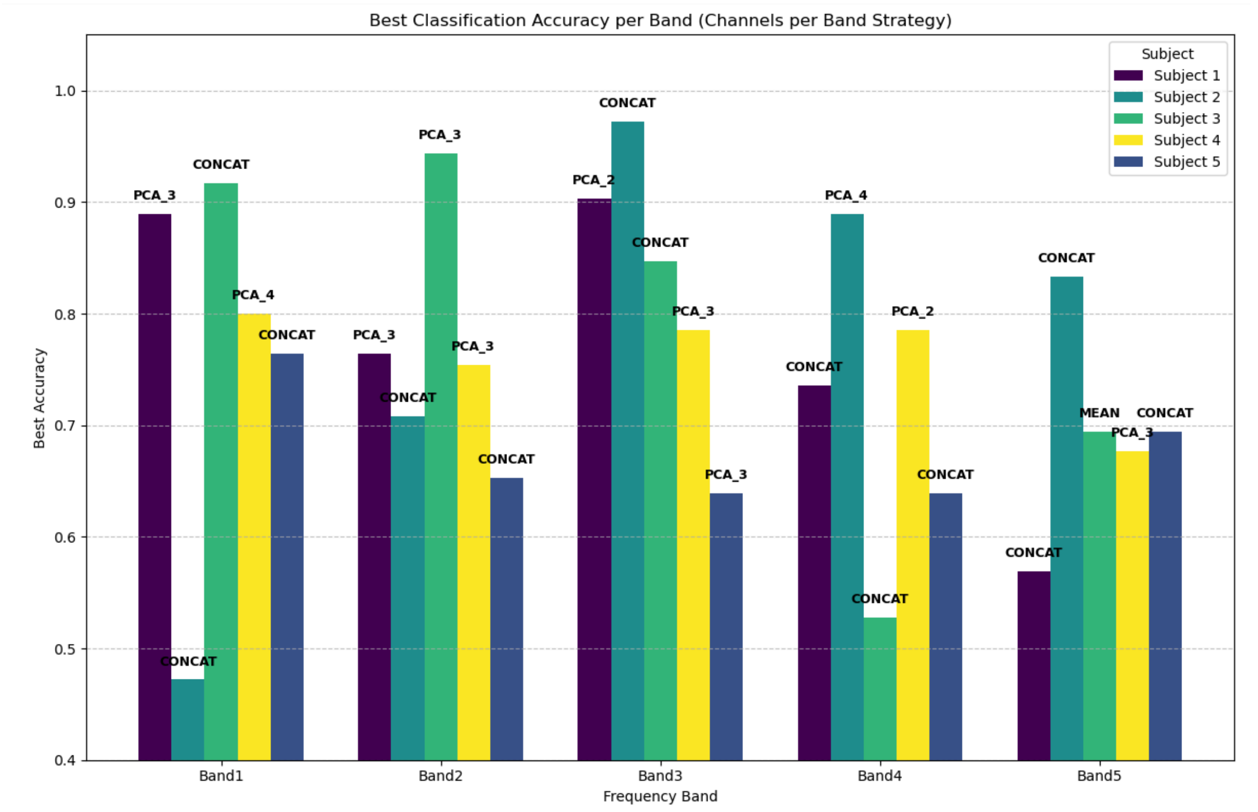
1- Combining all frequency bands for each channel



To combine all the frequency bands for each EEG channel, we tried four different methods: concatenation, mean, median, and PCA. Out of all these, concatenation worked the best and gave the highest classification accuracy across subjects. With concatenation, we simply join the features from all frequency bands into one big feature set for each channel. This way, we keep all the important details from each band, giving the model more information to learn from. It helps the classifier pick up patterns that could be missed if we simplified the data too much. On the other hand, the mean and median methods shrink the data by averaging or taking the middle value of features across all bands. This can remove some noise, but it also gets rid of useful and unique information from each band. These methods treat all bands the same, even though each one might have different signals that help the model. PCA (Principal Component Analysis) tries to reduce the size of the data by keeping only the parts that change the most. While this helps make the model faster and prevents overfitting, it also mixes up the original signals and makes them harder to understand. Sometimes, this means the model misses out on smaller but important patterns. Overall, concatenation stands out because it keeps all the original information from every frequency band, letting the model use everything available to make better predictions. It's a simple method, but it works really well because it doesn't throw away any useful data. This is why we chose it and saw the best results with it.

2- Combining all channels for each frequency band

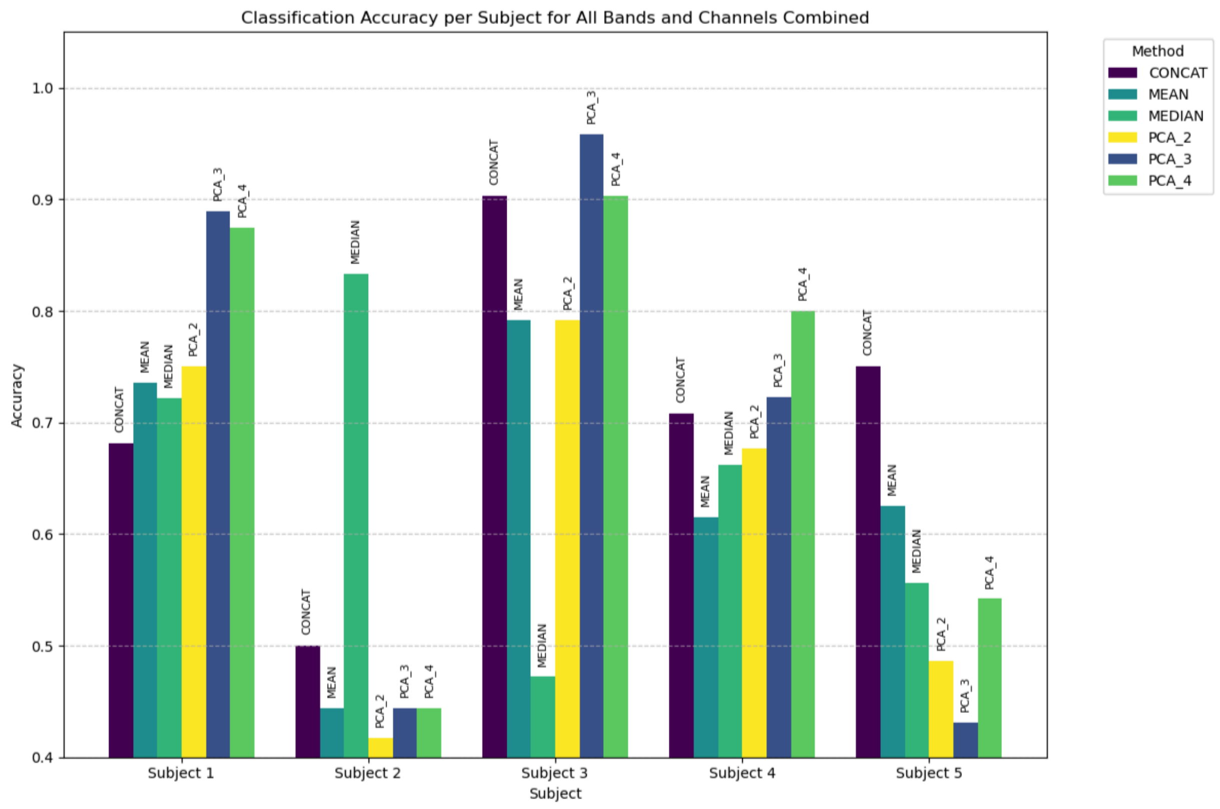
This graph represents the best classification accuracy for each frequency band after



combining all channels per band. Among the different feature combination methods tested: concatenation, mean, median, and PCA. Concatenation again shows the most consistent and strongest performance across subjects and bands. Concatenation works by merging the features from all channels within the same frequency band into a single, large feature vector. This approach preserves the full detail and variability from each channel, allowing the model to capture more complex patterns in the data. Compared to this, PCA (e.g., PCA_3 or PCA_4) performs well in some specific cases like for Band 2 and Band 4, but it's not as reliable overall. Since PCA reduces the number of features by projecting them into a smaller space, it may discard some subtle but useful information in the process. On the other hand, mean and median severely reduce the feature richness by collapsing multiple channel features into a single average or middle value, which leads to noticeable drops in accuracy in most bands. Overall, concatenation proves to be the most effective strategy for combining information across channels in each frequency band. It provides a richer and more complete representation of the data, which directly contributes to better classification performance for most subjects.

3- Combining all frequency bands and all channels

To combine all the EEG channels and frequency bands for each subject, we tested several feature aggregation strategies: concatenation, mean, median, and PCA with different



numbers of components (2, 3, or 4). Among these, PCA with 3 or 4 components often gave the best classification results, especially for subjects 1, 3 and 4. This shows that PCA was able to reduce the feature space while still keeping the key patterns the model needs to classify correctly. By focusing on the most important variations in the data, PCA helps the model avoid overfitting and speeds up learning. However, it doesn't always work best for everyone—for example, subject 2 had the highest accuracy with the median method instead. This highlights that some subjects might have more noise or outliers, where simpler aggregation methods like median can actually help by smoothing things out. Concatenation, while powerful in theory, didn't always perform best here, possibly because combining too many features from all channels and bands can overwhelm the model or introduce redundant information. Mean and median provide simpler summaries, but they can sometimes miss the complexity that matters. Overall, PCA stands out here.