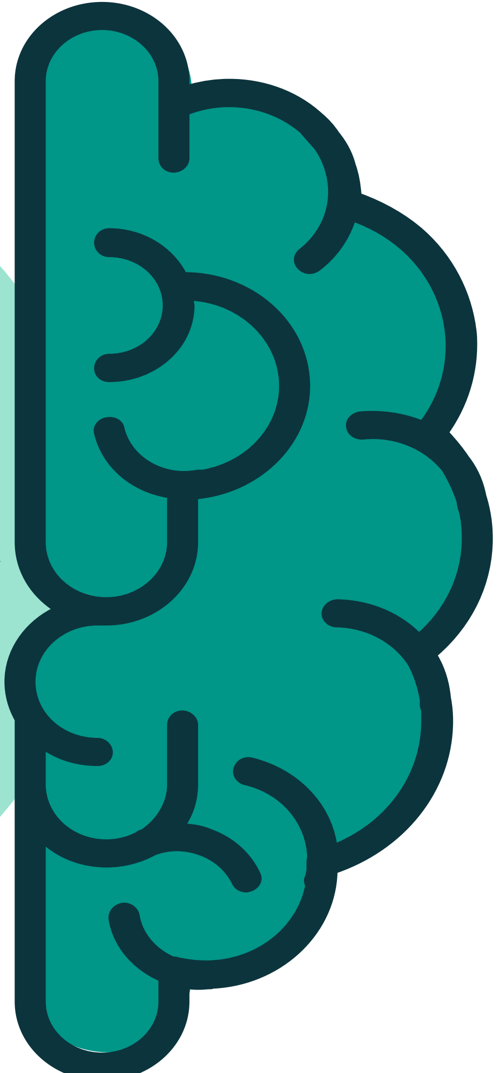
GNNs for classification of Multidimensional Time Series

By:
Alejandro Astruc Lopez



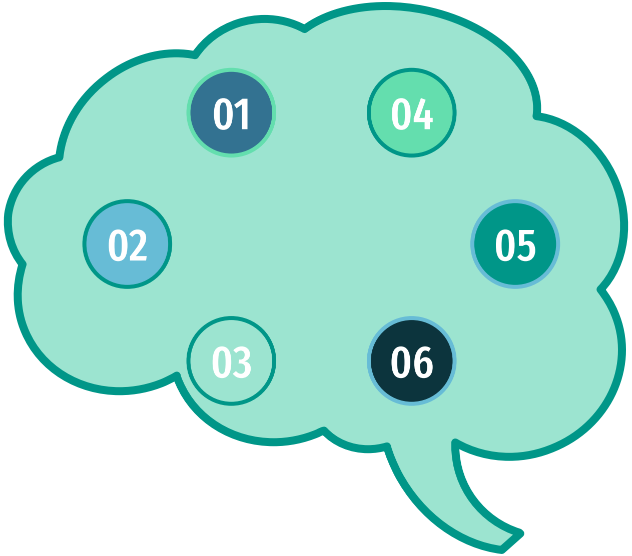
Introduction

Electroencephalography (EEG)

Electric Signals

Low signal-to-noise Ratio

High Dimensionality



Low Spatial Resolution

Thousands of Neurons

Large Synchronous act

Target Problem

- Eleven right-handed individuals: 4M and 7F; M. age = 55yrs, SD = 5.8.

- Decision-making tasks between 2 reaching movements.

- 12 blocks of 108 trials each.

- The factor of study is the three motivated states:
    - Solo, 0.
    - Easy, 1,
    - Hard, 2.

EEG data classification: Conventional Approach

Neural Oscillations

Also known as brainwaves.

Correlations

Analysing pairs of nodes.

Power Spectral Density

Frequency domain.

EEG data classification: Automated Feature Extraction

CNNs

Convolutional Neural Networks

LSTM

Long Short-Term Memory Networks

GNNs

Graph Neural Networks:

- Generalize CNNs.
- **Non-local**, **Long-distance** relationships
- Capture **non-Euclidean**, curved spaces.

Graph Convolutions

Graph Convolutions

$$\mathcal{G} = (V, E) \qquad A \in \mathbb{R}^{n \times n}$$

$$A_{uv} = \begin{cases} 1 & \text{if } (u,v) \in E \\ 0 & \text{if } (u,v) \notin E \end{cases}$$

$$h_v = \phi\left(x_v, \bigoplus_{u \in \mathcal{N}_v} \psi(x_v, x_u) \right)$$

$$E \subseteq V \times V$$

$$|V| = n$$

$$v \in V$$

$$x_v \in \mathbb{R}^p$$

$$X \in \mathbb{R}^{n \times p}$$

$$X = [x_1, \ldots, x_n]^T$$

GCN Convolution

$$\mathbf{X}' = \hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{\Theta},$$

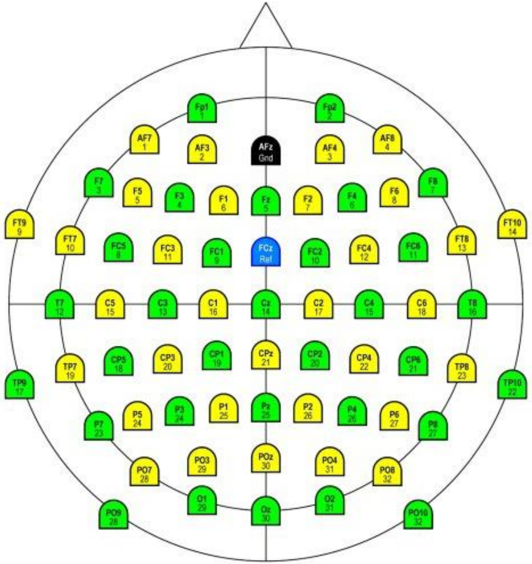
node-wise formulation:

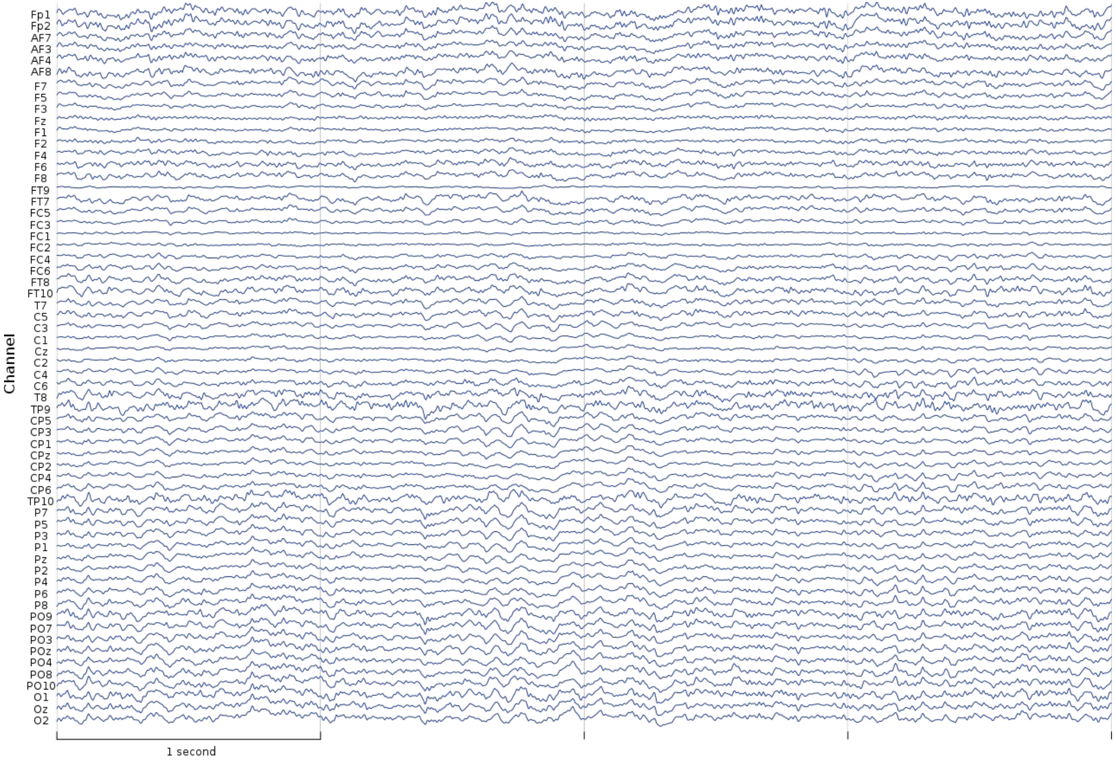
$$\mathbf{x}'_i = \mathbf{\Theta}^\top \sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{e_{j,i}}{\sqrt{\hat{d}_j \hat{d}_i}} \mathbf{x}_j$$

Methodology

Data

Data



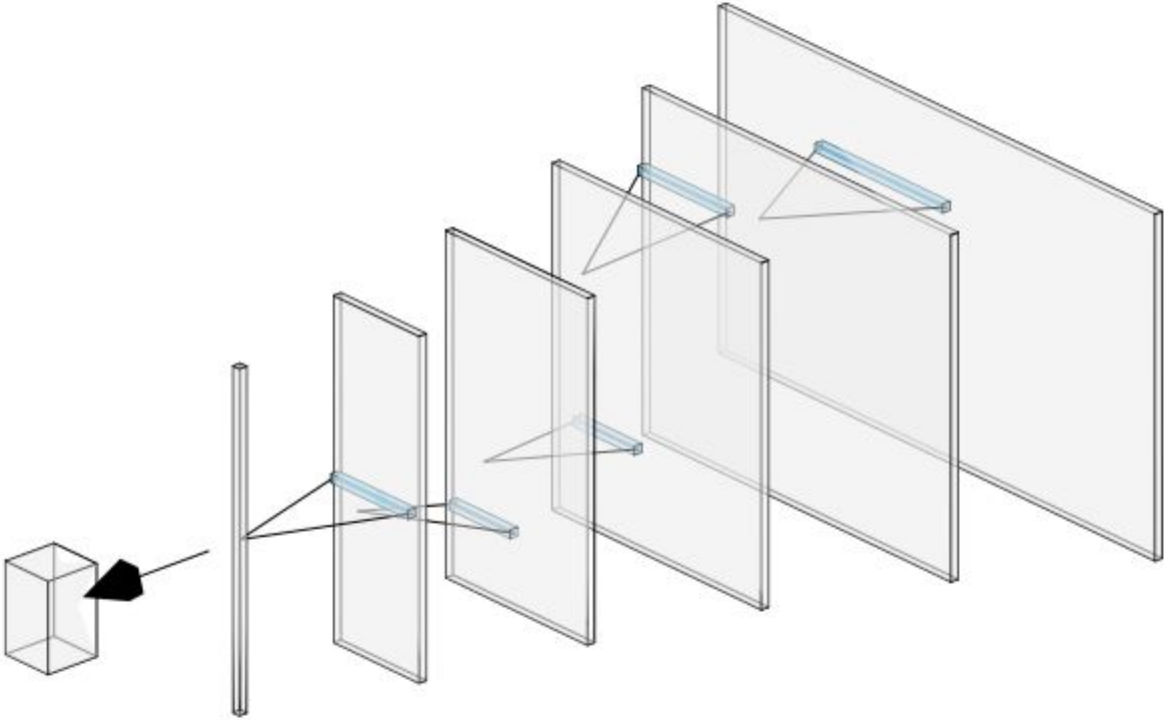


By Laurens R. Krol - Own work, CC0,
https://commons.wikimedia.org/w/index.php?curid=96843750

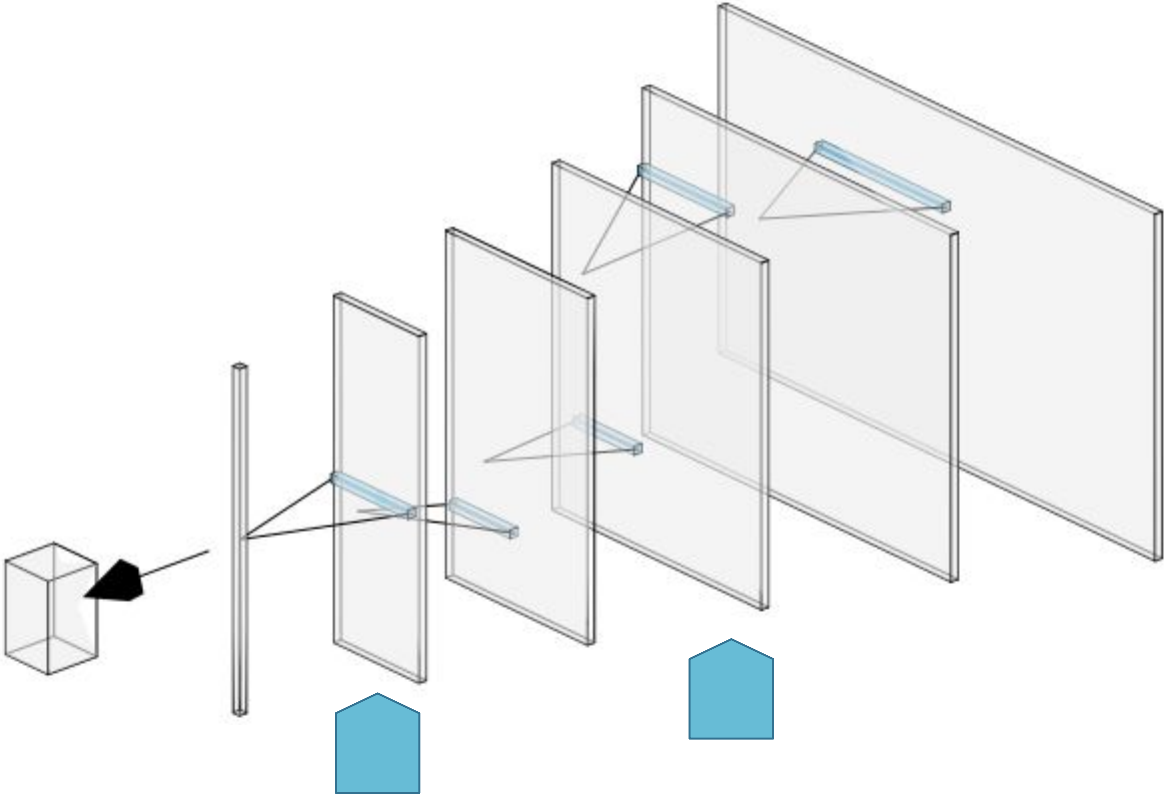
Methodology

Models

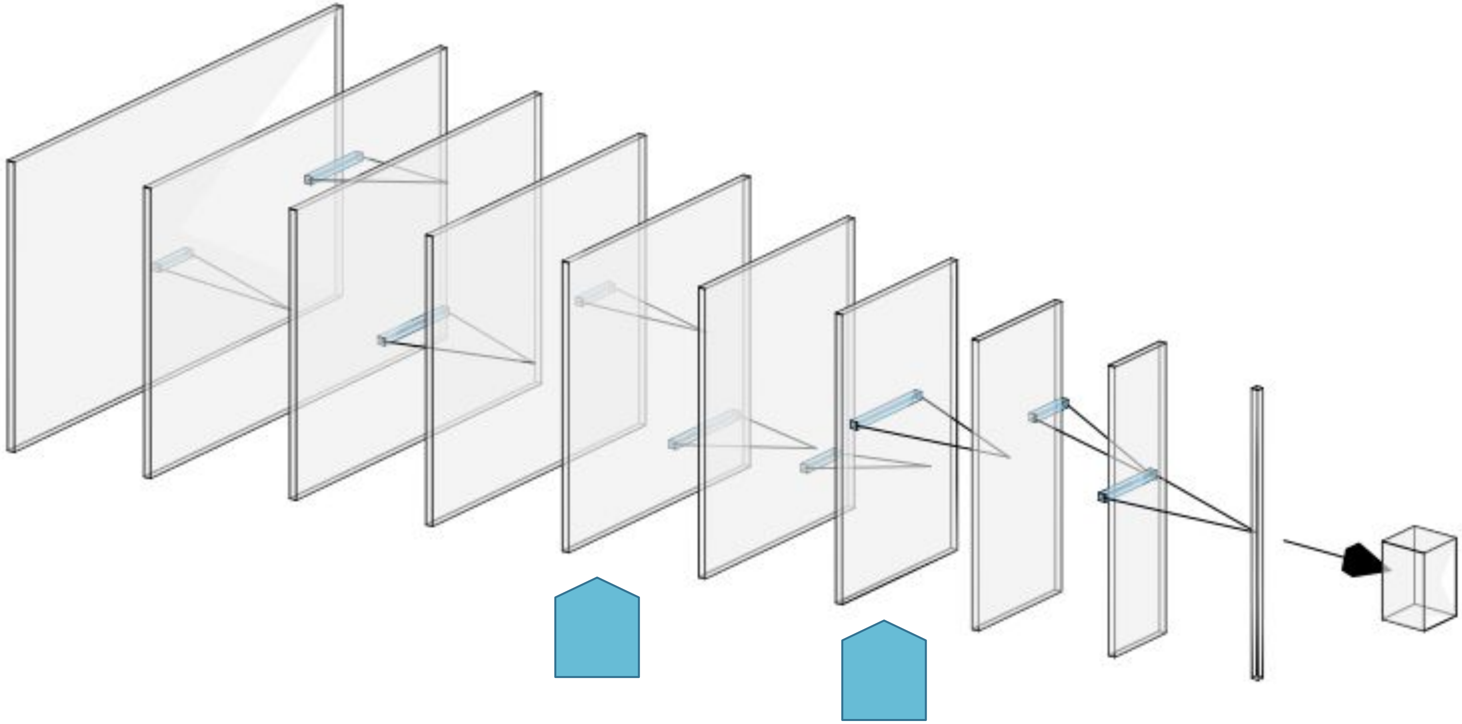
TimeAggNet



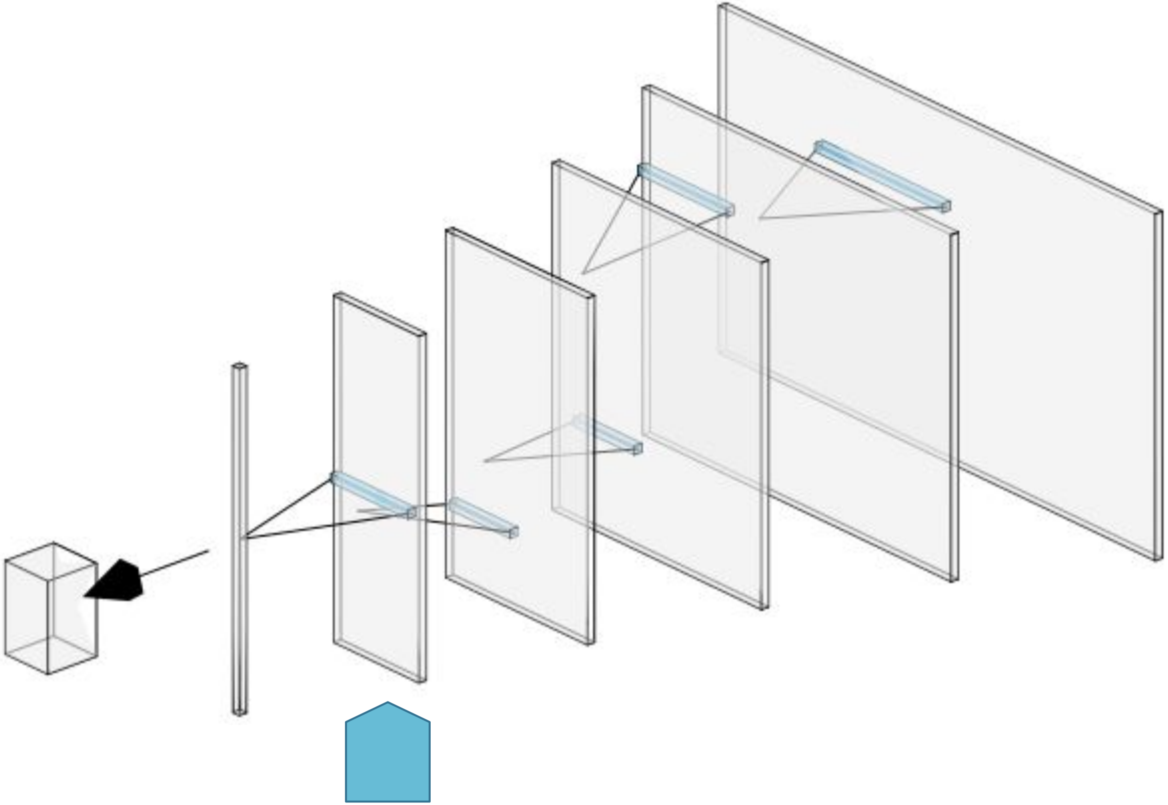
TimeGraphNet



DeepTimeGraphNet



SimpleTimeGraphNet



Adaptive Modification

$$A = XWX^T + I$$

$$X \in \mathbb{R}^{n \times \tau}$$

$$W \in \mathbb{R}^{\tau \times \tau}$$

Methodology

Experiments

Experiments

- ➔ 80% train 20% test split.
- ➔ 250 epochs using the Adam optimizer with a learning rate of 0.001.
- ➔ The class labels are one hot encoded and the loss function of choice was mean squared error (MSE).
- ➔ For the errors we will use a 95% confidence interval using a normal approximation based on the test set.

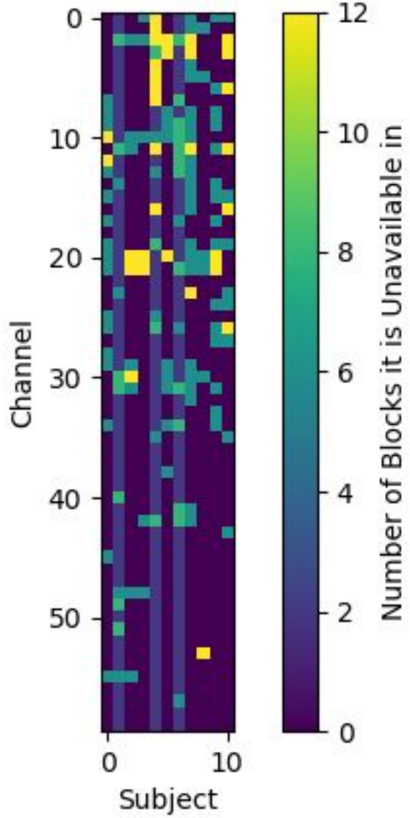
Results

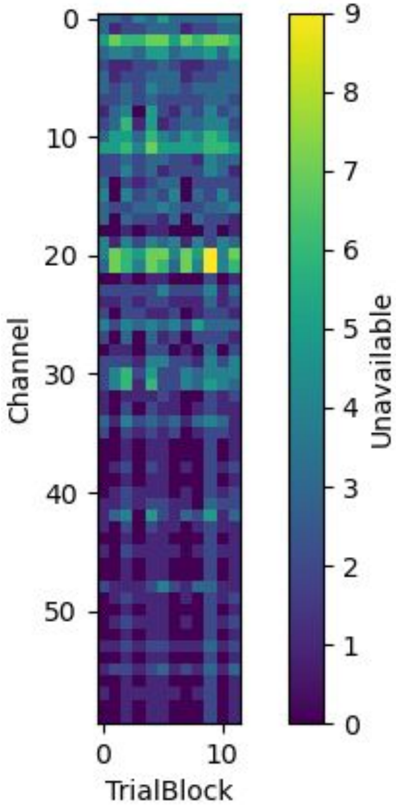
Data exploration

Class Balance across subjects

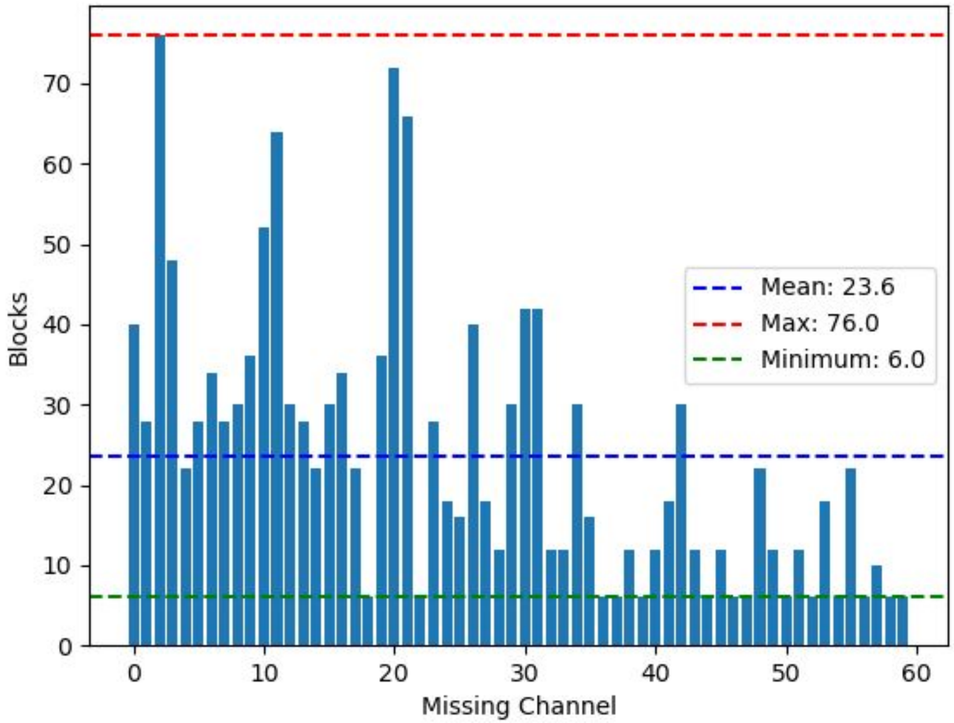
	% of class 0	% of class 1	% of class 2	Total samples
Subject 0	33	33	33	1296
Subject 1	36	32	32	1017
Subject 2	33	33	33	1296
Subject 3	33	33	33	1296
Subject 4	40	20	40	1080
Subject 5	33	33	33	1296
Subject 6	40	40	20	1080
Subject 7	33	33	33	1296
Subject 8	33	33	33	1296
Subject 9	33	33	33	1296
Subject 10	33	33	33	1296

Missing channels across subjects, within all blocks of the whole data





Missing channels across subjects, within all blocks of the whole dat



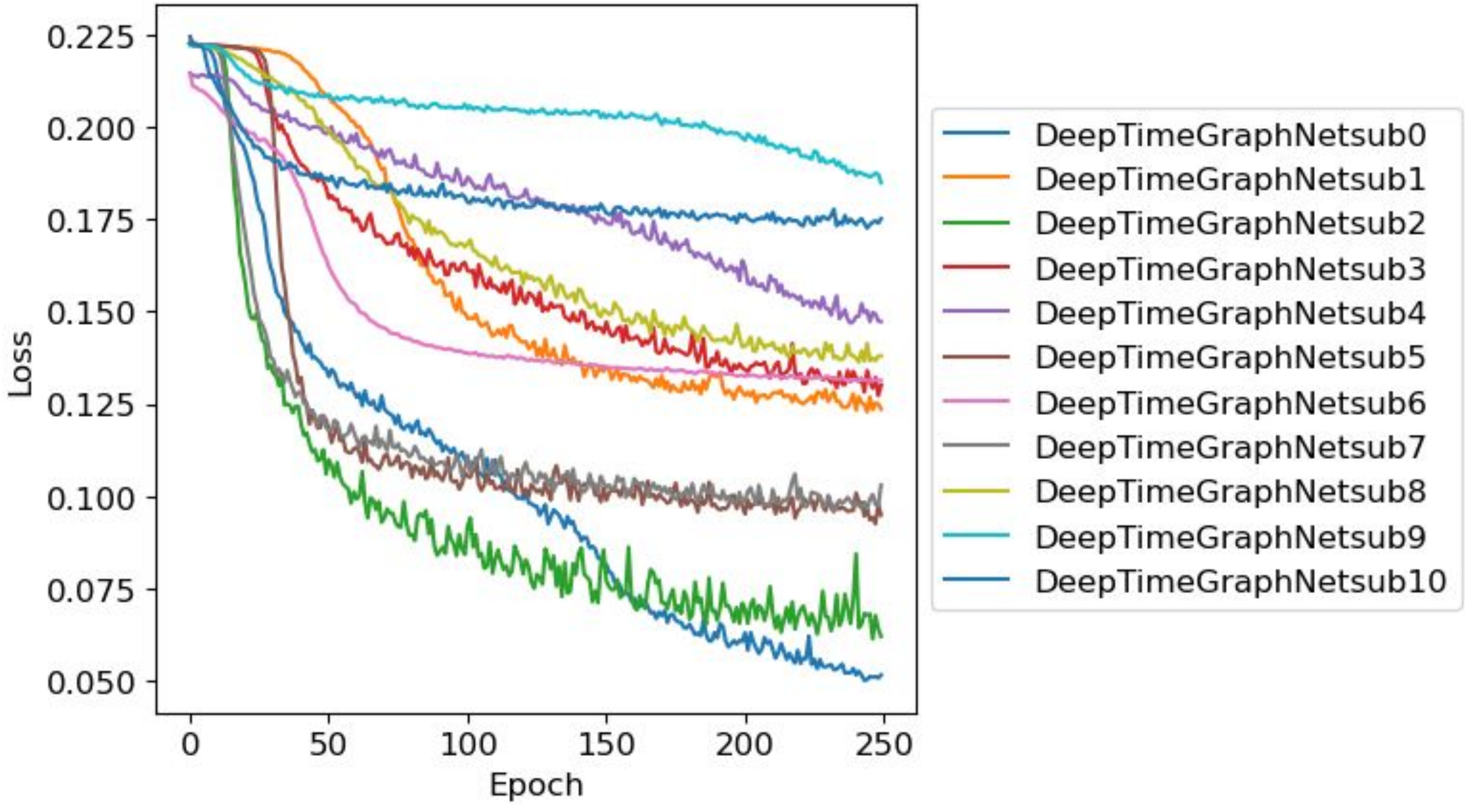
Results

Models: Training

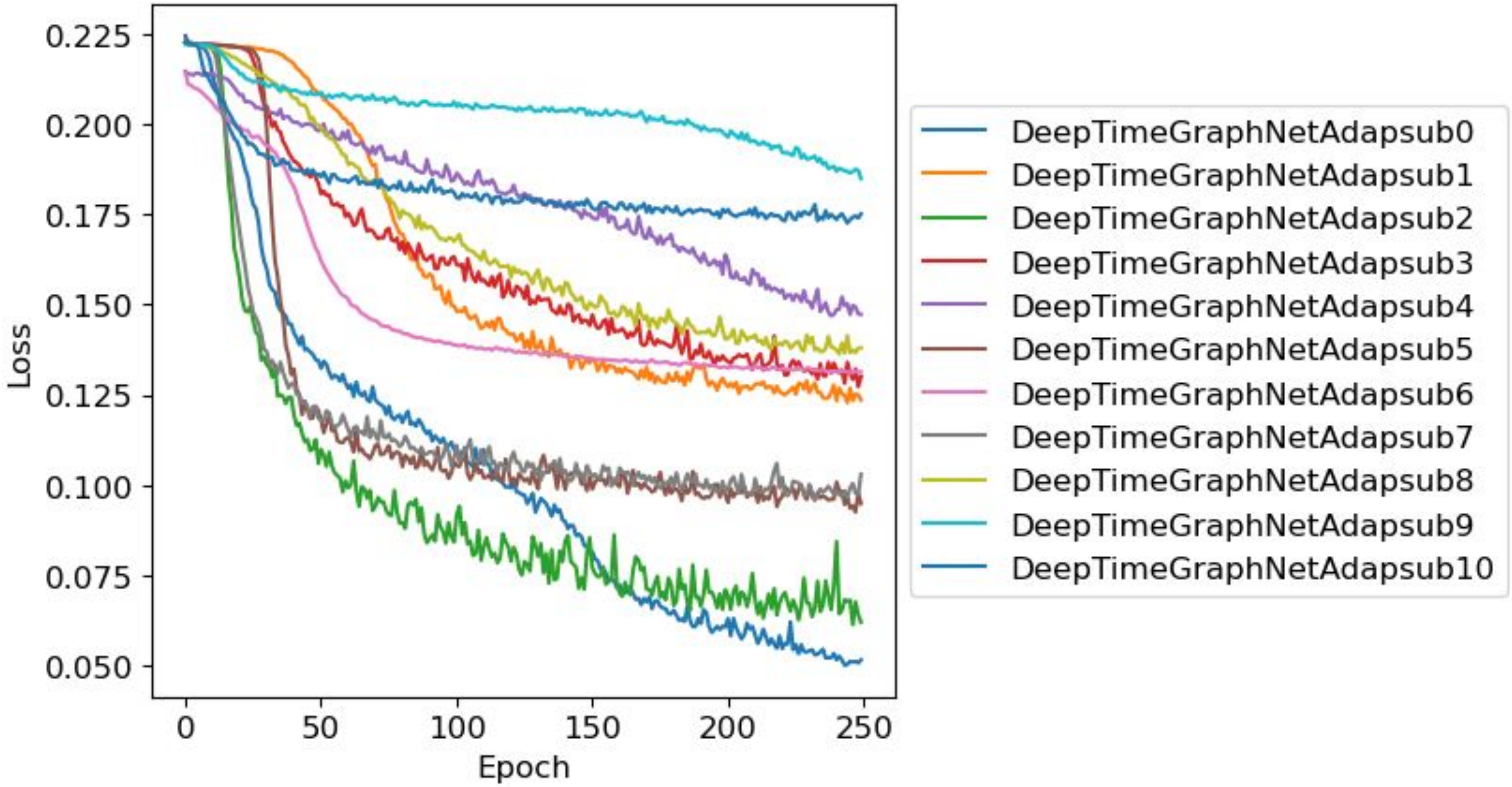
Average loss and metrics on the training set for all models across all subjects

	Loss	Accuracy	Recall	Precision	F1
TimeAggNet	0.112	**0.768**	**0.766**	**0.768**	**0.765**
TimeGraphNet	0.117	0.756	0.752	0.758	0.753
TimeGraphNetAdap	0.117	0.756	0.752	0.758	0.753
DeepTimeGraphNet	0.122	0.732	0.719	0.746	0.716
DeepTimeGraphNetAdap	0.122	0.732	0.719	0.746	0.716
SimpleTimeGraphNet	0.129	0.715	0.709	0.715	0.709
SimpleTimeGraphNetAdap	0.129	0.715	0.709	0.715	0.709

Training loss for the DeepTimeGraphNet



Training loss for the DeepTimeGraphNetAdap



Results

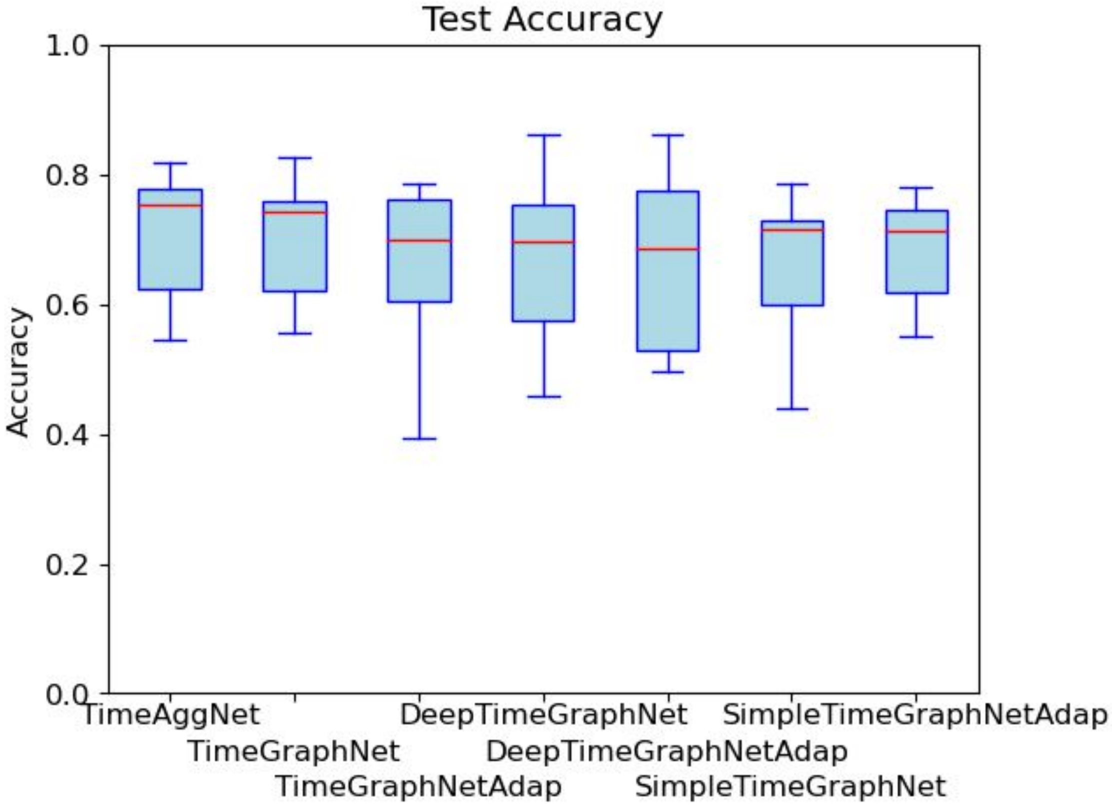
Models: Testing

Average test metrics for the different models across subjects

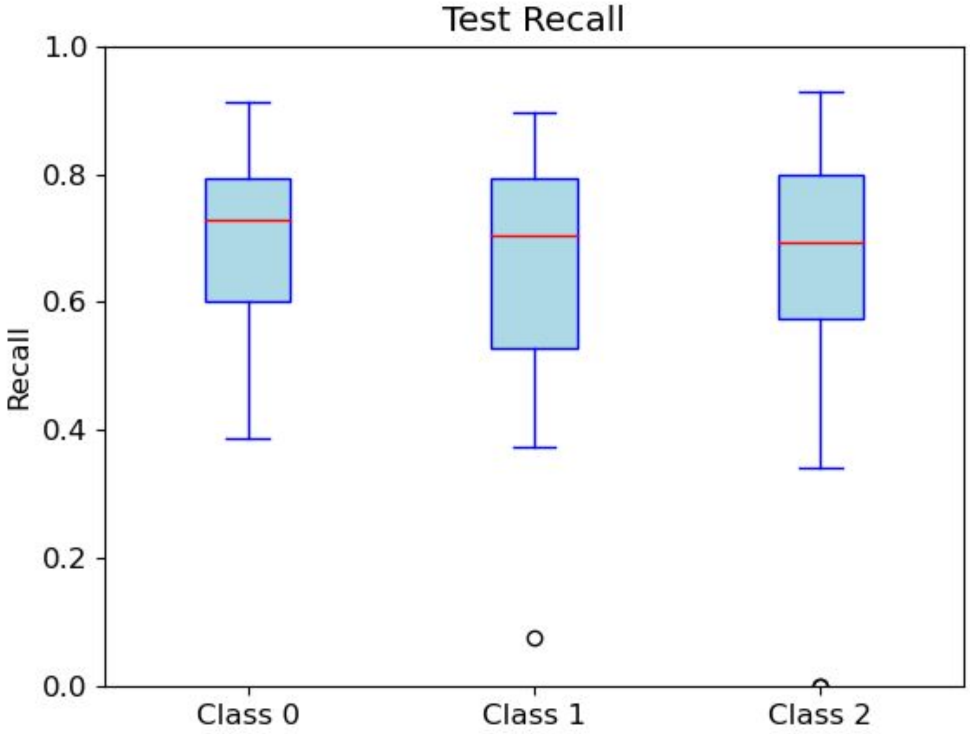
	Accuracy	Recall class 0	Recall class 1	Recall class 2
TimeAggNet	**0.709**	**0.742**	0.678	0.695
TimeGraphNet	0.700	0.713	0.655	**0.722**
TimeGraphNetAdap	0.671	0.672	0.661	0.660
DeepTimeGraphNet	0.678	0.696	0.671	0.635
DeepTimeGraphNetAdap	0.661	0.655	0.676	0.632
SimpleTimeGraphNet	0.664	0.663	0.593	0.716
SimpleTimeGraphNetAdap	0.684	0.706	**0.679**	0.637

	Precision class 0	Precision class 1	Precision class 2
TimeAggNet	0.718	0.699	**0.706**
TimeGraphNet	**0.719**	0.686	0.690
TimeGraphNetAdap	0.687	0.684	0.641
DeepTimeGraphNet	0.690	**0.709**	0.605
DeepTimeGraphNetAdap	0.690	0.660	0.633
SimpleTimeGraphNet	0.673	0.662	0.656
SimpleTimeGraphNetAdap	0.707	0.671	0.620

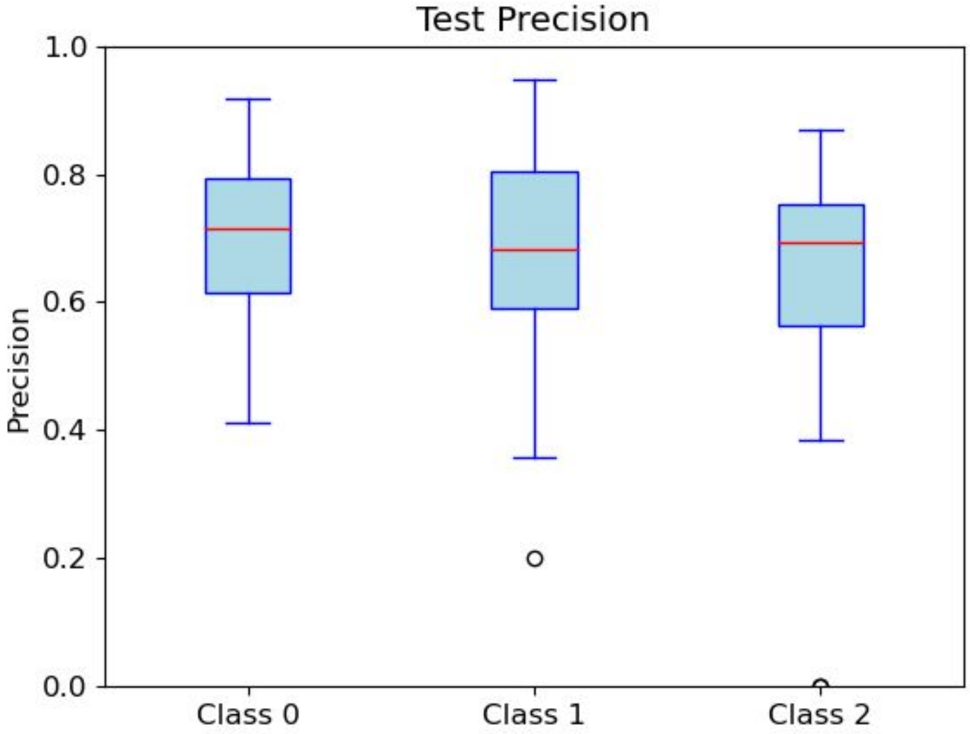
Test accuracy distributions for the different models across all subjects



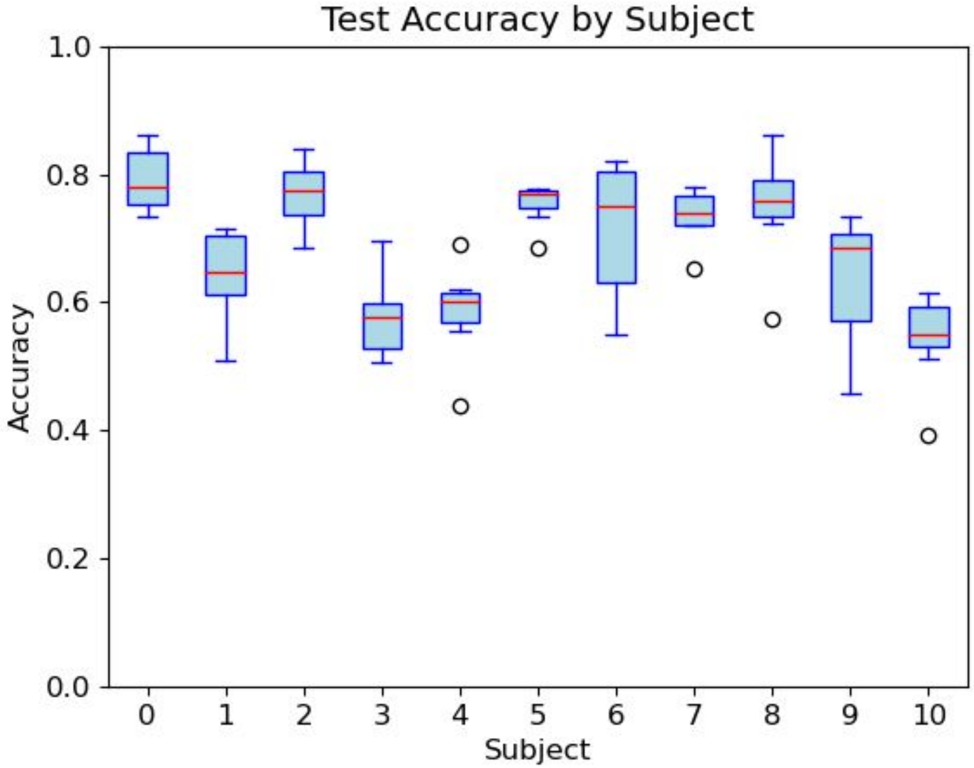
Recall metrics grouped by class for all models over all subjects



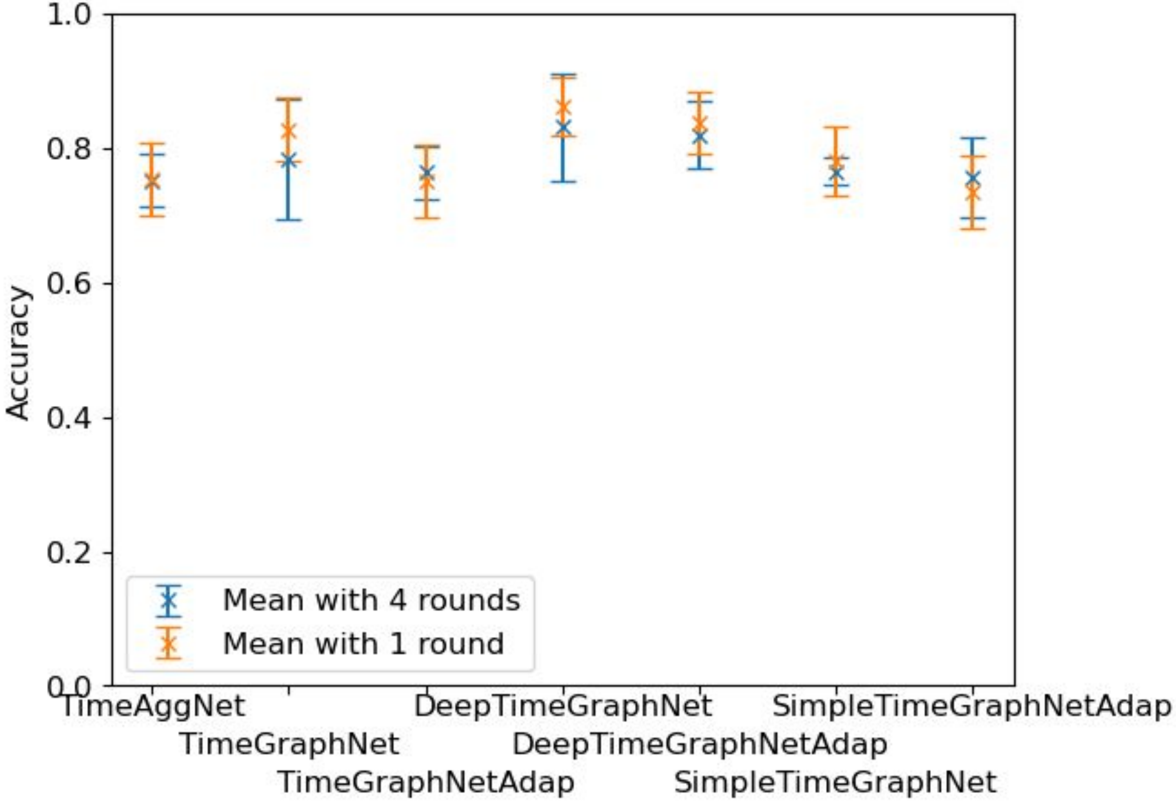
Precision test metrics grouped by class for all models over all subjects



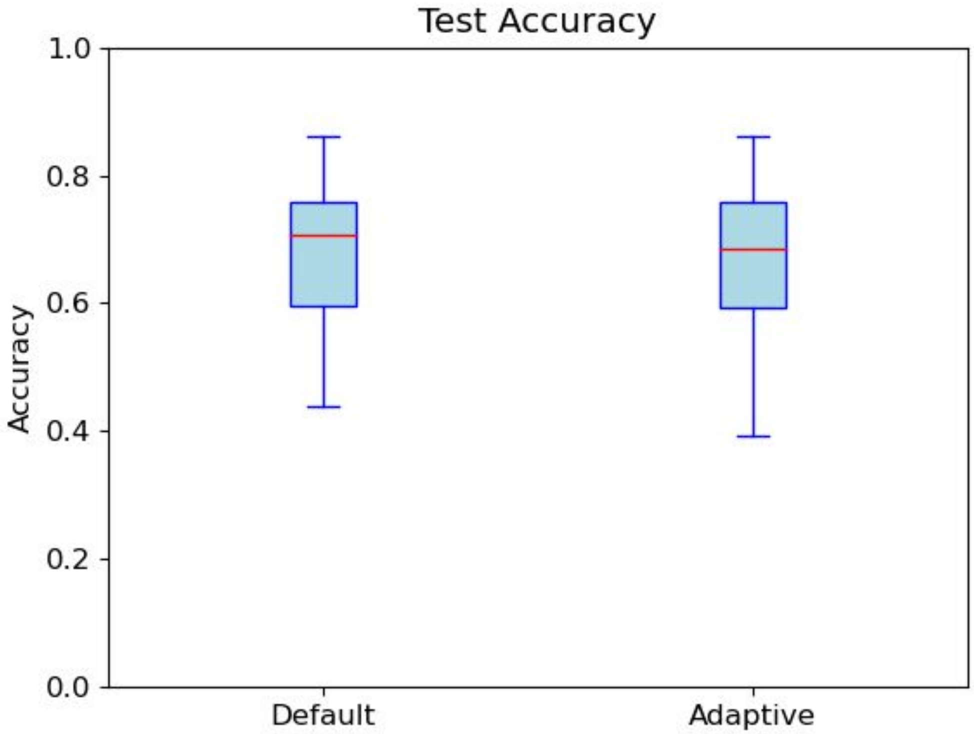
Accuracy distributions of all models grouped by subject



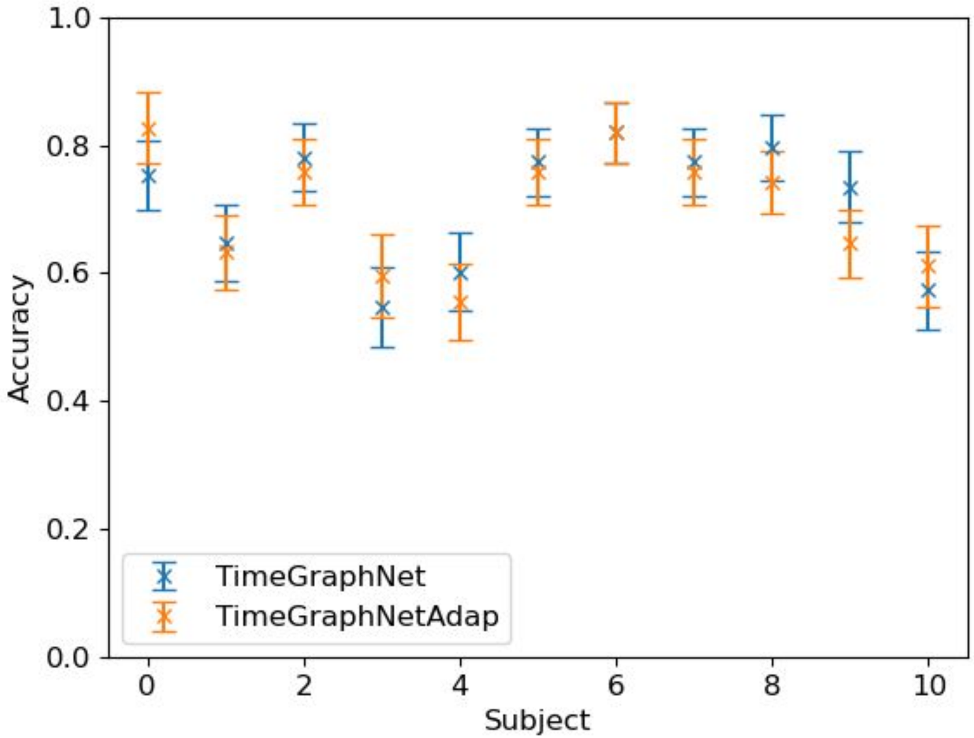
Average accuracies for subject 0 with a single and 4 repetitions for all models.



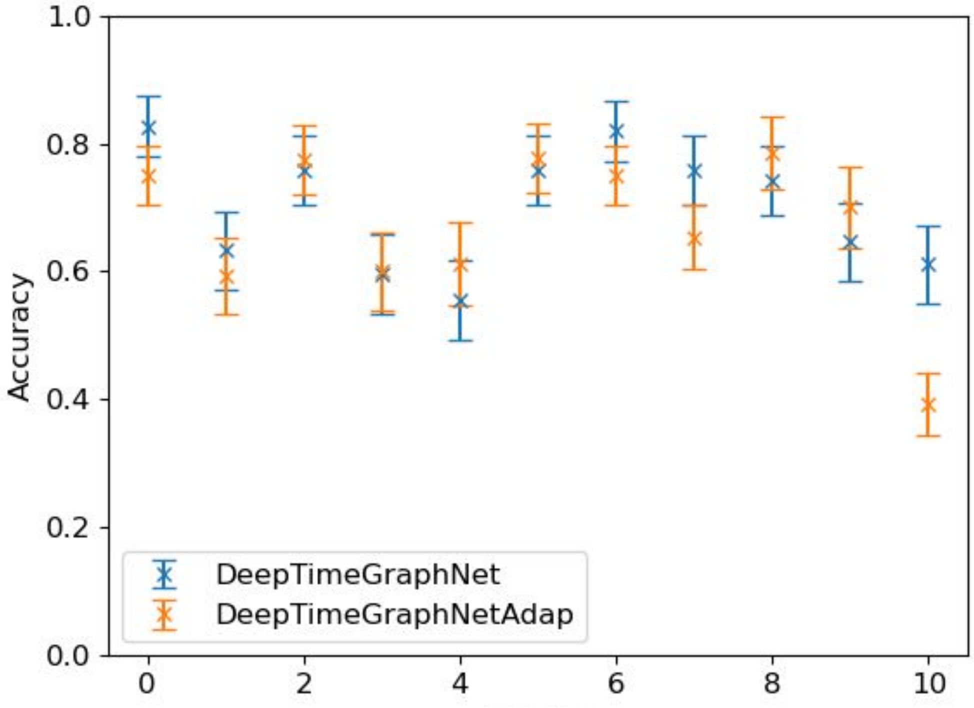
Test accuracy distribution over all graph models and subjects grouped into default models and adaptive modifications



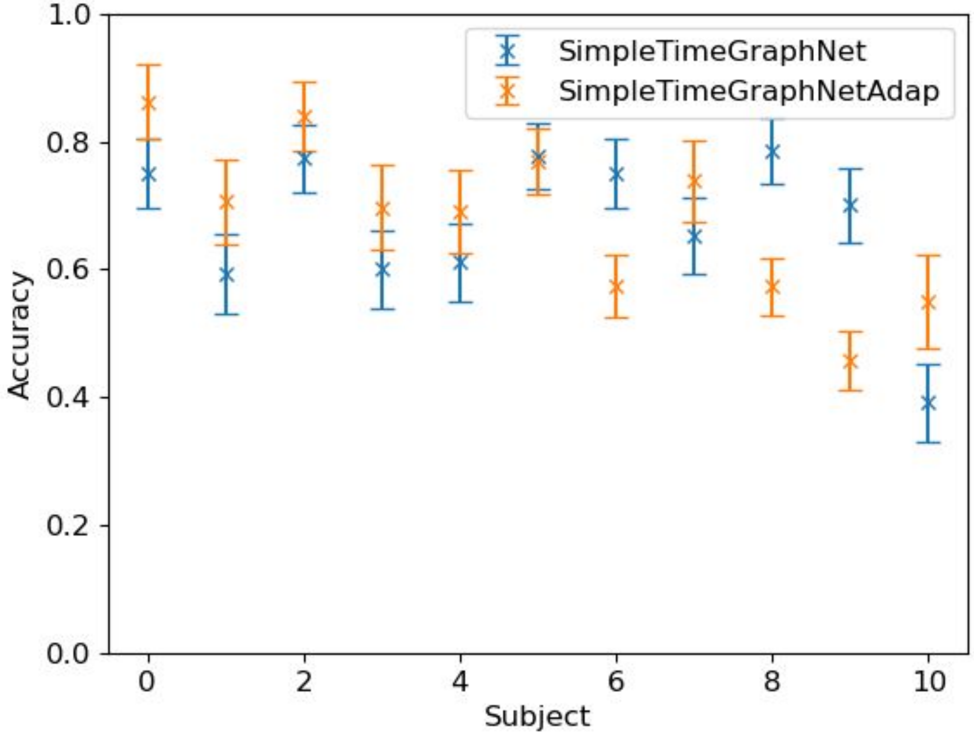
Test accuracies for the TimeGraphNet and TimeGraphNetAdap models on the different subjects



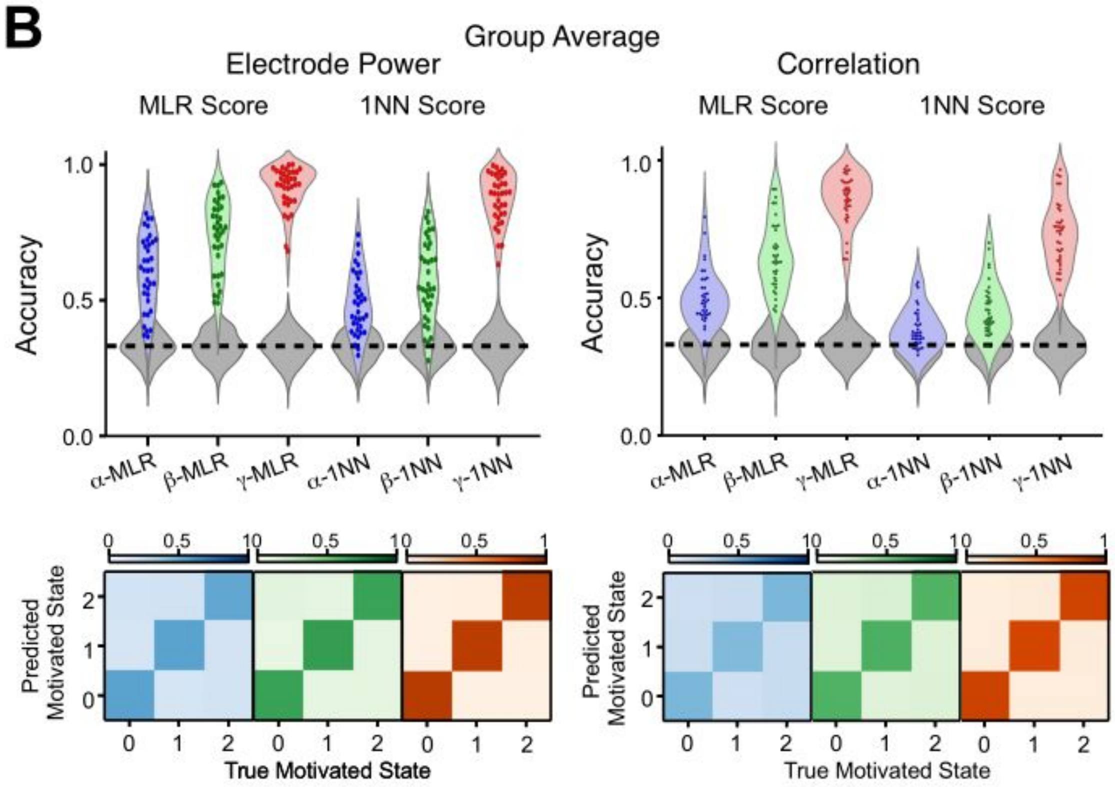
Test accuracies for the DeepTimeGraphNet and DeepTimeGraphNetAdap models on the different subjects



Test accuracies for the SimpleTimeGraphNet and SimpleTimeGraphNetAdap models on the different subjects



Comparison with previous study



Conclusions

Conclusions

01 Our models can automatically extract relevant features from the unfiltered signal for the motivation state classification task.

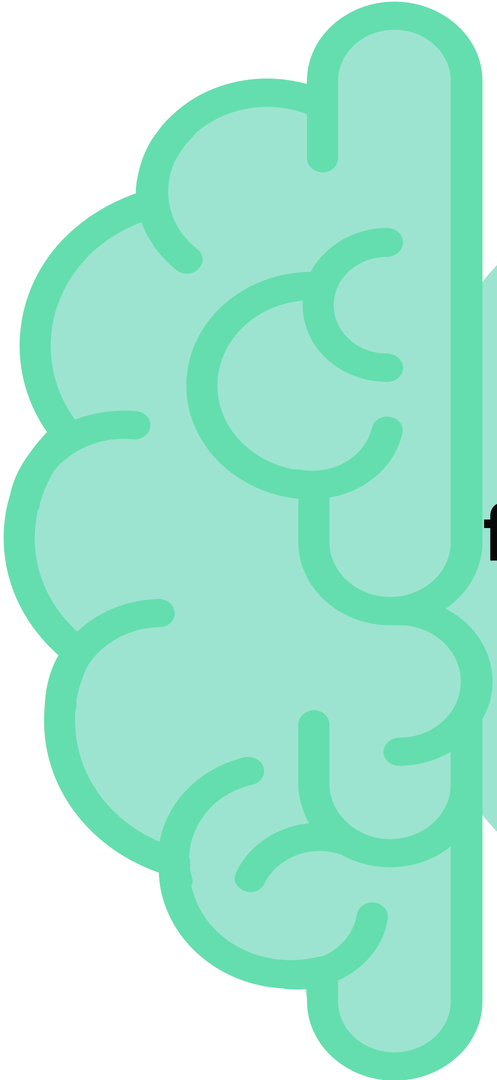
02 The accuracy of our models varies more across subjects the more complex they are, but allow for some of the highest test accuracies.

03 The encoding of the motivation state on the EEG signals is not similar enough across subjects for our models to obtain consistent performance.

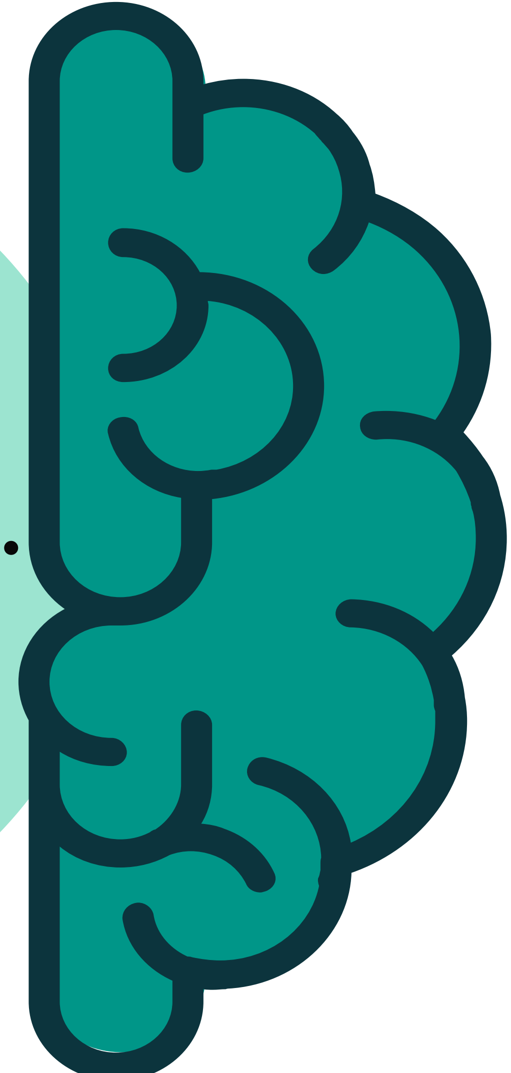
04 The self-adaptive modification we used offers no clear advantage over a fully connected graph representation.

Limitations & Future Work

- Lack of generality.
  - Node pooling layers readout functions.
- Class unbalance
  - Weighted losses.
- Inductive bias.
- Need for further proving of the models.
  - Frequency bands filtering.
  - Frequency domain.
  - Palliate the convergence problems.
  - Other self-adaptive strategies.



Thank you
for your attention.



References

The slide template is from **Slidesgo** and **Freepik**.

The CNN diagrams are from: NN-SVG: Publication-Ready Neural Network Architecture Schematics.