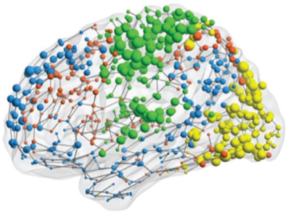
GCNs-Net: A Graph Convolutional Neural Network Approach for Decoding Time-resolved EEG Motor Imagery Signals

Shuyue Jia, Jan. 15th, 2023

Tasks

✓ Electroencephalogram (EEG) Tasks **Classification**

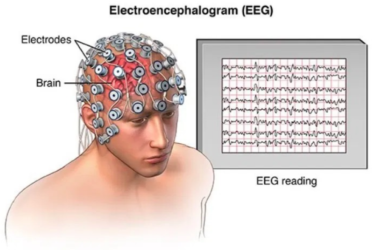
Control a wheelchair via EEG

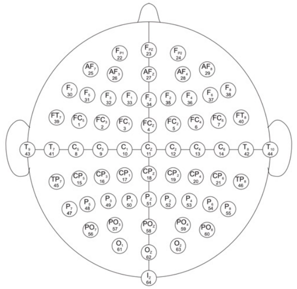


Functional Networks

mapping

interpret

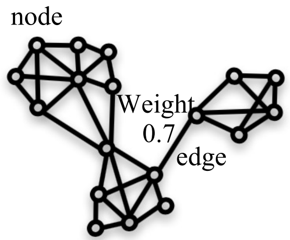




International 10-10 EEG System

topology

model



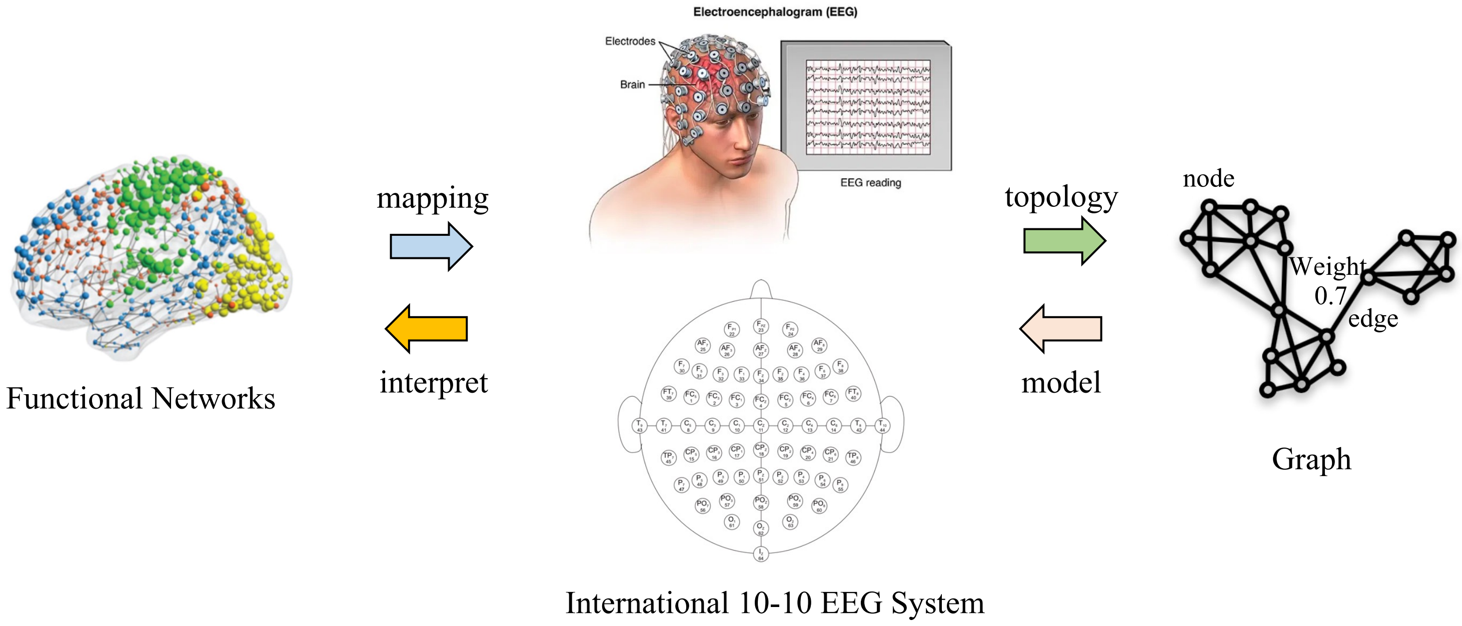
Graph

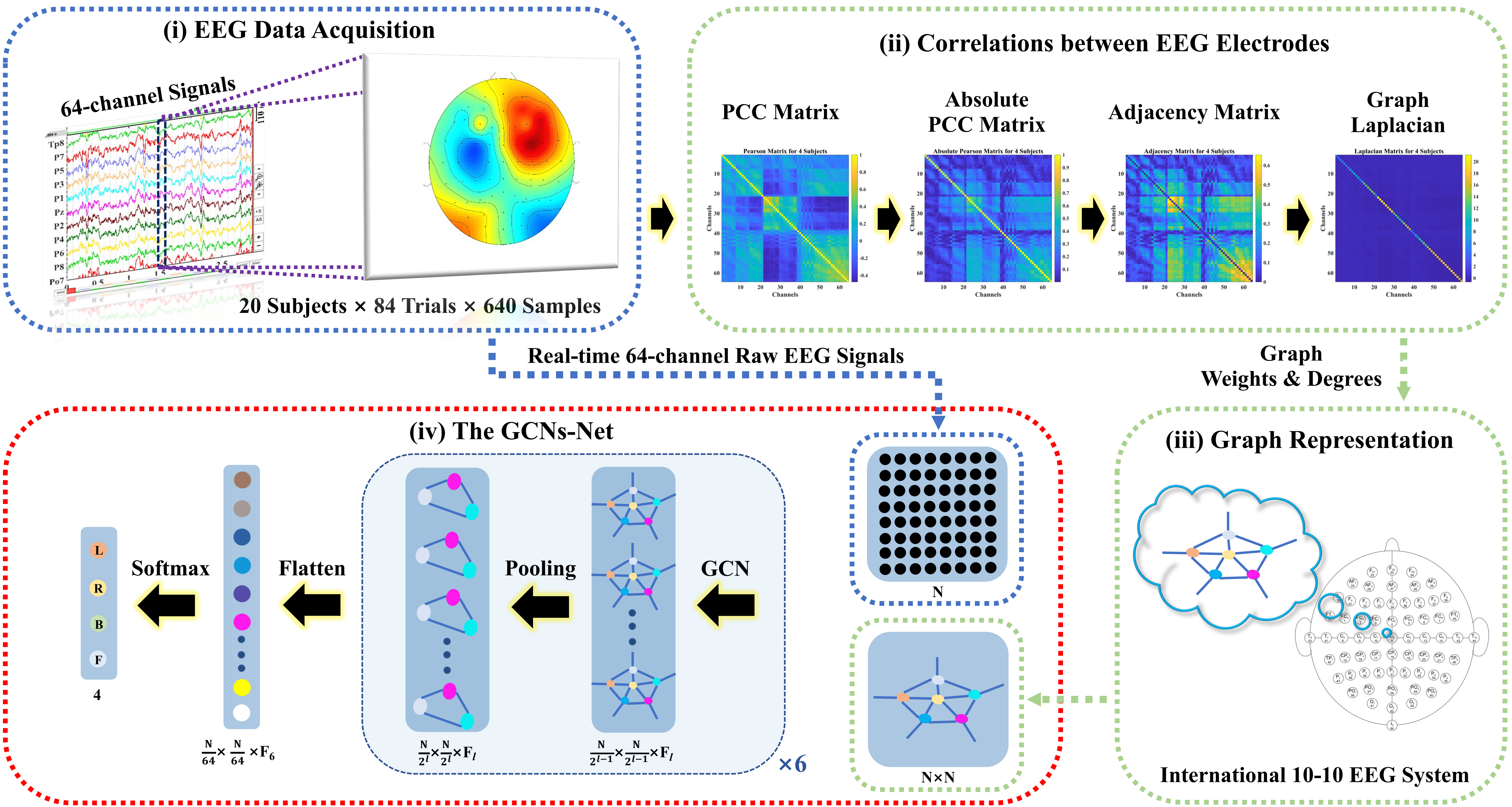
Interpret Functional Networks and better understand human brain

Image Credit: in the public domain.

EEG Research Novelty

- ✓ [**Motivation**] Graph Modeling for EEG Electrodes System
- ✓ [**Method**] Graph Representation Learning of EEG Signals





(i) EEG Data Acquisition
64-channel Signals
20 Subjects × 84 Trials × 640 Samples
(ii) Correlations between EEG Electrodes
PCC Matrix
Absolute PCC Matrix
Adjacency Matrix
Graph Laplacian
Pearson Matrix for 4 Subjects
Absolute Pearson Matrix for 4 Subjects
Adjacency Matrix for 4 Subjects
Laplacian Matrix for 4 Subjects
Channels
Real-time 64-channel Raw EEG Signals
Graph Weights & Degrees
(iii) Graph Representation
International 10-10 EEG System
(iv) The GCNs-Net
GCN
Pooling
Flatten
Softmax
L
R
B
F
4
N
N×N
×6

EEG Deep Learning

GCNs-Net: A Graph Convolutional Neural Network Approach for Decoding Time-Resolved EEG Motor Imagery Signals

Yimin Hou [1], Shuyue Jia [1, 2], Xiangmin Lun [1], Ziqian Hao [3], Yan Shi [1],
Yang Li [4], Rui Zeng [5], and Jinglei Lv [5] *

[1] School of Automation Engineering, Northeast Electric Power University
[2] Department of Computer Science, City University of Hong Kong
[3] Jinan Vocational College
[4] School of Electrical Engineering, Northeast Electric Power University
[5] School of Biomedical Engineering and Brain and Mind Center, The University of Sydney

EEG Deep Learning Library: https://github.com/SuperBruceJia/EEG-DL

Background

- **BCI**: establish connections between the brain and machines
  - (1) **Acquire and analyze brain signals** while conducting **actual** or **imagery** tasks
  - (2) **Control machines**
- **Significance**: help the disabled and understand the human brain
- **Types of BCI**:
  - Electroencephalography (EEG)
  - Magnetoencephalography (MEG)
  - Functional Magnetic Resonance Imaging (fMRI)
  - Invasive BCI Technologies (*e.g.*, Neuralink)
- **Reasons for using EEG for this project**:
  - Non-Invasiveness
  - High Temporal Resolution
  - Portability
  - Inexpensive Equipment

  } A potential market

- **Specific Task**: EEG Motor Imagery (*e.g.*, control a wheelchair via imagery-based EEG signals)
- **Our Research**: develop EEG-based BCI technologies to improve current stroke rehabilitation strategies

Image Credit: in the public domain.

Key Points in dealing with EEG time series

- **Individual Variability → Lower Classification Accuracy**
  - ✓ Low SNR
  - ✓ Different brain electrical conductivity ← different anatomical structure of brain
  - ✓ Electrodes' positional error

  Feature Extraction

  EEG Electrodes' Structure Modeling

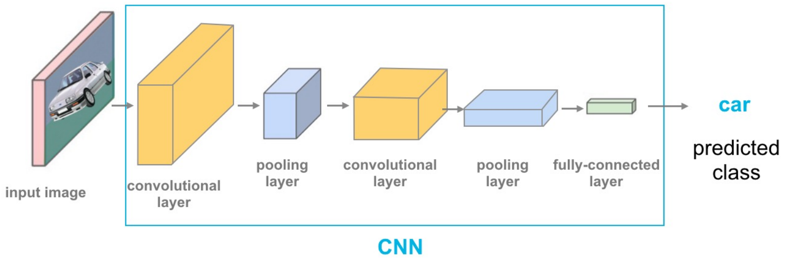
- **Slow Responding → Hard to develop Real-life Applications**
  - ✓ [**most literature**] Trial-level prediction (*e.g.*, 4 s)
  - ✓ Window/Slide-level prediction (*e.g.*, 0.4 s)
  - ✓ Time-resolved prediction (*e.g.*, 6.25 ms) (Our Work)

  Time-resolved or Window-based Signal Sampling

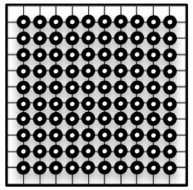
- **Lower Group-level Accuracy → Hard to develop Applications for a Group of People**
  - ✓ [**most literature**] Subject-level prediction (Our Work)
  - ✓ Group-level prediction (Our Work)

Motivation

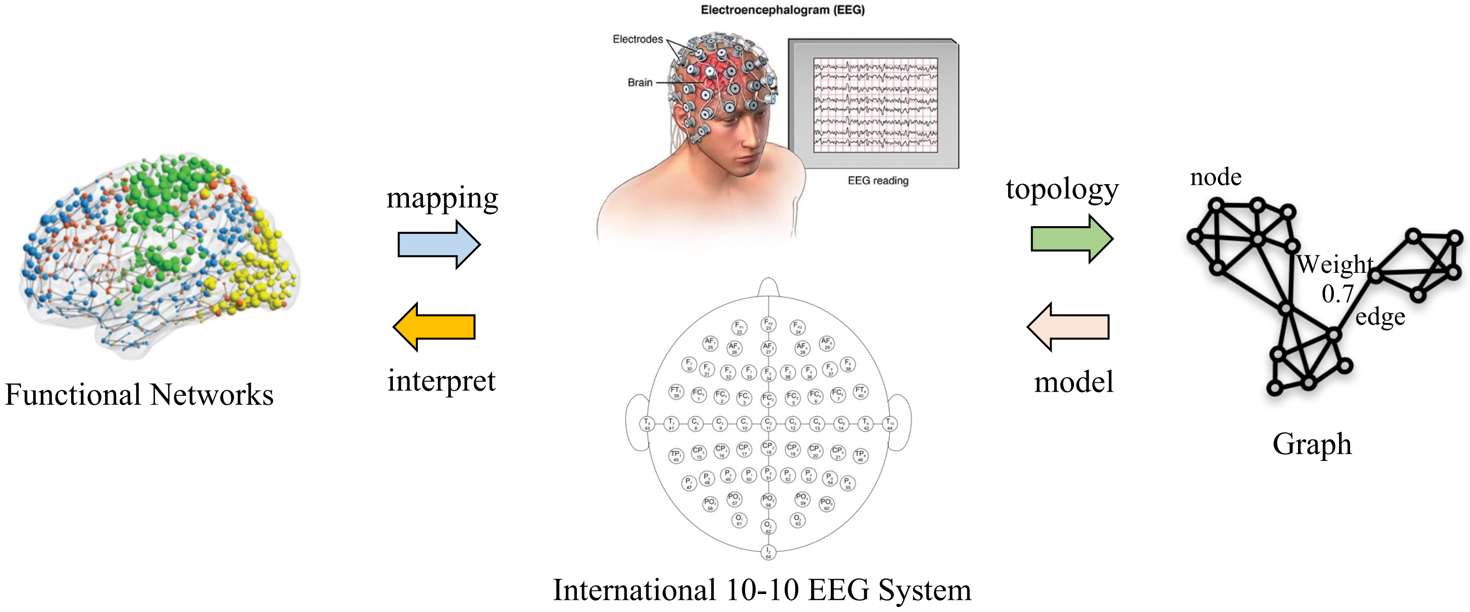
Convolutional Neural Networks:



- **Module: Convolution → Pooling → Fully-connected**
- **Modeling: Euclidean-Structured Data** (*e.g.*, Image, Speech, Natural Language)



- **Neuroscience** research has increasingly emphasized **Brain Network Dynamics**
  - Model **Functional Topological Connectivity** of EEG Electrodes → **Graph** (Non-Euclidean Structure)



Our Question

How to model the EEG System as a **Graph**?

How can we process EEG Signals via **Graph Representation Learning**?

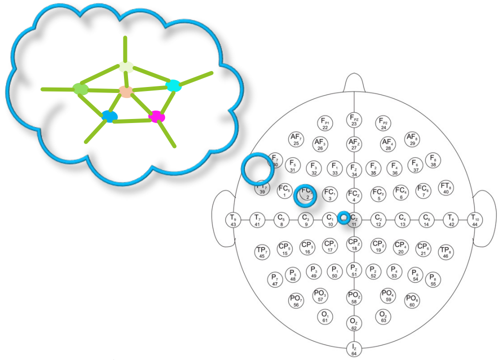
Image Credit: The PhysioNet Dataset and the Functional Network Image is in the public domain.

Can we directly apply convolutions on graphs?

- Traditional CNN **cannot** directly process graph signals
  - **Graph is irregular** (*i.e.*, unordered and vary in size)
  - Convolution **cannot** keep **Translation Invariance** on non-Euclidean signals
- **Graph Convolutional Neural Networks** (GCN)
  - Directly process **non-Euclidean graph-structured signals**
  - Consider **relational properties** (*e.g.*, correlations) between nodes

    → Model **Functional Topological Relationships** among EEG electrodes

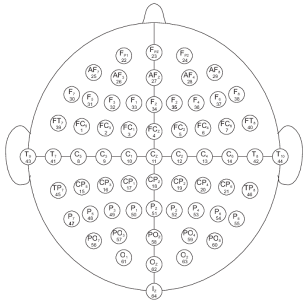
    → Analyze and interpret **Brain Network Dynamics**

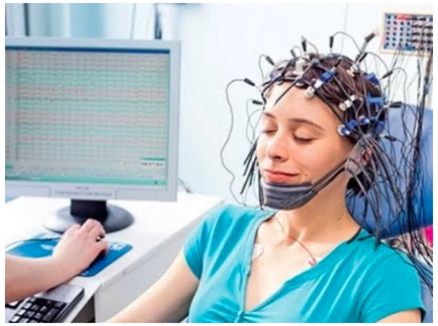


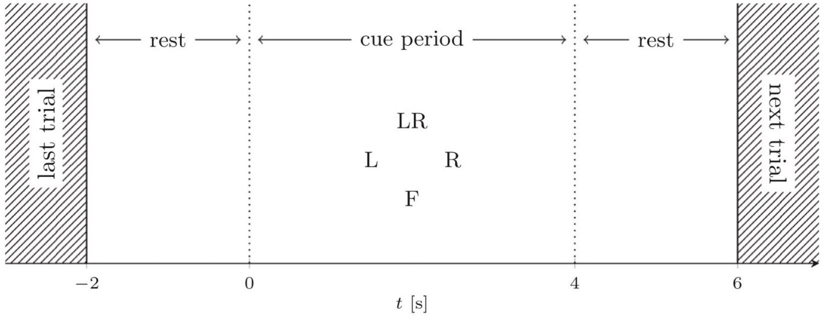
Benchmark Dataset

- The PhysioNet Dataset (EEG Motor Movement/Imagery Dataset)
- International 10-10 EEG System → **64 electrodes**

  (excluding electrodes Nz, F9, F10, FT9, FT10, A1, A2, TP9, TP10, P9, and P10)



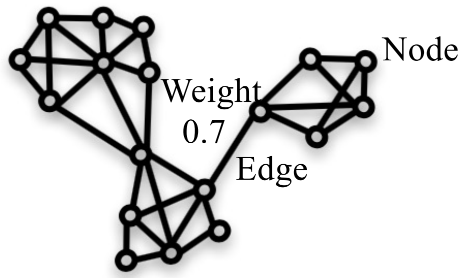




- **109 subjects (the largest number of participants in the field of EEG Motor Imagery)**
- Task: **4-class EEG Motor Imagery Classification**
  - ✓ Imagining (Task 1) left fist, (Task 2) right fist, (Task 3) both fists, (Task 4) both feet
- Each subject → 3 runs, 7 trials, 4 classes → 84 trials in total
- Each trial → 4 seconds experimental duration, 160 Hz Sampling Rate → 640 Time Points
- We apply the **Time-resolved Sampling Method**
  - ✓ Total samples per subject: 3 runs × 7 trials × 4 classes × 4 seconds × 160 Hz = 53,760 samples
  - ✓ Experimental Setting: 90% as the training set and the left 10% as the test set

Image Credit: The PhysioNet Dataset and the middle image is in the public domain.

Preliminary: Graph Representation



Definition: An Undirected and Weighted Graph with N nodes: $\mathbf{G} = \{\mathbf{V}, \mathbf{E}, \mathbf{A}\}$

- **V**: nodes (vertices), $|\mathbf{V}| = N$
- **E**: edges (links) that connect nodes
- **A**: weights (correlations) between nodes

Nodes Correlations: Pearson Matrix $\mathbf{P} \in \mathrm{R}^{N\times N}$ (denotes as PCC matrix)

- Measure the linear correlations between node **x** and node **y**
- μ is the mean, σ is the standard deviation, and $P_{x,y}$ is the Pearson Correlation Coefficient between node **x** and node **y**

$$P_{x,y} = \frac{\mathrm{E}((\mathbf{x} - \mu_x)(\mathbf{y} - \mu_y))}{\sigma_x \sigma_y}$$

- Absolute Pearson Matrix: $|\mathbf{P}| \in \mathrm{R}^{N\times N}$ and $|P_{ij}| \in [0, 1]$ → Note: In this work, we only consider **scale**.

Graph Weights: Adjacency Matrix $\mathbf{A} = |\mathbf{P}| - \mathbf{I} \in \mathrm{R}^{N\times N}$, where **I** is an Identity Matrix

Graph Degrees: Degree Matrix $\mathbf{D} \in \mathrm{R}^{N\times N}$

$$D_{ii} = \sum_{j=1}^{N} A_{ij}$$

Graph Representation: Combinatorial Laplacian $\mathbf{L} \in \mathrm{R}^{N\times N}$

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

Normalized:

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$$

Preliminary:

Spectral Theorem for Graph Laplacian **L**

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}}$$

$$\mathbf{L}\mathbf{U} = \boldsymbol{\Lambda}\mathbf{U}$$

– **U**: Fourier basis → **real** and **orthonormal** eigenvectors of **L**

– **Λ**: Fourier modes → the diagonal is the **ordered** and **real nonnegative** eigenvalues of **L**

Graph Fourier Transforms of Signal *f*

can be seen as the $e^{-j\omega t}$ in Fourier Transforms

$$\mathrm{F}[f(\boldsymbol{\lambda})] = \hat{f}(\boldsymbol{\lambda}) = \sum_{i=1}^{n} f(i) \times U(i)$$

$$\hat{f}(\boldsymbol{\lambda}) = \mathbf{U}^{\mathrm{T}} f \Leftrightarrow f = \mathbf{U}\hat{f}(\boldsymbol{\lambda})$$

$\hat{f}(\boldsymbol{\lambda})$ is the projection value of the Fourier basis **U**

Preliminary: Graph Convolution via Graph Fourier Transform

Notation:

Signal f

Signal h

F: Fourier Transforms

F^{-1}: Inverse Fourier Transforms

$\hat{f}(w)$: F(f)

$\hat{h}(w)$: F(h)

Note: Fourier Transforms of **Convolution in the spatial domain**

⇔

Point-wise Multiplication of two Fourier transformed signals

$$\mathrm{F}\left((f * h)_{\mathbf{G}}\right) = \hat{f}(w) \times \hat{h}(w)$$

Convolution $(f * h)_{\mathbf{G}} = \mathrm{F}^{-1}(\hat{f}(w) \times \hat{h}(w))$

$$\hat{f}(\boldsymbol{\lambda}) = \mathbf{U}^{\mathrm{T}} f$$

Hadamard Product (Element-wise Multiplication)

$$(f * h)_{\mathbf{G}} = \mathrm{F}^{-1}\left((\mathbf{U}^{\mathrm{T}} f) \odot (\mathbf{U}^{\mathrm{T}} h)\right)$$

$$f = \mathbf{U}\hat{f}(\boldsymbol{\lambda})$$

$$(f * h)_{\mathbf{G}} = \mathbf{U}\left((\mathbf{U}^{\mathrm{T}} f) \odot (\mathbf{U}^{\mathrm{T}} h)\right)$$

$$(f * h)_{\mathbf{G}} = \underset{[\mathrm{n} \times \mathrm{n}]}{\mathbf{U}} \; \underset{[\mathrm{n} \times \mathrm{n}]}{\mathrm{diag}[\hat{h}(\lambda_1), \hat{h}(\lambda_2), \ldots, \hat{h}(\lambda_{\mathrm{n}})]} \; \underset{[\mathrm{n} \times \mathrm{n}]}{\mathbf{U}^{\mathrm{T}}} \; \underset{[\mathrm{n} \times \mathrm{d}]}{f}$$

Source: https://en.wikipedia.org/wiki/Convolution_theorem

Graph Convolution

$$(f * h)_{\mathbf{G}} = \mathbf{U}\,\mathrm{diag}[\hat{h}(\lambda_1), \hat{h}(\lambda_2), \ldots, \hat{h}(\lambda_n)]\,\mathbf{U}^{\mathrm{T}} f$$

Activation Function

$$\boldsymbol{\mathcal{Y}} = \sigma(\mathbf{U}\mathbf{g}_{\boldsymbol{\theta}}\mathbf{U}^{\mathrm{T}}\boldsymbol{\chi})$$

$$\boldsymbol{\mathcal{Y}} = \sigma(\mathbf{U}\mathbf{g}_{\boldsymbol{\theta}}(\boldsymbol{\Lambda})\mathbf{U}^{\mathrm{T}}\boldsymbol{\chi})$$
$$\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$$

Approximate

K^{th} Polynomial Function

$$\mathbf{g}_{\boldsymbol{\theta}}(\boldsymbol{\Lambda}) = \sum_{k=0}^{K} \theta_k \boldsymbol{\Lambda}^k$$

$$\boldsymbol{\mathcal{Y}} = \sigma\left(\mathbf{U}\sum_{k=0}^{K} \theta_k \boldsymbol{\Lambda}^k\,\mathbf{U}^{\mathrm{T}}\boldsymbol{\chi}\right) = \sigma\left(\sum_{k=0}^{K} \theta_k\,(\mathbf{U}\boldsymbol{\Lambda}^k\mathbf{U}^{\mathrm{T}})\boldsymbol{\chi}\right) = \sigma\left(\sum_{k=0}^{K} \theta_k\,(\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\mathrm{T}})^k\boldsymbol{\chi}\right) = \sigma\left(\sum_{k=0}^{K} \theta_k\,\mathbf{L}^k\boldsymbol{\chi}\right)$$

$$\boldsymbol{\mathcal{Y}} = \sigma\left(\sum_{k=0}^{K} \theta_k\,\mathbf{L}^k\boldsymbol{\chi}\right)$$

Credit: Defferrard *et al.*, Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering, In NeurIPS 2016.

Graph Convolution

Convolution:
Weighted Sum

Weight Sharing

Node Aggregation
K is Filter Size

$$\mathcal{Y} = \sigma\left(\sum_{k=0}^{K} \theta_k \mathbf{L}^k \chi\right)$$

No need for Fourier Transform

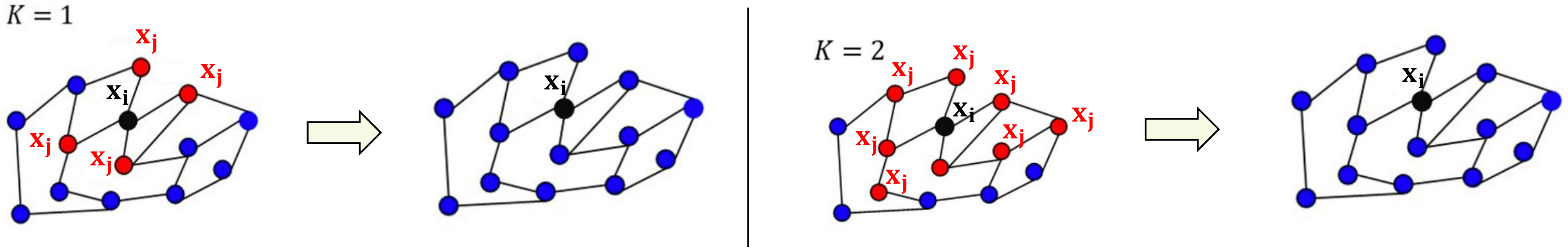
Beauty is in Simplicity

GCN Key Idea: Use "**edge information**" to **aggregate** "**node information**" to generate a new "**node representation**"

Laplace Operator
Local connectivity

$$\mathbf{x}_{\mathbf{new}} \leftarrow \mathbf{L}\mathbf{x}_{\mathbf{i}} = \sum_{j} A_{ij}(\mathbf{x}_{\mathbf{i}} - \mathbf{x}_{\mathbf{j}})$$

Localize in Space

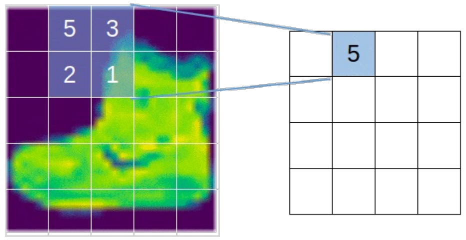


Pros:

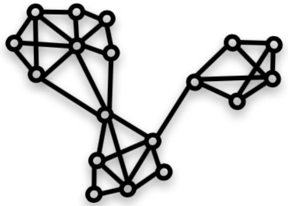
1. No need for Spectral Decomposition of $\mathbf{L}$
2. Less number of parameters (decrease model complexity) $\rightarrow K \ll N$

Cons: Need to compute $\mathbf{L}^k$

Pooling on Graphs (Graph Coarsening)



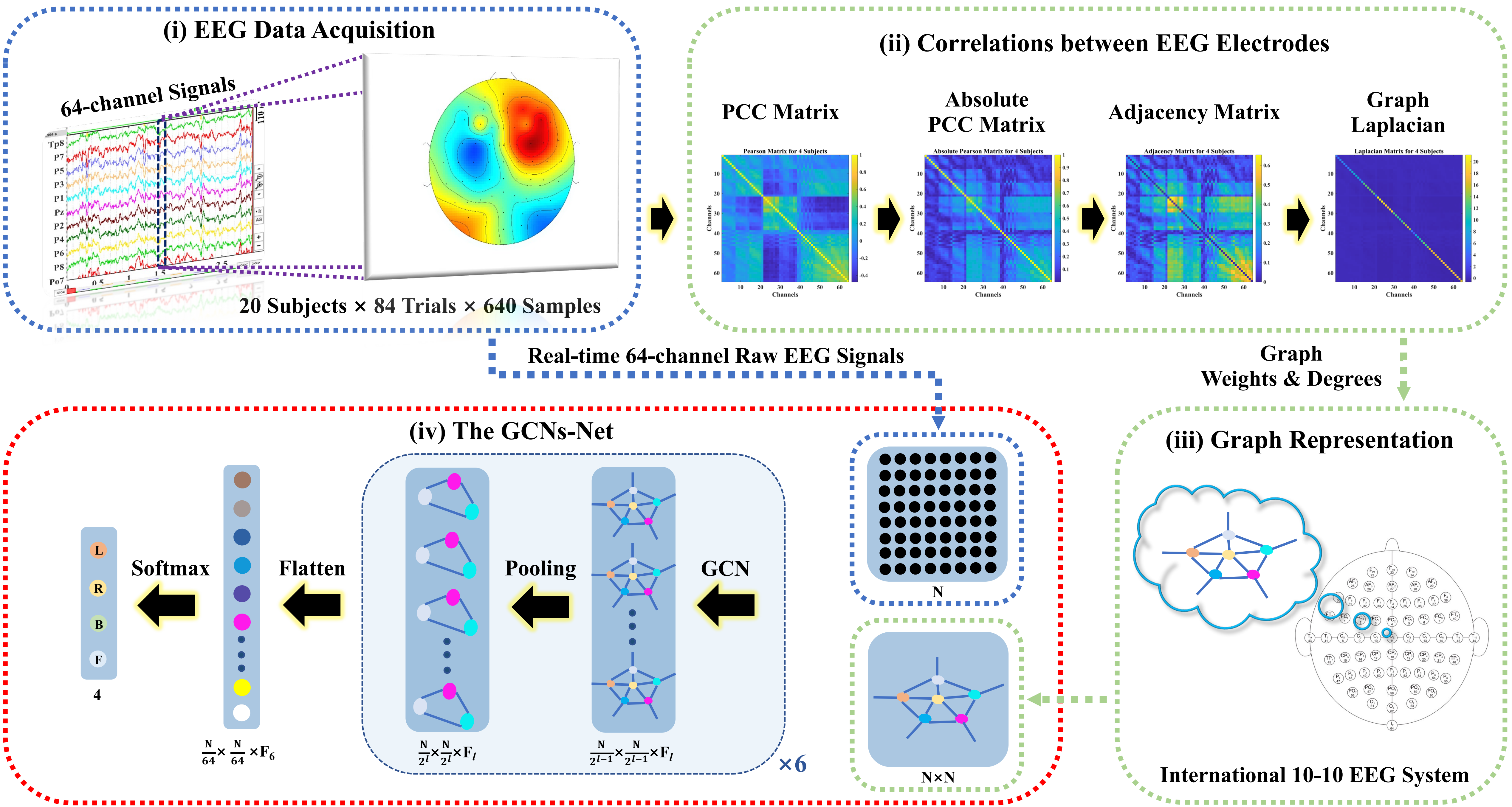
- Traditional CNN doesn't need to consider neighbors after convolutions
  - [**Euclidean Structure**] The output Feature Maps are "regular"
  - The neighbor is "meaningful"
- GCNs need to consider neighbors after convolutions
  - [**Non-Euclidean Structure**] The output **graphs' nodes are not arranged in any meaningful way**
  - Use **Graclus Multilevel Clustering Algorithm** to find "meaningful" neighbors
  - Minimize the ***Local Normalized Cut*** (a cluster grouping method)



$$-W_{ij}\left(\frac{1}{d_i}+\frac{1}{d_j}\right)$$

  - i and j denote node i and node j
  - W_{ij} is the **learned weight** between node i and node j

Image Credit: in the public domain.



(i) EEG Data Acquisition
64-channel Signals
20 Subjects × 84 Trials × 640 Samples
(ii) Correlations between EEG Electrodes
PCC Matrix
Absolute PCC Matrix
Adjacency Matrix
Graph Laplacian
Pearson Matrix for 4 Subjects
Absolute Pearson Matrix for 4 Subjects
Adjacency Matrix for 4 Subjects
Laplacian Matrix for 4 Subjects
Channels
Real-time 64-channel Raw EEG Signals
Graph Weights & Degrees
(iii) Graph Representation
International 10-10 EEG System
(iv) The GCNs-Net
GCN
Pooling
Flatten
Softmax
L
R
B
F
4
N
N×N
×6

Correlation among EEG electrodes
Two Subjects: Subject 10 and 5

Problem: **Individual Variability**

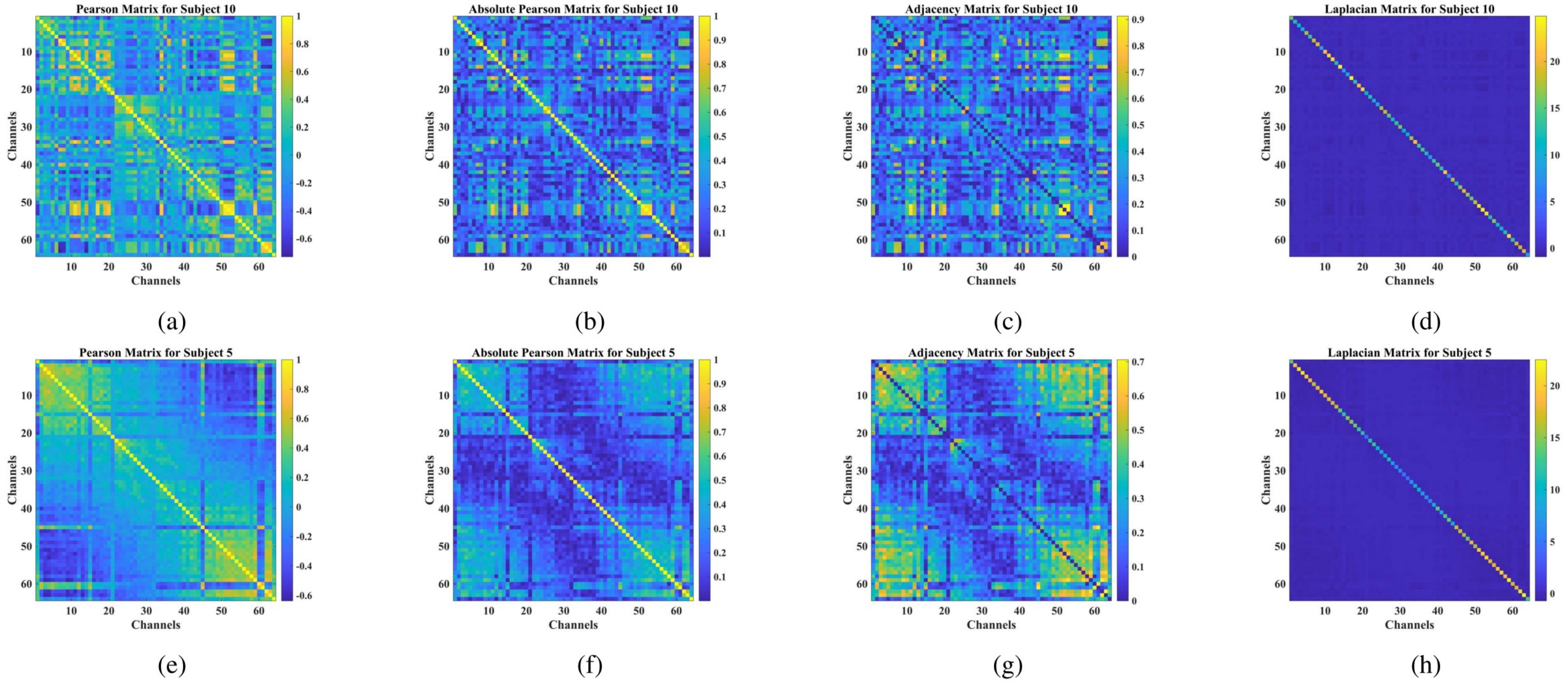


Fig. 6. PCC matrix, absolute PCC matrix, adjacency matrix, and graph Laplacian for Subjects 10 and 5 from the PhysioNet dataset. (a) PCC matrix for Subject 10. (b) Absolute PCC matrix for Subject 10. (c) Adjacency matrix for Subject 10. (d) Graph Laplacian for Subject 10. (e) PCC matrix for Subject 5. (f) Absolute PCC matrix for Subject 5. (g) Adjacency matrix for Subject 5. (h) Graph Laplacian for Subject 5.

Correlation among EEG electrodes
20 Subjects and 100 Subjects

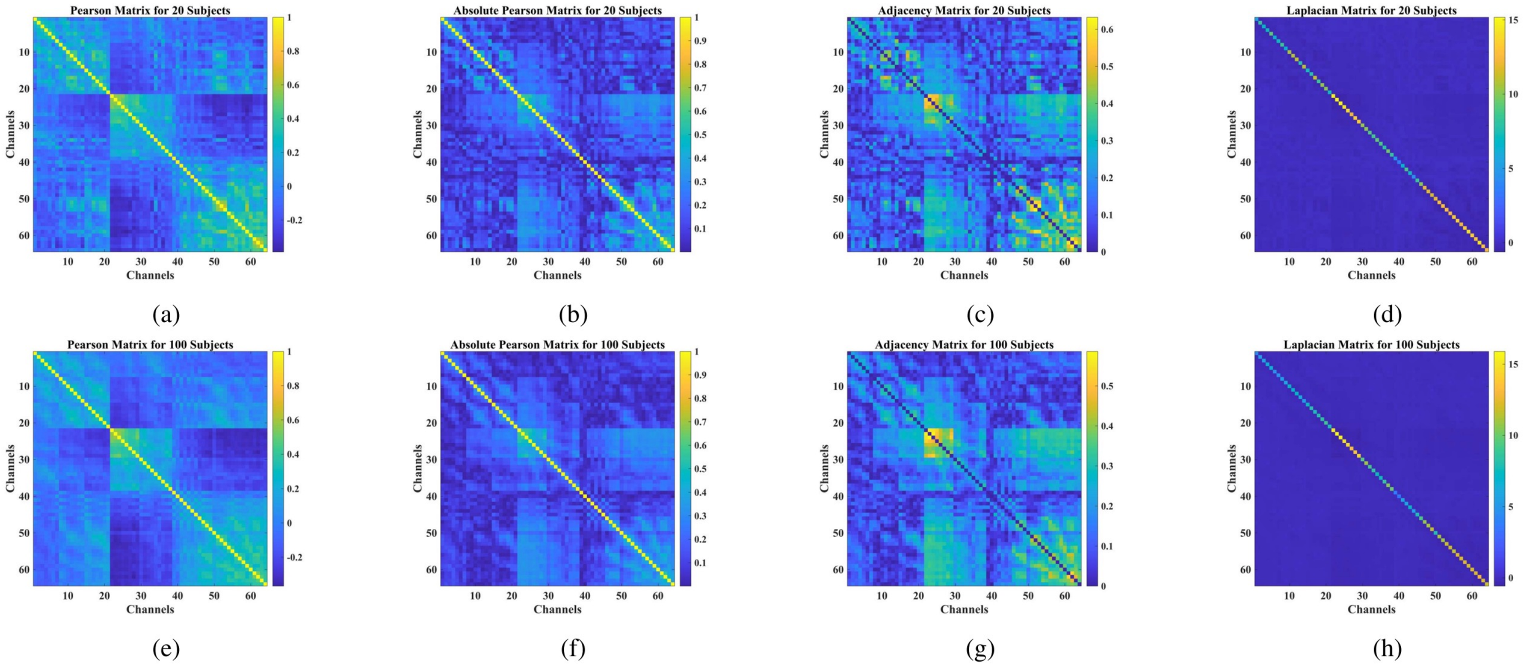


Fig. 2. PCC matrix, absolute PCC matrix, adjacency matrix, and graph Laplacian for 20 and 100 subjects, respectively, from the PhysioNet dataset. (a) PCC matrix for 20 subjects. (b) Absolute PCC matrix for 20 subjects. (c) Adjacency matrix for 20 subjects. (d) Graph Laplacian for 20 subjects. (e) PCC matrix for 100 subjects. (f) Absolute PCC matrix for 100 subjects. (g) Adjacency matrix for 100 subjects. (h) Graph Laplacian for 100 subjects.

Increasing ***the number of subjects*** alleviates ***individual variability***

Model Design for 64-electrode EEG System

TABLE I

IMPLEMENTATION DETAILS OF THE PROPOSED GCNS-NET ON THE PHYSIONET DATASET

Layer	Type	Maps	Size	Edges	Polynomial Order	Pooling Size	Activation	Weights	Bias
Softmax	Fully-connected	–	O	–	–	–	Softmax	$\frac{N}{64} \times \frac{N}{64} \times F_6 \times O$	O
Flatten	Flatten	–	$\frac{N}{64} \times \frac{N}{64} \times F_6$	–	–	–	–	–	–
P6	Max-pooling	F_6	$\frac{N}{32}$	$\sum_{i=1}^{\frac{N}{32}-1} i$	–	2	–	–	–
C6	Convolution	F_6	$\frac{N}{32}$	$\sum_{i=1}^{\frac{N}{32}-1} i$	K	–	Softplus	$F_5 \times F_6 \times K$	$\frac{N}{32} \times F_6$
P5	Max-pooling	F_5	$\frac{N}{16}$	$\sum_{i=1}^{\frac{N}{16}-1} i$	–	2	–	–	–
C5	Convolution	F_5	$\frac{N}{16}$	$\sum_{i=1}^{\frac{N}{16}-1} i$	K	–	Softplus	$F_4 \times F_5 \times K$	$\frac{N}{16} \times F_5$
P4	Max-pooling	F_4	$\frac{N}{8}$	$\sum_{i=1}^{\frac{N}{8}-1} i$	–	2	–	–	–
C4	Convolution	F_4	$\frac{N}{8}$	$\sum_{i=1}^{\frac{N}{8}-1} i$	K	–	Softplus	$F_3 \times F_4 \times K$	$\frac{N}{8} \times F_4$
P3	Max-pooling	F_3	$\frac{N}{4}$	$\sum_{i=1}^{\frac{N}{4}-1} i$	–	2	–	–	–
C3	Convolution	F_3	$\frac{N}{4}$	$\sum_{i=1}^{\frac{N}{4}-1} i$	K	–	Softplus	$F_2 \times F_3 \times K$	$\frac{N}{4} \times F_3$
P2	Max-pooling	F_2	$\frac{N}{2}$	$\sum_{i=1}^{\frac{N}{2}-1} i$	–	2	–	–	–
C2	Convolution	F_2	$\frac{N}{2}$	$\sum_{i=1}^{\frac{N}{2}-1} i$	K	–	Softplus	$F_1 \times F_2 \times K$	$\frac{N}{2} \times F_2$
P1	Max-pooling	F_1	N	$\sum_{i=1}^{N-1} i$	–	2	–	–	–
C1	Convolution	F_1	N	$\sum_{i=1}^{N-1} i$	K	–	Softplus	$1 \times F_1 \times K$	$N \times F_1$
Input	Input	1	N	$\sum_{i=1}^{N-1} i$	–	–	–	–	–

Model Optimization

- **Ablation Study**: **Optimal Model Structure** (64-electrode EEG System)
  - C6-P6-K2 with [16, 32, 64, 128, 256, 512] filters
- **Gradient Iterative Solver**: **Adam Optimizer** with **Stochastic Gradient Descent** (SGD) algorithm
  - Learning Rate: 0.01
  - Batch Size: 1,024
- **Activation Function**: **Softplus** (Smooth Rectified Linear Unit)

$$F(\mathbf{x}) = \log(1 + e^{\mathbf{x}})$$

- **Model Output**: **Softmax**: **y** are labels, $\hat{\mathbf{y}}$ are the final output tasks

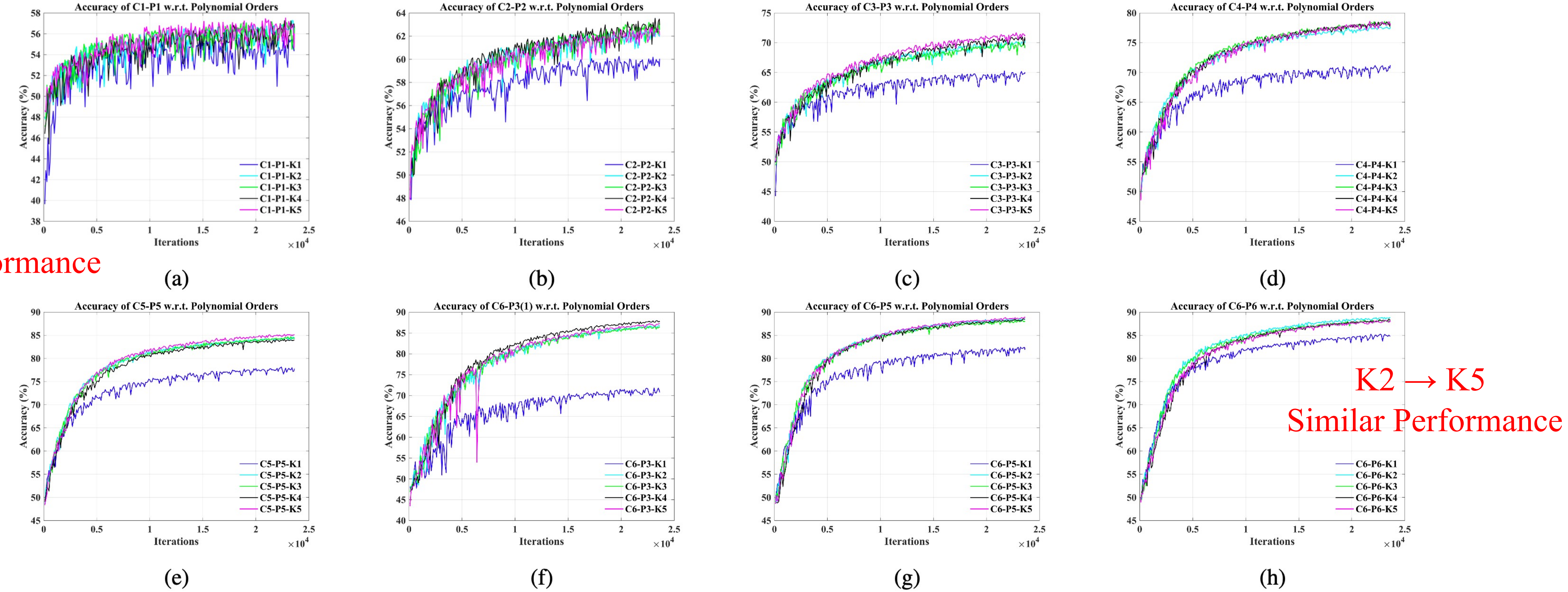
$$\hat{y}_i = \operatorname{argmax}\left(\frac{e^{y_i}}{\sum_{i=1}^{4} e^{y_i}}\right)$$

- **Loss Function**: **Cross-entropy Loss** with **L2 regularization**

$$\text{Loss} = -\sum_{i=1}^{4} y_i \log(\hat{y}_i) + \lambda\left(\sum_{j=1}^{n} w_j^2 + b_j^2\right)$$

$\lambda = 1 \times 10^{-6}$ is the coefficient of the L2 regularization.

Ablation Study



K1
Poor Performance

Fig. 3. Accuracy of some selected models regarding different polynomial approximation order. The models are selected from Table II. (a) Accuracy of the model C1-P1 (model 1). (b) Accuracy of the model C2-P2 (model 3). (c) Accuracy of the model C3-P3 (model 6). (d) Accuracy of the model C4-P4 (model 10). (e) Accuracy of the model C5-P5 (model 14). (f) Accuracy of the model C6-P3 (model 16). (g) Accuracy of the model C6-P5 (model 19). (h) Accuracy of the model C6-P6 (model 20).

Experimental Results
Groupwise Prediction and Subject-specific Adaptation

TABLE IV
PERFORMANCE COMPARISONS ON THE PHYSIONET DATASET

Related Work	Max. Accuracy	Avg. Accuracy	*p*-value	Level	Approach	Num. of Subjects
Dose *et al*. (2018) [22]	–	58.58%	–	Group	CNNs	105
	80.38%	68.51%	< 0.05	Subject		1
Ma *et al*. (2018) [60]	82.65%	68.20%	–	Group	RNNs	12
Hou *et al*. (2020) [20]	94.50%	–	–	Group	ESI-CNNs	10
	96.00%	–	> 0.05	Subject		1
Hou *et al*. (2022) [34]	94.64%	–	–	Group	BiLSTM-GCN	20
	98.81%	95.48%	> 0.05	Subject		1
Jia *et al*. (2022) [40]	94.16%	93.78%	–	Group	Graph ResNet	20
	98.08%	94.18%	> 0.05	Subject		1
Author	**89.39%**	**88.57%**	–	**Group**	**GCNs-Net**	**20**
	88.14%	–				**100**
	98.72%	**93.06%**		**Subject**		**1**

Note: ***p*-value < 0.05** → Statistically Significant Difference

Takeaways and Future Work

✓ **Graph Representation**

Graph Representation Learning to deeply extract **Network Patterns of Brain Dynamics** for EEG classification.

✓ **Model Converge**

Converge for both **Personalized and Groupwise Predictions**, indicating that the GCNs-Net is able to build a generalized representation of EEG time-series **against both Personalized and Groupwise Variations**.

✓ **Future Work**

Model EEG signals as **Dynamic Graphs** and process them via **Dynamic Graph Representation Learning**.

Thank you!

Any question?

Acknowledgment: Dr. Zhijian Hou and Dr. Dingquan Li.