electronics

Article

MST-DGCN: A Multi-Scale Spatio-Temporal and Dynamic Graph Convolution Fusion Network for Electroencephalogram Recognition of Motor Imagery

Yuanling Chen [1], Peisen Liu [2] and Duan Li [2,*]

[1] Engineering Training Center, Zhengzhou University of Light Industry, Zhengzhou 450001, China; 1996031@zzuli.edu.cn
[2] School of Electronic Information, Zhengzhou University of Light Industry, Zhengzhou 450001, China; 332207050668@email.zzuli.edu.cn
* Correspondence: liduan@hpu.edu.cn

Abstract: The motor imagery brain-computer interface (MI-BCI) has the ability to use electroencephalogram (EEG) signals to control and communicate with external devices. By leveraging the unique characteristics of task-related brain signals, this system facilitates enhanced communication with these devices. Such capabilities hold significant potential for advancing rehabilitation and the development of assistive technologies. In recent years, deep learning has received considerable attention in the MI-BCI field due to its powerful feature extraction and classification capabilities. However, two factors significantly impact the performance of deep-learning models. The size of the EEG datasets influences how effectively these models can learn. Similarly, the ability of classification models to extract features directly affects their accuracy in recognizing patterns. In this paper, we propose a Multi-Scale Spatio-Temporal and Dynamic Graph Convolution Fusion Network (MST-DGCN) to address these issues. In the data-preprocessing stage, we employ two strategies, data augmentation and transfer learning, to alleviate the problem of an insufficient data volume in deep learning. By using multi-scale convolution, spatial attention mechanisms, and dynamic graph neural networks, our model effectively extracts discriminative features. The MST-DGCN mainly consists of three parts: the multi-scale spatio-temporal module, which extracts multi-scale information and refines spatial attention; the dynamic graph convolution module, which extracts key connectivity information; and the classification module. We conduct experiments on real EEG datasets and achieve an accuracy of 77.89% and a Kappa value of 0.7052, demonstrating the effectiveness of the MST-DGCN in MI-BCI tasks. Our research provides new ideas and methods for the further development of MI-BCI systems.

Keywords: motor imagery (MI); electroencephalogram (EEG); brain-computer interface (BCI); multi-scale convolution; spatial attention; dynamical graph convolution; data augmentation

check for updates

Citation: Chen, Y.; Liu, P.; Li, D. MST-DGCN: A Multi-Scale Spatio-Temporal and Dynamic Graph Convolution Fusion Network for Electroencephalogram Recognition of Motor Imagery. *Electronics* **2024**, *13*, 2174. https://doi.org/10.3390/electronics13112174

Academic Editor: Hyun Jae Baek

Received: 4 May 2024
Revised: 27 May 2024
Accepted: 31 May 2024
Published: 3 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Brain-computer interfaces (BCIs) enable humans to interact directly with computers or other devices without the need for muscle activity, thereby achieving control and communication with the external world. They can restore the behavioral abilities of disabled individuals and enhance the capabilities of non-disabled individuals [1–3]. For populations such as those with physical disabilities, EEG-based motor imagery (MI) paradigms in BCIs are an effective means of restoring mobility [4,5], for example, in brain-controlled wheelchairs, brain-controlled exoskeletons, and so on [6–11]. For these populations, as long as the relevant areas of their brains function normally, they can generate task-related MI signals through training to restore mobility. The overall process of MI-BCI is illustrated in Figure 1.

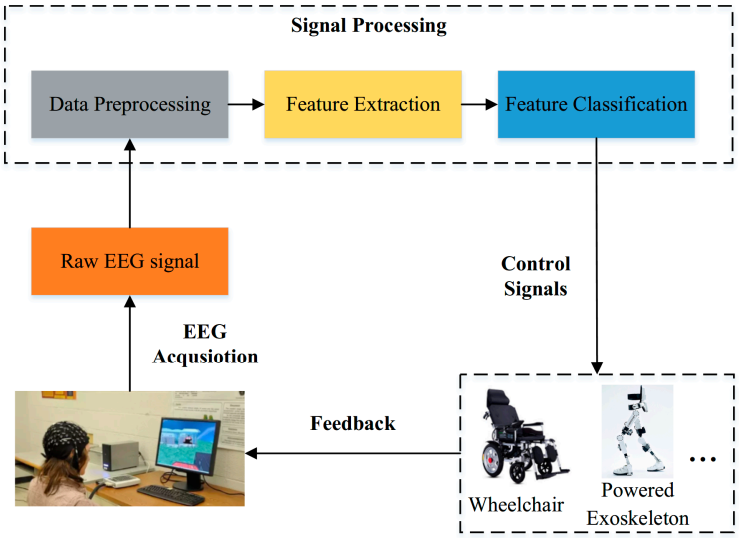

Figure 1. The overall process of an EEG-based MI-BCI system, including the EEG signal acquisition, signal processing, issuing control signals, and feedback.

The EEG-based MI-BCI system mainly consists of four processes: EEG signal acquisition, signal processing, issuing control signals, and feedback. In the EEG signal acquisition step, subjects perform different motor imagery tasks, and the neural activity is recorded by electrodes directly in contact with the scalp. After amplification and digitization, the raw EEG signal is obtained. In the signal-processing stage, the raw EEG signal undergoes data preprocessing, such as filtering and truncation, to remove any task-irrelevant signals. Then, feature extraction methods are applied to extract features from the preprocessed signal, which are then input into a classifier for classification, yielding the classification results. In the phase of issuing control signals, the computer sends corresponding control signals to the device based on the classification results. Finally, the real-time feedback of the control situation generated by the device is provided to the subject.

Early MI-BCI research primarily focused on feature extraction and classification. The main steps included designing effective feature extraction algorithms, such as CSP [12], FBCSP [13], and CCSP [14], to extract features (often accompanied by dimensionality reduction) from raw data. These features were then fed into machine-learning classifiers (such as LDA, SVM, etc.) to obtain the prediction results. However, this approach had two drawbacks. Firstly, the performance highly depended on the expertise of the researchers during these two stages. Suboptimal feature extraction methods or machine-learning classifiers may not fully capitalize on the extracted features, potentially leading to poorer results. Secondly, there were inherent differences among subjects (e.g., gender, age, cognitive abilities, physical health, etc.). When different subjects performed the same MI task, there were differences in the spatial distribution of the activated brain areas, signal amplitudes, and frequency components. However, both feature extraction methods and machine-learning classifiers exhibit limitations in accommodating individual variability. These techniques often fail to effectively adjust when processing data from multiple subjects at the same time [15].

In recent years, deep-learning methods have been applied to the classification of EEG signals and have shown promising results. Deep learning's key advantage is its ability to integrate feature extraction and classification into a single end-to-end model. By designing and training deep neural networks and feeding preprocessed raw EEG signals into them, classification results can be obtained. Moreover, when provided with adequate data, deep neural networks effectively learn individual features during training. This learning process helps overcome the challenges of handling individual differences, thus ensuring robust performance across multiple subjects simultaneously. However, the complexity of neural networks in deep learning should be considered carefully. Stacking

multiple convolutional and fully connected layers can significantly increase the model complexity and extend the training times. For example, Schirrmeister et al. [16] designed a deep CNN model with multiple convolutional layers, pooling layers, and fully connected layers, which required considerably more computational resources and longer periods of training compared to simpler models. To address this issue, Lawhern et al. [17] designed a lightweight network called EEGNet, which utilizes one-dimensional convolutional layers and depthwise separable convolutional layers to reduce the model parameters while extracting the signal features as much as possible. Subsequent research has extended EEGNet, such as Incep-EEGNet [18] and MI-EEGNet [19]. Another limitation of deep learning is its reliance on large datasets for the training of deep neural networks. This requirement is manageable in fields such as image processing [20,21] and natural language processing [22,23], where extensive data are typically available. However, in the context of MI-BCI, obtaining a dataset comparable in size to CIFAR-10 [24] or Fashion-MNIST [25] is nearly impossible. This difficulty arises from the fatigue associated with multiple tasks and the inherent instability and variability of EEG data. Therefore, the network itself needs to have strong feature extraction capabilities. Some studies [26,27] have employed attention mechanisms to extract sufficiently discriminative features, but they have not addressed the issue of the small data volume. Other studies [28,29] have attempted to expand the original dataset using data augmentation methods, yielding promising results.

Network neuroscience transforms collected EEG signals into graph representations, providing an innovative approach to EEG modeling. This method facilitates the analysis of brain network properties, including the network features, node significance, and information flow. Such analyses yield valuable insights into the brain's specific state, as documented in references [30–32]. Typically, in graph-based models, the electrodes serve as the nodes, with the number of electrodes used during data collection directly determining the number of graph vertices. The edges are commonly established based on the functional or structural connections between these nodes [33,34]. Graph neural networks (GNNs) extend the concept of convolution from Euclidean data to graph-structured data, offering promising solutions for the BCI field [35–37]. Unlike traditional CNNs or recurrent neural networks (RNNs), the non-Euclidean structure of the input data (graphs) endows GNNs with the ability to extract spatially dependent features. However, GNNs themselves require careful design, as the number of layers, node composition, edge composition, adjacency matrices, and other input data and parameters need to be adjusted; otherwise, the performance of GNNs may be affected. Additionally, similar to CNNs and RNNs, GNNs also require a large amount of data for training.

To address the above issues, we propose the MST-DGCN, an end-to-end MI-EEG classification network. As illustrated in Figure 2, the proposed MST-DGCN combines a CNN with a GNN, where information extracted from different scales is fed into a spatial attention module to obtain refined features. These features are then used to construct a graph, which is input into the GNN. Our model, the MST-DGCN, integrates spatial attention mechanisms and graph neural networks to utilize both global and local information effectively. This integrated approach enhances the exploitation of latent knowledge within the data while minimizing the negative impact of raw data variability on the GNN's performance. To address the issue of limited training data in deep learning, the original dataset is augmented to ensure that the network is adequately trained. Through extensive experiments and evaluations, we validate the effectiveness of the proposed method in MI-EEG classification tasks. The main contributions of this work can be summarized as follows:

- We proposed the MST-DGCN model for MI-EEG classification, which incorporates a multi-scale spatio-temporal module based on a CNN and a dynamic graph convolution module based on a GNN. This model extracts effective features relevant to the current task from multiple temporal scales and spatial structures.
- Based on [29,38], by combining the designed loss functions and various data augmentation techniques with the MST-DGCN, we successfully improved the train-

ing effectiveness of the network, resulting in more robust and accurate MI-EEG classification results.

- We conducted extensive experiments on real EEG datasets (Dataset IIa from the BCI Competition IV) and compared the results with multiple traditional classical methods and state-of-the-art approaches. Additionally, we designed ablation experiments to validate the contributions and effectiveness of each component of the model.

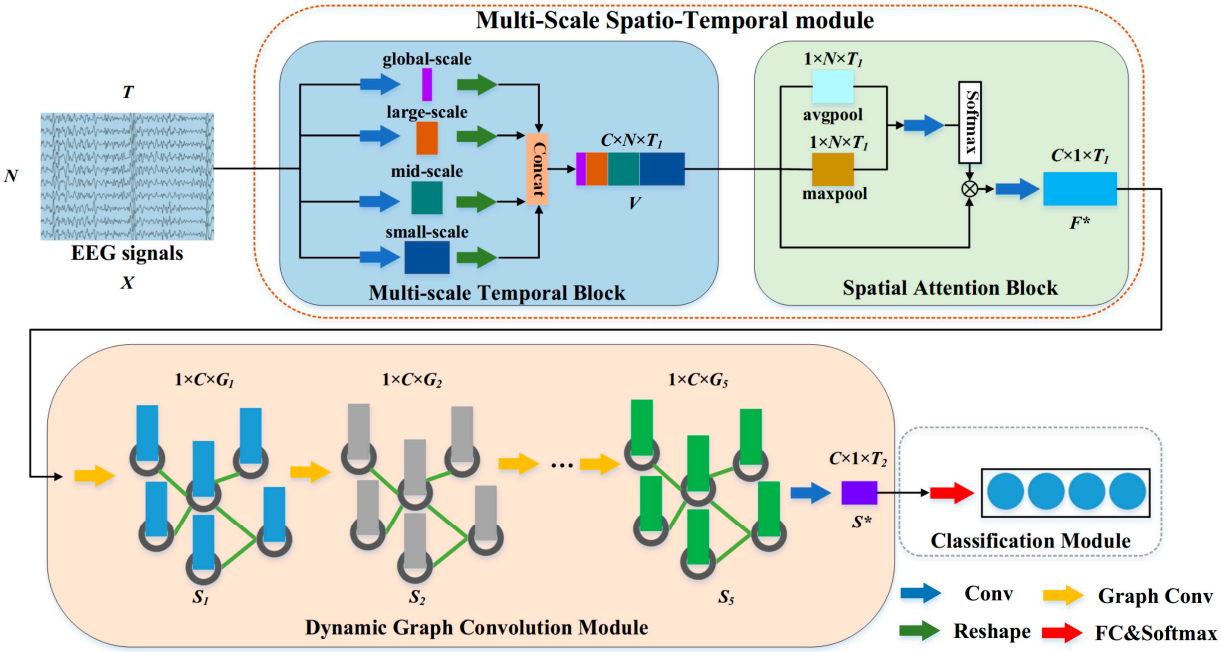

Figure 2. Overall architecture of the MST-DGCN.

The remainder of this paper is organized as follows. Section 2 provides a detailed description of the proposed MST-DGCN. Section 3 presents the experiments and results, including the data augmentation methods, training strategies, datasets, evaluation metrics, and experimental results. Finally, a discussion is offered in Section 4, followed by conclusions being drawn in Section 5.

2. Methods

In this section, we first describe the symbols and definitions used in this paper. Then, we provide a detailed introduction to the proposed MST-DGCN (as shown in Figure 2). Finally, we describe the implementation details of each module in the MST-DGCN.

2.1. Notions and Definitions

Each subject's EEG data are formatted as a set $S = \{(X_i, y_i), i = 1, \ldots, M\}$. Here, X_i represents the i-th EEG data as a two-dimensional matrix in $R^{N \times T}$, where N is the number of electrodes and T is the number of sampling time points. Each y_i corresponds to X_i and belongs to one of L categories, denoting the label. For instance, in the dataset used in this study, we have categories such as Y = {y_1 = "left hand", y_2 = "right hand", y_3 = "feet", y_4 = "mouth"}. The variable M denotes the total number of EEG records for each subject, consistent across all the subjects.

2.2. Multi-Scale Spatio-Temporal and Dynamic Graph Convolution Fusion Network (MST-DGCN)

In the MST-DGCN, three modules are designed: the multi-scale spatio-temporal module, the dynamic graph convolution module, and the classification module. In the multi-scale spatio-temporal module, different scales of temporal information in the signal are obtained using multi-scale temporal blocks. Then, this information is fed into a spatial attention block to extract the most valuable segments in terms of time and space. In the

dynamic graph convolution module, information processed by the previous convolution layers is used as the input for the graph. This input allows the module to capture the dynamic spatial dependencies among the nodes effectively. Finally, the refined spatio-temporal features from the dynamic graph neural network are input into the classification module to obtain the final classification results for the motor imagery.

2.2.1. Multi-Scale Spatio-Temporal Module

In motor imagery recognition based on EEG recordings, the collected signals generally have high temporal resolution, which is a characteristic inherent of EEG. Typically, EEG data for motor imagery have sampling frequencies ranging from 128 Hz to 1000 Hz. For each input sample $X \in R^{N\times T}$, it can be viewed as $[x_1, x_2, \ldots, x_N]$, a matrix composed of vector data collected from N channels, where any data of the i-th channel are denoted as x_i, with N = 22 and T = 500. Motor imagery generates specific phenomena known as event-related synchronization (ERS) and event-related desynchronization (ERD). These phenomena occur abruptly, typically within a short timeframe of 2 to 5 s during the overall recording period. This brief duration allows for the possibility of multiple occurrences within a single session. To capture global and local changes, the signal is reshaped into $1 \times N \times T$ (the input of the neural network is $c \times h \times w$) and passed through four different scales of convolutional kernels, including global-scale, large-scale, mid-scale, and small-scale, corresponding to extracting four different scales of features F_1, F_2, F_3, F_4. The corresponding transformations are as follows:

$$\mathcal{F}_i : X \rightarrow F_i \in R^{C\times N\times t_i},\ i = 1, \ldots, 4 \tag{1}$$

where $\mathcal{F}_i$ represents the transformation, F_i is the transformed feature, C denotes the number of channels after convolution (the term "channel" here refers to a concept in images, distinct from electrodes, which are physical entities in EEG caps that make contact with the scalp during EEG data acquisition), and t_i represents the length of the time points after convolution with different scale kernels. Inspired by Lawhern et al. [17] and Wu et al. [39], we set the sizes of the four convolution kernels used here to (64, 1), (40, 1), (26, 1), and (16, 1), with a stride set to one-eighth of the kernel length. Thus, after convolution, $t_1 = 55$,$t_2 = 93$,$t_3 = 159$,$t_4 = 243$. These are all 1D convolutional kernels that convolve the data along the time dimension. The longer the length of the convolutional kernel, the more temporal information it captures, resulting in smaller output features. C = 40 is used to control the number of convolutional kernels. Then, along the channel dimension, the output features are concatenated to obtain the final output result V, the process for which is as follows:

$$V = \text{concat}(F_1, F_2, F_3, F_4) \in R^{C\times N\times T_1} \tag{2}$$

Here, $T_1 = t_1 + t_2 + t_3 + t_4 = 550$, and "concat" denotes the concatenation along the channel dimension. Through the above steps, the original data X are transformed into discriminative data V with features at different scales.

After the multi-scale temporal block, the original input X has extracted refined features V with different scales of temporal information across multiple channels. At this point, the channel dimension of the original data has been expanded, and the number of time points has increased. However, the number of electrodes N (representing the original spatial information of EEG recordings) remains unchanged. Therefore, a spatial attention block is designed to extract the most discriminative features in both the spatial and channel dimensions. For the input data $V \in R^{C\times N\times T_1}$, the spatial features are extracted from V using spatial attention. Initially, Maxpool and Averagepool are applied to V for pooling:

$$F_M = \text{Maxpool}(V) \in R^{1\times N\times T_1} \tag{3}$$

$$F_A = \text{Avgpool}(V) \in R^{1\times N\times T_1} \tag{4}$$

By applying these two pooling methods, we obtain the maximum and average values for each spatial position. These result in two two-dimensional matrices. Subsequently, they are concatenated and sent for convolution:

$$F = \text{conv}(\text{concat}(F_M, F_A)) \in R^{1\times N\times T_1} \tag{5}$$

By applying a softmax operation to the feature map, we normalize its values to a range between 0 and 1. Then, we obtain the attention map F_S:

$$F_S = \text{softmax}(F) \in R^{1\times N\times T_1} \tag{6}$$

The attention map F_S learns the weights for each spatial position. Applying these weights to each position of the input data yields an updated F^*, where every position in V is updated.

$$F^* = V \otimes F_S \in R^{C\times N\times T_1} \tag{7}$$

Here, the symbol $\otimes$ represents the element-wise multiplication between the attention map F_S (compressed) and the data for each channel $v_i \in R^{N\times T_1}$. This process essentially performs a weighted sum update from the perspective of the channel dimension, resulting in the updated feature represented as F^*. Each v_i contains spatial information of the electrode positions and temporal information at different time scales. Thus, after passing through spatial attention, both spatial and temporal information are refined. This yields a refined feature map. Subsequently, to prepare for input into the graph neural network, F is fed into a one-dimensional convolutional layer with a kernel-size of $N \times 1$ to average the influence across different electrodes. Therefore, after the multi-scale spatio-temporal module, we obtain the refined feature $F^* \in R^{C\times 1\times T_1}$, containing crucial information from the global and local receptive fields of different electrodes in the EEG recordings.

2.2.2. Dynamic Graph Convolution Module

Unlike traditional methods and previous deep-learning approaches, graph neural networks transform the original EEG signals into graphs. This transformation leverages the inherent spatial structure of EEG data to establish functional connections and extract features. Consequently, these networks reveal patterns and predict complex relationships between different nodes, offering a more detailed analysis [33]. After the previous module, spectral graph convolutional networks (GCNs) are employed to construct the dynamic graph convolution network (DGCN) module. In contrast to traditional methods where electrodes serve as nodes, in this paper, refined features F^* are utilized to construct the input graph, where C corresponds to the number of nodes in the graph, and T_1 corresponds to the length of each node feature vector.

An undirected weighted graph can be represented as G = (V, A, H), where $V = [v_1, v_2, \ldots, v_C]$ denotes the vertex set, with v_i representing the i-th node. $A \in R^{C\times C}$ represents the adjacency matrix of graph G, used to denote the weights of the edges in the weighted graph. Each a_{ij} in A denotes the weight of the edge between the i-th and j-th nodes. $H \in R^{C\times T_1}$ represents the node features of dimension T_1. Given the existing nodes and node features, the key to forming a graph lies in determining the adjacency matrix, because a good adjacency matrix can represent the structural relationships between different nodes. The Pearson correlation coefficient measures the degree of linear correlation between two variables, ranging from -1 to 1. A positive correlation indicates a positive relationship between the two variables, a negative correlation indicates a negative relationship, and a correlation of zero indicates no linear correlation between the variables. Therefore, we use the Pearson correlation coefficient to assign weights in the adjacency matrix A for each graph within the dynamic graph convolution module. This method constructs graphs that

accurately reflect the structural relationships between the nodes. For input vectors p and q of length n, their Pearson correlation coefficient ρ_{pq} is calculated by the following formula:

$$\rho_{pq} = \frac{\sum_{i=1}^{n}\left(p_i - \bar{p}\right)(q_i - \bar{q})}{\sqrt{\sum_{i=1}^{n}\left(p_i - \bar{p}\right)^2 \cdot \sum_{i=1}^{n}\left(q_i - \bar{q}\right)^2}} \tag{8}$$

where p_i is the value at the i-th position of vector p, $\bar{p}$ is the mean of vector p, and q_i and $\bar{q}$ are defined similarly. Based on the Pearson correlation coefficient ρ, the value of each a_{ij} in the adjacency matrix A is calculated as follows:

$$a_{ij} = \begin{cases} \rho_{ij}, \ \rho_{ij} \geq \tau \\ 0, \ \text{otherwise} \end{cases} \tag{9}$$

where τ is a manually set threshold. In this study, τ is set to 0.5. After applying this method, approximately 60% of the values of a_{ij} exceed the threshold.

The spectral decomposition of a graph is defined as the eigendecomposition of the graph's Laplacian matrix [40]. The calculation formula for the graph's Laplacian matrix L is:

$$L = D - A \tag{10}$$

$$\hat{L} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{11}$$

where A is the adjacency matrix, D is the degree matrix of the graph, and I is the identity matrix. Usually, the normalized Laplacian $\hat{L}$ is used. The formula for the graph's Fourier transform is $\hat{H} = U^T H$, where U is the standardized orthogonal matrix of the eigenvectors of the graph's Laplacian matrix L. In a spectral GCN, the convolution operation (*) is defined as:

$$H * g = U((U^T H) \otimes \left(U^T g\right)) \tag{12}$$

Here, g represents the spatial domain kernel in the graph convolution, and $\otimes$ denotes the element-wise multiplication. The convolution in the original spectral GCN requires significant computational resources due to the decomposition of the graph's Laplacian matrix, particularly in calculating U. To reduce this computational burden, some studies have adopted a widely used approximation known as the Chebyshev GNN (ChebConv) [41]. The update equation for node embeddings in the ChebConv is defined as:

$$S = H * g \approx \sum_{i=1}^{K} \Theta_i T_i\left(\hat{L}'\right) \tag{13}$$

where $\Theta_i \in R^{K \times G \times G}$ is a learnable parameter, K is a hyperparameter representing K Chebyshev filters, and G is the number of nodes after the GCN layer. During the model's learning process, Θ_i is automatically updated. $\hat{L}'$ is calculated by the following equation:

$$\hat{L}' = \frac{2L}{\lambda_{max}} - I \tag{14}$$

where λ_{max} is the largest eigenvalue of L, typically approximated as $\lambda_{max} = 2$. For $T_i\left(\hat{L}'\right)$, its calculation is given by the following equation:

$$\begin{cases} T_i\left(\hat{L}'\right) = 2T_{i-1}\left(\hat{L}'\right) - T_{i-2}\left(\hat{L}'\right), \ i > 2 \\ T_2\left(\hat{L}'\right) = \hat{L}'H, \ i = 2 \\ T_1\left(\hat{L}'\right) = H, \ i = 1 \end{cases} \tag{15}$$

For the input features, after passing through 5 layers of the GCN, the final feature vectors S_5 are obtained, where $G_1 = 256$, $G_2 = 192$, $G_3 = 128$, $G_4 = 64$, $G_5 = 43$. To refine the results, C convolutional kernels of size $T_2 \times 1$ are used to change the dimensions of the feature map S_5. Therefore, after the dynamic graph convolutional module, the final feature mapping $S^* \in R^{C \times 1 \times T_2}$ is obtained.

2.2.3. Classification Module

In this module, three consecutive fully connected layers are applied to the convolved features S^* for dimensionality reduction. Subsequently, a softmax layer is utilized to transform the output into probabilities corresponding to each class, yielding the final predicted label $\hat{y}$. The operation is illustrated by the following equation:

$$\hat{y} = \text{Softmax}(W_3(W_2(\text{Relu}(W_1S^* + b_1)) + b_2) + b_3) \tag{16}$$

where W_i and b_i are the weight matrix and bias matrix of the i-th fully connected layer, $\hat{y} \in R^L$ is a vector of length L, where each position represents a probability value, and the sum of all the L positions equals 1. During prediction, the position with the highest probability value is taken as the final predicted category. The first layer is set to have 32 units, the second layer is set to have 8 units, and the last layer is set to have L units, where in this study, L = 4.

The loss function used in this paper is the cross-entropy loss function, and the final loss $loss_L$ can be obtained from the following equation:

$$loss_L = -\frac{1}{M}\sum_{i=1}^{M}\sum_{l=1}^{L} y_{i,l} \cdot \log\left(\hat{y}_{i,l}\right) + \lambda\|\Theta\|_1 + \mu\|A\|_1 \tag{17}$$

where M is the total number of samples in the current batch, $y_{i,l}$ is the true label of class l for the i-th sample, taking the value of 0 or 1; $\hat{y}_{i,l}$ is the predicted probability of the i-th sample belonging to class l; and Θ represents all the trainable parameters of the model. By applying the l_1-norm regularization to the model parameters and the dynamic adjacency matrix A in the $loss_L$, overfitting can be effectively avoided. λ and μ are hyperparameters, set to 0.000005 and 0.3, respectively, in this paper. All the experiments were conducted on a Windows11 system with an Nvidia RTX 3060 12GB GPU, using Python 3.6 and the PyTorch 1.10 platform. We trained the model for 300 epochs.

3. Experiments and Results

In this section, we will introduce the data augmentation methods, datasets, preprocessing techniques, and experiments conducted.

3.1. Datasets and Evaluation Metrics

To validate our approach, this study utilized the dataset 2a from the BCI Competition IV [42], which comprises recorded data from 22 EEG electrodes and 3 EOG electrodes for

9 participants. The data were sampled at 250 Hz, and bandpass filtering was applied from 0.5 Hz to 100 Hz by the data collectors. The electrode placement is illustrated in Figure 3.

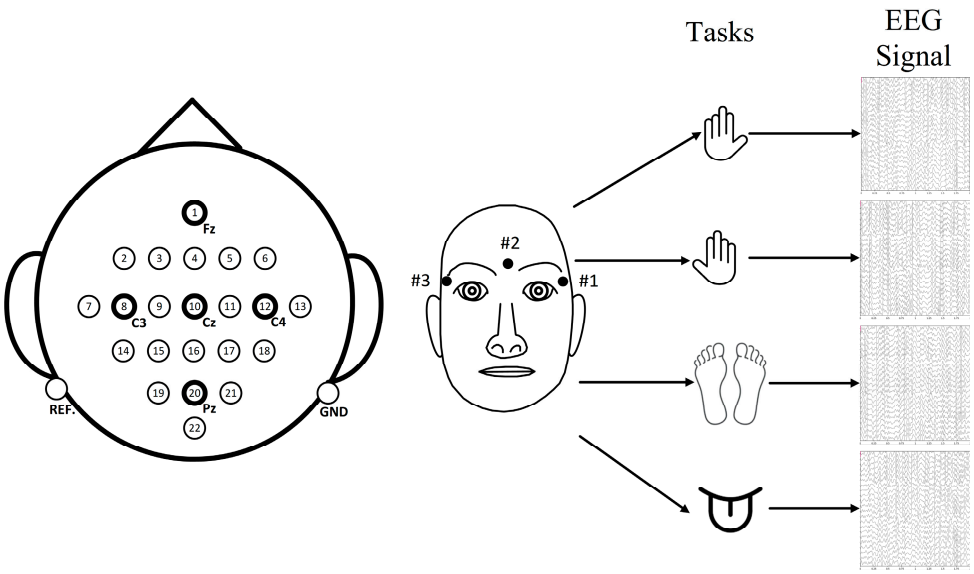

Figure 3. Electrode placement locations [42] and tasks.

The participants were instructed to perform four types of motor imagery tasks: left hand, right hand, tongue, and both feet. During the experiment, at t = 0 s, accompanied by a beep sound, a fixation cross appeared in the center of the screen to indicate the start of the current trial. At t = 2 s, an arrow appeared on the screen for 1.25 s, pointing left, right, down, or up (corresponding to the left hand, right hand, both feet, and tongue, respectively), prompting the participant to perform the corresponding motor imagery task. The participant continued the imagining until the arrow disappeared at t = 6 s, followed by a 1.5 s break time. Each participant completed two sessions, with each session having 72 trials across 4 categories. This resulted in a total of 288 trials per session and 576 trials per participant. The dataset comprised a total of 5184 trials.

This paper primarily evaluates the performance of the proposed strategy using the accuracy (denoted by symbol Acc) and Kappa value (denoted by symbol Kappa), as defined by the following equations.

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{18}$$

$$\text{Kappa} = 1 - \frac{1 - \text{P}_\text{o}}{1 - \text{P}_\text{e}} = \frac{\text{P}_\text{o} - \text{P}_\text{e}}{1 - \text{P}_\text{e}} \tag{19}$$

where TP is the number of samples correctly classified as positive, TN is the number of samples correctly classified as negative, FP is the number of samples incorrectly classified as positive, and FN is the number of samples incorrectly classified as negative. P_o is the actual accuracy of the classifier, which is the ratio of the number of samples correctly predicted by the classifier to the total number of samples. P_e is the chance-corrected accuracy predicted by the classifier, which is the expected ratio of correctly predicted samples in the absence of bias [43].

3.2. Data Augmentation and Experimental Setup

3.2.1. Data Augmentation

Previous studies in MI-BCI research that utilize deep learning have employed various data augmentation methods to augment the original dataset [29], leading to good improvements in the results. However, data augmentation methods should not be overly complex,

as this can increase the time cost of data processing and have a significant impact on the training and testing time of the model. Therefore, based on previous research, this paper adopts the following two data augmentation methods to augment the original data.

- Fourier Transform Surrogate (FTS): For the original signal X, the primary operation of the FTS is to calculate the Fourier coefficients for all the channels and then add random noise to their phases.

$$F[FTS(X)](f) = F[X](f)e^{i\Delta\varphi} \tag{20}$$

- where F represents the Fourier transform operator, f is a frequency, $\Delta\varphi$ is a frequency-specific random perturbation, and after adding the perturbation, the inverse Fourier transform is applied to obtain the newly augmented signal. This augmentation primarily encourages the model to learn the power spectral density (PSD) information of the signal. In this study, $\Delta\varphi$ is set to $9/10\ \pi$.
- SmoothTimeMask (STM): For the original signal X with a total time length of t, the STM mainly operates on a specific channel. It randomly samples a time point t_cut from the time range [0, t − Δt], and then it replaces the length of Δt with 0. This augmentation primarily encourages the model to be less affected by transient signal jumps that are irrelevant to the task, thus facilitating the learning of task-relevant patterns.

The effectiveness of these two methods is illustrated in Figure 4.

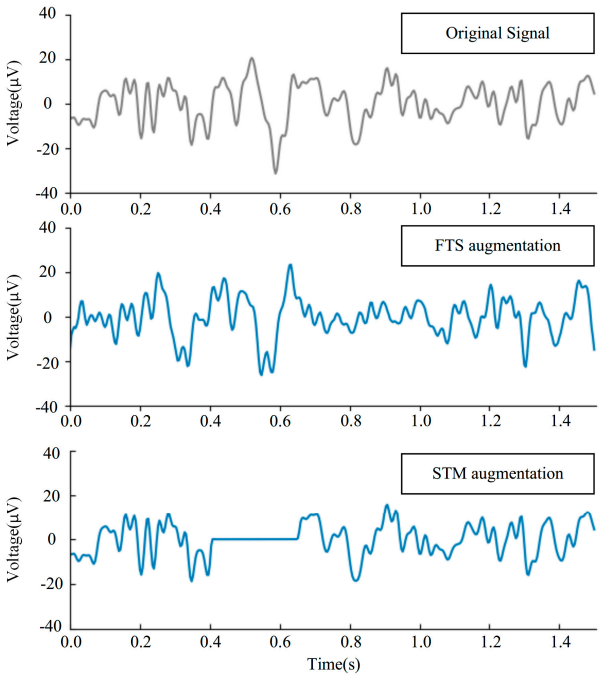

Figure 4. The data augmentation methods used.

Both data augmentation methods used were able to process the data efficiently in a short amount of time and obtain augmented data of the same size as the original data. We employed version 0.8 of the Braindecode library in Python for the data augmentation.

3.2.2. Preprocessing and Experimental Setup

In the preprocessing stage, the focus is on removing task-irrelevant factors from the raw data. Firstly, the 25 signal acquisition electrodes consist of 3 EOG electrodes and 22 EEG electrodes. In this study, data from the 3 EOG electrodes were removed to prevent interference from eye movement artifacts. Performing motor imagery tasks induces task-related synchronization (ERD) and task-related desynchronization (ERS) in different

brain regions, with the main perceptual motor rhythms being mu (8–13 Hz) and beta waves (18–30 Hz). The specific frequency ranges vary from person to person, thus bandpass filtering from 4–40 Hz was applied to the raw data to minimize the negative effects while retaining the MI task-related features. For each sample, data from 0.5 s to 2.5 s after the start of the trial were utilized, resulting in a size of 22 × 500 for each sample.

Both the FTS and STM data augmentation methods preserve the size of each data sample after augmentation, resulting in a dataset that is twice the original volume. Specifically, if the original dataset comprises N samples, each of size 22 × 500, the size of each sample remains unchanged at 22 × 500 after augmentation, effectively doubling the number of samples to 2N. Therefore, the total data volume becomes 2N × 22 × 500 at this point.

We also utilized a transfer-learning strategy to pre-train the proposed network model. For each participant i (i = 1, . . ., 9), we pre-trained the model using data from the other eight participants, froze the model's parameters, unfroze the model after augmenting the data of participant i, and then trained and tested participant i on the pre-trained model. In both the training and pre-training phases, the ratio of training data to testing data was 8:2, meaning an 80% training set and a 20% testing set.

3.3. Experimental Results

3.3.1. MI Recognition Results Based on MST-DGCN with Data Augmentation

To avoid errors caused by random data partitioning, the experiment was repeated three times, each time with random splits for the training and testing sets. The average of the three experiments was taken as the final result. The evaluation metrics obtained from the experiments (Table 1 and Figure 5) show that our proposed method achieved the highest overall recognition accuracy of 77.89% on the BCI-IV-2a database, which was achieved when using the FTS data augmentation method, with a corresponding Kappa value of 0.7025.

Table 1. MI recognition results of the MST-DGCN. The values in bold represent the best results.

Methods	Subjects' Acc (%)										Kappa
	A01	A02	A03	A04	A05	A06	A07	A08	A09	Avg	
MST-DGCN	85.71 ± 0.22	55.07 ± 1.73	92.36 ± 0.90	67.35 ± 0.87	65.90 ± 1.62	55.87 ± 3.12	88.54 ± 2.14	79.28 ± 0.93	84.31 ± 0.40	74.93 ± 0.21	0.6657
STM + MST-DGCN	84.10 ± 2.08	59.19 ± 5.00	92.47 ± 0.74	68.51 ± 1.50	68.86 ± 1.98	58.51 ± 1.74	88.91 ± 4.40	80.88 ± 0.52	**84.78** ± 0.86	76.25 ± 1.53	0.6833
FTS + MST-DGCN	**86.40** ± 1.19	**64.81** ± 5.95	**94.89** ± 0.35	**69.18** ± 3.28	**69.73** ± 0.33	**60.80** ± 1.26	**89.22** ± 1.70	**81.69** ± 0.56	84.26 ± 0.59	**77.89** ± 0.75	**0.7052**

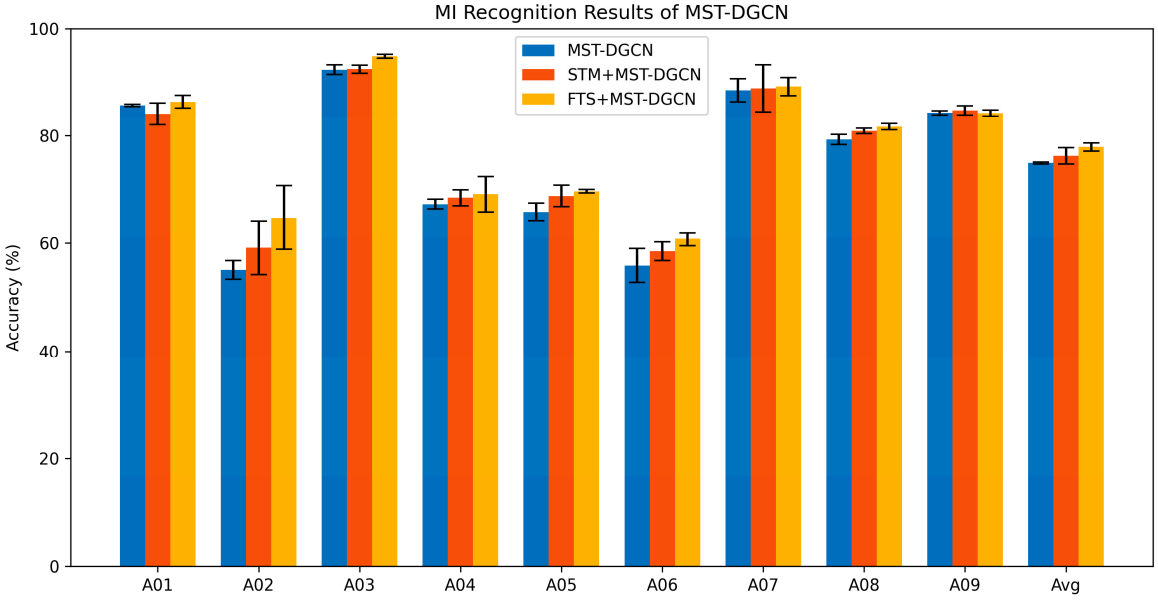

Figure 5. Bar chart of the MI recognition results of the MST-DGCN.

Despite the limited data in the BCI-IV-2a database, significant improvements were still achieved by using the data augmentation methods. Compared to not using data augmentation methods, the STM and FTS data augmentation methods increased the accuracy by 1.32% and 2.96%, respectively, with corresponding Kappa values increasing by 0.0176 and 0.0395, respectively. Inspired by Altaheri et al. [44] and Thanigaivelu et al. [45], we have plotted two row-normalized confusion matrices (Figure 6) to visually present the classification performance.

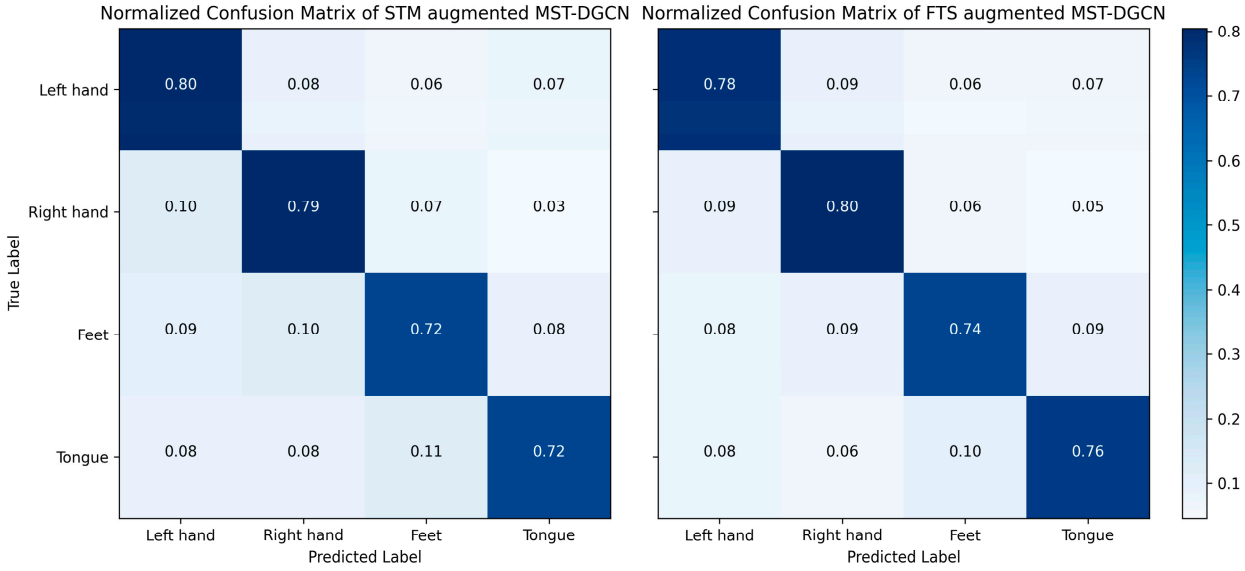

Figure 6. Confusion matrix of the MST-DGCN with STM and FTS augmentation.

In the confusion matrix, the rows represent the actual classes, while the columns represent the predicted classes. Each cell's value represents the number of samples for which the actual class matches the row index and the predicted class matches the column index. For example, the first row in Figure 6 represents the number of samples with the actual class of left-hand motor imagery, while the first column indicates the number of samples the model predicted as left-hand motor imagery. We have normalized the confusion matrix by rows. As shown in Figure 6, different augmentation methods have varying effects on different classes. The MST-DGCN augmented with STM demonstrates a slight advantage in classifying the left- and right-hand tasks, while the MST-DGCN augmented with FTS exhibits a greater advantage in classifying the foot and tongue tasks. The experimental results underscore the effectiveness and robustness of the proposed model.

To further understand the performance of FTS augmentation, we plotted its ROC curves, as shown in Figure 7. The micro-average ROC curve represents the micro-averaged true positive rate and false positive rate for all the classes, calculated by combining the true positive rates and false positive rates of all the classes into one ROC curve. This curve provides a better evaluation of the overall performance. The macro-average ROC curve represents the ROC curve obtained by averaging the ROC curves of each class. It is computed by averaging the true positive rates and false positive rates of each class separately, offering a better assessment of the classification performance for each class. In addition to these two curves, we can also examine the ROC curve for each individual class to understand the classification performance of the FTS across different classes, where classes 0, 1, 2, and 3 represent the tongue, foot, right-hand, and left-hand motor imagery, respectively. From Figure 7, it can be observed that the curves for classes 2 and 3 have larger areas under the curve, indicating better performance. On the other hand, the recognition results for classes 0 and 1 are relatively lower, suggesting that the latter two types of motor imagery are indeed more challenging than the former.

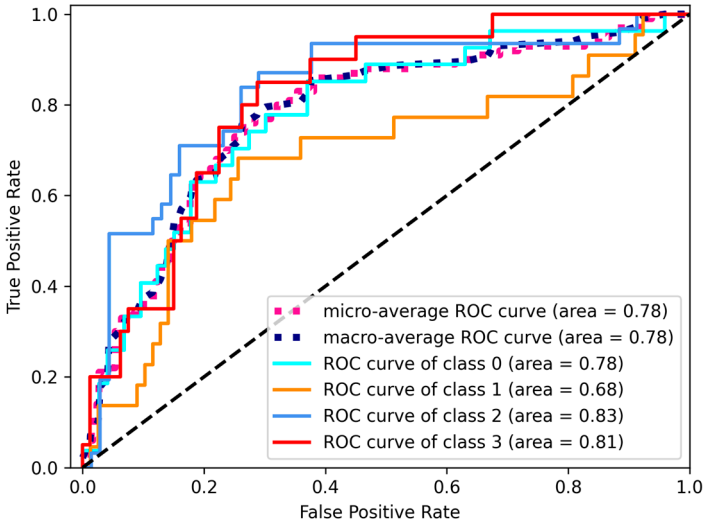

Figure 7. ROC curve plot of the MST-DGCN augmented by FTS.

3.3.2. Ablation Experiments of MST-DGCN

To evaluate the effectiveness of the modules in the proposed MST-DGCN model, we conducted ablation experiments to confirm the effects of each module used in our approach. The chosen data augmentation method was the FTS, as it had the best results. The results of the ablation experiments are presented in Table 2 below:

Table 2. Ablation experiments of the MST-DGCN. The Checkmark "✓" indicates the selection of the corresponding module, and the values in bold represent the best results.

Modules			Acc (%)
MT	**SA**	**DGC**	
✓			71.08%
	✓		68.73%
		✓	66.24%
✓		✓	73.80%
	✓	✓	72.32%
✓	✓		74.96%
✓	✓	✓	**77.89%**

where the MT module represents the multi-scale temporal module, the SA module represents the spatial attention module, and the DGC module represents the dynamic graph convolution module. From Table 2, it can be observed that the removal of any individual module has a relatively minor impact on the performance. The most significant change occurred when the MT module was removed, resulting in a decrease in the accuracy (Acc) of 5.57%. Removing the SA module and DGC module individually led to decreases in the accuracy of 4.09% and 2.93%, respectively. This demonstrates the crucial role of the MT module in effectively integrating global information. The combination of the MT and SA modules achieved the second best performance, indicating that our SA module can effectively refine the global information obtained by the MT module, highlighting the critical roles of the MT and SA modules in information fusion and extraction. It is worth noting that any network solely utilizing the DGC module did not perform well. We speculate that the DGC module may be more suitable for combination with other modules, relying on contextual information.

In the multi-scale temporal module, we used convolutional kernels of sizes (64, 1), (40, 1), (26, 1), and (16, 1) to extract temporal information at different scales. We conducted ablation experiments on the different scale convolutional kernels to further explore which scale plays a prominent role in the feature extraction. The results are shown in Table 3.

Table 3. Ablation experiments of the multi-scale temporal module. The Checkmark "✓" indicates the selection of the corresponding module, and the values in bold represent the best results.

Used Scales		Acc (%)
Small Scales (26, 1) and (16, 1)	**Large Scales (64, 1) and (40, 1)**	
✓		73.20%
	✓	75.44%
✓	✓	**77.89%**

From Table 3, we can see that the accuracy loss is smaller when using larger-scale convolutional kernels, while the accuracy loss is greater when using smaller-scale convolutional kernels. We think this may be because the motor imagery paradigm requires capturing the EEG signal features over a longer time range, and larger convolutional kernels can more effectively extract these features. Therefore, in the feature extraction process, larger convolutional kernels play a more important role in capturing relevant information.

3.3.3. The Impact of Model Parameters Used in MST-DGCN

In the MST-DGCN model, the DGC module utilizes a five-layer GCN. Various hyperparameters in the GCN (such as the number of GCN layers, settings for parameter updates, etc.) have a great impact on the results. Therefore, in this section, we explore these model parameters to demonstrate the effects of different parameters in the DGC module. In the experiments where parameters are modified, the other modules and parameters of the model remain unchanged. Similar to the previous section, we still use the best-performing FTS-augmented MST-DGCN for experimentation. Each experiment was also conducted three times to calculate an average value.

To validate the appropriateness of the number of layers used, we experimented with setting the number of GCN layers from one to seven while keeping the other parts unchanged. The experimental results are shown in Table 4.

Table 4. Difference in the average classification accuracy with varying numbers of GCN layers. The values in bold represent the best results.

Layers	Acc
1–2	All results < 70%
3	71.98%
4	75.59%
5	**77.89%**
6	72.34%
7	69.76%

From Table 4, it can be observed that selecting an appropriate number of GCN layers is crucial. Too many or too few GCN layers can lead to fluctuations in the results. The model achieved the best performance when using five GCN layers. This indicates that, for our task, five GCN layers offer the optimal feature extraction and representation-learning capabilities, enabling the model to better capture the features and patterns of MI signals. Using too few layers in a model may result in insufficient complexity to effectively capture the intricate features of the data. Conversely, employing too many layers can lead to overfitting and the learning of unnecessary features. Therefore, our conclusion is that, for the current task, selecting five GCN layers as the depth of the MST-DGCN is the most appropriate choice.

The hyperparameter μ in the loss function formula is responsible for adjusting the generated adjacency matrix A. Adjusting the hyperparameter μ affects the sparsity of the updated adjacency matrix, which in turn influences the information propagation and feature extraction capabilities of the graph convolutional layers. We conducted experi-

ments by setting μ from 0.1 to 0.5, with a step size of 0.1, to find the optimal value of μ. Table 5 displays the results of different μ values for the proposed MST-DGCN.

Table 5. Average classification accuracy results corresponding to different values of μ. The values in bold represent the best results.

μ	Acc
0.1	73.83%
0.2	75.31%
0.3	**77.89%**
0.4	76.56%
0.5	75.92%

From Table 5, it can be observed that when the value of μ is set to 0.3, the MST-DGCN achieves the highest classification accuracy.

4. Discussion

4.1. Analysis of Data Augmentation Methods

We used two data augmentation methods, STM and FTS, to augment the original dataset. To thoroughly analyze the effectiveness of the data augmentation methods used, we conducted an analysis for each participant. Figure 8 shows the results of the three methods across the nine participants.

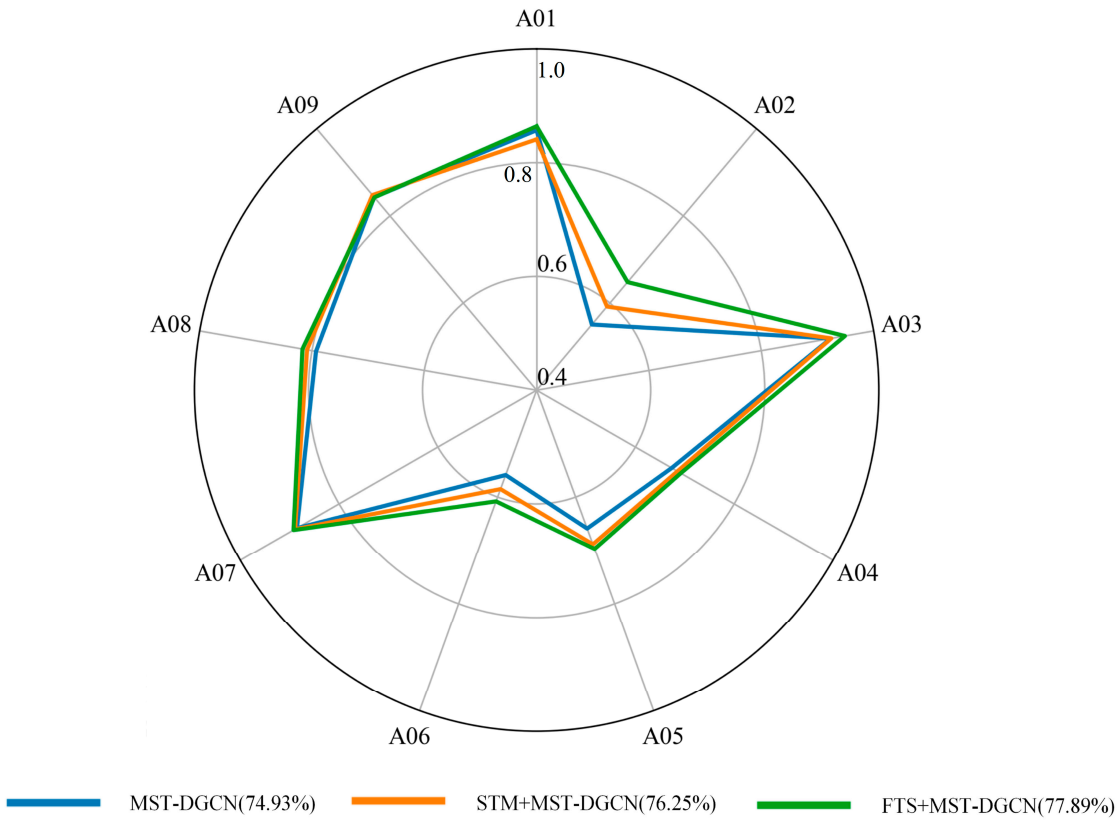

Figure 8. MI recognition results of the three methods.

From Figure 8, it can be observed that the FTS + MST-DGCN method achieved the best recognition result, reaching 77.89%. Even though the STM + MST-DGCN method performed better on participant nine (STM: 84.78% > FTS: 84.26%), the difference is not significant.

Additionally, for participants A01, A03, A07, A08, and A09, who already had high recognition accuracy (>80%), the improvement via these two data augmentation methods was low. The average improvement for these five participants after augmentation was only 0.19% (STM augmented) and 1.25% (FTS augmented). On the other hand, for participants A02, A04, A05, and A06, who had lower recognition accuracy (<70%), the improvement via data augmentation was significant. After augmentation, the average improvement for these four participants was 2.72% (STM augmented) and 5.08% (FTS augmented). The most significant improvement was observed for participant A02, with a 9.74% increase using FTS data augmentation. Both data augmentation methods expanded the original dataset. STM augmentation involved short-term smoothing of each channel, similar to the dropout in neural networks, aiming to help the model learn robust features and avoid overfitting. On the other hand, FTS encouraged the model to learn frequency domain information. Both augmentation methods enhanced the features, leading to consistent improvement effects.

Figure 8 also shows the differences in data quality among the different subjects. The quality of subjects A02, A04, A05, and A06 is significantly lower than that of the other subjects. We have also observed similar classification results in other studies [13,16,17].

4.2. Analysis of Model Complexity and Visualization

To evaluate the computational complexity of our MST-DGCN model, we computed multiple complexity indicators. We then integrated these indicators with the average classification accuracy for comparison against several other methods. The results of the comparison are shown in Table 6.

Table 6. Comparison of computational complexity.

Methods	Parameter (Million)	Inference Time (ms)	Decoding Time (ms)	Acc (%)
ConvNet	0.15	0.85	6.87	72.53
EEGNet	0.03	0.32	4.28	74.61
TS-SEFFNet	1.34	3.29	18.48	74.71
MST-DGCN	0.16	1.27	11.35	77.89

As shown in Table 6, the trainable parameters in the MST-DGCN are approximately 0.16 million, which is more than ConvNet's 0.15 million and EEGNet's 0.03 million. However, our model achieves higher accuracy than both of them and significantly lower than the TS-SEFFNet. The inference time represents the time required by the deep-learning model to provide the recognition result for each trial, while the decoding time represents the time from the inputting of raw EEG data to providing the recognition result, also shown in Table 6. The MST-DGCN significantly improves the classification accuracy while maintaining acceptable inference and decoding times, demonstrating its advantage.

We also visualized the average values of the dynamic adjacency matrices A used for all the participants, as shown in Figure 9.

The magnitude of the values in the adjacency matrix (the darkness of colors) reflects the correlation between connections. A redder color indicates a stronger positive correlation between the two corresponding connections, while a bluer color indicates a stronger negative correlation. To highlight the connections with strong correlations, the cube of the adjacency matrix A was computed and displayed, as shown on the right side of Figure 9. It can be observed that, through training, the MST-DGCN has indeed learned the critical connections.

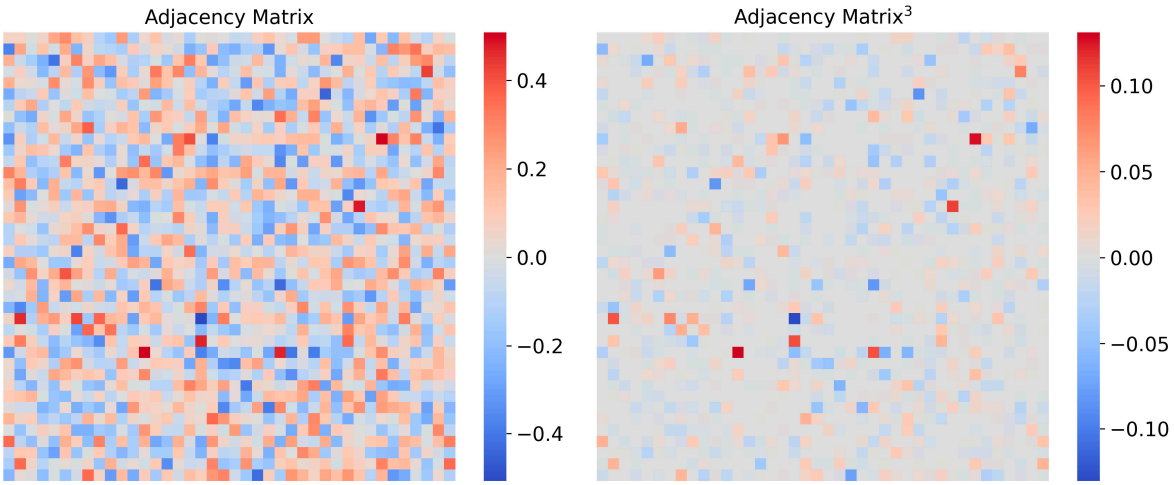

Figure 9. Visualization of the learned adjacency matrix A and its cube by the MST-DGCN, with a size of 40 × 40.

4.3. Comparison with Other Studies

We conducted a survey of the MI recognition literature that also utilizes the BCI-IV-2a database, summarizing and consolidating the results (Table 7).

Table 7. MI recognition results from the BCI-IV-2a database.

Methods	Year	Acc	Kappa
FBCSP [13]	2008	67.75%	0.5700
CCSP [14]	2009	66.51%	0.5535
ConvNet [16]	2017	72.53%	0.6337
EEGNet [17]	2018	74.61%	0.6615
Incep-EEGNet [18]	2020	74.07%	0.6543
TS-SEFFNet [26]	2021	74.71%	0.6628
MRGF [46]	2022	70.11%	0.6015
MI-DABAN [27]	2023	76.16%	0.6821
SCPGE [47]	2023	68.64%	0.5817
Our Method	2024	77.89%	0.7052

Ang et al. [13] decomposed the original EEG signals into multiple frequency bands using filter banks and then extracted the features using the common spatial pattern (CSP) method, achieving an accuracy of 67.75% and a Kappa value of 0.57 on the BCI-IV-2a database. Kang et al. [14] considered the relationship between the covariance matrices of different subjects and obtained a composite covariance matrix by weighted averaging the covariance matrices of subjects in the dataset. The accuracy and Kappa value for the MI classification were 66.51% and 0.5535, respectively. Schirrmeister et al. [16] designed a CNN model consisting of multiple convolutional layers, pooling layers, and fully connected layers, which could deeply extract the spatio-temporal features from EEG signals without relying on handcrafted feature extraction methods. The overall accuracy and Kappa value on the BCI-IV-2a database were 72.53% and 0.6337, respectively. Lawhern et al. [17] specifically designed a CNN for processing EEG data, called EEGNet, which was relatively lightweight and suitable for resource-constrained environments. The classification accuracy and Kappa value achieved using the BCI-IV-2a dataset were 74.61% and 0.6615, respectively. Riyad et al. [18] embedded inception modules into the architecture of EEGNet to better capture the multi-scale spatio-temporal features of EEG signals. The experimental results on the BCI-IV-2a dataset showed an accuracy of 74.07% and a Kappa value of 0.6543. Li et al. [26] used a squeeze-and-excitation feature fusion network to extract temporal-spectral features relevant to MI tasks, achieving an overall accuracy of 74.71% and a Kappa value of 0.6628. Xie et al. [46] enhanced the classification performance of motor imagery BCI

by constructing multiple Riemannian maps corresponding to multiple bandpass frequency bands and applying graph-embedding projection and mutual information-based graph fusion to simultaneously extract the spatial and spectral features from the Riemannian maps, achieving an accuracy of 70.11% and a Kappa value of 0.6015. Li et al. [27] improved the MI recognition performance of individual subjects by learning the knowledge of multiple subjects through an adversarial-learning method and two non-shared attention blocks for domain alignment, achieving an accuracy of 76.16% and a Kappa value of 0.6821. Ni et al. [47] combined graph-embedding projection and sparse constraints into a semi-supervised least squares model and then adaptively learned the similarity matrix and pseudo-label matrix based on manifold learning to obtain the most discriminative patterns, achieving an accuracy of 68.64% and a Kappa value of 0.5817. The discussed literature highlights the excellent accuracy and Kappa value of the proposed MST-DGCN in MI recognition, which is of great significance to the development of MI-BCI.

5. Conclusions

In this paper, we propose a network called the MST-DGCN for MI-EEG classification. It first utilizes the multi-scale spatio-temporal module, which incorporates multi-scale convolution and spatial attention, to extract and refine temporal and spatial information at different scales from the raw EEG signals. Then, multiple layers of dynamic graph convolution modules are employed to capture the spatial structural relationships among potential connections in the graph. Finally, a classification module consisting of convolutional and fully connected layers is used to output the final results. The experimental results demonstrate that our method achieves an identification accuracy of 77.89% and a Kappa value of 0.7052 on the BCI-IV-2a dataset, outperforming nine other methods. However, there are still areas for improvement in our method. For example, all the experiments conducted in this paper are set in offline scenarios, and future work could investigate integrating the model into real-time BCI systems to evaluate the performance under different conditions. Additionally, the data augmentation methods used could be further expanded. Both data augmentation methods provide stable improvements for the MST-DGCN. The FTS-augmented MST-DGCN shows better, but not statistically significant, results. This necessitates further exploration and innovation in data augmentation and experimental paradigms. There are also significant differences in data quality among different subjects in the dataset. Identifying available data from low-quality datasets to better understand the reliability of specific recordings is one of the issues that need to be addressed moving forward. This study was conducted on only one dataset. In the future, more extensive and detailed exploration of the model generalizability on multiple datasets is needed. These limitations and the corresponding future research directions contribute to the further development and enhancement of our method.

Author Contributions: Conceptualization, Y.C. and D.L.; methodology, Y.C., P.L. and D.L.; software, Y.C. and P.L.; validation, P.L.; formal analysis, Y.C. and D.L.; investigation, P.L.; resources, P.L.; data curation, P.L.; writing—original draft preparation, Y.C. and P.L.; writing—review and editing, Y.C. and D.L.; visualization, P.L.; supervision, Y.C. and D.L.; project administration, Y.C. and D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are openly available at the following URL/DOI: https://bbci.de/competition/iv/ (accessed on 25 October 2023).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wolpaw, J.R.; Birbaumer, N.; McFarland, D.J.; Pfurtscheller, G.; Vaughan, T.M. Brain–computer interfaces for communication and control. *Clin. Neurophysiol.* **2002**, *113*, 767–791. [CrossRef] [PubMed]
2. Lebedev, M.A.; Nicolelis, M.A. Brain–machine interfaces: Past, present and future. *Trends Neurosci.* **2006**, *29*, 536–546. [CrossRef] [PubMed]

3. Pfurtscheller, G.; Neuper, C.; Muller, G.; Obermaier, B.; Krausz, G.; Schlogl, A.; Scherer, R.; Graimann, B.; Keinrath, C.; Skliris, D. Graz-BCI: State of the art and clinical applications. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2003**, *11*, 1–4. [CrossRef] [PubMed]
4. Kübler, A.; Birbaumer, N. Brain–computer interfaces and communication in paralysis: Extinction of goal directed thinking in completely paralysed patients? *Clin. Neurophysiol.* **2008**, *119*, 2658–2666. [CrossRef] [PubMed]
5. Allison, B.Z.; Wolpaw, E.W.; Wolpaw, J.R. Brain–computer interface systems: Progress and prospects. *Expert Rev. Med. Devices* **2007**, *4*, 463–474. [CrossRef] [PubMed]
6. Carlson, T.; Millan, J.d.R. Brain-controlled wheelchairs: A robotic architecture. *IEEE Robot. Autom. Mag.* **2013**, *20*, 65–73. [CrossRef]
7. Bundy, D.T.; Souders, L.; Baranyai, K.; Leonard, L.; Schalk, G.; Coker, R.; Moran, D.W.; Huskey, T.; Leuthardt, E.C. Contralesional brain–computer interface control of a powered exoskeleton for motor recovery in chronic stroke survivors. *Stroke* **2017**, *48*, 1908–1915. [CrossRef] [PubMed]
8. Moldoveanu, A.; Ferche, O.; Moldoveanu, F.; Lupu, R.; Cinteză, D.; Irimia, D.; Toader, C. The travee system for a multimodal neuromotor rehabilitation. *IEEE Access* **2019**, *7*, 8151–8171. [CrossRef]
9. Staffa, M.; Giordano, M.; Ficuciello, F. A WiSARD network approach for a BCI-based robotic prosthetic control. *Int. J. Soc. Robot.* **2020**, *12*, 749–764. [CrossRef]
10. Mane, R.; Chouhan, T.; Guan, C. BCI for stroke rehabilitation: Motor and beyond. *J. Neural Eng.* **2020**, *17*, 041001. [CrossRef]
11. Sebastián-Romagosa, M.; Cho, W.; Ortner, R.; Murovec, N.; Von Oertzen, T.; Kamada, K.; Allison, B.Z.; Guger, C. Brain computer interface treatment for motor rehabilitation of upper extremity of stroke patients—A feasibility study. *Front. Neurosci.* **2020**, *14*, 591435. [CrossRef] [PubMed]
12. Grosse-Wentrup, M.; Buss, M. Multiclass common spatial patterns and information theoretic feature extraction. *IEEE Trans. Biomed. Eng.* **2008**, *55*, 1991–2000. [CrossRef]
13. Ang, K.K.; Chin, Z.Y.; Zhang, H.; Guan, C. Filter bank common spatial pattern (FBCSP) in brain-computer interface. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–8 June 2008; pp. 2390–2397.
14. Kang, H.; Nam, Y.; Choi, S. Composite common spatial pattern for subject-to-subject transfer. *IEEE Signal Process. Lett.* **2009**, *16*, 683–686. [CrossRef]
15. Lotte, F.; Congedo, M.; Lécuyer, A.; Lamarche, F.; Arnaldi, B. A review of classification algorithms for EEG-based brain–computer interfaces. *J. Neural Eng.* **2007**, *4*, R1. [CrossRef] [PubMed]
16. Schirrmeister, R.T.; Springenberg, J.T.; Fiederer, L.D.J.; Glasstetter, M.; Eggensperger, K.; Tangermann, M.; Hutter, F.; Burgard, W.; Ball, T. Deep learning with convolutional neural networks for EEG decoding and visualization. *Hum. Brain Mapp.* **2017**, *38*, 5391–5420. [CrossRef] [PubMed]
17. Lawhern, V.J.; Solon, A.J.; Waytowich, N.R.; Gordon, S.M.; Hung, C.P.; Lance, B.J. EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces. *J. Neural Eng.* **2018**, *15*, 056013. [CrossRef] [PubMed]
18. Riyad, M.; Khalil, M.; Adib, A. Incep-EEGNet: A convnet for motor imagery decoding. In Proceedings of the Image and Signal Processing: 9th International Conference, ICISP 2020, Marrakesh, Morocco, 4–6 June 2020; pp. 103–111.
19. Riyad, M.; Khalil, M.; Adib, A. MI-EEGNET: A novel convolutional neural network for motor imagery classification. *J. Neurosci. Methods* **2021**, *353*, 109037. [CrossRef] [PubMed]
20. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
22. Galassi, A.; Lippi, M.; Torroni, P. Attention in natural language processing. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4291–4308. [CrossRef] [PubMed]
23. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
24. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. 2009. Available online: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (accessed on 17 May 2023).
25. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.
26. Li, Y.; Guo, L.; Liu, Y.; Liu, J.; Meng, F. A temporal-spectral-based squeeze-and-excitation feature fusion network for motor imagery EEG decoding. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2021**, *29*, 1534–1545. [CrossRef] [PubMed]
27. Li, H.; Zhang, D.; Xie, J. MI-DABAN: A dual-attention-based adversarial network for motor imagery classification. *Comput. Biol. Med.* **2023**, *152*, 106420. [CrossRef] [PubMed]
28. Takahashi, K.; Sun, Z.; Solé-Casals, J.; Cichocki, A.; Phan, A.H.; Zhao, Q.; Zhao, H.-H.; Deng, S.; Micheletto, R. Data augmentation for Convolutional LSTM based brain computer interface system. *Appl. Soft Comput.* **2022**, *122*, 108811. [CrossRef]
29. Rommel, C.; Paillard, J.; Moreau, T.; Gramfort, A. Data augmentation for learning predictive models on EEG: A systematic comparison. *J. Neural Eng.* **2022**, *19*, 066020. [CrossRef] [PubMed]
30. Klepl, D.; He, F.; Wu, M.; Blackburn, D.J.; Sarrigiannis, P.G. Cross-frequency multilayer network analysis with bispectrum-based functional connectivity: A study of Alzheimer's disease. *Neuroscience* **2023**, *521*, 77–88. [CrossRef] [PubMed]

31. Cao, R.; Hao, Y.; Wang, X.; Gao, Y.; Shi, H.; Huo, S.; Wang, B.; Guo, H.; Xiang, J. EEG functional connectivity underlying emotional valance and arousal using minimum spanning trees. *Front. Neurosci.* **2020**, *14*, 355. [CrossRef] [PubMed]
32. Adebisi, A.T.; Veluvolu, K.C. Brain network analysis for the discrimination of dementia disorders using electrophysiology signals: A systematic review. *Front. Aging Neurosci.* **2023**, *15*, 1039496. [CrossRef]
33. Klepl, D.; Wu, M.; He, F. Graph neural network-based eeg classification: A survey. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2024**, *32*, 493–503. [CrossRef] [PubMed]
34. Xu, L.; Zhang, H.; Hui, M.; Long, Z.; Jin, Z.; Liu, Y.; Yao, L. Motor execution and motor imagery: A comparison of functional connectivity patterns based on graph theory. *Neuroscience* **2014**, *261*, 184–194. [CrossRef]
35. Kılıç, B.; Aydın, S. Classification of contrasting discrete emotional states indicated by EEG based graph theoretical network measures. *Neuroinformatics* **2022**, *20*, 863–877. [CrossRef]
36. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
37. Seo, Y.; Defferrard, M.; Vandergheynst, P.; Bresson, X. Structured sequence modeling with graph convolutional recurrent networks. In Proceedings of the Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, 13–16 December 2018; pp. 362–373.
38. Shen, L.; Sun, M.; Li, Q.; Li, B.; Pan, Z.; Lei, J. Multiscale temporal self-attention and dynamical graph convolution hybrid network for EEG-based stereogram recognition. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2022**, *30*, 1191–1202. [CrossRef]
39. Wu, H.; Niu, Y.; Li, F.; Li, Y.; Fu, B.; Shi, G.; Dong, M. A Parallel Multiscale Filter Bank Convolutional Neural Networks for Motor Imagery EEG Classification. *Front. Neurosci.* **2019**, *13*, 1275. [CrossRef]
40. Bo, D.; Wang, X.; Liu, Y.; Fang, Y.; Li, Y.; Shi, C. A survey on spectral graph neural networks. *arXiv* **2023**, arXiv:2302.05631.
41. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In Proceedings of the Advances in Neural Information Processing Systems 29 (NIPS 2016), Barcelona, Spain, 5–10 December 2016.
42. Brunner, C.; Leeb, R.; Müller-Putz, G.; Schlögl, A.; Pfurtscheller, G. *BCI Competition 2008–Graz Data Set A*; Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology: Graz, Austria, 2008; Volume Voume 16, pp. 1–6.
43. Dornhege, G. *Toward Brain-Computer Interfacing*; MIT Press: Cambridge, MA, USA, 2007.
44. Altaheri, H.; Muhammad, G.; Alsulaiman, M. Physics-Informed Attention Temporal Convolutional Network for EEG-Based Motor Imagery Classification. *IEEE Trans. Ind. Inform.* **2023**, *19*, 2249–2258. [CrossRef]
45. Thanigaivelu, P.S.; Sridhar, S.; Sulthana, S.F. OISVM: Optimal Incremental Support Vector Machine-based EEG Classification for Brain-computer Interface Model. *Cogn. Comput.* **2023**, *15*, 888–903. [CrossRef]
46. Xie, X.; Zou, X.; Yu, T.; Tang, R.; Hou, Y.; Qi, F. Multiple graph fusion based on Riemannian geometry for motor imagery classification. *Appl. Intell.* **2022**, *52*, 9067–9079. [CrossRef]
47. Ni, T.; He, C.; Gu, X. Semi-supervised classifier with projection graph embedding for motor imagery electroencephalogram recognition. *Multimed. Tools Appl.* **2024**, *83*, 14189–14209. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.