EEG Signal Classification for Motor Imagery Tasks Using a Deep Stacking Network Optimized by Particle Swarm Optimization

Xue Zhang
School of Big Data and Artificial Intelligence, Xinyang University, Xinyang, Henan, 464000 China
E-mail: yygts8368@outlook.com

Keywords: EEG signal processing, PSO, deep stacked neural network, motor imagery, brain-computer interface

Received: June 18, 2025

This research proposes a Deep Stacked Network (DSN) model optimized with a particle swarm optimization (PSO) algorithm for classifying electroencephalogram (EEG) signals in motor imagery tasks. Experiments based on the BCI Competition II dataset confirm the effectiveness of the proposed method, which achieves an accuracy of up to 89.42% and outperforms traditional deep models such as DNN and DBN. The study demonstrates that the combination of DSN and PSO enables robust and accurate recognition of brain states, which is important for brain-computer interface devices.

Povzetek: Članek predlaga model Deep Stacked Network (DSN), optimiziran z algoritom roja delcev (PSO) za klasifikacijo signalov elektroencefalograma (EEG) pri nalogah motorične imaginacije.

1 Introduction

As computer technology and signal processing techniques have rapidly developed, an increasing number of scholars have attempted to apply them in the field of neural learning. Electroencephalography (EEG) serves as an important non-invasive method for recording brain activity [1]. By using scalp electrodes, EEG captures the weak electrical signals produced by neuronal activity in the brain, which reflect dynamic changes in the brain under different task-related states. Common EEG rhythms, such as Delta waves, Alpha waves, and Beta waves, are closely associated with states of relaxation and focused attention [2]. Studying the patterns of these signal changes not only helps improve the accuracy of brain-computer interface systems [3] and user experience, but also provides a significant window into understanding brain function mechanisms.

Traditional methods for EEG feature extraction primarily include models such as the AutoRegressive Model (AR), Wavelet Transform, and Common Spatial Pattern (CSP) [4]. These methods extract features from signals in the time domain, frequency domain, or spatial domain through specific algorithms, making them suitable for particular application scenarios. In the classification stage, models such as Linear Discriminant Analysis (LDA), Artificial Neural Networks (ANN), and Support Vector Machines (SVM) [5] are widely employed. Although these methods can achieve effective classification of EEG signals to a certain extent, their feature extraction capabilities are limited by predefined algorithm frameworks and struggle to capture complex dynamic characteristics of the signals.

With the rapid development of Deep Learning technology [6], models such as Deep Belief Networks (DBN) [7], Sparse Autoencoders [8], and Convolutional Neural Networks (CNN) [9] have gradually become important tools for EEG signal analysis. These deep learning models, through multi-layered nonlinear transformations, can automatically extract higher-level abstract features from raw EEG signals, significantly improving classification performance. In recent years, various advanced methods related to deep learning have emerged, further propelling the development of EEG signal analysis. For example, Recurrent Neural Networks (RNN) [10], including variants such as Long Short-Term Memory (LSTM) [11] and Gated Recurrent Units (GRU) [12], are used to capture temporal dynamics in EEG signals. These models can handle long-term dependencies, making them particularly suitable for tasks requiring analysis of temporal characteristics. Graph Convolutional Neural Networks (GCN) [13] model inter-channel relationships in EEG signals based on graph structures, making them suitable for analyzing multi-channel EEG data. GCN can process non-Euclidean data, better representing spatial characteristics of EEG signals. Generative Adversarial Networks (GAN) [14] use adversarial training between a generator and a discriminator to generate high-quality EEG signal samples, making them suitable for data augmentation and signal reconstruction tasks. Attention mechanisms [15], combined with Transformer architectures or self-attention mechanisms, dynamically assign importance to different temporal and spatial features, improving the efficiency and accuracy of feature extraction.

Compared to traditional techniques, these deep learning methods not only have significant advantages in

feature extraction and classification performance, but also reduce manual intervention through end-to-end learning models, providing broader research directions for EEG signal analysis. In the future, designing more efficient and robust EEG signal analysis frameworks by combining multiple deep learning technologies will be one of the key research directions in the field [16].

Table 1: Summary of related works on EEG signal classification

Reference	Method	Dataset
[9]	CNN	BCI Competition II
[11]	LSTM	Epileptic EEG
[13]	GCN	SEED (Emotion)
[16]	DNN	BCI Competition IV
This Work	PSO-DSN	BCI Competition II

Table 1 gives a summary of related works on EEG signal classification. In summary, with the continuous progress of deep learning technology, feature extraction and classification techniques for EEG signals will see new breakthroughs. How to further improve model generalization ability and adaptability based on existing methods will be an important challenge for future research.

This paper proposes a new method for EEG signal feature learning based on the Deep Stacking Network (DSN) and further employs Particle Swarm Optimization (PSO) technology to optimize the hidden layer parameters of the DSN model. Through this approach, we aim to achieve accurate identification and classification of EEG signals for motor imagery tasks. The primary objective of this study is to improve the accuracy of EEG signal classification for motor imagery tasks by developing a hybrid model that combines the representational power of Deep Stacking Networks (DSN) with the global optimization capabilities of Particle Swarm Optimization (PSO). We hypothesize that using PSO to optimize the parameters of the DSN's hidden layers will lead to better feature learning and higher classification accuracy compared to standard deep learning models like DNN and DBN. This hypothesis is tested through comparative experiments on publicly available BCI competition datasets.

2 Method introduction

2.1 Brain-computer interface

The disciplines of brain cognitive science and signal processing technology play pivotal roles within the realm of Brain-Computer Interfaces (BCI), particularly in the acquisition, identification, classification, and feedback control of electroencephalogram (EEG) signals. EEG signals, which are recorded through scalp electrodes as subtle electrical manifestations of neuronal activity within the brain, encompass common brainwave patterns reflecting distinct states of consciousness [2]. To enable the control of external devices, including wheelchairs, prosthetics, or computer interactions, EEG signals undergo preprocessing stages such as filtering and denoising. Subsequently, these signals are subjected to feature extraction—employing temporal, spectral, or time-frequency characteristics—followed by classification through machine learning algorithms. The incorporation of a feedback regulation mechanism significantly enhances the performance of BCI systems, allowing for real-time adjustments to dynamically accommodate the fluctuating states of users' conscious awareness, shown as Figure 1. As human brain activity undergoes continuous variations in alignment with changes in consciousness, the study of these underlying principles holds the potential to elevate the precision and user experience of BCI technologies. Moreover, this research offers a crucial vantage point for the deeper comprehension of brain function.

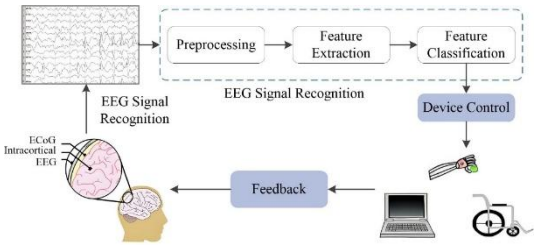

Figure 1: Basic principle of brain-computer interface system.

2.2 Deep stacking networks optimized via PSO

The Deep Stacking Network (DSN), a sophisticated architectural framework, is constituted by the amalgamation of multiple stacked expert layers, each housing a diverse array of experts, which may encompass varied classifiers or regressors. The outputs derived from each expert layer are meticulously amalgamated to produce the final output of the model. Unlike conventional models that rely on predefined parameter structures, DSN adeptly learns the optimal combination and weighting of experts in an adaptive manner based on the data itself. This inherent flexibility endows the model with the capacity to adeptly accommodate disparate data distributions, thereby enhancing its adaptability and performance across a spectrum of datasets [17]. Illustrative representations of the DSN model's intricate structure can be apprehended from Figure 2.

Supposing that each subnet encompasses an input layer, a hidden layer, and an output layer, the output of the hidden layer for the i-th subnet can be articulated as:

$$h_i = f(W_i x + b_i) \tag{1}$$

Herein, x represents the input vector, while W_i signifies the weight matrix mapping from the input layer to the hidden layer. Additionally, b_i denotes the bias vector of the hidden layer, and f stands as the activation function. The ultimate output y_i of the subnetwork can

thus be expressed as

$$y_i = g(U_i h_i + c_i) \tag{2}$$

where, U_i denotes the weight matrix from the hidden layer to the output layer, c_i represents the bias vector of the output layer, and g signifies the activation function of the output layer. The overall output of the deep stacked network is a composite of the outputs from each individual subnetwork, culminating in the ultimate output Y through a weighted summation process.

$$Y = \sum_{i=1}^{n} \alpha_i y_i \tag{3}$$

Among them, α_i represents the weight assigned to the output of the i-th subnetwork, and n signifies the total number of subnetworks. These weights can either be learned through training or predetermined based on empirical insights. It is important to note that this weighted sum combines the final outputs of each module, which are computed in parallel. However, the internal hidden representations are used to enrich the input to subsequent modules.

The loss function during the training process is defined as follows:

$$L = \frac{1}{m} \sum_{j=1}^{m} (Y_j - \hat{Y}_j)^2 \tag{4}$$

Herein, m denotes the quantity of training samples, Y_j represents the actual output of the j th sample, and $\hat{Y}_j$ signifies the network's predicted output. For classification tasks, the cross-entropy loss may be employed.

$$L = -\frac{1}{m} \sum_{j=1}^{m} \sum_{k=1}^{K} y_{jk} \log(\hat{y}_{jk}) \tag{5}$$

Here, K denotes the number of categories, y_{jk} represents the actual label indicating that the j^{th} sample belongs to the k^{th} category, and $\hat{y}_{jk}$ signifies the network's predicted probability that the j^{th} sample belongs to the k^{th} category.

During the training process, the gradient descent algorithm is typically employed to update the network's parameters, with the aim of minimizing the loss function. For the parameter θ, the update formula is

$$\theta = \theta - \eta \frac{\partial L}{\partial \theta} \tag{6}$$

where, η represents the learning rate, which governs the magnitude of each parameter update.

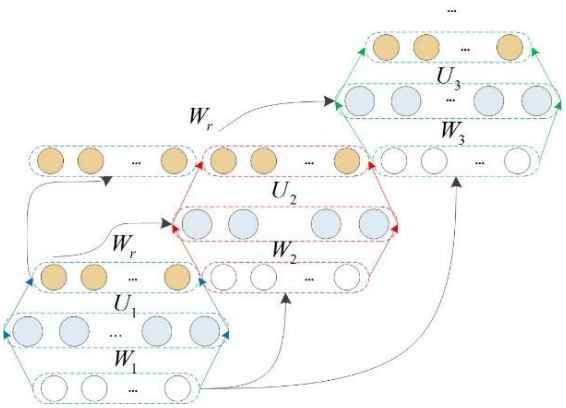

Figure 2: Basic principle of deep stacking networks

This article proposes a deep stacked network comprising two stacked modules, as depicted in Figure 3. The network adopts a modular design consisting of two separately trained modules, each utilizing target outputs for supervised learning. In the design where intermediate hidden layers are partially visible, starting from the second module, a portion of the hidden layer nodes are initialized by replicating the hidden layer nodes trained in the preceding module, and further expanded with a portion of unknown nodes. The hidden layer of each module is initialized using unsupervised pre-training with a Restricted Boltzmann Machine (RBM). This approach increases the complexity and expressive power of the hidden layer [18]. The model regards the hidden layers as intermediate layers rather than final output layers, thereby making the training of hidden layers more flexible. In terms of network architecture, the number of hidden layer nodes increases layer by layer, embodying the concept of partial visibility in hidden layers, which involves inheriting and expanding upon the hidden layers of the previous module. By employing separate training for each module, the computational complexity is equivalent to stacking multiple single-hidden-layer neural networks, thus maintaining flexibility while controlling computational costs.

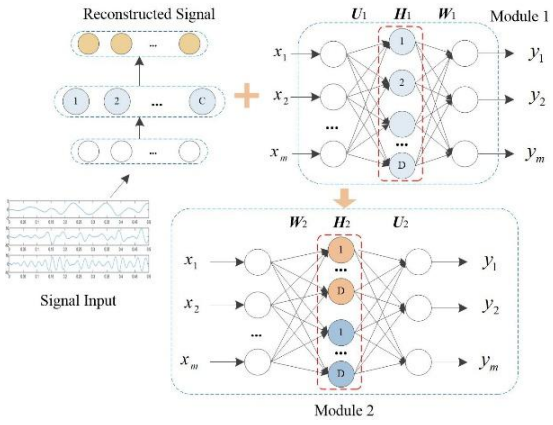

Figure 3: The proposed deep stacked network framework for signal recognition

Particle Swarm Optimization (PSO), as a global optimization algorithm, effectively explores the parameter space of hidden layers, circumventing the local optima issue encountered by traditional gradient descent methods, thereby enhancing the model's generalization capability and predictive performance [19]. The integration of the Particle Swarm Optimization algorithm with the deep stacked network (DSN) for parameter optimization, particularly in the context of visible parameters within hidden layers, represents an innovative approach that marries evolutionary algorithms with deep learning. The primary objective of this methodology is to elevate the performance and generalization capacity of the DSN, leveraging PSO's global search prowess to discover superior parameter configurations, especially where partial visibility of hidden layer parameters is a factor.

The detailed procedure can be observed in Figure 4. Initially, a multi-layered DSN model is constructed, wherein the visible parameters of the hidden layers are encoded into position vectors of PSO particles. Subsequently, the performance metrics of the validation set are defined as the fitness function. By harnessing PSO's global search capabilities, the algorithm iteratively refines the particle positions and velocities until the termination criteria are met.

Within the particle swarm, the update formulae for the particle's velocity and position are as follows:

$$v_i^{(t+1)} = wv_i^{(t)} + c_1 r_1 (p_{best,i} - x_i^{(t)}) + c_2 r_2 (g_{best} - x_i^{(t)}) \quad (7)$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)} \quad (8)$$

where, $x_i^{(t)}$ and $v_i^{(t)}$ denote the position and velocity of particle i at time step t. c_1 and c_2 represent the learning factors, governing the particle's inclination to move towards its own best position, $p_{\text{best},i}$. r_1 and r_2 are uniformly distributed random numbers between [0,1]. g_{best} signifies the historical best position of the entire population.

The final DSN model exhibiting the highest fitness was selected, and its performance was validated on the test dataset, thereby achieving highly efficient and stable parameter optimization. The fitness function is presented as follows.

$$fitness = \frac{1}{2N} \sum_{i=1}^{N} \sum_{j=1}^{m} (y_{ij} - t_{ij})^2 \quad (9)$$

Herein, N denotes the number of training samples, and m represents the count of output nodes; y_{ij} and t_{ij} respectively signify the actual output value and the target output value of the j-th component of the i-th sample.

All models were implemented using Python 3.8 and the PyTorch framework, running on a workstation with an Intel i7-12700K CPU and an NVIDIA GeForce RTX 3080 GPU.

The PSO-DSN training procedure begins by initializing a DSN with M modules and setting PSO parameters including swarm size, learning factors, and maximum iterations. First, layer-wise unsupervised pre-training is performed for each module's hidden layer using Restricted Boltzmann Machines. Then, the PSO optimization phase commences, where each particle's position vector encodes the concatenated weights and biases of the DSN's hidden layers. During each iteration, particles are evaluated by decoding their positions into DSN parameters, performing forward passes on the training set, and calculating fitness using Mean Squared Error. Particle velocities and positions are updated following standard PSO equations, and this process repeats until reaching the maximum iteration count. Finally, the global best particle's position is decoded into the DSN, followed by a supervised fine-tuning step using gradient descent on the entire training set, before evaluating the optimized model on the held-out test set.

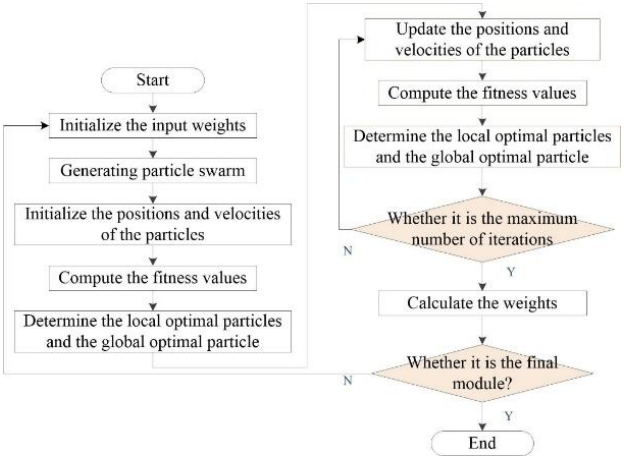

Figure 4: Particle swarm optimization algorithm for parameter optimization

2.3 Deep belief network

Deep Belief Networks (DBNs) [20], a sophisticated deep learning model, are primarily constructed by stacking multiple Restricted Boltzmann Machines (RBMs), as illustrated in Figure 5. For an RBM comprising a visible layer (corresponding to the input layer in Figure 5) and a hidden layer, its energy function is defined as

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i w_{ij} h_j \quad (10)$$

where, a_i represents the bias of the visible unit i, b_j denotes the bias of the hidden unit j, and w_{ij} signifies the connection weight between the visible unit i and the hidden unit j.

Based on the energy function, the joint probability distribution of the RBM is as follows:

$$P(v,h) = \frac{e^{-E(v,h)}}{Z} \quad (11)$$

$$Z = \sum_{v,h} e^{-E(v,h)} \quad (12)$$

where, the normalization constant Z is referred to as the partition function.

Given the hidden layer, the conditional probability of the visible layer is:

$$P(v_i = 1 \mid h) = \sigma(a_i + \sum_j w_{ij} h_j) \quad (13)$$

Given the visible layer, the conditional probability of the hidden layer is

$$P(h_j = 1 \mid v) = \sigma(b_j + \sum_i v_i w_{ij}) \quad (14)$$

where, σ denotes the sigmoid function.

During the training of deep belief networks, a combined approach is typically employed, integrating layer-wise unsupervised pre-training with supervised fine-tuning. For Restricted Boltzmann Machines (RBMs), the Contrastive Divergence (CD) algorithm is utilized to approximate the gradient calculation, facilitating the updating of weights and biases.

$$\Delta w_{ij} = \alpha([v_i h_j]_{\text{data}} - [v_i h_j]_{\text{recon}}) \quad (15)$$

$$\Delta a_j = \alpha([h_j]_{\text{data}} - [h_j]_{\text{recon}}) \quad (16)$$

$$\Delta b_i = \alpha([v_i]_{\text{data}} - [v_i]_{\text{recon}}) \quad (17)$$

Here, $[]_{\text{data}}$ and $[]_{\text{recon}}$ represent the average probabilities of corresponding unit activations over the real data and reconstructed data, respectively; α denotes the learning rate.

Upon the conclusion of pretraining, the output from the hidden layer of the final RBM is connected to an output layer (e.g., the Output layer), and supervised fine-tuning is performed using the Backpropagation algorithm [21].

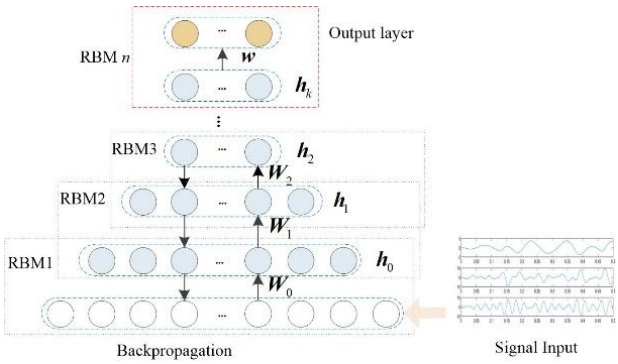

Figure 5: Structure of deep belief network model

3 Experiment and analysis

3.1 Data set and evaluation index

The dataset employed in this experiment is the BCI Competition II Data Set III and IV, which are pivotal components of the second iteration of the Brain-Computer Interface (BCI) Competition. Each of these datasets involves distinct subjects and experimental designs. Data Set III encompasses the EEG recordings of a 25-year-old healthy male, captured using a 118-channel BioSemi ActiveTwo system with a sampling rate of 256 Hz. The experimental tasks involved the imagination of movements of the left hand, right hand, or feet, conducted over two separate sessions on different days. Data Set IV, on the other hand, features the data from a patient suffering from Amyotrophic Lateral Sclerosis (ALS), recorded via a 64-channel EEG system at a sampling rate of 128 Hz. Similar to Data Set III, the experimental tasks in Data Set IV also involved motor imagery, particularly focusing on imagining movements of the left hand, right hand, or feet.

During the experimental process, the raw EEG data underwent preprocessing to mitigate noise and extract salient features. This preprocessing phase involved the application of a band-pass filter to eliminate low-frequency drift and high-frequency noise, segmentation of data into distinct trial segments based on markers, and the removal of electrooculographic (EOG) artifacts and other spurious signals. The experimental framework culminated in the identification and classification of EEG signals through the integration of a Deep Stacking Network (DSN) and Particle Swarm Optimization (PSO). Data were split into 70% training, 15% validation, and 15% test sets. We performed subject-dependent evaluation using 5-fold cross-validation.

3.2 Parameter configuration

When employing Restricted Boltzmann Machines (RBMs) for feature learning, the judicious setting of hyperparameters is pivotal to the model's performance. In this study, the initial values for the learning rate and penalty factor were set at 0.001, incorporating a decay strategy to progressively enhance the model's convergence. The momentum learning rate commenced from a minimal value of 0.0, gradually increasing linearly to 1.0 over 100 epochs, and was further refined through cross-validation to harmonize the model's complexity with its performance. As illustrated in Figure 6, the selected RBM hyperparameters were configured with an initial learning rate of 0.1, a momentum learning rate of 0.6, and a penalty factor of 0.05. Through the judicious arrangement of these parameters, the reconstruction error was mitigated, effectively augmenting the RBM's efficacy in feature learning tasks. Additionally, the parameter settings for the Particle Swarm Optimization algorithm are detailed in Table 2.

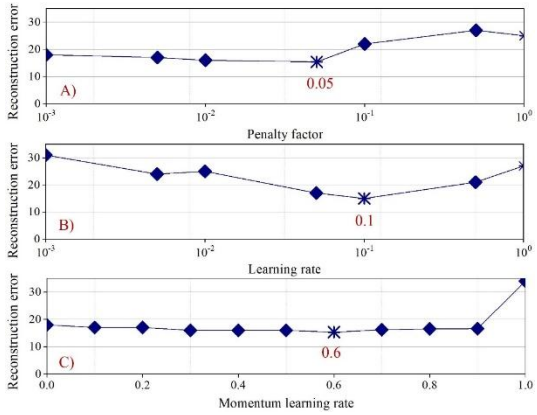

Figure 6: Reconstruction errors of the three RBM parameters

Table 2: Parameter configuration for the particle swarm optimization algorithm

Parameters	Value
Swarm Size	10
Learning Factor	2
Position Range	[-1,1]
Velocity Range	[-0.5,0.5]

The PSO parameters were selected based on a trade-off between computational efficiency and optimization performance. A small swarm size of 10 was chosen to reduce the computational overhead of fitness evaluation, which involves training a DSN module. The standard learning factor value of 2.0 is widely used in the PSO literature [19] to balance the influence of the particle's personal best and the swarm's global best, promoting stable convergence. While larger swarms may offer more exhaustive search, preliminary experiments showed diminishing returns beyond a size of 10 for this specific problem.

3.3 Evaluation of recognition effect

To verify the efficacy of the proposed method, the model was employed in the recognition of brain electrical signals associated with left- and right-hand motor imagery. The number of training samples was varied from 100 to 240 to analyze the model's scalability and data efficiency. As shown in Table 3, with the increase in the number of training samples, the training time progressively extended, in parallel with a rise in recognition accuracy. Specifically, when the training samples increased from 100 to 240, the training time grew from 9.86 seconds to 20.27 seconds, while the recognition accuracy improved from 78.5% (±1.8%) to 89.42% (±1.2%) (results reported as mean ± standard deviation over 5-folds). This phenomenon suggests that a larger volume of training data aids in enhancing the model's generalization capability, thereby achieving a higher recognition accuracy, although it concurrently results in an extended training duration.

Moreover, the data illustrates that with the augmentation of sample numbers, the margin by which accuracy improves gradually diminishes. For instance, the accuracy increased by approximately 4.09% when the samples grew from 100 to 130; yet, from 210 to 240 samples, the improvement was merely around 1.39%. This may indicate the existence of a saturation point, beyond which additional training samples contribute less significantly to accuracy enhancement.

Additionally, taking into account the unique nature of electroencephalogram (EEG) data, such as signal non-stationarity and individual variability, utilizing a larger number of training samples facilitates the model in more effectively capturing these characteristics, thus enhancing its robustness. However, an excessively prolonged training time could constrain the model's applicability in real-time scenarios. To provide a more detailed performance analysis, class-wise precision, recall, and F1-score were calculated for the best-performing model (5 modules, 240 samples). The model achieved an average F1-score of 0.89, with scores of 0.90 for 'Left Hand' imagery and 0.88 for 'Right Hand' imagery, indicating balanced performance across the two classes.

Table 3: Impact of training samples on recognition accuracy and training time

Samples	Training Time (s)	Accuracy (%)
100	9.86	78.5 ± 1.8
130	10.14	83.51 ± 1.5
160	12.50	85.09 ± 1.4
190	17.93	87.67 ± 1.3
210	18.84	88.03 ± 1.3
240	20.27	89.42 ± 1.2

3.4 Comparative experiment

This study applies the PSO-optimized hidden layer parameter approach to the DNN, DBN, and DSN deep learning models, conducting comparative experiments. Four subjects, namely S1-4, were selected from the dataset. Figure 7 illustrates the disparity in the motor imagery recognition performance of these models across the subjects. To ensure a fair comparison, all baseline models were configured with standard settings: DNN used a single hidden layer (256 units) with the Adam optimizer; DBN employed a two-layer RBM structure (256-128 units) with backpropagation fine-tuning; DSN adopted a two-module architecture (128-256 nodes) incorporating RBM pre-training; and PSO-DSN enhanced the DSN with PSO optimization.

According to Figure 7, as the number of stacked modules increases, the models exhibit distinct trends in recognition accuracy, closely related to their respective characteristics. The DNN model demonstrated relatively

inferior performance in this experiment, particularly under a high number of stacked modules, where the improvement in recognition accuracy was minimal. Therefore, the DNN is more suitable for handling simpler tasks or datasets with more distinct features. Secondly, the DBN model showcased a more stable performance in this experiment, with a notably significant increase in recognition accuracy at moderate numbers of stacked modules. However, as the number of stacked modules grows, the improvement in the DBN's recognition accuracy diminishes gradually, indicating potential issues of overfitting and increased training complexity at higher module counts.

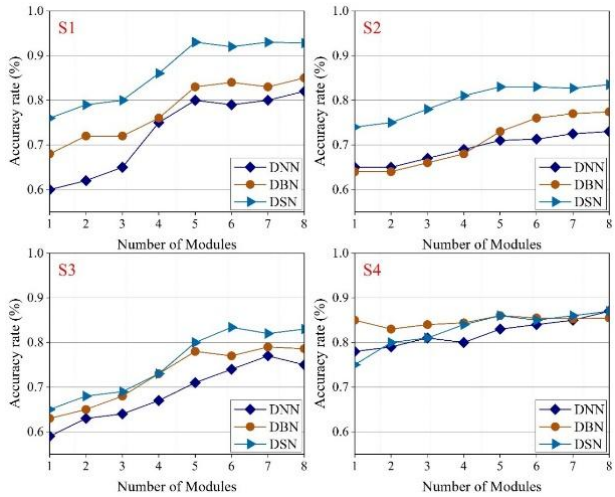

Figure 7: Recognition accuracy of various deep learning models across four subjects

The DSN model demonstrated superior performance in this experiment, particularly under configurations with a higher number of stacked modules, where its recognition accuracy significantly surpassed that of the DNN and DBN models. A key feature of the DSN model is the incorporation of a supervised learning mechanism, which optimizes model parameters through layer-wise supervised training, thereby enhancing its ability to capture intricate patterns in the data. This advantage renders the DSN model particularly effective in handling complex tasks, such as EEG signal recognition, especially when the number of modules is high, leading to a notable increase in recognition accuracy. Additionally, the DSN model exhibits greater stability under high module counts, indicating a robust resistance to overfitting. Consequently, the DSN model is better suited for tasks demanding high precision and complexity.

Experimental data suggests that 4-5 stacked modules represent the optimal configuration for most models, with the DSN model achieving peak recognition accuracy at 5 modules. As the number of stacked modules increases, the overall recognition accuracy tends to rise, though the rate of improvement diminishes, signaling the presence of a saturation point. Beyond this point, additional stacked modules contribute marginally to accuracy enhancement while significantly increasing training time and computational complexity. Therefore, in practical applications, the selection of an appropriate number of stacked modules necessitates a balance between recognition accuracy and computational resources. Moreover, an increase in the number of stacked modules substantially elevates model complexity, leading to longer training and inference times. This could pose challenges for tasks requiring real-time performance, such as real-time control in brain-computer interfaces. Thus, while models with a higher number of stacked modules offer superior recognition accuracy, their practical implementation must account for real-time performance requirements.

4 Discussion

This study proposed a PSO-optimized DSN model for EEG signal classification. The results demonstrate that our method achieves a competitive accuracy of 89.42% on the BCI Competition II Dataset III, which compares favorably with state-of-the-art methods such as CNNs, LSTMs, and GCNs reported in the literature for similar motor imagery tasks [9,11,13].

The superior performance of the DSN model, particularly with a higher number of stacked modules, can be attributed to two main factors. First, the layer-wise supervised training mechanism allows each module to learn discriminative features directly relevant to the classification task, mitigating the vanishing gradient problem common in very deep networks. Second, the integration of PSO provides a global search strategy for optimizing the visible parameters of the hidden layers, helping the model escape local optima that might trap gradient-based methods. This is especially beneficial for the DSN architecture, where the interaction between modules can create a complex optimization landscape.

However, this performance comes at a computational cost. As shown in Table 3, training time increases with more samples and modules. While PSO-DSN was more accurate than DBN and DNN, its training time was longer due to the PSO iteration process. This presents a trade-off between accuracy and computational efficiency, which must be considered for real-time BCI applications. Future work could focus on developing more efficient hybrid optimization strategies.

A limitation of this study is the use of a relatively small swarm size in PSO. While it was chosen for computational tractability, exploring adaptive swarm sizes or other meta-heuristic algorithms could yield further improvements. Furthermore, the study focused on subject-dependent analysis; evaluating the model's cross-subject generalization capability remains an important area for future research.

5 Conclusion

This paper introduces a method for EEG-based motor imagery classification, based on a neural network optimized by Particle Swarm Optimization. This model integrates the deep network architecture of DSN with the PSO optimization algorithm, combining unsupervised learning with supervised training to enhance the model's capability in extracting features from EEG signals. The study demonstrates that, through a layer-wise learning mechanism, it effectively captures high-order features of EEG signals, laying a solid foundation for precise identification and classification. Comparative analysis reveals that the DSN model outperforms other models in tasks related to EEG signal recognition, particularly when using a higher number of stacked modules, where its recognition accuracy significantly surpasses that of DNN and DBN. The supervised learning mechanism of the DSN model enables it to better grasp complex characteristics of the data, exhibiting strong resistance to overfitting. However, as the number of stacked modules increases, the model's training time and complexity rise significantly, necessitating a balance between recognition accuracy and computational resources. Future research could focus on optimizing the structure and training methods of the DSN model to enhance its performance and efficiency in practical applications, including exploring its cross-subject robustness and applicability to other EEG paradigms. Overall, this approach offers new opportunities for EEG signal analysis and holds great potential for deep application and expansion in fields such as brain-computer interfaces, continuously propelling interdisciplinary research between neuroscience and engineering technology to new heights.

References

[1] McFarland D J, Wolpaw J R. EEG-based brain–computer interfaces[J]. current opinion in Biomedical Engineering, 2017, 4: 194-200.doi: 10.1016/j.cobme.2017.11.004

[2] Wu D, King J T, Chuang C H, et al. Spatial filtering for EEG-based regression problems in brain–computer interface (BCI)[J]. IEEE Transactions on Fuzzy Systems, 2017, 26(2): 771-781.doi: 10.1109/TFUZZ.2017.2688423

[3] Hwang H J, Kim S, Choi S, et al. EEG-based brain-computer interfaces: a thorough literature survey[J]. International Journal of Human-Computer Interaction, 2013, 29(12): 814-826.doi: 10.1080/10447318.2013.780869

[4] Amin H U, Malik A S, Ahmad R F, et al. Feature extraction and classification for EEG signals using wavelet transform and machine learning techniques[J]. Australasian physical & engineering sciences in medicine, 2015, 38(1): 139-149.doi: 10.1007/s13246-015-0333-x

[5] Aggarwal S, Chugh N. Review of machine learning techniques for EEG based brain computer interface[J]. Archives of Computational Methods in Engineering, 2022, 29(5): 3001-3020.doi:10.1007/s11831-021-09703-6

[6] Altaheri H, Muhammad G, Alsulaiman M, et al. Deep learning techniques for classification of electroencephalogram (EEG) motor imagery (MI) signals: A review[J]. Neural Computing and Applications, 2023, 35(20): 14681-14722.doi: 10.1007/s00521-021-06352-5

[7] Movahedi F, Coyle J L, Sejdić E. Deep belief networks for electroencephalography: A review of recent contributions and future outlooks[J]. IEEE journal of biomedical and health informatics, 2017, 22(3): 642-652.doi: 10.1109/JBHI.2017.2727218

[8] Qiu Y, Zhou W, Yu N, et al. Denoising sparse autoencoder-based ictal EEG classification[J]. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2018, 26(9): 1717-1726. doi: 10.1109/TNSRE.2018.2864306

[9] Rajwal S, Aggarwal S. Convolutional neural network-based EEG signal analysis: A systematic review[J]. Archives of Computational Methods in Engineering, 2023, 30(6): 3585-3615. doi: 10.1007/s11831-023-09920-1

[10] Übeyli E D. Analysis of EEG signals by implementing eigenvector methods/recurrent neural networks[J]. Digital Signal Processing, 2009, 19(1): 134-143.doi: 10.1016/j.dsp.2008.07.007

[11] Tsiouris K M, Pezoulas V C, Zervakis M, et al. A long short-term memory deep learning network for the prediction of epileptic seizures using EEG signals[J]. Computers in biology and medicine, 2018, 99: 24-37.doi:10.1016/j.compbiomed.2018.05.019

[12] Prakash V, Kumar D. A modified gated recurrent unit approach for epileptic electroencephalography classification[J]. Journal of Information and Communication Technology, 2023, 22(4): 587-617.doi: 10.32890/jict2023.22.4.3

[13] Du G, Su J, Zhang L, et al. A multi-dimensional graph convolution network for EEG emotion recognition[J]. IEEE Transactions on Instrumentation and Measurement, 2022, 71: 1-11.doi: 10.1109/TIM.2022.3204314

[14] Fahimi F, Dosen S, Ang K K, et al. Generative adversarial networks-based data augmentation for brain–computer interface[J]. IEEE transactions on neural networks and learning systems, 2020, 32(9): 4039-4051.doi:10.1109/TNNLS.2020.3016666

[15] Feng L, Cheng C, Zhao M, et al. EEG-based emotion recognition using spatial-temporal graph convolutional LSTM with attention mechanism[J]. IEEE Journal of Biomedical and Health Informatics, 2022, 26(11): 5406-5417.doi: 10.1109/JBHI.2022.3198688

[16] Dose H, Møller J S, Iversen H K, et al. An end-to-end deep learning approach to MI-EEG signal classification for BCIs[J]. Expert Systems with

Applications, 2018, 114: 532-542.doi: 10.1016/j.eswa.2018.08.031

[17] Palangi H, Ward R, Deng L. Convolutional deep stacking networks for distributed compressive sensing[J]. Signal Processing, 2017, 131: 181-189.doi: 10.1016/j.sigpro.2016.07.006

[18] Zhang N, Ding S, Zhang J, et al. An overview on restricted Boltzmann machines[J]. Neurocomputing, 2018, 275: 1186-1199. doi: 10.1016/j.neucom.2015.03.026

[19] Han F, Yao H F, Ling Q H. An improved evolutionary extreme learning machine based on particle swarm optimization[J]. Neurocomputing, 2013, 116: 87-93.doi:10.1016/j.neucom.2011.12.062

[20] Zambra M, Testolin A, Zorzi M. A developmental approach for training deep belief networks[J]. Cognitive Computation, 2023, 15(1): 103-120.doi:10.1007/s12559-022-10085-5

[21] Nahid A A, Mikaelian A, Kong Y. Histopathological breast-image classification with restricted Boltzmann machine along with backpropagation[J]. Biomedical Research, 2018, 29(10): 2068-2077.doi:10.4066/biomedicalresearch.29-17-3903