TYPE Original Research
PUBLISHED 07 April 2025
DOI 10.3389/fsens.2025.1548729

Check for updates

OPEN ACCESS

EDITED BY
Muhammad Zia Ur Rehman,
Aalborg University, Denmark

REVIEWED BY
Muhammad Farrukh Qureshi,
Namal College, Pakistan
Saadullah Farooq Abbasi,
University of Birmingham, United Kingdom

*CORRESPONDENCE
Zhe Li,
✉ kenan_zhentanc@yeah.net

RECEIVED 20 December 2024
ACCEPTED 24 March 2025
PUBLISHED 07 April 2025

CITATION
Li Z (2025) Mamba with split-based pyramidal convolution and Kolmogorov-Arnold network-channel-spatial attention for electroencephalogram classification.
Front. Sens. 6:1548729.
doi: 10.3389/fsens.2025.1548729

Mamba with split-based pyramidal convolution and Kolmogorov-Arnold network-channel-spatial attention for electroencephalogram classification

Zhe Li[1,2]*

[1]Hankou University, Jiangxiazhen, China, [2]Wuhan University of Engineering Science, Wuhan, China

Deep learning is widely used in brain electrical signal studies, among which the brain–computer interface is an important direction. Deep learning can effectively improve the performance of BCI machines, which is of great medical and commercial value. This paper introduces an efficient deep learning model for classifying brain electrical signals based on a Mamba structure enhanced with split-based pyramidal convolution (PySPConv) and Kolmogorov-Arnold network (KAN)-channel-spatial attention (KSA) mechanisms. Incorporating KANs into the attention module of the proposed KSA-Mamba-PySPConv model better approximates the sample function while obtaining local network features. PySPConv, on the other hand, swiftly and efficiently extracts multi-scale fusion features from input data. This integration allows the model to reinforce feature extraction at each layer in Mamba's structure. The model achieves a 96.76% accuracy on the eegmmidb dataset and demonstrates state-of-the-art performance across metrics such as the F1 score, precision, and recall. KSA-Mamba-PySPConv promises to be an effective tool in electroencephalogram classification in brain–computer interface systems.

KEYWORDS

mamba, Kolmogorov-Arnold network, electroencephalogram, deep learning, BCI

1 Introduction

Electroencephalogram (EEG) has been a hotspot for medical, computer science, and commercial research, and it is often used to diagnose brain diseases and to study human mental activities. Brain–computer interfacing is a promising technology for scientists and engineers, which converts human EEG signals into programs that machines can recognize. Deep learning is an important tool in brain electricity studies and has been employed by many scholars, as exemplified below.

The DeepConvnet model, employed by Schirrmeister et al., achieved a classification accuracy of 76.7% on the PhysioNet EEG Motor Movement/Imagery Database (eegmmidb). The model utilized multiple layers of conventional convolutional neural network (CNN) convolutions to construct deep learning networks, achieving 63.7% and 83.2% accuracy on the BCI-2a and BCI-2b datasets, respectively (Schirrmeister et al., 2017). The EEGnet model, adopted by Lawhern et al., achieved an accuracy of 79.3% on the eegmmidb dataset (Lawhern et al., 2018). This model utilized depth-separable convolutions to construct multi-layer deep neural networks, thereby facilitating the segregation of channels and regions while reducing the parameter count. Jia et al. developed a multi-branch multi-scale CNN (MMCNN) (Jia et al., 2021) that decoded the original EEG signal without filtering or other pre-processing techniques. It also successfully characterized information in various frequency bands and thus determined the optimal convolution scale. Roots et al. introduced a multi-branch two-dimensional (2D) CNN that employed distinct hyperparameter values for each branch, resulting in accuracies of 84.1% and 83.8% when applied to the eegmmidb dataset for performing and imagining motor actions, respectively (Roots et al., 2020). Chowdhury et al. developed an EEGNet Fusion V2 model that enhanced the extracted features via diverse filters, which yielded a spectrum of features. Subsequently, these features were integrated into the fusion layer to generate more intricate features (Chowdhury et al., 2023). To identify spectral features and improve the decoding of motor imagery electroencephalogram (MI-EEG), Li et al. employed a novel time-spectrum squeezed-excitation feature fusion network with multi-stage wavelet convolutions in parallel for multi-spectral convolution block capture (Li et al., 2021). Hou et al. combined bidirectional long and short-term memory (BiLSTM), attention mechanisms, and a graph convolutional neural network (GCN) to enhance the decoding performance. This was achieved by leveraging the feature topology estimated from the comprehensive data set to accurately identify the human body's intention to move from the raw EEG signals (Hou et al., 2022). Steady-state visual evoked potential (SSVEP) represents one of the most frequently utilized control signals in brain–computer interface systems. In an interdisciplinary classification scenario, Chen et al. proposed an SSVEP classification model based on the highly effective deep learning Transformer structure, which fully exploited harmonic information and established a methodology based on filter bank technology (Chen et al., 2022). Luo et al. employed a shallow mirror Transformer comprising a multi-head self-attentive layer with a global receptive field to detect and utilize discriminative segments across input EEG trials. They also constructed mirror EEG signals and mirror network structures based on integrated learning to improve classification accuracy (Luo et al., 2023). Keutayeva and Abibullaev developed a hybrid model that fused a CNN with a visual Transformer for decoding motion image EEG signals. The CNN was employed to extract local features, whereas the Transformer was utilized to perceive global dependencies. The model demonstrated 80.44% and 74.73% accuracy on the BCI-2a and BCI-2b datasets, respectively, which represented a significant improvement over previous models (Keutayeva and Abibullaev, 2023).

In recent years, there has been a notable increase in the popularity of Kolmogorov-Arnold networks (KANs) as an alternative to the multi-layer perceptron (MLP) (Vaca-Rubio et al., 2024). KANs utilize the Kolmogorov-Arnold representation theorem, which enables the activation functions of a neural network to be executed on edges. This facilitates the "learning" of the activation functions and enhances the model performance. KANs lack linear weights; each weight parameter is replaced by a univariate function parameterized as a spline. Smaller KANs can be visualized intuitively and achieve comparable or superior accuracies in data fitting and partial differential equation (PDE) solutions compared to larger MLPs.

The Mamba model addresses the limited efficiency of Transformers in long sequence processing by combining linear layers, gating, and selective structured state space models (Gu et al., 2023); its core is a selectivity mechanism that efficiently compresses and filters contextual information. The hardware algorithm significantly improves computational speed by scanning rather than convolving.

However, the mamba model is very limited to handle local features, and the obtained features have a large redundancy. Therefore, we can use some new methods to improve the local feature extraction ability of mamba, using the attention module to screen the reinforcement main features. In light of the studies above, we propose a novel deep learning model integrating a Mamba backbone splicing a split-based pyramidal convolution (PySPConv) module and a KAN-channel-spatial attention (KSA) mechanism. The model is designated as KSA-Mamba-PySPConv, and its objective is to leverage the Mamba and KAN architectures to enhance model classification capabilities and reduce resource costs.

It includes the literature review, model methodology, experimental design, discussion of experiments, and conclusion. The primary contributions of this paper are as follows:

1) We propose the novel KSA mechanism, which incorporates a KAN network into the attention mechanism. This integration aims to enhance the module's feature extraction capabilities, leveraging the fitting approximation capacity of KANs.
2) We employ the novel PySPConv scheme to replace the standard convolutions in Mamba, aiming to address the limitations of Mamba's local feature extraction capability while minimizing computational overhead.
3) We conduct experiments using the proposed KSA-Mamba-PySPConv model and multiple existing models on eegmmidb to compare their accuracy, F1 score, and recall.

2 Related work

2.1 KANs

MLPs use a multi-layer linear function plus a nonlinear activation function to model and approximate the input-output relationships of a sample, which consists of a large trainable data matrix, as expressed by $f(\mathbf{x}) \approx \sum_{i=1}^{N(c)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$.In this formuna, x is the input data,w$_\mathrm{i}$ is the weight of x, b$_\mathrm{i}$ is the bias, σ is activation function, a$_\mathrm{i}$ is the ratio of Scale coefficient. MLPs consume a lot of memory and computational resources in complex tasks and are prone to overfitting. In recent studies, KANs have been shown to outperform MLPs in accuracy and interpretability, with smaller KANs achieving accuracy comparable to or better than larger MLPs in data fitting and PDE solving. Besides, KANs can be represented intuitively as the summation of multiple spline functions and, therefore, have stronger interpretability. The functional relationship of KANs can be expressed by

$$f(\mathbf{x}) = \sum_{q=1}^{Q} \Phi_q \left(\sum_{p=1}^{P} \phi_{q,p}(x_p) \right).$$

Here, $f(\mathbf{x})$ is the multivariate function to be represented; Φ_q denotes a combinatorial function that can be learned and is typically used at higher network levels; $\phi_{q,p}$ is the learnable unitary function corresponding to an activation function on the network edges, generally parameterized as a spline function; x_p is the pth component of the input vector; Q and P are the number of combinatorial and unitary functions, respectively.

KANs can normally achieve comparable or better performance than wider MLPs with fewer parameters. However, KANs have more parameters than MLPs for the same depth and width. The training process of KANs is much more complex than that of conventional neural networks, and its training speed is 10 times slower than that of MLPs. In practice, the resources consumed by KANs are huge and difficult to implement in high-dimensional spaces. Therefore, leveraging the advantages of KANs while compensating for their shortcomings is a question worth studying.

2.2 Mamba and transformers

Transformers capture global features more efficiently than CNNs, though with n^2 computational complexity, n is the length of sequence. In contrast, Mamba exhibits linear complexity and can address the memory consumption issue of Transformers when processing long sequences. Moreover, Mamba adopts hardware-aware parallel algorithms to optimize graphics processing unit (GPU) memory usage and improve the design of the state space model (SSM) architecture, which achieves higher efficiency. Mamba also performs selective processing of input information, which means it can focus on specific information in the input sequence. As a result, Mamba is five times faster than Transformers in inference (predicting or generating texts), and its performance can match that of a Transformer twice its size in certain areas. However, Mamba uses complex S6 and MLP components, making the model complicated and less interpretable. Moreover, it is weak for local feature extraction of sequences. Improving the local feature extraction capability and simplifying the complexity of Mamba are research areas worthy of investigation.

2.3 Pyramidal convolution and split-based convolution

Pyramidal Convolution (PyConv) utilizes a pyramid structure with different kernel sizes and depths to capture details on various levels (Duta et al., 2021). The PyConv architecture has multiple levels of kernels, gradually increasing kernel size from the bottom (level 1) to the top (level n) while reducing depth. This approach aims to capture diverse scale information at different layers. In PyConv, the different kernel types complement each other to enhance the network's recognition capabilities. Smaller kernels excel at focusing on fine details, capturing information about small objects or specific regions; larger kernels gather a more robust overview of larger objects or contextual information. The PyConv architecture exhibits parameter and computational resource requirements comparable to conventional convolution while benefiting from its ability to leverage multi-threaded parallel processing. This configuration makes PyConv exceptionally efficient. PyConv's 50-layer network outperformed a baseline ResNet with 152 layers in recognition performance while reducing the number of parameters by 2.39 times, computational complexity by 2.52 times, and layer count by over three times (Duta et al., 2021).

Split-based convolution (SPConv) splits the input feature map into a representative part and an uncertain redundant part (Zhang et al., 2020). The representative part is processed with relatively heavy computation to extract intrinsic information, while the uncertain redundant part uses lightweight operations to handle tiny details. The SPConv architecture employs a k = 3 convolutional layer to extract essential information and a lightweight k = 1 convolutional layer to supplement fine-grained hidden details. The final step of the process involves merging the extracted features from the two parts using a parameter-free feature fusion module. Therefore, SPConv consistently outperforms baselines in accuracy and inference time while showing significant reductions in floating-point operations per second (FLOPs) and parameter counts. Experiments on Cifar10, ImageNet, and Microsoft Common Objects in Context (MS-COCO) datasets demonstrated that networks using SPConv achieved state-of-the-art (SOTA) performance in accuracy and inference speed at the GPU level. The parameter count for SPConv could also be reduced by 2.8 times while maintaining superior performance and inference speed (Zhang et al., 2020).

3 Methodology

3.1 KSA-seq attention

EEG involves multichannel one-dimensional (1D) data, for which the features of individual channel waveforms and the potential relationships between different channels must be considered. KAN has a stronger fitting ability than MLP, and it can obtain the waveform features of a single channel better. On the other hand, the attention mechanism of the lateral axis captures the feature relationships between different channels. We fuse the attention mechanisms in both directions to obtain a more comprehensive EEG feature relationship. We call this attention mechanism KSA, and the algorithmic steps for realizing the KSA attention mechanism are as follows in Algorithm 1:

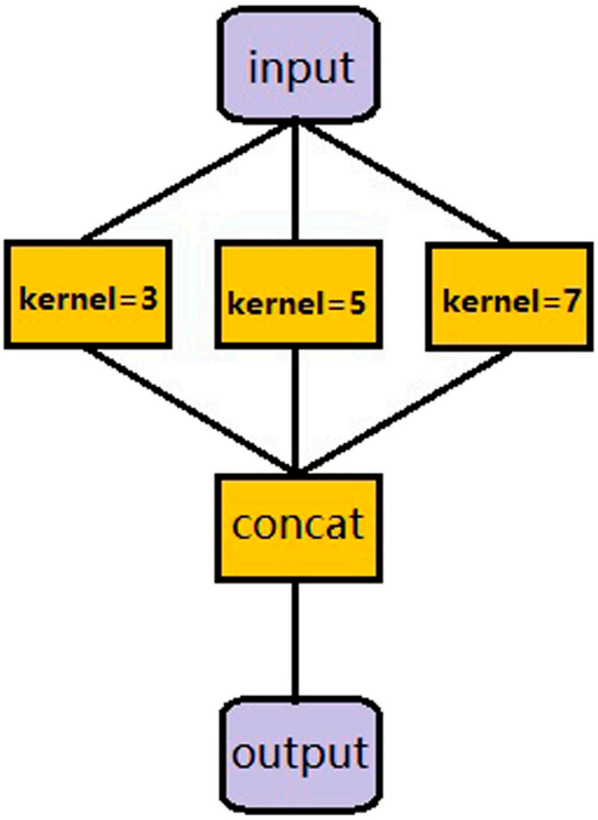


FIGURE 1
Structure of mixconv1d.

```
Input:x: (B,C,L)
Output: y: (B,C,L)
        1. r1: (B,C,1) ← AdaptiveAvgPool1d (B,C,L)
        2. r2: (B,C,1) ← AdaptiveMaxPool1d (B,C,L)
        3. r1′: (B,C,1) ← KAN_Expand (KAN_Compress (r1′))
        4. r2′: (B,C,1) ← KAN_Expand (KAN_Compress (r2′))
        4. m: (B,C,L) ← (r1′+r2′) *x
         ▷ KAN-seq attention is done
        1. m: (B,C,L)
        2. n1: (B,1,L) ← Mean (m,dim = 1)
        3. n2: (B,1,L) ← Max (m,dim = 1)
        4. n: (B,2,C) ← Concat (n1,n2,dim = 1)
        4. SpatialAtt: (B,1,C) ← conv1d (B,2,C), kernel = 7
        5. out: (B,C,L) ← SpatialAtt *m
         ▷ SP1D-seq attention is done
Return out
```

Algorithm 1. KAN-seq attention.

The algorithm describes the implementation of the attention mechanism and the main input and output data. KAN_Compress represents using the KAN-MLP architecture to compress channels, which is achieved by setting the number of output channels Cout of the KAN network to be 1/r of the number of input channels Cin (where r is the scaling ratio). KAN_Expand represents using the KAN-MLP architecture to expand channels, which is accomplished by configuring the number of output channels Cout to expand to r times the number of input channels Cin. We obtain channel-axis

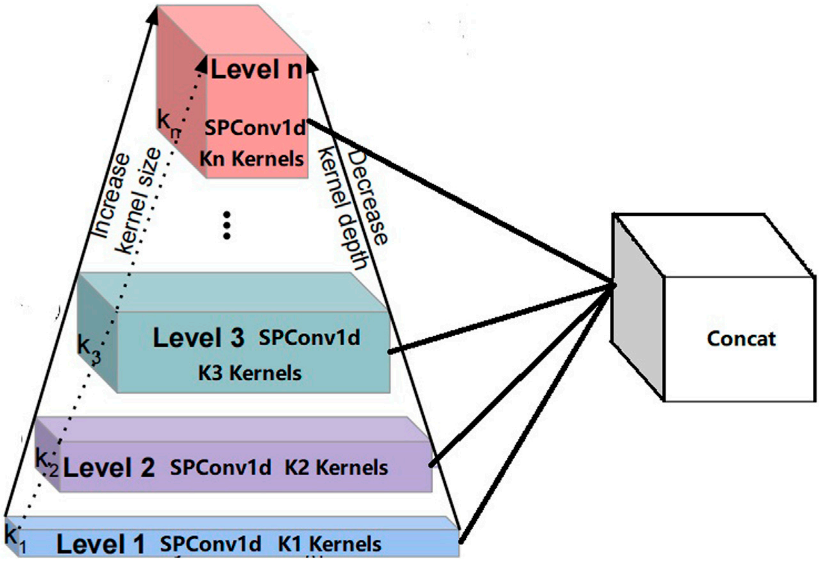


FIGURE 2
Structure of PySPConv.

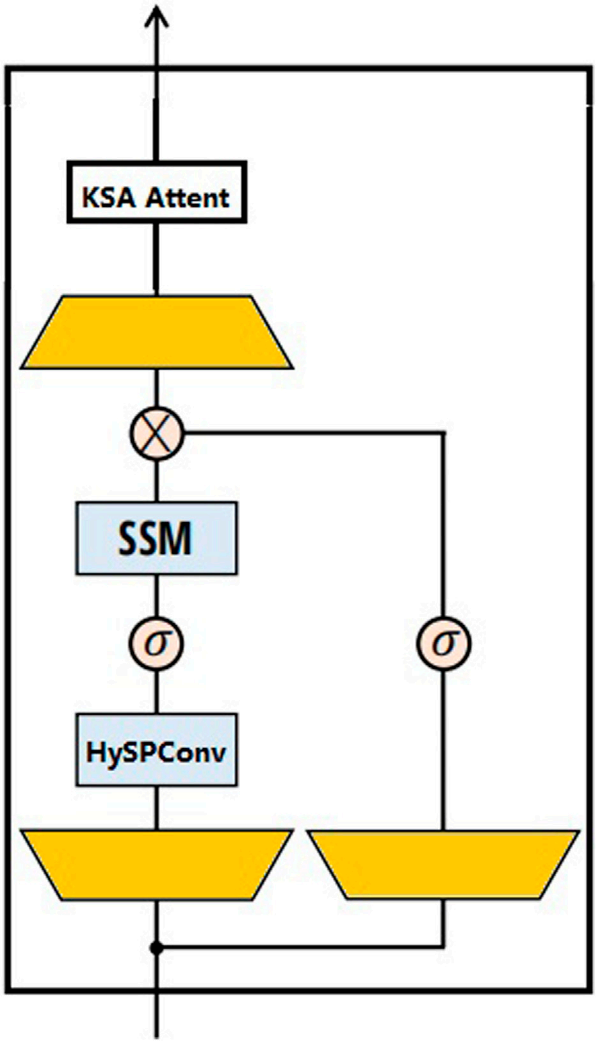


FIGURE 3
Structure of the KSA-Mamba-PySPConv block.

attention by performing channel transformation with KAN-seq attention. Then, we use the 1D spatial attention algorithm to calculate the spatial sequence features, which can obtain the y-axis attention. By combining the operations of both sections, we obtain multi-dimensional attention for the input sequence data. We call this module KSA-seq attention.

3.2 PySPConv

Mamba is a novel selective structured SSM that can efficiently deal with long sequential data while maintaining linear time complexity. In Mamba modules, a conventional CNN is used to extract local features. However, CNN-extracted features are limited by the size of kernels and the number of layers, which means that CNN suffers from a lack of flexibility and restricted abilities. Therefore, a more capable local feature extractor needs to be used to obtain better local features.

We design a mixed convolutional model that uses different-sized CNN kernels to extract features of varying receptive fields. We utilize k = 3, k = 5, and k = 7 convolution operations to obtain features under different receptive fields. These features are then combined to form the final output. This module addresses the limitation of narrow receptive fields in conventional CNNs and enhances the quality of extracted features by incorporating multiple convolution sizes. We name this module “mixconv1d.” Its structure is represented in Figure 1, which consists of convolutions with k = 3, k = 5, and k = 7, as well as a feature concatenation compression module.

To obtain multi-scale features, we leverage the pyramid convolutional technique and modify it to enhance its performance and speed while minimizing computational complexity. Specifically, we replace the conventional convolution in the pyramid structure with a lightweight and efficient separable convolution. Here, we employ SPConv, which splits the features into representative and redundant parts, using k = N and k = 1 convolutions, respectively. This approach captures the main features in the representative part and details features in the redundant part as supplementary information. This design makes the convolutions efficient and lightweight. Replacing the conventional convolutions in a pyramid convolution with SPConv, which uses different kernel sizes, can reduce computational burdens and enable efficient feature extraction at different scales. This novel convolution module is called PySPConv, whose architecture is shown in Figure 2. It is composed of one-dimensional SPConv convolutions with a kernel size of Kn and a feature concatenation module. We can independently configure the kernel sizes and the number of feature layers, which makes this convolutional module highly flexible to fit our needs.

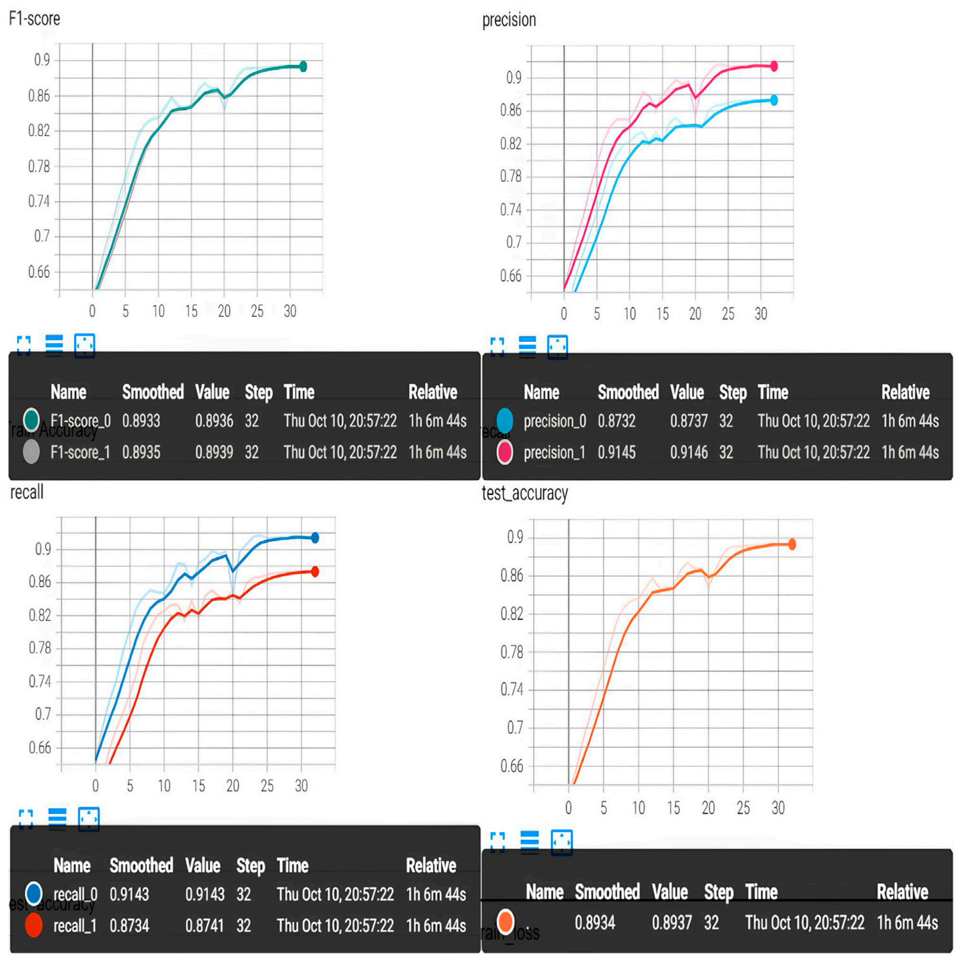
In subsequent experiments, we will compare the performance of the two convolutional modules. We will test and evaluate various metrics to confirm the advantages of PySPConv in computational load and performance.

3.3 Mamba-PySPConv with KSA attention

We include this new PySPConv module in the Mamba structure and employ the KSA attention mechanism between each layer of Mamba blocks, which enables the model to filter out the important features in the input and improves the expressive power of the model. We also add the residual structure between different layers of Mamba so that the features obtained from shallow Mamba blocks can be fused into deeper Mamba blocks, which improves the model convergence, enriches the extracted features, and fully utilizes the features obtained from each block layer. This final model is called KSA-Mamba-PySPConv, and its block structure is shown in Figure 3.

4 Experiments and results

To evaluate the performance of the aforementioned models, we train and evaluate them on the eegmmidb dataset.



Name	Smoothed	Value	Step	Time	Relative
F1-score_0	0.8933	0.8936	32	Thu Oct 10, 20:57:22	1h 6m 44s
F1-score_1	0.8935	0.8939	32	Thu Oct 10, 20:57:22	1h 6m 44s

Name	Smoothed	Value	Step	Time	Relative
precision_0	0.8732	0.8737	32	Thu Oct 10, 20:57:22	1h 6m 44s
precision_1	0.9145	0.9146	32	Thu Oct 10, 20:57:22	1h 6m 44s

Name	Smoothed	Value	Step	Time	Relative
recall_0	0.9143	0.9143	32	Thu Oct 10, 20:57:22	1h 6m 44s
recall_1	0.8734	0.8741	32	Thu Oct 10, 20:57:22	1h 6m 44s

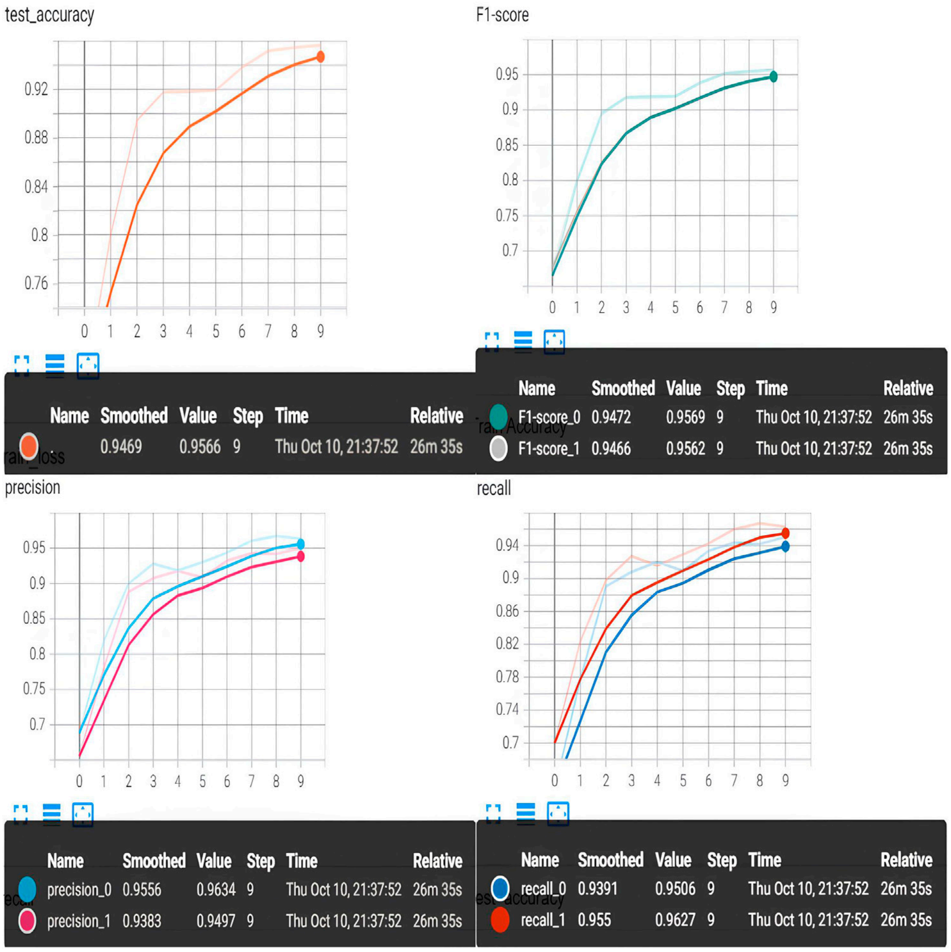
Name	Smoothed	Value	Step	Time	Relative
	0.8934	0.8937	32	Thu Oct 10, 20:57:22	1h 6m 44s

FIGURE 4
Original Mamba results.

4.1 Experiments

Dataset introduction. The eegmmidb dataset contains over 1,500 1-min and 2-min EEG recordings from 109 volunteers, which were obtained from subjects completing a series of motor/imagery tasks. Motor imagery or movement tasks were recorded as EEG signals from 64 channels positioned upon the subject's scalp. Each channel was annotated with three codes: T0, T1, and T2. T0 designates the rest period; T1 signifies the movement of the left hand in selected tasks; T2 denotes the movement of the right hand for certain tasks. Of the 109 participants, six individuals lacked sufficient data recordings and were excluded from the training experiment. All trials involved sustained and continuous movements of 4–4.1 s for execution and imagery tasks. To ensure consistent dataset representation, 4-s trial segments were extracted and clipped, removing any static states or extraneous non-experimental segments. The sampling rate was 160 Hz, and after each trial segment's clipping, 640 samples were obtained.

Experimental setup. We utilize the Magnetoencephalography and Electroencephalography (MNE) library to read raw general data format (GDF) files from the eegmmidb database. A 60 Hz bandpass filter is applied to remove power line interferences. A low-pass filter with a cutoff frequency of 0.5 Hz is then used to suppress low-frequency noise. Finally, a bandpass filter ranging from 1 to 60 Hz is employed to attenuate high-frequency artifacts. The "T1" labels are converted into "0" labels, and the "T2" labels are converted into "1" labels. To maintain consistency in the dataset, the 640 continuous 4-s action data samples are divided into four non-overlapping windows of 160 samples each, which maintains the labeling of the original experiment. Datasets are divided into the motor task data, imagery task data, and data for both tasks. The data obtained is stored in a matrix format. We divide the processed EEG data matrix into a training set,test set and a validation set at a ratio of 7:2:1. The Adam optimizer is used to train the model, with an initial learning rate of 0.0001. Every 20 epochs during the training process, the learning rate is adjusted by 0.1 times its original value. The input sequence length of this Mamba model is 160, the state dimension is 256, and it has three layers. To shorten the training time, we use GPU servers and set a batch size of 24 for our training process. The configuration of the GPU server used in the experiment is as follows: the CPU model is AMD EPYC 9654, the graphics card is an RTX 4090 with 24 GB of video memory, and the system is equipped with 128 GB of RAM. It is also possible to conduct training on a laptop with more than 8 GB of video memory, although the process may be



	Name	Smoothed	Value	Step	Time	Relative
●		0.9469	0.9566	9	Thu Oct 10, 21:37:52	26m 35s

	Name	Smoothed	Value	Step	Time	Relative
●	F1-score_0	0.9472	0.9569	9	Thu Oct 10, 21:37:52	26m 35s
●	F1-score_1	0.9466	0.9562	9	Thu Oct 10, 21:37:52	26m 35s

	Name	Smoothed	Value	Step	Time	Relative
●	precision_0	0.9556	0.9634	9	Thu Oct 10, 21:37:52	26m 35s
●	precision_1	0.9383	0.9497	9	Thu Oct 10, 21:37:52	26m 35s

	Name	Smoothed	Value	Step	Time	Relative
●	recall_0	0.9391	0.9506	9	Thu Oct 10, 21:37:52	26m 35s
●	recall_1	0.955	0.9627	9	Thu Oct 10, 21:37:52	26m 35s

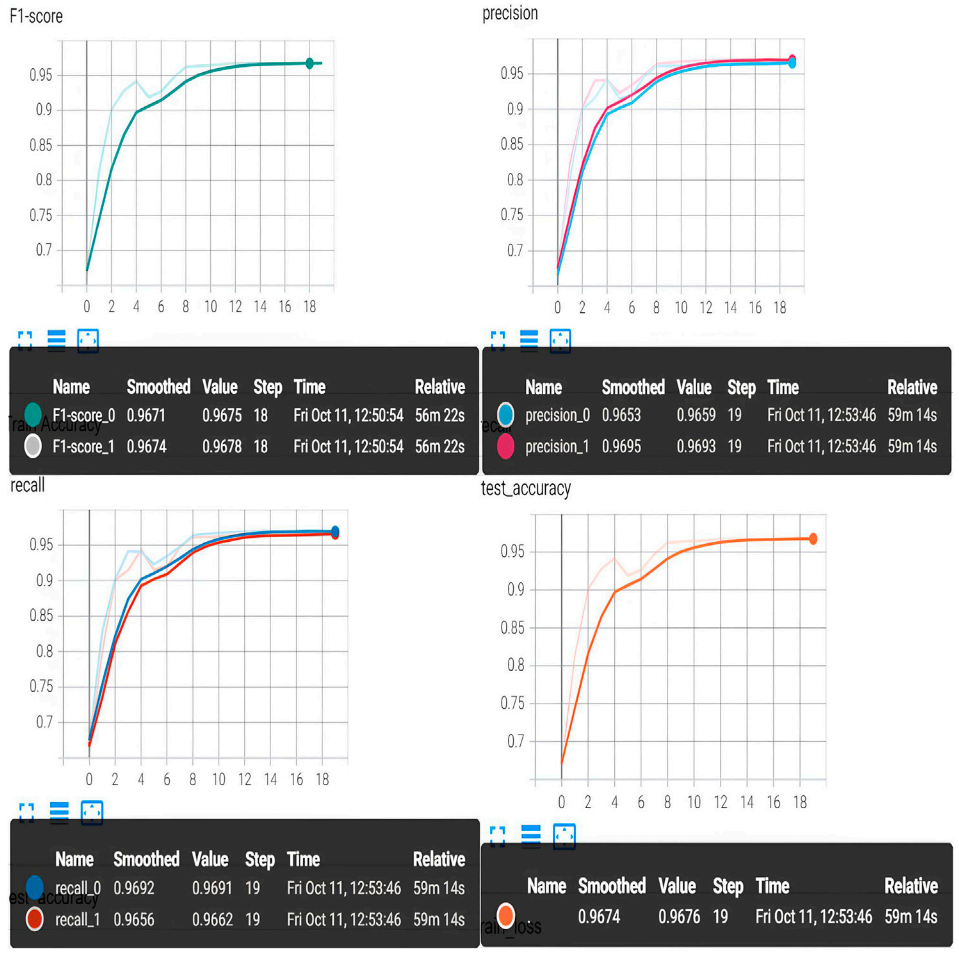
FIGURE 5
Mamba-MixConv1d results.

slower. We perform five sets of comparative experiments using different models to validate the model performance, namely, the original mamba model experiment, Mamba-mixconv1d model experiment, KSA-Mamba-PySPConv model experiment, executed motor task dataset experiment, and imagery motor task dataset experiment.

4.2 Results

Original mamba results. Figure 4 presents the experimental results of the original Mamba model on the brain-EEG motor imagery recognition task. The model employs a standard 1D convolution to extract local features, with a kernel size of 5. We compare the classification performance of the original Mamba model with EEGNet Fusion V2. The original Mamba model achieves significantly improved recognition performance, achieving an accuracy of 89.4%, and its precision, F1 score, and recall values all approach those of EEGNet Fusion V2, being higher than 85%. Furthermore, the original Mamba model requires fewer parameters compared to EEGNet Fusion V2. Increasing the size of the CNN kernel in the Mamba module yields improved model performance, demonstrating that the Mamba structure is more effective in handling sequence data classification tasks compared to multi-layer deep neural networks (DNNs).

Mamba-MixConv1d results. To enhance the Mamba model's ability to extract local features, we replace the original 1D convolution with a MixConv1d convolutional module. Figure 5 presents the experimental results of the Mamba-MixConv1d model on the EEG motor imagery recognition task. We compare the classification performance of the Mamba-MixConv1d model with the original Mamba model. The Mamba-MixConv1d model achieves an accuracy of 95.6%, which is 6% higher than that of the original Mamba model. This improved model exhibits a precision and F1 score approaching 95%, and its recall is near 94%, showing that Mamba-MixConv1d significantly outperforms the original Mamba model. Experimental results conclusively demonstrate that the MixConv1d module possesses strong local feature extraction capabilities and solves the problem of insufficient local feature extraction in the original Mamba model. However, adding more convolutional branches and larger kernel sizes and performing additional fusion calculations result in substantial computational costs and memory consumption.



	Name	Smoothed	Value	Step	Time	Relative
●	F1-score_0	0.9671	0.9675	18	Fri Oct 11, 12:50:54	56m 22s
○	F1-score_1	0.9674	0.9678	18	Fri Oct 11, 12:50:54	56m 22s

	Name	Smoothed	Value	Step	Time	Relative
●	precision_0	0.9653	0.9659	19	Fri Oct 11, 12:53:46	59m 14s
●	precision_1	0.9695	0.9693	19	Fri Oct 11, 12:53:46	59m 14s

	Name	Smoothed	Value	Step	Time	Relative
●	recall_0	0.9692	0.9691	19	Fri Oct 11, 12:53:46	59m 14s
●	recall_1	0.9656	0.9662	19	Fri Oct 11, 12:53:46	59m 14s

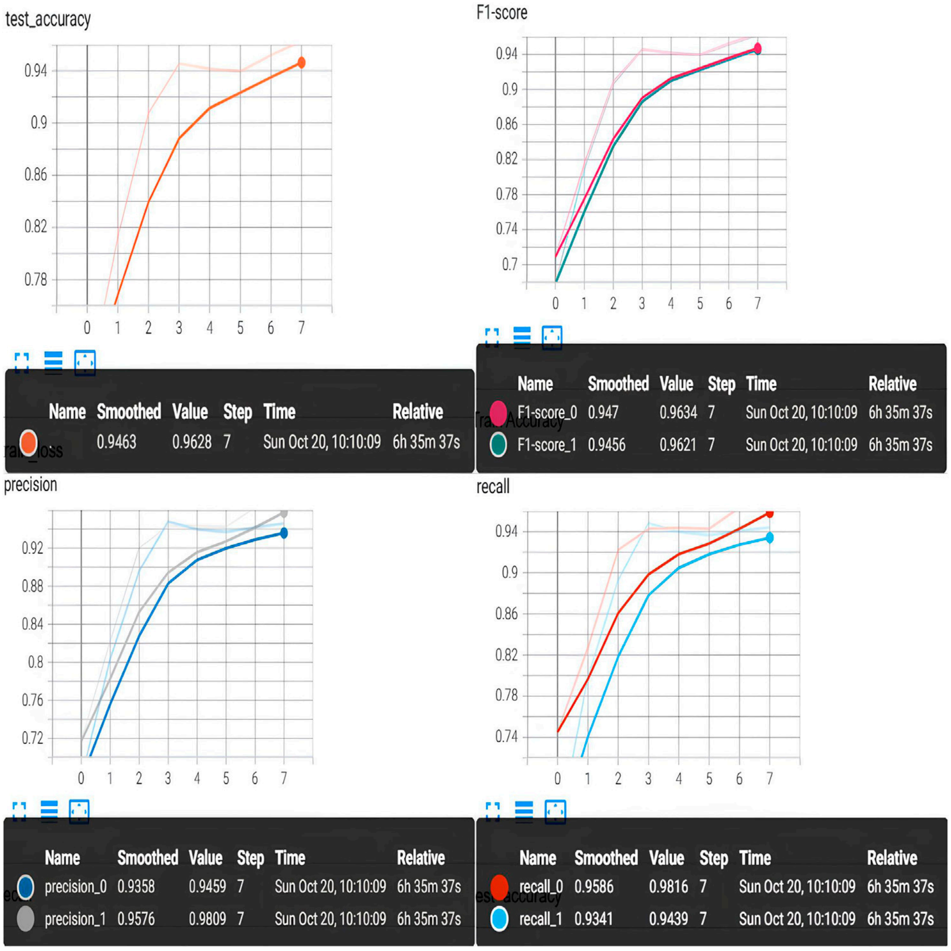
	Name	Smoothed	Value	Step	Time	Relative
●		0.9674	0.9676	19	Fri Oct 11, 12:53:46	59m 14s

FIGURE 6
KSA-Mamba-PySPConv results.

KSA-Mamba-PySPConv results. To reduce the computational load of the convolutional module, we replace MixConv1d with PySPConv. In addition, to improve the expressive power and feature quality of the Mamba structure output, we add the KSA attention module to enhance the channel and spatial features, capturing relationships between different parts of brain activity and detecting more complex data patterns. In Figure 6, we present the experimental results for the KSA-Mamba-PySPConv model on the EEG motor imagery recognition task. We compare its classification performance with that of the Mamba-MixConv1d model. The KSA-Mamba-PySPConv model achieves an accuracy rate of 96.76%, which is 1.76% higher than that of the Mamba-MixConv1d model. The precision, F1 score, and recall of the KSA-Mamba-PySPConv model are all above 96.5%, exceeding those of the Mamba-MixConv1d model by approximately 1%–2%. Experimental results demonstrate that KSA-Mamba-PySPConv possesses stronger local feature extraction capability and exhibits better overall performance than Mamba-MixConv1d. The statistical comparison of the parameters of the two convolutional modules finds that PySPConv has 21.1% fewer parameters compared to MixConv1d when configured with the same kernel size. PySPConv carries fewer redundant features and boasts higher feature extraction efficiency and faster computation speed. Moreover, PySPConv allows for autonomous adjustment and configuration of the depth and kernel size of the convolutional layers while enabling the addition of more convolutional branches. Using the SPConv method significantly reduces the computational burden associated with adding branches and adjusting depths in PySPConv.

Executed motor task dataset experiment results. We test the performance of the KSA-Mamba-PySPConv model using the executed motor task dataset, and the results are shown in Figure 7. We compare the results to those from EEGNet Fusion V2. For the executed motor task, the KSA-Mamba-PySPConv model achieves an accuracy of 96.28%, 6.68% higher than that of EEGNet Fusion V2 (Chowdhury et al., 2023). The precision, F1 score, and recall of KSA-Mamba-PySPConv are all higher than those of EEGNet Fusion V2 by approximately 6.5%. This demonstrates that the KSA-Mamba-PySPConv model outperforms EEGNet Fusion V2 in executed motor movement tasks on eegmmidb.

Imagery motor task dataset experiment results. We also test the performance of the KSA-Mamba-PySPConv model using the imagery motor task dataset, and the results are shown in Figure 8. In this domain, the KSA-Mamba-PySPConv model



	Name	Smoothed	Value	Step	Time	Relative
●		0.9463	0.9628	7	Sun Oct 20, 10:10:09	6h 35m 37s

	Name	Smoothed	Value	Step	Time	Relative
●	F1-score_0	0.947	0.9634	7	Sun Oct 20, 10:10:09	6h 35m 37s
●	F1-score_1	0.9456	0.9621	7	Sun Oct 20, 10:10:09	6h 35m 37s

	Name	Smoothed	Value	Step	Time	Relative
●	precision_0	0.9358	0.9459	7	Sun Oct 20, 10:10:09	6h 35m 37s
●	precision_1	0.9576	0.9809	7	Sun Oct 20, 10:10:09	6h 35m 37s

	Name	Smoothed	Value	Step	Time	Relative
●	recall_0	0.9586	0.9816	7	Sun Oct 20, 10:10:09	6h 35m 37s
●	recall_1	0.9341	0.9439	7	Sun Oct 20, 10:10:09	6h 35m 37s

FIGURE 7
Experiment results on the executed motor task dataset.

achieves an accuracy of 96.33%, 6.73% higher than that of EEGNet Fusion V2. The precision, F1 score, and recall of KSA-Mamba-PySPConv are all higher than those of EEGNet Fusion V2 by approximately 6.5%. This demonstrates that the KSA-Mamba-PySPConv model outperforms EEGNet Fusion V2 in imagery motor tasks on eegmmidb.

5 Discussion

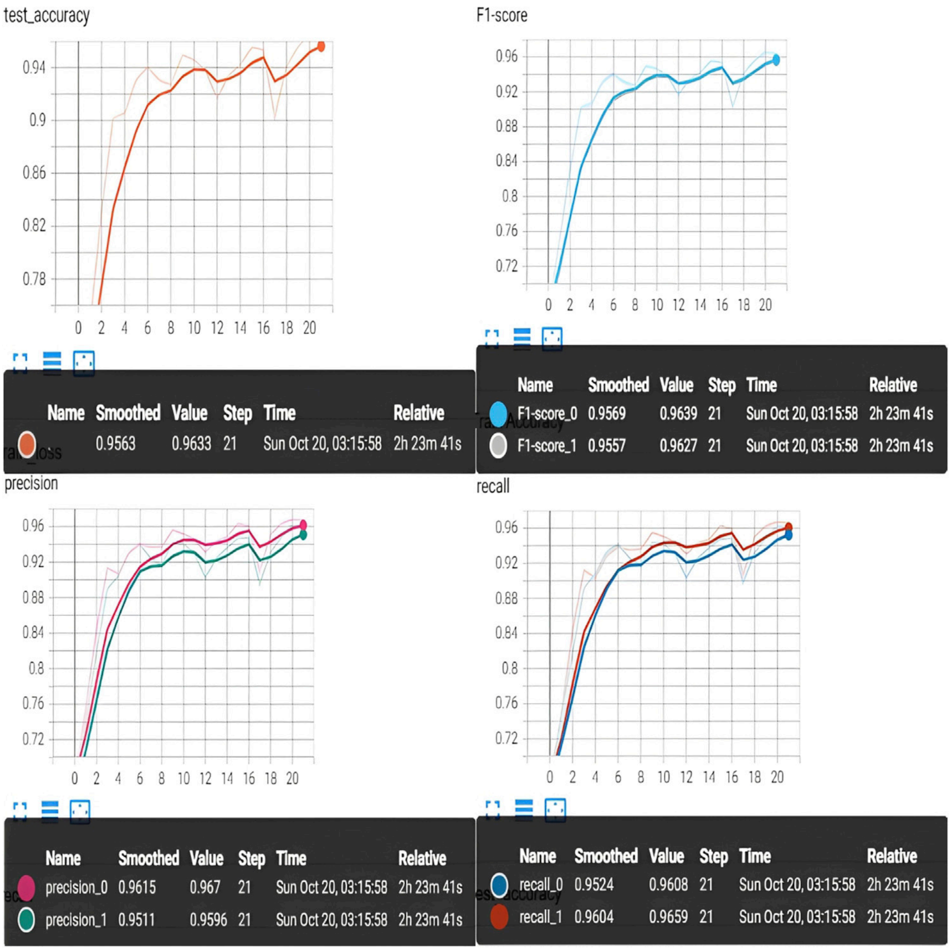
This section discusses the performance differences between various models and their respective advantages and disadvantages. First, the experimental results for executed and imagery motor movement tasks are organized and compared for the models detailed in this study, as summarized in Table 1.

The original Mamba model demonstrates good performance in classifying heart rates, validating the model's effectiveness at handling 1D data. The use of MixConv1D in Mamba significantly improves classification accuracy, proving that MixConv1D enhances convolutional features through local feature extraction and boosts overall model performance. When using PySPconv and KSA attention modules in conjunction with Mamba, compared to using MixConv1D alone, we observe improved classification accuracy. Furthermore, the parameters of PySPconv are 21.1% fewer than those of MixConv1D, which can be obtained by torchstat, highlighting the efficiency and lightweight nature of this module.

The results of some advanced deep learning models and our method for executed motor movement tasks are compared in Table 2.

As can be seen in Table 2, our new model exceeds previous research models in the performance metrics of the classification task. The KSA module helps the model better capture the nonlinear relationships in the data. PySPConv allows the model to process data sparsely, thereby reducing the number of parameters and computational complexity.

The results for imagery motor movement tasks are shown in Table 3. It can be seen that the proposed KSA-Mamba-PySPConv model also demonstrates excellent classification performance in imagery motor movement tasks, reaching the SOTA level.



Name	Smoothed	Value	Step	Time	Relative
●	0.9563	0.9633	21	Sun Oct 20, 03:15:58	2h 23m 41s

Name	Smoothed	Value	Step	Time	Relative
● F1-score_0	0.9569	0.9639	21	Sun Oct 20, 03:15:58	2h 23m 41s
● F1-score_1	0.9557	0.9627	21	Sun Oct 20, 03:15:58	2h 23m 41s

Name	Smoothed	Value	Step	Time	Relative
● precision_0	0.9615	0.967	21	Sun Oct 20, 03:15:58	2h 23m 41s
● precision_1	0.9511	0.9596	21	Sun Oct 20, 03:15:58	2h 23m 41s

Name	Smoothed	Value	Step	Time	Relative
● recall_0	0.9524	0.9608	21	Sun Oct 20, 03:15:58	2h 23m 41s
● recall_1	0.9604	0.9659	21	Sun Oct 20, 03:15:58	2h 23m 41s

FIGURE 8
Experiment results on the imagery motor task dataset.

TABLE 1 Results for executed and imagery motor movement tasks.

Model	Accuracy %	Precision %		Recall %		F1 score %		Params	flops
		Left	Right	Left	Right	Left	Right		
Original Mamba	89.37	87.4	91.5	91.4	87.4	89.4	89.4	716.802k	2.369G
Mamba-MixConv1d	95.66	96.3	94.9	95.1	96.3	95.7	95.6	1513M	5.434G
KSA-Mamba-PySPConv	96.76	96.6	96.9	96.9	96.6	96.7	96.8	1193k	4.206G

6 Conclusion

In this paper, we propose a novel architecture called KSA-Mamba-PySPConv for the EEG imagery/motor movement classification tasks. The proposed scheme includes a KSA attention mechanism and a PySPConv module to enhance the features extracted from a single module layer. The KSA attention mechanism achieves enhanced and filtered channel and spatial features by integrating the KAN network with attention mechanisms. PySPConv utilizes different convolutional kernels to extract pyramid-like multi-scale features and employs split operation and parameter-free feature fusion algorithms to achieve lightweight and efficient convolutions. These configurations enable KSA-Mamba-PySPConv to outperform conventional EEG classification models and achieve SOTA performance. The model exhibits excellent performance across different tasks on eegmmidb, proving its strong generalization capabilities. When deploying this model in a practical BCI system, we may need to consider the model's size and the consumption of computational resources. Therefore, techniques such as quantization and pruning might be employed for the deployment of the model. In the future, we will explore pruning algorithms and optimization methods to enhance the speed of this model.

TABLE 2 Results for executed motor movement tasks.

Model	Accuracy %	Precision %		Recall %		F1 score %		Params	Flops
		Left	Right	Left	Right	Left	Right		
DeepConvNet (Schirrmeister et al., 2017)	76.6	76.2	77.1	77.3	76.0	76.7	76.5	97.302K	203.453M
ShallowConvNet (Schirrmeister et al., 2017)	79.3	79.2	79.3	79.2	79.3	79.2	79.3	80B	580K
MMCNN (Jia et al., 2021)	81.4	82.2	80.6	80.5	82.3	81.3	81.4	-	-
EEGNet (Lawhern et al., 2018)	66.6	69.1	64.8	59.9	73.3	64.2	68.8	1.114K	132.251M
EEGNet Fusion (Roots et al., 2020)	84.1	84.2	84.5	83.8	83.9	84.0	84.2	17.682K	1.597G
EEGNet Fusion V2 (Chowdhury et al., 2023)	89.6	89.9	89.4	89.4	89.8	89.7	89.6	9.636M	16.546G
KSA-Mamba-PySPConv	96.28	94.59	98.09	98.16	94.39	96.34	96.21	1.193M	4.206G

TABLE 3 Results for imagery motor movement tasks.

Model	Accuracy %	Precision %		Recall %		F1 score %	
		Left	Right	Left	Right	Left	Right
DeepConvNet (Schirrmeister et al., 2017)	76.2	76.5	75.9	76.0	76.4	76.3	76.1
ShallowConvNet (Schirrmeister et al., 2017)	78.2	78.2	78.3	78.7	77.8	78.5	78.0
MMCNN (Jia et al., 2021)	81.6	81.7	81.5	81.9	81.2	81.8	81.3
EEGNet (Lawhern et al., 2018)	68.4	68.3	68.4	69.2	67.5	68.8	67.9
EEGNet Fusion (Roots et al., 2020)	83.8	85.0	83.3	82.9	84.8	83.9	84.0
EEGNet Fusion V2 (Chowdhury et al., 2023)	87.8	88.1	87.5	87.5	88.1	87.8	87.8
KSA-Mamba-PySPConv	96.33	96.7	95.96	96.08	96.59	96.39	96.27

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/supplementary material.

Author contributions

ZL: Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review and editing.

Funding

The author(s) declare that no financial support was received for the research and/or publication of this article.

Conflict of interest

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Generative AI was used in the creation of this manuscript.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Chen, J., Zhang, Y., Pan, Y., Xu, P., and Guan, C. (2022). A transformer-based deep neural network model for SSVEP classification.

Chowdhury, R. R., Muhammad, Y., and Adeel, U. (2023). Enhancing cross-subject motor imagery classification in EEG-based brain–computer interfaces by using multi-branch CNN. *Sensors* 23 (18), 7908. doi:10.3390/s23187908

Duta, I. C. (2021). *Pyramidal convolution: rethinking convolutional neural networks for visual recognition*. Cornell University - arXiv, Cornell University - arXiv.

Gu, A., and T. D. Mamba (2023). Linear-time sequence modeling with selective state spaces.

Hou, Y., Jia, S., Lun, X., Zhang, S., Chen, T., Wang, F., et al. (2022). Deep feature mining via the attention-based bidirectional long short term memory graph convolutional neural network for human motor imagery recognition. *Front. Bioeng. Biotechnol.* 9. doi:10.3389/fbioe.2021.706229

Jia, Z., Lin, Y., Wang, J., Yang, K., Liu, T., and Zhang, X. (2021). "MMCNN: a multi-branch multi-scale convolutional neural network for motor imagery classification," in *Proceedings of the machine learning and knowledge discovery in databases: European conference, ECML PKDD 2020, ghent, Belgium, 14–18 september 2020; proceedings, Part III* (Berlin, Germany: Springer), 736–751.

Keutayeva, A., and Abibullaev, B. (2023). Exploring the potential of attention mechanism-based deep learning for robust subject-independent motor-imagery based BCIs. *IEEE Access* 11, 107562–107580. doi:10.1109/access.2023.3320561

Lawhern, V. J., Solon, A. J., Waytowich, N. R., Gordon, S. M., Hung, C. P., and Lance, B. J. (2018). EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces. *J. Neural Eng.* 15, 056013. doi:10.1088/1741-2552/aace8c

Li, Y., Guo, L., Liu, Y., Liu, J., and Meng, F. (2021). A temporal-spectral-based squeeze-and-excitation feature fusion network for motor imagery EEG decoding. *IEEE Trans. Neural Syst. Rehabilitation Eng.* 29, 1534–1545. doi:10.1109/tnsre.2021.3099908

Luo, J., Wang, Y., Xia, S., Lu, N., Ren, X., Shi, Z., et al. (2023). A shallow mirror transformer for subject-independent motor imagery BCI. *Comput. Biol. Med. Sept* 164, 107254. doi:10.1016/j.compbiomed.2023.107254

Roots, K., Muhammad, Y., and Muhammad, N. (2020). Fusion convolutional neural network for cross-subject EEG motor imagery classification. *Computers* 9, 72. doi:10.3390/computers9030072

Schirrmeister, R. T., Springenberg, J. T., Fiederer, L. D. J., Glasstetter, M., Eggensperger, K., Tangermann, M., et al. (2017). Deep learning with convolutional neural networks for EEG decoding and visualization. *Hum. Brain Mapp.* 38, 5391–5420. doi:10.1002/hbm.23730

Vaca-Rubio, C. J., Blanco, L., Pereira, R., and Caus, M. (2024). Kolmogorov-arnold networks (KANs) for time series Analysis.

Zhang, Q., Jiang, Z., Lu, Q., Han, J., Zeng, Z., Gao, S., et al. (2020). "Split to Be slim: an overlooked redundancy in vanilla convolution," in *Proceedings of the twenty-ninth international joint conference on artificial intelligence*. doi:10.24963/ijcai.2020/442