PTSD Prediction Based on EEG Feature Extraction and CNN-LSTM Model

Zhenhua Qu, Weipu Wu, Yuan Cao*

Abstract—**ABSTRACT: This study focuses on firefighters and the post-traumatic stress disorder (PTSD) symptoms they experience. As a crucial rescue force in society, firefighters often encounter high-risk and high-stress work environments, such as fires, natural disasters, and accident rescues, which can readily lead to PTSD. Currently, the diagnosis of PTSD primarily relies on the patient's medical history, symptoms, and professional psychological evaluations. By leveraging the spatiotemporal characteristics of EEG signals, we employed convolutional neural networks (CNN) and long short-term memory networks (LSTM) to process the spatial and temporal features of EEG signals, respectively, resulting in the construction of a CNN-LSTM model. Firstly, the differential entropy features of different frequency bands were extracted after time-domain segmentation. The differential entropy data structure for each frequency band was then transformed from one-dimensional to two-dimensional, based on electrode positions using two-dimensional mapping. Subsequently, the two-dimensional differential entropy maps for each frequency band were stacked to form a three-dimensional dataset. Finally, this segmented three-dimensional data was processed to yield four-dimensional data. Secondly, the data were fed into the CNN-LSTM model. Firstly, the spatial features in the EEG signals were processed using a convolutional neural network.The processed data were then fed into a long short-term memory (LSTM) network, where the temporal sequence information was further processed. The experimental results demonstrate that the CNN-LSTM model achieves an accuracy of 80%, a precision of 55.26%, a recall of 57.32%, and an F1 score of 56.27% in predicting PTSD. This performance represents a significant improvement compared to common machine learning models and other deep learning models.This study introduces an innovative electroencephalogram (EEG) - driven approach. By integrating time - frequency domain feature extraction with a deep hybrid neural network, it enables effective prediction of post - traumatic stress disorder (PTSD). Initially, the study segments the raw EEG signals in the time domain and extracts differential entropy features across multiple frequency bands. These features are highly sensitive to the nonlinear complexity changes in EEG activities. To address the limitation of traditional one-dimensional features that overlook spatial structures, based on the spatial distribution of electrodes, the differential entropy data of each frequency band is transformed into two-dimensional scalp topographic maps, thereby retaining spatial adjacency information. Subsequently, the two - dimensional feature maps of different frequency bands are stacked along the channel dimension to form a three - dimensional data matrix that encapsulates spatiotemporal information. Through the application of a sliding time - window technique, this matrix is further expanded into a four - dimensional tensor (channel × frequency × space × time), enabling comprehensive capture of the dynamic spatiotemporal patterns of EEG. In the model construction phase, a CNN - LSTM cascade architecture is employed. The initial convolutional layers leverage multi - scale kernels to process the three - dimensional EEG data in parallel, automatically extracting high - order features in the spatial - frequency domain. The subsequent LSTM network then models the sequential features output by the CNN, effectively capturing the long-term dependencies in EEG activities. Experimental results demonstrate that this hybrid model excels in the PTSD prediction task, achieving an accuracy of 80%, a precision of 55.26%, a recall of 57.32%, and an F1 - score of 56.27%. These results represent a substantial improvement over single models. This study not only validates the efficacy of multimodal EEG features in characterizing the pathology of PTSD but also offers a novel deep - learning framework with spatiotemporal modeling capabilities for predicting mental disorders using neuroimaging. It holds significant promise for facilitating the clinical translation of objective biomarkers for PTSD.**

Index Terms—**EEG signals; PTSD prediction; Convolutional neural network; Long short-term memory network; Hybrid neural network CNN-LSTM.**

I. Introduction

PTSD , also known as Post-Traumatic Stress Disorder, is a severe psychological condition. It is a mental disorder that arises when an individual experiences, witnesses, or is confronted with one or more events involving actual death, threats of death, or serious physical injury to themselves or others [1]. PTSD is not contagious, but it can significantly impair the quality of life for those affected and increase the risk of suicide. High-risk groups primarily consist of military personnel directly involved in combat and disaster relief, rescue workers, and witnesses of major accidents [2].For firefighters, they often engage in rescue operations during fires, earthquakes, floods, and various other accidents. These high-risk and high-stress environments make them a high-risk group for PTSD. Therefore, predicting PTSD in firefighters is crucial for preventing the development of serious mental health issues. Early detection allows for targeted interventions and treatments, which can help mitigate the severe consequences of PTSD. Currently, the primary diagnostic methods for PTSD include clinical assessments, history collection, and the use of standardized assessment tools such as structured interview tables and self-rating scales [3]. EEG, or electroencephalogram, records the electrical signals generated by neurons in the brain. These signals reflect the state of brain function. In patients with PTSD, there are characteristic changes in EEG patterns that may be closely related to the pathophysiological mechanisms of the disorder. Specifically, individuals with PTSD may exhibit abnormal arousal responses, altered frontal and temporal lobe activity, and reduced connectivity in their EEG activity.

Manuscript received November 20, 2024; revised June 20,2025.

This work was supported in part by the Doctoral Research Initiation Fund of Shandong University of Technology under Grant No. 417037.

Zhenhua Qu is a graduate student in statistics at the School of Mathematics and Statistics, Shandong University of Technology, China. (e-mail: 18463064907@163.com)

Weipu Wu is a graduate student in applied statistics at the School of Mathematics and Statistics, Shandong University of Technology, China. (e-mail: 2863375150@qq.com)

Yuan Cao is an associate professor at the School of Mathematics and Statistics, Shandong University of Technology, China. (Corresponding author, e-mail: yuancao@sdut.edu.cn)

TABLE I
CRITERIA FOR CLASSIFYING THE SEVERITY OF PTSD BY SELF-RATING SCALE SCORES

Self - rating scale score	PTSD rating
0 - 20	Class 1
21 - 35	Class 2
36 - 50	Class 3
>50	Class 4

Therefore, monitoring and analyzing the EEG of firefighters can provide crucial information for the diagnosis, treatment, and prevention of PTSD.

II. DATA SOURCE AND PREPROCESSING

A. Sources of data

The data used in this study were EEG data from the 2024 Applied Statistics professional degree graduate case competition, aimed at predicting PTSD in rescue workers. The CNN-LSTM model was validated in the processing of EEG signals. This dataset collected EEG signals from 25 subjects during the experiment, and the corresponding subjects' PCL total scores were provided as an indicator of PTSD presence. The EEG data included eight types of brain wave data: Delta, Theta, Alpha1, Alpha2, Beta1, Beta2, Gamma1, and Gamma2, recorded once per second. To achieve optimal results in data processing, we chose to use Python software, which is well-suited for handling and analyzing EEG data.Based on the scores from the self-rating scale, PTSD can be categorized into four levels, with the classification criteria presented in Table I: As shown in the table, scores ranging from 0 to 20 are classified as Class 1, indicating no significant PTSD symptoms. Scores between 21 and 35 fall into Class 2, characterized by mild PTSD symptoms, such as mild re-experiencing of trauma, a certain degree of avoidance behavior, and mild increased alertness. Class 3 encompasses scores from 36 to 50, reflecting more severe PTSD symptoms, including an increased frequency of traumatic experiences, heightened alertness, avoidance, and numbness, often accompanied by sleep disturbances. Scores above 50 are categorized as Class 4, which is marked by severe PTSD symptoms. In addition to the significant intensification of the aforementioned characteristics, individuals may also experience severe depression, changes in their outlook on life and values, and significant personality alterations [4].

B. Data preprocessing

In terms of data processing, the acquired EEG data undergo preprocessing, which includes noise removal, artifact removal, data standardization, and normalization. Once the preprocessing is complete, the 200 seconds of experimental data are segmented, using a ten-second interval to divide each participant's data into 20 segments. Individuals with PTSD often experience severe emotional issues, including persistent negative emotions such as anger, anxiety, depression, and fear, as well as a decrease in emotional stability. They may become irritable, overreactive, and highly sensitive to external stimuli. Based on the findings of pertinent research, it has been established that DE features exhibit superior recognition capabilities in the realm of emotion recognition and classification [5]. Therefore, in this paper, we opt to calculate the differential entropy for each time period [6], and the formula of the differential entropy is:

$$h(X) = \int_{-\infty}^{+\infty} f(x) \log(f(x))\, dx \tag{1}$$

In the above equation, X is a random variable; $f(x)$ represents the probability density function of X. In a fixed frequency band, its distribution can be approximated as Gaussian distribution$N(\mu, \sigma^2)$. After replacing $f(x)$ with Gaussian normal distribution, $h(X)$ can be expressed as follows:

$$h(X) = \frac{1}{2} \log(2\pi e \sigma_i^2) \tag{2}$$

In the above equation, e denotes the Euler constant, and σ_i represents the standard deviation of the time series.

According to previous research results, the location information of the motor of EEG signal plays an extremely important role in emotion recognition [7]. Therefore, we retain the spatial information of electrodes, and use the spatial position of electrodes to transform the DE data structure of each frequency band from one-dimensional to two-dimensional mapping [8]. The mapping rule is illustrated in Fig 1. Subsequently, we stack the two-dimensional differential entropy feature maps of different frequency bands to obtain a three-dimensional feature set. This set encapsulates the spatial and frequency information of the EEG signal, as well as complementary information across various frequency bands. Initially, we divided the original EEG signal into 20 equal segments and processed each segmented EEG segment individually. Ultimately, we derived the four-dimensional features through these operations. Consequently, each EEG segment can be represented as$S_n \in R^{H \times W \times M \times 2T}$, where H and W represent the height and width of the two-dimensional feature map respectively. M denotes the number of bands; $2T$ denotes 2 times the time slice .

To mitigate the common issue of overfitting in image classification tasks, we utilize data augmentation techniques to expand the dataset [9].By applying transformations such as flipping, zooming, and cropping to the images, we generate additional data samples. This approach enhances the model's generalization capabilities and reduces the risk of overfitting [10].

III. ESTABLISHMENT OF CLASSIFICATION MODEL

A. Convolutional Neural Networks

The convolutional neural network progressively extracts abstract features from images by stacking multiple layers. In the convolutional layer, filters perform sliding convolution operations on the input image to extract local features. Subsequently, the pooling layer samples the output from the convolutional layer, thereby reducing the dimensionality of the feature map. Finally, the fully connected layer integrates the extracted features and outputs the final prediction result [11]. The basic structure of a CNN typically includes an input layer, convolutional layer, pooling layer, fully connected layer, and output layer [12]. The convolutional layer is the

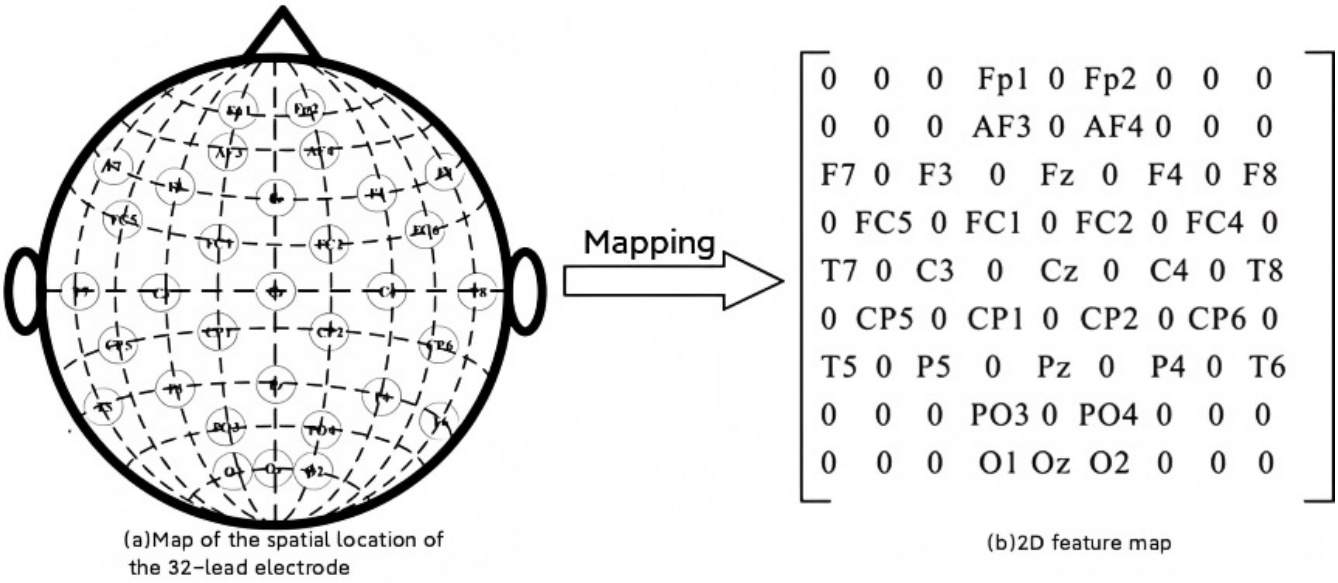

Fig. 1. The two-dimensional mapping relationship diagram of the electrodes, (a) represents the spatial position of the electrodes, and (b) represents the mapping of the electrode position into a two-dimensional feature map

core component of the CNN, responsible for extracting local features from the image. This is achieved by sliding a filter of a specific size across the input image and performing a convolution operation on each region [13]. The calculation formula of convolutional layers mainly involves the size of the output feature map, and the number of parameters. The formula for calculating the size of the output feature map of a convolutional layer is:

$$N = \lfloor \frac{W - F + 2P}{S} \rfloor + 1 \tag{3}$$

Nrepresents the size of the output feature map; W denotes the size of the input feature map;F is the size of the convolution kernel; Prefers to the padding size. Padding involves adding extra zero values around the boundaries of the input feature map to control the size of the output feature map; S stands for the stride, which is the distance the convolution kernel moves across the input feature map.When the calculation result is not an integer, it is usually rounded down, which is represented by the notation $[x]$for the floor function.

The formula for calculating the number of parameters in a convolutional layer is:

$$params = c\_o \times (k\_w \times k\_h \times c\_i) + b \tag{4}$$

The c_o represents the number of channels in the output feature map; k_w andk_h denote the width and height of the convolution kernel, respectively; c_i is the number of channels in the input feature map; b refers to the bias term.

The pooling layer primarily reduces the dimensionality of the feature map and minimizes noise interference, which helps to mitigate overfitting and enhance the model's generalization capability to some extent. The fully connected layer integrates the extracted features and performs classification or regression predictions. The modules of a convolutional neural network are illustrated in Fig 2: CNNs possess several advantages, including local perception and weight sharing, invariance to translation, rotation, and scale, multi-level feature extraction, and efficient computational performance. These strengths have made CNNs notably successful in the fields of image processing, speech recognition, and natural language processing, establishing them as the technology of choice in these domains [14].Convolutional neural networks can be categorized into one-dimensional, two-dimensional, and three-dimensional CNNs. One-dimensional CNNs are commonly used to process time series data. Two-dimensional CNNs are typically employed for processing data such as text and images. Three-dimensional CNNs are often utilized for processing medical images, videos, and other volumetric data [15].The basic operation process of CNN is as follows:

1) Data Preparation: Given an image dataset $X = \{x_1, x_2, \cdots, x_n\}$, the images are first pre - processed. This pre - processing step is crucial as it standardizes the input data, making it more suitable for the subsequent operations. Common pre - processing techniques include normalizing pixel values and resizing the images to a consistent size. Once pre - processed, the data is loaded into the system and divided into batches. These batches are essential for the training process, as they allow for efficient handling of the data and enable the model to learn from the dataset in a structured manner.

2) Feature Extraction: CNN model with multiple convolutional layers is constructed. In the first layer, filters are employed to extract primary features from the input images, resulting in the generation of feature maps. The Rectified Linear Unit (ReLU) activation function is then applied to introduce non - linearity. This non - linearity is vital as it enables the model to learn complex patterns and relationships within the data. Subsequently, convolutional layers are alternated with pooling layers. Pooling layers, such as max - pooling or average - pooling, play a significant role in reducing the dimensionality of the feature maps while retaining the most important features. After several layers of such operations, feature maps at different levels are obtained. These feature maps encapsulate the abstract features of the images, ranging from low - level to high - level, and are used

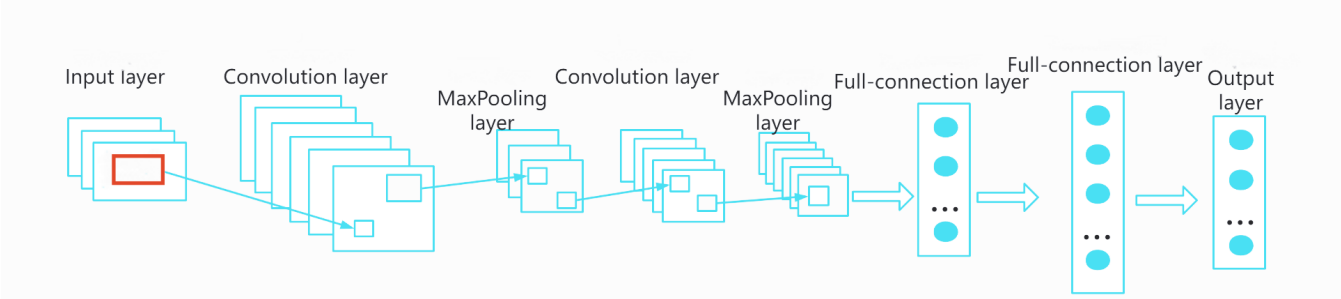

Fig. 2. Structural diagram of CNN

for subsequent decision - making processes.

3) Classification Decision: The feature maps from the last layer are flattened into a one - dimensional vector. This vector is then fed into a fully - connected layer, where both linear and non - linear transformations are applied. The number of neurons in the output layer is equal to the number of classes in the classification task. For example, in a 10 - class classification problem, the output layer will have 10 neurons. The softmax activation function is used in the output layer to transform the output values into a probability distribution. This probability distribution indicates the likelihood of the input image belonging to each class. The class with the highest probability is selected as the prediction result. During the training process, the prediction is compared with the true labels, and the cross - entropy loss is calculated to evaluate the accuracy of the model's predictions. This comparison and loss calculation are essential for guiding the model's learning process and improving its performance over time.

4) Model Optimization: Based on the calculated loss value, backpropagation is used to compute the gradients of the loss with respect to the network's parameters. These gradients indicate the direction in which the parameters should be adjusted to minimize the loss. The parameters are then updated in the opposite direction of the gradients. An appropriate optimizer is selected to perform this update operation. The optimizer takes into account the gradients and the specified hyperparameters to adjust the parameters effectively. Additionally, hyperparameters and the model's structure can be adjusted to optimize the model's performance further. This may involve tuning parameters such as the learning rate and weight decay, or making changes to the architecture, such as adding or removing convolutional and fully - connected layers.

5) Model Evaluation and Application: he trained CNN model is evaluated using a validation set or a test set. Key performance metrics, such as accuracy, recall, and the F1 - score, are calculated to assess the model's performance. If the evaluation results are satisfactory, indicating that the model has achieved a sufficient level of accuracy and generalization ability, the model is applied to real - world image classification tasks. In these tasks, the model predicts the classes of new, unseen images, demonstrating its practical value in various applications, such as image recognition, object detection, and content - based image retrieval.The specific steps of the CNN algorithm are as follows:

Algorithm 1 CNN Training Algorithm

Input: Input Images X, Number of Classes C, Learning Rate η, Epochs E
Output: Trained Model Parameters W, b

Initialize: Weights W and Biases b for all layers
for $e \leftarrow 1$ **to** E **do**
  for $x \in X$ **do**
    $h \leftarrow x$ ▷ Input layer
    for Convolutional Layer l **do**
      $h \leftarrow \text{ReLU}(\text{Conv}(h, W_l) + b_l)$
      $h \leftarrow \text{MaxPool}(h)$
    end for
    $h \leftarrow \text{Flatten}(h)$ ▷ Flatten features
    for Fully Connected Layer l **do**
      $h \leftarrow \text{ReLU}(h \cdot W_l + b_l)$
    end for
    $p \leftarrow \text{Softmax}(h \cdot W_{\text{out}} + b_{\text{out}})$ ▷ Output probabilities
    Compute Loss L and Gradients ∇L
    Update parameters: $W \leftarrow W - \eta \cdot \nabla W$, $b \leftarrow b - \eta \cdot \nabla b$
  end for
end for
return W, b

B. Long Short-Term Memory network

LSTM is a specialized type of recurrent neural network designed to address the issues of vanishing and exploding gradients, thereby effectively managing long-term dependencies that are common in traditional recurrent neural networks [16]. LSTM possesses robust expressive capabilities and is extensively utilized in various domains such as natural language processing, speech recognition, and time series prediction. The fundamental structure of an LSTM comprises three gating mechanisms: the forget gate, the input gate, and the output gate. These gates determine which information to discard and which to retain. Through intricate interactions and cooperation, these gating units empower LSTM to effectively capture long-term dependencies within sequential data [17]. The initial component of an LSTM is the forget gate, which discerns the information to be discarded. The forget gate can be mathematically represented as:

$$f_t = \sigma(W_f[x_t, h_{t-1}] + b_f) \tag{5}$$

The second component of an LSTM is the input gate, which decides what information to retain. The input gate can be mathematically represented as:

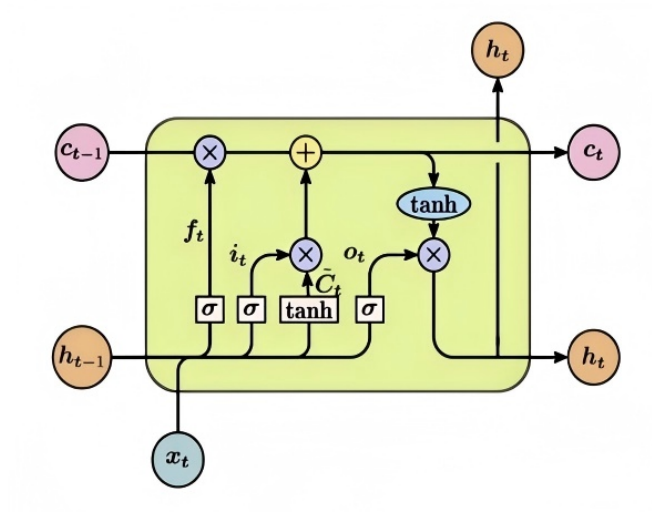

Fig. 3. Structural diagram of LSTM

$$i_t = \sigma(W_i[x_t, h_{t-1}] + b_i) \tag{6}$$

The third component of an LSTM is the output gate, which decides what information to output. The output gate can be mathematically represented as:

$$o_t = \sigma(W_o[x_t, h_{t-1}] + b_o) \tag{7}$$

Here, f_t represents the forget gate and ranges from [0,1], where 0 means all information is discarded and 1 means all information is retained. i_t denotes the input gate; o_t denotes the output gate; σ_i is the Sigmoid function. $x_t$is the input at time t, and $h_t - 1$ is the output at time t-1. Let σ_i denote the activation function; W_f denotes the weight matrix for the forget gate; b_f is the bias vector. The output h_t at time t can be expressed as:

$$h_t = o_t \tanh(\tilde{C}_t) \tag{8}$$

Where tanh is the hyperbolic tangent function; $\tilde{C}_t$ represents the memory information at time t and is calculated as follows:

$$\tilde{C}_t = f c_{t-1} + i \tanh(W_c[x_t, h_{t-1}] + b_c) \tag{9}$$

The input gate, output gate, and forget gate allow the LSTM unit to store, retrieve, and update long-term sequence information, and together they determine the final output result. The structure diagram of LSTM is shown in Fig 3 [18]:

Based on the LSTM flowchart and the formulas provided, the forgetting gatef_t , the input gate i_t and the output gate o_t are obtained by Sigmoid transformation of the current input x_t and the previous output $h_t - 1$. The forget gate f_t is applied to the previous memory information $c_t - 1$. The input gate i_t is applied to the input information processed by the hyperbolic tangent function of x_t and $h_t - 1$. The memory information $c_t - 1$ processed by the forget gate f_t and the input information processed by the input gate i_t are summed to obtain the current memory information $\tilde{C}_t$. By applying the output gate to the memory information $\tilde{C}_t$ processed by the hyperbolic tangent function at this time, the current output h_t is obtained. At the same time, the current output h_t also serves as the input for the next time step. The basic operation process of CNN is as follows:

1) Data Preparation: Data Input: Given a time series dataset $T = \{t_1, t_2, \ldots, t_n\}$, where each t_i represents a time series sample. For example, in the context of analyzing financial data, each sample could be a sequence of stock prices over a specific period.

Data Preprocessing: Standardize the time series data by mapping it to a specific interval.This standardization is crucial as it accelerates the convergence of model training. Additionally, based on the input requirements of the model, split and pad the data to form fixed-length sequence samples. This ensures that all input samples have a consistent format, which is necessary for the model to process them efficiently.

Data Loading and Batching: Load the preprocessed data into the memory and divide it into batches. Batching the data is an essential step for subsequent model training as it improves the training efficiency. By processing data in batches, the model can update its parameters more effectively and make better use of computational resources.

2) Feature Extraction and Temporal Modeling: Building the LSTM Model: Each LSTM layer is composed of multiple memory units. Each memory unit contains an input gate, a forget gate, and an output gate. These gate structures play a vital role in effectively handling the long-term dependencies in time series data. They allow the model to selectively remember or forget information over time, which is crucial for capturing the underlying patterns in the data.

Feature Learning and Sequence Modeling: Feed the time series samples into the LSTM layer step by step according to the time steps. The memory units of the LSTM layer, based on the current input and the state from the previous time step, selectively retain or update the memory information through the gating mechanism. In this way, the model can learn the dynamic features of the time series, such as trends, seasonality, and sudden changes.

Output Feature Mapping: After being processed by the LSTM layer, output feature vectors that contain both the long-term and short-term features of the time series. These feature vectors serve as the basis for subsequent classification or prediction tasks. They encapsulate the essential information of the time series, enabling the model to make accurate predictions or classifications.

3) Classification or Prediction Decision: Fully Connected Layer and Output Layer: Input the feature vectors output by the LSTM layer into the fully connected layer. Through the weight matrix and bias vector, further fuse and transform these features. For classification tasks, the number of neurons in the output layer is equal to the number of classes. The softmax activation function is used to generate the probability distribution for each class, indicating the likelihood of the input sample belonging to each class. For regression tasks, the output layer has only one neuron, and a linear activation function is applied to output the predicted value.

Decision Making and Evaluation: In classification tasks, select the class with the highest probability as the prediction result. In regression tasks, directly output the predicted value. During the training process, compare the prediction results with the true labels. For classification tasks, use the cross-entropy loss function to measure the difference between the

predicted probability distribution and the true distribution. For regression tasks, use the mean squared error loss function to evaluate the accuracy of the predicted values. These loss functions provide a quantitative measure of the model's performance and guide the training process to improve the model's accuracy.

4) Model Optimization: Calculating Loss and Backpropagation: Based on the calculation result of the loss function, use the backpropagation algorithm to compute the gradients of the loss function with respect to each parameter (weights and biases) in the network. Due to the relatively complex structure of the LSTM model, the backpropagation process needs to handle the gradient calculation of the memory units and gating mechanisms carefully. This requires a deep understanding of the internal workings of the LSTM model to ensure accurate gradient calculation and parameter update.

Parameter Update and Optimization: Select an appropriate optimizer, such as Stochastic Gradient Descent (SGD), Adagrad, Adadelta, or Adam. According to the calculated gradients and the set hyperparameters, update the network parameters. In addition, optimize the model's performance by adjusting hyperparameters, such as the learning rate, the number of hidden units, and the number of LSTM layers, as well as modifying the model structure, such as adding or removing LSTM layers or fully connected layers. These adjustments can help the model better fit the data and improve its generalization ability.

5) Model Evaluation and Application: Model Evaluation: Evaluate the trained LSTM model using a validation set or a test set. For classification tasks, calculate metrics such as accuracy, recall, and F1-score to measure the model's performance in correctly classifying samples. For regression tasks, calculate metrics such as the mean squared error (MSE) and the mean absolute error (MAE) to assess the accuracy of the predicted values. These metrics provide a comprehensive evaluation of the model's performance on new data and help determine whether the model is suitable for practical applications.

Model Application: Apply the trained and evaluated LSTM model to practical time series classification or prediction tasks. Use the model to classify or predict new, unseen time series data. For example, in the field of finance, the model can be used to predict stock price trends; in industrial applications, it can be used to predict equipment failures. By applying the model to real-world scenarios, its practical value and effectiveness can be verified.The specific steps of the CNN algorithm are as follows:

Algorithm 2 LSTM Training Algorithm

Input: Input Sequence $X = \{x_1, x_2, \ldots, x_T\}$, Hidden Size h, Initial Hidden State $h_0 = 0$, Initial Cell State $c_0 = 0$
Output: Output Sequence $H = \{h_1, h_2, \ldots, h_T\}$

Initialize: Weights W_i, W_f, W_c, W_o and Biases b_i, b_f, b_c, b_o
$h_{t-1} \leftarrow h_0$, $c_{t-1} \leftarrow c_0$ ▷ Initial states
$H \leftarrow []$ ▷ Output sequence
for $t \leftarrow 1$ **to** T **do**
    Compute input gate: $i_t \leftarrow \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
    Compute forget gate: $f_t \leftarrow \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
    Compute candidate cell state: $\tilde{c}_t \leftarrow \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$
    Update cell state: $c_t \leftarrow f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$
    Compute output gate: $o_t \leftarrow \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$
    Update hidden state: $h_t \leftarrow o_t \odot \tanh(c_t)$
    Append output: $H \leftarrow H \cup \{h_t\}$
end for
return H

C. CNN-LSTM model

The EGG signal is a typical time series signal, characterized by its spatio-temporal correlation and instability. CNN can effectively extract spatial features from EGG signals, while LSTM are adept at capturing their temporal features. To classify the degree of PTSD suffered by rescue workers, we employed a hybrid CNN-LSTM network model. This model combines a convolutional neural network with a long short-term memory network, skillfully handling both the temporal and spatial features in EEG signals, and has achieved excellent performance in EEG processing [19].

The combination of CNN and LSTM primarily presents two architectures: series and parallel. Compared to parallel architectures, series architectures demonstrate superior performance in modeling time series data. The working mechanism involves initially using CNN to efficiently extract features from the input data, followed by passing these informative feature sequences to the LSTM layer for further time series analysis and modeling. This sequential processing not only helps to capture temporal dependencies in the data more accurately but also enhances the overall performance of the model. Additionally, series architectures generally exhibit better generalization because they mitigate the risk of overfitting by processing data at different levels separately. It is important to note that the feature sequence extracted by the CNN is sparsified to some extent before being passed to the LSTM, and this step is crucial for reducing the model's over-dependence on training data. Therefore, the series architecture is chosen to construct the model in this paper.

The model is composed of four main components. The first component is the feature input, which takes in the four-dimensional feature map of the differential entropy for each segment of the preprocessed EEG signal. The second component focuses on extracting spatial features from the EEG signals, utilizing a convolutional neural network to capture spatial information from the signals of each time period. The third component is dedicated to extracting temporal information, where a long short-term memory network is employed to discern hidden temporal features from the output of the

convolutional neural network. Finally, the fourth component involves the fully connected layer and the SoftMax layer, which are used to perform the final classification of the EEG signals.The structure of the model is shown in Fig 4:

The specific steps of the CNN-LSTM algorithm are as follows:

Algorithm 3 CNN-LSTM for Sequence Processing

Input: Input Sequence $X = \{x_1, x_2, \ldots, x_T\}$, Number of Classes C, Learning Rate η, Epochs E
Output: Trained Model Parameters

Initialize: CNN Weights W_{cnn}, Biases b_{cnn}
Initialize: LSTM Weights W_{lstm}, Biases b_{lstm}
Initialize: Output Layer Weights W_o, Bias b_o
for $e \leftarrow 1$ **to** E **do**
  for $x \in X$ **do**
    CNN Feature Extraction:
    $h_{cnn} \leftarrow x$
    for Convolutional Layer l **do**
      $h_{cnn} \leftarrow \text{ReLU}(\text{Conv}(h_{cnn}, W^l_{cnn}) + b^l_{cnn})$
      $h_{cnn} \leftarrow \text{MaxPool}(h_{cnn})$
    end for
    $h_{cnn} \leftarrow \text{Flatten}(h_{cnn})$
    LSTM Sequence Processing:
    $h_{lstm} \leftarrow \text{LSTM}(h_{cnn}, W_{lstm}, b_{lstm})$
    Output Layer:
    $y \leftarrow \text{Softmax}(W_o \cdot h_{lstm} + b_o)$
    Compute Loss L and Gradients ∇L
    Update Parameters: $W_{cnn} \leftarrow W_{cnn} - \eta \cdot \nabla W_{cnn}$
    Update Parameters: $W_{lstm} \leftarrow W_{lstm} - \eta \cdot \nabla W_{lstm}$
    Update Parameters: $W_o \leftarrow W_o - \eta \cdot \nabla W_o$
  end for
end for
return Trained Parameters W_{cnn}, W_{lstm}, W_o

To adapt the data to the dimensional requirements of CNN and LSTM, we employed the reshape function to process the four-dimensional features, expanding the dimensions of the EEG data to (samples, time_steps, height, width, channels). Initially, the resolution of the DE features after processing the raw data was 1000200. However, upon incorporating CNN and LSTM, the data feature dimensions expanded significantly. Consequently, we reduced the resolution of the DE features to 20040, thereby lowering the computational cost and saving time.

In the CNN component, the model utilizes a dual Conv2D layer to extract the spatial features from each sample data. To help prevent overfitting when training with small sample sizes, a Dropout technique is applied after the dual CNN. The shape of the convolution kernel for Conv2D is defined as (height, width, channels), where height and width represent the dimensions of the convolution kernel, and channels denote the number of input data channels. In this experiment, the image data is in RGB format, with the number of input channels being 3 (red, green, and blue). The shape of the convolution kernel for the first Conv2D layer is set to (3,3), meaning the height and width of the convolution kernel are each 3 pixels. Sixteen 33 convolution kernels are employed to extract features from the input image, that is, filters=16, and the input image is processed through these filters. Note that the convolutional layer will produce 16 feature maps. The ReLU function is employed as the activation function for the first Conv2D layer. For each time step, the input to the convolutional layer has the shape (time_steps, height, width, channels), and the output from the convolutional layer has the shape (25, 20, 38, 198, 16). This output is then fed into the pooling layer, which utilizes max pooling. For the convolutional output at each time step, a $2x2$ pooling window is applied, reducing the height and width by half. The resulting shape after max pooling is (25, 20, 19, 99, 16). The output from the first Conv2D pooling layer is then passed into the convolutional layer of the second Conv2D, which also uses a $3x3$ convolution kernel, and the activation function remains the ReLU function.After the convolution operation, the data dimensions are (25, 20, 17, 97, 32). The output from the convolutional layer is fed into the pooling layer for max pooling, using a $2x2$ pooling window. Consequently, the data dimensions after the pooling layer become (25, 20, 8, 48, 32). Following the pooling step, the Flatten operation is applied to flatten the data, converting the 3D feature map of each time step into a 1D feature vector. The resulting output dimension is (25, 20, 15360), meaning that the 3D data of each time step is transformed into a 1D format.

In the LSTM component, the model employs a two-layer LSTM to extract temporal features. The input data for the LSTM is the output from the two-layer CNN. The first LSTM layer processes the 15360-dimensional features at each time step and converts them into 64-dimensional hidden states, resulting in an output shape of (25, 20, 64). The output from the first LSTM layer is then fed into the second LSTM layer, which further transforms the 64-dimensional hidden states at each time step into the final 32-dimensional states. The final output has a shape of (25, 32).

The data processed by the CNN and LSTM are fed into the fully connected layer. This layer converts the 32-dimensional input into an 8-dimensional output, using ReLU as the activation function. The data type processed by the first fully connected layer is (25, 8). Subsequently, the data is passed into the second fully connected layer, where the 8-dimensional input is transformed into a 4-dimensional output representing class probabilities. The SoftMax function is employed as the activation function to classify the EEG signals, resulting in a final data shape of (25, 4). The Adam optimizer is used, with a learning rate set to 0.0001 during the training process. The batch size for the LSTM layer is set to 128. To control overfitting during training, a Dropout layer is introduced, and an early stopping condition is added. The model will stop training if there is no improvement after 10 iterations.

D. Evaluation Criteria

To more scientifically evaluate the classification performance of the network model, we employ accuracy, precision, recall, and F1 Score as the evaluation metrics for the model.It is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \tag{10}$$

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{11}$$

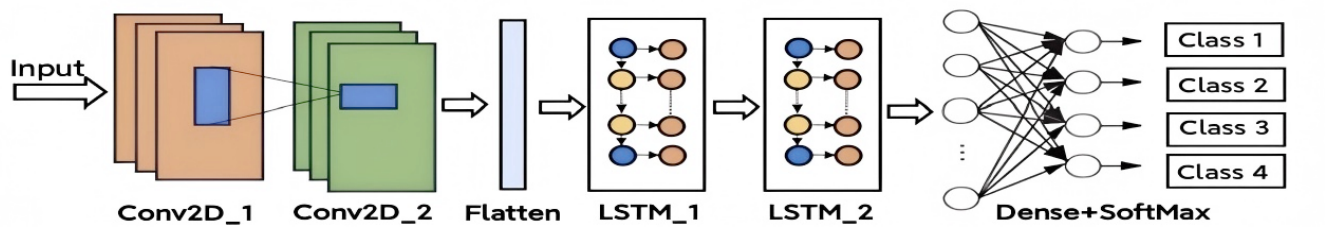

Fig. 4. Structural diagram of CNN-LSTM

TABLE II
THE RESULTS OF THE ABLATION EXPERIMENT

Model	Accuracy	Precision	Recall	F value
CNN	72%	51.64%	50.72%	51.18%
LSTM	76%	53.28%	51.39%	52.32%
CNN-LSTM	80%	55.26%	57.32%	56.27%

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{12}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \tag{13}$$

TP represents true positives, which is the number of positive samples that the model correctly predicts as positive. TN denotes true negatives, which is the number of negative samples that the model correctly predicts as negative. FP stands for false positives, which is the number of negative samples that the model incorrectly predicts as positive (also known as false alarms). FN indicates false negatives, which is the number of positive samples that the model incorrectly predicts as negative (also known as misses).

IV. Experimental results and analysis

A. Ablation experiment

To demonstrate the superiority of this model, we conducted ablation experiments on this dataset. By successively removing a module and comparing the resulting classification performance, we verified the effectiveness of the corresponding module used in the model. This primarily involved removing the LSTM layer while retaining only the CNN layer, allowing us to assess the capability of the CNN layer alone in capturing sequence information. Conversely, we also removed the CNN layer and retained the LSTM layer to evaluate the role of the LSTM layer in capturing sequence information. The results of the ablation experiments are shown in Fig.5 and TABLE II:

The table above presents the experimental results for models using only CNN, only LSTM, and the hybrid CNN-LSTM neural network. From the results, it is evident that the model using only CNN performs the worst in terms of accuracy, precision, and other metrics, while the hybrid CNN-LSTM model achieves the best performance. CNN excels at extracting spatial features but is less sensitive to temporal features. On the other hand, LSTM has significant advantages in processing time series data and can accurately capture long-term dependencies in EEG signals. By combining the strengths of both CNN and LSTM, the CNN-LSTM model is capable of extracting spatial features and capturing temporal features in EEG signals. This combination enables the CNN-LSTM model to perform well in EEG recognition tasks. When dealing with complex EEG signals, CNN-LSTM can more comprehensively capture the useful information within the signals, thereby improving the accuracy and robustness of recognition.

B. Loss function and Accuracy

In classification problems, the loss function is a critical metric for assessing a model's performance. In this experiment, the model utilizes Categorical Crossentropy as its loss function. This is a widely used loss function in multi-class classification tasks, which measures the difference between the probability distribution predicted by the model and the true label's probability distribution.

The experimental results are shown in Fig 6, which shows the trend plot of the loss values as the training iterations goes deeper in the training set and the validation set. At the beginning of training, the loss on both the training set and the validation set decreases sharply, which indicates that the performance of the model is improving and the model fits the training data better and better. Accuracy is another crucial metric for evaluating the performance of a model. We have conducted experiments to assess the model's accuracy, and the experimental results are shown in Fig 7.

Similar to the loss function, the model's accuracy increases significantly at the beginning of training. However, after 25 iterations, the rate of improvement slows down and eventually plateaus. The training set's accuracy can reach 80%, while the validation set's accuracy exceeds 60%, indicating that the model's accuracy is satisfactory

C. Confusion matrix

A Confusion Matrix is a specific table layout used to visualize the performance of algorithms, particularly in classification tasks within supervised learning. It helps us understand the relationship between the predicted results of a classification model and the actual outcomes. The classification report derived from the confusion matrix serves as an evaluation metric for assessing the performance of the CNN-LSTM model in classification tasks. As illustrated in Fig 8, the confusion matrix displays both accurately and inaccurately predicted data points. The sum of samples in each column of the confusion matrix represents the total number of samples predicted for that category, while the sum of samples in each row indicates the actual number of samples belonging to that category. The values on the diagonal represent the number of samples correctly predicted as instances of their

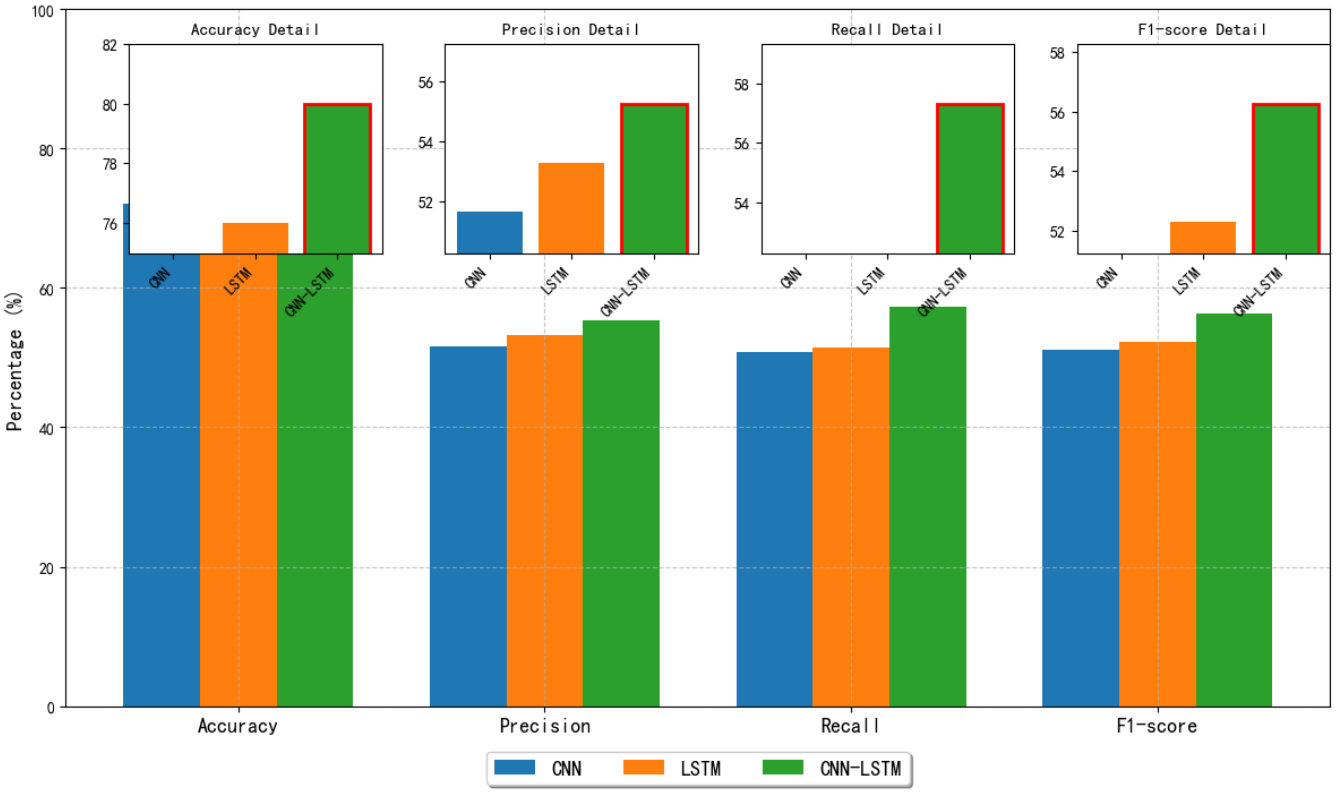

Fig. 5. The results of the ablation experiment

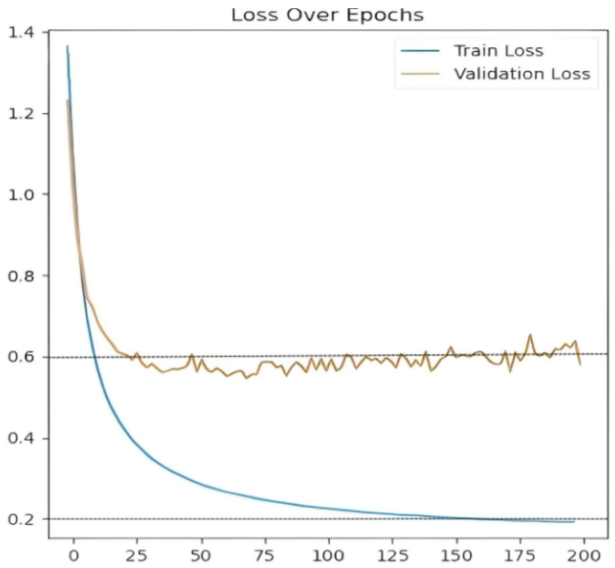

Fig. 6. Loss functions on training sets and validation sets

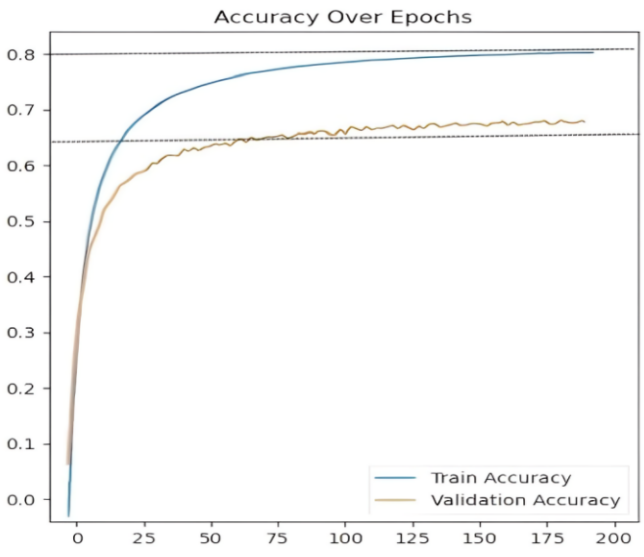

Fig. 7. Loss functions on training sets and validation sets

respective categories. The results indicate that the CNN-LSTM model achieves higher classification accuracy across all four categories, with Class 1 demonstrating the best performance.

D. Comparing machine learning models

Many machine learning algorithms have shown promising results in the detection and diagnosis of PTSD. To thoroughly verify the superior performance of the CNN-LSTM model proposed in this study [20], we conducted a detailed comparative analysis of the experimental results against other commonly used network models. This step is designed to further highlight the advantages of the CNN-LSTM model in the specific application scenario of this experiment by comparing the performance metrics of different models. The specific results are shown in TABLE III:

From the comparison of model performance presented in the table above, it is evident that the hybrid neural network model, CNN-LSTM, studied in this paper, achieves an accuracy level that surpasses other common models. Additionally, it demonstrates excellent performance in precision, recall, and F1 score.

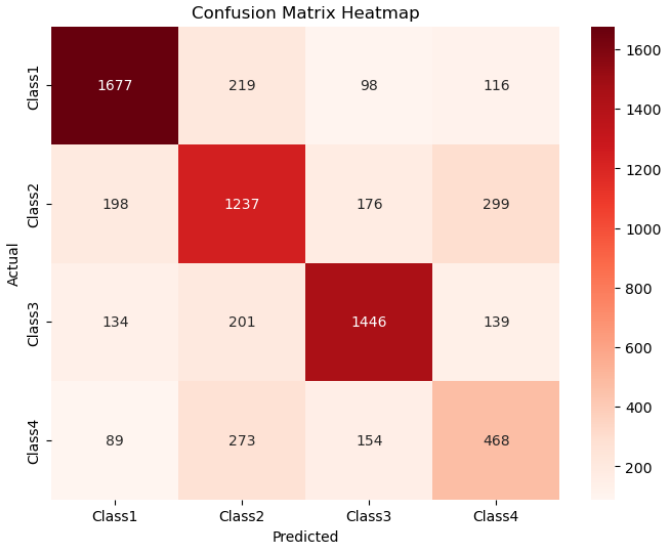

Fig. 8. Confusion matrix of CNN-LSTM model

TABLE III
PERFORMANCE OF MODELS ON THIS CLASSIFICATION PROBLEM

Model	Accuracy	Precision	Recall	F value
SVM	48%	33.45%	34.64%	33.87%
KNN	46%	31.74%	27.02%	25.95%
RF	60%	53.32%	50.84%	51.29%
CNN - LSTM	80%	55.26%	57.32%	56.27%

V. CONCLUSION

The CNN-LSTM model is designed by combining the strengths of convolutional neural networks in spatial feature extraction and long short-term memory networks in time series modeling. To fully leverage the key information of time, space, and frequency in EEG signals, a four-dimensional feature structure incorporating this information is devised. Initially, the preprocessed four-dimensional feature structure is fed into the CNN layer to automatically capture its spatial features. The output from the CNN is then passed to the LSTM layer to further extract temporal features. Finally, the classification task is completed through the fully connected layer and the SoftMax layer. Compared to other hybrid models, the CNN-LSTM model demonstrates superior performance. An ablation experiment was conducted, and the results indicate that the hybrid model outperforms the single models. The results were visually analyzed, and a confusion matrix was designed. The confusion matrix effectively illustrates the model's prediction accuracy, particularly for Class 1. In summary, the model in this experimental study has achieved promising results in predicting PTSD.

REFERENCES

[1] B. Runciman, "Treating PTSD and PHOBIAS and MORE," in *ITNOW*, vol. 56, no. 4, pp. 63-63, Dec. 2014.

[2] V. Rozgic, A. Vazquez-Reina, M. Crystal, A. Srivastava, V. Tan and C. Berka, "Multi-modal prediction of PTSD and stress indicators," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, 2014, pp. 3636-3640.

[3] J. A. Cruz, J. C. Marquez, A. M. Mendoza, J. I. Reyes, V. C. Tango and S. V. Prado, "EEG-based Characterization and Classification of Severity for the Diagnosis of Post-Traumatic Stress Disorder (PTSD)," in *2023 5th International Conference on Bio-engineering for Smart Technologies (BioSMART)*, Paris, France, 2023, pp. 1-7.

[4] A. R. Mashhadi, M. H. Moradi and Z. Ghanbari, "Staging combat-related PTSD," in *2021 28th National and 6th International Iranian Conference on Biomedical Engineering (ICBME)*, Tehran, Iran, Islamic Republic of, 2021, pp. 178-181.

[5] Y. Li et al., "EEG-based Emotion Recognition Under Convolutional Neural Network with Differential Entropy Feature Maps," in *2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, Tianjin, China, 2019, pp. 1-5.

[6] W. -L. Zheng, J. -Y. Zhu and B. -L. Lu, "Identifying Stable Patterns over Time for Emotion Recognition from EEG," in *IEEE Transactions on Affective Computing*, vol. 10, no. 3, pp. 417-429, 1 July-Sept. 2019.

[7] T. F. Quatieri et al., "An Emotion-Driven Vocal Biomarker-Based PTSD Screening Tool," in *IEEE Open Journal of Engineering in Medicine and Biology*, vol. 5, pp. 621-626, 2024.

[8] Chao H, Dong L, Liu Y, Lu B. "Emotion Recognition from Multiband EEG Signals Using CapsNet," *Sensors (Basel)*. 2019 May 13;19(9):2212.

[9] Y. Li, "An Intelligent Computer-Aided Analysis Framework for the Impact of PTSD on Rescuers based on EEG Signal Visualization Algorithm," in *2023 International Conference on Inventive Computation Technologies (ICICT)*, Lalitpur, Nepal, 2023, pp. 1558-1561.

[10] P. Annapureddy et al., "Predicting PTSD Severity in Veterans from Self-reports for Early Intervention: A Machine Learning Approach," in *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, Las Vegas, NV, USA, 2020, pp. 201-208.

[11] S. M. Makhdoomi, M. Rakhra, D. Singh and A. Singh, "Artificial-Intelligence based Prediction of Post-Traumatic Stress Disorder (PTSD) using EEG reports," in *2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, Uttar Pradesh, India, 2022, pp. 1073-1077.

[12] A. Beykmohammadi, Z. Ghanbari and M. H. Moradi, "PTSD Diagnosis using Deep Transfer Learning: an EEG Study," in *2022 29th National and 7th International Iranian Conference on Biomedical Engineering (ICBME)*, Tehran, Iran, Islamic Republic of, 2022, pp. 9-13.

[13] R. S. Deshmukh, V. Jagtap and S. Paygude, "Facial emotion recognition system through machine learning approach," in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2017, pp. 272-277.

[14] H. Avula, R. R and A. S Pillai, "CNN based Recognition of Emotion and Speech from Gestures and Facial Expressions," in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, Coimbatore, India, 2022, pp. 1360-1365.

[15] K. K. Dutta, "Multi-class time series classification of EEG signals with Recurrent Neural Networks," in *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, Noida, India, 2019, pp. 337-341.

[16] S. F. Pane, J. Ramdan, A. G. Putrada, M. N. Fauzan, R. M. Awangga and N. Alamsyah, "A Hybrid CNN-LSTM Model With Word-Emoji Embedding For Improving The Twitter Sentiment Analysis on Indonesia's PPKM Policy," in *2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Yogyakarta, Indonesia, 2022, pp. 51-56.

[17] T. Yamunarani et al., "Analysis of Post-Traumatic Stress Disorder (PTSD) Using Electroencephalography," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Kamand, India, 2024, pp. 1-6.

[18] K. S. Gill, V. Anand, R. Chauhan, A. Choudhary and R. Gupta, "CNN, LSTM, and Bi-LSTM based Self-Attention Model Classification for User Review Sentiment Analysis," in *2023 3rd International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, Bangalore, India, 2023, pp. 1-6.

[19] M. N. Shahzad, H. Ali, T. Saba, A. Rehman, H. Kolivand and S. A. Bahaj, "Identifying Patients With PTSD Utilizing Resting-State fMRI Data and Neural Network Approach," in *IEEE Access*, vol. 9, pp.

[20] I. J. Dristy, A. M. Saad and A. A. Rasel, "Mental Health Status Prediction Using ML Classifiers with NLP-Based Approaches," in *2022 International Conference on Recent Progresses in Science, Engineering and Technology (ICRPSET)*, Rajshahi, Bangladesh, 2022, pp. 1-6.

Zhenhua Qu Zhenhua Qu was born in Weifang, Shandong Province in 2001. The author is a 2023 master student of Shandong University of Technology, majoring in statistics.

The author's current major interests include deep learning, image recognition, statistical learning, and intelligent algorithms.

Weipu Wu Weipu Wu was born in Yancheng, Jiangsu Province in 2001. The author is a 2023 master student of Shandong University of Technology, majoring in applied statistics.

The author's major interests are machine learning, intelligent algorithms, data mining, and time series.

Yuan Cao Yuan Cao was born in 1987 in Zibo City, Shandong Province. The author is an associate professor at the School of Mathematics and Statistics, Shandong University of Technology, Zhangdian District, Zibo City, Shandong Province.

The author has been engaged in data mining and intelligent information processing research since 2010.

Participated in the completion of 2 projects of the National Natural Science Foundation, 1 project of the Doctoral program of the Ministry of Education, and now presides over one project of the National Natural Science Foundation Youth Science Fund, one project of the Shandong Natural Science Foundation Doctoral Fund, and one key project of the Hunan Key Laboratory open project. Hubei Province key laboratory open subject general project. He has published (including receiving) more than 20 research papers in important academic journals and conferences at home and abroad, including 18 papers indexed by SCI and 4 papers indexed by EI.