A Comparative Performance Analysis of CNN, LSTM, and CNN-LSTM Models for Classifying Sleep Stages Using Electroencephalography Signals

Zainab N. Nemer[1], Saba Abdual Wahid Saddam[2], Rana J.AL_Sukeinee[3], Esra'a J. Harfash[4]
[1,2,4]Department of Computer Science, College of Computer Science and Information Technology, University of Basra, Iraq.
[3]Department of Physics Science, College Science, University of Basra, Iraq
E-mail: zainab.nemer@uobasrah.edu.iq , Saba.Saddam@uobasrah.edu.iq , rana.jabbar@ uobasrah.edu.iq ,esra.harfash@uobasrah.edu.iq

Keywords: LSTM networks, convolutional neural networks (CNN), sleep stage classification, hybrid models, EEG signals

Received: January 16, 2025

This paper aims to study and analyze the performance of CNN, LSTM and CNN+LSTM deep neural networks individually in the problem of sleep cycle stage classification, by analyzing the recorded EEG signals, and highlight the strengths of each model in this classification problem. The deep learning models selected in this work are reliable models in the world of experiments and medical classification. The data used here is the EEG signal represented by the sleep-edf-expanded-npzs database which carries information about the different sleep stages, which are five stages: wakefulness (W), non-REM stage 1 (N1), non-REM stage 2 (N2), non-REM stage 3 (N3), and rapid eye movement (REM). This data was split by training by 70% and testing by 30%. The models were trained based on cross-entropy loss and Adam optimizer. The results obtained in this study showed that the classification rates are as follows: CNN model accuracy (0.8736%), LSTM model accuracy (0.8306%), and LSTM + CNN hybrid model accuracy (0.8813%). These results indicate the good performance of the CNN model, which can be attributed to its strong ability to extract spatial features from data. The results of the LSTM approach demonstrate its ability to track and interpret the temporal characteristics of the sleep signal. For the CNN-LSTM model, which combines the strengths of temporal LSTM and spatial CNN, the result of combining features in this sequence showed superior results than either CNN or LSTM alone in processing sleep signals. Statistical measures were used to validate the results obtained. The F-statistic was 949.78666666523 and the probability value was 6.119863 e-14, which validate the results obtained for each model.

Povzetek:

1 Introduction

Sleep is essential for children, adolescents, and adults' health and well-being and is a biological necessity. It is essential for mood, emotional well-being, cardiovascular health, and brain health. Restoring sleep is strongly associated with improved physical, mental, and cognitive health as well as a lower chance of accidents and injuries brought on by weariness and drowsiness. Sleep is an essential component of the human daily routine. On the other hand, inadequate or disrupted sleep can result in a decline in general physical health as well as possible cognitive and psychological damage. [1][2].

The American Academy of Sleep Medicine (AASM) has classified wakefulness (W), rapid eye movement (REM), and non-rapid eye movement (NREM) sleep of increasing depth into N1, N2, and N3 as the five stages of wake and sleep. The body typically goes through these phases four to six times, with an average of ninety minutes spent in each step [3][4].

Although there are several ways to evaluate the brain's stages of sleep, electroencephalography (EEG) is still the preferred tool for studying human sleep [5]. In sleep research, electroencephalography (EEG) is the most often utilized technology. Sample data are given for a normal EEG examination conducted at night while REM (rapid eye movement) and NREM sleep are occurring [6].

With the availability of sleep EEG data, many researchers in the field of machine learning, specifically in the field of deep learning, have practical experience in sleep classification through different stages of methods and techniques. Among these networks is the LSTM model, as LSTM structures have the ability to learn long-term based on sequential data, which makes them well suited for tasks such as predicting time series such as sleep signal. Therefore, they are relied upon to solve special problems in computational biology and computational biology. [7][8].

Convolutional neural networks (CNNs) are very effective at capturing spatial features and have shown excellent results in detecting complex patterns within EEG data. CNNs and LSTMs are also widely used in EEG signal analysis. Performance in time series classification problems can be improved by combining CNNs and LSTMs (CNN-LSTM hybrid model), While CNNs can learn local patterns in the data, LSTMs can identify long-term dependencies in sequential data. Combining CNNs and LSTMs exploits spatial and temporal patterns [9,10].

Most studies have relied on one of the above-mentioned models individually, namely CNN, LSTM, or CNN-LSTM, and some of these studies have relied on the Sleep-EDF database or other databases. However, no study has compared these three models on the same database. Most previous studies have evaluated these models individually or using different EEG datasets.

This study aims to bridge this gap by presenting a study that evaluates and compares the effectiveness of three models: LSTM, CNN, and a hybrid model (CNN, LSTM)—in automatically classifying sleep stages, which involves conducting a comparative study among these three models using the same EEG signal database. . The study aims to:

1. Identify which of these models achieves the highest performance according to the same criteria, in terms of the Sleep-EDF database and processing methods.
2. Provide a comprehensive systematic evaluation to determine the best-performing model.
3. Identify whether standalone CNN or LSTM models, or a hybrid CNN with LSTM model, provide superior performance.
4. Also, test which hybrid architecture (CNN_LSTM) or (LSTM_CNN) is better.

Through multiple experiments, it was found that the hybrid architecture performs better, especially the LSTM_CNN architecture. Furthermore, the CNN model is capable of providing superior results, given its limited capabilities to spatial features, while LSTM also demonstrated good results.

The study's objectives are clearly outlined, centering on a comparison of performance and an identification of each model's strengths in sleep stage classification.

The rest of the study is organized as follows. Section 2 provides a literature review on sleep stage classification. Typical materials and methods are covered in Section 3. Implementation and results discussion are presented in Section 4. Study Conclusion and Recommendations are outlined in Section 5.

2 Literature review

Numerous researchers have attempted to address and enhance the effectiveness of sleep stage classification through a variety of concepts and approaches; however, deep learning algorithms are among the most significant approaches that have explored this area in order to accomplish automated sleep stage recognition. The study conducted by Nicola M et al. (2019) to automatically score sleep stages using EEG signals from a single channel, a unique cascaded recurrent neural network (RNN) architecture based on long short-term memory (LSTM) blocks is proposed. And to choose the most pertinent features, fifty-five temporal and frequency-domain features were taken out of the EEG signals and supplied into feature reduction algorithms.[11].

Convolutional Neural Networks (CNNs) is important models in sleep stage classification problem. Asma et al. (2020) study highlights a 13-layer 1D convolutional neural network CNN for automatic feature extraction and sleep stage classification using single-channel EEG signal [12].

Furthermore, Zhao et al. (2022), They proposed an algorithm suitable for different physiological signals that can achieve automatic sleep stage from start to finish without any manual feature extraction. Here, one-dimensional convolutional neural network and long short-term memory are adopted. This method can automatically segment sleep into 5 stages including wakefulness, non-rapid eye movement sleep (N1~N3) and rapid eye movement sleep using EEG signals. [13].

The CNN model has a distinct role in classifying sleep stages, Luis A. et al. (2023), in this paper, are used a convolutional neural network (CNN) based on 5- and 2-class models to study the performance of automatic sleep stage categorization utilizing autonomously selected characteristics from electroencephalogram (EEG) inputs. We established two 2-class sleep stage classification techniques and evaluated how well they performed in comparison to the predictions from a 5-class model. The public ISRUC-Sleep dataset, which includes six EEG channels and 100 participants, was used for tests. All models were constructed using a CNN called EEGNet. [14].

In order to address the issue of training data imbalance, Enes E. A. et al. (2023), in this study suggests a novel hybrid CNN+LSTM neural network design that makes use of focus loss and discrete cosine transform techniques. Using k-fold cross-validation techniques (subject-wise), the model was trained on four distinct databases. When using two channels (EEG-EOG), the best accuracy was 87.11%, [15].

Nowadays, many existing techniques depend on handcrafted features. Only a few methods, meanwhile, are able to recover the temporal information required to determine the stages of sleep. LSTMs will be able to learn transition rules, while convolutional neural networks will be able to extract time-invariant features. A thorough analysis of recent advancements was given by Kotla R. et al. (2023), This paper proposes a deep learning model to automatically score sleep stages using single-channel EEG recordings that combines CNN and LSTM. The model was trained and tested on the Sleep-EDF-v1 dataset, which is publicly available. The single-channel Fpz-Cz EEG was used and scored according to the AASM standard. An overall accuracy of 83.38% was achieved. [16].

Supratak et al. (2017), are used a two-step training technique to effectively train the DeepSleepNet deep learning network, which is introduced in this work for automatic sleep stage grading based on raw single-channel EEG. Different single-channel EEGs (F4-EOG (left), Fpz-Cz, and Pz-Oz) from two public sleep data sets are used to assess this concept [17].

In order to acquire task-specific filters for classification without requiring prior domain knowledge, Tsinalis et al. (2016) are used the convolutional neural networks (CNNs) to utilized in this work to automatically score sleep stages based on single-channel electroencephalography (EEG). In the CNN's stochastic gradient descent (SGD) optimization, class-balanced random sampling was used to prevent performance bias in favor of the most representative sleep stages [18].

Pham et al. (2023) was presented an automatic sleep stage categorization system utilizing a combination of transformer, long-short-term memory (LSTM), and convolutional neural network (CNN) models using single-channel EEG recordings [19].

To overcome the critical limitations Casciola et al. (2021) are presented a deep learning (DL) model for automated sleep staging of HB EEG data. The solution includes a simple band-pass filtering, a data augmentation step, and a model using convolutional (CNN) and long short-term memory (LSTM) layers [20]

A new model dubbed CAttSleepNet was presented by Li et al. (2022) for automatic sleep stage identification using single-channel EEG, and incorporate an attention module into the convolutional neural network (CNN) so that it can use the contextual information inside the epoch to learn the weights of local sequences of EEG signals. The global connections of succeeding epochs are then encoded using a two-layer bidirectional long-short-term memory (Bi-LSTM) [21].

Table 1: Summary information of existing method

Study	**Dataset**	**Model (Network)**	**Evaluation Metrics**	**Accuracy**
Nicola M. et al. (2019)	Use of a publicly available sleep-EDF database.	RNN with LSTM	Accuracy and Confusion Matrix	The accuracy of classification for *five* sleep stages is 86.7%
Asma et al. (2020)	sleep-EDF dataset	1D CNN (13-layer)	k-fold cross-validation.set k to 20	Precision of 94.09%, 74.73%, 96.43%, and 71.02%, respective of classifying five sleep stages.
Zhao et al. (2021)	Sleep-EDF, ISRUC	1D CNN-LSTM	Accurcy, Precision, Recall, F1-score	The accuracy of staging is 93.47% using the Fpz-Cz electroencephalogram signal. When using the Fpz-Cz and electroencephalogram signal, the highest accuracy o94.15%.
Luis A. et al. (2023)	ISRUC-Sleep dataset)	CNN (EEGNet)	5-class and 2-class classification accuracy	In the best case, the average AUROC of 0.964,0.967,0.982 and 0.929 for the stratified 2-class models
Enes E. A. et al. (2023)	DRM-SUB, ISRUC3, and SleepEDF-20 databases.	hybrid CNN-LSTM	k-fold validation strategies (subject-wise)	highest score was 87.11% accuracy, 81.81% Kappa score, and 79.83% MF1 when using two channels (EEG-EOG)
Kotla R. et al. (2023)	Sleep-EDF-v1 Dataset	single-channel EEG recordings (CNN +LSTM)	Classification accuracy	83.38% accuracy using single-channel Fpz-Cz EEGs.
Supratak et al. (2017)	Sleep-EDF and MASS	CNN-LSTM	Classification accuracy	(MASS: 86.2%-81.7, Sleep-EDF: 82.0%-76.9) .
Tsinalis et al. (2016)	sleep PSG dataset	CNN	Classification accuracy	Accuracy for each sleep stage (82%, range 80–84%), total accuracy (74%, range 71–76%) for all patients, and average F1 scores (81%, range 79–83%).
Pham et al. (2023)	ISRUC	CNN-LSTM	Classification accuracy	The experimental evaluation on the ISRUC S1 and S3 sleep

				datasets, It attains 82.40% and 80.37% accuracy, respectively..
Casciola et al. (2021)	electroencephalography (EEG) headbands (HB)	CNN-LSTM	Classification accuracy	74% (±10%) validation accuracy on low-quality two-channel EEG headband data and 77% (±10%) on gold-standard PSG.
Li et al. (2022)	Sleep-EDF	CNN-BiLSTM with Attention Mechanism	Accuracy(k-fold cross-validation.)	Achieving an accuracy of 86.9%.

3 Materials and Methods

3.1. Model Architectures and Mathematical Operations

In this work, research was conducted to know the strength of each model of LSTM, CNN and CNN+LSTM in classifying the different stages of sleep signal according to the features that each model possesses. The following will include the mathematical aspect of building each model with an explanation of the architecture that was adopted in this work..

3.1.1 LSTM Model

Long short-term memory (LSTM) has revolutionized both the fields of machine learning and neural computing. One of the reasons for the success of this recurrent network is its ability to handle the exploding/vanishing gradient problem, which is a difficult problem to overcome when training recurrent or very deep neural networks[22]. LSTMs possess the capacity to process sequential data and retain information from previous steps in the sequence, enabling them to predict future steps effectively. This characteristic makes them highly suitable for tasks involving long-term dependencies [23]. Fig. 1, is illustrated the common LSTM unit is composed of a cell state (denoted by C_t), a forget gate (f_t), an input gate (i_t), and an output gate (o_t). The three gates regulate the flow of information into and out the channel and important information over arbitrary time intervals can be remembered[24]

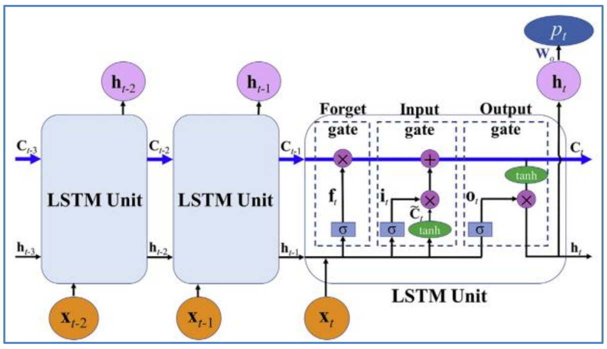


Figure 1: Structures long short-term memory (LSTM)[24]

The functions of a single cell LSTM are detailed in Algorithm , where σ denotes the logistic sigmoid function, tanh represents the hyperbolic tangent function, and $\odot$ denotes the Hadamard product, which is the element-wise product [25]:

Given:
1. W Matrix of input-to-hidden weights
2. U Matrix of hidden-to-hidden weights
3. b Vector of Bias

Input:
1. x_t Vector of input at time step t
2. c_{t-1} Vector of previous memory cell state
3. h_{t-1} Vector of previous hidden state

Output:
1. c_t Vector of memory cell state
2. h_t Vector of the hidden state

Process:
1. Compute the input gate vector:
$$i_t = \sigma\left(W^{(i)}x_i + U^{(i)}h_{t-1} + b^{(i)}\right) \quad (1)$$
2. Compute the forget gate vector:
$$f_t = \sigma\left(W^{(f)}x_i + U^{(f)}h_{t-1} + b^{f}\right) \quad (2)$$
3. Compute the output gate vector:
$$o_t = \sigma\left(W^{(0)}x_i + U^{(o)}h_{t-1} + b^{(0)}\right) \quad (3)$$
4. Compute the memory cell state vector:
$$c_{t=i_t} \odot u_t + f_t \odot c_{t-1} \quad (4)$$
where:
$$u_t = \tanh\left(W^{(u)}x_i + U^{(u)}h_{t-1} + b^{(u)}\right) \quad (5)$$
5. Compute the hidden state vector:
$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

3.1.2 CNN model

One type of deep learning algorithm that has shown great success in tasks like classification is CNN. Convolutional neural network (CNN) is the most advanced deep learning technique due to its ability to learn features independently [26]. Below are some of the advantages CNN has over other traditional neural networks [27]:

1. CNN's weight sharing feature, which lowers the number of trainable network parameters and helps the network improve generalization and prevent overfitting, is the primary justification for considering it.
2. The model output is very structured and heavily dependent on the retrieved features as a result of

learning the feature extraction layers and the classification layer simultaneously.

3. Compared to other neural networks, CNN makes large-scale network deployment considerably simpler.

Figur (2) show the Building block of a typical CNN[28]

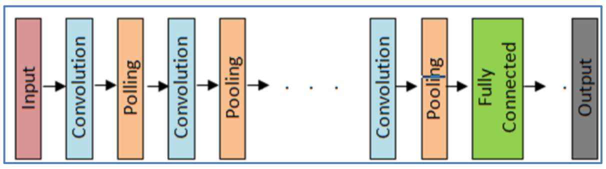


Figure 2: The Conceptual model of CNN

Below is an explanation of the formulas for the CNN steps:

N :is the Number of input features ,
x_i :is the Input data at position i ,
w_{ij} :the Weight connecting input i to output j ,
b_j: :Bias for the j-th output neuron and σ is the
a_i :is input (that resulted of provis layer)
z_i :is Output feature map
o_j :is the output of fully connected layer
σ : is the Activation function

1. Convolution Layers: The convolution layer (CONV) uses filters that perform convolution operations[29]:

$$z_i = b_i + \sum x_i \in W_{ij} * x_i \qquad (7)$$

After the convolution, an activation function such as the Rectified Linear Unit (ReLU) is applied to introduce non-linearity[30][31]:

$$g_i = ReLU(z_i) = \max(0, z_i) \qquad (8)$$

2. Pooling Layer : Pooling operation is an important operation in deep learning. Pooling operation can reduce the feature dimension, the number of parameters, the complexity of computation, and the complexity of time[32]: $P_i = max(z_{i:i+s})$

3. Fully Connected Layer: The fully connected layer (FC) operates on a flattened input where each input is connected to all neurons:

$$o_j = \sigma\left(\sum_{i=0}^{N-1} w_{ij}.a_i + b_j\right) \qquad (9)$$

4. Loss Function: This prediction error tells the network how o their prediction from the actual output, and then this error will be optimized during the learning process of the CNN model. This function calculate as[33]:.

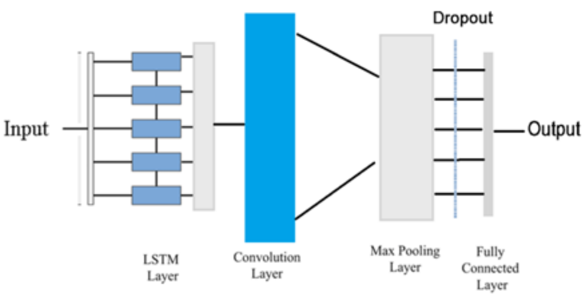
$$Loss = -\sum_{i=1}^{\substack{output\\size}} y_i.\log \hat{y}_i \qquad (10)$$

Where L is the loss value,M is the number of classes,y_i_ is the actual value of class i (either 1 or 0) and **y^$_i$** is the predicted probability for class **i**.

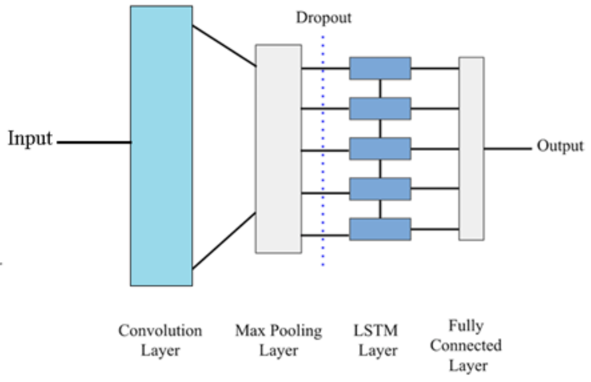
3.1.3 CNN+LSTM Model

Hybrid networks (CNNs, LSTMs) provide a solution for utilizing both structural and spatial information, combining the strengths of both spatial and temporal networks (CNNs). A CNN-LSTM network is used in a variety of problems, including activity recognition [34]. It is a combination of CNN layers that first extract spatial features, then feed these features as input to LSTM layers to provide time-sequence predictions.

In the LSTM-CNN architecture, the order is reversed: the EEG time series is first processed by the LSTM layers to model temporal dependencies, and the resulting features are then fed to the CNN for analysis by the CNN layers to extract higher-level spatial features. Figure 3 illustrates both hybrid models: (a) CNN-LSTM and (b) LSTM-CNN:



(a): CNN-LSTM MODEL [35]



(b): LSTM –CNN MODEL

Figure 3: The basic architecture of CNN-LSTM and LSTM-CNN

3.2. Dataset and Preprocessing

The analysis was performed on the Sleep-EDF database that is in the public domain, and it contains 150 files origin from multiple files of EEG signals from multiple dumplings. The files contain recordings of all sleep stages: Wake (W), Non-REM stages (N1, N2, N3), and REM.

3.2.1 Dataset Description

All EEG signals were stored in npz format ,and Each signal file contains the following key components:

- EEG Signal Data (x): Represents the brain's electrical activity recorded during the study.
- Labels (y): Corresponding annotations for the sleep stages, categorized into five distinct classes:
  1. Wake (W)
  2. Non-REM Stage 1 (N1)
  3. Non-REM Stage 2 (N2)
  4. Non-REM Stage 3 (N3)
  5. REM (Rapid Eye Movement)
- Sampling Frequency (fs): Defines the signal acquisition rate, typically set at 100 Hz.
- Channel Labels (ch_label): Specifies electrode placements, such as (EEG Fpz-Cz) and (EEG Pz-Oz).

Example of a file**:** The file "SC4001E0.npz" includes:

- Shape of x: (841, 3000, 1), which indicates 841 signal segments, each containing 3000 samples (30 seconds per segment at 100 Hz).
- Shape of y: (841,), which contains the corresponding sleep stage labels for each segment.
- Sample Signals:

```
[[[  8.111 ],
  [ 17.488 ],
  [ 21.239 ],
  ...
  [-10.361 ],
  [-11.112 ],
  [ -2.109 ]],
 [[-10.736 ],
  [-11.393 ],
  [ -4.454 ],
  ...
  [ 58.842 ],
  [ 48.339 ],
  [ 53.684 ]]]
```

After the file EEG data is read, the labels are extracted from it, and all the signals are collected into a single array. Also the labels are merged into a single array.

3.2.2 Preprocessing Steps

The EEG signals were prepared for analysis through the following preprocessing steps:

1. After the file EEG data is read, the labels are extracted from it, and all the signals are collected into a single array. Also the labels are merged into a single array.
2. Signal normalization: All signals were scaled to a standard distribution with mean 0 and standard deviation 1, using (Z-score normalization) to maintain numerical stability during model training..
3. Segmentation: Signals were divided into fixed-length windows of 30 seconds (3000 samples), in alignment with standard sleep study practices.
4. Class Balancing:
   - The dataset exhibited an imbalance across the five sleep stages.
   - Techniques such as oversampling and SMOTE (Synthetic Minority Oversampling Technique) were used to ensure fair representation of all classes.

   The distribution of sleep stage classes in the training dataset before and after the application of the SMOTE technique is shown in Table (2)

Table (2): The sleep stage classes Distribution before and after SMOTE augmentation

Sleep classes	WITHOUT SMOTE	WITH SMOTE
Class 0 Wake (W)	65357	54225
Class 1 (N1)	21323	54044
Class 2 (N2)	67606	54009
Class 3 (N3)	12833	54006
Class 4 REM	25511	54140

5. The data was split into 80% training and 20% testing.

These steps ensured the EEG data were standardized, balanced, and ready for deep learning model training. There is note of EEG data preprocessing in this study involved several critical steps, including normalizing the signal with Z-score scaling, dividing each recording into fixed 30-second segments (3000 samples), and achieving class balance through SMOTE. The analysis was conducted using a single EEG channel (Fpz-Cz) without the need for channel fusion.

To conduct experiments and ensure reproducibility, we used the Kaggle platform, equipped with 16 GB of RAM and P100 processors. The code was implemented using Python 3.8 and the TensorFlow 2.10.0 library. A fixed random seed (Seed = 42) was used for all libraries (Python, NumPy, and TensorFlow). Using fixed random sampling, the data was split into a training set (80%) and a test set (20%) to ensure reproducibility.

3.3 Training and Evaluation Basics

There are some basics that must be prepared before starting the (training process), which are as follows:

- As mentioned above, the NPZ files containing the EEG signals and their respective labels were read and combined into one array. The longest signals were normalized to 3,000 samples (i.e., using the Reflect Padding method). The classes were then balanced using the SMOTE technique to generate samples and ensure a fair representation of all categories. Finally, the data was divided into a training set (80%) and a test set (20%).
- Implemented using TensorFlow/Keras, following a structured training pipeline.
- Loss function: The categorical cross-entropy loss function was used for the multi-class classification task, due to its high suitability for handling the five distinct sleep stages.

- Optimizer: A learning rate scheduler was incorporated to dynamically adjust the learning rate in response to training progress. The Adam optimizer was also used due to its effectiveness in managing sparse gradients.
- Hyperparameter adjustment is critical, as it has a direct impact on model accuracy, generalization, and other performance metrics. A grid search approach was applied here.
- Each model was trained for a predefined number of epochs (50).
- An early stopping technique was used, set with patience=4, to monitor the loss value. This means that over four consecutive training cycles, the training process stops if the validation loss does not improve. The primary goal of early stopping is to prevent overfitting by monitoring the model's performance on the validation dataset during training..

Once training has completed, the model is evaluated using a range of metrics which offer a holistic evaluation of both overall and per-class performance:

Classification Report: for each classes ,have precision, recall, and F1-score along with micro, macro, and weighted averages

Confusion Matrix: To shows how many samples were classified correctly and incorrect for each class

Accuracy: The accuracy of training and validation.

ROC Curves and AUC: To dives information of how well the model can distinguish between classes at different threshold settings.

4. Implementation and Results Discussion

Selecting the appropriate architecture becomes essential when analyzing temporal signals, such as audio, EEG, or other sequential data, since feature extraction and temporal information retention must be balanced. This is where carefully considered architectural design is useful, guaranteeing: enhanced CNN feature extraction that uses convolutions to find spatial patterns. Maintaining temporal context involves examining the connections between various time points using LSTM. Increasing stability and decreasing computational cost through the use of techniques like pooling layers, dropout, and batch normalization. Consequently, a careful selection of architecture is one of the most important elements that determines a model's success since it is founded on current research and real-world experiences to guarantee precise and effective performance on the intended objectives.

4.1. Model Performance

Given the nature of our dataset, we conducted extensive experiments to overcome overfitting and the difficulty of learning the network[35].

We conducted extensive experiments to determine the most appropriate architecture for LSTM, CNN, and Hybrid CNN,LSTM models to suit the nature of the dataset for classifying sleep stages from EEG signals. We chose an architecture that would balance complexity and performance, overcome overfitting and the difficulty of learning the network, and achieve a stable final implementation. The experiments we conducted included:

1. Determining the appropriate number of layers in each model.
2. Determining the appropriate number of neurons for each layer.
3. After determining the appropriate number of layers and neurons in each model, we began testing the importance of the dense layer and the number of units it should occupy.
4. Determining the appropriate dropout value.

4.1.2 LSTM

Several experiments were conducted on the LSTM model architecture.

1. The first of these experiments was to determine the appropriate number of layers and the number of neurons in each layer of the model for this type of problem. We found that the two-layer model performed best. Furthermore, the two-layer LSTM model, with the first layer containing 100 neurons and the second layer containing 50 neurons, was the best and most stable among the other model architecture options tested. Table (3) shows the overall experiments conducted to determine the number of layers and neurons in each layer, to arrive at the best LSTM architecture in terms of performance.This number of LSTM layers and number of units is to control most of the features in the temporal EEG signal received via the first layer, and then pass the most salient features from the first layer to the second LSTM layer. Furthermore, this arrangement avoids overfitting.[36][37].

2. After determining the most appropriate number of layers and their sizes :(128 and 64), then come the test of dense layer in terms of number, location, and number of units. Table (4) shows the results of these tests. We note from the table that a dense layer with (200 units) is ideal for achieving stability and increasing performance . This first dense layer focuses on the features extracted from the previous LSTM layers to achieve the best prediction. The second dense layer, whose output is 5 values, is combined with the SoftMax function to perform classification tasks. SoftMax is a function that returns a probability vector, which predicts the probability of each class x_i [38][39] (the class here represents one of the five sleep stages).

3. Various values of dropout were tested, as dropout is actually a form of regularization intended to help prevent overfitting by increasing test accuracy, possibly at the expense of training accuracy. This avoids overfitting and improves implementation stability. The implementation of the dropout algorithm relies on randomly dropping neurons during training to avoid co-adaptation of feature detectors, which leads to random variables [40][41].

For each minibatch in the training set, dropout layers randomly separate the inputs from the previous layer to the next in the network architecture with probability p [42].

After Dense Layer 1, we applied a dropout layer to reduce overfitting and help the LSTM model generalize. The dropout value (0.3) was found to be the most appropriate, which was determined by testing a range of values from dropout = 0.2 to dropout = 0.5, and we found that implementation stability was achieved with dropout = 0.3. Table (5) shows the experiments conducted to determine the most appropriate dropout value.

4. Batch size is an important measure of model loss. This parameter represents the number of training samples that will be used during training to perform a single update to the network parameters. Batch size affects model training in terms of the time required for convergence and the amount of overfitting[43]. We studied the effect of different batch sizes on the performance of the LSTM network. Table (6) shows all the experiments we performed in this test.

According to the results shown in Tables 3 to 6, the LSTM architecture shown in Table 7 can be considered a successful and reliable model for classifying different sleep stages.

Table 3:The results efficiency with different LSTM layers

No. of layer	No. of units	Training accuracy	validation accuracy	Training loss	Validation loss
Two	(64, 32)	0.8044	0.7895	0.4958	0.5344
Two	(128, 64)	0.8569	0.8273	0.3617	0.4592
three	(128, 64, 32)	0.8519	0.8260	0.3739	0.4594

Table 4: The results efficiency with different Dense units values of LSTM(128 ,64)

Dense units value	Training accuracy	validation accuracy	Training loss	Validation loss
100	0.8170	0.8060	0.4682	0.4953
200	0.8492	0.8274	0.3923	0.4524
300	0.8421	0.8222	0.4063	0.4571

Table 5: The results efficiency with different Dropout values of LSTM(128 ,64)

Dropout value	Training accuracy	validation accuracy	Training loss	Validation loss
0.2	0.8568	0.8303	0.3707	0.4523
0.3	0.8445	0.8258	0.4033	0.4531
0.4	0.8225	0.8074	0.4583	0.4972

Table 6: The results efficiency with and without Batch normalization LSTM(128 ,64)

Batch normalization	Training accuracy	validation accuracy	Training loss	Validation loss
Without	0.8569	0.8273	0.3617	0.4592
With	0.8644	0.8301	0.3540	0.4524

Table (7) : The sequential Model of LSTM

Layer (type)	Output Shape	Parameters
lstm (LSTM)	(None, 3000, 128)	66,560
lstm_1 (LSTM)	(None, 64)	49,408
dense (Dense)	(None, 200)	13,000
batch_normalization	(None, 200)	800
dropout (0.2)	(None, 200)	0
dense_1 (Dense)	(None, 5)	1,005

4.1.2 CNN

We first tested the appropriate architecture in terms of the number of layers and neural units per layer. In CNN-EEG application areas such as sleep, the average number of CNN layers is relatively high. Some studies use CNN architectures with up to 13 layers, while CNN architectures with 3, 4, 6, 5, and 8 layers are typically used, depending on the nature of the data [44].

From our experiments shown in Table 8, we found that the best CNN architecture performed best with 3 layers containing 32, 64, and 128 units, respectively. We note that the higher the number of layers and units, the higher the training and validation accuracy. However, some models, such as 64, 128, and 256, suffer from increased validation loss, which may indicate potential overfitting.

Based on our findings regarding the best architecture (32, 64, and 128 layers), we conducted several experiments to improve the results and model stability, which can be described as follows:

1. We conducted an experiment on the CNN architecture by placing a batch normalization layer between the convolution layer and the max pooling layer to stabilize training and performance. We also conducted a test without a batch normalization layer. The reason for this experiment is that the EEG signal during sleep is inherently unstable and variable. The batch normalization layer allows for much higher learning rates, while adopting a less conservative approach to initialization. It also acts as a regularizer, learning a more efficient model and also enabling faster neural networks [45][46].
   The results obtained are shown in Table (9), where we observed an improvement in performance with batch normalization. The network was more stable, with a validation accuracy of 0.8736 and a validation loss of 0.3784. Without batch normalization, we observed an increase in the validation loss (0.4060), indicating overfitting..
2. Max pooling is an important layer in CNNs, as it reduces the dimensionality of features while preserving important information. Max pooling can be applied to downsample the convolutional output bands, reducing variance. This feature is very effective with variable EEG signals [47]. In this study, we tested the model without max pooling and with max pooling (with different pooling sizes tested), as shown in Table 10. We found that a pooling size of 2 is an appropriate window size to halve the number of parameters, achieve balance, and preserve information.
3. It is also useful to apply the dropout property using CNN. The dropout property is placed at the end of each Conv layer to prevent overfitting, preserve information, and increase test accuracy. We found through experimentation that the best value is dropout = 0.3. Values higher than 0.3 showed an increase in validation loss; see details in Table 11.
4. The best performance in terms of validation accuracy was achieved at 300 units in the dense layer, which is close to the value at 200 units (see Table 12). The dense network with 200 units was chosen because it achieves the optimal balance between training and generalization. The reason is that in the 300-unit network, the difference between training accuracy and validation accuracy is approximately 0.0514, which is larger than the difference at 200 units (0.0395), indicating the possibility of overfitting in the 300-unit case. The same applies to the training loss and validation loss in both cases, as we note that the validation loss increases slightly in the 300-unit case. Furthermore, the performance of the 100-unit dense layer is poor.

After completing various extensive experiments to arrive at the most suitable CNN architecture for the problem of classifying sleep stages from EGG signals, we show in Table (13) the architecture that has the best stability and learning performance.

Table 8:The results efficiency with different CNN layers

Table 9: The results with and without Batch normalization CNN model

No. of layer	No. of units	Training accuracy	validation accuracy	Training loss	Validation loss
Two	(32, 64)128	0.9240	0.8518	0.1990	0.4748
Two	(64, 128)128	0.9170	0.8536	0.2076	0.4759
Three	(32, 64, 128)64	0.9131	0.8736	0.2210	0.3784
Three	(64, 128, 256)128	0.8799	0.8631	0.2943	0.4146
Four	(16,32,64,128)64	0.8954	0.8734	0.2677	0.4077
Four	(32, 64, 128, 256)128	0.9080	0.8700	0.2307	0.4305

Batch normalization	Training accuracy	validation accuracy	Training loss	Validation loss
With	0.9131	0.8736	0.2210	0.3784
Without	0.9208	0.8719	0.2056	0.4060

Table 10: The results with different **Pooling size value** CNN model

Pooling size	Training accuracy	validation accuracy	Training loss	Validation loss
2	0.9119	0.8735	0.2192	0.3893
3	0.8904	0.8666	0.2831	0.4048
4	0.8630	0.8547	0.3513	0.4112

Table 11: The results with different Dropout values of CNN model

Dropout value	Training accuracy	validation accuracy	Training loss	Validation loss

0.2	0.9178	0.8682	0.2006	0.4452
0.3	0.9131	0.8736	0.2210	0.3784
0,4	0.9005	0.8702	0.2546	0.3754

Table 12: The results efficiency with different Dense units values of CNN MODEL

Dense units value	**Training accuracy**	**validation accuracy**	**Training loss**	**Validation loss**
100	0.8821	0.8618	: 0.2890	0.4121
200	0.9131	0.8736	0.2210	0.3839
300	0.9252	0.8788	0.1924	0.3843

Table 13 : The sequential Model of CNN

Layer (type)	**Output Shape**	**No. Parameters**
conv1d (Conv1D)	(None, 2998, 32)	128
batch_normalization	(None, 2998, 32)	128
max_pooling1d (2)	(None, 1499, 32)	0
dropout (0.3)	(None, 1499, 32)	0
conv1d_1 (Conv1D)	(None, 1497, 64)	6,208
batch_normalization_1	(None, 1497, 64)	256
max_pooling1d_1 (2)	(None, 748, 64)	0
dropout_1 (0.3t)	(None, 748, 64)	0
conv1d_2 (Conv1D)	(None, 746, 128)	24,704
batch_normalization_2	(None, 746, 128)	512
max_pooling1d_2 (2)	(None, 373, 128)	0
dropout_2 (0.3t)	(None, 373, 128)	0
flatten (Flatten)	(None, 47744)	0
dense (200)	(None, 200)	9,549,000
dropout_3 (0.3t)	(None, 200)	0
dense_1 (Dense)	(None, 5)	1,005

4.1.3 Hybrid CNN and LSTM

The following describes the experiments conducted to test the hybrid CNN-LSTM architecture, which naturally involves combining a set of features from each model to achieve the goal of optimal classification of sleep EGG data. This combination process relied on the best CNN and LSTM architectures achieved based on the experiments mentioned above. This involved adopting a two-layer LSTM, with the first layer containing (128, 64) units. The CNN model architecture (32, 664, 128) was also used, with the rest of the details held constant for each.
The first part is tested the CNN_LSTM architecture. The order of this model is such that CNN layers come first to extract features from signals. The output of the CNN features is then transferred to the LSTM layers, which analyze the input data to extract the final features. Here, we tried to discover the effect of extracting spatial features first and then passing them to extract temporal features.
In the second part, we tested the LSTM_CNN architecture: In this part, we tested the effect of extracting temporal features first and passing them to the second layer to explore spatial features. We conducted our experiments on the LSTM_CNN architecture, whose order is the opposite of that of CNN_LSTM. The results shown in Table (13) indicate that the LSTM_CNN architecture performs best in terms of training accuracy, validation accuracy, and training loss. Although the validation loss is slightly higher than the training loss, indicating that overfitting is starting to occur, the LSTM_CNN architecture performs best because extracting temporal and then spatial features is the most successful sequence.
LSTM_CNN architecture can be considered a successful model for classifying different sleep stages. Table (14) illustrates the architecture adopted for this model.

Table 13: The results accuracy of hybrid **Architectures**

Hybride	**Training accuracy**	**validation accuracy**	**Training loss**	**Validation loss**

Architecture				
CNN_LSTM	0.8583	0.8436	0.3577	0.4147
LSTM_CNN	0.9111	0.8813	0.2293	0.4218

Table 14:: The sequential Model of hybrid LSTM-CNN

Type of Layer	**Output Shape**	**No. Parameters**
lstm (LSTM)	**(None, 3000, 128)**	**66,560**
lstm_1 (LSTM)	**(None, 3000, 64)**	**49,408**
conv1d (Conv1D)	**(None, 2998, 32)**	**6,176**
batch_normalization	**(None, 2998, 32)**	**128**
max_pooling1d (MaxPooling1D)	**(None, 2997, 32)**	**0**
dropout (Dropout)	**(None, 2997, 32)**	**0**
conv1d_1 (Conv1D)	**(None, 2995, 64)**	**6,208**
batch_normalization_1	**(None, 2995, 64)**	**256**
max_pooling1d_1 (MaxPooling1D)	**(None, 1497, 64)**	**0**
dropout_1 (Dropout)	**(None, 1497, 64)**	**0**
conv1d_2 (Conv1D)	**(None, 1495, 128)**	**24,704**
batch_normalization_2	**(None, 1495, 128)**	**512**
max_pooling1d_2 (MaxPooling1D)	**(None, 747, 128)**	**0**
dropout_2 (Dropout)	**(None, 747, 128)**	**0**
flatten (Flatten)	**(None, 95616)**	**0**
dropout_3 (Dropout)	**(None, 95616)**	**0**
dense (Dense)	**(None, 200)**	**19,123,400**
dropout_4 (Dropout)	**(None, 200)**	**0**
dense_1 (Dense)	**(None, 5)**	**1,005**

.4.2. Comparative Analysis

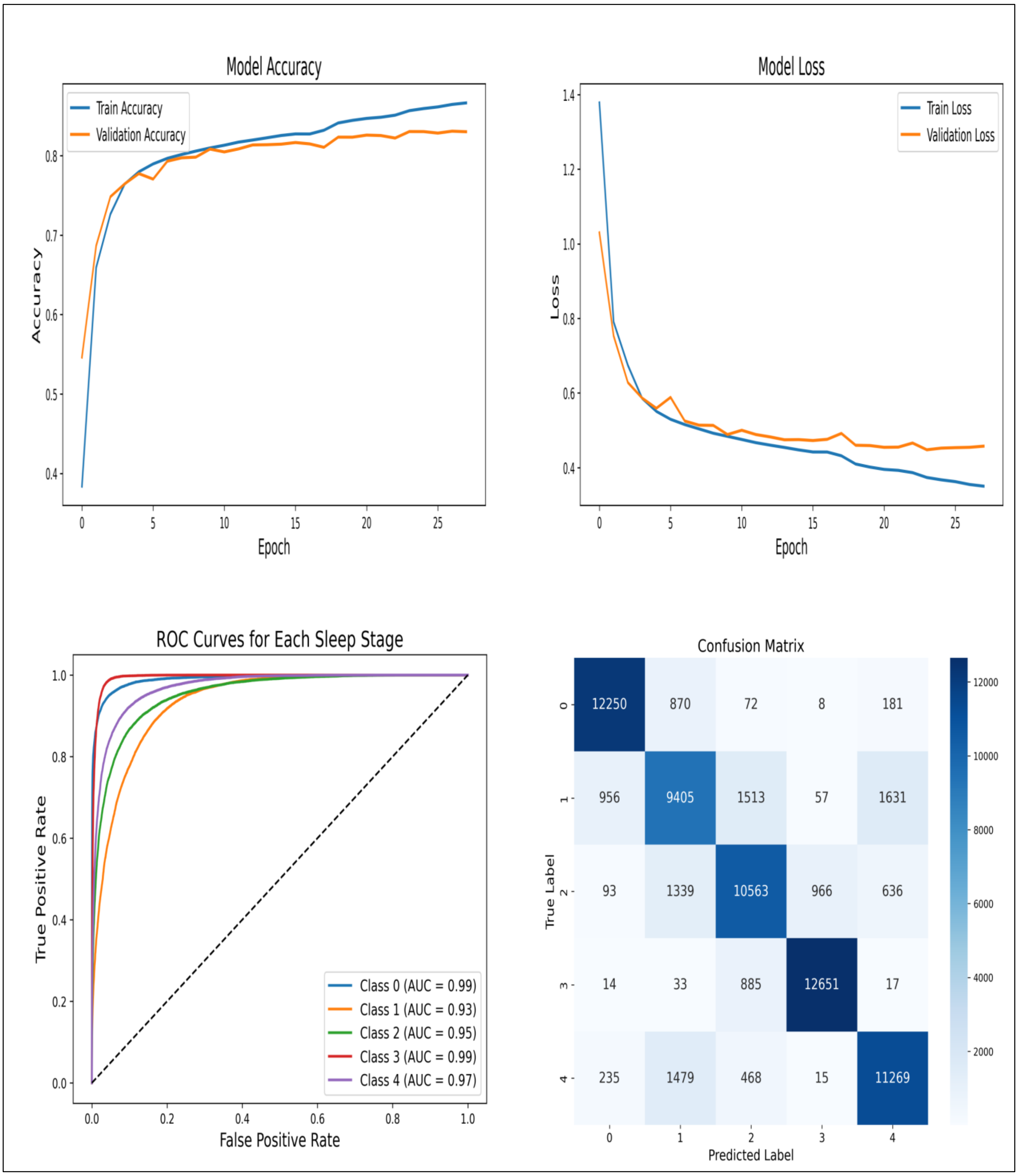
Based on the results presented in the previous paragraphs, and based on well-known performance metrics, in this section we review the key features of each of the three models—LSTM, CNN, and the CNN-LSTM hybrid model—in terms of performance and efficiency. We know that each model has the ability to recognize and process specific signal characteristics:

LSTM Model: The LSTM model demonstrated good performance in recognizing sleep classes, especially with the two-layer architecture, the first with 128 units and the second with 64 units. From the evaluation performance graphs in Figure (3), we note that the best training and completion accuracies are 0.8492 and 0.83, respectively, demonstrating the model's good learning ability.From the training and loss curves, we note that the model is stable and does not suffer from overfitting or underfitting.The confusion matrix results indicate good classification for all classes. Although some classes, particularly Class 1, show confusion with Class 2 by 1513 and Class 3 by 1631, the classification results are considered good and acceptable.ROC curves show good results for each class. Even for Class 1, where AUC is 0.93, which is high..

Although the input data consisted of time series or sequence-like structures, the CNN model had no difficulty in managing temporal relationships and extracting the most important spatial information. It outperformed the LSTM in recognizing unique aspects of the input signal, as shown in the results in Figure 4. The training accuracy reached 0.9131% and the validation accuracy reached 0.8736%, which are good values and indicate that the model does not suffer from significant overfitting. The training loss reached 0.22, and the validation loss reached 0.383, these values indicate the model's stability.And the confusion matrix results, based on the results on the diagonal, indicate that the model classified correctly , where the Class 3 performed well with 13,258 correct cases and 319 incorrect cases, and Some cases, such as Class 1 and Class 2, exhibit confusion, but the overall results are good. From the information on the ROC curves, we can also conclude that the CNN model has an impressive ability to distinguish between different classes.

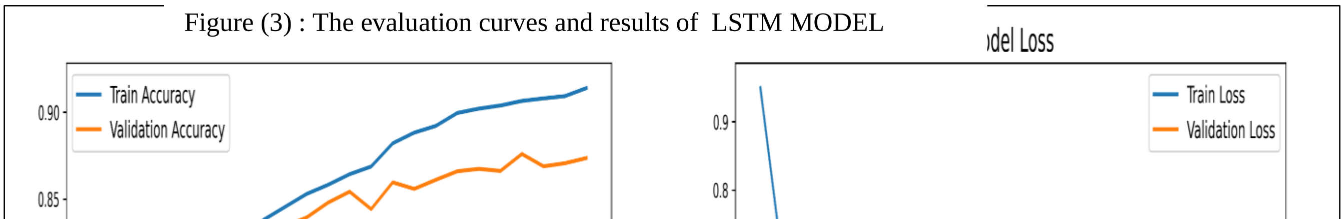
The hybrid CNN/LSTM model demonstrated significant superiority over both the CNN and LSTM models individually on most evaluation metrics. The hybrid LSTM_CNN model (LSTM first, then CNN) achieved the highest results among all tested models. From the evaluation performance graphs in Figure (5), we observe the following: The model's accuracy in distinguishing between the learning and validation phases is good. We note that the training and validation curves increase with each epoch, indicating that the model does not suffer from overfitting. The model's loss curves also reflect a good learning balance, with no significant difference between the training and validation curves. The confusion matrix results indicate good classification for all classes,the Class 3 appears to be the highest-ranking class, with

13,210 correct instances out of 13,600, while the remaining classes performed well. The ROC curves for each class show good results, indicating a high classification ability for the model, AUC values range between 0.96 and 1.00, demonstrating the model's robustness in classifying classes. Class 3 is the best-ranking class, with an AUC of 1.00.



True \ Predicted	0	1	2	3	4
0	12250	870	72	8	181
1	956	9405	1513	57	1631
2	93	1339	10563	966	636
3	14	33	885	12651	17
4	235	1479	468	15	11269

Figure (3) : The evaluation curves and results of LSTM MODEL



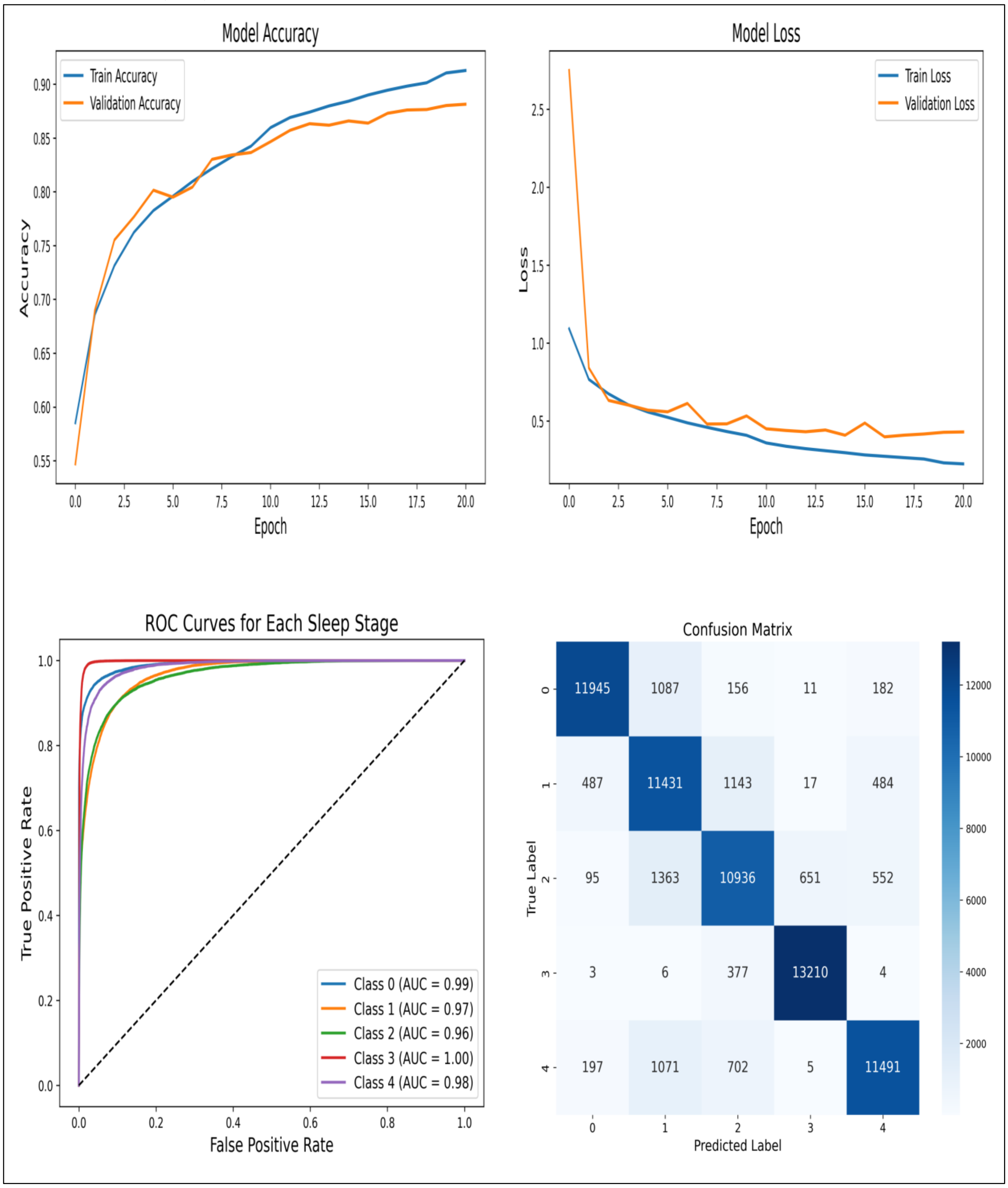


Figure (5) : The evaluation curves and results of Hybrid LSTM+ CNN MODEL

5. Conclusion and Recommendations

This section relies on extensive experiments and results obtained from performance metrics such as accuracy, loss, confusion matrix, and ROC curves, on LSTM, CNN, and hybrid CNN/LSTM models. From the summary of the results in Table (15), we note that all models demonstrated strong performance across the four performance metrics. Stability and learning demonstrate that these models are well-suited to handling and classifying sleep signals. All models provided very similar results in terms of accuracy and loss, demonstrating the strength of each model in this task.

Table(15) :Final table of Computational Efficiency of models

Feature	**LSTM**	**CNN**	**LSTM + CNN**
Training accuracy	0.8644	0.9131	0.9111
Validation Accuracy	0.8301	0.8736	0.8813
Training loss	0.3540	0.2210	0.2293
Validation Loss	0.4577	0.3893	0.4218
F1 score	0.83	0.87	0.87
Number of Parameters	130,773	9,709,033	19,277,249
Total Training Time	19559.48 sec	4156.67 sec	18922.81 sec
Average Training Time	698.24 sec	148.14 sec	900.67 sec
Average Inference Time per Sample	0.003884 sec	0.000109 sec	0.003985 sec

However, in terms of comparison between these models:

1. The LSTM-CNN model achieved the best balance between training accuracy (0.9111) and validation accuracy (0.8813), with a relatively low loss. This is because, from a scientific perspective, there is a benefit to combining temporal and spatial features, especially in the LSTM-CNN order, which significantly enhanced classification capabilities, the temporal signals are processed first, and then spatial information is extracted from them.Although the LSTM-CNN model outperforms the others and is a good choice in terms of balancing accuracy and temporal and spatial representation efficiency, we recommend using any of the proposed models CNN or LSTM, depending on the available capabilities and the purpose of the study , as they also demonstrated excellent results, as shown in Table 15.
2. Number of parameters: We note that LSTM is the lightest, with only 130,000 parameters, compared to CNN, which has about 9 million, and LSTM+CNN, which has about 19 million.
3. In terms of training time, CNN demonstrated remarkable speed, averaging only 148 seconds per cycle, compared to LSTM (698 sec.) and LSTM+CNN (900 sec.). Therefore, CNN is suitable for such datasets in terms of speed.
4. In terms of inference time, CNN outperforms, with an average of only 0.0001 sec. per sample. On the other hand, LSTM and LSTM+CNN are slower, about 0.004 sec.

References

[1] Ramar, K., Malhotra, R. K., Carden, K. A., Martin, J. L., Abbasi-Feinberg, F., Aurora, R. N., Kapur, V. K., Olson, E. J., Rosen, C. L., Rowley, J. A., Shelgikar, A. V., & Trotti, L. M., "Sleep is essential to health: An American Academy of Sleep Medicine position statement. Journal of Clinical Sleep Medicine", 17(10), 2115–2119, (2021). https://doi.org/10.5664/jcsm.9476

[2] A. Crivello, P. Barsocchi, M. Girolami, and F. Palumbo,**"**The Meaning of Sleep Quality: A Survey of Available Technologies,**"**,*IEEE Access*, vol. 7, pp. 167374–167390, (2019). . https://doi.org/10.1109/access.2019.2953835

[3] S. M. M. de Mooij, T. F. Blanken, R. P. P. P. Grasman, J. R. Ramautar, E. J. W. Van Someren, and H. L. J. van der Maas,**"**Dynamics of sleep: Exploring critical transitions and early warning signals**"**,Computer Methods and Programs in Biomedicine, vol. 196, Sep. 2020, Art. no. 105448.https://doi.org/10.1016/j.cmpb.2020.105448

[4] A. K. Patel, V. Reddy, and J. F. Araujo, "Physiology, sleep stages", StatPearls, StatPearls Publishing, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK526132/

[5] R. Cox and J. Fell, "Analyzing human sleep EEG: A methodological primer with code implementation", Sleep Medicine Reviews, vol. 50, Art. no. 101353, 2020. https://doi.org/10.1016/j.smrv.2020.101353

[6] I. G. Campbell, "EEG recording and analysis for sleep research" ,Current Protocols in Neuroscience, vol. 49, no. 1, pp. 10.2.1–10.2.19, Oct,2009.https://doi.org/10.1002/0471142301.ns1002s49

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. ,1997. https://doi.org/10.1162/neco.1997.9.8.1735

[8] C. Li, Y. Qi, X. Ding, J. Zhao, T. Sang, and M. Lee,"A deep learning method approach for sleep stage classification with EEG spectrogram",International Journal of Environmental Research and Public Health, vol. 19,no.10,p.6322,May,2022.https://doi.org/10.3390/ijerph19106322

[9] O. Tsinalis, P. M. Matthews, and Y. Guo, "Automatic sleep stage scoring using time–frequency analysis and stacked sparse autoencoders", Annals of Biomedical Engineering, vol. 44, no. 5, pp. 1587–1597, May 2016. https://doi.org/10.1007/s10439-015-1444-y

[10] S. Sasai, T. Koike , S. K. Sugawara, S. Okazaki, K. Watanabe, N. Sadato and Iguchi, Y. ,"Network-wide reorganization of procedural memory during NREM sleep revealed by fMRI". eLife, 6, e24987, 2017. https://doi.org/10.7554/eLife.24987

[11] M. Nicola, U. R. Acharya, and F. Marcelloni,"Cascaded LSTM recurrent neural network for automated sleep stage classification using single-channel EEG signals",Computers in Biology and Medicine, vol. 106, pp. 71–81, Jan. 2019.https://doi.org/10.1016/j.compbiomed.2019.01.013

[12] A. Salamatian and A. Khadem, "Automatic sleep stage classification using 1D convolutional neural network", Frontiers in Biomedical Technologies, vol. 7, no. 3, 2022. http://dx.doi.org/10.18502/fbt.v7i3.4616

[13] D. Zhao, R. Jiang, M. Feng, J. Yang, Y. Wang, X. Hou, and X. Wang, "A deep learning algorithm based on 1D CNN-LSTM for automatic sleep staging," Technol. Health Care, 2022. DOI: 10.3233/THC-212847.

[14] A. Luic, A. Takashi, and M. Molinas, "EEG-based 5- and 2-class CNN for Sleep Stage Classification," IFAC-PapersOnLine, vol. 56, no. 2, pp. 3211–3216, 2023. DOI: 10.1016/j.ifacol.2023.10.1458.

[15] E. Enes and O. Seral, "CoSleepNet: Automated sleep staging using a hybrid CNN-LSTM network on imbalanced EEG-EOG datasets," Biomed. Signal Process. Control, vol. 80, Feb. 2023, Art. no. 104299. DOI: 10.1016/j.bspc.2022.104299.

[16] R. Kotla, N. Polavarapu, and P. Safa, "A CNN-LSTM Model for Sleep Stage Scoring Using EEG Signals," in Proc. Int. Conf. Commun., Circuits, Syst. (IC3S), 2023. DOI: 10.1109/IC3S57698.2023.1016917.

[17] A. Supratak, H. Dong, C. Wu, and Y. Guo, "DeepSleepNet: A model for automatic sleep stage scoring based on raw single-channel EEG," IEEE Trans. Neural Syst. Rehabil. Eng., 2017. DOI: 10.1109/TNSRE.2017.2721116.

[18] O. Tsinalis, P. M. Matthews, Y. Guo, and S. Zafeiriou, "Automatic sleep stage scoring with single-channel EEG using convolutional neural networks," arXiv preprint, arXiv:1610.01683, 2016. DOI: 10.48550/arXiv.1610.01683.

[19] D. T. Pham and R. Mouček, "Automatic Sleep Stage Classification by CNN-Transformer-LSTM using single-channel EEG signal," in Proc. IEEE Int. Conf. Bioinformatics Biomed. (BIBM), Shanghai, China, Dec. 2023. DOI: 10.1109/BIBM58861.2023.10385687.

[20] A. A. Casciola, J. Bossert, and N. Elbuluk, "A Deep Learning Strategy for Automatic Sleep Staging Based on Two-Channel EEG Headband Data," Sensors, vol. 21, no. 10, p. 3316, 2021. DOI: 10.3390/s21103316.

[21] T. Li, B. Zhang, H. Lv, S. Hu, Z. Xu, and Y. Tuergong, "CAttSleepNet: Automatic End-to-End Sleep Staging Using Attention-Based Deep Neural Networks on Single-Channel EEG," Int. J. Environ. Res. Public Health, vol. 19, no. 9, p. 5199, 2022. DOI: 10.3390/ijerph19095199.

[22] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," Artif. Intell. Rev., vol. 53, no. 1, 2020. DOI: 10.1007/s10462-020-09838-1.

[23] S. M. Al-Selwi, M. F. Hassan, S. J. Abdulkadir, A. Muneer, E. H. Sumiea, A. Alqushaibi, and M. G. Ragab,"RNN-LSTM: From applications to modeling techniques and beyond—Systematic review," J. King Saud Univ.—Comput. Inf. Sci., 2024. DOI: 10.1016/j.jksuci.2024.102068.

[24] X. Wei, L. Zhang, H. Yang, L. Zhang, and Y. Yao,"Machine learning for pore-water pressure time-series prediction: Application of recurrent neural networks," Geosci. Frontiers, vol. 12, no. 1, pp. 453–467,Jan. 2021.
DOI: 10.1016/j.gsf.2020.04.011.

[25] A. Thakkar and K. Chaudhari, "A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions," Expert Syst. Appl., vol. 177,p.114800,2021.DOI:http://dx.doi.org/10.1016/J.ESWA.2021.114800.

[26] J. M. Dahr and A. S. Gaafar, "Performance evaluation of the convolutional neural networks for object identification using RGB and binary images," Informatica, vol. 48, no. 21, pp. 177–188, 2024. DOI: 10.31449/inf.v48i21.6568.

[27] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," J. Big Data, vol. 8, art. no. 53, 2021. DOI: 10.1186/s40537-021-00444-8.

[28] F. Sultana, A. Sufian, and P. Dutta, "Advancements in Image Classification using Convolutional Neural Network," in Proc. 2018 4th Int. Conf. Res. Comput. Intell. Commun. Netw**. (ICRCICN)**, Kolkata, India, Nov. 2018, pp. 122–129. doi: 10.1109/ICRCICN.2018.8718718

[29] S. S. Nisha and M. N. Meeral, "Applications of Deep Learning in Biomedical Engineering," in *Handbook of Deep Learning in Biomedical Engineering: Techniques and Applications*, Academic Press, 2021, pp. 245–270, doi: 10.1016/B978-0-12-823014-5.00008-9.

[30] T. Kessler, G. Dorian, and J. H. Mack, "Application of a Rectified Linear Unit (ReLU) based artificial neural network to cetane number predictions," in Proc. ASME 2017 Int. Combust. Engine Div. Fall Tech. Conf., Oct. 2017. DOI: 10.1115/ICEF2017-3614.

[31] M. Blot, M. Cord, and N. Thome, "Max-min convolutional neural networks for image classification," in Proc. 2016 IEEE Int. Conf. Image Process. (ICIP), Phoenix, AZ, USA, Sep. 2016,pp. 3678–3682.doi: 10.1109/ICIP.2016.7533046

[32] T. Zhou, C. XiaoYu, L. Huiling, and Y. XinYu, "Pooling operations in deep learning: From 'invariable' to 'variable'," Biomed. Res. Int., vol.

2022, art. no. 4067581, 2022. DOI: 10.1155/2022/4067581.

[33] A. Ghosh, F. Sultana, A. Sufian, and A. Chakrabarti, "Fundamental concepts of convolutional neural network," in Intell. Syst. Ref. Libr., Springer, 2020, pp. 1–20. DOI: 10.1007/978-3-030-32644-9_36.

[34] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," IEEE Trans. Pattern Anal. Mach. Intell.,2017. DOI: 10.1109/TPAMI.2016.2599174.

[35] I. Priyadarshini and C. Cotton, "A novel LSTM–CNN–grid search-based deep neural network for sentiment analysis," J. Supercomput., vol. 77, pp. 13911–13932, 2021. DOI: 10.1007/s11227-021-03838-w.

[36] Y. Li, "K-LSTM-ECM Model for Predicting Poverty Alleviation Impacts of Digital Financial Inclusion," Informatica, vol. 49, no. 17, 2022, doi: 10.31449/inf.v49i17.7621..

[37] B. H. D. Koh and W. Woo, "Multi-view temporal ensemble for classification of non-stationary signals," IEEE Access, vol. 7, pp. 34189–34202, 2019. DOI: 10.1109/ACCESS.2019.2903571.

[38] M. Franke and J. Degen, "The softmax function: Properties, motivation, and interpretation," Unpublished, 2023 https://doi.org/10.31234/osf.io/vsw47 .

[39] R. C. Staudemeyer and E. Rothstein Morris, "Understanding LSTM: A tutorial into long short-term memory recurrent neural networks," *arXiv*, vol. 1909.09586v1, 2019. DOI: 10.48550/arXiv.1909.09586.

[40] Staudemeyer, R. C., & Rothstein Morris, E. (2019). Understanding LSTM: A Tutorial into Long Short-Term Memory Recurrent Neural Networks. arXiv, 1909.09586v1. https://arxiv.org/abs/1909.09586v1

[41] P. Baldi and P. Sadowski, "The dropout learning algorithm," Artif. Intell., vol. 210, pp. 78–122, 2014. DOI: 10.1016/j.artint.2014.02.004.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," J. Mach. Learn. Res., vol. 15, no. 56, pp. 1929–1958, Jun. 2014. doi:10.5555/2627435.2670313

[43] P. M. Radiuk, "Impact of training set batch size on the performance of convolutional neural networks for diverse datasets," Inf. Technol. Manag. Sci., vol. 20, pp. 20–24, 2017. DOI: 10.1515/itms-2017-0003.

[44] S. Rajwal and S. Aggarwal, "Convolutional neural network-based EEG signal analysis: A systematic review," Arch. Comput. Methods Eng., vol. 30, pp. 3585–3615, 2023. DOI: 10.1007/s11831-023-09920-1.

[45] M. Hasani, "An empirical study on position of the batch normalization layer in convolutional neural networks," in Proc. 2019 5th Iranian Conf. Signal Process. Intell. Syst. (ICSPIS), 2019, pp. 1–5. DOI: 10.1109/ICSPIS48872.2019.9066113.

[46] S. Lagnaoui, Z. En-Naimani, and K. Haddouch, "The effect of normalization and batch normalization layers in CNNs models: Application to plant disease classifications," in Proc. 6th Int. Conf. Big Data Internet Things, Springer, 2023, pp. 250–262. DOI: 10.1007/978-3-031-28387-1_22.

[47] H. Gholamalinezhad and H. Khosravi, "Pooling methods in deep neural networks, a review," CoRR, vol. abs/2009.07485, 2020. doi:10.48550/arXiv.2009.07485.

.