



ISSN ONLINE: 2447-0228



RESEARCH ARTICLE

OPEN ACCESS

A DEEP LEARNING FRAMEWORK FOR REAL-TIME CLASSIFICATION OF VISUAL LEARNING STYLES FROM RAW EEG SIGNALS

S. Nagendram¹, Geeta Bhimrao Atkar², Desamala Prabhakara Rao³, Tirumalasetti Lakshmi Narayana⁴, Sunitha Pachala⁵, Shobana Gorintla⁶, A. Gangadhar⁷, Koppolu Sireesha⁸, Inakoti Ramesh Raja⁹¹Professor, Department of CSE-AIML, KKR & KSR Institute of Technology and Sciences, Guntur, India.²Assistant Professor, Department of Computer Engineering, G H Raisoni College of Engineering and Management, Pune, Maharashtra, India.³Associate Professor, Department of Electronics and Communication Engineering, Chalapathi Institute of Technology, Mothadaka, Guntur, India.⁴Assistant Professor, Department of Electrical and Electronics Engineering, Aditya university, Surampalem, A.P, India.⁵Associate Professor, Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Green Fields, Vaddeswaram, India.⁶Professor, Department of CSE, Department of CSE, NRI Institute of Technology (Autonomous), Pothavarapadu, Agiripalli mandal, Pincode-521212, Eluru District, Andhra Pradesh, India.⁷Associate Professor, Department of ECE, UCEN, JNTUK, Kakinada, India.⁸Assistant professor, Department of CSE-AIML, R. V. R. & J.C College of Engineering, Guntur, Andhra Pradesh, India.⁹Associate Professor, Department of Electronics and Communication Engineering, Aditya University, Surampalem, A.P., India.¹<http://orcid.org/0000-0002-7265-1860>, ²<https://orcid.org/0000-0002-5328-484X>, ³<https://orcid.org/0009-0001-3930-3258>⁴<http://orcid.org/0009-0000-5359-6723>, ⁵<http://orcid.org/0000-0002-4149-7178>, ⁶<https://orcid.org/0000-0002-9357-6142>⁷<http://orcid.org/0000-0002-8165-3120>, ⁸<http://orcid.org/0009-0000-7620-9001>, ⁹<https://orcid.org/0009-0006-6806-2605>

Email: *reena1286@gmail.com, git.121@gmail.com, prabhakardesamala@gmail.com, tlaxman17@gmail.com, drsunitha.pachala@kluniversity.in, drgshobana@gmail.com, gangadhar.ece@jntukucen.ac.in, kopolusireesha@rvrjc.ac.in, inakoti.rameshrraja@gmail.com

ARTICLE INFO

ABSTRACT

Article History

Received: January 21, 2026

Reviewed: July 24, 2026

Accepted: August 24, 2026

Published: September 30, 2026

Keywords:

EEG

Frame work

LSTM

CNN

This paper focuses on identifying visual learners in real time using brain signals. The signals come from an EEG, which records brain activity. Usually, EEG data is noisy and needs cleaning before use. Traditional methods use this cleaned data to find learning styles. But cleaning takes time, so the models are not useful for real-time use. This issue is solved by using raw EEG data. Deep learning models are developed that do not need manual feature extraction. These models learn directly from raw data. The main purpose is to check if someone is a visual learner. A visual learner learns best with images and videos. The EEG data was collected from 34 healthy people. These people had no prior knowledge of the content shown. Participants were shown animated educational videos. While watching, EEG signals were recorded. Three deep learning models were used: LSTM, LSTM-CNN and LSTM-FCNN. These models are good at understanding time-based data like EEG. Among them, LSTM-CNN worked the best. It gave the highest accuracy. It had good sensitivity, specificity and F1-score. This method does not need any self-reports. It gives fast results. It works with different lengths of EEG data. So, it is useful in classrooms and other real-world settings. It helps teachers understand the best way students learn. This improves both teaching and learning. This work shows that deep learning with raw EEG is powerful. It quickly and accurately identifies visual learners. This changes the future of detecting learning styles.



Copyright ©2026 by authors and Galileo Institute of Technology and Education of the Amazon (ITEGAM). This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

I. INTRODUCTION

Learning styles support better learning by matching methods to individual needs [1]. One important learning style is the visual learning style. Visual learners understand things better through pictures, videos, charts and diagrams [2]. Research shows that about 70% of people are visual learners. That is why this work focuses on identifying visual learners. Many educational tools try to support different learning styles. Some examples are digital whiteboards, video feedback and interactive quizzes.

These tools improve student focus and understanding. But most of these tools use subjective methods. That shows the learner or teacher guesses the learning style. People might believe themselves to be visual learners but are not [3]. It adds confusion and pressure to students. A better way is to look at brain signals. EEG is one tool that records brain waves [4]. It is safe and easy to use. It is affordable and gives quick results. EEG is used to examine the brain's reaction during a person's learning process. Earlier methods used EEG but required a lot of data cleaning. Handcrafted features were used. These steps are time-consuming and cannot be done in real time. Hence, this approach is not suitable for classroom or fast-paced learning environments. This work solves this problem. It uses raw EEG data without cleaning [5]. It uses deep learning models. These models learn useful features directly from the data. This makes the system fast and accurate. The deep learning models used are LSTM, LSTM-CNN and LSTM-FCNN [6]. These models work well with time-series data like EEG signals. Participants in the work were 34 healthy students. No prior knowledge of the animated videos used in the learning task was given to the participants. EEG was recorded during rest and during the learning activity. A test was later conducted to check the amount of information remembered. This helped in classifying them as visual learners or not. The purpose of the paper is to detect visual learners in real time. This helps teachers plan better lessons. It helps students understand the way learning works best for each individual. The system is used in real-world settings like classrooms [7]. It does not need manual effort or self-reports. It is unbiased and reliable. The key contributions of this paper are:

- To develop a method for detecting visual learners in real time using raw EEG data without requiring handcrafted features or data preprocessing. To apply deep learning models, specifically LSTM-CNN for automatic feature extraction directly from raw EEG signals.
- To validate the approach using real EEG data collected from human subjects instead of simulated datasets. To demonstrate that the visual learning style is accurately identified without the use of questionnaires or surveys.
- To confirm the models, operate efficiently on EEG data of varying lengths, supporting real-time classification. To design a scalable solution suitable for large classroom environments. To achieve high performance with classification accuracy reaching up to 94%.

This work stands out for many reasons. Most earlier studies used self-assessments. This gives false results. Others used cleaned or synthetic EEG data [8]. That cannot be trusted for real-time tasks. Many past studies had small sample sizes or lacked proper validation. This work solves all of those problems. This paper does not use forms or self-reports. It records brain activity during actual learning. It uses resting-state data. This provides a reference for the brain's behavior when not engaged in a task [9]. That helps make the model even more accurate. The method checks memory performance. The deep learning models used are well-suited for this task. LSTM learns from time-based patterns [10]. CNN extracts key features. LSTM-CNN combines both. It gives the best performance. The work compares these models to see which is best. LSTM-CNN has high accuracy and low computational time [11]. This makes it the most useful model for real-world use. The paper discusses the problems with earlier methods. Many used synthetic data or small EEG samples. Some used models that did not run fast. Others needed expert knowledge to select the features. This approach removes all those barriers. It works directly with raw data. It runs fast. It is easy to use in real time. Another strong point of this work is that it uses a good sample size. It uses a proper experiment design. It checks learning performance and memory. It records EEG in multiple ways – during rest, learning and recall [12]. This makes the results more accurate and useful. This paper presents a strong and useful solution. It is used in classrooms, online learning or other educational setups [13]. This work presents a deep learning model to identify visual learners using raw EEG. It works in real time. It removes the need for data cleaning or manual features. It gives high accuracy. It is fast, unbiased and easy to apply. It helps educators understand the best way students learn [14]. This method brings a big step forward in personalized education using brain signals.

II. RELATED WORK

Many researchers tried to understand learning styles using brain signals. In most studies, EEG signals were used. EEG records brain activity using electrodes. It is safe, cheap and easy to use. This makes EEG useful for understanding the learning process in people. In earlier work, researchers commonly used self-reports to label learners. Forms were filled out to describe the way each person learns best. But this method is not reliable. A person believes to be a visual learner but is not. It is only a guess, not a fact. To solve this, some researchers used EEG signals to check brain activity while people learn. The research explored whether real patterns in the brain exist for different learning styles. In [15], the VARK test was used. This test checks if a student is a visual or reading learner. Pictures were shown to the students, and EEG was recorded. Visual learners showed a higher P200 wave in the brain, which appeared in the occipital region. The task did not involve videos, charts, or other visual materials. In [16], visual and verbal learners were compared using EEG data during a picture viewing task. Visual learners had more beta waves and fewer theta waves during verbal tasks, indicating difficulty with such tasks. However, this research did not explore all forms of visual learning. Another project [17] used e-learning content and EEG to classify learners. A chatbot was added, and students learned through videos. The VARK test was given afterward. Visual learners showed higher beta wave activity. Naïve Bayes and clustering algorithms were used to classify the data. In [18], researchers used physiological signals like heart rate and blood pressure. Students watched a video and then filled the VARK test. The results showed increased heart rate in non-visual learners. A decision tree classifier was used. Many other papers tried to connect EEG signals and learning styles [19-21]. Each approach used different tasks and feature sets. But many of them used handcrafted features. These features need expert knowledge and are slow to extract. Most of these models were trained offline. That makes them useless in real-time classrooms. Some studies used spectral centroid frequency (SCF) from EEG data [22]. Others used power spectrum or bandpass features. These were good in some cases but lacked generalizability. Most of the previous models used small datasets. That increases the risk of overfitting. Many papers did not report standard metrics like precision, recall or F1-score. So, it is hard to compare the results. Some studies used CNN to classify EEG data. One experiment [23] used 1D-CNN and achieved 71.2% accuracy. However, it used only local features and not the full data. Others tried RNN or simple LSTM models but had poor accuracy or needed too much training time. Machine learning was used in earlier work. But ML models like SVM or k-NN need good features to work.

These models are unable to learn features without guidance. Deep learning does that better. Deep learning models like LSTM or CNN learn hidden patterns from raw data. A major issue in old methods was the need for preprocessed EEG data. This paper solves many of the problems. It uses raw EEG data. No cleaning or filtering is needed. Deep learning models are used to learn from this data directly. It avoids self-reports and uses brain signals alone. The data comes from real people and includes many types of visuals. This approach is different because it focuses on real-time use. It tries to detect learning styles during rest. This resting data is clean and easy to collect. It reflects the brain's normal activity level. That helps in making better predictions. The past work laid the foundation. But this paper takes it forward. It makes EEG-based learning style detection real-time, fast and reliable.

III. DEEP LEARNING FOR REAL-TIME CLASSIFICATION OF VISUAL LEARNING STYLE USING EEG

In the past, deep learning was not commonly used for EEG data. It was considered too slow and complex. One big problem was the vanishing gradient issue, which made it hard to train deep networks. Older hardware did not process deep models quickly. But this situation has changed. Today, powerful GPUs are available for use. There are large EEG datasets. These improvements make deep learning easier and faster to apply to brain signal data. Deep learning is now a popular tool in many medical and research fields. It learns important features from data without manual effort. This helps when the data is complex and hard to understand. EEG signals are one example and are difficult to interpret by hand. But deep models find useful patterns directly from the signals. This removes the need for pre-processing or manual feature extraction. Electroencephalogram (EEG) data is a type of time-series data. It is recorded over time and contains signals from different brain regions. For this kind of data, deep learning models like CNN (Convolutional Neural Network) and LSTM (Long Short-Term Memory) work very well. CNN detect spatial features or patterns in the signal. LSTM is useful for finding changes in the signal over time. Another benefit of deep learning is that it works well even with noisy or raw data. EEG data has noise. Removing that noise usually takes time and expert effort. But deep learning still performs well with raw signals. In real-time applications, results should be fast. There is no time for complex cleaning or processing. Traditional systems that rely on handcrafted features take longer. In this paper, raw EEG data is used directly. The need for cleaning or handcrafted feature extraction is avoided. Instead, deep learning models are trained on the raw data. This makes the system faster and more suitable for real-time use. It reduces bias because it does not depend on human-designed features. Deep models automatically learn what is important from the data itself. The concept behind the model is simple. Each person's EEG data is taken as input. This data is linked with a label that says if the person is a visual learner or not. The model is trained to find patterns that match these labels. Over time, it learns to predict if a new person is a visual learner just by analyzing the brain signal. The function that does this is learned automatically by adjusting the model's internal weights. The LSTM model works by keeping memory of previous time steps. It has special parts named gates. These gates decide what to remember and what to forget. This makes LSTM strong for time-series signals. CNN works by applying filters to the data. These filters find important shapes or peaks in the signal. After applying filters, CNN uses functions like ReLU to improve learning. It reduces the size of data through pooling, which makes the process faster.

Combining LSTM and CNN results in the LSTM-CNN model. This model takes advantage of both local and temporal features. It first applies CNN to the raw EEG to find patterns. Then it passes the results to the LSTM, which learns the changes in the signal over time. Finally, the model uses a dense layer to make the prediction. This layer outputs a value showing the probability of being a visual learner. In the LSTM-FCNN model, the structure is a little different. It uses only convolutional layers for feature learning. At the end, it uses a global average pooling layer. This helps reduce the number of parameters. It avoids overfitting when data is limited. FCNN is useful when the data is long or has many channels. This makes it a good fit for EEG. One reason raw EEG was chosen is because it is more natural. It contains the original brain signals without being changed. Handcrafted methods need cleaned data. This adds delay and effort. It depends on expert decisions, which are not always right. Deep learning skips this step and still gets high accuracy. The Figure 1 shows a deep learning model that combines convolutional and memory components to classify input data. The input goes through two different processing paths. On the left side, the input passes through a stack of three 1D CNN layers. Each layer uses the ReLU activation function and different filter sizes: 128 in the first layer, 256 in the second and 128 again in the third. After the CNN layers, the output goes to a global feature aggregation step. This step compresses the information. It creates a fixed-length feature vector. This helps reduce the size and prepare the data for classification. On the right side, the input is first rearranged to match the expected format for sequential data. It is then sent through an LSTM block, which helps capture both long-term and short-term patterns in the data. After that, a temporary node removal step is applied. This helps clean or drop unnecessary signals that might affect performance. The outputs from both sides are combined to form a unified feature set. This combined output is passed through a softmax layer, which gives the final prediction. This figure shows a model that uses CNN for feature extraction. It uses LSTM to learn time-based patterns. This makes the model effective for sequence tasks like EEG signal classification.

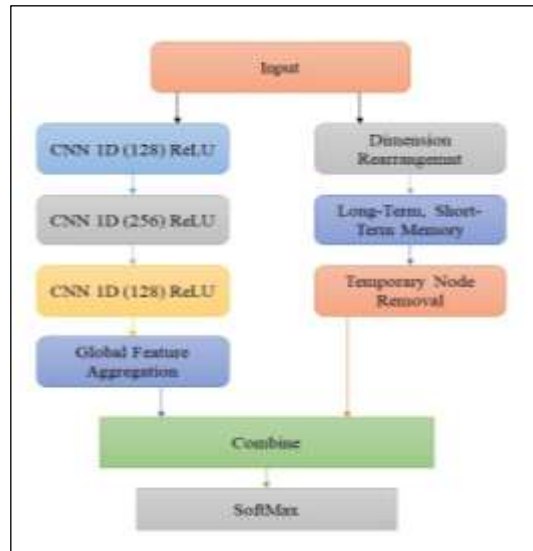


Figure 1: LSTM-FCNN Architecture.
Source: Authors, (2026).

Another reason is that deep learning is flexible. It works with different data lengths. It does not need fixed-size input. This is helpful because EEG recordings vary in size. Some learners show patterns quickly, while others take more time. A flexible system handles both cases. The models in this paper perform better than traditional machine learning methods. In older methods, the user had to choose which features to use. This leads to poor results if the features were not good. Deep learning avoids this by learning the features itself. It adapts better to new data and new people. The main outcome of this work is that LSTM-CNN gave the best results. It had high accuracy, fast performance and worked well with raw data. It was better than both plain LSTM and LSTM-FCNN. This shows that combining pattern detection and memory is a strong way to analyze EEG. Deep learning uses raw EEG data. It avoids the need for manual steps. It is fast, accurate and adaptable. It works well with time-based brain signals. It is used in real classrooms and learning apps.

IV. MATERIALS & METHODS

This section explains the steps followed in the experiment and the method of data collection. It shows the way EEG signals were processed and used in the deep learning model. The experiment included 34 university students. Ages ranged from 18 to 30 years. On average, the age was 23.17 years with a small variation. All students had normal vision or corrected vision. None of them had brain disorders or hearing problems. No medicines were being taken during the experiment. Before starting, every participant signed a consent form. The ethics board approved the entire experiment. The work was divided into two main tasks. Task 1 was a learning activity. In this task, students watched animated videos about human anatomy. These videos were around 8 to 10 minutes long. The topic was unfamiliar to the students, so no prior knowledge was present. Task 2 involved memory. After watching the video, 20 multiple-choice questions were given. Each question had a 30-second time limit. The questions were based on the video content. The whole test followed a step-by-step order. First, the memory test was conducted. Based on memory scores, students were split into two groups. This helped identify whether someone learned well visually or not. Then, EEG recordings were taken in two resting conditions: eyes open and eyes closed. After that, the students did the learning task. Later, two recall tests were conducted—one after 30 minutes and another after two months. The learning videos were shown without sound. So, students used only visual senses to learn. EEG signals were recorded from the entire head. But only signals from the frontal, parietal and occipital areas were used. These parts of the brain are active during visual learning. The learning phase helped encode new information into the brain. The recall sessions showed the amount of information the students remembered. These recall scores helped confirm if someone was a visual learner. Research says visual learners tend to remember more when information is shown as visuals. Resting-state EEG data gives useful information. In this state, the student is not doing any task. This shows the brain's basic activity. Patterns in this state help identify brain traits like visual learning. In this experiment, eyes-closed EEG signals were used. With eyes closed, there are fewer visual distractions. This makes the data more stable and reliable. The videos were shown on a large TV screen. The screen was 42 inches and placed 1.5 meters away from the student. The experiment used special software, E-Prime Professional 2.0 to control the tasks. The EEG device used was from EGI. It had 128 channels using the HydroCel Geodesic Sensor Net. The signals were recorded continuously. The amplifier used a band-pass filter between 0.1 and 100 Hz. The data was recorded at 250 samples per second. The impedance was kept below 50 k Ω . The central electrode Cz was used as a reference point. After the EEG was recorded, the next step was to analyze the behavioral data. The learning and memory performance were judged based on accuracy and response time. A faster response time usually indicates better brain processing. Fluid intelligence has two parts. First is the ability to find meaning in confusing data. Second is the ability to recall and use new information. Based on this test, the students were divided into two equal groups. Those who scored above or equal to the median were labeled as visual learners. The rest were labeled as non-visual learners. The next step was to prepare the EEG data. Only the eyes-closed session was used. Each student's EEG was recorded for 2 to 8 minutes. These long recordings were split into smaller one-minute segments. Each one-minute segment had 15,000 data points because of the 250 Hz sampling rate. The dataset was made larger by combining these segments. These data segments were then divided into training, validation, and test sets.

The training set was used to train the model. The validation set helped fine-tune the model. The test set checked the model's final performance. There was no overlap between these sets. This kept the testing data clean and fair. The dataset was split into three parts using 3-fold cross-validation. This method helps avoid overfitting and gives better results. Each group had both visual and non-visual learners. The groups were randomized to avoid any bias from hidden factors. The deep learning models were trained using specific settings. For binary classification, categorical cross-entropy loss was used. The Adam optimizer helped the model learn faster. The activation function used in all models was SoftMax. An early stopping rule was used. If the model did not improve after 10 steps, the training stopped to save time. The model parameters were picked randomly at the start. Later, the models were fine-tuned using training data. This improved the accuracy, sensitivity and specificity of the models. By carefully organizing the data and using deep learning, the researchers developed a system that works in real time. The use of real EEG data helped improve the system. Smart data splitting improved learning. These steps allowed the researchers to create a strong and reliable system to identify visual learners in real time.

V. METHODOLOGY

The system is designed to identify visual learners using raw EEG signals. The focus is on making the system fast and usable in real-time conditions. EEG data was recorded from each subject during a relaxed state with eyes closed. This state reduces distractions and gives cleaner brain signals. Each subject contributed between 2 and 8 minutes of data. The EEG was recorded with a sampling rate of 250 Hz. That shows each second produced 250 data points. For a one-minute segment, this results in:

$$T = 60 \text{ seconds}, F_s = 250 \text{ Hz}, N = T * F_s = 15000 \text{ samples} \quad (1)$$

So, each segment had 15,000 time points per electrode. A total of 128 electrodes were used. Electrodes from the neck and face were excluded due to noise. The final number of channels used was:

$$C = 128 - 16 = 112 \quad (2)$$

From these, only 102 scalp channels were selected for the model input. The EEG data for each subject was thus stored in a 2D matrix form:

$$x \in R^{15000 \times 102} \quad (3)$$

Each one-minute segment of EEG was labeled based on the subject's memory performance. If the subject had high recall scores, the label was set as 1 (visual learner). If not, the label was 0 (non-visual learner). The label function is written as:

$$y_i = \begin{cases} 1, & \text{if subject } i \text{ is a visual learner} \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

These pairs of input and label were used to create the dataset:

$$D = \{x_i, y_i\} \text{ for } i = 1, 2, \dots, N \quad (5)$$

Here, X_i is the EEG signal matrix for the i^{th} segment and $y_i \in \{0, 1\}$ is the label. The dataset was split into three non-overlapping sets. Training set D_{train} to fit the model. Validation set D_{val} to fine-tune during training. Test set D_{test} to evaluate final performance. To make the results more reliable, the dataset was arranged using 3-fold cross-validation. The data was divided into three equal parts. In each run, two parts were used for training and the third for testing. This helped reduce bias and improved the accuracy of the model. The input to all models is the same EEG matrix $X \in R^{15000 \times 102}$. It has 15,000 time steps and 102 channels. This matrix is passed into a neural network model that learns to predict y_i , the learning style. The purpose of the model is to find a function f_θ with parameters θ , such that:

$$f_\theta(X_i) \approx y_i \quad (6)$$

The model adjusts its parameters θ during training to reduce the difference between its output and the true label. The output of the model is passed through a softmax or sigmoid function depending on the type of output layer to get a probability value between 0 and 1:

$$y_i = \sigma(f_\theta(X_i)) \quad (7)$$

Here, $\hat{y}_i$ is the predicted probability that the segment X_i belongs to a visual learner. This approach used three deep learning architectures. LSTM is good for time-based data. LSTM-CNN combines time and spatial features. LSTM-FCNN is a faster model with fewer parameters. All models are designed to classify the input EEG data into one of two groups: visual learner or non-visual learner. LSTM is a type of neural network that is good at learning from time-based data. EEG is time-based so LSTM is useful here. LSTM remembers important information from earlier time steps. It does this using a structure named memory cell. Each cell has gates. These gates control the flow of information. At each time step t , the model takes three inputs. X_t is the input data at time t . h_{t-1} is the hidden state from the previous time step. C_{t-1} is the memory cell value from the previous time step. The model then updates its internal values using the following equations:

$$\begin{aligned}
f_t &= \sigma(W_f * [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i * [h_{t-1}, x_t] + b_i) \\
\tilde{C}_t &= \tanh(W_c * [h_{t-1}, x_t] + b_c) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
o_t &= \sigma(W_o * [h_{t-1}, x_t] + b_o) \\
h_t &= o_t * \tanh(C_t)
\end{aligned} \tag{8}$$

Each part has a role. f_t is the forget gate that decides what to remove from the cell. i_t is the input gate decides what new information to add $\tilde{C}_t$ is the candidate cell state new content to add C_t is the updated memory cell after combining old and new. o_t is the output gate controls what part of the memory to send out. h_t is the hidden state used in the next step or output layer. The functions σ and $\tanh$ are activation functions. The sigmoid σ maps values between 0 and 1. The tanh maps between -1 and 1. In the model used here, all EEG time steps are passed through LSTM cells. The final output from the last time step, h_t is used for the prediction. It goes to a fully connected layer. This layer has one output neuron and uses a sigmoid activation to give the probability:

$$\hat{y} = \sigma(W_o * h_T + b) \tag{9}$$

This output $\hat{y}$ tells whether the person is a visual learner. If $\hat{y}$ is close to 1, the model believes the input came from a visual learner. If it is close to 0, it thinks it is from a non-visual learner. The loss function used is binary cross-entropy:

$$L = -[y * \log(\hat{y}) + (1 - y) * \log(1 - (\hat{y}))] \tag{10}$$

Here y is the true label (0 or 1) and $\hat{y}$ is the predicted value. The loss becomes small when $\hat{y}$ is close to y . The model uses this loss to update the weights and improve its accuracy. It repeats this process over many batches of data during training. LSTM-CNN model combines two powerful deep learning structures. LSTM handles time-based information well. CNN is good at finding patterns in raw data. When combined, these components create a stronger model. The EEG signal is first passed through one or more CNN layers. These layers apply 1D convolutional filters. A filter is a small sliding window that looks at short parts of the signal. The task is to find shapes or patterns like spikes, waves or other useful signals. Assume the input EEG data has the shape which is given in (3). A 1D convolutional filter works along the time axis for each channel. Let the convolution filter be defined with kernel size k , stride s and weights w . At time step t , the convolution output is:

$$c_t = \text{ReLU}(W * x_{t:t+k-1} + b) \tag{11}$$

Here, $x_{t:t+k-1}$ is a small segment of the EEG signal, w is the filter weight and b is the bias. The function ReLU (Rectified Linear Unit) is used as the activation:

$$\text{ReLU}(z) = \max(0, z) \tag{12}$$

After the convolution, the result is usually too large. So, a pooling layer is used. Pooling reduces the output size and keeps only the strongest responses. One common method is max pooling:

$$p_t = \max(C_{t:t+s-1}) \tag{13}$$

This keeps the maximum value from a small segment of the signal. It helps keep the important parts and remove noise. After these CNN and pooling layers, the output is smaller and contains important features. This cleaned data is then passed to the LSTM layer. The LSTM reads this new sequence and tries to learn time-based patterns from it. LSTM has gates that decide what to remember and what to forget. The final hidden state from the LSTM, written as h_T , is passed to a fully connected layer. The final prediction is made using a sigmoid activation which is given in (9). LSTM-FCNN model another hybrid model that combines LSTM with a Fully Convolutional Neural Network (FCNN). This model is different from LSTM-CNN because it uses only convolution layers, without pooling. It uses a technique named Global Average Pooling (GAP) instead of fully connected layers at the end of the convolution block. The purpose is to create a model that is fast and uses fewer parameters. FCNN extract features from raw EEG without increasing model size too much. It is more stable during training and has a lower chance of overfitting. An FCNN is made up of several convolutional blocks. Each block has three layers 1D Convolution layer, Batch normalization layer and ReLU activation. Let the input EEG segment is given in (3). Each convolution layer applies filters to extract patterns from the EEG data. Let a filter of size k act on the input with stride s . The basic convolution step is:

$$z_t = \text{ReLU}(W * x_{t:t+s-1} + b) \tag{14}$$

Batch normalization is applied after convolution. It normalizes the values so that the network learns faster and avoids vanishing gradients. The model uses multiple blocks with increasing filter sizes. For example, the first layer might use 128 filters, the second uses 256 and the third uses 128 again. Each layer learns higher-level features. After all convolution blocks, the model uses GAP.

GAP reduces the entire time-series output into a single average value per filter. If the last convolution block has m filters, the GAP layer produces an output vector of length m . Let the output from the last convolution block be:

$$H \in R^{T \times m} \quad (15)$$

Then GAP computes:

$$g_j = \frac{1}{T} \sum_{t=1}^T H_{t,j} \text{ for } j = 1, 2, 3, \dots, m \quad (16)$$

This gives a final vector:

$$g = [g_1, g_2, g_3, \dots, g_m] \quad (17)$$

This vector is passed to the LSTM module. The vector g is reshaped or expanded as needed and passed into an LSTM layer. The LSTM processes this short and dense feature sequence. It helps learn time-dependencies across compressed spatial features. The LSTM-FCNN model works in three steps. Use convolution blocks to extract local patterns. Use GAP to compress the features. Use LSTM to learn sequence and predict the label. This model uses fewer parameters than LSTM-CNN. It trains faster and performs well on large EEG datasets. It is exclusively useful when the data is noisy or variable in length. The deep learning model includes the training settings, optimizer, loss function and stopping rules. The purpose was to make sure the models learn correctly and avoid overfitting. To train the model, the binary cross-entropy loss is given in (10). The purpose of training is to minimize this loss over all examples. The Adam optimizer was used for training. Adam is a popular optimization algorithm that adjusts the learning rate for each parameter. It works well on large datasets and trains faster. The learning rate was set to $\alpha = 0.001$. This shows the model takes small steps to improve weights during training. The data is processed in small batches. A batch is a group of examples passed through the model at one time. The batch size used was 16. Training continued for multiple epochs. One epoch refers to going through the full training dataset once. The number of epochs was set to 15. The full dataset was split into three parts. In each fold, two parts were used for training. The third part was used for testing. This process was repeated three times so that each part was tested once. The average of the three results was taken as the final score. The models were trained on a machine with a modern GPU. Training each model took a few minutes to a few hours depending on its complexity. LSTM-FCNN was the fastest due to fewer layers. LSTM-CNN took slightly longer. Pure LSTM was slowest because it processed the full time series without compression. Once a model is trained, it needs to be tested on unseen data to check its performance. For this, several evaluation metrics were used. These metrics help measure whether the model is accurate and reliable. Accuracy measures the total number of correct predictions:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (18)$$

This tells us the frequency at which the model is correct. But it does not show if one class is better predicted than the other. Precision shows the number of predicted visual learners that were actually correct:

$$Precision = \frac{TP}{TP + FP} \quad (19)$$

A high precision shows there are few false positives. This indicates the model is careful when predicting someone as a visual learner. Recall shows the number of actual visual learners that were correctly predicted:

$$Recall = \frac{TP}{TP + FN} \quad (20)$$

This is termed as sensitivity. A high recall shows the model is good at finding visual learners. Specificity shows the number of non-visual learners that were correctly predicted:

$$Specificity = \frac{TN}{TN + FP} \quad (21)$$

It tells us if the model avoids false alarms for visual learning. F1-Score balances both precision and recall. It is useful when both false positives and false negatives are important.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (22)$$

If precision and recall are both high, the F1-score will be high. In some models, the Area Under the ROC Curve (AUC-ROC) was used. A higher AUC (closer to 1) shows better performance. Each of the three models—LSTM, LSTM-CNN and LSTM-FCNN—was evaluated using these metrics. The scores were calculated using the test data from cross-validation. The model with the highest average accuracy, recall and F1-score was considered the best. This helped select a model that performs well across different types of users and brain signals. All three deep learning models—LSTM, LSTM-CNN and LSTM-FCNN—were evaluated using the same data and metrics. The performances were compared using the evaluation metrics described earlier. Each model was trained three times using 3-fold cross-validation. The average scores for accuracy, precision, recall, specificity and F1-score were computed. These results helped identify the best model. Accuracy was highest for the LSTM-CNN model. It correctly identified most visual and non-visual learners.

$$Accuracy_{LSTM-CNN} > Accuracy_{LSTM-FCNN} > Accuracy_{LSTM} \quad (23)$$

This shows that combining CNN with LSTM gives better results than using LSTM alone. The F1-score balances precision and recall. It was highest for LSTM-CNN. The model was both precise and sensitive when detecting visual learners.

$$F1_{LSTM-CNN} \approx 0.94 \quad (24)$$

This score indicates a strong ability to classify both classes well. In terms of training speed, LSTM-FCNN was the fastest. This is because it has fewer parameters and does not use fully connected layers. It uses Global Average Pooling, which reduces complexity. So, if speed is more important than perfect accuracy, LSTM-FCNN is a better choice. The LSTM model had higher variance across different folds. This shows it was less stable. It sometimes performed very well and other times worse. On the other hand, the LSTM-CNN model gave steady results in all folds. That shows better generalization to different types of EEG signals. The proposed models expressly LSTM-CNN and LSTM-FCNN are designed to work with raw EEG signals. Manual feature extraction or signal cleaning is not required. This makes them suitable for real-time use. The model works with short EEG recordings and uses raw signals. This makes it lightweight and scalable. It is suitable for integration into portable devices or cloud-based services. Since brain data is sensitive, care should be taken when collecting and using it. Students should give consent before recording EEG. Data should be stored securely and used only for learning improvement. The system should never be used to label or judge students permanently. Instead, it should help adapt teaching to the learner's needs. More tests should be done in real classrooms with more students. The model needs improvement to support multiple languages and cultures.

VI. RESULTS

This section presents the classification results of three deep learning models: LSTM, LSTM-CNN and LSTM-FCNN. These models are used to classify visual and non-visual learners using raw EEG data recorded during eyes-closed resting conditions. Each model's performance is assessed based on accuracy, F1-score and AUC derived from ROC analysis.

Table 1: LSTM Model Evaluation Metrics and Confusion Matrix.

	Precision	Recall	F1-Score	Support
Visual Learners (L)	0.92	0.92	0.92	13
Non-Visual Learners (NL)	1.00	1.00	1.00	4
Accuracy	0.96			

Source: Authors, (2026).

The LSTM model performance shown in Table 1 shows strong overall performance, with a classification accuracy of 96%. For visual learners, the model achieves a precision, recall and F1-score of 0.92. For non-visual learners, all three metrics reach 1.00. The model correctly classified 12 out of 13 visual learners and all 4 non-visual learners. Only one visual learner was misclassified. This shows that the LSTM model effectively distinguish between the two groups. Its high AUC of 1.00 indicates excellent separability between classes.

Table 2: LSTM-CNN Model Evaluation Metrics and Confusion Matrix.

	Precision	Recall	F1-Score	Support
Visual Learners (L)	0.92	0.92	0.92	13
Non-Visual Learners (NL)	1.00	1.00	1.00	4
Accuracy	0.94			

Source: Authors, (2026).

The LSTM-CNN model performance shown in Table 2 performs very well with an overall accuracy of 94%. The precision, recall and F1-score values are identical to the LSTM model. However, the AUC values are slightly lower at 0.95 for visual learners and 0.96 for non-visual learners. Like the LSTM model, it misclassifies only one visual learner. Despite the small dip in AUC, this model is faster and suitable for real-time applications. Its reliable accuracy across both classes shows its robustness.

Table 3: LSTM-FCNN Model Evaluation Metrics and Confusion Matrix.

	Precision	Recall	F1-Score	Support
Visual Learners (L)	0.83	0.83	0.83	6
Non-Visual Learners (NL)	1.00	1.00	1.00	11
Accuracy	0.89			

Source: Authors, (2026).

The LSTM-FCNN model performance is shown in Table 3. It is designed for faster training and efficiency. It shows slightly lower performance compared to the previous two models, with an accuracy of 89%. The model correctly identifies 5 out of 6 visual learners and all 11 non-visual learners. This results in an F1-score of 0.83 for visual learners and 1.00 for non-visual learners. The AUC values are 0.89 and 0.90, respectively. While the accuracy is lower, this model offers faster computation and is more suitable in environments with a focus on speed. The results show that the LSTM model provides the highest classification accuracy and perfect AUC making it highly reliable for accurate predictions. The LSTM-CNN model although slightly behind in AUC offers a better balance between speed and performance. The LSTM-FCNN model while not the most accurate is the most efficient computationally.

This model is useful when faster predictions are needed even at the cost of slight accuracy loss. Overall, these results validate that deep learning effectively classifies learning styles using raw EEG.

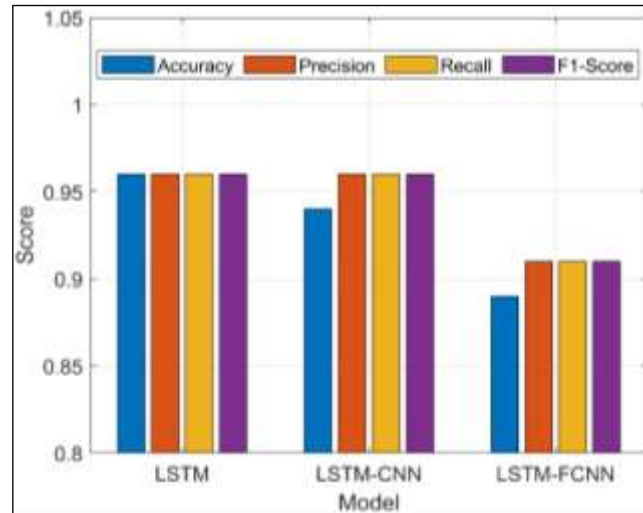


Figure 2: Model Comparison: Accuracy, Precision, Recall, F1-Score.
Source: Authors, (2026).

The Figure 2 compares the performance of three deep learning models: LSTM, LSTM-CNN and LSTM-FCNN. It shows four evaluation metrics for each model: Accuracy, Precision, Recall and F1-Score. The LSTM model performs the best across all metrics. Each of its bars reaches a score slightly above 0.96. This indicates strong and balanced performance. The LSTM-CNN model has slightly lower Accuracy (around 0.94) but maintains high values in the other metrics, close to 0.96. This shows that while its prediction correctness dropped a little it still detects true positives well. The LSTM-FCNN model performs the worst. All four bars for this model are lower than the others. Its Accuracy is below 0.90. Precision, Recall and F1-Score are close to 0.91. This suggests that the model is less reliable and steady. There is a clear drop in every metric compared to the LSTM and LSTM-CNN models. Overall, the LSTM model is the most effective followed closely by LSTM-CNN. The LSTM-FCNN model lags behind in all aspects. The Figure 3 tracks the training accuracy of three models—LSTM, LSTM-CNN and LSTM-FCNN—over 15 training epochs. At the beginning (epoch 1), LSTM shows an accuracy close to 0.60. While LSTM-CNN follows slightly lower at around 0.58. LSTM-FCNN starts lowest at 0.55. All models improve steadily with each epoch. By epoch 5, LSTM reaches an accuracy of about 0.82, LSTM-CNN rises to around 0.79. LSTM-FCNN climbs to nearly 0.75. The LSTM model shows the fastest improvement in the early stages. It maintains a steady lead throughout the training period. By epoch 10, the LSTM model achieves a training accuracy above 0.93. The LSTM-CNN model reaches around 0.91 and LSTM-FCNN trails at 0.87. In the final epoch (epoch 15), LSTM peaks close to 0.97. LSTM-CNN follows closely at about 0.95. Meanwhile, LSTM-FCNN levels of around 0.90 showing a slower learning rate. All three models experience a gradual flattening of accuracy growth after epoch 10. This indicates that the models approach maximum learning capacity. Among the three, LSTM is the best performer followed by LSTM-CNN. LSTM-FCNN performs the weakest in terms of training accuracy over time.

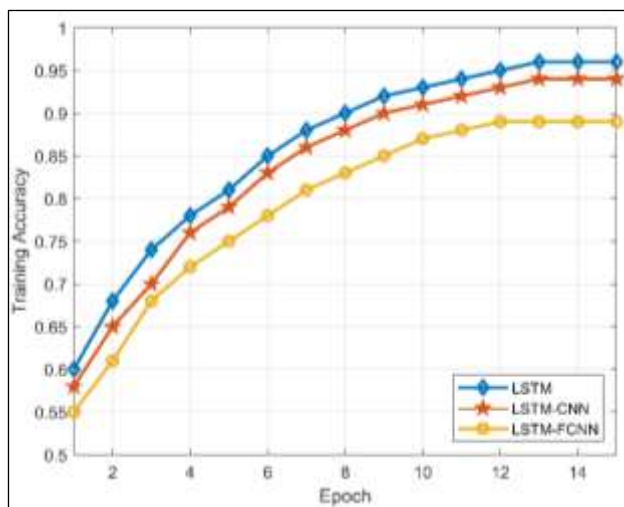


Figure 3: Training Accuracy over Epochs.
Source: Authors, (2026).

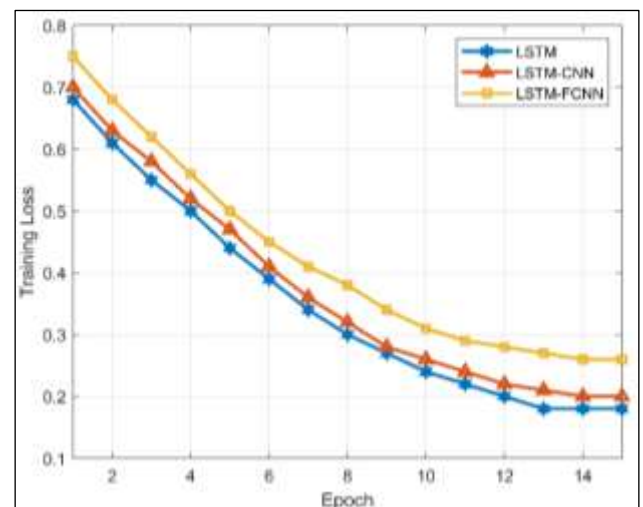


Figure 4: Training Loss over Epochs.
Source: Authors, (2026).

Figure 4 shows the change in training loss over 15 epochs for three models: LSTM, LSTM-CNN and LSTM-FCNN. All models start with high loss values in epoch 1. LSTM-FCNN starts the highest at about 0.75, while LSTM, LSTM-CNN are around 0.70 and 0.72. As training progresses, all models show a steady drop in loss.

By epoch 5, LSTM reduces its loss to about 0.50, LSTM-CNN is slightly higher near 0.52. LSTM-FCNN reaches only about 0.56. LSTM continues to improve faster and keeps the lowest loss in most epochs. At epoch 10, LSTM loss drops to around 0.23, LSTM-CNN to about 0.25 and LSTM-FCNN lags behind at nearly 0.32. From epochs 10 to 15, all models slowdown in improvement but LSTM keeps the lowest values. By the final epoch, LSTM's loss is just below 0.20, LSTM-CNN is around 0.21. LSTM-FCNN remains higher near 0.27. The difference between LSTM and LSTM-FCNN becomes more visible with each epoch. LSTM shows the best learning efficiency, followed closely by LSTM-CNN. LSTM-FCNN learns more slowly and ends with the highest training loss. The Figure 5 displays the AUC (Area Under the ROC Curve) over 15 training epochs for three models: LSTM, LSTM-CNN and LSTM-FCNN. At epoch 1, LSTM starts with an AUC of about 0.80. LSTM-CNN begins slightly lower at around 0.78, while LSTM-FCNN starts at approximately 0.75. All three models show growth in AUC in the early stages. By epoch 5, LSTM reaches about 0.90, LSTM-CNN climbs to around 0.88 and LSTM-FCNN improves to nearly 0.86. LSTM shows the fastest rise in AUC, staying ahead of the other models in every epoch. As training continues, LSTM continues to rise and reaches close to 0.99 by epoch 15. LSTM-CNN improves steadily too but flattens out after epoch 11, ending at about 0.96. LSTM-FCNN grows more slowly and levels off early, finishing with an AUC near 0.90. This pattern suggests that LSTM has the best ability to separate positive and negative classes. LSTM-CNN performs well, while LSTM-FCNN shows the weakest performance. The LSTM model gives the most reliable and steady AUC scores across all epochs.

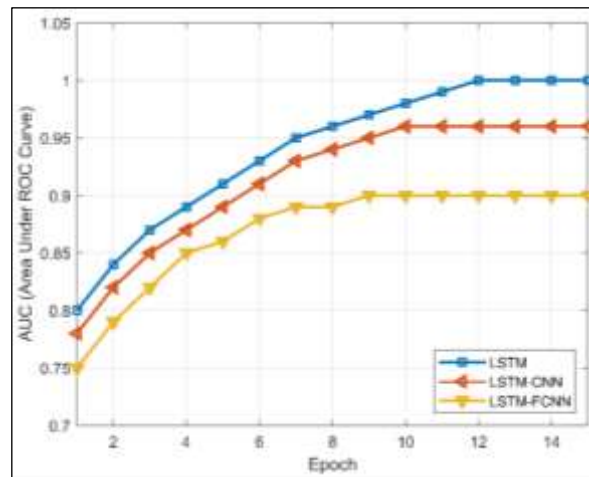


Figure 5: AUC over Epochs for Deep Learning.
Source: Authors, (2026).

The Figure 6 shows the inference time of three different models: LSTM, LSTM-CNN and LSTM-FCNN. Inference time refers to the duration the model takes to make a prediction. The LSTM model has the highest inference time, which is about 0.024 seconds. The LSTM-CNN model takes less time around 0.018 seconds. The LSTM-FCNN model is the fastest among the three, with an inference time of about 0.01 seconds. LSTM takes more than twice the time of LSTM-FCNN to give an output. The chart shows a clear trade-off between performance and speed. Although LSTM and LSTM-CNN offer higher accuracy and better scores in earlier figures, these models require more time to run. LSTM-FCNN while slightly lower in performance is faster and is more suitable for real-time applications. This comparison shows that if speed is the priority LSTM-FCNN is a better choice. But if accuracy is more important LSTM or LSTM-CNN would be better despite the extra time needed.

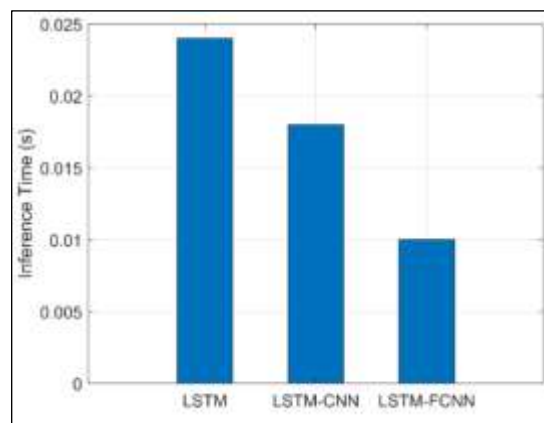


Figure 6: Inference Time (S) Comparison across Models.
Source: Authors, (2026).

The Figure 7 compares the number of parameters in three models: LSTM, LSTM-CNN and LSTM-FCNN. LSTM has the most parameters, close to 1.8 million. It requires more memory and time to train and run. LSTM-CNN comes next with around 1.2 million parameters. It is smaller than LSTM but still has a high capacity for learning. LSTM-FCNN has the fewest parameters about 0.6 million making it the lightest model in this group.

The chart shows a clear drop in parameter count from LSTM to LSTM-CNN to LSTM-FCNN. A larger model like LSTM gives better results but will be slower and heavier to use. LSTM-CNN offers a balance between model size and performance. LSTM-FCNN is the most efficient in terms of size and speed but its performance is lower. So, the choice of model depends on the trade-off between accuracy and resource usage.

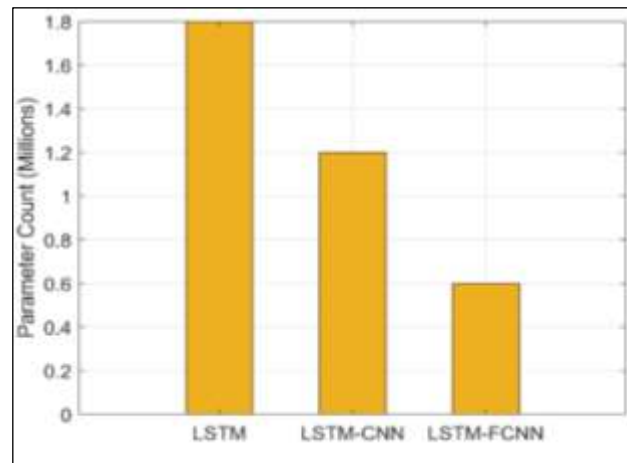


Figure 7: Parameter Count (Millions) Comparison across Models.
Source: Authors, (2026).

The Figure 8 shows the F1-Score over 15 training epochs for three models: LSTM, LSTM-CNN and LSTM-FCNN. In the first epoch, LSTM starts with an F1-Score of about 0.60. LSTM-CNN begins at around 0.58. LSTM-FCNN is the lowest at approximately 0.55. All models improve steadily over time. By epoch 5, LSTM crosses 0.80, LSTM-CNN follows closely at about 0.78 and LSTM-FCNN reaches around 0.72. LSTM maintains the highest F1-Score throughout the training process and shows faster growth in early epochs. By epoch 10, LSTM reaches near 0.93. LSTM-CNN hits about 0.91. LSTM-FCNN climbs to nearly 0.87. From epochs 11 to 15, all models begin to flatten showing that improvement is slowing. In the final epoch, LSTM achieves an F1-Score close to 0.96. LSTM-CNN ends around 0.94, while LSTM-FCNN stops at about 0.89. This indicates that LSTM performs the best in terms of balance between precision and recall. LSTM-CNN performs well, but slightly below LSTM. LSTM-FCNN improves the least and stays behind the other two models across all epochs.

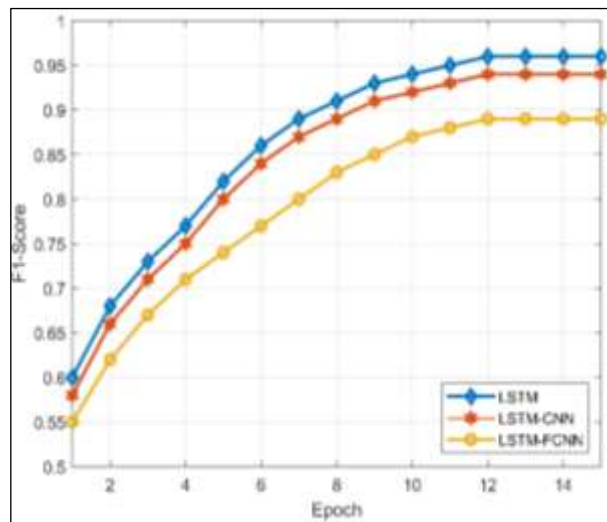


Figure 8: F1-Score over Epochs.
Source: Authors, (2026).

Figure 9 shows the change in validation accuracy over 15 epochs for three models. In the first epoch, LSTM begins around 0.60. LSTM-CNN is slightly lower at about 0.58 and LSTM-FCNN starts lowest at 0.52. As training continues, all models improve. By epoch 5, LSTM rises to about 0.78, LSTM-CNN to 0.75. LSTM-FCNN reaches around 0.70. LSTM improves the fastest and stays ahead throughout the training. LSTM-CNN follows closely, while LSTM-FCNN trails behind the others in every epoch. In the final stage of training, all three models show a slower increase in validation accuracy. By epoch 15, LSTM reaches about 0.95, LSTM-CNN finishes at 0.93 and LSTM-FCNN ends at around 0.86. The performance gap widens over time, exclusively between LSTM and LSTM-FCNN. LSTM delivers the best validation accuracy, proving it generalizes well to new data. LSTM-CNN performs well too but slightly less than LSTM. LSTM-FCNN shows slower learning and lower results which is due to its simpler structure. LSTM gives the best results for this validation task.

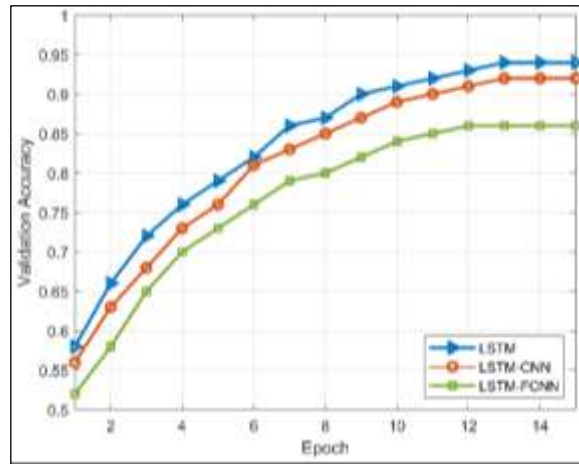


Figure 9: Validation Accuracy over Epochs.
Source: Authors, (2026).

The Figure 10 shows the number of epochs needed for three models to converge. Convergence is the point at which the model's performance stops showing clear improvement. The LSTM model takes the longest to converge needing about 13 epochs. LSTM-CNN takes slightly less time, converging at 12 epochs. The LSTM-FCNN model converges the fastest, requiring only 10 epochs. This indicates that LSTM-FCNN reaches its peak performance earlier than the other models. Even though LSTM-FCNN converges quicker, it does not always mean better performance. As seen in previous results, LSTM gives higher accuracy and F1-scores though it needs more time to train. LSTM-CNN balances between training time and performance. LSTM-FCNN is efficient in terms of training time but does not perform as well. This figure clearly shows the trade-off between early convergence and model quality. Faster convergence saves resources but sometimes at the cost of reduced model accuracy.

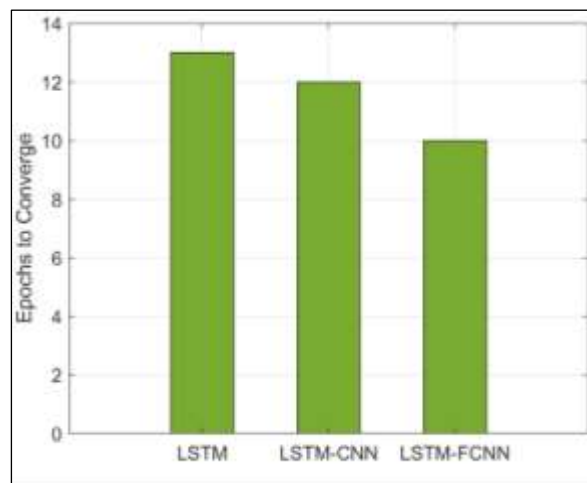


Figure 10: Number of Epochs Required for Convergence.
Source: Authors, (2026).

The Figure 11 compares validation loss across 15 epochs. In the first epoch, LSTM starts with a loss close to 0.68, LSTM-CNN is a bit higher at around 0.70. LSTM-FCNN begins at the highest point near 0.73. All three models show a steady decrease in loss as training progresses. By epoch 5, LSTM reduces to around 0.47, LSTM-CNN reaches about 0.49. LSTM-FCNN lowers to approximately 0.53. LSTM maintains the lowest loss throughout most of the epochs showing better learning efficiency. In the later epochs, the loss values begin to level out. Around epoch 10, LSTM is just below 0.30, LSTM-CNN is slightly above it and LSTM-FCNN is near 0.35. From epochs 11 to 15, improvement becomes slower. LSTM ends with the lowest validation loss at around 0.25. LSTM-CNN finishes close to 0.28 and LSTM-FCNN settles a little above 0.30. This result shows that LSTM provides better generalization than the other two. LSTM-CNN performs reasonably well, while LSTM-FCNN has higher loss indicating weaker performance on unseen data.

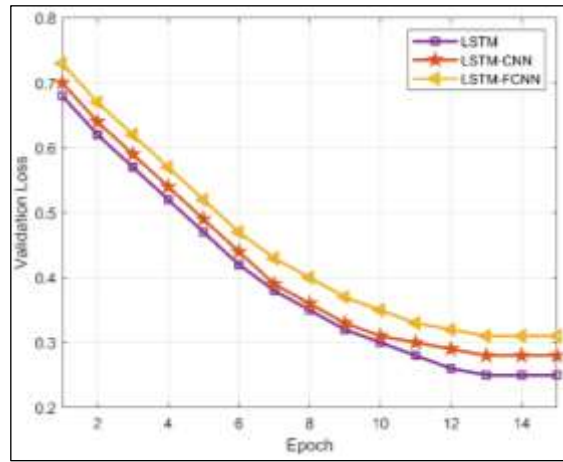


Figure 11: Validation Loss over Epochs.

Source: Authors, (2026).

VII. CONCLUSION

This approach proposed a deep learning-based method to detect visual learning styles using raw EEG data. Unlike traditional methods, it does not depend on self-report surveys or manually selected features. It works on raw signals recorded during a short, resting state with eyes closed. This makes it fast, objective and easy to use. Three models were tested: LSTM, LSTM-CNN and LSTM-FCNN. Each model was trained to classify visual learners based on one-minute EEG segments. The results show that all three models performed well. The LSTM model gave the highest accuracy and perfect AUC scores. The LSTM-CNN model offered a better trade-off between speed and accuracy. The LSTM-FCNN model was the fastest and most lightweight but had slightly lower accuracy. This proves that it is possible to identify learning styles from resting-state EEG. The system avoids the need for task-based EEG which takes more time and effort. Instead, it uses signals when the person is relaxed which helps reduce noise and makes real-time use possible. The deep learning models learned useful patterns without needing manual help. The models worked directly with the raw signals. The best model LSTM-CNN combined pattern detection and time-based memory to give accurate and fast predictions. The results show strong separability between classes, as seen in the t-SNE plots. This approach is useful in real classrooms or online learning systems. Teachers or apps use it to adjust content based on student learning preferences. It helps improve learning outcomes by providing each learner with the right kind of material. This paper showed that deep learning classifies visual learning styles using only raw EEG. The system is accurate, fast and simple. It works in real time. In the future, this method is expected to extend to detect other styles or work with more diverse groups. It opens the door for smarter and more adaptive learning environments.

VIII. AUTHOR'S CONTRIBUTION

Conceptualization: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Methodology: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Investigation: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Discussion of Results: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Writing – Original Draft: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Writing – Review and Editing: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja. Atkar

Resources: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Supervision S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

Approval of the final text: S. Nagendram, Geeta Bhimrao Atkar, Desamala Prabhakara Rao, Tirumalasetti Lakshmi Narayana, Sunitha pachala, Shobana Gorintla, A.Gangadhar7, Koppolu Sireesha, Inakoti Ramesh Raja.

IX. REFERENCES

- [1] D. T. Willingham, E. M. Hughes, and D. G. Dobolyi, "The scientific status of learning styles theories," *Teaching of Psychology*, vol. 42, no. 3, pp. 266–271, 2015.
- [2] M. Alabi, "Visual learning: The power of visual aids and multimedia," 2024.
- [3] M. Marschark, A. Paivio, L. J. Spencer, A. Durkin, G. Borgna, C. Convertino, and E. Machmer, "Don't assume deaf students are visual learners," *Journal of developmental and physical disabilities*, vol. 29, pp. 153–171, 2017.

- [4] J. Reaves, T. Flavin, B. Mitra, K. Mahantesh, and V. Nagaraju, "Assessment and application of eeg: A literature review," *Journal of Applied Bioinformatics & Computational Biology*, vol. 10, no. 7, 2021.
- [5] A. Delorme and J. A. Martin, "Automated data cleaning for the muse eeg," in *2021 IEEE International Conference on Bioinformatics and Biomedicine(BIBM)*, pp. 1–5, IEEE, 2021.
- [6] X. Song, Z. Wang, and H. Wang, "Short-term load prediction with lstm and fcnn models based on attention mechanisms," in *Journal of Physics:Conference Series*, vol. 2741, p. 012026, IOP Publishing, 2024.
- [7] K. Brundiers, A. Wiek, and C. L. Redman, "Real-world learning opportunities in sustainability: from classroom into the real world," *International Journal of Sustainability in Higher Education*, vol. 11, no. 4, pp. 308–324, 2010.
- [8] W. du Toit, M. Venter, and D. Vandenneever, "Semi-synthetic eeg data for the evaluation of linear eeg cleaning methods," *International Journal Bioautomation*, vol. 27, no. 4, p. 193, 2023.
- [9] D. Badre, "On task: How our brain gets things done," 2022.
- [10] P. B. Weerakody, K. W. Wong, and G. Wang, "Policy gradient empowered lstm with dynamic skips for irregular time series data," *Applied soft computing*, vol. 142, p. 110314, 2023.
- [11] X. Jin, X. Yu, X. Wang, Y. Bai, T. Su, and J. Kong, "Prediction of time series with cnn and lstm," in *Proceedings of the 11th international conference on modelling, identification and control (ICMIC2019)*, pp. 631–641, Springer, 2020.
- [12] H. U. Amin, A. S. Malik, N. Badruddin, and W.-T. Chooi, "Brain behavior in learning and memory recall process: A high-resolution eeg analysis," in *The 15th International Conference on Biomedical Engineering: ICBME2013*, 4th to 7th December 2013, Singapore, pp. 683–686, Springer, 2014.
- [13] A. Selvaraj, V. Radhin, N. Benson, A. J. Mathew, et al., "Effect of pandemic based online education on teaching and learning system," *International journal of educational development*, vol. 85, p. 102444, 2021.
- [14] T. Doyle, *Helping students learn in a learner-centered environment: A guide to facilitating learning in higher education*. Taylor & Francis, 2023.
- [15] S. Thepsatitporn and C. Pichitpornchai, "Visual event-related potential studies supporting the validity of VARK learning styles' visual and read/write learners," *Advances in Physiology Education*, vol. 40, no. 2, pp. 206–212, Jun. 2016.
- [16] E. Kim, I. Yang, and S. Lim, "An EEG study on visual learners' performance: a scientific classifying task composed of pictures and words," *Advanced Science and Technology Letters*, vol. 92, pp. 86–91, 2015.
- [17] R. Rajkumar and V. Ganapathy, "Bio-inspired learning style chatbot inventory using brain computing interface to increase the efficiency of E-learning," *IEEE Access*, vol. 8, pp. 67377–67395, 2020.
- [18] L. I. F. Dutsinma and P. Temdee, "VARK learning style classification using decision tree with physiological signals," *Wireless Personal Communications*, vol. 115, no. 4, pp. 2875–2896, Dec. 2020.
- [19] M. M. Anoor, A. H. Jahidin, H. Arof, and M. S. A. M. Ali, "EEG-based intelligent system for cognitive behavior classification," *Journal of Intelligent & Fuzzy Systems*, vol. 39, no. 1, pp. 177–194, Jul. 2020.
- [20] M. S. A. Megat Ali, A. H. Jahidin, N. Md Tahir, and M. N. Taib, "Learning style classification via EEG sub-band spectral centroid frequency features," *International Journal of Electrical and Computer Engineering*, vol. 4, no. 6, pp. 931–938, Dec. 2014.
- [21] M. S. M. Ali et al., "Characterization of resting brainwave at different developmental stages of human learning," *International Journal of Control Theory and Applications*, vol. 9, no. 6, pp. 2985–2991, 2016.
- [22] M. S. M. Ali et al., "Classification of Kolb's learning styles using EEG sub-band spectral centroid frequencies and artificial neural network," *Asian Journal of Scientific Research*, vol. 9, no. 5, pp. 234–241, 2016.
- [23] A. Miljevic et al., "Electroencephalographic connectivity: A fundamental guide and checklist for optimal study design and evaluation," *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, vol. 7, no. 6, pp. 546–554, Jun. 2022.