mathematics

MDPI

Article

Deep Learning Algorithms for Behavioral Analysis in Diagnosing Neurodevelopmental Disorders

Hasan Alkahtani [1,2], Zeyad A. T. Ahmed [3], Theyazn H. H. Aldhyani [1,4,*], Mukti E. Jadhav [5] and Ahmed Abdullah Alqarni [1,6]

[1] King Salman Center for Disability Research, P.O. Box 94682, Riyadh 11614, Saudi Arabia
[2] Computer Science Department, King Faisal University, P.O. Box 400, Al-Ahsa 31982, Saudi Arabia
[3] Department of Computer Science, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad 431004, India
[4] Applied College in Abqaiq, King Faisal University, P.O. Box 400, Al-Ahsa 31982, Saudi Arabia
[5] Department of Computer Sciences, Shri Shivaji Science and Arts College, Chikhli Dist Buldana 443201, India
[6] Department of Computer Sciences and Information Technology, Al Baha University, P.O. Box 1988, Al Baha 65431, Saudi Arabia
* Correspondence: taldhyani@kfu.edu.sa

Abstract: Autism spectrum disorder (ASD), or autism, can be diagnosed based on a lack of behavioral skills and social communication. The most prominent method of diagnosing ASD in children is observing the child's behavior, including some of the signs that the child repeats. Hand flapping is a common stimming behavior in children with ASD. This research paper aims to identify children's abnormal behavior, which might be a sign of autism, using videos recorded in a natural setting during the children's regular activities. Specifically, this study seeks to classify self-stimulatory activities, such as hand flapping, as well as normal behavior in real-time. Two deep learning video classification methods are used to be trained on the publicly available Self-Stimulatory Behavior Dataset (SSBD). The first method is VGG-16-LSTM; VGG-16 to spatial feature extraction and long short-term memory networks (LSTM) for temporal features. The second method is a long-term recurrent convolutional network (LRCN) that learns spatial and temporal features immediately in end-to-end training. The VGG-16-LSTM achieved 0.93% on the testing set, while the LRCN model achieved an accuracy of 0.96% on the testing set.

Keywords: autism spectrum disorder; deep learning; computer vision; behavior analysis; hand flapping; human action recognition; video classification; long short-term memory networks; convolutional neural network

MSC: 68Q32

check for updates

Citation: Alkahtani, H.; Ahmed, Z.A.T.; Aldhyani, T.H.H.; Jadhav, M.E.; Alqarni, A.A. Deep Learning Algorithms for Behavioral Analysis in Diagnosing Neurodevelopmental Disorders. *Mathematics* **2023**, *11*, 4208. https://doi.org/10.3390/math11194208

Academic Editors: José Rodellar, Francesc Pozo and Yolanda Vidal

Received: 6 September 2023
Revised: 30 September 2023
Accepted: 2 October 2023
Published: 9 October 2023

1. Introduction

Autism spectrum disorder (ASD) is a comprehensive term that comprises a diverse range of diseases distinguished by challenges in social interaction and communication, including repetitive behavioral patterns [1]. ASD has a significant impact on the lives of children and their families, yet there is no specific therapy for it. The number of children with autism is increasing annually, leading to increased anxiety and fear in families for their children, prompting them to have their children's behavior tested by psychologists or with diagnostic scans of their brain functions to ensure that they are free from autism.

There are behavioral and motor signs through which autism can be diagnosed, including hand flapping, which refers to the closing and opening of the fingers quickly for a few moments with the movement of the arms. This condition can occur due to involuntary muscle spasms, or so-called myoclonic tremors, which may arise at any time anywhere in the body, including the hands. Psychiatry experts define such self-stimulating behaviors as hand flapping, shaking, etc. One of the reasons for this is that the child does not obtain the appropriate sensory inputs they need at that time, and thus they cannot correctly express their feelings. In most cases, abnormal behaviors occur in response to excitement, anxiety, or anger. Hand flapping in children is one of the most common early signs of ASD.

Analyzing children's behavior to diagnose whether they have autism requires a considerable amount of time to observe them during their daily activities. The abnormal behaviors that are signs of autism appear randomly for a short period of time. Monitoring their behavior by psychologists requires time and effort, and the financial cost is paid by the family. In recent years, deep learning algorithms and hardware advancements have supported the application of artificial intelligence technologies to detect self-stimulatory behaviors automatically.

The motivation behind this study is to develop a system capable of detecting the early signs of autism in children. This would assist clinicians in selecting the appropriate behavioral therapy and support families in remote areas where access to advanced diagnostic devices is limited.

The use of modern technology can contribute to the detection of autism in children, helping to obtain a proper diagnosis and enabling early treatment intervention. In this research paper, a real-time system using deep learning and computer vision to monitor and detect abnormal behavior in children was developed, which might help to diagnose autism. Two deep learning methods VGG-16 and LRCN were trained on the Self-Stimulatory Behavior Dataset (SSBD) [2]. The SSBD is a publicly available dataset collected from children with autism, which includes videos posted by parents/caregivers on various internet sites, such as YouTube, covering a range of children's daily activities. The models developed in this research can extract the kinetic features resulting from the movement of the hands in the videos and classify natural and abnormal movements in real-time.

This work makes important contributions in developing and validating deep learning solutions for the real-time automated screening of autism spectrum disorder (ASD) from video data. Specifically, novel convolutional and recurrent neural network architectures are proposed to classify hand flapping behaviors associated with ASD. The long-term recurrent convolutional network (LRCN) model achieves the state-of-the-art accuracy of 96% on the Self-Stimulatory Behavior Dataset, demonstrating the feasibility of computer vision techniques for the automated detection of behavioral markers. The deep network implementations, benchmark results, and demonstration of real-time screening from videos could significantly advance research and clinical practice in automated ASD screening. This work establishes deep learning as a promising approach for recognizing subtle behaviors indicative of ASD from widely available video sources, potentially enabling large-scale screenings. The accurate real-time detection of hand flapping behaviors contributes key techniques and benchmarks to work toward computer-aided diagnosis and early intervention for ASD.

2. Related Work

Researchers have proposed several methods for studying and monitoring behavior to detect autism. Some studies have relied on observation, using videos to analyze motor behavior [2–7]. Other studies have used sensors attached to the child's hand or body to collect acceleration data [8–10]. There are studies based on eye tracking [11] as well as electroencephalography (EEG), which aids in the acquisition of brain signals corresponding to different states from the scalp surface area [12]. In addition, functional magnetic resonance imaging (functional MRI; fMRI) has been used to evaluate brain activity by detecting changes in blood flow, with most researchers using the ABIDE dataset [13]. Finally, studies have extracted information from facial features to detect autism using deep learning [14].

2.1. Video-Based Behavior Analysis

This subsection discusses previous studies on behavior analysis based on videos. Rajagopalan et al. [2,3] proposed the SSBD, which consists of videos of autistic children during their daily activities. They used a histogram of dominant motions with a histogram of optical flow. Their binary classification model of headbanging and spinning achieved an accuracy of 86.6%, while their classification of headbanging, spinning, and hand flapping in a three-way challenge achieved an accuracy of 76.3%. MediaPipe was used to extract hand landmarks from videos from the same dataset [2], which were then fed into LSTM and MobileNetV2. This model was presented by Lakkapragada et al. [4]. They used two classes of SSBD: hand movement and control. They achieved a high accuracy result using MobileNetV2, with a test F1 score of 84.0 ± 3.7. Ali et al. [5] developed a model for ASD behavior diagnosis. They collected and annotated a set of recordings of stereotypical children's behavior recorded in an uncontrolled context during their ASD diagnosis. The dataset (388 videos) was separated into five categories: arm-flapping, clapping, to-taste, jump-up, and others. They achieved the best accuracy when using a fusion of the two streams of Inflated 3D Networks (I3D), three channels; red, green, and blue (RGB), and optical flow (85.6 to 86.04%). Rehg et al. [6] analyzed various children's behaviors in their dataset, the Multimodal Dyadic Behavior Dataset (MMDB). Over 160 structured adult–child social interactions were collected to provide data on various behaviors, including motor, gestural, emotional, and vocal behaviors, using cameras and sensors.

Head movements have also been used to detect autism. For example, Zhao et al. [7] analyzed the behavior of the two groups of children using videos. The first group consisted of 20 autistic people, while the second consisted of 23 with typical development. Each person was asked ten questions and asked to answer by shaking their head or nodding. Using OpenFace, six features were extracted, namely, head rotation range (RR) and the amount of rotation per minute (ARPM) in the pitch, yaw, and roll directions. The decision tree classifier with two features had the highest classification accuracy of 92.11%.

2.2. Wearable-Sensor-Based Behavior Analysis

This subsection discusses monitoring a child's hands/body movements for a certain length of time using wearable sensors to identify self-stimulatory behaviors. Westyn et al. [8] studied self-stimulatory behavior using three-axis accelerometer modules. Non-autistic people wore accelerators on their right wrist, the back of their waist, and their left ankle. To gather data, non-autistic individuals were instructed to replicate the behaviors of an autistic patient. Hidden Markov models (HMMs) were then used to analyze the accelerometer data. Ploetz et al. [9] obtained acceleration data by attaching sensors to each participant's limbs. Behavioral episodes were created from the collected data in order to detect aggression, disruption, and self-injury behaviors. These episodes were then used to extract features that were used to train models. Sarker et al. [10] used smartwatch accelerometer sensors to collect data on repetitive behaviors, such as hand flapping, headbanging, and repetitive dropping, to inform an intervention for individuals with autism. Their model achieved an accuracy of 69%.

Recently, the use of eye-tracking technology has been applied in the field of autism detection through patterns of looking behavior while watching stimuli such as dynamic or static [11,15]. However, one of the disadvantages of this technique is that it needs to prepare a suitable place according to a specific test protocol so that the child directs their eyes to the eye-tracking device installed at the bottom of the screen, and if they look to a different place, the eye tracker would not be able to record the eye movement. In addition, eye-tracking devices are very expensive, making it difficult for developing countries to use this technology at the present time. Also, using a sensor that is attached to the child's body, for example, in the arm or chest, may be inappropriate because children, especially children with autism, are allergic to things they are not used to.

To overcome these difficulties, this study seeks to find an easy and low-cost test method by which it can help diagnose children with autism by taking a video clip during the child's daily activity and sending it to the proposed system, which will, in turn, detect if their abnormal behavior indicates signs of autism.

3. Materials and Methods

Hand flapping is one of the most commonly observed stimming behaviors in children with ASD. This behavior is characterized by a repeated activity that may last for a short period. Hand flapping can manifest as a stimming behavior in several distinct ways, including forcefully moving fingers, finger clicking, and broader arm motions. This section elaborates on the systematic process steps of the proposed work designed for the video classification of specific human activities, like hand flapping, in contrast to normal behavior (as illustrated in Figure 1).

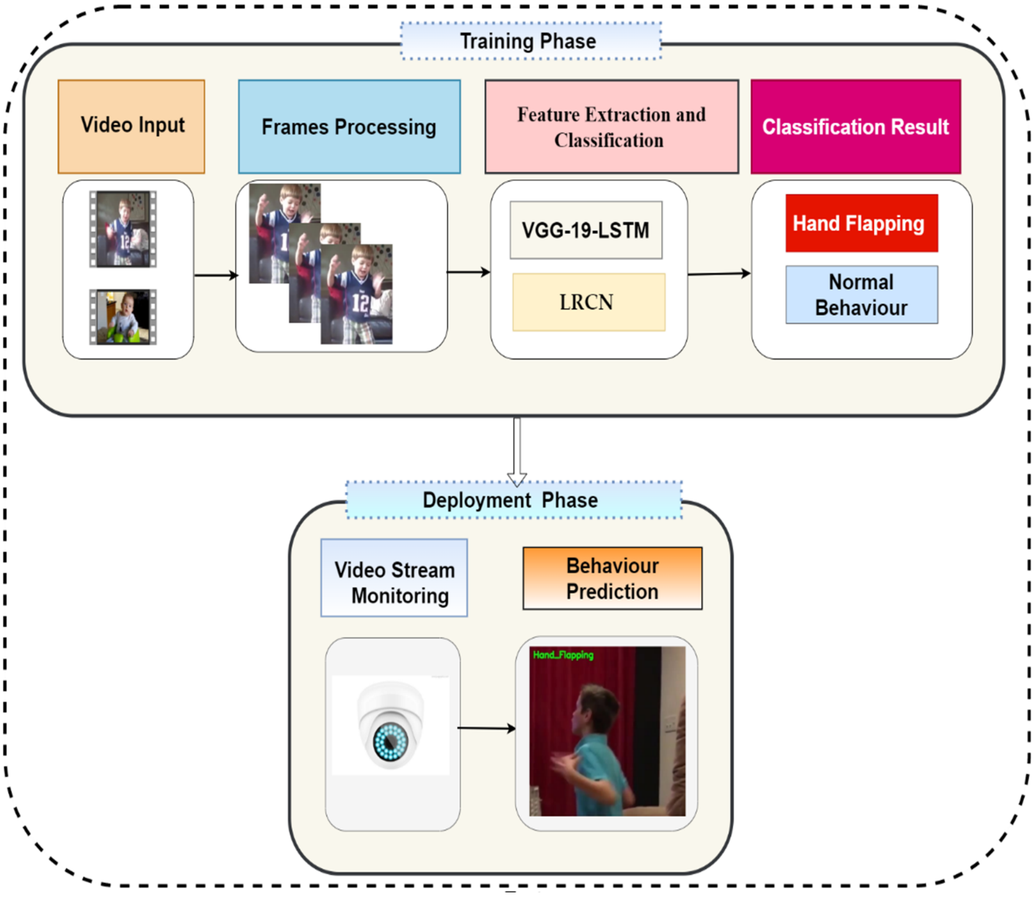


Figure 1. The proposed methodology for detecting hand flapping.

The initial step in our methodology is the gathering of a comprehensive video dataset. Following this, the second step entails data preprocessing, which includes tasks such as the frame extraction, normalization, and labeling of the videos. Subsequently, the third step involves partitioning the data into distinct training and testing sets. In the fourth step, we focus on training the deep learning model based on the preprocessed data. The fifth step involves meticulously observing and evaluating the performance of these trained models. In the final, crucial step, the models' effectiveness is tested using a real-time video dataset, allowing us to gauge its capability in analyzing children's behaviors for potential autism detection.

3.1. Dataset

The deep learning models were trained on the SSBD [2]. The SSBD is a publicly available dataset collected from children with autism. Figure 2 shows some frames from the SSBD. The data consist of recorded videos showing autistic children's behaviors, such as hand flapping, spinning, and headbanging (Figure 2). These videos were shared on publicly available internet sites by parents and caregivers. The original dataset [2] consists of 75 videos, which were collected from websites such as YouTube, Vimeo, and Dailymotion. Due to YouTube privacy concerns, only 66 were downloaded.

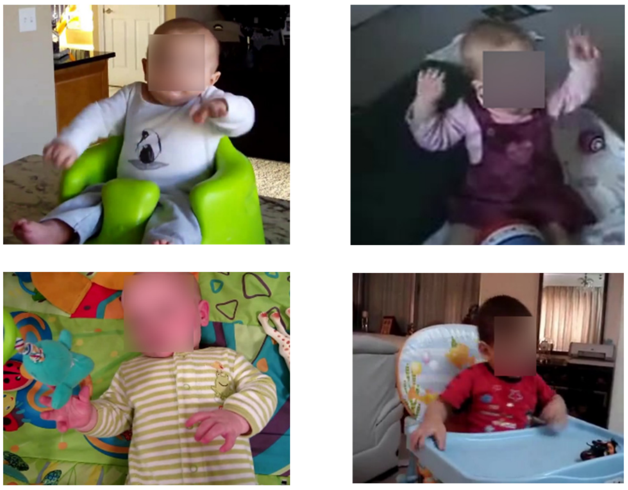


Figure 2. Some frames of the two types of behavior, with the first row showing hand flapping and the second showing normal behavior. The faces of the children are blurred to protect their privacy.

3.2. Preprocessing

The URLs of 75 YouTube videos were provided by the SSBD creators [2], annotated at the beginning and end of abnormal behaviors indicating autism. The average duration per video is 90 s. The videos in this dataset contain several behavioral movements in different periods. Thus, there was a need to cut the sections based on the behavioral movements [4], such as normal behavioral and hand flapping or hand movement. After the videos were cut, a new dataset was created consisting of two classes of behavior, with each video being 2–4 s in duration. Each video was a sequence of updated frames to create the appearance of motion. We extracted frames from each video and preprocessed them as follows. First, we used a video file path as input. Then, we read the video file frame by frame, resized and normalized each frame, and appended them into a list. We created two NumPy arrays: all frame and class labels. The class labels were converted to a one-hot encoded format. Then, the dataset was split into 80% for training and 20% for testing.

3.3. Deep Learning Algorithms

This paper demonstrates the use of robust deep learning models for human behavior recognition and classification through videos. However, videos are sequences of images arranged in a specific order to make motion. Various methods could be used for video classification, such as CNN integrated with RNN, 3D convolutional networks, and long short-term memory networks (LSTMs). This research paper proposed the use of two deep learning models: VGG-16- LSTM and LRCN models. In the following sections, we provide a comprehensive description of the fundamental components encompassed in our proposed models, as well as a thorough showing of their architectural designs.

3.3.1. Convolutional Neural Network Components

In the ensuing subsection, we elucidate the foundational concepts underpinning the architectures of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). We commence by delineating the characteristics and functionalities of the convolution layer, max-pooling layer, and dense layer. Thereafter, a detailed exposition on the RNN, with a particular emphasis on the Long Short-Term Memory (LSTM) network, is presented.

A CNN (or ConvNet) is a type of neural network used in deep learning. It is generally used to detect patterns in images as well as for computer vision, natural language processing, spatial data analysis, signal processing, and other uses. A CNN is inspired by the structure of the visual cortex and closely resembles the connecting arrangement of neurons in the human brain. Convolution is one of the most important parts of artificial neural networks, which is responsible for feature maps. The best way to deal with spatial features from frames is to use a CNN framework. In convolutional layers, several filters are used to extract important features from the image. The output is sent to a fully connected "dense" network after passing through a series of convolutional layers.

Convolutional Layers (Conv2D)

The two-dimensional (2D) convolution layer, known as Conv2D, is the most commonly used form of convolution. A filter or kernel slides through the 2D input data in a Conv2D layer, performing elementwise multiplication. The outcome will be aggregated into a single output pixel as a result. For each spot it slides over, the kernel will perform the same operation, transforming a 2D feature matrix into a new 2D feature matrix. The images are compressed in a convolutional layer to make them easier to process while retaining the essential features that help in the prediction process.

Batch Normalization

Batch normalization is a technique for speeding up and stabilizing artificial neural networks by re-centering and rescaling the inputs to the layers [16].

Pooling Layer

Considering the large number of parameters generated by convolution layers, we decided to use one of two approaches to minimize weights due to their time-consuming mathematics. One of the methods, max-pooling or average-pooling, was used to lower the weights. Maximum values in stride were used to calculate max-pooling. The average pooling was calculated by obtaining the mean value of each window in stride. In our model, MaxPooling2D was used to calculate the maximum values in stride, followed by GlobalAveragePooling2D. Each feature map was converted into a single value using global average pooling, since the number of feature maps was large. For an input of H × W × C, it returned a tensor with dimensions of 1 × C by taking the global average of height and width, using the global average pooling strategy to reduce overfitting and enhance the model performance.

Dense Layer and Activation Function

The dense layer is a layer of neurons where each neuron obtains information from every other neuron in its preceding highly dense layer. The output of convolutional layers is sent into a dense layer, which is utilized to classify the image. The training of the neural networks followed two paths: forward and backpropagation. Here, the flattened output was fed to the neural network in the forward path. The neural network reduced the loss error in backpropagation and learned more features by attempting training iterations. The SoftMax/sigmoid function received the parameters from the dense layer and mathematically calculated the probability to predict the output. The input of the network can be represented by x, the output represented by y, the bias represented by b, and the weight represented by w. The activation function is then used to calculate the output based on Formula (1).

$$y = x1 \times w1 + x2 \times w2 + b \tag{1}$$

Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) units, a subclass of recurrent neural networks (RNNs), have emerged as an innovative solution to inherent challenges encountered in traditional RNN architectures. Among the most pressing challenges is the susceptibility of simple RNNs to vanishing or exploding gradient problems, especially when dealing with long-term dependencies. LSTMs, distinguished by their intricate architecture, are adept at mitigating these issues [17–19].

At the heart of the LSTM's architecture lies the concept of the memory cell or cell state, which encapsulates the unit's capacity for long-term memory retention. The LSTM framework integrates three cardinal gates: input, forget, and output gates, as shown in Figure 3. The input gate assesses the salience of incoming information, determining its retention potential. In juxtaposition, the forget gate appraises the relevance of existing data, orchestrating its potential excision if deemed non-essential. The output gate, meanwhile, evaluates the pertinence of specific data within the temporal context, governing its deployment. Two types of inputs are used by the LSTM cell: the output from the previous hidden state and the observation at time *t*. With the exception of the concealed state, there is no knowledge about the past that can be recalled. The cell state, or memory cell, is the fundamental concept of LSTMs.

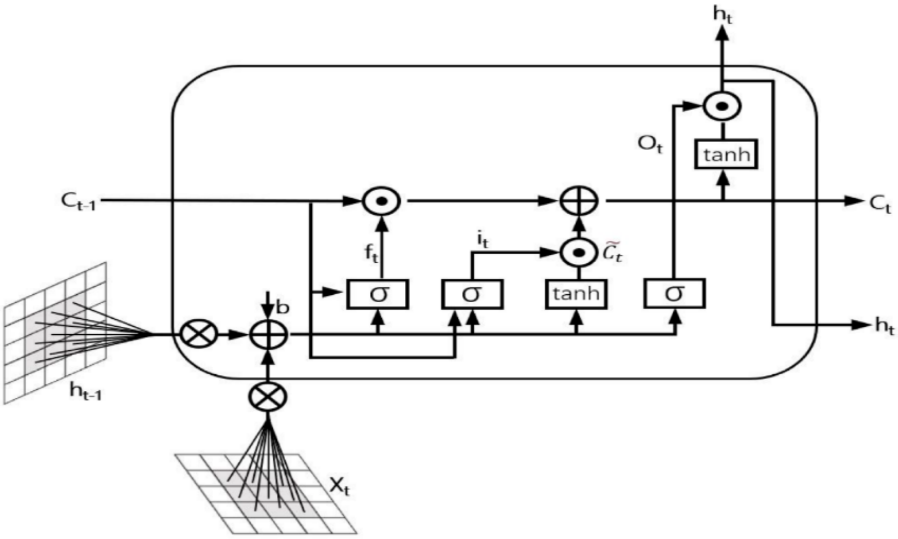


Figure 3. One cell structure of CNN-LSTM.

Assuming that i_t, f_t, and o_t represent the input gate, forget gate, and output gate, respectively, and ∘ represents a sigmoid function, then w is the weight of the respective gate (x) neurons, h_{t-1} is the output of the previous LSTM block at timestamp ($t-1$), x_t is the input at the current timestamp, and b biases the respective gate (x). The recursive nature of the cell is shown by the looping arrows. With this technique, information from prior periods may be kept in the LSTM cell. The forget gate, which is below the cell state, changes the cell state. The input modulation gate changes the cell state. The previous cell state is forgotten by multiplying it with the forget gate, and new information is added to the prior cell state through the output of the input gates, according to Equation (2). The cell state can be calculated using Equation (2).

$$\boldsymbol{c}_t = f_t \circ c_{t-1} + i_t \circ \widetilde{c}_t \tag{2}$$

The forget vector is sometimes referred to as the forget gate. The output of the forget gate is used to determine what information the cell state should get rid of by multiplying 0 by a place in the matrix. If the forget gate's output is 1, then the data are kept in the current cell state for future use. The sigmoid function is derived from the equation and applied to the weighted input/observation as well as the prior hidden state. The forget gate can be calculated using Equation (3).

$$\boldsymbol{f}_t = \sigma\Big(W_f[h_{t-1}, x_t] + b_t\Big) \tag{3}$$

The input gate is another name for the save vector. Each cell has a set of gates that regulate whether information may be stored in the long-term memory. The activation functions of each gate are critical. In a sigmoid function, the input range is [0, 1]. The sigmoid function can only add memory, not erase or forget it, as the equation for the cell state is a sum of the preceding cell state. A float number between [0 and 1] cannot be forgotten; it can only be added between [0 and 1]. Input modulation gates have a tanh activation function for this reason: [−1, 1] tanh has a memory-erasing range of [−1, 1]. The input gate can be calculated using Equation (4) and the input modulation gate using Equation (5).

$$\boldsymbol{i}_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{4}$$

$$\widetilde{\boldsymbol{c}}_t = tanh(W_c[h_{t-1}, x_t] + b_c) \tag{5}$$

3.4. Architecture of the Deep Learning Models

In this section, the architecture of advanced deep learning models, proposed for human activity classification, is discussed. These models aim to detect specific patterns, such as hand flapping, which is often associated with the early signs of autism. The proposed models are the VGG-16-LSTM and LRCN.

3.4.1. VGG16-LSTM Model

Transfer learning refers to the practice of using feature representations obtained from a pre-existing model in order to circumvent the need for training a new model from its initial state. The pre-trained models are often trained on huge datasets that serve as a benchmark at the forefront of computer vision. The weights derived from the models may be reapplied to further computer vision applications.

The pre-trained model involves a considerable amount of time; in training for certain tasks, it might take up to a few weeks to train when using a massive dataset and a high-configuration computer and GPUs. VGG-16 [20] is trained on the ImageNet database for the purpose of image recognition. Google's ImageNet database [21] has 14 million images of different objects. Reduced training time and increased generalization are the biggest benefits of transfer learning. Figure 4 display structure of VGG16-LSTM model.

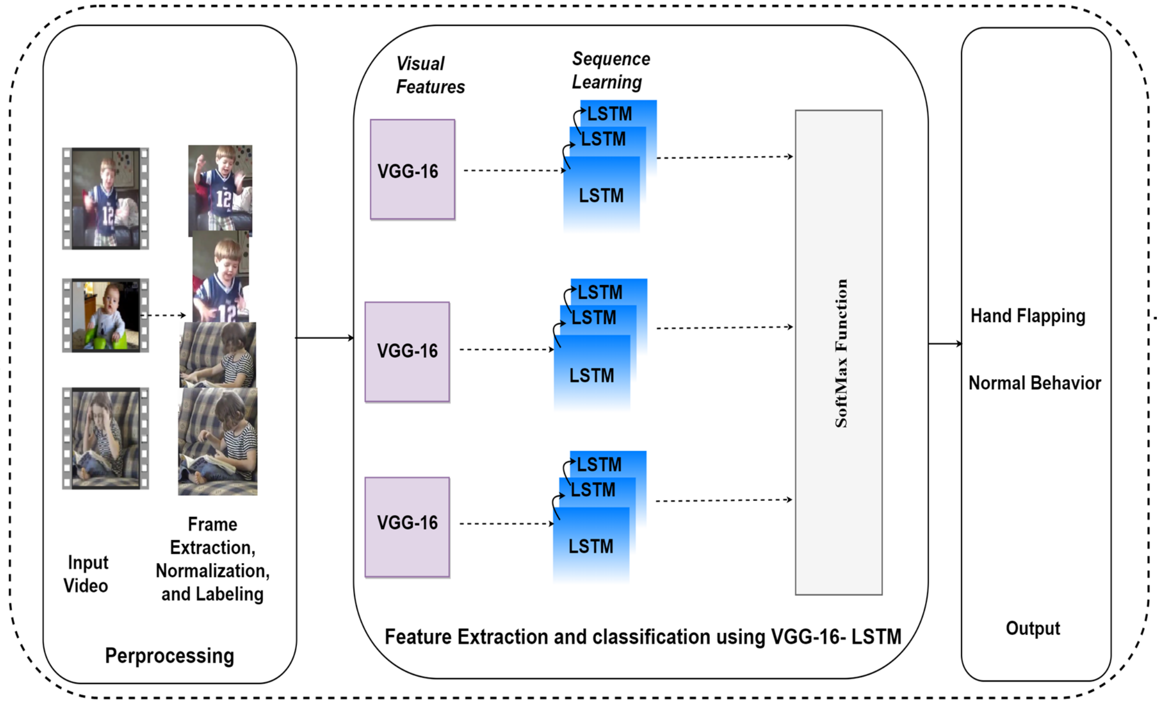


Figure 4. The architecture of the VGG-16-LSTM model.

The VGG-16 backbone is used for features extracted from the frames and which are then input into the LSTM. The LSTM unit is a memory cell that stores the information of the entries that have been seen up to that point in time for each time step. In other words, VGG16 is used for the spatial feature and LSTM for the temporal feature. The 16 in VGG16 refers to the fact that there are 16 layers with weights. The VGG16 model has 21 layers in total, 13 convolutional layers, 5 max-pooling layers, and 3 dense layers, but there are only 16 weight layers, also known as the learnable parameter layers [5]. VGG16 takes input tensors with a 224 × 244 image size and three RGB channels. The most notable aspect of VGG16 is that its creators focused on creating convolution layers of 3 × 3 filters with stride 1, and they constantly employed the same padding and max-pooling layers of 2 × 2 filters with stride 2, rather than having a large number of hyper-parameters. The convolution and max-pooling layers have a consistent configuration across the whole design. Conv-1 Layer contains 64 filters, Conv-2 Layer contains 128 filters, Conv-3 Layer contains 256 filters, and Conv-4 and Conv-5 Layers each include 512 filters. Three Fully Connected (FC) layers are added after a stack of convolutional layers; the first two have 4096 channels each, while the third completes classification for the 1000-way ImageNet Large Scale Visual Recognition Challenge, and so has 1000 channels (one for each class). The final layer is called the softMax layer.

In our VGG-16-LSTM model, we utilized VGG-16 as a feature extractor by using its pre-trained weights on ImageNet. To do this, we froze the convolutional layers of VGG-16 so their weights remained fixed during training on our dataset. This allowed us to extract generalized visual features from the video frames using this pretrained network. After extracting spatial features from each frame with VGG-16, we added a max-pooling layer to downsample the feature maps and reduce their dimensionality. The pooled features for each frame were then flattened into vectors and sequenced chronologically.

These sequential feature vectors were passed into LSTM blocks containing 32 internal units. The LSTM analyzed the temporal relationships and the evolution of the features over time. This enabled the modeling of the dynamic hand flapping motions across frames.

Finally, the LSTM output for each time segment was fed into a dense layer with Softmax activation for classification into either the hand flapping or normal behavior categories. We effectively combined the spatial feature extraction capabilities of VGG-16 with the temporal sequence modeling of LSTMs through this architecture. The VGG16-LSTM Model Parameters are presented in Table 1.

Table 1. VGG16-LSTM Model Parameters.

Parameter	Details
Base Model	VGG16 (pretrained on ImageNet)
Trainable Layers	Fine-tuned layer
Feature Extractor	TimeDistributed VGG16
Flatten Layer	TimeDistributed Flatten()
LSTM Layer	32 units
Output Layer	Dense with SoftMax activation
Loss Function	Categorical Crossentropy
Optimizer	Adam
Training Epochs	100
Batch Size	4
Early Stopping	Patience: 15

3.4.2. Long-Term Recurrent Convolutional Model

An LRCN is a type of deep learning network used for visual recognition and description. Combining CNNs for visual feature extraction from video frames and LSTMs for image embedding transformation is the essential point of LRCNs. In other words, convolutional layers are used to extract spatial features from the frames, and the spatial features are sent to LSTM layers at each time step to model temporal sequences, as shown in Figure 5. In this way, the network learns spatial and temporal features immediately in an end-to-end training process, which makes the model more stable [22–26].

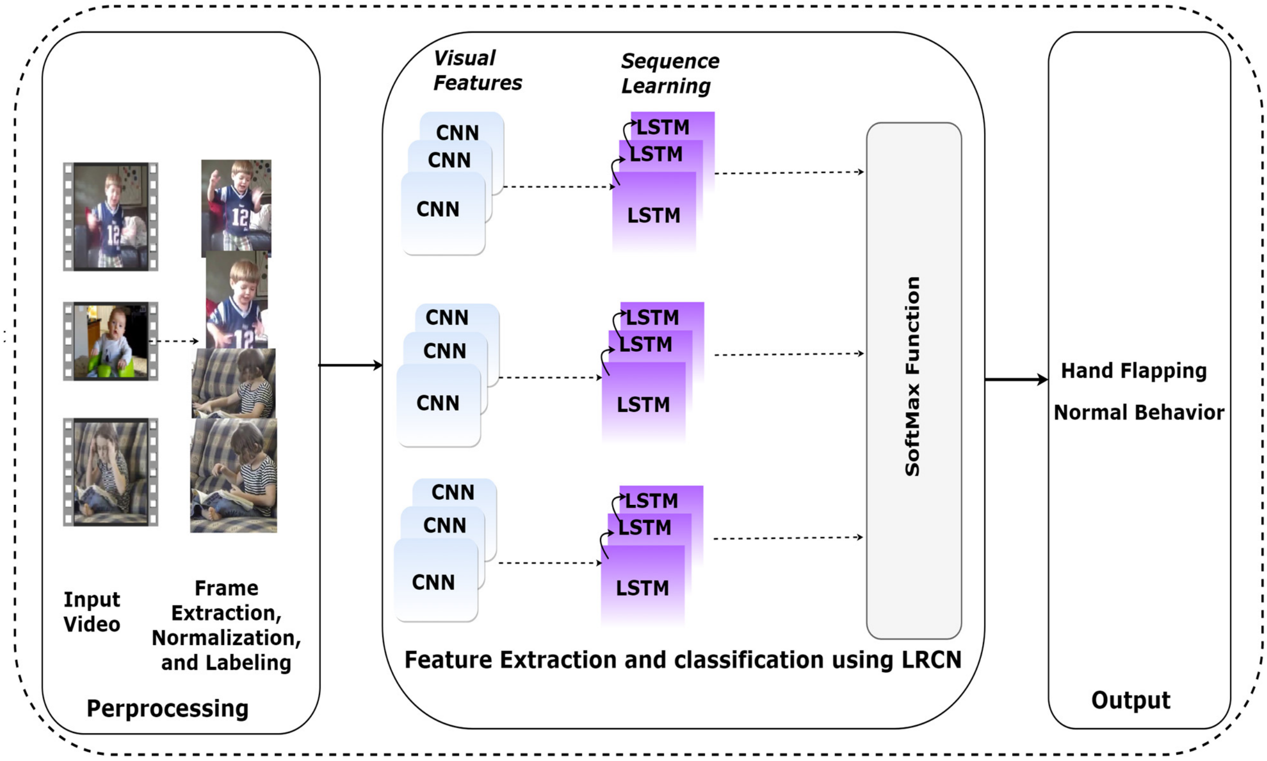


Figure 5. The architecture of the LRCN model.

The LRCN uses a CNN to process the different lengths of visual input, and the outputs go into a stack of recurrent sequence models called an LSTM. The final result of the sequence models is a prediction that can be variable in length. Due to its time-varying inputs and outputs, the LRCN is a perfect model for tasks like activity recognition and video captioning and description. The mathematical description of LRCN models is as follows: When a visual input vector v_t is sent via feature transformation $\phi(v_t)$ parametrized as v, a fixed-length vector $\phi_V(v_t) \in \Re^d$ is produced. Here is how the sequence model works: The feature space representation of the visual input sequence $\langle \phi_1, \ldots, \phi_T \rangle$ is mapped with parameter w to a sequence of hidden states h_t and output z_t/y_t. At each time step t, W transfers an input x_t (which in the majority of experiments here is $\phi_V(v_t)$) and a hidden state h_{t-1} to an output z_t/y_t and an updated hidden state h_t. Mapping is carried out in a sequential manner: $h_1 = f_W(x_1, 0), h_2 = f_W(x_2, h_1)$ up to h_T. At the end of each time step, the output is calculated as follows:

$$P(y_t) = \frac{\exp(W_{zc} z_{t,c} + b_c)}{\sum_{c' \in C} \exp(W_{zc'} z_{t,c'} + b_{c'})} \tag{6}$$

This means that most of the time ϕ_v (v_t), the convolutional inference, and training can be completed in parallel over time. End-to-end training is provided (i.e., on both v and w). The cost function is equivalent to the negative log probability of the training data (x, y), which is calculated using Formula (7). Minimization is accomplished through the use of basic Stochastic gradient descent (SGD) backpropagation through time (BPTT for the

LSTM). In certain circumstances, ConvNet is pretrained independently in order to obtain a higher rate of convergence.

$$-\ell(V, W) = -\log P_{V,W}(y_t \mid x_{1:t}, y_{1:t-1}) \tag{7}$$

The LRCN model used in CNNs for spatial feature learning and in an LSTM for temporal modelling. This architecture is well suited for video classification tasks. The input to the LRCN is a sequence of video frames, with each frame containing color images of fixed height and width. The model first processes each frame using four TimeDistributed convolutional layers. These convolution operations extract spatial features and patterns from the individual frames. Max-pooling and dropout layers in between the convolutional layers help reduce overfitting. After the convolutional feature extraction, a TimeDistributed flattened layer transforms the spatial representations into 1D feature vectors for each frame. These vectors sequence the frame-level features chronologically.

An LSTM layer with 32 memory units is then applied over the sequential feature vectors. This recurrent layer analyzes temporal relationships and context across the frames. The LSTM's memory capabilities are crucial for modelling the dynamics in the video.

Finally, a dense output layer with softmax activation predicts class probabilities. The LRCN is trained end-to-end via backpropagation for video classification into defined categories. Optimization is completed using RMSprop and categorical cross-entropy loss.

In summary, the LRCN architecture effectively combines the complementary strengths of CNNs and LSTM networks. The convolutional layers extract informative spatial features from individual frames, while the LSTM models temporal dependencies across frames for video analysis. The LRCN Model Parameters are presented in Table 2.

Table 2. LRCN Model Parameters.

Parameter	Details
Convolutional Layers	– 16 filters, (3, 3) kernel, 'same' padding, ReLU activation
	– MaxPooling2D (4 × 4)
	– Dropout (rate: 0.25)
	– 32 filters, (3, 3) kernel, 'same' padding, ReLU activation
	– MaxPooling2D (4 × 4)
	– Dropout (rate: 0.25)
	– 64 filters, (3, 3) kernel, 'same' padding, ReLU activation
	– MaxPooling2D (2 × 2)
	– Dropout (rate: 0.25)
	– 64 filters, (3, 3) kernel, 'same' padding, ReLU activation
	– MaxPooling2D (2 × 2)
Flatten Layer	TimeDistributed Flatten()
LSTM Layer	32 units
Output Layer	Dense with softmax activation
Loss Function	Categorical Crossentropy
Optimizer	RMSprop
Training Epochs	100
Batch Size	6
Early Stopping	Patience: 15

4. Experimental Results

The deep learning models were trained on the SSBD [2] for children's behavior classification in order to extract the abnormal behavior traits that indicate autism. The models were trained on an Intel 8th Generation Core i7 laptop with NVidia GeForce GTX 1070 GPU. We utilized the Python programming language in conjunction with prominent deep learning libraries, namely TensorFlow and Keras [27–29].

4.1. Evaluation Metrics

A classification report is a statistical measurement of performance in the deep learning field. Its objective is to demonstrate the performance of the training classification model, including its accuracy, recall, F1 score, and overall support.

Accuracy is defined as the ratio of correct predictions to the total number of predictions made. *Accuracy* is calculated as follows:

$$Accuracy = \frac{True\ Positive\ + True\ Negative}{True\ Positive\ + True\ Negative + \ False\ Positive\ + False\ Negative} \tag{8}$$

Precision is defined as the proportion of true positives to the total number of true and false positives in a particular sample. *Precision* is calculated as follows:

$$Precision = \frac{True\ Positive}{True\ Positive\ +\ False\ Positive} \tag{9}$$

Recall is defined as the proportion of true positives to the sum of true positives and false negatives. *Recall* is calculated as follows:

$$Recall = \frac{True\ Positive}{True\ Positive\ +\ False\ Negative} \tag{10}$$

The F1 score is a weighted harmonic mean of accuracy and recall. When the F1 score is closer to the number 1.0, it means the model has high performance. The F1 score is calculated as follows:

$$F1\ Score = \frac{2 \times (\ Precision\ \times\ Recall\ )}{Precision\ +\ Recall} \tag{11}$$

4.2. Results

Table 3 summarizes the classification reports of the VGG-16-LSTM and LRCN deep learning models. Due to the VGG-16-LSTM and LRCN models, it scored well in the hand flapping class (recall = 100%) and normal behavior class (recall = 87%). The LSTM captures temporal dynamics and sequences, making it ideal for modeling motion-based video data. The LSTM model finds the most accurate class samples. The VGG-16-LSTM and LRCN models were scored (precision = 100%) for normal behavior, showing that they can accurately predict this class. The VGG-16-LSTM model scored a lower percentage (recall = 87%) in the normal behavior class due to the fact that the numbers of the class were classified as false negatives. The pretrained CNN and LSTM models balance spatial appearance and temporal patterns; according to F1 score metrics, the VGG-16-LSTM model was successful (F1 score = 93% for both classes).

Table 3. Classification results.

Models	Class	Precision	Recall	F1 Score	Support
VGG-16-LSTM	Hand flapping	0.88	1.00	0.93	14
	Normal behavior	1.00	0.87	0.93	15
LRCN	Hand flapping	0.93	1.00	0.97	14
	Normal behavior	1.00	0.93	0.97	15

The LRCN model's high score showed high precision (precision = 93–1.00%) for both hand flapping and normal behavior, and it showed accurate predictions for both classes. The tailored convolutional architecture and RMSprop optimization may extract distinguishing features. Sequence modeling helped identify true simples with recall values of (recall = 100%) for "Hand Flapping Class" and (recall = 93%) for "Normal Behavior." Finally, the LRCN model had high performance for detecting ASD.

4.3. Performance of the Proposed Models

The VGG-16-LSTM model was trained on 100 epochs using four batch sizes and the Adam optimizer [26]. The early stop method ended training after 33 epochs to prevent overfitting and minimize training time. The model achieved 100% accuracy on training and 93% on validation data (Figure 6a). The model training and validation loss are shown in Figure 6b.

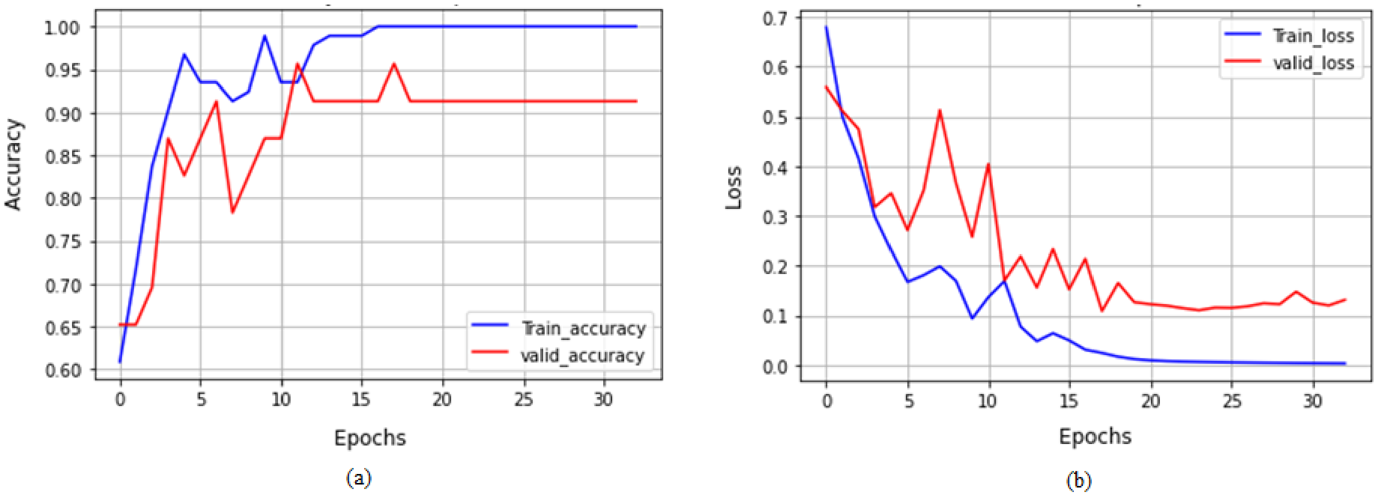


Figure 6. Plot of VGG-16 LSTM. (**a**) Accuracy. (**b**) Loss.

This LRCN model was trained on 100 epochs using six batch sizes and the RMSprop optimizer. The early stop method ended training after 50 epochs. The model achieved accuracies of 0.967% and 0.965% on the training and validation data, respectively. Figure 7a shows the accuracy plot of training and validation, while Figure 7b shows the model loss in training and validation.

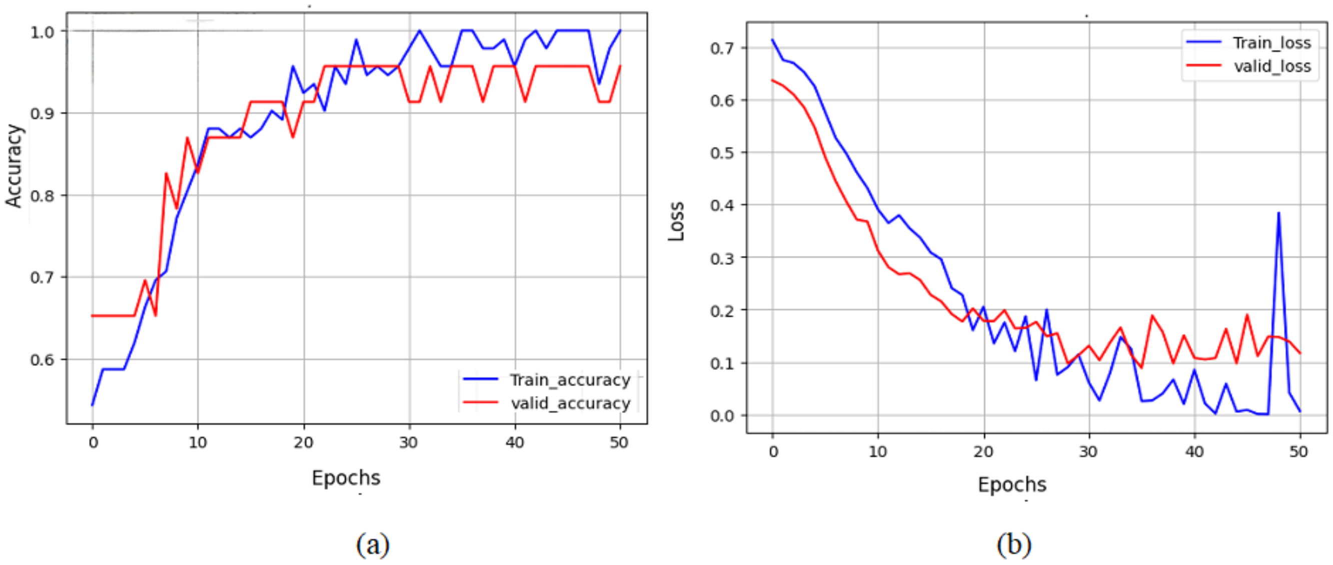
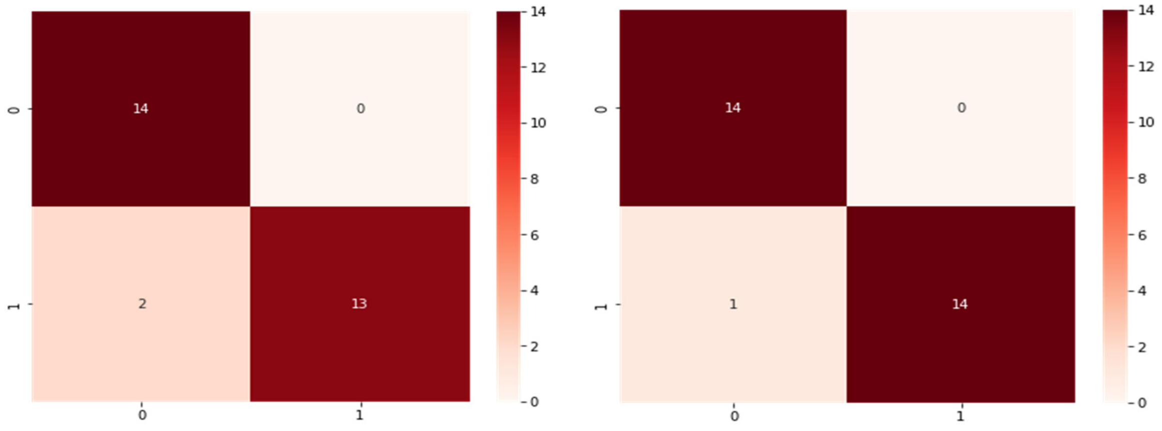


Figure 7. Plot of LRCN. (**a**) Accuracy. (**b**) Loss.

4.4. Confusion Matrix of the Proposed Models

The classification results may be summarized using a confusion matrix, as shown in Figure 8. Hand flapping belongs to class 0, while normal behavior belongs to class 1. The predictions of each class are included in a table, along with the number of right and wrong predictions. Class predictions are shown in the columns of the matrix, while actual classes are shown in rows in Figure 8. Overall, there are four options: True positives (TP): cases in which the classifier predicted "hand flapping" and the behavior was actually hand flapping. True negatives (TN): cases in which the classifier predicted "normal behavior" and the behavior was actually normal. False positives (FP): cases in which the classifier predicted "normal" but the behavior was actually hand flapping. False negatives (FN): cases in which the classifier predicted "hand flapping" but the behavior was actually normal.



(**a**) VGG-16-LSTM model (**b**) LRCN model

Figure 8. Plot of confusion matrix for the proposed models.

4.5. Real-Time Testing of the Model

The classification results of the models indicate high performance, supporting the adoption of this knowledge base to develop the test interface using computer vision. This system can monitor behavior directly via surveillance cameras or video recordings. This will help psychologists and parents diagnose abnormal children's behavior, which may be a sign of autism, such as hand flapping. Figure 9 shows the testing results of the Video-Based

Behavior Analysis for Autism Diagnostics. The snapshot on the left shows the detection of normal behavior, and the one on the right shows the snapshot detection of hand flapping.

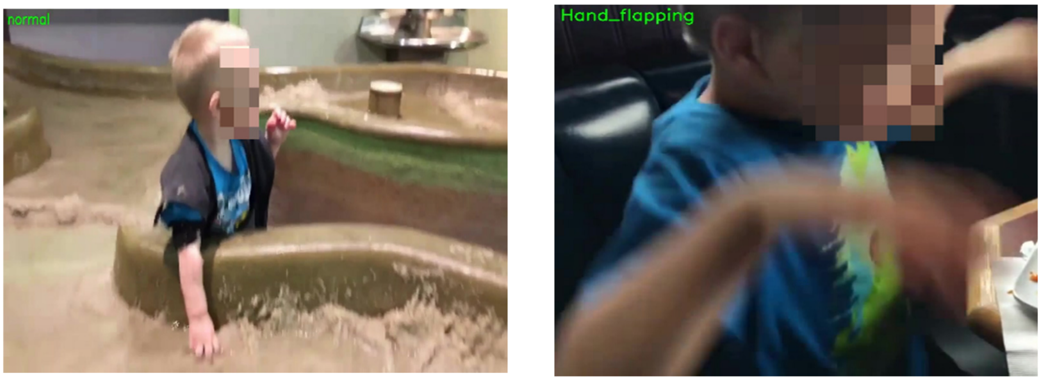


Figure 9. Real-time test results.

5. Discussion

Autism diagnosis is a complex task because it is based on human behavior, which requires long observation. This research paper presented a real-time system to detect the abnormal behavior of children with autism through videos. This system is based on deep learning and computer vision. The proposed models obtained higher accuracy compared to current methods. Our video classification models were trained on two classes of abnormal behavior (hand movement or hand flapping), as well as normal behavior. The videos were extracted from the SSBD [2].

This study has the advantage that it was based on algorithms such as VGG-16-LSTM and LRCNs, which have the ability to extract characteristics from video clips in real-time. In addition, the test video may be captured in a normal environment free of restrictions while the child is engaged in normal activities. The ASD child is known to hate the restrictions and protocols of similar tests. In addition, there is the possibility of viewing the result directly without statistical efforts or converting the video from one format to another, which is difficult for the psychologist to understand.

In this study, we conducted experiments utilizing two distinct deep learning models: the VGG-16-LSTM and LRCN. The outcomes of these experiments are presented in Table 4, showcasing the classification results of the models. Notably, the VGG-16-LSTM model exhibited an accuracy of 93% in effectively extracting both spatial and temporal features, accompanied by a sensitivity of 86%. Comparatively, the LRCN model emerged as the most proficient, achieving the highest accuracy of 96% in extracting spatial and temporal features with a sensitivity of 93%.

Table 4. The model results.

Model	Accuracy	Sensitivity	Specificity
VGG-16-LSTM	0.93	0.86	1.0
LRCN	0.96	0.93	1.0

Comparing the developed models to Lakkapragada et al. [4], the proposed models achieved higher F1 scores than existing models, which used MediaPipe to extract the hand landmarks and fed them into an LSTM and MobileNetV2. Their model achieved an F1 score of 84.0 ± 3.7. Lakkapragada et al. [4] used the same classes that we used in this study. Rajagopalan et al. [2,3] used the histogram of dominant motions with the histogram of optical flow. Their classification model achieved a binary accuracy of 86.6% when separating headbanging from spinning and an accuracy of 76.3% when identifying headbanging, spinning, and hand flapping in three classes. Ali et al. [5] examined ASD behavior diagnosis using a large collection of videos of stereotypical children's behavior. Their fusion of a two-stream I3D model (RGB and optical flow) achieved accuracies ranging from 85.6 to 86.04%. Table 5 shows the comparative analysis.

Table 5. Comparative analysis.

Model	Dataset	Method	Result
Rajagopalan et al. [3]	SSBD	Histogram of Dominant Motions	86.6%
Lakkapragada et al. [4]	SSBD	Hand Landmarks and MobileNetV2	F1 score of 84.0 ± 3.7
Ali et al. [5]	388 videos	Fusion of Two-Stream I3D	85.6 to 86.04%
Our study	SSBD	LRCN	**96%**

After reviewing and comparing our developed models with different existing models in recent studies, we found that the developed models were superior due to the proposed system based on the VGG-19-LSTM and LRCNs, which have the ability to extract spatial and temporal features automatically and effectively from videos. We applied the knowledge base of the models to computer vision to create a system to recognize the behavior in real-time. The proposed system can monitor behaviors using a surveillance camera at home or in psychiatric clinics to diagnose autism. However, despite the results we obtained, it should be noted that there are some limitations in this dataset. The videos were captured from different angles using cameras of varying quality, and there were differences in light intensity.

The motivation behind this study is to develop a system capable of detecting the early signs of autism in children. This would assist clinicians in selecting the appropriate behavioral therapy and support families in remote areas where access to advanced diagnostic devices is limited. Figure 10 shows final accuracy of propose model against different existing systems.

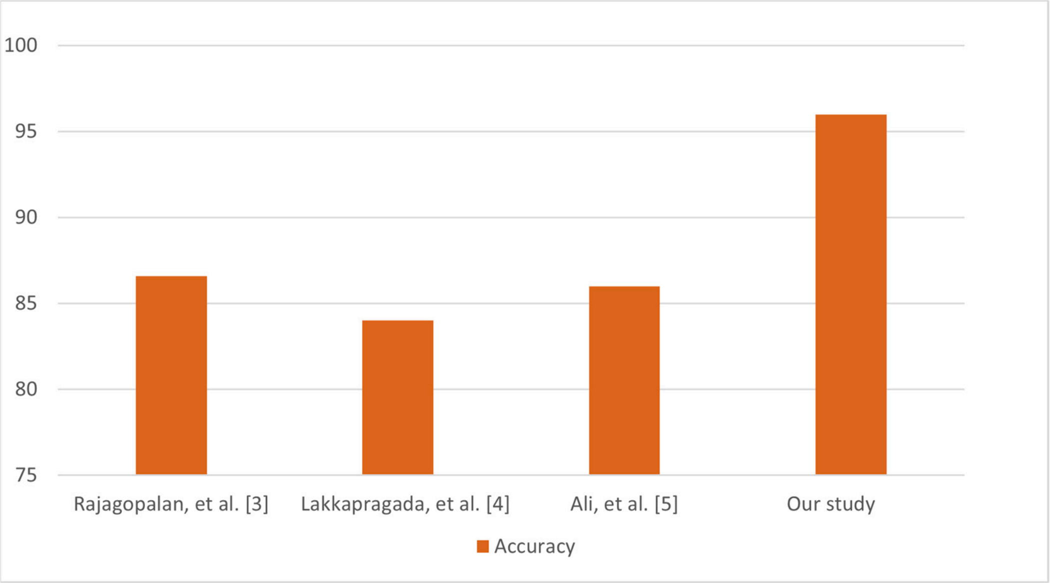


Figure 10. Comparison study between the proposed deep learning model and existing models.

6. Conclusions

In this research, deep learning with computer vision is used to analyze behavior from videos for the purpose of diagnosing autism. To create the real-time system, we trained our models on videos of kids going about their daily lives in an uncontrolled setting. The videos included both typical and deviant behavior, such hand flailing. On the testing set, the VGG-16-LSTM scored 93%. On training data, the LRCN model obtained 100% accuracy, whereas on validation data, it achieved 96% accuracy.

In order to improve the study's usefulness and practical applicability for psychologists and parents who are diagnosing aberrant behaviors, like hand flapping, which may be an indication of autism, we used computer vision techniques and the insights from our trained models to create a Video-Based Behavior Analysis for Autism Diagnostics. Through the use of these modern technical developments, our goal is to progress the field of autism identification, facilitating early intervention and subsequent therapies for kids.

Consequently, this could lead to enhanced social communication abilities in autistic children, thereby mitigating the difficulties encountered by their families. One of the system's most notable features is how little equipment it needs to function, which makes it suitable for implementation in a variety of settings, such as remote and underdeveloped places without access to sophisticated diagnostic techniques. The proposed deep learning models utilize and offer powerful capabilities for analyzing video data. The VGG-16 architecture provides learned feature representations that capture spatial relationships, while the LSTM layer models temporal dynamics critical for classifying motions and behaviors over time. Fine-tuning pretrained VGG-16 weights enabled robust feature extraction. Our results demonstrate that these models can effectively identify behavioral patterns from uncontrolled, real-world videos. This study's future scope includes the possibility of conducting high-quality video collations, which would improve the accuracy of the categorization results.

Author Contributions: Conceptualization, H.A., Z.A.T.A. and T.H.H.A.; methodology, H.A., Z.A.T.A., M.E.J. and A.A.A.; software, Z.A.T.A. and T.H.H.A.; validation, H.A., Z.A.T.A. and T.H.H.A.; formal analysis, H.A., Z.A.T.A., M.E.J. and A.A.A.; investigation, H.A., Z.A.T.A. and T.H.H.A.; resources, H.A., Z.A.T.A., M.E.J. and A.A.A.; data curation, Z.A.T.A. and T.H.H.A.; writing—original draft preparation, Z.A.T.A. and T.H.H.A.; writing—review and editing, H.A. and T.H.H.A.; visualization, H.A. and T.H.H.A.; supervision, H.A. and T.H.H.A.; project administration, H.A. and T.H.H.A.; funding acquisition, Z.A.T.A., T.H.H.A. and A.A.A. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to the King Salman center For Disability Research for funding this work through Research Group no KSRG-2023-097.

Data Availability Statement: https://rolandgoecke.net/research/datasets/ssbd/ (accessed on 12 July 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Volkmar, F.R.; Lord, C.; Bailey, A.; Schultz, R.T.; Klin, A. Autism and pervasive developmental disorders. *J. Child Psychol. Psychiatry* **2004**, *45*, 135–170. [CrossRef] [PubMed]
2. Rajagopalan, S.; Dhall, A.; Goecke, R. Self-stimulatory behaviours in the wild for autism diagnosis. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 2–8 December 2013; pp. 755–761.
3. Rajagopalan, S.S.; Goecke, R. Detecting self-stimulatory behaviours for autism diagnosis. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 1470–1474.
4. Lakkapragada, A.; Kline, A.; Mutlu, O.C.; Paskov, K.; Chrisman, B.; Stockham, N.; Washington, P.; Wall, D. Classification of Abnormal Hand Movement for Aiding in Autism Detection: Machine Learning Study. *arXiv* **2021**, arXiv:2108.07917.
5. Ali, A.; Negin, F.; Bremond, F.; Thümmler, S. Video-based Behavior Understanding of Children for Objective Diagnosis of Autism. In Proceedings of the VISAPP 2022-International Conference on Computer Vision Theory and Applications, Online, 6–8 February 2022.
6. Rehg, J.; Abowd, G.; Rozga, A.; Romero, M.; Clements, M.; Sclaroff, S.; Essa, I.; Ousley, O.Y.; Li, Y.; Kim, C.; et al. Decoding children's social behavior. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 3414–3421.
7. Zhao, Z.; Zhu, Z.; Zhang, X.; Tang, H.; Xing, J.; Hu, X.; Lu, J.; Qu, X. Identifying Autism with Head Movement Features by Implementing Machine Learning Algorithms. *J. Autism Dev. Disord.* **2022**, *52*, 3038–3049. [CrossRef] [PubMed]

8. Westeyn, T.; Vadas, K.; Bian, X.; Starner, T.; Abowd, G.D. Recognizing mimicked autistic self-stimulatory behaviors using HMMs. In Proceedings of the International Symposium on Wearable Computers, ISWC, Osaka, Japan, 18–21 October 2005; pp. 164–167. [CrossRef]
9. Ploetz, T.; Hammerla, N.Y.; Rozga, A.; Reavis, A.; Call, N.; Abowd, G.D. Automatic assessment of problem behavior in individuals with developmental disabilities. In Proceedings of the 14th ACM International Conference on Ubiquitous Computing (Ubicomp 2012), Pittsburgh, PA, USA, 5–8 September 2012; p. 2.
10. Sarker, H.; Tam, A.; Foreman, M.; Fay, N.; Dhuliawala, M.; Das, A. Detection of stereotypical motor movements in autism using a smartwatch-based system. In *AMIA Annual Symposium Proceedings*; American Medical Informatics Association: Bethesda, MD, USA, 2018; Volume 2018, p. 952.
11. Ahmed, Z.A.T.; Jadhav, M.E. A Review of Early Detection of Autism Based on Eye-Tracking and Sensing Technology. In Proceedings of the 2020 International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 26–28 February 2020; pp. 160–166.
12. Djemal, R.; AlSharabi, K.; Ibrahim, S.; Alsuwailem, A. EEG-based computer aided diagnosis of autism spectrum disorder using wavelet, entropy, and ANN. *BioMed Res. Int.* **2017**, *2017*, 9816591. [CrossRef] [PubMed]
13. Subah, F.Z.; Deb, K.; Dhar, P.K.; Koshiba, T. A deep learning approach to predict autism spectrum disorder using multisite resting-state fMRI. *Appl. Sci.* **2021**, *11*, 3636. [CrossRef]
14. Ahmed, Z.A.; Aldhyani, T.H.; Jadhav, M.E.; Alzahrani, M.Y.; Alzahrani, M.E.; Althobaiti, M.M.; Alassery, F.; Alshaflut, A.; Alzahrani, N.M.; Al-madani, A.M. Facial Features Detection System to Identify Children with Autism Spectrum Disorder: Deep Learning Models. *Comput. Math. Methods Med.* **2022**, *2022*, 3941049. [CrossRef] [PubMed]
15. Ahmed, Z.A.; Jadhav, M.E. Convolutional Neural Network for Prediction of Autism based on Eye-tracking Scanpaths. *Int. J. Psychosoc. Rehabil.* **2020**, *24*, 2683–2689. [CrossRef]
16. Available online: https://databricks.com/glossary/convolutional-layer (accessed on 22 May 2023).
17. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
18. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.K.; Woo, W.C. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1–9.
19. Abuqaddom, I.; Mahafzah, B.; Faris, H. Oriented Stochastic Loss Descent Algorithm to Train Very Deep Multi-Layer Neural Networks Without Vanishing Gradients. *Knowl.-Based Syst.* **2021**, *230*, 107391. [CrossRef]
20. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
21. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
22. Alsubari, S.N.; Deshmukh, S.N.; Al-Adhaileh, M.H.; Alsaade, F.W.; Aldhyani, T.H. Development of Integrated Neural Network Model for Identification of Fake Reviews in E-Commerce Using Multidomain Datasets. *Appl. Bionics Biomech.* **2021**, *2021*, 5522574. [CrossRef] [PubMed]
23. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2625–2634.
24. Huang, X.; Chan, K.-H.; Ke, W.; Sheng, H. Parallel Dense Video Caption Generation with Multi-Modal Features. *Mathematics* **2023**, *11*, 3685. [CrossRef]
25. Wang, Y.; He, Z.; Wang, L. Truck Driver Fatigue Detection Based on Video Sequences in Open-Pit Mines. *Mathematics* **2021**, *9*, 2908. [CrossRef]
26. Reddy, B.R.; Kumar, R.L. Classification of health care products using hybrid CNN-LSTM model. *Soft Comput.* **2023**, *27*, 9199–9216. [CrossRef]
27. Chollet, F. Keras. GitHub Repository. 2015. Available online: https://github.com/fchollet/keras (accessed on 2 May 2021).
28. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. {TensorFlow}: A system for {Large-Scale} machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
29. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.