

CNN-LSTM Autoencoder for Behavioral Anomaly Detection in Children with ASD

Hasri Akbar Awal Rozaq¹, Dzihni Safwa Alifah^{*2}

¹Graduate School of Informatics, Department of Computer Science, Gazi University, Turkey

² Department of Information Technology, Faculty of Computer Science, Universitas Amikom Purwokerto, Indonesia

e-mail: hakbar.rozaq@gazi.edu.tr¹, dzihnisafwaa@gmail.com²

Article History	Abstract
Received : 18 June 2026 Revised : 27 August 2026 Accepted : 9 September 2026 Published : 18 September 2026 Keywords: <i>Anomaly Detection, Asd, Cnn-Lstm Autoencoder.</i>	Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder characterized by differences in social communication and repetitive behavioral patterns. Behavioral analysis based on video is commonly performed through manual observation, which can be time-consuming and subjective. This study proposes a CNN-LSTM Autoencoder approach to detect behavioral anomalies in children with ASD based on video data. The CNN Autoencoder extracts spatial features from video frames, while the LSTM Autoencoder learns temporal patterns from sequential features. The model was trained using 88 videos collected from children with ASD, while five videos from a public dataset were used for testing. The experimental results produced mean reconstruction errors of 0.1000, 0.1184, and 0.2079 for the training, validation, and testing data, respectively, with a threshold of 0.2520. Among 65 testing sequences, 51 were classified as in-distribution with respect to the ASD training data, while 14 were identified as anomalies. These results indicate that the proposed approach can identify behavioral deviations from the learned ASD behavioral distribution. However, the system is not intended to provide a clinical diagnosis of ASD.

^{*}Corresponding author : dzihnisafwaa@gmail.com

1. INTRODUCTION

Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder that can affect communication, social interaction, and behavioral patterns from early childhood [1]. Children with ASD may exhibit characteristic behaviors, such as repetitive movements, atypical social responses, limited eye contact, and distinctive activity patterns. Early detection is important to enable children to receive appropriate support and intervention at an early stage [2]. However, ASD identification still relies heavily on clinical observation by professionals, such as psychologists and medical specialists, which can require considerable time and cost, while access to specialized healthcare services remains uneven [2]. These challenges highlight the need for supportive approaches that can facilitate behavioral analysis in a faster, more objective, and structured manner through computational technologies [3].

The development of artificial intelligence, particularly deep learning, provides opportunities to analyze children's behavioral patterns using video data. Video can capture visual and temporal information related to movements, responses, and behavioral changes associated with ASD characteristics [4][5]. Previous research has demonstrated that video-based approaches can support ASD classification by analyzing children's responses to specific stimuli [5]. In addition, repetitive behaviors such as arm flipping, head banging, and spinning exhibit visual and temporal patterns that can be analyzed through sequences of video frames [6]. Therefore, video represents a potential source of information for systematically representing ASD-related behavioral patterns, as behavioral sequences can provide information beyond that available from individual frames.

In computer vision, Convolutional Neural Networks (CNNs) are capable of extracting visual features from images or video frames, while Long Short-Term Memory (LSTM) networks can learn temporal dependencies between frames in sequential data [7][8]. The combination of CNN and LSTM enables a model to capture both visual characteristics and temporal changes in behavioral patterns. Previous studies have also demonstrated that spatio-temporal features extracted from video can be used to analyze behavioral characteristics and the severity of ASD-related behaviors [9]. However, existing studies have primarily focused on classification or recognition tasks involving multiple behavioral or diagnostic categories. When training data consist only of children with ASD, conventional classification approaches may require separate non-ASD data as a reference class, which may not always be available. This limitation highlights the need for an approach capable of learning the distribution of ASD behavioral patterns without requiring a separate non-ASD class during training.

Based on this limitation, this study formulates the problem as a one-class anomaly detection task, in which the model learns the distribution of behavioral patterns from ASD data during training. A CNN-LSTM Autoencoder is employed to process videos as sequences of frames, extract visual features using CNN, and learn temporal patterns using LSTM. The model subsequently generates a reconstruction error to measure the deviation of test data from the learned ASD behavioral distribution. Test sequences with a reconstruction error below or equal to a predefined threshold are considered consistent with the learned distribution (in-distribution), whereas sequences exceeding the threshold are identified as anomalies. Therefore, this study aims to detect deviations in video-based behavioral patterns from the learned ASD data distribution. The resulting anomaly detection is not intended to provide a clinical diagnosis or determine non-ASD status, but rather to serve as a supporting approach for analyzing the consistency of behavioral patterns in video data with the learned ASD distribution.

2. METHOD

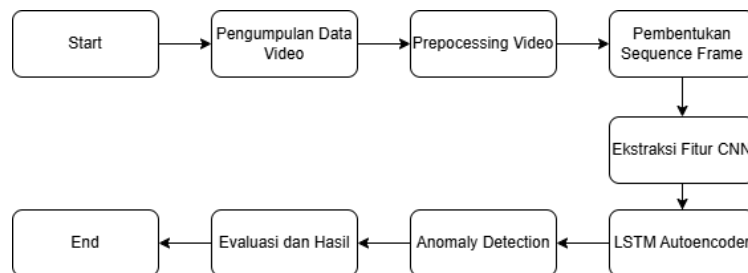


Figure 1. Research Method

2.1. Video Data Collection

The dataset used in this study consisted of behavioral videos of children with Autism Spectrum Disorder (ASD) obtained from special needs schools in the Purbalingga and Banjarnegara regions, covering elementary to high school levels, together with videos from a public dataset. A total of 88 locally collected videos were initially prepared as training data, while five videos from the public dataset were used as testing data. The local ASD videos were used to learn the behavioral distribution of the training data, consistent with the one-class anomaly detection approach employed in this study.

During sequence formation, not all of the 88 training videos could be used because some videos did not contain a sufficient number of frames to form the required 30-frame sequences. After this filtering process, 60 videos were used for training and 16 videos were used for validation. The remaining videos that did not meet the minimum frame requirement were excluded from sequence formation. The final sequence dataset consisted of 313 training sequences, 97 validation sequences, and 65 testing sequences.

Because the dataset contains videos of individual children, privacy and confidentiality were considered throughout the data processing stage. The identities of the children were not

used as variables in the modeling process, and the research output was not intended to provide a clinical diagnosis.

The dataset is stored in the "Autism" folder and divided into training and testing sets. The training data is used to learn ASD behavioral patterns—and is further split into training and validation subsets—while the testing data is used to evaluate the degree of similarity between the videos and the ASD patterns.

The dataset used in this study consisted of behavioral videos of children with Autism Spectrum Disorder (ASD) obtained from special needs schools in the Purbalingga and Banjarnegara regions, covering elementary to high school levels, together with videos from a public dataset. A total of 88 locally collected videos were initially prepared as training data, while five videos from the public dataset were used as testing data. The local ASD videos were used to learn the behavioral distribution of the training data, consistent with the one-class anomaly detection approach employed in this study.

During sequence formation, not all of the 88 training videos could be used because some videos did not contain a sufficient number of frames to form the required 30-frame sequences. After this filtering process, 60 videos were used for training and 16 videos were used for validation. The remaining videos that did not meet the minimum frame requirement were excluded from sequence formation. The final sequence dataset consisted of 313 training sequences, 97 validation sequences, and 65 testing sequences.

Because this study adopts a one-class anomaly detection setting, the model was trained exclusively on ASD behavioral data and did not use a separate non-ASD class during training. The testing data were used to examine whether behavioral sequences were consistent with or deviated from the ASD behavioral distribution learned during training. Therefore, an anomaly detected by the proposed model represents a deviation from the learned ASD behavioral distribution rather than a direct indication of non-ASD status.

The relatively small number of testing videos and the absence of a non-ASD control group constitute important limitations of this study. Consequently, the current evaluation does not establish the clinical diagnostic capability of the proposed model or its ability to distinguish ASD from non-ASD populations. The findings should instead be interpreted as an initial evaluation of behavioral conformity to the ASD distribution learned from the available training data. Future studies should evaluate the proposed approach using a larger and more diverse dataset, including additional subjects and, where available, non-ASD control data.

2.2. Video Preprocessing

Video preprocessing is the stage carried out to prepare data before it is processed by the model. In this stage, the video is converted into frames so that it can be read by the CNN model. The steps include:

1. Frame Extraction

Videos from the collected dataset are extracted into sets of frames. This process is necessary because CNNs operate on images or frames. Each video is broken down into multiple frames at specific intervals using the 'Frame_Step' parameter.

2. Frame Resizing

Once extracted, the frames are resized to 128 x 128 pixels. Resizing ensures uniform frame dimensions, facilitating easier processing by the model.

3. Pixel Normalization

Pixel values are subsequently normalized from the 0–255 range to the 0–1 range. This normalization places the input on a more stable scale and aids the model's learning process.

2.3. Formation of sequence frames

A total of 88 training videos and 5 testing videos were used. Following the preprocessing stage, not all videos could be utilized, as some lacked a sufficient number of frames to form a 30-frame sequence. Consequently, the dataset comprised 60 training videos and 16 validation videos, while the testing set remained at 5 videos.

Subsequently, each video was converted into a sequence consisting of 30 frames. This process yielded 313 training sequences, 97 validation sequences, and 65 testing sequences. Based on the number and length of the sequences, the total number of available frames is 9,390 for training, 2,910 for validation, and 1,950 for testing. All frames have a resolution of 128 x 128 pixels with three RGB color channels.

Table 1. Sequence Table

Data Type	Number of Sequences	Total Frames
Training Data	313	9390
Validation Data	97	2910
Testing Data	65	1950

2.4. CNN Feature Extraction

In this study, feature extraction is performed using a CNN Autoencoder. This method was selected because it is capable of learning essential information from the data through a process of compression and reconstruction, thereby producing a more compact feature representation that nonetheless retains the input data's key characteristics[11]. Unlike a standard CNN initialized with random weights, a CNN Autoencoder is pre-trained to understand patterns within the input data, resulting in features that are more informative and better aligned with the specific characteristics of the data used[12].

The CNN autoencoder architecture consists of two components: an encoder and a decoder. The encoder extracts essential information from a 128 x 128 x 3 input frame using convolutional layers, ReLU activation, max pooling, flattening, and dense layers [13]. The output is a 128-dimensional feature vector representing the frame's key information. Meanwhile, the decoder reconstructs the feature vector into an image that closely resembles the original, enabling the encoder to learn how to generate optimal feature representations.

Once the training process is complete, the encoder component can serve as a feature extractor to generate a more compact and informative feature representation of the input data[14]. Each video frame is processed by the encoder to produce a 128-dimensional feature vector. Subsequently, the feature vectors from each frame are arranged according to their temporal order to form a feature sequence, which serves as input for the LSTM Autoencoder to learn temporal patterns within the video.

2.5. LSTM Autoencoder

LSTM Autoencoders are widely used for sequential data because they are capable of learning long-term temporal dependencies and reconstructing learned data patterns to facilitate anomaly detection[15]. This model consists of two components: an encoder and a decoder. The encoder's task is to learn a pattern representation from the input sequence, while the decoder is responsible for reconstructing that sequence.

Since the model utilizes ASD data, it learns the patterns present within that dataset. If a test video exhibits patterns similar to those found in the ASD training videos, the model will yield a low reconstruction error. Conversely, if the test video's patterns differ from those learned by the model, the reconstruction error will be higher.

2.6. Anomaly Detection

To ensure reproducibility, the architecture and training configuration of the proposed CNN-LSTM Autoencoder are described in detail. The CNN Autoencoder receives video frames with an input size of $128 \times 128 \times 3$ and transforms each frame into a 128-dimensional feature representation. The encoder consists of convolutional, activation, pooling, and dense layers, while the decoder reconstructs the input frame from the learned representation. After the CNN Autoencoder training process, only the encoder is retained and used as the feature extractor for the subsequent temporal modeling stage.

The extracted 128-dimensional frame-level features are arranged according to their temporal order to form sequences of 30 frames. These feature sequences are then processed by the LSTM Autoencoder. The LSTM encoder learns temporal dependencies within each sequence, while the decoder reconstructs the input feature sequence. The reconstruction error between the original and reconstructed sequences is subsequently used as the basis for anomaly detection.

At this stage, the LSTM Autoencoder model calculates the reconstruction error value based on the difference between the original feature sequence and the model's reconstruction output. This value is used to measure how well the data conforms to the learned patterns. Data with a reconstruction error exceeding a specific threshold is categorized as an anomaly[16]. The reconstruction error is calculated using the following equation.

$$RE = \frac{1}{T} \sum_{t=1}^T (x_t - \hat{x}_t)^2$$

The threshold value is determined based on the distribution of reconstruction errors in the training data, utilizing the mean and standard deviation. This approach is used to establish the maximum limit of data variation considered consistent with the patterns learned by the model. Consequently, data with a reconstruction error exceeding the threshold is categorized as an anomaly[17]. The threshold is calculated using the following equation.

$$T = \mu_{\{RE_{\{train\}}\}} + 2\sigma_{\{RE_{\{train\}}\}}$$

The reconstruction error value of the test data is then compared against the established threshold. If the reconstruction error is less than or equal to the threshold, the data is categorized as in-distribution—meaning it aligns with the ASD patterns learned by the model. Conversely, if the reconstruction error exceeds the threshold, the data is categorized as an anomaly.

2.7. Evaluation and Results

Evaluation was conducted to measure the model's ability to analyze the conformity of behavioral patterns in test videos against the distribution of ASD training data. In this study, anomaly detection decisions were determined by comparing the reconstruction error value with a threshold. Data with a reconstruction error below the threshold were categorized as in-distribution, whereas data exceeding the threshold were categorized as anomalies. In addition to reconstruction error, this study calculated a similarity score as supplementary information to illustrate the degree of resemblance between the test data and the ASD behavioral patterns learned by the model. The similarity score is a metric used to indicate the level of closeness or resemblance between two data representations, where a higher value indicates a greater degree of similarity[18]. In this study, the similarity score was calculated based on the comparison between the reconstruction error and the threshold.

$$Similarity\ Score = \frac{100}{1 + \left(\frac{RE}{Threshold}\right)}$$

3. RESULTS

3.1. Results of Data Collection

Following the data collection process, a total of 88 videos were obtained as training data and 5 videos were used as testing data. In this study, the dataset was not divided using a specific ratio-based scheme. Instead, the training and testing data were obtained from different sources in accordance with the objective of implementing a one-class anomaly detection approach. The training data consisted of videos of children with ASD collected from special needs schools and were used as reference data for learning the distribution of ASD behavioral patterns. Meanwhile, the testing data were obtained from a public dataset and were used to evaluate whether the behavioral patterns in the test videos remained consistent with the ASD distribution learned by the model or were identified as anomalies.

Table 2. Number of Video Samples

Data Type	Number of Videos
Training Data	88
Testing Data	5

3.2. Preprocessing and Sequence Formation Results

At this stage, each video was converted into individual frames, resized to 128×128 pixels, and normalized by scaling pixel values from the range of 0–255 to 0–1. The processed frames were subsequently arranged into sequences, where each sequence consisted of 30 consecutive frames. This approach enabled the model to capture temporal information contained within the video data.

The division of training and validation data was performed at the video level rather than at the sequence level. This strategy was adopted to prevent sequences originating from the same video from being distributed across both the training and validation sets. The resulting sequence distribution is presented in Table 3.

Table 3. Number of Sequences

Data Type	Number of Sequences
Training Data	313
Validation Data	97
Testing Data	65

3.3. CNN-LSTM Autoencoder Modeling Results

A CNN was employed to extract visual features from each video frame. To obtain more representative features, the CNN was first trained using a CNN Autoencoder architecture. The CNN Autoencoder was trained using 1,500 training frames and 400 validation frames extracted from ASD videos. After the training process, the encoder component was utilized as a feature extractor to generate a 128-dimensional feature vector for each frame. These feature vectors were subsequently organized into feature sequences.

Table 4. Feature Sequence Dimensions

Data Type	Feature Sequence Shape
Training Data	$313 \times 30 \times 128$
Validation Data	$97 \times 30 \times 128$
Testing Data	$65 \times 30 \times 128$

The generated feature sequences were then used as input for the LSTM Autoencoder. The model was trained to reconstruct feature sequences derived from ASD training data in order to learn the temporal patterns embedded within the data. The training process was conducted for 30 epochs using both training and validation sequences. The training performance of the model is illustrated in Figure 2.

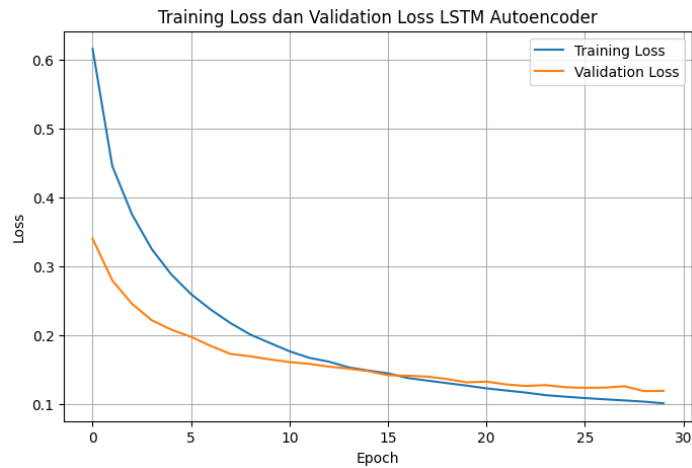


Figure 2. Training and Validation Loss of the LSTM Autoencoder

Based on Figure 2, both training loss and validation loss exhibited a consistent downward trend throughout the training process. Initially, both loss values were relatively high and gradually decreased until reaching a stable state in the final epochs. These results indicate that the model successfully learned the temporal patterns present in the ASD training data. Furthermore, the relatively small gap between the training and validation loss curves suggests that the model achieved satisfactory generalization performance without exhibiting significant overfitting.

3.4. Anomaly Detection Results

The anomaly detection stage was performed by calculating the reconstruction error, which represents the difference between the original feature sequence and the reconstructed sequence generated by the model. A lower reconstruction error indicates that the data closely matches the distribution of the training data, whereas a higher reconstruction error suggests a greater deviation from the learned distribution.

Table 5. Mean Reconstruction Error

Data Type	Mean Reconstruction Error
Training Data	0.10004011
Validation Data	0.11835161
Testing Data	0.20794532

The threshold value used in this study was 0.25203454, which was calculated using the mean reconstruction error of the training data plus two times its standard deviation. Sequences with reconstruction error values below or equal to the threshold were categorized as in-distribution with respect to the ASD training data. Conversely, sequences with reconstruction error values exceeding the threshold were classified as anomalies. In addition to reconstruction error, the system also calculated a similarity score as supplementary information.

Table 6. Mean Similarity Score

Data Type	Mean Similarity Score
Training Data	74.38739%
Validation Data	70.16097%
Testing Data	56.24792%

3.5. Evaluation Results

Based on the evaluation of 65 testing sequences, 51 sequences were classified as in-distribution with respect to the ASD training data, while 14 sequences were identified as anomalies.

Table 7. Sequence Classification Results

Decision	Number of Sequences
In-distribution with respect to ASD training data	51
Anomaly with respect to ASD training data	14
Total	65

Further evaluation was conducted at the video level by calculating the average reconstruction error and similarity score for each testing video. The results are presented in Table 8.

Table 8. Video-Level Evaluation Results

Video	Total Sequences	Mean Error	Similarity Score	Decision
Kimberly Bentura	7	0.212573	56.586323%	In-distribution
spearskelly	6	0.140697	64.412415%	In-distribution
tifabebe	39	0.221947	54.140911%	In-distribution
Jessica PS	3	0.073840	77.536949%	In-distribution
katelynallyson	10	0.230692	52.942974%	In-distribution

Based on Table 8, all testing videos produced mean reconstruction error values below the established threshold and were therefore classified as in-distribution with respect to the ASD training data at the video level. However, 14 individual sequences were still identified as anomalies. This finding indicates that although the overall videos remain consistent with the ASD training data distribution, certain segments contain behavioral patterns that differ from those learned during training.

The reconstruction error distribution further demonstrates that the training data generally yielded lower error values than the testing data. This outcome is expected because the model was trained using ASD training data. The testing data exhibited higher reconstruction errors due to behavioral variations that differed from those observed in the training set. Nevertheless, most testing sequences remained below the threshold value.

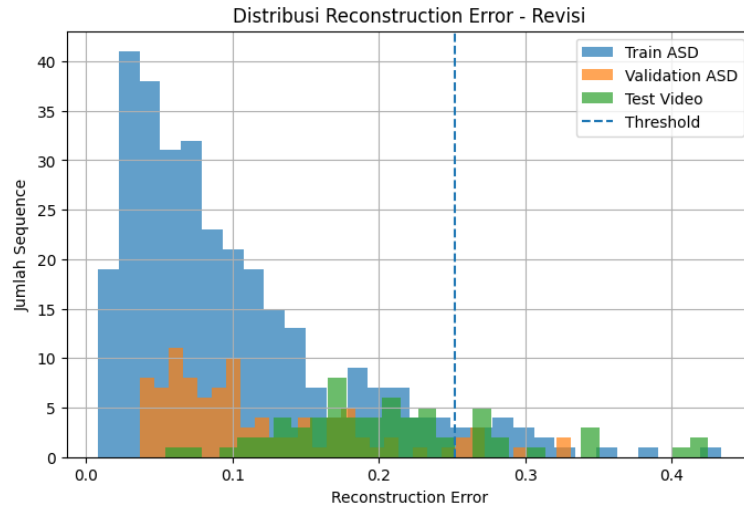


Figure 3. Reconstruction Error Distribution

Based on Figure 3, the reconstruction error distribution of the training data is concentrated at lower values compared to that of the testing data. This indicates that the model was able to reconstruct training samples more effectively because these patterns were learned during training. The validation data exhibited a distribution similar to that of the training data, suggesting that the model retained the ability to generalize to unseen ASD samples. In contrast, the testing data showed a wider distribution and higher reconstruction errors, indicating the presence of behavioral

variations not fully represented in the training data. Despite this variation, most testing sequences remained below the threshold and were therefore classified as in-distribution.

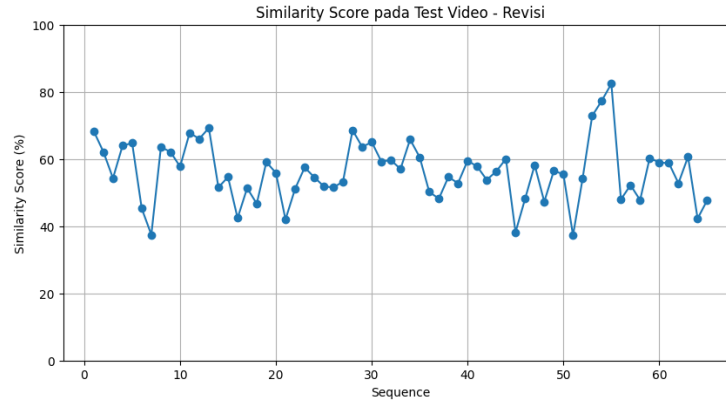


Figure 4. Similarity Score of Testing Videos

Based on Figure 4, the similarity scores obtained from the testing videos were moderate, with an average value of 56.24792%. This result indicates that the testing data still shared behavioral characteristics with the ASD patterns learned by the model, although the degree of similarity was lower than that observed in the training and validation datasets. Therefore, the similarity score was used as supporting information to facilitate result interpretation, while the final decision remained based on the comparison between reconstruction error and the predefined threshold.

The findings of this study should not be interpreted as a clinical diagnosis of ASD or non-ASD. An in-distribution classification indicates that the behavioral patterns contained in a test video remain consistent with the distribution of the ASD training data, whereas an anomaly classification indicates the presence of behavioral patterns that differ from the learned distribution.

4. DISCUSSION

The findings of this study demonstrate that the proposed CNN-LSTM Autoencoder was able to effectively learn behavioral patterns from ASD video data. This is evidenced by the consistent decrease in both training loss and validation loss throughout the training process until convergence was achieved. Such behavior indicates that the model successfully captured the temporal characteristics present in the ASD training data without exhibiting significant overfitting. Furthermore, the use of a CNN Autoencoder as a feature extractor enabled the generation of compact feature representations while preserving essential information from each frame. These feature representations were subsequently utilized by the LSTM Autoencoder to learn temporal dependencies across video sequences[19].

The experimental results also revealed that the reconstruction errors obtained from the training and validation datasets were generally lower than those of the testing dataset. This finding suggests that the model was able to reconstruct samples with characteristics similar to those encountered during training more accurately. Although the testing data produced higher reconstruction errors, most sequences still remained below the predefined threshold. Therefore, the majority of testing samples were considered consistent with the ASD behavioral pattern distribution learned by the model.

The primary objective of this study was to analyze the conformity of behavioral patterns in test videos with respect to the ASD training data distribution using an anomaly detection approach. Based on the evaluation results, 51 out of 65 testing sequences were classified as in-distribution, while 14 sequences were identified as anomalies. At the video level, all testing videos were categorized as in-distribution because their average reconstruction errors remained below the established threshold. These findings indicate that the proposed CNN-LSTM Autoencoder was

capable of distinguishing behavioral sequences that were consistent with the learned ASD patterns from those that exhibited deviations from the training distribution.

Despite these promising results, several limitations remain. The number of videos used in this study was relatively limited, and the training dataset consisted exclusively of children with ASD, which may not fully represent the diversity of behavioral patterns observed in broader populations. In addition, the system's decisions were based solely on reconstruction error values, making it unable to explicitly identify the specific behaviors responsible for anomalous detections. Future studies may benefit from incorporating larger and more diverse datasets, employing pose estimation techniques such as MediaPipe to obtain more detailed motion representations, and developing behavioral classification mechanisms that not only detect anomalies but also identify the types of behaviors present in video sequences.

5. CONCLUSION

This study proposed a CNN-LSTM Autoencoder framework for analyzing the conformity of behavioral patterns in children with Autism Spectrum Disorder (ASD) based on video data. The framework consisted of video preprocessing, sequence generation, feature extraction using a CNN Autoencoder, and temporal pattern learning using an LSTM Autoencoder. Experimental results showed that the training, validation, and testing datasets achieved mean reconstruction error values of 0.10004011, 0.11835161, and 0.20794532, respectively, with a threshold value of 0.25203454. Evaluation on 65 testing sequences indicated that 51 sequences were classified as in-distribution with respect to the ASD training data, while 14 sequences were identified as anomalies. Furthermore, all testing videos were categorized as in-distribution at the video level because their average reconstruction errors remained below the predefined threshold. These findings demonstrate that the proposed CNN-LSTM Autoencoder was capable of learning spatio-temporal behavioral patterns from ASD video data and identifying deviations based on reconstruction error analysis. Although the system is not intended for clinical diagnosis, it shows the potential of anomaly detection approaches as a supporting tool for analyzing the conformity of behavioral patterns with ASD training data distributions.

BIBLIOGRAPHY

- [1] H. Hodges, C. Fealko, and N. Soares, "Autism spectrum disorder : definition , epidemiology , causes , and clinical evaluation," vol. 9, no. 8, 2020, doi: 10.21037/tp.2019.09.09.
- [2] C. Okoye *et al.*, "Early Diagnosis of Autism Spectrum Disorder : A Review and Analysis of the Risks and Benefits," vol. 15, no. 8, 2023, doi: 10.7759/cureus.43226.
- [3] A. M. Penney, J. Greenon, I. S. Schwartz, and A. M. Estes, "' On-Time Autism Intervention ' : A Diagnostic Practice Framework to Accelerate Access," vol. 13, no. February, pp. 1–7, 2022, doi: 10.3389/fpsy.2022.784580.
- [4] M. Serna-aguilera, "A Novel Dataset for Video-Based Neurodivergent Classification Leveraging Extra-Stimulatory Behavior," 2024.
- [5] H. Wang, J. Guo, X. Ma, S. Fu, Q. Yang, and Y. Xu, "Online Self-Evolving Anomaly Detection in Cloud Computing Environments," pp. 1–10, 2021.
- [6] A. K. Singh and V. Singh, "Advanced Gesture Recognition for Autism Spectrum Disorder Detection : Integrating YOLOv7 , Video Augmentation , and VideoMAE for Naturalistic Video Analysis," 2024.
- [7] L. Alzubaidi *et al.*, *Review of deep learning : concepts , CNN architectures , challenges , applications , future directions*. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.
- [8] S. Mathew, A. Subramanian, Pooja, B. Ms, and manoj kumar Rajagopal, "HUMAN ACTIVITY RECOGNITION USING DEEP LEARNING APPROACHES: SINGLE FRAME CNN AND CONVOLUTIONAL LSTM Sheryl Mathew, Annapoorani Subramanian, S. Pooja, Balamurugan MS School of Electronics Engineering, Vellore Institute of Technology, Chennai,

-
- India,” 2023.
- [9] A. Ali *et al.*, “Weakly-supervised Autism Severity Assessment in Long Videos,” 2024.
- [10] M. Kim, J. Kim, J. Yu, and J. Kyun, “Active Anomaly Detection based on Deep One-class Classification,” pp. 1–8, 2023.
- [11] Y. Lee, C. Park, N. Kim, and J. Ahn, “LSTM-Autoencoder Based Anomaly Detection Using Vibration Data of Wind Turbines,” pp. 1–24, 2024.
- [12] K. Berahmand, F. Daneshfar, E. Sadat, and S. Yuefeng, *Autoencoders and their applications in machine learning : a survey*. Springer Netherlands, 2024. doi: 10.1007/s10462-023-10662-6.
- [13] M. M. Taye, “Theoretical Understanding of Convolutional Neural Network :,” 2023.
- [14] L. G. Apr, “Dor Bank, Noam Koenigstein, Raja Giryes,” pp. 1–22, 2021.
- [15] Y. Wei, J. Jang-jaccard, W. Xu, F. Sabrina, S. Camtepe, and M. Boulic, “LSTM-Autoencoder based Anomaly Detection for Indoor Air Quality Time Series Data,” 2022.
- [16] G. Pang, C. Shen, L. Cao, and A. V. A. N. D. E. N. Hengel, “Deep Learning for Anomaly Detection : A Review,” vol. 54, no. 2, 2021.
- [17] I. Zidaoui, C. Wemmert, C. Joannis, S. Isel, J. Wertel, and J. Vazquez, “Application of ResNet and Autoencoder models for anomaly detection in wastewater networks data : A Comparative study of supervised and unsupervised approaches,” *Flow Meas. Instrum.*, vol. 105, no. May, p. 102946, 2025, doi: 10.1016/j.flowmeasinst.2025.102946.
- [18] L. Jackowska-strumillo, “Hybrid ANN-Based and Text Similarity Method for Automatic Short-Answer Grading in Polish,” 2025.
- [19] S. Chen, “Auto-Encoders in Deep Learning — A Review with New Perspectives,” pp. 1–54, 2023.