Sign-Language Recognition using CNN & RNN

Pampa Sow Mondal
210050113
210050113@cse.iitb.ac.in

Sayantika Mandal
22d0379
22d0379@iitb.ac.in

Pratiksha Deka
210050122
210050122@iitb.ac.in

Shubham Roy
22d0378
royshubham@cse.iitb.ac.in

***Abstract*—Our project endeavors to bridge communication barriers between sign language users and the wider community by dissecting video sequences into frames, rich in both spatial and temporal features. We explore two distinct approaches for Indian sign language recognition from videos: a 3D-Convolutional Neural Network (CNN) approach using 18 layer Resnet3D (R3D18) model and a combined CNN-RNN model (ResNet50 + LSTM layers). Results highlight the promising performance of the R3D18 model, showcasing its efficacy in capturing spatio-temporal features. Additionally, the ResNet50 + LSTM model exhibits potential for further improvement with extended training, suggesting the opportunity for refining its accuracy and robustness. These findings contribute to advancing the field of sign language recognition.**

1. Introduction

Sign language is a complex and nuanced mode of communication, employing visual gestures, hand shapes, and facial expressions to convey meaning and facilitate interaction among the deaf and hard-of-hearing communities. Sign languages, contrary to common misconceptions, are not universal; rather, each region and community often has its distinct sign language, with unique grammar, syntax, and vocabulary, e.g., Indian Sign language (ISL), American Sign Language (ASL), and Portuguese Sign Language. There are three types of sign languages: spelling each alphabet using fingers, sign vocabulary for words, using hands and body movement, facial expressions, and lip movement.

Sign-Language recognition holds incredible potential to narrow the communication barrier between the deaf community and those who don't use sign language. The first solution to the problem is using 3D-Convolutional Neural Network (CNN). 3D-CNN, a model designed for video analysis, incorporates 3D convolutions. It divides the 3D convolution into two sequential steps: a 2D spatial convolution followed by a 1D temporal convolution. The "(2+1)D" convolutional blocks in 3D CNN introduce an extra non-linear step between spatial and temporal convolutions, doubling the model's pattern capturing ability without adding parameters. This optimized approach improves the model's capacity to represent intricate functions, aiding optimization and resulting in lower training and testing losses. This modification significantly enhances the 3D CNN's capability to efficiently capture both spatial and temporal features, optimizing its learning process for better sign language recognition. The second solution to the problem is using CNN and Long short-term memory (LSTM) together. LSTMs excel

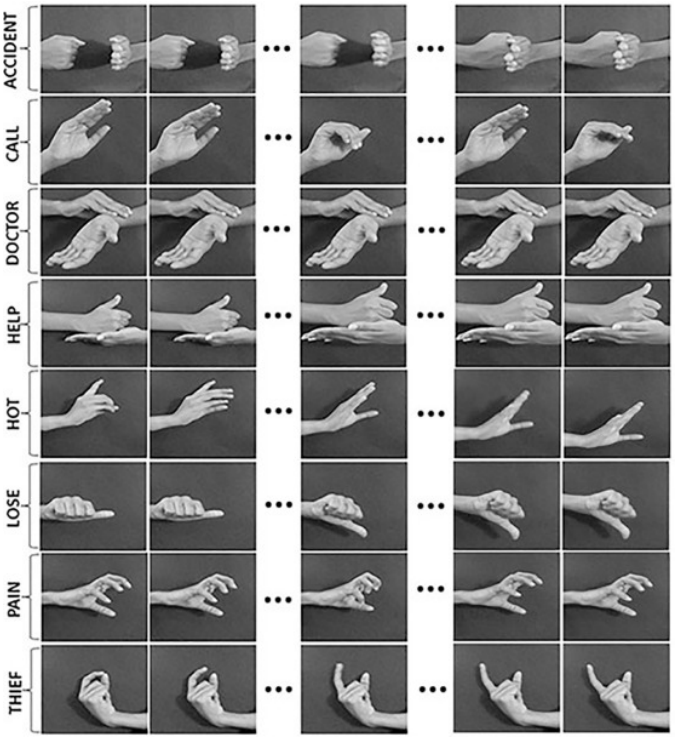

Figure 1. ISL used in emergency situations (depicted as sequence of frames

in modeling sequences and time series data. They possess memory cells to retain information over long sequences. We employed the 18 layer ResNet3D (R3D18) [1] for the 3D CNN approach and ResNet-50 [2] for the CNN architectures when working with the Indian Sign Language (ISL) dataset [3].

2. Related Work

Real-time sign language gesture recognition studies have primarily utilized CNN and Recurrent Neural Networks (RNN) for interpreting gestures in languages like Argentinean Sign Language (LSA). The CNN (Inception-v3 model) effectively captures spatial features, while LSTM, a type of RNN, handles temporal aspects. In their work, Masood et al. [4] focused on Real-Time Sign Language Gesture (Word) Recognition using CNN and RNN, achieving remarkable accuracy. Adithya et al. [3] achieved 90% accuracy using a feature-driven method with a multiclass SVM and 96.25% accuracy employing a data-driven approach using a pre-trained CNN (GoogleNet) with LSTM for ISL gesture recognition.

3. Dataset and prepocessing

All the dataset and preprocessing is heavily inspired by [3]. The dataset comprises RGB videos of hand gestures for eight ISL (Indian Sign Language) words: 'accident', 'call', 'doctor', 'help', 'hot', 'lose', 'pain', and 'thief'. These gestures, except for 'doctor', are dynamic. Videos were captured from 26 adults (12 males, 14 females) aged 22 to 26 years, representing diverse skin colors across India.

Data Collection: Participants' hand gestures were recorded against a black background with normal indoor lighting to emphasize motion features over skin color variations. This setup reduces interference from skin-color-related issues and clothing, aiding gesture recognition.

Dataset Structure: It contains two folders:

- **Raw_Data:** Original videos sized at 1280x720 pixels.
- **Cropped_Data:** Videos cropped to remove excessive background and resized to a uniform 500x600 pixels.
- **File Naming Convention:** Files are labeled as per the ISL word, participant identifier (001 to 026), and sample number (01 or 02). For instance, 'accident_003_02' denotes the second sample of the word 'accident' by the 3rd participant.

Dataset Contents: It includes a total of 421 '.avi' format videos for the 8 gestures, each category including 50 or 52 videos, enabling research and benchmarking for ISL recognition improvements.

The dataset's focus on motion features for dynamic gestures, the emphasis on diversity in skin color, and the standardized setup with a black background contribute to its utility in advancing hand gesture recognition technology, particularly for emergency communication in Indian Sign Language.

Initial data was in the form of video samples of 2 seconds each, we extracted the frames of all the video and only considered a total of 16 frames out of those frames. All the frames were equally spaced, this was done so that we have to manage less data, and equally spaced frames will capture the movement uniformly. We also applied a few transformations to the frames. For model, a given example will consist of 16 frames stacked together and then converted into a tensor. The whole tensor will have a single label corresponding to the video to which the frames belong. For a given frame, we apply the following set of transformations. First, the image is resized to a reduced dimension. If the model is CNN+RNN, then reduce it to 224x224, or else 112x112. After that, we apply a horizontal flip on the image with a p-value of 0.5. Then we pass the frame through the random_affine function, which applies a random transformation to the image, whether rotation, scaling, or a combination of both. Finally, we convert the frame to a tensor and then normalize it. After the frames are transformed, they are ready to get stacked and combined to a single example in the dataset.

4. Methodology

We employed two distinct techniques for sign language detection from videos: utilizing a 3D CNN and a combination of CNN and RNN. The motivation behind this dual approach is to capture both spatial and temporal features within video frames, enhancing the model's ability to recognize and classify sign language gestures accurately.

In the 3D CNN approach, we employed the ResNet18 architecture adapted for three-dimensional data. This model

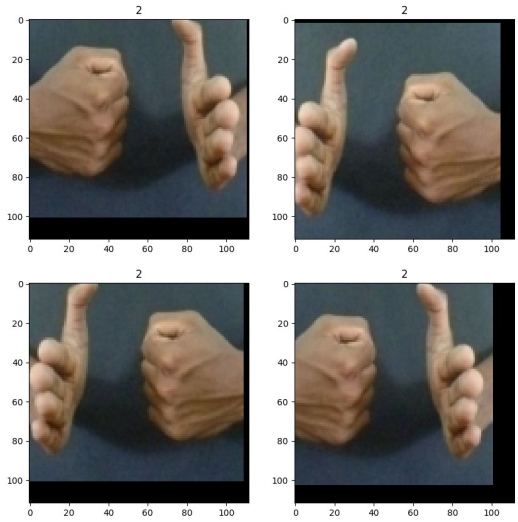

Figure 2. Final set of frames for category 'accident'

is specifically designed to capture spatial and temporal information simultaneously. The ResNet3D-18 model processes video frames as a sequence, enabling it to discern patterns evolving over time. The convolutional layers in the network are instrumental in extracting spatial features, while the 3D aspect ensures temporal nuances are considered.

For the CNN + RNN approach, we utilized the ResNet50 model for spatial feature extraction, coupled with an RNN, specifically LSTM layers, for temporal feature analysis. The ResNet50 backbone processes individual frames to capture spatial information, and the subsequent LSTM layers process the sequence of these features, allowing the model to understand temporal relationships. The final classification into one of the 8 sign language categories is performed using a fully connected layer. The flow-diagram of the methodology is shown in Figure 3.

4.1. Resnet3D-18 model

The Resnet3D-18 (R3D18) model [1] extends the traditional ResNet architecture of 18 layers to accommodate the temporal dimension in video data. By leveraging residual blocks and temporal pooling layers, it effectively captures both spatial and temporal features, making it well-suited for tasks involving video analysis, such as sign language detection. Residual networks allow short-circuiting path between different layers which in turn benefits training. Each residual block consists of two 3D convolution layers with a shortcut connection. The shortcut connection is used to skip one or more layers, facilitating the smooth flow of of gradients during backpropagation, mitigating the vanishing gradient problem during training. To aggregate temporal information and downsample the feature maps, temporal pooling layers (usually max pooling) are employed at strategic points in the network. Temporal pooling helps reduce the dimensionality of the data while retaining essential temporal features. The final layers of the R3D18 include one or more fully connected layers, culminating in an output layer with the number of neurons corresponding to the number of classes in the classification task. These layers are responsible for making the final decision on the class of the input video sequence. The output of the model is a probability distribution over the classes, indicating the likelihood of the input video belonging to each class. The architecture of R3D18 is shown in Figure 4. We loaded the pre-trained R3D18 model from TorchVision PyTorch library. The fully connected layer of the pre-trained model is replaced with a new one, adapting the number of output classes to the specific problem, in this case, the 8 categories of Indian Sign Language gestures.

4.2. Resnet50 model

ResNet50 [2] is a powerful and widely used deep learning model pretrained on the large ImageNet dataset. Its residual blocks and skip connections enable the training of deep networks, making it suitable for extracting complex features and patterns from images, which is especially valuable in tasks involving visual recognition, including sign language detection.

The architecture is a deep convolutional neural network comprising 50 layers in 4 stages [6]. It employs a series of residual blocks (different number of blocks in each of the 4 stages), each containing a bottleneck structure consisting of 1x1, 3x3, and 1x1 convolutions. The bottleneck architecture reduces computational complexity while retaining expressive power, facilitating the learning of intricate hierarchical features. Skip connections in each residual block enable the allow the output of one layer to be added to the output of another layer situated deeper in the network. In essence, they create shortcut paths for the gradient during backpropagation, aiding the training of deep networks. The network incorporates spatial pooling layers to reduce the spatial dimensions of feature maps and a final fully connected layer for task-specific classification.

We loaded the pre-trained ResNet50 model using PyTorch and replaced the final fully connected layer with a

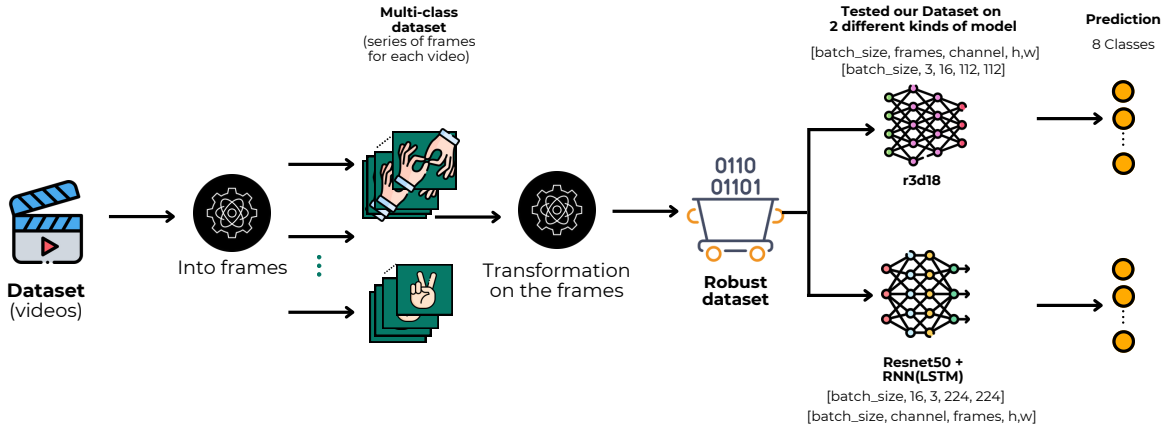

Figure 3. Sign language gesture recognition methodology

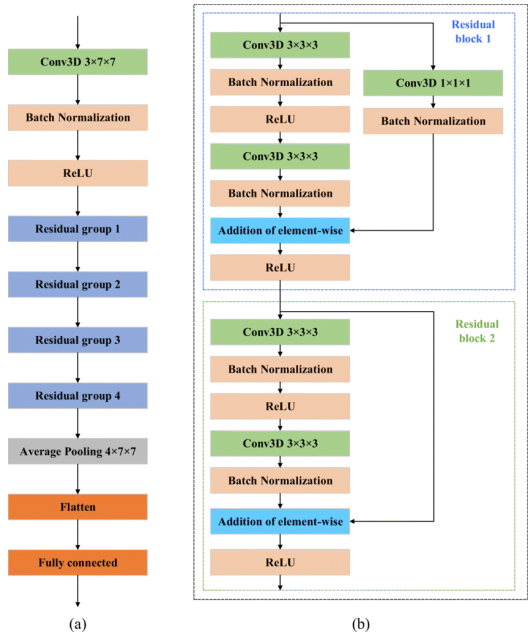

Figure 4. R3D18 model architecture [5]

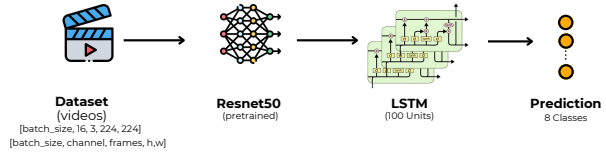

Figure 5. Resnet50 + LSTM approach

module with no-trainable parameters. The output of the last pool layer which gives a 2048-dimensional vector representing the convoluted features of the image are given as input to an LSTM layer consisting of 100 units. The 3 LSTM layers are crucial for analyzing the temporal sequence of features extracted by the ResNet50. LSTMs excel at capturing long-term dependencies and patterns in sequential data, making them well-suited for understanding the the dynamic evolution of gestures in a video. The final output is obtained by passing the output of the last LSTM time step through a fully connected layer performing the classification into one of the 8 sign language categories. This approach is depicted in Figure 5. Additionally, we also extracted the outputs from some intermediate and equally spaced time steps in order to analyze the model performance at intermediate frames.

5. Experimental setup

The preprocessed dataset is split into training, validation, and testing sets with 80-10-10 ratio respectively. Hence there are total of 333 videos in the train set, 37 in the validation set and 42 in the test set. The input image format for both the models are of different dimensions: for R3D18 it is 16x3x112x112 signifying 16 RGB image frames, each of dimension 3x224x224. Similarly for Resnet50 + LSTM model, the input dimension is 3x16x224x224 as seen in Figure 3. The train set is organized in to batches of 32 and for evaluating the model, the batch size is set to 8 due to the lesser number of videos in the validation and test sets.

The training setup for the models involves configuring various parameters and utilizing a training/validation loop to optimize the model's performance. The stochastic Adaptive Moment Estimation (ADAM) optimizer with a learning rate of 3 x 10^{-5} was used to minimise the Cross Entropy Loss, suitable for multi-class classification tasks. The learning rate

is adjusted during training using a learning rate scheduler, which decreases the learning rate by a factor of 0.5 if the validation loss does not improve for a specified number of epochs (patience). The training loop spans a specified number of epochs (20 for R3D18 model as the model converged fast and 100 for ResNet50), with early stopping implemented if the validation loss does not improve for a given patience period (set to 10 epochs). The best model's weights are saved during training if the validation loss reaches a new minimum. This setup ensures the efficient training of the model, with mechanisms in place to prevent over-fitting and adjust the learning rate dynamically for optimal convergence. Additionally, the training history, including loss and accuracy metrics, is recorded for later analysis.

6. Results and Analysis

In this section we evaluate the performance of the two approaches we used to detect the ISL gestures from a video. The final classification results in the case of ResNet50 + LSTM model are based on the outputs from only the last time-step of the LSTM layer, i.e. the 16^{th} frame from a video. However, we have also extracted the classified outputs at every 4^{th} intermediate frame, i.e. $4^{th}, 8^{th} and 11^{th}$ frame, for deeper analysis. We also plotted the optimization (loss) and performance (accuracy) learning curves as shown in Figures 7 and 8 to analyze the fit of the model on the train and validation data. Table 1 summarizes the classification results for both the models on the test set.

6.1. R3D18 model performance

The results presented in Table 1 indicate that the pre-trained R3D18 model achieves impeccable accuracy of 100% within just 20 epochs. The learning curves depicted in Figure 7 further support this observation, showcasing a convergence of both training and validation losses to a stable point with minimal gaps between them. Consequently, the model effectively classifies all videos into the specified 8 classes or gestures.

6.2. ResNet50 + LSTM model performance

The under-fitted learning curves for the ResNet50 + LSTM model suggest untapped potential for further learning and improvements. Unfortunately, due to GPU configuration limitations, we could only run the model for 100 epochs on CPU, as the training time for 100 epochs exceeded practical constraints. Despite this, the loss curve in Figure 8 indicates that the loss had reached a minimum and would likely stabilize with prolonged training. While the training accuracy started to stabilize, the validation accuracy, albeit noisy, reached a maximum of 94.59%. As reported in Table 1, the accuracy on the test set is 88.1%, indicating room for improvement with extended training.

The gestures predicted at every intermediate fourth frame for the entire test set are summarized in the confusion matrices displayed in Figure 6. Notably, the rightmost matrix in Figure 6 represents the confusion matrix for the final model predictions. In the initial two intermediate frames (4^{th} and 8^{th}), gestures such as 'pain,' 'hot,' and 'call' are frequently confused, likely due to their similar open-hand movements, as illustrated in Figure 1. For instance, Figure 9 provides an example where the gesture 'call' is misclassified as 'hot' in the 4^{th} frame. Additionally, the gesture 'lose' consistently fails to be classified correctly, often predicted as 'pain,' 'call,' 'accuracy,' or 'thief,' as evident in all the confusion matrices in Figure 6. This confusion can be attributed to the model's insufficient training, given the dissimilarity in hand gestures for these words.

Overall, these findings underscore the promising performance of the R3D18 model and the potential for further enhancements in the ResNet50 + LSTM model with extended training.

7. Conclusion

In conclusion, our project successfully addresses communication barriers for sign language users through an in-depth exploration of two recognition approaches on the ISL dataset. The R3D18 model demonstrates exceptional accuracy, achieving 100% within 20 epochs, showcasing its proficiency in capturing spatiotemporal features. Despite limitations in the ResNet50 + LSTM model's training duration, it exhibits untapped potential, with a test accuracy of 88.1%. Confusion matrices reveal insights into misclassifications, emphasizing the need for extended training to refine accuracy. Overall, our findings contribute significantly to advancing sign language recognition, highlighting the strengths and areas for improvement in both explored models.

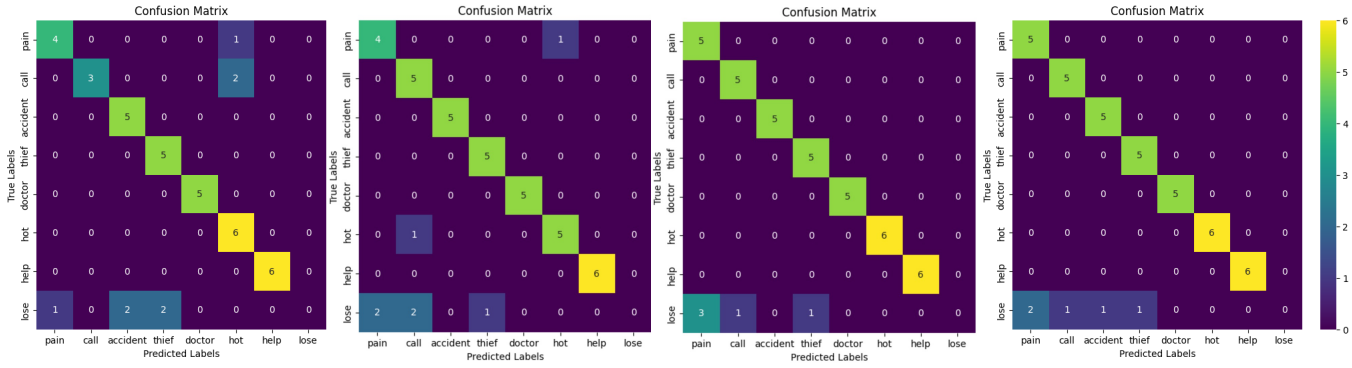

Figure 6. ResNet50 + LSTM model confusion matrix for (from left to right) $4^{th}, 8^{th}, 12^{th}$ and 16^{th} (final) frames

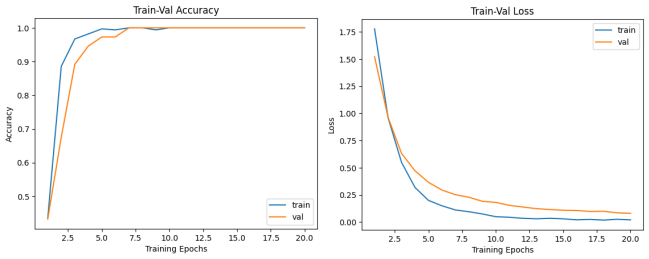

Figure 7. R3D18 model learning curves (loss and accuracy curves)

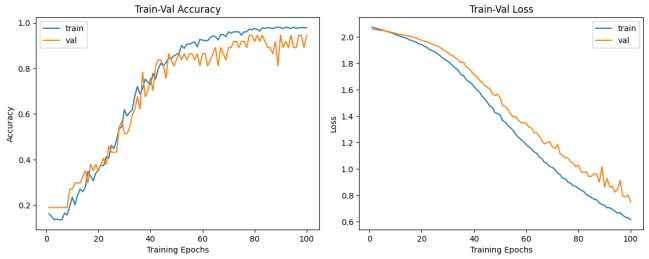

Figure 8. ResNet50 + LSTM model learning curves (loss and accuracy curves)

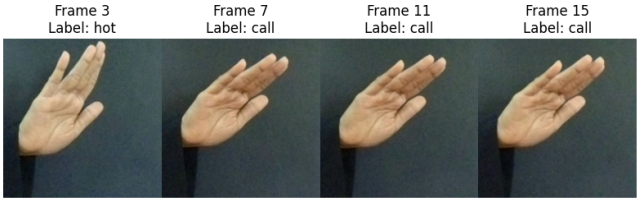

Figure 9. Predicted gestures by ResNet50 + LSTM model for intermediate frames

References

[1] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[3] V. Adithya and R. Rajesh, "Hand gestures for emergency situations: A video dataset based on words from indian sign language," *Data in Brief*, vol. 31, p. 106016, 2020.

[4] H. T. M. A. S. Masood, A. Srivastava, "Real-time sign language gesture (word) recognition from video sequences using cnn and rnn," 01 2018.

[5] L. Li, S. Qin, Z. Lu, K. Xu, and Z. Hu, "One-shot learning gesture recognition based on joint training of 3d resnet and memory module," *Multimedia Tools and Applications*, vol. 79, pp. 1–31, 03 2020.

[6] A. Bhatti, M. Umer, S. Adil, M. Ebrahim, D. Nawaz, and F. Ahmed, "Explicit content detection system: An approach towards a safe and ethical environment," *Applied Computational Intelligence and Soft Computing*, vol. 2018, 07 2018.

TABLE 1. RESULTS ON THE ISL DATASET

Model	Epochs	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
R3D18	20	100	100	100	100
ResNet50 + LSTM	100	78.74	88.1	82.86	88.1