Computer Modeling in Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.064588

ARTICLE

Video-Based Human Activity Recognition Using Hybrid Deep Learning Model

Jungpil Shin[1,*], Md. Al Mehedi Hasan[2], Md. Maniruzzaman[3], Satoshi Nishimura[1] and Sultan Alfarhood[4]

[1]School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, 965-8580, Japan
[2]Department of Computer Science & Engineering, Rajshahi University of Engineering & Technology, Rajshahi, 6204, Bangladesh
[3]Statistics Discipline, Khulna University, Khulna, 9208, Bangladesh
[4]Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh, 11543, Saudi Arabia

*Corresponding Author: Jungpil Shin. Email: jpshin@u-aizu.ac.jp

Received: 19 February 2025; Accepted: 23 May 2025; Published: 30 June 2025

ABSTRACT: Activity recognition is a challenging topic in the field of computer vision that has various applications, including surveillance systems, industrial automation, and human-computer interaction. Today, the demand for automation has greatly increased across industries worldwide. Real-time detection requires edge devices with limited computational time. This study proposes a novel hybrid deep learning system for human activity recognition (HAR), aiming to enhance the recognition accuracy and reduce the computational time. The proposed system combines a pre-trained image classification model with a sequence analysis model. First, the dataset was divided into a training set (70%), validation set (10%), and test set (20%). Second, all the videos were converted into frames and deep-based features were extracted from each frame using convolutional neural networks (CNNs) with a vision transformer. Following that, bidirectional long short-term memory (BiLSTM)- and temporal convolutional network (TCN)-based models were trained using the training set, and their performances were evaluated using the validation set and test set. Four benchmark datasets (UCF11, UCF50, UCF101, and JHMDB) were used to evaluate the performance of the proposed HAR-based system. The experimental results showed that the combination of ConvNeXt and the TCN-based model achieved a recognition accuracy of 97.73% for UCF11, 98.81% for UCF50, 98.46% for UCF101, and 83.38% for JHMDB, respectively. This represents improvements in the recognition accuracy of 4%, 2.67%, 3.67%, and 7.08% for the UCF11, UCF50, UCF101, and JHMDB datasets, respectively, over existing models. Moreover, the proposed HAR-based system obtained superior recognition accuracy, shorter computational times, and minimal memory usage compared to the existing models.

KEYWORDS: Human activity recognition; BiLSTM; ConvNeXt; temporal convolutional network; deep learning

1 Introduction

Human activity recognition (HAR) is one of the most active challenges in the field of computer vision, enabling the construction of machines that understand human behavior and intentions and provide better services. Specifically, HAR can detect human criminal acts (unusual events) from images and videos obtained from surveillance cameras [1] around town and in facilities and assist in the detection of human falls in hospitals, nursing homes, and care facilities, thus reducing the time between the occurrence of an incident and human recognition and preventing increased risk. Similarly, in the entertainment field, HAR can be

easily applied to human-computer interaction (HCI) tasks [2,3], such as reflecting human behavior in games, and it is in increasing demand for many applications.

Recently, automated systems have gained more attention in the industrial world due to their ability to improve productivity and reduce computational costs [1]. Cloud computing utilizes an internet connection to facilitate the transfer of substantial data volumes to IT/OT(Information Technology/Operation Technology) applications. It is also easy to request additional computing resources as needed, making it easy to support machine learning (ML)- and deep learning (DL)-based models that require a large amount of resources. However, cloud computing also requires constant internet connectivity and is unsuitable for use in industries with intermittent network connectivity or without network connectivity. Cloud computing is also bandwidth-intensive because it requires large amounts of data to be sent to the servers where computation and storage take place. This can lead to high communication costs in an environment where vast amounts of information are generated. Moreover, round-trip network latency can cause application response times to be seconds to minutes. This can be problematic for use cases that require near-real-time response and decision-making (e.g., abnormal detection and fall detection) [4]. Edge computing has also gained more attention as a solution to the disadvantages of cloud computing, but edge computing resources are limited, so it is necessary to propose an automated system that can operate with as few resources as possible.

The HAR recognition method depends on the type of data, and sensor-based HAR methods that use data from accelerometers and gyro sensors provided by wearable sensors are also attracting attention. Sensor-based approaches are superior in terms of protecting user privacy, having fewer location restrictions, and recognizing body motion-related behavior. Thakur et al. [5,6] proposed a unique DL-based approach involving multi-head convolutional neural networks (CNNs) and long short-term memory (LSTM), and reported state-of-the-art results using a large benchmark dataset and an original dataset, which was recorded by smartphone sensors. Computer vision is an active and important field in computer science, and it includes ML- and DL-based approaches. It is the process by which computers process information obtained mainly from visual sensors, such as cameras. The AlexNet [7] system was proposed for image classification in 2012. Following that, VGGnet [8] and ResNet [9] were developed to solve the problem of gradient loss due to multilayering and to incorporate residual connection technology that enables further multilayering. A model with a high trade-off between the number of parameters and recognition accuracy was proposed for image classification.

Recently, various transformer-based models have been proposed, such as the vision transformer (ViT) [10], which adapts the transformer and multi-head self-attention modules [11] from natural language processing to computer vision. These models have become the primary models used in various tasks in the field of computer vision, including HAR. In HAR tasks, decisions based solely on individual frames are often inadequate because human activities generally span a sequence of frames. This temporal dependency means that to accurately classify an activity, it is essential to consider the context provided by multiple consecutive frames. This helps to capture the transitions and continuous nature of activities, which is crucial for effective recognition and classification [12].

In this study, a DL-based approach for HAR in video classification was designed by combining a CNN and RNN(Recurrent Neural Network). A video was regarded as a sequence of frames, and the CNN-based component was used to extract spatial features from individual frames, capturing details about objects and movements within each frame. Following that, the RNN-based approach was used for the analysis of temporal information and changes. The combination of CNN- and RNN-based approaches has shown partial success. However, the dependency on previous steps in the RNN, LSTM, and GRU(Gate Recurrent Unit) approaches hampers parallel processing, leading to prolonged training and inference times [11]. To solve these problems, we designed a hybrid DL-based approach for HAR by combining ConvNeXt (pre-trained

model) [13] and a temporal convolutional network (TCN)-based model. Our proposed system provided lower inference times and superior recognition accuracy. In summary, this study provides the following novel contributions:

1. The primary objective of video surveillance systems is the precise recognition of various human activities within visual data. We conducted validation testing by employing four well-known benchmark datasets: UCF11, UCF50, UCF101, and JHMDB. To enhance the feature generation efficiency, reduce the computational time, and minimize memory usage, this study used data augmentation methods, transfer learning, and ConvNeXt.
2. We reshaped the features extracted by ConvNeXt as sequence data, and a TCN, which consists of residual join and extended convolution, was employed. This approach enabled time-dependent dependencies to be captured and facilitated classification tasks, all while supporting parallel processing.
3. Finally, the proposed combined ConvNeXt and TCN method is faster than the vision transformer in terms of the inference speed and has lower memory usage. This means that we can expect our proposed method to have a faster inference speed and lower memory usage than previous methods.

2 Related Works

In this section, we review some existing studies that are relevant to our proposed system and address the challenges of the HAR task. Various DL-based methods have been proposed for HAR. For example, 3DCNN was employed to capture both spatial information and temporal relationships [14]. The I3D-based model [15] can easily handle both RGB (Red, Green, Blue) and optical flow data with robust feature extraction, and it has been widely used in various domains, such as abnormal detection. Moreover, R3D [16], S3D [17], T3D [18], and LTC [19] have also been used successfully for HAR tasks. Despite the successful utilization of 3DCNN-based architectures for HAR tasks, their efficiency declines with longer frame sequences due to escalating computational demands.

Our approach was to address the HAR task with video using a combination of CNN- and RNN-based models. While 2DCNN captured only the spatial information for a single frame, many HAR methods contain some kind of temporal dependency. To solve this issue, we used an RNN (a sequential analysis model), along with the spatial information extracted by a 2DCNN model. This combination enables us to capture the temporal changes over time, effectively addressing the temporal dependencies present in human behavior recognition tasks.

Ullah et al. [20] proposed a densely connected bidirectional LSTM (BiLSTM) method. To reduce redundancy and complexity, they extracted CNN-based features from a video at intervals of four, six, and eight frames and learned continuous information between frame features with a multilayered BiLSTM. They used three benchmark datasets (UCF-101, YouTube Action, and HMDB51) to evaluate their proposed system and obtained a recognition accuracy of 91.21% for UCF-101, 92.84% for YouTube Action, and 87.64% for the HMDB51 dataset.

Ahmad et al. [21] proposed a system combining a CNN with the BiGRU method by employing a GRU with a simple structure and low computational costs for LSTM. They employed VGG16, consisting of only 16 weighted layers, to reduce the computational cost. They extracted features using the VGG16 model and evaluated the potentiality of these features using a random forest (RF)-based model to reduce the training complexity and remove noisy and redundant features. They also used three benchmark datasets and obtained a recognition accuracy of 91.79% for UCF-101, 93.38% for YouTube Action, and 71.89% for the HMDB51 dataset.

Hussain et al. [22] focused on the shortcomings of conventional CNN-based models and convolutional layers, which limit the learning of long-range dependencies beyond the receptive field. They attempted to solve the problem of long-range spatial dependencies using the vision transformer, which has a large receptive field and does not use convolutional layers. The extracted features were trained to learn temporal dependencies using an LSTM-based model and achieved a recognition accuracy of 96.14% for UCF50 and 73.71% for the HMDB51 dataset.

Wensel et al. [23] also focused on transformers and proposed the recurrent transformer (ReT) to solve the problem of RNN-, LSTM-, and GRU-based models not being able to perform parallel processing in a series analysis model. In their experiments, they used four combinations of models: ResNet50 + LSTM, ResNet50 + ReT, ViT + LSTM, and ViT + ReT. They optimized the parameters of the LSTM and ReT models. They also evaluated the performance of the proposed method using four benchmark datasets. They reported that the ViT-ReT-based model achieved superior recognition accuracies of 94.70% for UCF-101, 92.40% for YouTube Action, 97.10% for UCF50, and 78.40% for the HMDB51 dataset.

Existing studies have used ViT- and VGGNet-based models with a large number of parameters (86.6M parameters and 138.4M parameters). The greater number of parameters enhanced the expressiveness of the models, facilitating the capture of intricate patterns and enabling more adaptable responses to diverse data. However, this also increased computational costs, posing challenges for edge computing, which has constrained resources. Additionally, the LSTM and RNN models lack parallel processing capabilities, potentially increasing the computational time and also impacting real-time detection.

3 Materials and Methods

3.1 Proposed Methodology

We propose a hybrid deep learning framework that combines ConvNeXt for spatial feature extraction and a TCN for sequential modeling. ConvNeXt captures high-level patterns in the input data, while the TCN effectively models temporal dependencies using dilated convolutions. The extracted features from ConvNeXt are fed into the TCN, ensuring that both spatial and temporal characteristics are learned optimally. Fig. 1 illustrates this architecture, depicting the interaction between ConvNeXt and the TCN in our framework. We divided our experimental protocol into four parts: (i) data partitioning, (ii) feature extraction, (iii) classification, and (iv) performance evaluation. First, we divided our experimental datasets (using four datasets: UCF11, UCF50, UCF101, and JHMDB) into three parts: a training set, validation set, and test set. We took 70% of the dataset as the training set, 20% for the test set, and the remaining data were used as a validation set. We also converted all videos into frames and took images for each partition. The second stage was feature extraction, which involved extracting features from each frame of the videos by combining a CNN and a vision transformer and then feeding those features into a time sequence analysis model. Here, the CNN captures the spatial information of the video, while the time sequence analysis model captures the temporal information. Subsequently, we extracted features from each frame using a tuned feature extractor and reshaped them so that they had the same sequence order as the original video. Finally, we trained the sequence analysis models using the training and validation sets and evaluated their performance using the test set.

3.2 Experimental Dataset

This study used four benchmark datasets, UCF11, UCF50, UCF101, and JHMDB, to evaluate the performance of the proposed HAR-based system. Descriptions of each dataset are provided in the following subsections.

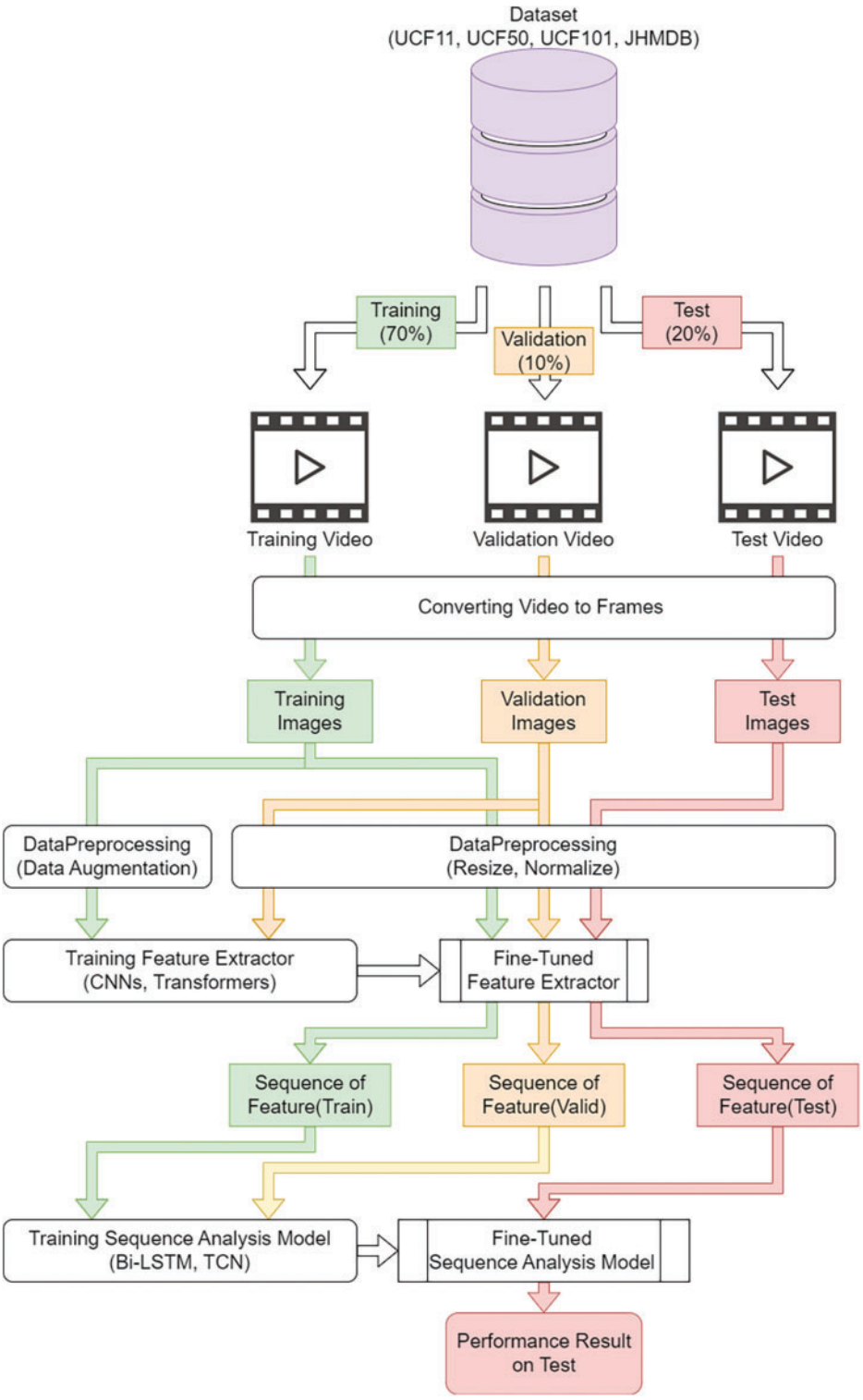

Figure 1: Overview of proposed HAR system created by combining ConvNeXt and TCN

3.2.1 UCF11

The UCF11 dataset [24] consists of data from 11 distinct sports action categories: volleyball, basketball, golf, horseback riding, biking/cycling, tennis, diving, football, swinging, jumping, and walking with a dog. It includes video clips from 25 subjects, with each subject contributing more than four clips. These clips share common features, such as the same actor, similar backgrounds, and consistent viewpoints. However, the dataset is challenging due to substantial variations in camera motion, object appearance and pose, object scale, viewpoint, background, and illumination conditions. This dataset has 1600 original videos in total. We used 1120 original videos for the training set; they yielded 178,981 clipped videos. Another 160 original videos were assigned to the validation set, resulting in 26,104 clipped videos. The rest of the original videos (320 videos) were utilized for the test set and produced 51,450 clipped videos.

3.2.2 UCF50

The UCF50 dataset [25] contains 50 action categories, with video clips collected from YouTube in ".avi" format. Each class was categorized into groups that share common features. For instance, one group features a person playing a piano four times from different viewpoints. The dataset is highly diverse, encompassing a wide range of human activities, with significant variation in camera motion, poses, object appearances, viewpoints, backgrounds, and illumination conditions. UCF50 consists of 6681 original videos. This study partitioned the dataset as follows: 4676 original videos were allocated for the training set, yielding 784,700 clipped videos. About 668 original videos were designated for the validation set, resulting in 112,624 clipped videos. The remaining original videos (1337 videos) were reserved for the test set, producing 225,583 clipped videos.

3.2.3 UCF101

The UCF101 dataset [26] was one of the largest realistic datasets for HAR, contains approximately 13,320 videos across 101 action classes. These videos were extracted from YouTube and fall, are categorized into five main types: sports, human body movement, playing music, human-to-human interaction, and human-to-object interaction. The dataset presents significant challenges due to the similarities between different classes, as well as variations in illumination conditions and viewpoints. Unlike many other HAR datasets, which focus on unrealistic, pre-planned actions performed by actors, UCF101 consists of real-life videos captured in natural settings. UCF101 contains a total of 13,320 original videos. Among them, 9324 original videos were used for the training set, resulting in 1,448,573 clipped videos. Another 1332 original videos were assigned to the validation dataset, resulting in 203,724 clipped videos. The remaining original videos (2664 videos) were assigned to the test set, and we made 418,082 clipped videos from them.

3.2.4 JHMDB

The JHMDB dataset [27] consists of 21 distinct action categories, including activities such as catching, clapping, hair-brushing, baseball bat swinging, gunshot firing, jumping, and more. It includes a total of 923 videos, each featuring a variety of actions, making it a challenging dataset for activity recognition. Among them, 649 original videos were assigned to the training dataset, resulting in 3380 clipped videos. Another 93 original videos were used for the validation dataset, resulting in 448 clipped videos. The remaining original videos (186 videos) were assigned to the test dataset, resulting in 1023 clipped videos.

3.3 Proposed Architecture

Fig. 2 presents an overview of the proposed DL-based method for HAR. The system consists of two main components: a spatial feature extractor (CNN) to capture spatial information and a sequence analysis model (RNN) to capture temporal information. The innovation of this paper lies in using ConvNeXt as a spatial feature extractor and TCN as a sequence analysis model. Initially, frames were extracted from the videos, followed by the extraction of spatial-based features using a pre-trained model. Subsequently, the extracted features were reshaped into sequence data, and the sequence analysis model was employed to predict the final human activity. Each step of the proposed system is further elaborated in the following subsections.

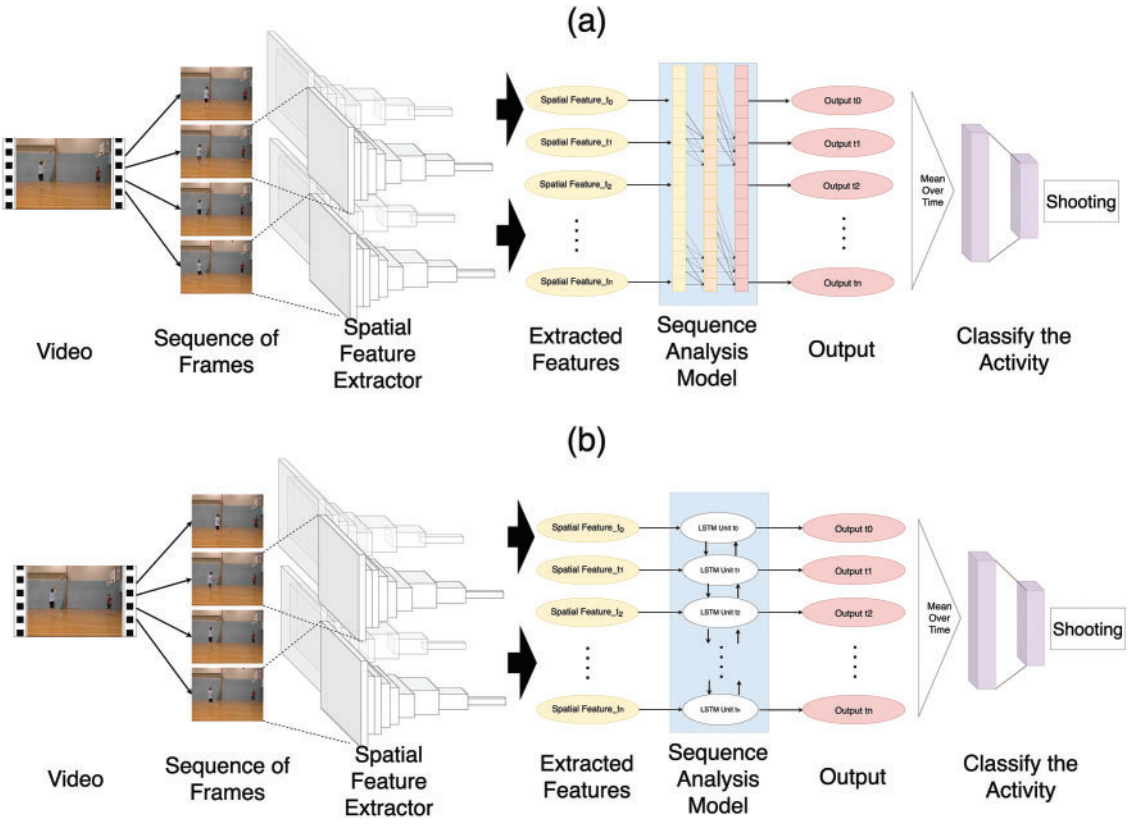

Figure 2: Overview of proposed architecture for HAR: (a) Feeding extracted features to TCN, calculating the mean outputs, and passing them into a fully connected layer for activity classification. (b) Feeding extracted features to LSTM and classifying the activity based on the mean output of LSTM

3.4 Spatial Feature Extractor

To capture the spatial information from the videos, features were extracted using pre-trained image classification models. The image classification models inherently captured the spatial information for each frame. We considered the videos as sequences of frames, and spatial information was obtained by adapting an image classification model for each frame. Four image classification models were used to extract the spatial information and features. Brief descriptions of these models are presented as follows.

3.4.1 ResNet50

ResNet is an image recognition model proposed in 2015 by He et al. [9]. At that time, it was expected that the more layers a CNN-based model had, the better its performance would be. However, these models faced the problem of learning poorly due to the gradient loss problem caused by multilayering. To solve this problem, ResNet has a residual block with skipped connections that is incorporated into the model, thereby eliminating the gradient loss problem. Other models that have been proposed include ResNet18 and ResNet34, which have shallower layers, and ResNet101 and ResNet152, which have deeper layers. In this experiment, ResNet50 was used because it does not have a large number of total parameters and it provides an adequate performance; additionally, since this model has been used for comparison in previous studies [23], it was also employed in this study in an ablation study for comparison.

3.4.2 EfficientNetB4

EfficientNet is an image recognition model proposed in 2019 by Tan and Le [28]. Conventional models at that time had non-uniform model scaling, making it difficult to design an efficient model due to serious computational costs and an increasing number of parameters. Therefore, Tan and Le proposed a new scaling method that scales the different dimensions (depth, width, and resolution) equally, and this approach achieved a high performance with a minimal increase in the computational cost and number of parameters. In this study, we employed EfficientNetB4, which has fewer parameters than ResNet50 but performs equally well, and compared it to our proposed method as part of an ablation study.

3.4.3 Vision Transformer

The vision transformer (ViT) was proposed in 2020 by Dosovitskiy et al. [10] as an alternative to CNN-based models for image recognition. CNN-based models can easily capture local features due to their convolutional nature but struggle with capturing long-range spatial dependencies. To solve this limitation, ViT patchifies images and introduces a self-attention mechanism. This mechanism calculates the attention for each patch, enabling the model to compute based on these calculations. In this experiment, ViT_B_16, which is the ViT model with the lowest number of parameters, was used for the experimental comparison. Additionally, to make it easy to perform a comparison with state-of-the-art approaches [23], we used it for comparison and an ablation study.

3.4.4 ConvNeXt

ConvNeXt is a model proposed in 2022 by Liu et al. [13] that achieves a higher performance than the ViT system using a traditional CNN-based approach. Transformer-based models have been proposed, and the model is based on the ResNet50 model. Derived from ResNet50, this model is compact and exhibits a higher performance than the Swin transformer [29] model. This achievement is attributed to optimizations in various aspects, including the stage ratio, patchiness adopted in the vision transformer, kernel size of the convolution layer, activation function, and normalization layer, all aligned with the transformer model's state. For this experiment, we also used ConvNeXt_tiny, which had fewer parameters than ViT_B_16 but provided a higher recognition accuracy. We expect that these advantages will contribute to the reduction of the required computing resources and the extraction of powerful spatial features.

3.5 Sequence Analysis Model

As explained in the previous section, image classification models can capture the spatial information of a single frame, but it is difficult for them to capture temporal information and changes. Therefore, by

employing a sequence analysis model such as an RNN, we can capture spatial and temporal information from a video by analyzing the features extracted from each frame as a sequence.

3.5.1 Long Short-Term Memory

RNNs were introduced to efficiently analyze patterns and learn dependencies in serial data [30]. However, when they first appeared, there were problems with learning dependencies on long timescales due to gradient loss or gradient explosion. Therefore, LSTM networks with a special structure, with inputs, outputs, forget gates, and memory cells, were implemented to learn longer dependencies more efficiently. Eqs. (1)–(7) represent the computations carried out by LSTM units:

$$i_t = \sigma((x_t + s_{t-1})W^i + b_i), \tag{1}$$

$$f_t = \sigma((x_t + s_{t-1})W^f + b_f), \tag{2}$$

$$o_t = \sigma((x_t + s_{t-1})W^o + b_o), \tag{3}$$

$$g = \tanh((x_t + s_{t-1})W^g + b_g), \tag{4}$$

$$c_t = c_{t-1} \cdot f_t + g \cdot i_t, \tag{5}$$

$$s_t = \tanh(c_t) \cdot o_t, \tag{6}$$

$$final\_state = softmax\left(V \cdot \frac{\sum_{t=0}^{Length} s_t}{Length}\right). \tag{7}$$

In the equations, x_t represents the input data at time t, and s_t represents the state of the hidden layer at time t. The input gate it (Eq. (1)) calculates how much new information is reflected by x_t and s_{t-1}. f_t is the forget gate (Eq. (2)) at time t and calculates how much of the previous information in the memory cell is retained. o_t represents the output gate (Eq. (3)) at time t and calculates how much information from long-term memory will be retained in the next step. g is computed through the activation function Tanh based on the input x_t and the state of the hidden layer s_{t-1} at the previous step (Eq. (4)). Then, using the computed g, the input gate i_t, the memory cell c_{t-1} from the previous step, and the forget gate f_t, the information in the memory cell c_t is updated (Eq. (5)). Finally, the states and outputs of the hidden layer s_t at time t are computed using the information of the memory cell and the output gate o_t (Eq. (6)). The mean outputs of the LSTM are fed into the fully connected layers, and finally a softmax function is used to obtain the prediction probabilities for each class (Eq. (7)). LSTM and BiLSTM have been used in previous studies [22,23,31] and achieved outstanding performances. We also employed these similar methods to make it easier to perform a comparison.

3.5.2 Temporal Convolutional Network

The TCN is a network that uses residual connections and dilated convolution layers to capture long-term dependencies. The dilated convolution layer is implemented for the purpose of increasing the receptive field, as shown in Fig. 3, where the layers are not convolved with adjacent elements but with the gaps between them. Also, the spacing between the convolution gaps becomes wider with each layer. Residual connections are then implemented in each layer to prevent gradient loss due to multilayering and help stabilize the learning

process. In contrast to RNN models, where the next computation cannot be performed until the previous step's computation is finished, the TCN enables computations to be performed in parallel, thus lowering the computational time. Because of these advantages, we also adopted the TCN.

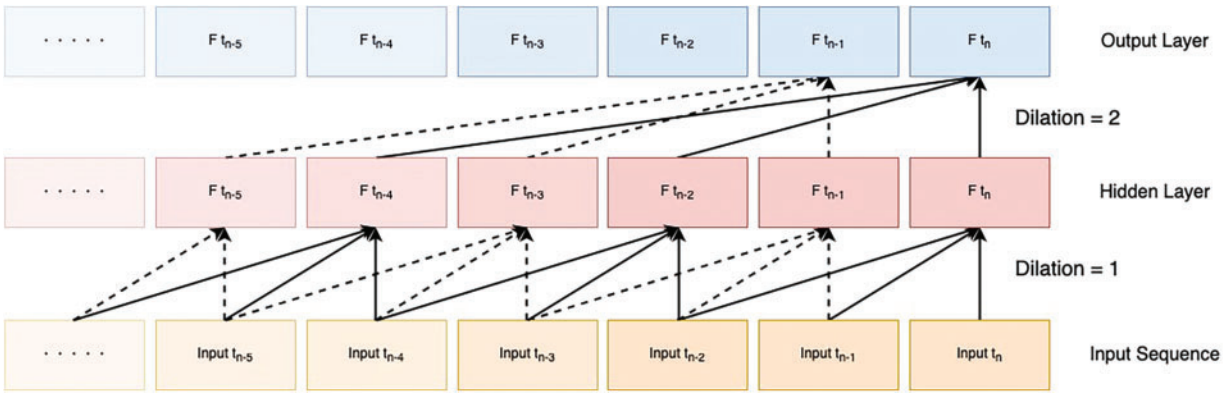

Figure 3: Dilated convolutional layer

3.6 Implementing the Spatial Feature Extractor

We implemented the four models introduced in the Proposed Methodology section. The four models are provided as pre-trained models in PyTorch and Torchvision. In this experiment, those pre-trained models were used and fine-tuned to extract optimized features for HAR by performing transfer learning. Transfer learning is a technique that reuses the weights and learning results of DL-based models trained on large-scale datasets of different problems to provide effective and efficient solutions for specific challenges. Using this method, models can be adequately trained to produce powerful features using image data on only a small amount of human behavior.

3.7 Training Procedure for Feature Extractor

3.7.1 Data Preprocessing and Data Augmentation

To train the image classification model as the feature extractor, images (or frames) were extracted from videos using OpenCV as a preprocessing step. Data enhancement was performed during training using the $create\_transform$ function provided by the timm library. Specifically, the input size was set to 224 × 224 pixels. The recommended input size was established for each image classification model. However, since larger input sizes are generally more computationally expensive, the images are resized to the most commonly used size of 224 × 224 pixels. The input images are normalized by subtracting the mean (0.485, 0.456, 0.406) and dividing by the standard deviation (0.229, 0.224, 0.225) for each channel. During training, the RandomAffine transform, horizontal and vertical rotation, ColorJitter, auto-augmentation [32], and RandomErasing were applied to prevent overfitting problems.

3.7.2 Training Procedure and Configuration

ResNet50-, EfficientNetB4-, ViT-, and ConvNeXt-based models were separately trained for 30 epochs; the batch size was 32, the optimizer was SGD [33], and training was performed using CosineLR Scheduler, with the learning rate ranging from 0.00125 to 0.000125. Label smoothing techniques were used to

suppress overfitting. Table 1 summarizes the details of the experiments. Finally, these features were used in classification models, which were trained for 30 epochs.

Table 1: Training configurations for ResNet50-, EfficientNetB4-, Vision Transformer-, and ConvNeXt-based models

Parameter name	Value
Input shape	(3, 224, 224)
Training epoch	30
Batch size	32
Optimizer	SGD [33]
Learning rate scheduler	CosineLRScheduler
Base learning rate (Max.)	1.25e-02
Minimum learning rate (Min.)	1.25e-04
Label smoothing [34]	0.01

3.7.3 Extract Features from Video

A fine-tuned classification model was used to extract features that were used to train the sequence analysis model. A resized and standardized frame of 224 × 224 pixels was passed to the fine-tuned image classification model to extract features that were later fed into the combined layers of the image classification model. The extracted features were stored as sequence data in the same order as the original video.

3.8 Training Procedure for Sequence Analysis Model

We implemented a sequence analysis model for HAR based on the sequence data, while the features were extracted by the fine-tuned classification model. Fig. 4 shows an overview of the BiLSTM-based model with adam optimizer [35]. The input data were fed into a BiLSTM-based model to allow it to learn long-term temporal dependencies, and the extracted features were used for classification. As explained in the Proposed Methodology section, the mean outputs of the LSTM were fed to the fully connected layer, and finally a softmax function was applied to obtain the prediction probabilities for each class.

Fig. 5 shows an overview of the TCN-based model. The TCN consisted of three temporal blocks. Each block consisted of a dilated 1D convolutional layer and residual connection. The deeper the position of the block, the larger the dilated size of the dilated 1D convolutional layer. This expanded the receptive field and captured more long-term time-dependent relationships. After passing through the temporal blocks, the average value was extracted for the time direction. The final output was then compressed into the number of classes to perform classification using the linear layer. Tables 2 and 3 summarize the parameters of each model and the training configuration parameters.

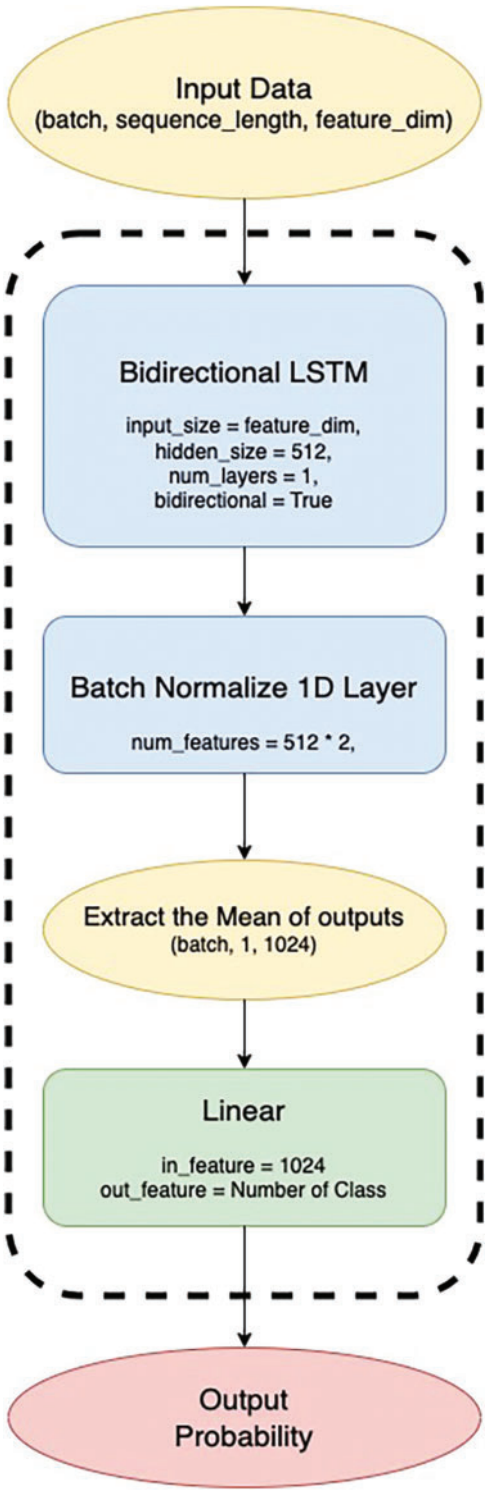

Figure 4: Architecture of BiLSTM-based model

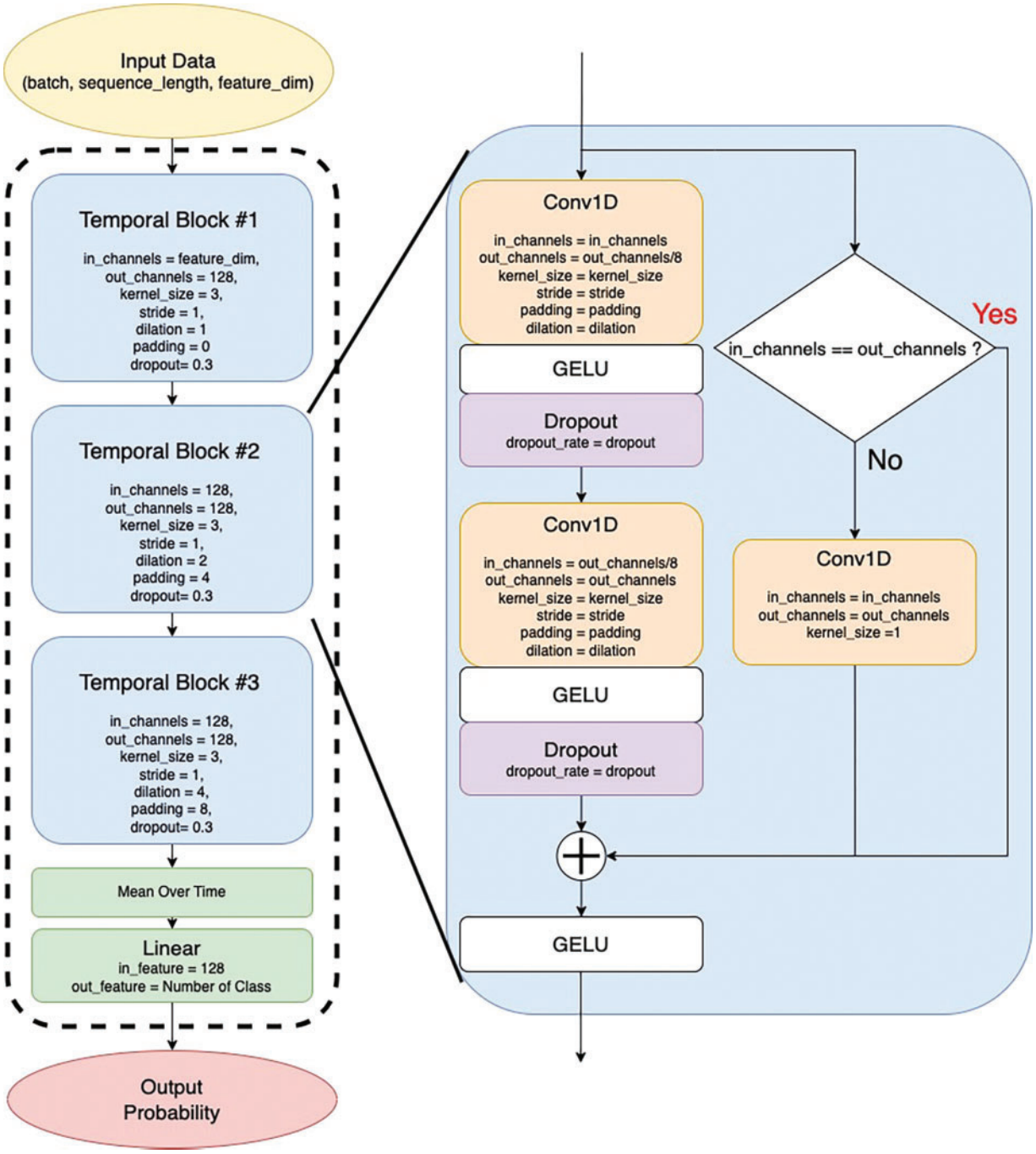

Figure 5: Architecture of TCN and temporal block-based models

Table 2: Hyperparameter tuning ranges and training configuration for BiLSTM-based model

Parameter	Value
Input Dim.	Depends on feature extractor
Sequence length	32
Hidden_Size	64–1024
Num_Layers	1–3
Number of classes	Depends on datasets
Training epoch	100
Batch size	512
Optimizer	Adam [35]
Learning rate scheduler	CosineLRScheduler
Base learning rate (Max.)	5e-04–5e-06
Minimum learning rate (Min.)	5e-07
Label smoothing [34]	0.01

Table 3: Hyperparameter tuning ranges and training configuration for TCN-based model

Parameter	Value
Input Dim.	Depends on feature extractor
Sequence length	32
Num_Channels	[64–256, 64–256, 64–256]
Kernel_Size	2–5
Dropout	0.3
Number of classes	Depends on datasets
Training epoch	100
Batch size	512
Optimizer	Adam [35]
Learning rate scheduler	CosineLRScheduler
Base learning rate (Max.)	5e-06–5e-04
Minimum learning rate (Min.)	5e-07
Label smoothing [34]	0.01

4 Experimental Environment and Performance Metrics

4.1 Experimental Environment

We performed our all experiments in Python and PyTorch [36], which were used primarily for the implementation and training of the models. A GPU PC with an Intel Core i9 13900 CPU and an NVIDIA Geforce RTX4090 was used to conduct our experiments. We used the DL-based frameworks "PyTorch" and "Torchvision" for feature extraction and classification. The configuration of the training parameters for the ResNet50-, EfficientNetB4-, VisionTransformer-, and ConvNeXt-tiny-based models for feature extraction is presented in Table 1. The hyperparameter and training parameter ranges for the BiLSTM- and TCN-based models are also presented in Tables 2 and 3. This study subdivided the datasets using a ratio of 70:10:20, where 70% of the datasets were used for the training set, 10% for the validation set, and the remaining 20%

for the test set. Also, one sample was made from a continuous 32-frame video, which is about the same as a 1-second video clipped from the original video. Then, by sliding the clip position from the original video, multiple 32-frame clipped videos were created from one original video, and these were used for model training. We trained our proposed system based on the training set and validation set, and we evaluated the performance using the test set. We used four performance metrics, the accuracy, precision, recall, and F1-score, for four datasets (UCF11, UCF50, UCF101, and JHMDB) to show the robustness and efficiency of our proposed system.

4.2 Performance Metrics

For the measurement of the model performances, we employed the accuracy (Acc), macro-average precision (Prec), macro-average recall (Rec), and macro-average F1-score (FS), which were computed from the confusion matrix. The computational formulae for calculating each performance metric are shown in Eqs. (8)–(11):

$$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{8}$$

$$\text{Prec} = \frac{1}{C}\sum_{i=1}^{C}\frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, \tag{9}$$

$$\text{Rec} = \frac{1}{C}\sum_{i=1}^{C}\frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}, \tag{10}$$

$$\text{FS} = \frac{1}{C}\sum_{i=1}^{C}\frac{2TP_i}{2TP_i + \text{FP}_i + FN_i}. \tag{11}$$

Here, C is the total number of classes; TP_i is the number of true positives for class i; TN_i is the number of true negatives for class i; FP_i is the number of false positives for class i; and FN_i is the number of false negatives for class i.

5 Results and Discussion

This study designed a set of eight combination systems by crisscrossing four feature extractors (ResNet50, EfficientNetB4, ViT_B_16, and ConvNeXt-T) and two sequential models (BiLSTM and TCN). We evaluated all combination systems on the test set for each dataset. We chose the best combination of feature extractors and sequential models, which provided outstanding performances. In the following section, a detailed explanation of the performance evaluation is provided for the UCF11, UCF50, UCF101, and JHMDB datasets.

5.1 UCF11

The recognition accuracy and other performance metrics in each combination system for the UCF11 dataset are illustrated in Table 4. We observed that the TCN-based model achieved better recognition accuracy compared to BiLSTM for both the validation and test sets. The combination of ConvNeXt-T with the TCN provided the highest recognition accuracy of 97.73% compared to other combination systems. We observed that the lowest recognition accuracy of 88.71% was achieved by the combination of ResNet50 and BiLSTM. Our proposed (ConvNeXt with TCN) system also obtained a precision of 97.58%, a recall of 97.19%, and an F1-score of 97.35% for the UCF11 dataset.

Table 4: Performance comparisons (in %) of various sequential models for UCF11 dataset. Bold values indicate the results of our proposed system

Feature extractor	Model	Valid Acc	Test set			
			Acc	Prec	Rec	FS
ResNet50	BiLSTM	91.62	88.71	88.71	87.54	88.03
ResNet50	TCN	93.48	90.70	90.37	90.10	90.09
EfficientNetB4	BiLSTM	95.47	93.02	93.12	92.54	92.76
EfficientNetB4	TCN	97.38	95.74	95.63	95.31	95.45
ViT_B_16	BiLSTM	98.80	96.84	96.78	96.49	96.58
ViT_B_16	TCN	98.85	97.04	96.88	96.66	96.72
ConvNeXt-T	BiLSTM	98.14	96.94	96.87	95.36	96.56
ConvNeXt-T	**TCN**	**99.21**	**97.73**	**97.58**	**97.19**	**97.35**

5.2 UCF50

Table 5 presents the recognition accuracy and other performance metrics of eight combination systems for the UCF50 dataset, with the last row indicating the performances of our proposed method. When ResNet50 and ConvNeXt are employed as feature extractors, the performance trends align with those observed on the UCF11 dataset. However, when EfficientNetB4 and ViT are used as feature extractors, there is a slight improvement in the performance of BiLSTM compared to the TCN. Our hybrid system, combining ConvNeXt and the TCN, achieved a recognition accuracy of 98.81%, a precision of 98.36%, a recall of 98.36%, and an F1-score of 98.34%.

Table 5: Performance comparisons (in %) of various sequential models for UCF50 dataset. Bold values indicate the results of our proposed system

Feature extractor	Model	Valid Acc	Test set			
			Acc	Prec	Rec	FS
ResNet50	BiLSTM	84.30	85.33	82.67	80.70	81.42
ResNet50	TCN	93.29	93.75	93.17	91.88	92.35
EfficientNetB4	BiLSTM	94.02	93.45	92.14	92.38	92.17
EfficientNetB4	TCN	91.98	92.12	90.96	90.20	90.49
ViT_B_16	BiLSTM	98.71	98.43	98.08	97.75	97.88
ViT_B_16	TCN	98.42	98.35	97.95	97.71	97.79
ConvNeXt-T	BiLSTM	98.62	98.40	97.84	97.89	97.84
ConvNeXt-T	**TCN**	**98.75**	**98.81**	**98.36**	**98.36**	**98.34**

5.3 UCF101

The performance results on the UCF101 dataset are given in Table 6. Using EfficientNetB4, ViT, and ConvNeXt as feature extractors, the TCN outperformed BiLSTM. On the other hand, the TCN-based model has shown a lower performance than BiLSTM when ResNet50 is employed as a feature extractor. The hybrid model combining ConvNeXt and the TCN achieved the highest recognition accuracy (98.46%) for the

UCF101 dataset. Our proposed method also obtained a precision of 98.06%, a recall of 97.99%, and an F1-score of 97.98%.

Table 6: Performance comparisons (in %) of our proposed system for UCF101 dataset. Bold values indicate the results of our proposed system

Feature extractor	Model	Valid Acc	Test set			
			Acc	Prec	Rec	FS
ResNet50	BiLSTM	86.48	86.01	83.88	82.56	83.00
ResNet50	TCN	83.58	82.37	80.07	77.80	78.58
EfficientNetB4	BiLSTM	93.54	92.53	91.02	91.07	90.93
EfficientNetB4	TCN	94.50	93.87	93.06	92.47	92.63
ViT_B_16	BiLSTM	98.54	98.01	97.41	97.46	97.37
ViT_B_16	TCN	98.62	98.05	97.45	97.48	97.39
ConvNeXt-T	BiLSTM	98.79	98.33	97.85	97.89	97.83
ConvNeXt-T	**TCN**	**99.07**	**98.46**	**98.06**	**97.99**	**97.98**

5.4 JHMDB

Finally, the performance results on the JHMDB dataset are given in Table 7. The hybrid model combining ConvNeXt and the TCN also achieved the highest accuracy (83.38%) for JHMDB. Our proposed method also obtained a precision of 79.24%, a recall of 78.37%, and an F1-score of 77.35%.

Table 7: Performance comparisons (in %) of our proposed system for JHMDB dataset. Bold values indicate the results of our proposed system

Feature extractor	Model	Valid Acc	Test set			
			Acc	Prec	Rec	FS
ResNet50	BiLSTM	70.76	56.01	55.38	44.62	47.37
ResNet50	TCN	63.62	57.28	46.16	45.88	44.10
EfficientNetB4	BiLSTM	56.92	51.42	43.97	41.66	39.91
EfficientNetB4	TCN	71.88	61.38	49.60	51.32	47.47
ViT_B_16	BiLSTM	82.59	79.57	77.03	77.49	75.27
ViT_B_16	TCN	79.91	80.94	77.94	78.08	75.69
ConvNeXt-T	BiLSTM	77.68	79.18	77.82	74.86	74.55
ConvNeXt-T	**TCN**	**81.25**	**83.38**	**79.24**	**78.37**	**77.35**

5.5 Comparison of Proposed System with the State of the Art

The comparison of our proposed system with existing state-of-the-art systems in terms of the recognition accuracy on only the test dataset is presented in Table 8 for UCF11, Table 9 for UCF50, Table 10 for UCF101, and Table 11 for JHMDB. The last rows of Table 8–11 show the recognition accuracy of the proposed method, and the bold values represent the highest recognition accuracies of our proposed system. Existing approaches like CNN-BiLSTM [20], 3DCNN [37], VGG-BiGRU [21], and ViT-ReT [23] obtained recognition

accuracies of 92.84%, 85.20%, 93.38%, and 92.40%, respectively, for HAR using the UCF11 dataset, while our proposed system achieved a recognition accuracy of 97.73% using the same UCF11 dataset, which indicates an almost 4% improvement in the recognition accuracy over the existing methods [21].

Table 8: Comparison of proposed system and existing systems for UCF11 dataset. Bold values indicate the results of our proposed system

Method	Year	Split ratio (train/val/test) (%)	Acc (%)
CNN-BiLSTM [20]	2018	60/20/20	92.84
3DCNN [37]	2022	70/10/20	85.2
VGG-BiGRU [21]	2023	60/20/20	93.38
ViT-ReT [23]	2023	80 (train)/20 (test)	92.4
Proposed system	**2024**	**70/10/20**	**97.73**

Table 9: Comparison of proposed system and existing systems for UCF50 dataset. Bold values indicate the results of our proposed system

Method	Year	Split ratio (train/val/test) (%)	Acc (%)
3DCNN [37]	2022	70/10/20	82.2
ViT-LSTM [22]	2022	Not given	96.14
ViT-ReT [23]	2023	80 (train)/20 (test)	92.4
Proposed system	**2024**	**70/10/20**	**98.81**

Table 10: Comparison of proposed system and existing systems for UCF101 dataset. Bold values indicate the results of our proposed system

Method	Year	Split ratio (train/val/test) (%)	Acc (%)
CNN-BiLSTM [20]	2018	60/20/20	91.21
Hybrid Model [38]	2020	75 (train)/25 (test)	89.3
VGG-BiGRU [21]	2023	60/20/20	91.79
ViT-ReT [23]	2023	80 (train)/20 (test)	94.7
Proposed system	**2024**	**70/10/20**	**98.46**

Table 11: Comparison of proposed system and existing system for JHMDB dataset. Bold values indicate the results of our proposed system

Method	Year	Split ratio (train/val/test) (%)	Acc (%)
Deep_BiLSTM [31]	2024	80 (train)/20 (test)	76.3
Proposed system	**2024**	**70/10/20**	**83.38**

As shown in Table 9, we considered the recognition accuracy for the UCF50 dataset for 3DCNN [37], ViT-LSTM [22], and ViT-ReT [23]. These methods achieved recognition accuracies of 82.2%, 96.14%, and 92.40%, respectively, for the UCF50 dataset. The second highest recognition accuracy of 96.14% was obtained

by ViT-LSTM [22] for the UCF50 dataset. However, our proposed system obtained the highest recognition accuracy of 98.81%, improving the recognition accuracy by up to almost 2.67% for the UCF50 dataset.

The comparison of the recognition accuracy of our proposed system and that of previous systems for the UCF101 dataset is shown in Table 10. The existing hybrid model obtained an 89.30% recognition accuracy for the UCF101 dataset [38]. However, other existing systems such as CNN-BiLSTM [20], VGG-BiGRU [21], and ViT-ReT [23] obtained recognition accuracies of 91.21%, 91.79%, and 94.70%, respectively. We observed that the ViT-ReT [23] system obtained the second highest recognition accuracy, while our system enhanced the recognition accuracy by almost 3.76% and achieved a recognition accuracy of 98.46% for the UCF101 dataset. This demonstrates the robustness and effectiveness of our proposed system for HAR.

The comparison of the recognition accuracy of the proposed system and that of a previous system for the JHMDB dataset is shown in Table 11. MobileNetV2-LSTM obtained a 76.3% recognition accuracy for the JHMDB dataset [31]. We observed that our proposed system achieved the highest recognition accuracy (83.38%), obtaining an almost 7% improvement in the recognition accuracy compared to the existing system [31].

5.6 Inference Time and Memory Usage

In this section, we measured the inference time (forward time) and memory usage of the spatial feature extractor and sequence analysis model used in our experiments. The experiments were carried out on a GPU PC equipped with an NVIDIA Geforce RTX3070 and an Intel Core i7 10700. We generated dummy inputs with a shape of (1, 3, 224, 224), assuming one sample (one frame) of input, and measured the average inference time of 300 trials for each spatial feature extractor.

In the same way, we assumed that the feature extractor was the ConvNeXt model, generated dummy inputs with a shape of (1, 32, 768), and measured the average inference time in 300 trials for each sequence analysis model. The memory usage referenced the "Param size" in the *summary* function of the "torchinfo" library. We show the inference time (forward time) and memory usage for the spatial feature extractor and the sequence analysis model.

5.6.1 Inference Time and Memory Usage for Each Model

Table 12 shows crucial performance metrics for various spatial feature extractors (SFEs), including the inference time, memory usage, and number of parameters. Among the SFEs evaluated, ConvNeXt-tiny demonstrated superior performance, characterized by the shortest inference time and minimal memory consumption. This efficiency is highly advantageous, particularly in resource-constrained environments where computational resources are limited.

Table 12: Computational time (in ms) and memory usage (in MB) for each spatial feature extractor

Feature extractor	Mean (Std)	Memory usage	Params (m)
ResNet50	5.3230 (0.0369)	272.55	25.6
EfficientNetB4	14.1278 (0.0660)	343.36	19.3
ViT_B_16	6.0222 (0.0957)	333.91	86.6
ConvNeXt-T	4.4615 (0.0277)	243.16	28.6

In addition, Table 13 presents crucial performance metrics for various SAMs, including the inference time, memory usage, and number of parameters. Notably, the inference speed of our proposed TCN model

was observed to be more than three times faster than that of the BiLSTM-based model. One possible explanation for this disparity could be the sequence length considered during the evaluation, which was set to only 32 frames.

Table 13: Computational time (in ms) and memory usage (in MB) for each SAM

Feature extractor	Mean (Std)	Memory usage	Params
BiLSTM	0.4207 (0.0075)	3.99	0.9 M
TCN	1.3443 (0.0149)	1.85	0.3 M

Table 14 provides valuable information on the inference times of sequence analysis models for different sequence lengths. In particular, BiLSTM exhibited increased inference times as the sequence length grew longer, whereas the TCN maintained constant inference times regardless of the sequence length. This behavior was attributed to the architectural differences between the models. Although BiLSTM relied on sequential processing, necessitating the completion of each step before proceeding to the next, the TCN's architecture enables parallel computation, allowing it to process input sequences in constant time regardless of length.

Table 14: Computational time (in ms) vs. sequence length

Model	Sequence length					
	16	32	64	128	256	512
BiLSTM	0.334	0.452	0.603	0.901	1.473	2.624
TCN	1.335	1.344	1.320	1.318	1.301	1.310

5.6.2 Inference Time and Memory Usage in Combination Methods

This section discusses the inference time and memory usage for each feature extractor and sequence analysis model combination. Table 15 shows the inference time and memory usage for each combination. As mentioned in the previous section, ConvNeXt's short inference time (forward time) was a significant advantage. The combined ConvNeXt and TCN method was slightly slower than the combined ResNet50 and BiLSTM method. However, the combined ConvNeXt and TCN method obtained a superior performance. The experiment also showed that the models combining ConvNeXt with the TCN and BiLSTM were faster than the ViT model, which was used as a feature extractor in previous studies. Moreover, the combination of ConvNeXt with the TCN and BiLSTM surpasses ViT in inference speed while maintaining low memory usage, making it more suitable for edge computing.

Table 15: Computational time (in ms) and memory usage (in MB) in combination methods

Feature extractor	Model	Time	Memory usage
ResNet50	BiLSTM	5.77	281.95
ResNet50	TCN	6.67	279.44
EfficientNetB4	BiLSTM	14.56	352.67
EfficientNetB4	TCN	15.42	348.98
ViT_B_16	BiLSTM	6.45	337.9
ViT_B_16	TCN	7.35	335.76
ConvNeXt-T	BiLSTM	4.88	247.15
ConvNeXt-T	TCN	5.82	245.01

6 Conclusion, Limitations, and Future Work

Various image classification models have been proposed in the field of computer vision and applied to the HAR task. However, edge computing for real-time processing has limited computational resources and may not be suitable for use with dense and complex models. Human behavior generally involves temporal dependencies, and RNNs and LSTM must be used to capture these relationships. Due to their structure, RNN and LSTM models cannot perform parallel processing, and the inference time can be long. In this paper, the objective was to develop a method that achieves high accuracy in human behavior recognition while reducing the required computational resources by combining a relatively small image classification model with a TCN capable of parallel processing. The experimental results show that the combined ConvNeXt and TCN method achieved a recognition accuracy of 97.73% for UCF11, 98.81% for UCF50, 98.46% for UCF101, and 83.38% for JHMDB, respectively.

In future work, we will continue to develop a HAR system that has a higher recognition accuracy, shorter inference speeds, and lower memory usage. In particular, our experiments have shown that the image classification model has a significant impact on the inference speed and memory usage. The development of an image classification model with a smaller size and faster inference speed is expected to contribute not only to the HAR task but also to all tasks related to the image classification field. We will adopt a skeleton-based approach or a sensor-based method for HAR that is not affected by the background.

Acknowledgement: The authors extend their appreciation to King Saud University for funding this research through Ongoing Research Funding Program (ORF-2025-890), King Saud University, Riyadh, Saudi Arabia.

Funding Statement: This research is funded by the Ongoing Research Funding Program (ORF-2025-890), King Saud University, Riyadh, Saudi Arabia.

Author Contributions: The authors confirm their contributions to the paper as follows: Conceptualization, Jungpil Shin, Md. Al Mehedi Hasan, and Md. Maniruzzaman; methodology, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, and Sultan Alfarhood; software, Md. Al Mehedi Hasan and Md. Maniruzzaman; validation, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, and Satoshi Nishimura; formal analysis, Jungpil Shin, Md. Al Mehedi Hasan, and Satoshi Nishimura; investigation, Jungpil Shin and Md. Al Mehedi Hasan; resources, Md. Maniruzzaman, Satoshi Nishimura, and Sultan Alfarhood; data curation, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, Satoshi Nishimura, and Sultan Alfarhood; writing—original draft preparation, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, and Satoshi Nishimura; writing—review and editing, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, Satoshi Nishimura, and Sultan Alfarhood; visualization, Jungpil Shin, Md. Al Mehedi Hasan, Md. Maniruzzaman, Satoshi Nishimura, and Sultan Alfarhood; supervision, Jungpil Shin; project administration, Jungpil

Shin and Satoshi Nishimura; funding acquisition, Jungpil Shin and Sultan Alfarhood. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: This study used four publicly available: UCF11 [24], UCF50 [25], UCF101 [26], and JHMDB [27]. One can easily access these four datasets in studying human actions and enhancing recognition algorithms.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Lin W, Sun MT, Poovandran R, Zhang Z. Human activity recognition for video surveillance. In: 2008 IEEE International Symposium on Circuits and Systems (ISCAS); 2008 May 18–21; Seattle, WA, USA. p. 2737–40. doi:10.1109/ISCAS.2008.4542023.
2. Iosifidis A, Tefas A, Pitas I. Multi-view action recognition based on action volumes, fuzzy distances and cluster discriminant analysis. Signal Processing. 2013;93(6):1445–57. doi:10.1016/j.sigpro.2012.08.015.
3. Weinland D, Ronfard R, Boyer E. A survey of vision-based methods for action representation, segmentation and recognition. Comput Vis Image Understand. 2011;115(2):224–41. doi:10.1016/j.cviu.2010.10.002.
4. Mishra SR, Mishra TK, Sanyal G, Sarkar A, Satapathy SC. Real time human action recognition using triggered frame extraction and a typical CNN heuristic. Patt Recogn Lett. 2020;135(11):329–36. doi:10.1016/j.patrec.2020.04.031.
5. Thakur D, Guzzo A, Fortino G. Attention-based multihead deep learning framework for online activity monitoring with smartwatch sensors. IEEE Int Things J. 2023;10(20):17746–54. doi:10.1109/JIOT.2023.3277592.
6. Thakur D, Roy S, Biswas S, Ho E, Chattopadhyay S, Shetty S. A novel smartphone-based human activity recognition approach using convolutional autoencoder long short-term memory network. In: 2023 IEEE 24th International Conference on Information Reuse and Integration for Data Science (IRI); 2023 Aug 4–6; Bellevue, WA, USA. p. 146–53. doi:10.1109/IRI58017.2023.00032.
7. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems. Red Hook, NY, USA: Curran Associates, Inc.; 2012. p. 1097–105. doi: 10.1145/3065386.
8. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556. 2014.
9. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 770–8.
10. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, et al. An image is worth 16 x 16 words: transformers for image recognition at scale. In: Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2021 Jun 20–25; Nashville, TN, USA. p. 3156–64.
11. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Advances in Neural Information Processing Systems 30 (NIPS 2017); Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 5998–6008.
12. Mekruksavanich S, Jitpattanakul A. Biometric user identification based on human activity recognition using wearable sensors: an experiment using deep learning models. Electronics. 2021;10(3):308. doi:10.3390/electronics10030308.
13. Liu Z, Mao H, Wu CY, Feichtenhofer C, Darrell T, Xie S. A ConvNet for the 2020s. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022 Jun 18–24; New Orleans, LA, USA. p. 11976–86.
14. Ji S, Xu W, Yang M, Yu K. 3D convolutional neural networks for human action recognition. IEEE Transact Patt Analy Mach Intellig. 2012;35(1):221–31. doi:10.1109/TPAMI.2012.59.

15. Carreira J, Zisserman A. Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition; 2017 Jul 21–26; Honolulu, HI, USA. p. 6299–308.
16. Tran D, Wang H, Torresani L, Ray J, LeCun Y, Paluri M. A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition; 2018 Jun 18–23; Salt Lake City, UT, USA. p. 6450–9.
17. Xie S, Sun C, Huang J, Tu Z, Murphy K. Rethinking spatiotemporal feature learning: speed-accuracy trade-offs in video classification. In: Proceedings of the 2018 European Conference on Computer Vision (ECCV); 2018 Sep 8–14; Munich, Germany. p. 305–21. doi:10.1007/978-3-030-01267-0_19.
18. Diba A, Fayyaz M, Sharma V, Karami AH, Arzani MM, Yousefzadeh R, et al. Temporal 3D ConvNets: new architecture and transfer learning for video classification. arXiv:1711.08200. 2017.
19. Hussein N, Gavves E, Smeulders AW. Timeception for complex action recognition. In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun 15–20; Long Beach, CA, USA. p. 254–63.
20. Ullah A, Ahmad J, Muhammad K, Sajjad M, Baik SW. Action recognition in video sequences using deep bi-directional LSTM with CNN features. IEEE Access. 2018;6:1155–66. doi:10.1109/ACCESS.2017.2778011.
21. Ahmad T, Wu J, Alwageed HS, Khan F, Khan J, Lee Y. Human activity recognition based on deep-temporal learning using convolution neural networks features and bidirectional gated recurrent unit with features selection. IEEE Access. 2023;11:33148–59. doi:10.1109/ACCESS.2023.3263155.
22. Hussain A, Hussain T, Ullah W, Baik SW. Vision transformer and deep sequence learning for human activity recognition in surveillance videos. Comput Intell Neurosci. 2022;2022(3):3454167. doi:10.1155/2022/3454167.
23. Wensel J, Ullah H, Munir A. ViT-ReT: vision and recurrent transformer neural networks for human activity recognition in videos. IEEE Access. 2023;11:72227–49. doi:10.1109/ACCESS.2023.3293813.
24. Liu J, Luo J, Shah M. Recognizing realistic actions from videos "in the wild". In: 2009 IEEE Conference on Computer Vision and Pattern Recognition; 2009 Jun 20–25; Miami, FL, USA. p. 1996–2003.
25. Reddy KK, Shah M. Recognizing 50 human action categories of web videos. Mach Vis Applicat. 2013;24(5):971–81. doi:10.1007/s00138-012-0450-4.
26. Soomro K, Zamir AR, Shah M. UCF101: a dataset of 101 human actions classes from videos in the wild. arXiv:1212.0402. 2012.
27. Jhuang H, Gall J, Zuffi S, Schmid C, Black MJ. Towards understanding action recognition. In: 2013 International Conference on Computer Vision (ICCV); 2013 Dec 1–8; Sydney, NSW, Australia. p. 3192–9. doi:10.1109/ICCV.2013.396.
28. Tan M, Le Q. EfficientNet: rethinking model scaling for convolutional neural networks. In: 2019 International Conference on Machine Learning; 2019 Jun 10–15; Long Beach, CA, USA. p. 6105–14.
29. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, et al. Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision; 2021 Oct 10–17; Montreal, QC, Canada. p. 10012–22.
30. Werbos PJ. Backpropagation through time: what it does and how to do it. Proc IEEE. 1990;78(10):1550–60. doi:10.1109/5.58337.
31. Hassan N, Miah ASM, Shin J. A deep bidirectional LSTM model enhanced by transfer-learning-based feature extraction for dynamic human activity recognition. Appl Sci. 2024;14(2):603–21. doi:10.3390/app14020603.
32. Cubuk ED, Zoph B, Mane D, Vasudevan V, Le QV. AutoAugment: learning augmentation policies from data. arXiv:1805.09501. 2018.
33. Sutskever I, Martens J, Dahl G, Hinton G. On the importance of initialization and momentum in deep learning. In: ICML'13: Proceedings of the 30th International Conference on International Conference on Machine Learning; 2013 Jun 16–21; Atlanta, GA, USA. p. 1139–47. doi:10.5555/3042817.3043064.
34. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 2818–26. doi:10.1109/CVPR.2016.308.

35. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv:1412.6980. 2014.
36. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: an imperative style, high-performance deep learning library. Adv Neural Inf Process Syst. 2019;32. doi:10.48550/arXiv.1912.01703.
37. Vrskova R, Hudec R, Kamencay P, Sykora P. Human activity classification using the 3DCNN architecture. Appl Sci. 2022;12(2):931. doi:10.3390/app12020931.
38. Jaouedi N, Boujnah N, Bouhlel MS. A new hybrid deep learning model for human action recognition. J King Saud Univ-Comput Inform Sci. 2020;32(4):447–53. doi:10.1016/j.jksuci.2019.09.004.