CHAPTER-3

SEQ2DENSE U-NET: A DENSE SEGMENTATION MODEL FOR H AR

The part of the work is adopted from-

T. Meena and K. Sarawadekar, "Seq2Dense U-Net: Analyzing Sequential Inertial Sensor data for Human Activity Recognition using Dense Segmentation Model," in **IEEE Sensors Journal,** vol. 23, no. 18, pp. 21544-21552, 15 Sept.15, 2023, **DOI: 10.1109/JSEN.2023.3301187**

3.1 Introduction

Human activity recognition (HAR) is the technique of utilizing Artificial Intelligence (AI) to recognize and categorize human actions from unprocessed data gathered from diverse sources. HAR offers a variety of uses due to the extensive adoption of acquisition tools like cameras and cellphones as well as the capability to record data about human activities. Human activity analysis and brain-computer interface both benefit from HAR. The primary task in the activity recognition is the selection and deployment of appropriate sensors to record and analyze the signals [118] . Broadly HAR can be classified into two categories: sensor data and image or video data classification. Activity detection using sensor data is currently a prominent research area among researchers as the use of smart wearable technology in daily life has increased significantly.

Wearable technology incorporating intelligent sensors is rapidly evolving and becoming more affordable, a diverse range of new applications, including sleep state detection [119], health diagnosis [120], biometric recognition [121], gender recognition [122], smart home [123], medical support [124], robotic control [125], and e-learning [126] are evolving. Furthermore, a number of unique issues must be addressed in order to fully realize the potential of such technologies and promote the prevalent use of intelligent wearable devices. According to the theoretical aspect, any sensor that can detect posture could be placed at different body parts for data collection. But this practice will lead to a huge number of sensors mounted on the body which can create inconvenience to the person. So, instead of wearing multiple sensors, people prefer to have a single device with a number of functionalities such as a wireless headphone, smart phone, virtual reality headset, or smart glasses. During daily human activities, the whole human body produces some movements and limited devices

attached to it produces complex signals. Human activities and movements of the human body interfere with each other.

Wearable sensor devices are usually used to gather time series data, human activity recognition utilizing these data is a complex process and performed according to the following steps. Initially the time series data is pre-processed and segmented followed by feature extraction and then finally classifying the activities using classification model. HAR based on classic machine learning (ML) methods require the manual extraction of features. Deep learning (DL) has been used to develop a machine that can learn and extract features without the complex manual process, and it significantly reduces the load of the feature extraction process [13] [127]. As a result, DL-based HAR outperforms traditional ML methods, such as Convolutional Neural Network (CNN), which is commonly utilized to perform both simple and complex tasks.

Since each human activity has a different duration and it might be challenging to pinpoint an action's exact start and end periods, the continuous segmentation of sequence data coming from input sensors is a complex problem. As of now, both ML-based and DL-based HAR methods label the samples into fixed-length windows using labelling strategies. After that, the classification model is used to predict the label for all data in every fixed length window [128]. There are primarily two methods for labelling sliding windows [129]. In the first approach, the most prominent sample label in the frame is used as the frame label. The second approach is selection of the final time stamp in the frame of the sample class as the label of the frame. Since all samples of a frame do not have exactly the same label, a few cases will be misidentified. These methods may result in incorrect labelling which further lowers the recognition accuracy. This is known as the multi-class window problem. It is a very common

challenge in sensor-based activity recognition and has a significant negative impact in classifying short-term activity sequences which makes classification a challenging task. A common method for improving recognition accuracy is the application of a small-size window for the segmentation of time series data. However, this is a time-consuming technique which compromises the speed for achieving an improved accuracy. The Dense labelling technique can be used to directly label and forecast activities based on sampling points, which is one approach to the multi-class window problem [114]. But this method forcibly labels the classes into a particular class and thus leads to misclassification of the activities. Therefore, we adopt a technique of sequential feeding of sensor data to our Seq2Dense U-Net model for the task of HAR from the smart devices.

The simplest approach for solving the multi-class window challenge is by categorizing and recognizing the actions directly on the basis of sequential feeding of the sampling points. Therefore, we use a model which can directly take time series data and predict the human actions according to the acceleration intensities of each action and thus improve the performance of the model. In the conventional CNN model, maximum pooling operations reduces high output resolution and cause a size inconsistency between the input and the final response from the model. Because of this size inconsistency, the existing CNN-based HAR models are only suitable for the sliding window technique prediction and not for the dense prediction. In current era, many new DL models have been developed in the domain of image feature extraction, such as the Mask R-CNN [130], U-Net [131], Full Convolutional Network (FCN) [132], SegNet [133]. These models have the capability of recognizing and classifying image based on pixel-information. These architectures incorporate the up-sampling technique thus achieving the same resolution image as that of the original image. When

compared to other frameworks, the U-Net model uses the up-sampling and down-sampling operations for the pixel level segmentation and classification. It superimposes the features from the up-sampling and down-sampling blocks with the help of skip connections which makes it an efficient model. The superimposition will reinject the lost features of the sensor data at the time of encoding and help in achieving better accuracy. Since time series data can be regarded as a one-dimensional sequential vector, we proposed the Seq2Dense U-Net based on the U-Net model for human activity recognition. The proposed model takes sensor data as the input and classify the activities based on the sampling points, without the use of dense labelling. In this work, we introduce the concept of a Seq2Dense U-Net architecture which is specifically designed for HAR tasks. This architecture combines the benefits of both the U-Net and dense layers, enabling the effective processing of sequential data while preserving spatial information. By employing this architecture, we achieve improved performance in activity recognition compared to traditional approaches. We focus on utilizing data captured from inertial sensors, which are commonly found in wearable devices such as smartphones or smartwatches. These sensors provide valuable information about human motion, making them suitable for HAR tasks. The proposed approach takes advantage of this data source to accurately recognize and classify human activities. This work indeed represents one of the pioneering attempts to apply the Seq2dense U-Net structure to HAR tasks using time sequence data. While previous studies have explored U-Net architectures for HAR, they typically focused on image-based inputs or used other network variations. The proposed work extends the applicability of U-Net models to sequential data and demonstrates their effectiveness in capturing temporal dependencies for activity recognition.

The rest of the chapter is organized as follows: first review of related methods in the context

of the proposed work is presented in Section 3.2. Next, in Section 3.3 the methodology including architecture and in Section 3.4 implementation details are presented. Section 3.5 consists of results and discussion; Section 3.5 consists of network depth analysis. The ablation study is presented in Section 3.6 followed by conclusion is presented in Section 3.7.

3.2 Related Work

Normal human everyday activities are always complicated and ongoing and may entail transitional and aberrant actions. Recent HAR studies emphasize routine activities (e.g., walk, stand, sit, and run). Early HAR research involves manual feature extraction from inertial sensor data. After feature extraction, the features are categorized using a variety of classification approaches. The current HAR ML techniques fall into three categories: discriminative classification, generative model-based classification, and deep learning-based models [134].

3.2.1. Discriminative Model-based Classification

k-Nearest Neighbour (k-NN), Artificial Neural Network (ANN), Support Vector Machine (SVM), Decision Tree (DT), and Convolutional Neural Network (CNN) are the key components of the classification technique of HAR based on discriminative model. Recent HAR research have concentrated on common activities (such as stand, sit, walk, and run). Traditional machine learning techniques were used to differentiate static and dynamic gestures in normal human activity. The authors in [135] concluded that the classification accuracy of HAR may be greatly improved by position-aware multi-sensors (PAMS). In [136], the authors deployed CNN, LSTM, and other deep learning techniques to classify human activities. The accelerometer data's autoregressive coefficients were extracted as the attributes of HAR in [137] and support vector machine was used to recognize the human

activities. Running, standing, jumping, and walking showed good activity recognition performance. The authors in [138] developed a location-independent activity detection model using the Decision Tree algorithm and classified five activities using the built-in accelerometer of the smartphone. A hierarchical system [139] was developed for the classification of 15 activities, where the top part generates the augmented feature vector using the autoregressive method and the bottom part uses Linear Discriminant Analysis (LDA) and ANN to process the eigenvector of the tri-axial accelerometer data.

3.2.2. Generative Model-based Classification

Hidden Markov Model (HMM) and the Naive Bayes method are the two basic classification algorithms based on generative models for HAR. In [140], a dynamic activity recognition approach was proposed using HMM to capture temporal smoothness and regularity. In [141], an HAR model was introduced based on the semi-Markov random domain from accelerometer data, which performed well for challenging activities like eating and operating a vehicle. In [142], Principal Component Analysis (PCA) was utilized to decrease the feature vector's dimension, and a Bayesian classification system was presented to identify physical human activity from a single tri-axial accelerometer data set that was wrapped around the waist.

3.2.1. Deep Learning Models

Deep learning has significantly improved feature extraction from images over the past few years, and it has now been applied to time series analytic study. There are three types of deep learning techniques for HAR. The first type utilizes CNN for HAR and extracts feature automatically from input sensor. One Dimensional CNN [143] was proposed for the behaviour identification of human from the data collected by built-in sensors of

smartphones. The raw data is transformed to reduce the potential rotational interference in the raw data, and used as an input vector, improving the efficiency when compared to the standard random forest method. Panwar M et al. [127] used a hand-mounted tri-axial accelerometer to investigate a deep learning-based approach for the prediction of arm movement in regular activities. CNN is applied for the automatic extraction of valuable features. The mean identification accuracy is significantly high as compared to clustering-based algorithms, Linear Discriminant Analysis, and Support Vector Machine. The second type consists of Recurrent Neural Network (RNN) which capture the time dependency of sensor data. An ensembled HAR model [144] based on the combination of Deep Long Short-Term Memory was proposed which achieved a high accuracy on the Skoda, PAMAP2, and Opportunity datasets. A model [145] based on a Binarized LSTM Network (B-BLSTM-RNN) was proposed that process sensor data from several locations while remaining insensitive to distortions and transformations of the input data. The third type is the implementation of fusion models for activity recognition. Ordóñez et al. [129] provided a general deep learning framework for HAR with the help of convolutional and LSTM recurrent units, that has capability to learn feature representations automatically and modelling the temporal relationships between their activation. On the Skoda and Opportunity datasets, recognition accuracy is 9% higher than in the earlier research. Khan et al. [146] developed a CNN and LSTM hybrid model for the extraction of spatial and temporal features respectively.

In the recent times, an FCN-based human activity recognition algorithm [147] was proposed which achieved dense prediction of HAR from wearable gadgets and thoroughly conducted experiments on three datasets. In this experiment, a weighted F1-measure of

88.7% on the Opportunity Locomotion dataset, 59.6% on the Opportunity Gesture dataset, 89.3% on Subject 1 and 88.3% on Subject 2 on the Hand Gesture dataset, and 79.0% is achieved on the self-obtained dataset from the Hospital. But dense labelling causes the misalignment of some activities as they forcefully label the activities.

In comparison to aforementioned work, the proposed Seq2Dense U-Net predicts the human activities for each sample more precisely from the input time series sensor data.

3.3 Methodology

3.3.1. Seq2Dense U-Net Architecture

U-Net is the expansion and evolution of traditional CNN-based autoencoders which uses redefined skip connections to superimpose the features lost during the pooling operations. The architecture is named as Seq2Dense because the sequential data points are converted into feature vectors and then each learned feature vector is represented as neurons in the neural network which serves as a pipeline for classification using dense-label classification. Seq2Dense U-Net comprises of three major convolutional blocks: the encoder block, decoder block, and classification block. The overall schematic representation of Seq2Dense U-Net is shown in **Figure 3.1.**

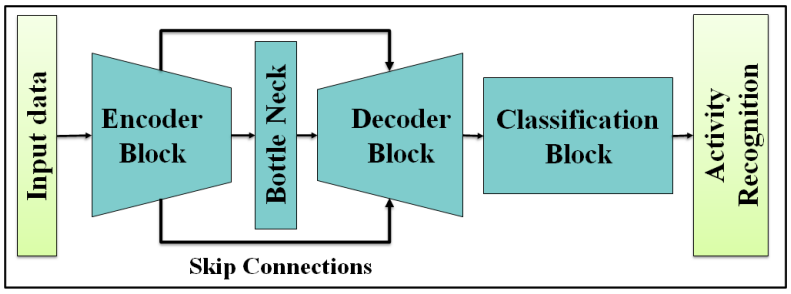

Figure 3.1 Schematic representation of Seq2Dense U-Net.

Convolutional layers are used to learn intrinsic feature vectors from sequential data points. These layers in conjunction with pooling layers constitute the individual units of the encoder

block. The encoder block is responsible for the encoding of input sequence. The bottleneck ensures the smooth flow of feature vectors from the encoding block to the decoding block. The decoding block is responsible for the decoding of feature vectors with the help of up sampling layers and stacked convolutional layers. The skip connections between individual encoder and decoder blocks reintroduces the lost features. The classification block takes the decoded feature vectors as an input. The multi-channel feature vectors are then converted into a one-dimensional vector using flatten() function. This therefore acts as a pipeline for the class-wise activity recognition. The detailed architecture of Seq2Dense U-net is shown in **Figure 3.2**.

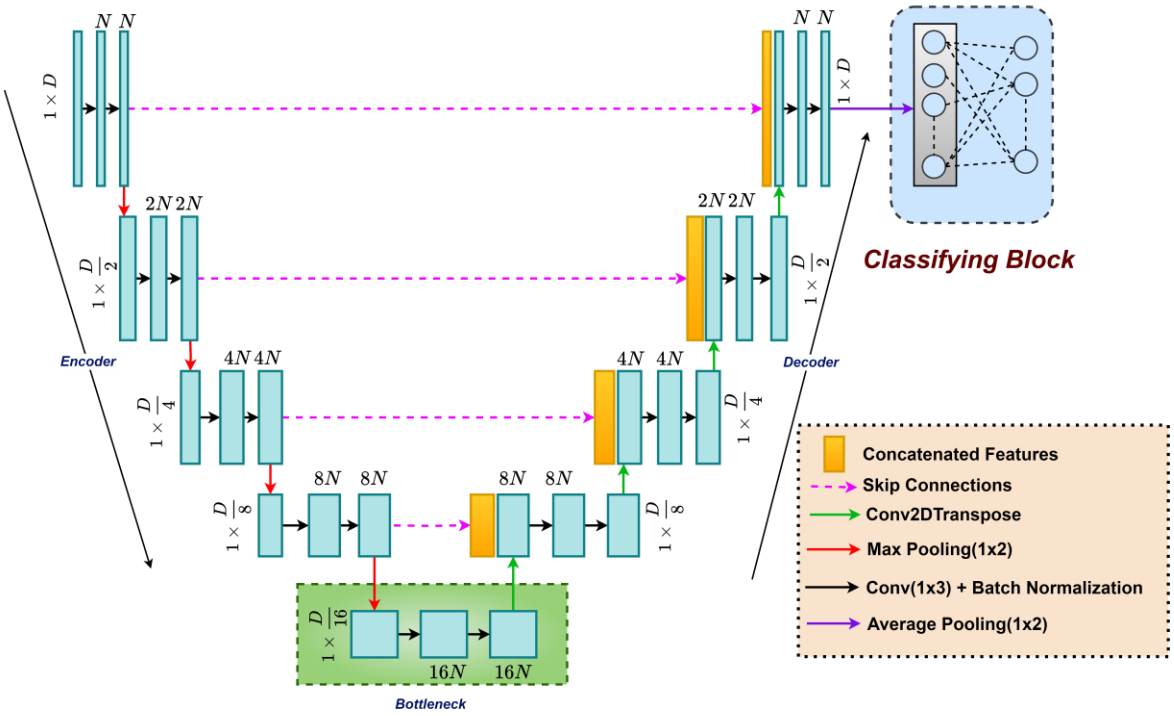

Figure 3.2 In both network blocks, basic convolution operations are performed followed by RELU activation. 1×2 max-pooling operations are performed in the encoding unit for down sampling. Convolution transpose operations are used to up-sample the feature vectors during the decoding phase. The architecture is designed for an input sequence of length (1 × N × 1), where N is the number of sampling points in the input sequence. A feature map is represented by each blue box. The orange boxes represent the feature vectors that are concatenated to the blue up-sampled feature vectors. Operations are represented by arrows.

3.3.1.1. Encoder Block:

The encoder block is a series of four fully connected modules followed by a max-pooling layer. Each module is comprised of two convolutional units. The operation performed on individual neurons n of an input feature vector $v_{input} \in R^{N\times 1}$ for an individual convolutional unit can is represented in algorithm 1 presented in Section 3.5.1. Each convolutional unit comprises of a convolutional layer having a filter size of (1×3) followed by a Batch Normalization layer and RELU as an activation function. The operation is repeated twice. The purpose of the Batch Normalization layer is to normalize the data in the range of (-1,1) to enable faster convergence (in terms of gradients) of the model. The resultant feature vector is passed through a max-pooling layer of filter size (1×2):

$$v_i^{Enc} = \mathrm{f}\left((W_i^{Conv} * D(v_{i-1}^{Enc}) + b_i); \theta_i\right) \ \forall i \in (1,4) \tag{1}$$

$$f(v_i) = \max(v_i, 0) \tag{2}$$

where v_i^{Enc} represents the feature vector learnt during a convolutional module, $W \in R^{1xN}$ represents an N-dimensional weight vector, * represent the convolutional operator, $D(\cdot)$ represents the max-pooling operation, $f(\cdot)$ represents the activation function, b_i represents the bias vector and θ_i represents the set of parameters for an individual convolutional module. The inclusion of θ_i allows for a more detailed representation of parameters of the convolutional module, which includes the parameters related to the activation function, regularization techniques, or any other parameters specific to the module. These parameters are distinct from the weight and bias parameters and contribute to the overall flexibility and expressiveness of the convolutional module. Therefore, while **W** and $\mathbf{b_i}$ are fundamental to the convolutional operation, the parameter set θ_i provides a broader scope for incorporating

module-specific parameters that influence the learning process and contribute to the performance of the model. However, the first encoder module in equation (1) receives v_{input} as an input vector. Therefore,

$$D(v_0^{Enc}) \equiv v_{input} \in \{v_1, v_2, \dots, v_N\} \quad (3)$$

The purpose of the encoder block is to extract features progressively for sensor data to increase the dimensional representation of sensor data. The proposed Seq2Dense U-Net architecture, however, deviates from the U-Net model in the concluding stage.

3.3.1.2. Bottleneck:

The final module of the encoding layer is connected to the bottleneck v_4^{Enc} via max-pooling layer, D. The bottleneck block acts as a link between the encoder and the decoder block. It is responsible for transferring the feature vectors to the decoder block learnt by the encoder block. The bottleneck is a fully convoluted block comprising of two convolutional layers of filter size (1×3). Each convolutional layer is followed by a batch normalization block and RELU activation function. The feature vectors are then passed to the decoder block via a up sampling block of filter size (1×3). The bottleneck can be represented as:

$$v_B = \mathrm{f}((W_B^{Conv} * D(v_4^{Enc}) + b_B); \theta_B) \quad (4)$$

where v_B represents the feature vector learnt in the bottleneck layer, W_B^{Conv} represents the weight vector, b_B represents the biasing vector and θ_B represents the set of parameters for the bottleneck (B).

3.3.1.3. Decoder Block:

The decoder block is a series of four fully connected modules. Each module begins with up sampling of the feature vectors. The resultant feature vectors are then fed to the two successive convolution operations followed by RELU as the activation function. The lost

features are reintroduced by the concatenation of the encoder block v_i^{Enc} with the decoder block v_i^{Dec}. The concluding step of the decoder block consist of three blocks. An individual decoder module can be represented as:

$$v_i^{Dec} = \mathrm{f}\big((W_i^{Conv} * (D(v_i^{Enc})||U(v_{i-1}^{Dec})) + b_i); \theta_i\big) \ , \forall\ i \in (1,4) \tag{5}$$

where U(·) represents the up-sampling layer, || represents the concatenation operation.

The first decoding module receives the feature vector of bottleneck layer v_B. Unlike the concluding stage of U-Net, which consist of a sequence of convolutional layers, the final step in Seq2Dense U-Net is fed to the classification block which comprises of the average pooling layer followed by the flattening for the prediction of various activities.

3.3.1.4. The Classification Block:

The classification block primarily consists of a flatten layer which is used to convert all the extracted feature vectors from a pooling operation into a single M-dimensional linear vector. It is followed by a fully connected dense layer to establish a many to one relation (R) between the learned weight vectors (W) and the classifying neurons represented by a, and followed by the N dimensional classifying vector ($v_{classifier} \in R^{Nx1}$) where N represents the number of classes. A SoftMax activation function is used for the final activity recognition in order to predict a multinomial probability distribution of individual activity class. The categorical cross-entropy loss function is defined as:

$$L(y_{true}, y_{pred}) = -\sum_i y_{true,i} \log\left(y_{pred,i}\right) \tag{6}$$

Here, y_{true} represents the true labels (one-hot encoded), y_{pred}represents the predicted probabilities, and ***i*** iterates over each class. The classification block is mathematically represented as:

$$v_{\text{classifier}} = argmin_{\theta} L\left(\left(\frac{\exp(v^{Dec})}{\sum_{i,j} v_{i,j}^{Dec}}\right); \theta_L\right) \quad (7)$$

where L(·) represents the loss function, θ_L represents the parameter set for the learning process which is updated via backpropagation.

3.4 Implementation Details

3.4.1. Pre-processing

In this section, the pre-processing of the sensor data is discussed. In this work, all the datasets are pre-processed to make it a standard normal distribution. The time series sensor data can be represented as

$X_{H \times W} = \{(x_{1,1}, x_{1,2}, \ldots\ldots x_{1,W}), (x_{2,1}, x_{2,2}, \ldots\ldots x_{2,W}), \ldots\ldots\ldots (x_{H,1}, x_{H,2}, \ldots\ldots x_{H,W})\}$

where H is the length of sensor data and W is the number of samples. When the raw data is not normally distributed, it can affect the performance of the network. The importance of pre-processing the data to achieve a standard normal distribution lies in the fact that raw sensor data often does not follow a normal distribution, which negatively affect the performance of model. Non-normally distributed data lead to a model that is less accurate, and are harder to train due to issues like gradient vanishing or exploding. Therefore, standardization of the data is crucial. To make the data standard, we performed standard scalar operation followed by the normalisation as shown in (7).

$$X_{ij} = \frac{\left(x_{ij}^{input} - \mu\right)}{s} \quad \forall\, i \in [0, H), \quad j \in [0, W) \quad (7)$$

where *x* is the feature vector in the time series data, μ represent the mean value of all the feature vectors and s is the standard deviation of all feature vectors.

Since our task is supervised learning, the categorical data labels represented by $Y_{NUM} = \{Y_1, Y_2, \ldots\ldots\ldots, Y_i\}$

where Y_i represents the i^{th} activity class of the sample are first encoded. Since we are dealing with multi-class classifications, the labels are one-hot encoded. The pre-processing techniques are precisely mentioned in algorithm 1.

ALGORITHM 1: DATA PRE-PROCESSING

1. Input: csv file with feature attributes.
2. Output: Data frames consisting of linear sequence of attributes and one hot encoded value as labels.
3. For i ← 0 to M
    (1) $X_{linear} = [X_{N \times i}, X_{N,\ldots\ldots} X_{N(i+1)-1}]$
    (2) For j ← 0 to N
    (3) $Z_{linear} = (X_{linear} - \mu) / \sigma$
    (4) $N_{linear} = (Z_{linear} - (Z_{linear})_{max}) / ((Z_{linear})_{max} - (Z_{linear})_{min})$
    (5) Append N_{linear} in X_{test}
4. Label encoding followed by one-hot encoding technique.

3.4.2. Implementation and Training

Learning rate, batch size, and number of epochs are parameters in the training process. Batch size is set to 32 in order to expedite the training process and use less memory. The learning rate is set at 0.0001 since a higher learning rate could cause the model to diverge. In order to optimise epochs and prevent over-fitting, the stop early approach is also used. This method stops the training process when the training loss reduces over the course of 10 successive epochs while the validation loss increases. The number of the filter for the convolutional layer is set as 16, 32, 64, 128 and 256 from the upper side to the lower side of the Seq2Dense U-Net. The runtime for each epoch is 3, 2, and 1 seconds for UCI-HAR, UCI-HAPT and Sanitation dataset, respectively. The time for predicting the activity is approximately 0.16

seconds. The batch normalization is used in the overall network to reduce the overall covariance shift and make the architecture more stable at the time of training. The details of the hyperparameters of Seq2Dense U-Net are given in **Table 3.1.**

Table 3.1 Hyperparameters for the Seq2Dense U-Net

Parameters	Encoder Block	Bottleneck	Decoder Block	Classification Block
No. of Conv. Layers	8	2	8	0
No. of Conv. Filter	16,32,64,128	256	16,32,64,128	0
No. of Conv. transpose Layers	-	1	3	0
kernel Size	1×3	1×3	1×3	0
Strides	-	1×2	1×2	0
Kernal Initializer	he_normal	he_normal	he_normal	Glorot_uniform
Activation	RELU	RELU	RELU	SOFTMAX
No. of hidden neurons	-	-	-	256
No. of neurons (Classes)	-	-	-	N
Loss function	-	-	-	Categorical Cross Entropy

3.4.3. System Specification

The training and testing of the proposed Seq2Dense U-Net model is carried out on the system with the configuration as Intel RTX A4000 chip and Intel i9 processor. The system runs on a Windows 11 operating system with 64 bits. The development of the proposed model includes the following libraries - TensorFlow 2.9.0, Python 3.9, Keras 2.6.0 and Scikit-learn 1.1.1. The Adam optimizer is used to train the model. The loss function used is Categorical_Cross_Entropy.

3.5 Result and Discussion

3.5.1. Results

The performance of all the three datasets is summarised in **Table 3.2**. The performance of each activity for all the three datasets is evaluated in terms of precision, recall, and F1 score. From **Table 3.2** it is clearly evident that the proposed model has achieved a comparable high score of in terms of accuracy, precision, recall F1, MCC, and Kappa score.

Table 3.2 Performance of the proposed Seq2dense U-Net on all three dataset

Metrics	**Sanitation**	**UCI-HAPT**	**UCI-HAR**	**mhealth**
Accuracy	0.9095	0.9474	0.9547	0.9011
Precision	0.9031	0.9393	0.9414	0.9121
Recall	0.9000	0.9383	0.9406	0.9011
F1 score	0.9011	0.9379	0.9406	0.8928
MCC	0.8884	0.9274	0.9487	0.8950
Kappa Score	0.8883	0.9272	0.9486	0.8918

For the sanitation dataset, the proposed model gives an accuracy score of 0.9095, precision of 0.9031, recall of 0.9000, F1 score as 0.9011 followed by MCC and kappa score as 0.8884 and 0.8883, respectively. **Figure 3.3 [A]** represents the performance score for each class of sanitation dataset. It is clearly visible that the proposed model is able to correctly predict the similar activities which is a very difficult task.

In the case of UCI-HAPT dataset, the proposed model also achieves an accuracy score of 0.9474, precision of 0.9393, recall of 0.9383, F1 score as 0.9379 followed by MCC and Kappa score as 0.9274 and 0.9272, respectively. **Figure 3.3 [B]** represents the performance score for each class of UCI-HAPT dataset. It is clearly seen from **Figure 3.3 [B]** that the proposed model is able to distinguish between the static activities i.e., sitting standing laying and dynamic activities i.e., walking upstairs and walking downstairs as well as the transition

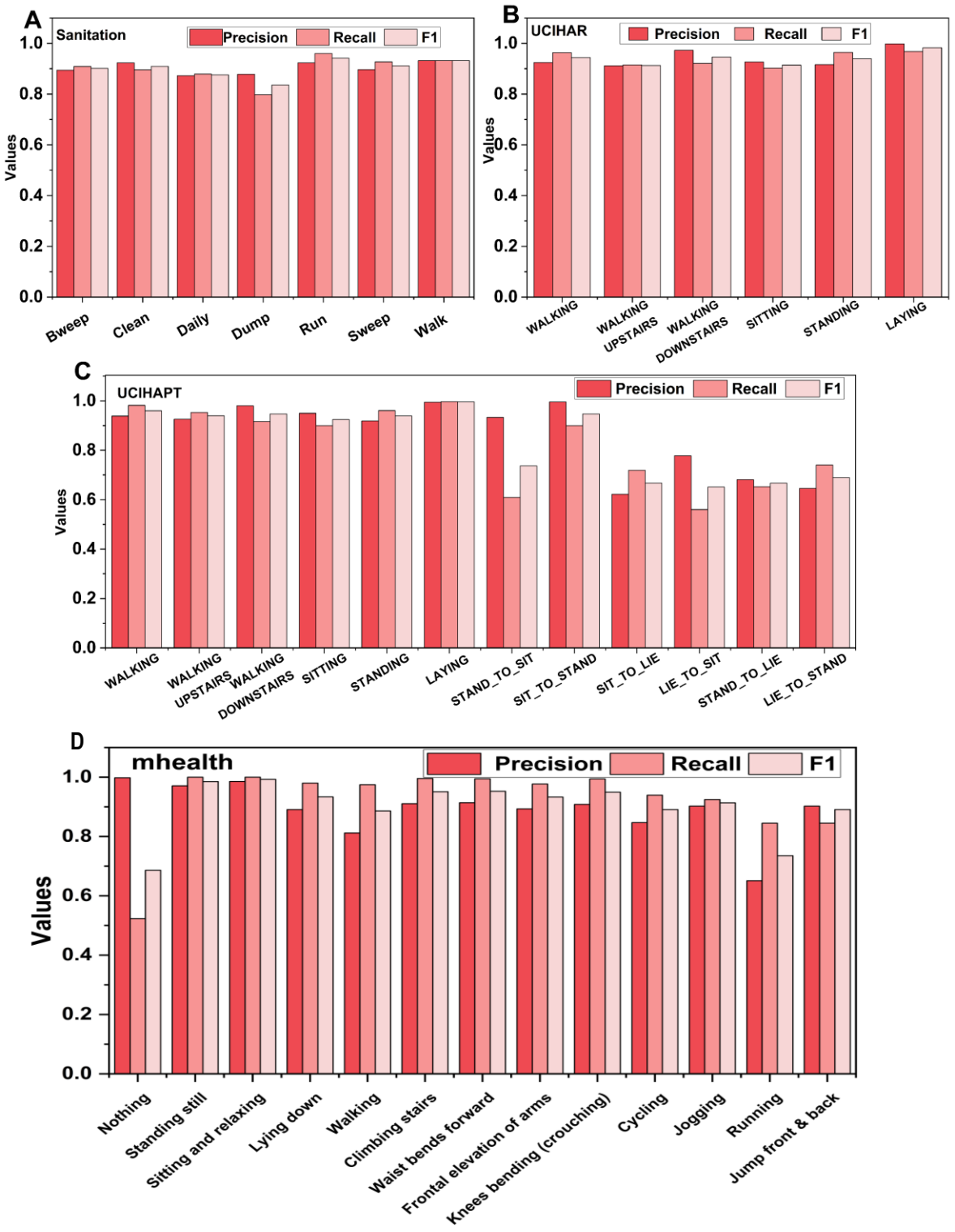

Figure 3.3 Performance of Seq2Dense U-Net on [A] Sanitation dataset [B] UCI-HAR Dataset [C] UCI-HAPT Dataset [D] mhealth Dataset.

activities which are stand_to_sit, sit_to_stand, sit_to_lie, lie_to_sit, stand_to_lie, lie_to_stand very precisely.

For the UCI-HAR dataset, the proposed model gives an accuracy score of 0.9547, precision of 0.9414, recall of 0.9406, F1 score as 0.9406 followed by MCC and Kappa score as 0.9487 and 0.9486, respectively. **Figure 3.3 [C]** presents the performance score for each class of UCI-HAR. We can clearly see that the proposed model is able to correctly predict the similar activities which is a very difficult task.

For the mhealth dataset, the proposed model gives an accuracy score of 0.9011, precision of 0.9121, recall of 0.9011, F1 score as 0.8928 followed by MCC and Kappa score as 0.8950 and 0.8918, respectively. **Figure 3.3 [D]** presents the performance score for each class of mhealth. We can clearly see that the proposed model is able to correctly predict the similar activities which is a very difficult task. These results confirm that the proposed Seq2Dense U-Net is highly capable of classifying different activities.

3.5.2. Comparison with other models

In this section, the performance of the proposed model is compared with the discriminative machine learning and deep learning models. We used accuracy, F1-score, MCC, Kappa and test inference time to compare discriminative machine learning and deep learning models on UCI HAR dataset. **Table 3.3** presents the results of this comparative analysis. The discriminative machine learning models are mainly SVM with linear and radial basil function RBF kernel, XGboost, Decision Tree. The deep learning models are CNN, LSTM, Convolutional LSTM (Conv LSTM), Fully Convolutional Network (FCN), and U-Net. The accuracy of SVM, XGboost, Decision Tree and Random Forest is 0.906, 0.911, 0.938,

Table 3.3 Comparative analysis of Different models with UCI-HAR dataset.

Model			Accuracy	F1	MCC	Kappa	Test Inference time
Discriminative Machine Learning	SVM	RBF	0.906	0.905	0.905	0.905	1.895
		Linear	0.911	0.911	0.913	0.913	0.221
	XGboost		0.938	0.938	0.926	0.926	0.018
	Decision Tree		0.859	0.858	0.831	0.831	**0.007**
	Random Forest		0.876	0.872	0.845	0.842	0.480
Deep Learning	CNN		0.923	0.923	0.894	0.893	0.695
	LSTM		0.913	0.912	0.897	0.895	1.723
	ConvLSTM		0.750	0.741	0.702	0.700	34.30
	FCN		0.907	0.904	0.890	0.888	0.615
	U-Net		0.934	0.937	0.907	0.905	0.272
	Seq2Dense U-NET		**0.954**	**0.940**	**0.948**	**0.948**	0.167

0.859 and 0.876, respectively. The corresponding F1 scores are 0.905, 0.911, 0.938, 0.858, and 0.872, respectively. The performance of XGboost is high because it can handle complex, high-dimensional datasets efficiently and automatically capture feature interactions which results in better performance. It incorporates regularization techniques to prevent overfitting. In comparison, SVM may struggle with large datasets and requires careful hyperparameter tuning. Decision trees are prone to overfitting which results in lower performance. However, the test inference time for decision tree is the least because it involves simple calculations based on splitting criteria, such as Gini impurity or information gain. They do not require iterative optimization processes like gradient descent, and weight updates are not as complex as in other models.

The accuracy of the CNN, LSTM, ConvLSTM, FCN, U-Net are 0.923, 0.913, 0.750, 0.907, and 0.934, respectively. The corresponding F1 score are 0.923, 0.912, 0.741, 0.904, and 0.937, respectively. Sensor data consists of spatial and temporal information which is a

challenging task for CNN. LSTMs can capture temporal information but are not able to effectively leverage the spatial information present in sensor data. The results of all the aforesaid methods are compared and the outcomes are presented in terms of the bar chart and confusion matrix in **Figure 3.4** and **Figure 3.5**, respectively.

The ConvLSTMs combine the strengths of CNN and LSTM have difficulty separating the spatial and temporal features in sensor data thus performing the worst. FCNs may have difficulty in capturing fine details and boundaries in sensor data, limiting their ability to analyze subtle variations. The output of the U-Net is a 3D feature vector which represents the learnt spatial feature vectors. Converting the 3D feature vector to a dense fully connected

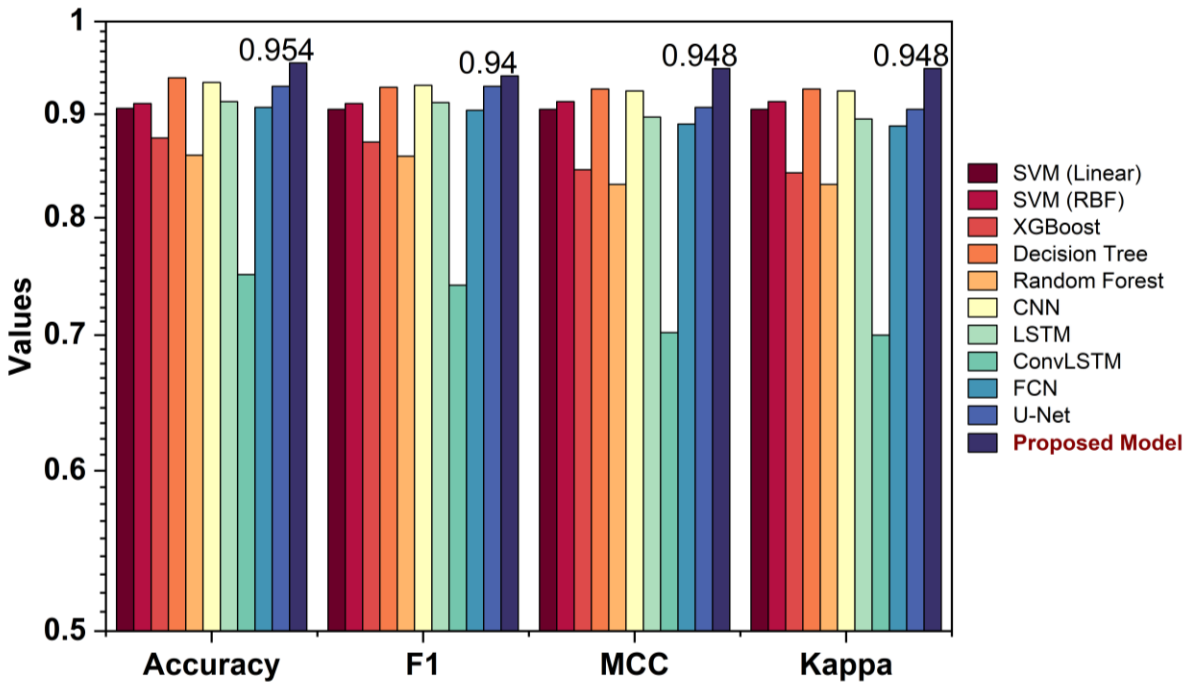

Figure 3.4 Performance of other models on UCI HAR dataset.

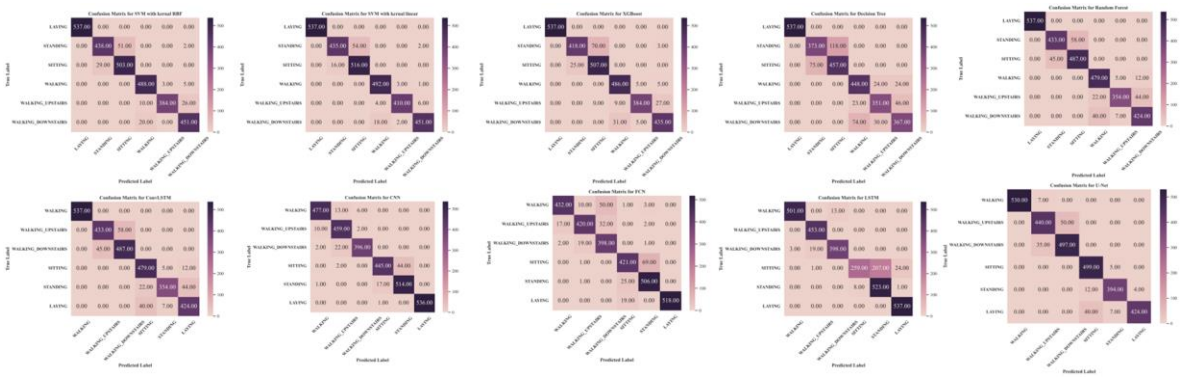

Figure 3.5 Confusion matrix of other models on UCI-HAR dataset.

layer results in a significantly reduced feature vector which results in significant information loss. However, the proposed Seq2Dense U-Net involves the 1D input data which results in less information loss resulting in high accuracy.

3.6 Network Depth Analysis

In order to evaluate the effect of depth of Seq2Dense U-Net, we performed several experiments on all three datasets, in which the model hyperparameters were kept the same as mentioned in the implementation section. We considered model depth as the number of convolutional modules in the encoder and decoder blocks. We examined the performance of the proposed model by varying the depth of the model to validate the optimal depth for the proposed model in terms of F1-score. The results of the analysis are shown in **Figure 3.6**. For all the dataset the F1-score increases as the depth increases. The model with a depth of five reaches the maximum F1-score, thus validating our claim. With further increase in the depth, the F1-score starts dropping because as the network depth increases the number of convolutional filters also increases which results in the increase of the number of trainable parameters as shown in **Table 3.4**. As a result, the complexity of the model increases

resulting in overfitting of the model. Thus, we can say that 5 is the optimal number of depth for the proposed model.

Table 3.4 Number of Parameters for different depth.

Depth	Total	Trainable	Non trainable
2	26,423	26,295	128
3	65,415	65,095	320
4	191,527	190,823	704
5	**685,415**	**683,943**	**1,472**
6	2,721,895	2,718,887	3,008
7	10,859,879	10,853,799	6,080

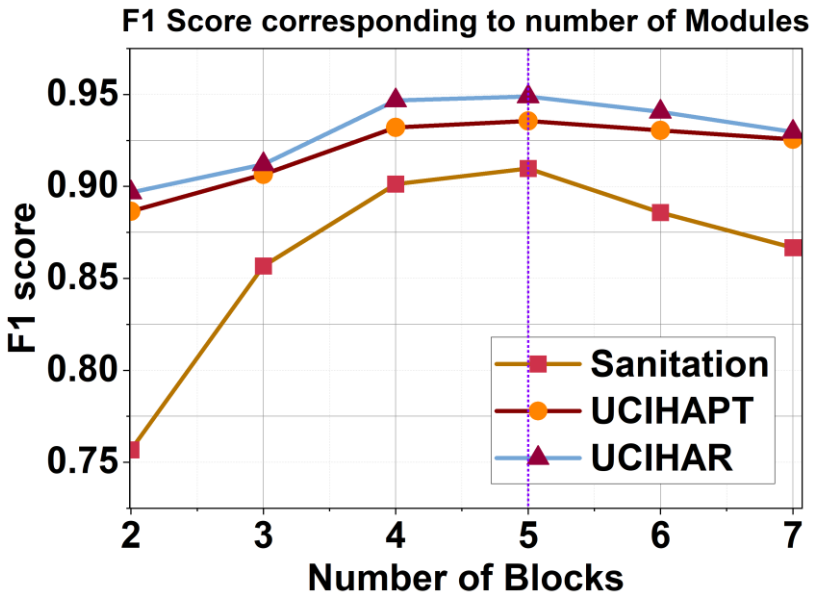

Figure 3.6 F1 score corresponding to the number of blocks in the proposed model.

3.7 Ablation Study

In this section, we detail the need of the classifying block as a combination of average pooling

followed by a fully connected dense layer in the proposed model. To explain this, we implement different LSTMs in the classification block namely: LSTM, Bidirectional LSTM (BiLSTM), Stacked LSTM and Stacked Bidirectional LSTM. The combinations are (i) U-Net+LSTM (ii) U-Net+biLSTM (iii) U-Net+Stacked_LSTM, and (iv) U-Net+Stacked_biLSTM.

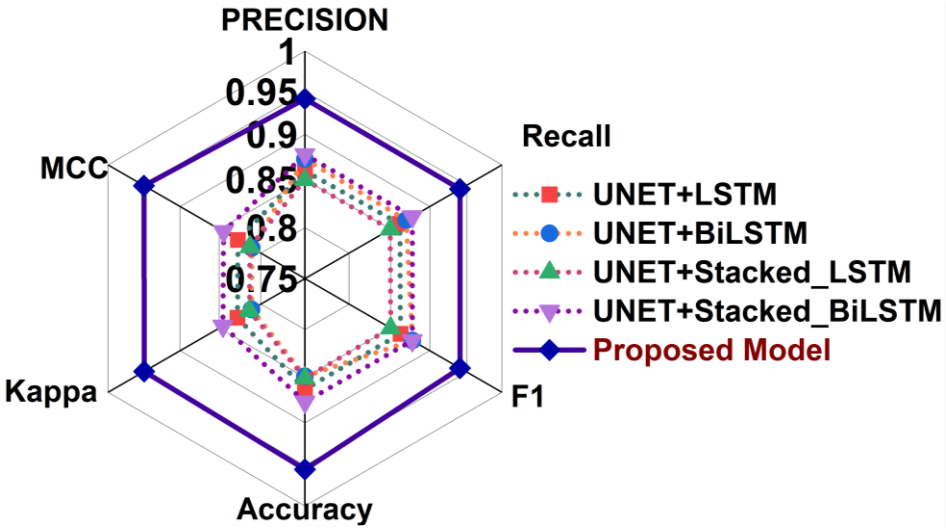

Figure 3.7 Accuracy metrices for the different models used in ablation study.

We have also varied the number of parameters such as the number of hidden layers (L) and the number of hidden neurons in each layer represented as N in **Table 3.5** with the best performing model represented in blue. Combinations such as 'Unet+LSTM' and 'U-Net+biLSTM' have been experimented with a single hidden layer (L=1) with N varying from 8 to 64. Stacked versions of these combinations: 'U-Net+Stacked_LSTM' and 'U-Net + Stacked_biLSTM' involve experiments with varying hidden layers as well as varying hidden neurons in individual layers. We plotted the Figure 6 with these best performing models. The performance of these models were compared on the basis of accuracy, F1 score, MCC and kappa score. The details of the performed experiments are illustrated in **Table 3.5**.

From **Table 3.5** we can clearly say that the accuracy is in the range of 79% to 89%, F1 score is in the range of 78% to 89%, MCC and Kappa are in the range of 76% to 86% which is very less as compared to the proposed model. LSTMs have a tendency to overfit and sensitive to random weight initialization. Since pretrained weights are not used in this study, LSTMs and its variants tend to underperform for the given dataset. The failure of bidirectional LSTM to generate better outputs than its counterparts is because the datapoints are not strictly a time series dataset. Therefore, the LSTMs and its variants are not performing as better as the proposed model.

Table 3.5 Accuracy, F1 score, MCC and Kappa for different experiments Performed on Classifying Block

Model	L	N	F1	Accuracy	MCC	Kappa
U-Net + LSTM	1	8	0.8372	0.8474	0.8046	0.8041
		16	**0.8624**	**0.8627**	**0.8277**	**0.8272**
		32	0.8058	0.8057	0.758	0.7557
		64	0.8499	0.8501	0.8128	0.8125
U-Net +Stacked_LSTM	2	**8**	**0.8500**	**0.8507**	**0.8134**	**0.8128**
		16	0.8123	0.8123	0.7648	0.7648
		32	0.8446	0.8435	0.8067	0.8056
		64	0.846	0.8441	0.8081	0.8065
	3	8	0.7828	0.7904	0.7379	0.7367
		16	0.822	0.8216	0.779	0.7778
		32	**0.8491**	**0.8488**	**0.8108**	**0.8105**
		64	0.8409	0.8395	0.8005	0.7996
U-Net + biLSTM	1	8	0.8733	0.873	0.8407	0.8403
		16	0.8587	0.8594	0.8233	0.8228
		32	**0.8774**	**0.8766**	**0.8455**	**0.8448**
		64	0.8386	0.8368	0.7991	0.7976
U-Net+Stacked_biLSTM	2	8	0.8551	0.8547	0.8184	0.8183
		16	**0.8738**	**0.874**	**0.8418**	**0.8409**
		32	0.8627	0.8627	0.8276	0.8275
		64	0.8653	0.866	0.832	0.8318
	3	8	0.859	0.8594	0.8238	0.8233
		16	0.8545	0.8547	0.8178	0.8171
		32	**0.8938**	**0.8934**	**0.8669**	**0.8667**
		64	0.8395	0.8594	0.7989	0.7977

3.8 Concluding Remarks

HAR from sensor data is an attractive topic of research because of its applications in the areas of IoT and healthcare. In this frame of reference, many studies proposed remarkable results using the accelerometer and gyroscope data for classifying different activities. These studies either use sliding windows or dense labelling techniques to recognize the human activities corresponding to the signal acquired through the smart devices. These approaches face the challenge of a multi-class window that incorrectly labels different classes of sampling points within a window as a class. The proposed Seq2Dense U-Net for human activity recognition differs from the existing ML and DL-based methods and efficiently classify different activities from the sensor-based time series data. The results show that the proposed model performs better than the existing methods. We have optimized the network depth to determine the optimal number of modules in the network. Further, we have evaluated the proposed model based on the various evaluation metrics and subsequently performed the ablation study to justify the performance and effectiveness of the proposed method. The model proposed in this paper can also classify short-duration activities with improved accuracy.

The chapter is summarized as follows:

- The concept of classifying block in the existing U-Net architecture for the task of activity recognition.
- A Comparative study with SOTA models on the latest metrices for extensive evaluation of the model has been presented.
- The depth analysis of the network is performed to find the optimum depth of the architecture.