Grenze International Journal of Engineering and Technology, June Issue

Human Activity Recognition using BiLSTM and Comparison with Transformer Model

Vivek Shukla[1] and Dr. Satya Prakash Sahu[2]
[1-2]National Institute of Technology/Information Technology, Raipur Chhattisgarh, India
Email: viveks2992@gmail.com, spsahu.it@nitrr.ac.in

***Abstract*— The categorization of human activities using temporal sequence information is attracting interest in deep learning techniques like Recurrent Neural Networks. RNN technique called Long Short Term Memory works well for classifying time series data as it can effectively manage long term dependency and vanishing gradient issues. In this work, we assess the BiLSTM model performance in identifying human activities and comparing it with transformer model. The results show that while the bidirectional approach slows down with increasing dataset size, it improves recognition quality slightly over the LSTM method. In contrast, the transformer model, while not enhancing the accuracy much, can shorten running times when compared to BiLSTM.**

***Index Terms*— Recurrent Neural Network (RNN), Long Short Term Memory (LSTM), Bidirectional Long-Short Term Memory (BiLSTM), Transformer model, Human Activity Recognition (HAR)**

I. INTRODUCTION

In the realm of computer vision and artificial intelligence, the ability to accurately estimate the poses of human beings from images or video footage has profound implications across various domains. Human pose estimation, a foundational task within this field, involves the precise localization of key body joints or landmarks, enabling the reconstruction of human body configurations in two or three dimensions. Its applications span a wide spectrum, from enhancing human-computer interaction and revolutionizing sports analysis to facilitating advancements in healthcare, robotics, and beyond. At the heart of this transformative capability lies the fusion of cutting-edge deep learning techniques with an insatiable quest to comprehend the intricate language of the human body.

Our work uses deep learning to drive its exploration of the dynamic field of human position estimation. The landscape, once dominated by hand-crafted features and heuristic algorithms, has witnessed a paradigm shift with the advent of convolutional neural networks (CNNs), RNNs and its variants. These neural architectures, known for their prowess in capturing complex spatial and temporal patterns, have elevated the accuracy and robustness of human pose estimation to unprecedented levels and the application of attention mechanism from the transformer model which helps in parallel processing of input has increased the overall reliability for HAR and also makes the processing faster. This evolution has ushered in an era of more accurate, versatile, and adaptable pose estimation models, with the potential to revolutionize human machine interaction.

II. LITERATURE STUDIES

Many research studies use deep learning model to determine human pose estimation. [1] compare GRUs to

Grenze ID: 01.GIJET.10.2.548_3

traditional RNNs and LSTMs to determine whether the gating mechanisms in GRUs provide advantages in terms of modeling sequential data. [2] makes use of convLSTM, a long short-term memory network with an incorporated convolutional network that enables it to recognize spatial elements in data while also taking the temporal relationship into consideration. [3] uses Long term Recurrent Convolutional Network approach that combines CNN and LSTM into a single layer. [4] human activity recognition using a Coarse-Fine Convolutional Deep-Learning Strategy. [5] Comparing Bidirectional and Unidirectional LSTM Networks for the Human Activity Recognition. [6] proposes a real-time recognition scheme that integrates readings from MEMS (Micro-Electro-Mechanical System) sensors with deep learning algorithms (CNN and LSTM). [7] presents a novel approach that uses CNN and BiLSTM with different kernel dimensions to collect information at different resolutions. This research successfully chooses the best possible video representation and uses BiLSTM and conventional CNN to effectively extract spatial and temporal information from sensor data. [8] In this research, a unique method for skeletal data-based human activity recognition is proposed. The approach blends supervised and unsupervised learning algorithms to deliver real-time performance and qualitative outcomes. The suggested approach uses a two-step framework: in the first stage, activities are grouped according to their similarity using an unsupervised clustering technique, and in the second stage, graph convolutional networks are used to classify the data that has been given to each group. [9] In this study, four models of deep learning are introduced: a CNN with a Gated Recurrent Unit (GRU); another CNN with a GRU and attention; a CNN with a GRU and another CNN; and a CNN with an LSTM and another CNN. These models were trained with CSI amplitude data obtained by a CSI tool in order to perform Human Activity Recognition (HAR). When these models were tested for efficacy against other contemporary techniques, the findings were superior. [10] This article used an Android application for the accelerometer, gyroscope, and linear acceleration to collect raw data from a smartphone sensor. (called H-Activity). In addition, a hybrid deep learning model that is CNN-LSTM is suggested. This model is enabled by the self-attention algorithm, which improves the system's predicting skills. [11] provide a dilated convolutional neural network (DCNN) based bi-directional long short-term memory (BiLSTM) based attention mechanism that distinguishes between various human behaviors in the videos by selectively focusing on useful elements in the input frame. In this work, [12] a generic HAR framework based on Long Short-Term Memory (LSTM) networks for time-series domains is provided for sensor data from smartphones. A comparison analysis is conducted on four baseline LSTM networks to examine the effects of using various types of smartphone sensor data. Furthermore, to enhance recognition performance, a 4-layer CNN-LSTM hybrid LSTM network is suggested.

III. Issues And Challenges

By going through the above literatures, we found that the combination of CNN and LSTM as a single model works well in determining human activities and further use of Bidirectional LSTM marginally improves the accuracy compare to unidirectional LSTM but it takes more time to train. BiLSTM consume more training time as the size of the dataset increases.

Hence, we have applied attention mechanism used in transformer neural network in place of BiLSTM and compare both the models separately and found the uses of multihead mechanism reduces the training time along with marginally increasing the accuracy.

IV. Proposed Methodology

The proposed method for human activity architecture uses two models: CNN followed by BiLSTM and CNN followed by Transformer multi head attention mechanism. In the study, we use the UCI101 dataset, which has collected data from mobile sensors.

A. Model Architecture

a. CNN Layer: The pre-processed data is first fed into a 16-layer Convolution layer which is then followed by dropout layer of 25% and Max-pool layer. Then two more CNN layer of size 32 and 64 respectively is used with same dropout layer and maxpool layer as discussed in previous layer. Convolutional layers process input images by applying filters and extracting features at various spatial hierarchies. To save computation and memory, pooling layers down sample the characteristics.

b. Transformer Method: The output of CNN layer is then fed into multihead Attention layer with number of attention heads in the multi-head attention mechanism is 4 and dimention of the key size is 512 keys. Attention layer followed by layer normalization that normalize sum the multi-head input and the output of the multi-head attention layer.

c. BiLSTM Method: The CNN layer output is then input into a single layer BiLSTM with 64 neurons, commonly utilized for extracting temporal information.
The model's output is then obtained by using an output layer (dense layer with softmax classifier) after activation is accomplished through the use of sigmoid activation. The Adam optimizer is best suited for this, along with the categorical crossentropy loss function is used. Model architecture shown in Fig. 1.
Unlike BiLSTM, which must process input sequentially due to their recurrent nature, transformers can handle input sequences in parallel. Because every token in the sequence may be processed simultaneously, this results in faster training time. Transformers can handle larger sequences without appreciably increasing calculation time, but RNNs are constrained by the length of sequences they can process because of the sequential nature of computing. Hence, Transformer also slightly improves the accuracy when compared to BiLSTM.

B. Long Short-Term Memory (LSTM):

RNNs are widely used for speech detection [13] and text generation [14]. However, training a network with a long series of dynamic data can be challenging due to the vanishing gradient problem. To overcome this issue, advanced RNN variants known as Long Short-Term Memory (LSTM) are employed [15] shown in Fig. 2. By explicitly enabling the network decide when to update hidden layer states with new data and when to forget about past hidden layer states, LSTM solves the vanishing gradient problem.
LSTM consists of two main components: the hidden states h_t and the cell states c_t. The cell state acts as memory, while the hidden state carries information from previous time steps and serves to make predictions or transfer information to the next step. LSTM is equipped with three gates: the output gate o_t the input gate i_t, the forget gate f_t and Formulas given by [15] for calculation of all the gates are discussed below:

$$\text{Input gate: } i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (1)$$

$$\text{Forget gate: } f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \quad (2)$$

$$\text{Candidate update: } \widetilde{c_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (3)$$

$$\text{Cell state updation: } c_t = f_t \cdot c_{t-1} + i_t \cdot \widetilde{c_t} \quad (4)$$

$$\text{Output gate: } o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$\text{Hidden state updation: } h_t = o_t \cdot \tanh(c_t) \quad (6)$$

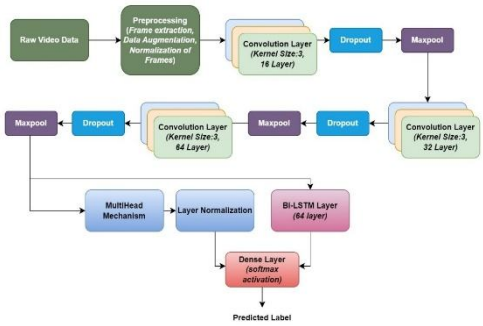

Fig. 1 Model Architecture

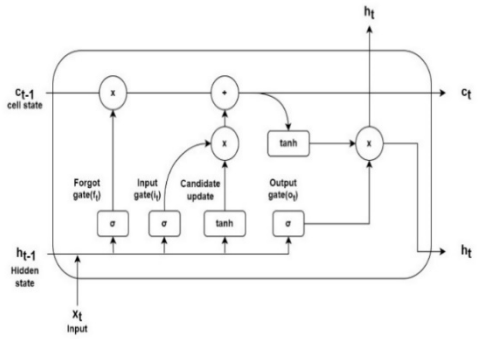

Fig. 2 LSTM Cell Structure

The LSTM unit uses the sigmoid nonlinearity (σ) for the gates and the hyperbolic tangent non-linearity *tanh* for processing.

C. Bidirectional Long Short-Term Memory (BiLSTM)

We have used an enhanced version of the conventional LSTM, that is the BiLSTM network [16]. BiLSTM network analyses input data in both forward and backward directions by gathering information from each time step's past and future context. The network's understanding and modelling of *h* temporal dependencies are enhanced by this bidirectional processing. BiLSTM layer structure shown in Fig. 3.
a. Forward Pass: Through a series of LSTM cells, the input sequence is processed forward. Based on the input at the current time step x_t and the preceding hidden state $\vec{h}_{t-1}$ and cell state $\vec{c}_{t-1}$, each LSTM cell computes the

hidden state $\vec{h}_T$ and cell state $\vec{c}_t$. The sequence of hidden states $[\vec{h}_1, \vec{h}_2, \dots, \vec{h}_T]$, is the result of the forward LSTM cells, where *T* is the input sequence's length.

b. Backward Pass: A different set of LSTM cells processes the input sequence backward. Based on the data at the current timestep x_t, these cells calculate the next hidden state $\overleftarrow{h}_{t-1}$ and cell state $\overleftarrow{c}_{t-1}$, as well as a backward hidden state $\overleftarrow{h}_T$ and backward cell state $\overleftarrow{c}_t$. A series of backward hidden states $[\overleftarrow{h}_1, \overleftarrow{h}_2, \dots, \overleftarrow{h}_T]$ is the result of the backward LSTM cells.

c. Output Concatenation: The final output of the Bidirectional LSTM is obtained by concatenating the forward and backward hidden states at each time step. At time step *t*, the bidirectional hidden state is found mathematically by:

$$h_t^{bidirectional} = [\overleftarrow{h_t}, \overrightarrow{h_t}] \quad (7)$$

The bidirectional hidden states can be used directly for tasks like sequence-to-sequence, regression, or classification, or they can be passed through several layers. The information gathered from the forward and backward passes is used to compute the final output.

Backpropagation through time *(BPTT)* is used to update the model parameters (weights and biases) during training. The estimated output and the ground truth are used to calculate the loss, which is then minimized by optimizing the model.

D. Transformer Model

RNNs and LSTM are best suited for the sequence task but these model have limitations in parallel processing of data and still suffers from vanishing gradient problem while handle long sequences of data. We implement transformer attention mechanism to overcome this limitation.

Transformer is a deep learning model architecture and was first described in the 2017 research "Attention is All You Need" by [17]. Transformers can effectively capture long-range dependencies in data, hence it becomes an essential building component for many machine translation task, natural language processing (NLP) and other sequence-to-sequence tasks.

Components of Transformer that we have used for our model are as follows:

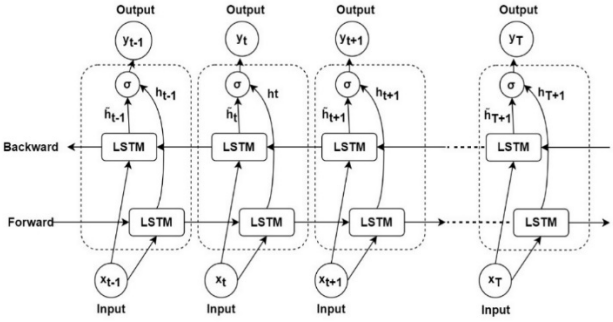

Fig. 3 BiLSTM Layer Structure

a. Self-Attention Mechanism: The transformer architecture's self-attention mechanism is a crucial part. It permits the model to process each element while concentrating on distinct segments of the input sequence. The self-attention mechanism modifies the corresponding output sequence positions by comparing all members of the input sequence with one another. To calculate a representation of a given sequence, it connects various points within the sequence.

It computes a set of attention scores for each element in the sequence in relation to all other elements given a sequence of input vectors, also known as embedding (Fig. 4). A compatibility function, which is typically a dot product, is used to calculate the attention score for the element at position *i* in relation to the element at position *j*.

$$\text{Attention}(Q_i, K_j) = \frac{Q_i \cdot K_j}{\sqrt{d_k}} \quad (8)$$

Where Q_i, K_j are the query and key vectors and d_k denotes the dimensionality of the key vectors respectively. The softmax function is then used to normalise these attention scores, resulting in attention weights:

$$\text{AttentionWeight}(Q_i, K) = \frac{\exp(\text{Attention}(Q_i, K))}{\sum_j \exp(\text{Attention}(Q_i, K_j))} \quad (9)$$

Each position's final output is the weighted sum of the values *(V)*:

$$\text{AttentionOutput}_i = \sum_j \text{AttentionWeight}(Q_i, K_j) \cdot V_j \quad (10)$$

b. Multi-Head Attention: The Transformer enables the model to pay attention to various elements of the input sequence by using multiple self-attention heads in parallel as shown in Fig. 5. Learnable weight matrices for the query, key, and value projections are unique to each attention head. Each head's output is concatenated, then linearly transformed.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \cdot W_O \quad (11)$$

$$\text{Where, head}_i = \text{Attention}(Q \cdot W_{Qi}, K \cdot W_{Ki}, V \cdot W_{Vi})$$

c. Feedforward Neural Network: Each of the attention output is routed via a feedforward network that is fully connected based on position. This is comprised of a *ReLU* activation wrapped between two linear transformations.

$$\text{FFN}(x) = \text{ReLU}(x \cdot W_1 + b_1) \cdot W_2 + b_2 \quad (12)$$

Where W_1, b_1, W_2, b_2 are learnable parameters.

V. Experimental Setup and Result

A. Implementation

The challenge of recognising human activity is a multiclass classification problem. As a result, we classify the activity samples into one of several candidate classes. Mobile sensors are used to record activities as time series data. Given a sensor data sequence, the Training model is intended to generate predictions of the actual set of labels *y*. Transformer model has the advantage of being able to handle long-term dependence. As a result, Transformer multihead attention mechanism can remember critical information about the received input. This allows them to predict the next input with great accuracy.

Classification difficulties necessitate the use of a loss function to describe model accuracy. The resultant divergence from the true values is calculated by the loss function. As a cost function, we employed the cross-entropy loss (log loss function), which quantifies the model's inaccuracy by measuring the model's capacity to discover a relationship between input data and output labels.

We divided the dataset into 75% and 25% parts [18]. Our model uses 75% of the data for training and the remaining 25% for testing. We have further divided our training data into training (80%) and validation data (20%) to track the model performance to avoid overfitting.

We used the Adam optimisation function for training the model [19]. Adam is an adaptive stochastic gradient descent algorithm. The speed and effectiveness with which the optimizer modifies the network parameters (weights and biases) determines the selection criterion.

Overfitting of models is a common problem in machine learning. If the model is over-fitted, the resulting model is nearly meaningless. This problem arises when the model's loss function is large in the testing data but smaller in the training data. There is a significant difference in accuracy between training and testing when a model is overfitted.

Our model makes use of the Dropout function, a formalisation technique widely used in deep learning contexts [20], in an effort to reduce overfitting. It selects cells (neurons) at random from the neural layer and simply hides them. The training and optimisation procedures for the neural network are then repeated in a loop. It will hide a few more cells (neurons) in the ensuing loop till the training is finished. It therefore forces the remaining neurons to make up the slack during training by arbitrarily excluding part of them. In both of our models, the dropout layer is employed. Dropping out is an excellent method of achieving regularisation. It lessens the likelihood that the network may become unduly dependent on a few neurons.

We tweak the hyperparameter to find values that predict the best values. Hyperparameter that we consider for the classification are: *Learning Rate:* It is a critical parameter which determines the step size using which model adjust its weight during the training. *Epochs:* While training number of times whole dataset passed through the neural network are epochs. *Batch Size:* The overall number of training instances that were used in a single iteration. *Number of Hidden Units or Layers:* The architecture of the network, including hidden layers and the number of

neurons in each layer. *Activation Function:* The function that determines the output of a node or neuron in a neural network. Common choices include *ReLU* (Rectified Linear Unit), *sigmoid*, and *tanh*. *Optimizer:* To minimize the loss optimizer algorithm are used such as Adam, Stochastic Gradient Descent (SGD).

a.Bidirectional LSTM Approach: Convolution and BiLSTM layers will be combined into a single model for implementation. Convolutional layers are employed to extract spatial information from the frames; the spatial features that are retrieved are then sent to BiLSTM layers for temporal sequence modelling at each time-step.

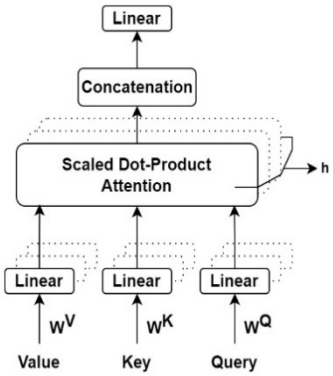

Fig. 4 Single-head Attention

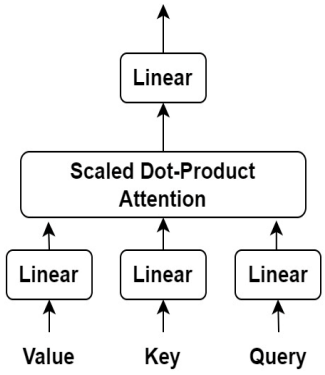

Fig. 5 Multi-Head Attention

In this manner, the network immediately captures spatiotemporal information during an end-to-end training, producing a reliable model.

b. Transformer Approach: Time Distributed is used to apply convolutional layers to every frame in the sequence. This facilitates the independent extraction of spatial characteristics from every frame.

We combine CNN with multihead Layer of transformer. In this design, long-range connections and relationships within the sequence of features extracted from frames are captured by the Transformer layer. This enables the model to learn intricate patterns and focus on various segments of the sequence.

Then, using Layer Normalization, the input and output of the multihead attention layer are merged. In this step, normalisation is introduced and the transformer's training is stabilised.

Finally, classification is done using the Transformer layer's output. The last vector in the sequence is chosen, and the final output is generated by applying a dense layer with softmax activation.

B. Evaluation

We used Python v3.9.18 code built with TensorFlow and Keras version 2.10.0 to execute the trials on a Windows 10 workstation. The CPU is an Intel(R) Xeon (TM), 3.80 GHz processor with 256GB of RAM. We run our model on UCI101 dataset [21]. The accuracy of the models on the dataset and the training times are shown in the following.

a. UCI101 Dataset: UCI101 dataset consist of 101 different action classes, and each classes have 120-140 videos associated with it (Fig. 6). After resizing and normalizing the image, we extract the necessary frames. The data from the selected classes is then extracted and the appropriate dataset is created.

We form two dataset from the UCI101 dataset, named as C-2 and C-14. Details of both C-2 and C-14 are shown in Table I and Table II respectively. C-2 has two classes for binary classification, whereas C-14 consist of 14 distinct classes for multiclass classification.

Table III shows the performance of Bidirectional model and Transformer model on C-2 and C-14 dataset along with all the hyperparameters. These results obtain after multiple experiment and tuning of parameters.

TABLE I DATASET(C-2) SAMPLE DESCRIPTION

S.No.	Action	Training	Testing
1	PlayingGuitar	118	42
2	PlayingSitar	119	38

TABLE II DATASET(C-14) SAMPLE DESCRIPTION

S.No.	Action	Training	Testing
1	Archery	106	39
2	BenchPress	117	43
3	BodyWeightSquats	78	34
4	Bowling	107	48
5	CleanAndJerk	84	28
6	GolfSwing	109	30
7	HorseRiding	133	31
8	JavelinThrow	92	25
9	JumpRope	104	40
10	PullUps	77	23
11	PushUps	76	26
12	Shotput	110	34
13	SkateBoarding	95	25
14	TaiChi	73	27

TABLE III UCI101 DATASET ACCURACY AND TIME

Parameters	C-2	C-14
Epochs	50	200
Iteration	24	136
Batch Size	8	8
Learning rate	0.001	0.001
Dropout	0.25	0.25
Accuracy		
Bidirectional	97.50	90.73
Transformer	98.75	91.17
Training Time (in sec.)		
Bidirectional	60	1005
Transformer	58	803

Confusion matrix on C-14 and C-2 dataset using Bidirectional model are shown in Fig. 7 and Fig. 9 respectively. Similarly, Confusion matrix on C-14 and C-2 dataset using Transformer model are shown in Fig. 8 and Fig. 10 respectively.

Fig. 6 Dataset Visualization

VI. CONCLUSION

After comparing both the models for human activity recognition in this paper, the findings demonstrated that the training accuracy for the dataset is marginally increased when the transformer model but the training time is considerably reduced. Furthermore, we plan to classify and recognize human activities using a variety of models used in machine learning and deep learning. Scale up our model to handle large datasets and diverse populations and explore techniques for improving generalization across different users, environments, and activities.

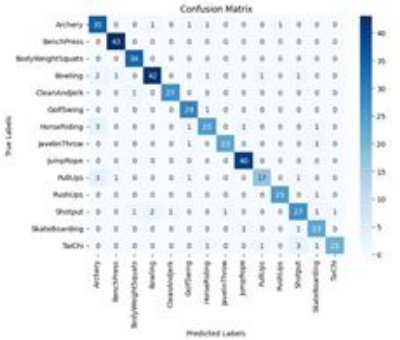

Fig. 7. Bidirectional Approach on C-14

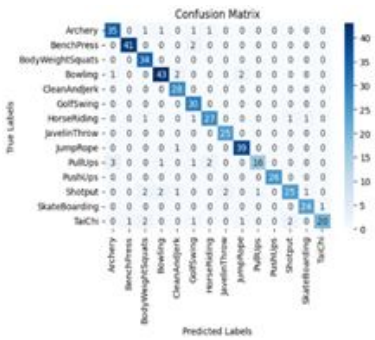

Fig. 8 Transformer Approach on C-14

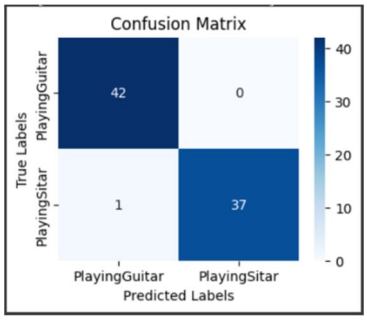

Fig. 9 Bidirectional Approach on C-2

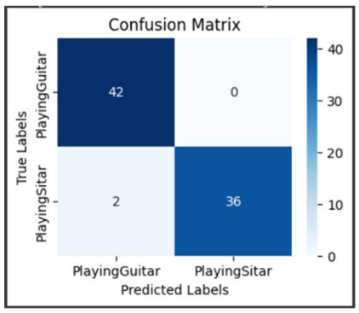

Fig. 10 Transformer Approach on C-2

REFERENCES

[1] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 12 2014.

[2] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. kin Wong, and W. chun Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," 6 2015.

[3] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadar- rama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, pp. 677–691, 4 2017.

[4] C. Avile´s-Cruz, A. Ferreyra-Ram´ırez, A. Zu´n˜iga-Lo´pez, and J. Villegas- Corte´z, "Coarse-fine convolutional deep-learning strategy for human activity recognition," Sensors (Switzerland), vol. 19, 4 2019.

[5] L. Alawneh, B. Mohsen, M. Al-Zinati, A. Shatnawi, and M. Al-Ayyoub, "A comparison of unidirectional and bidirectional lstm networks for human activity recognition," in 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 1–6, IEEE, 2020.

[6] J. Ye, X. Li, X. Zhang, Q. Zhang, and W. Chen, "Deep learning-based human activity real-time recognition for pedestrian navigation," Sensors (Switzerland), vol. 20, 5 2020.

[7] O. Nafea, W. Abdul, G. Muhammad, and M. Alsulaiman, "Sensor-based human activity recognition with spatio-temporal deep learning," Sensors, vol. 21, pp. 1–20, 3 2021.

[8] E. A. Budisteanu and I. G. Mocanu, "Combining supervised and unsu- pervised learning algorithms for human activity recognition," Sensors, vol. 21, 9 2021.

[9] E. Shalaby, N. ElShennawy, and A. Sarhan, "Utilizing deep learning models in csi-based human activity recognition," Neural Computing and Applications, vol. 34, pp. 5993–6010, 4 2022.

[10] M. A. Khatun, M. A. Yousuf, S. Ahmed, M. Z. Uddin, S. A. Alyami, S. Al-Ashhab, H. F. Akhdar, A. Khan, A. Azad, and M. A. Moni, "Deep cnn-lstm with self-attention model for human activity recognition using wearable sensor," IEEE Journal of Translational Engineering in Health and Medicine, vol. 10, 2022.

[11] K. Muhammad, A. Ullah, A. S. Imran, M. Sajjad, M. S. Kiran,G. Sannino, and V. H. C. de Albuquerque, "Human action recognition using attention based lstm network with dilated cnn features," Future Generation Computer Systems, vol. 125, pp. 820–830, 2021.

[12] S. Mekruksavanich and A. Jitpattanakul, "Lstm networks using smart- phone data for sensor-based human activity recognition in smart homes," Sensors, vol. 21, p. 1636, 2021.

[13] O. Vinyals, S. V. Ravuri, and D. Povey, "Revisiting recurrent neural networks for robust asr," pp. 4085–4088, IEEE, 3 2012.

[14] I. Sutskever, J. Martens, and G. Hinton, "Generating text with recurrent neural networks," 2011.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, pp. 1735–1780, 11 1997.

[16] A. Graves, S. Ferna´ndez, and J. Schmidhuber, "Bidirectional lstm net- works for improved phoneme classification and recognition," pp. 799– 804, Springer, 2005.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin, "Attention is all you need," Advances in neural information processing systems, vol. 30, 2017.

[18] K. K. Dobbin and R. M. Simon, "Optimally splitting cases for training and testing high dimensional classifiers," BMC medical genomics, vol. 4, pp. 1–8, 2011.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhut- dinov, "Dropout: a simple way to prevent neural networks from overfit- ting," The journal of machine learning research, vol. 15, pp. 1929–1958, 2014.

[21] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones.," vol. 3, p. 3, 2013.