C-LSTM Deep Learning Architecture for Human Activity recognition

Yuning Lei
Department of Electrical and Computer Engineering
Queen's University
Kingston, Canada
17yl6@queensu.ca

***Abstract*—The ability to recognize human activities is important and useful nowadays. Many applications are based on human activity recognition such as fitness trackers, smart homes, and security systems. In this paper, we implement the C-LSTM which is a deep learning architecture that combines convolutional neural network and long-short-term memory (LSTM) on a dataset including accelerometer data on performing different daily activities to classify each activity. The C-LSTM outperforms other deep learning models and has an accuracy of 92.5% on classifying different activities.**

Keywords— Human activity recognition, deep learning, CNN, LSTM, C-LSTM

I. INTRODUCTION

The ability to interpret human body gestures and motions in order to determine human activity is known as human activity recognition. In this paper, we mainly focus on analyzing human daily activities from body-worn sensors. The ability to recognize human activities is important and useful nowadays. Many applications are based on human activity recognition, such as health applications on apple watches, smart homes, and security systems. Therefore, human activity recognition is one of the most important tasks to AI and interactive systems. In this problem, there are mainly two types of data sources, one is based on body-worn sensors like IMU, and another one is based on the video from cameras. By comparing with datasets from body-worn sensors and cameras, we believe that the body-worn sensor will have better performance and broader applications. Because the body-worn sensors are less affected by the constraints of the environment such as obscurants of vision of cameras, and datasets from body-worn sensors contain fewer unnecessary information.

Human activity recognition tasks have common challenges: interclass variability and interclass similarity. For example, the motion such as downstairs and upstairs are similar on the dataset from the accelerometer. Therefore, it is important to develop a robust feature representation method to better classify different activities. In the past five years, the deep learning model achieved success in data analysis. Convolutional Neural Networks (CNN) have outperformed other deep learning approaches in computer vision, audio recognition, and natural language processing (NLP) [1]. This is due to its capacity to capture local correlations of spatial or temporal features. Task-dependent and non-handcrafted characteristics are extracted by the CNN.

However, CNN is only able to learn local responses from temporal or geographical data but not sequential correlation. Since human motion is continuous, there will be missing features when using CNN. On the other hand, a Recurrent neural network (RNN) with a memory cell to simulate temporal dependencies in time series problems is as known as long-short-term memory (LSTM). But the LSTM is unable to extract features om parallel. In summary, both CNN and LSTM has their advantages and disadvantages when dealing with human activity recognition problem. In our experiment, we proposed an architecture that combines CNN and LSTM together to solve human activity recognition problems.

II. RELATED WORKS

The dataset for human activity recognition which is collected by using wearable sensors is usually time-series data and collected from 20 Hz to 200 Hz [2]. Analyzing this kind of data usually follows a pipeline-based strategy [3]. The first step is data pre-processing which includes noise reduction and dimensionality reduction. Specifically, time-series data will be separated into continuous segments through the sliding window method. Then it will be feature extraction and feature selection. The last step is feeding the selected feature into a classifier to get the result.

Deep learning has become the most popular trend in human activity recognition over the last decade, with tremendous success in a variety of HAR tasks. Convolutional networks have been the most common approach so far because of its capacity to capture local correlations while also extracting high-level correlations. There were researches by Chen [4] and Yang [5] focusing on using CNN to solve the human activity recognition problems from accelerometer data in 2015. The CNN model from these two papers is very similar, and both reached an accuracy of around 90%. Other than that, Ignatov [6] improved the performance by adding the statistical features into CNN model. This method achieved an accuracy of around 95%.

The research focuses on using a recurrent neural network to solve the human activity recognition is not very common. Deep recurrent networks have been used to recognise human activities in two different scenarios. First, Neverova et al. [7] evaluated a number of recurrent algorithms for identifying persons in a big dataset using movement data captured from their cell phones. Second, in 2016, Ordonez et al. [8]examined the performance of recurrent networks with convolutional neural networks on two datasets. RNN can deal with variable-length input sequences and uncover long-term dependencies as a sequence model [9].

However, both CNN and LSTM have their advantages and disadvantages on human activity recognition, and we believe that by merging these networks into a single framework, human activity identification performance can be enhanced. In 2018, Yu [9] applied a combination of CNN and LSTM to a human activity recognition problem. Their algorithms

employ multi-layer CNNs and feed the result of a fully linked CNN layer into the RNN as inputs and name this architecture as C-LSTM. In this paper, we created an algorithm that is similar to the C-LSTM and applied it to human activity recognition, with the goal of improving recognition accuracy.

III. ARCHITECTURE

In this section, we will show the basic structure of our algorithm. As mentioned in the above section, the C-LSTM combines both CNN (convolutional layer) and LSTM (recurrent layers). In our architecture, the convolutional layers are used for extracting features. Therefore, a more abstract representation of the input sensor data can be obtained. Because it usually leads to better results when we feed features derived from raw sensor signals than directly using raw data [11]. Even raw sensor signals can be fed into neural networks, whether feedforward or recurrent. The recurrent layers in our algorithms create a feature map based on the temporal dynamics of the raw sensor data. Then the feature maps will be def into the recurrent layers to find out the sequential correlation and determine the class of activity. The architecture of the C-LSTM models is shown in Fig 1.

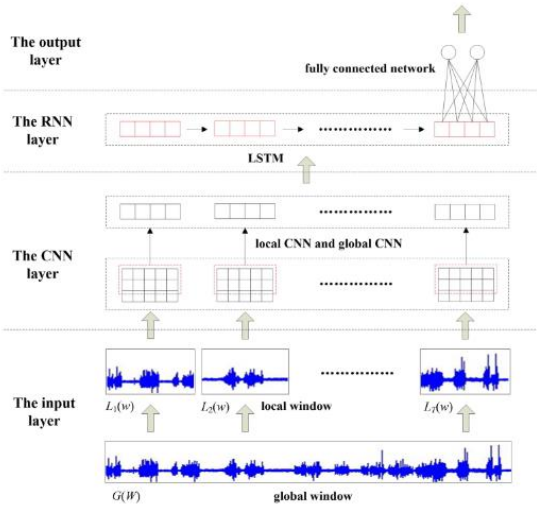

Figure 1:The architecture of C-LSTM [12]

A. Convolutional Layers

To extract feature maps based on the temporal dynamics of input sensor data, 2D convolutional layers are suggested to solve this problem [13]. The structure of the 2D convolutional layers is shown in Fig 2.

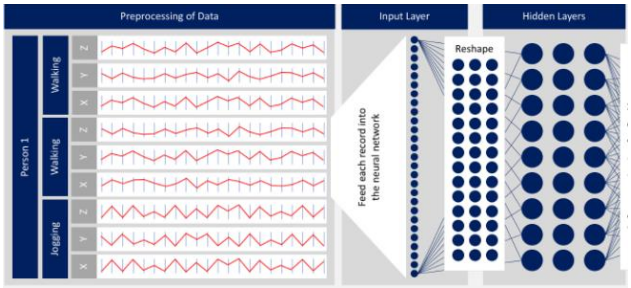

Figure 2: The Structure of convolutional layers [14]

The input sensor data has the shape of 80 * 3, where 80 represents the number of the time frame in each data segment and 3 represents the number of accelerometer readings (x, y, z axis). The input layer is a vector with 240 elements which represent 80 time slices for three accelerometer readings. There are three hidden layers and each contains 100 nodes. Three hidden layers are fully connected. The output of previous layer is the input of the next layer. The final layer can represent the input sensor data in a more abstract way [10]. The equation for calculating each element in the feature map are shown in Eq 1.

$$c_j = f(w_j * m + b) \quad (1)$$

Where m is a filter for the convolution operation and in our case, m is a 2 by 2 matrix. f is a nonlinear transformation function. In our case, we used Relu as the nonlinear function [15].

After convolution, feature maps are commonly subjected to max-over-pooling or dynamic k-max pooling to identify the most or k-most essential features. Pooling, on the other hand, will break such sequence structure due to the discontinuous selected features, while LSTM is designed for sequence input. We won't use pooling after the convolution process because we're stacking an LSTM neural network on top of CNN [1].

,

B. Long Short-Term Memory Networks

Recurrent neural networks (RNNs) use chain-like neural network topology to propagate historical data. It examines the current input x_t as well as the previous output of hidden state h_{t-1} at each time step when processing sequential data. But the standard RNN has the problem of vanishing feature, which means it cannot learn the correlation between two elements with a large gap. To solve this problem, LSTM was first introduced in [9]. The structure of LSTM is shown in Fig 3.

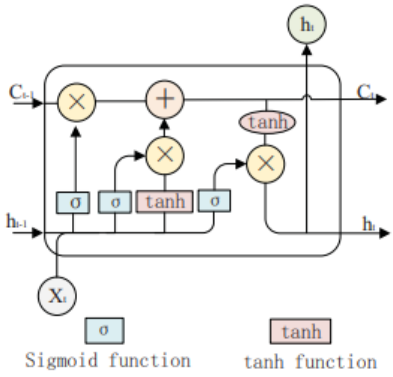

Figure 3: A basic LSTM cell [10]

The memory cells is the key feature in LSTM, making it easier to understand temporal correlations across extended time scales. The input of an LSTM is routed into different gates that conduct different operations on the cell memory, as shown in Fig.1. The gates, which consist of a sigmoid function and an element-wise multiplication function, assist us in determining which operation is performed on the cell memory. As shown in Fig 3, the input gate determines what new information should be kept in memory, the output gate determines what information should be output, and the forget gate determines what information should be forgotten. The activation is calculated in the same way that it is in RNN. The following is the equation used on the LSTM layers:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2)$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (3)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (5)$$

$$h_t = o_t \tanh(C_t) \quad (6)$$

Where i, f, o are the input gate, forget gate and output gate respectively. W are the weight metrices. The activation vectors and hidden values are shown in the Eq 4 and Eq 5.

IV. EXPERIMENT

A. *Data Preprocessing*

The C-LSTM network was tested by using dataset from WISDM lab [16]. In this dataset, six subjects were recorded by a single accelerometer. Each subject performed a session with six daily activities including walking, jogging, upstairs, downstairs, sitting and standing. Data was recorded at 20 Hz and input space has a dimension of three channels which are x-axis, y-axis, and z-axis. A sample of input sensor data is shown in Fig 4.

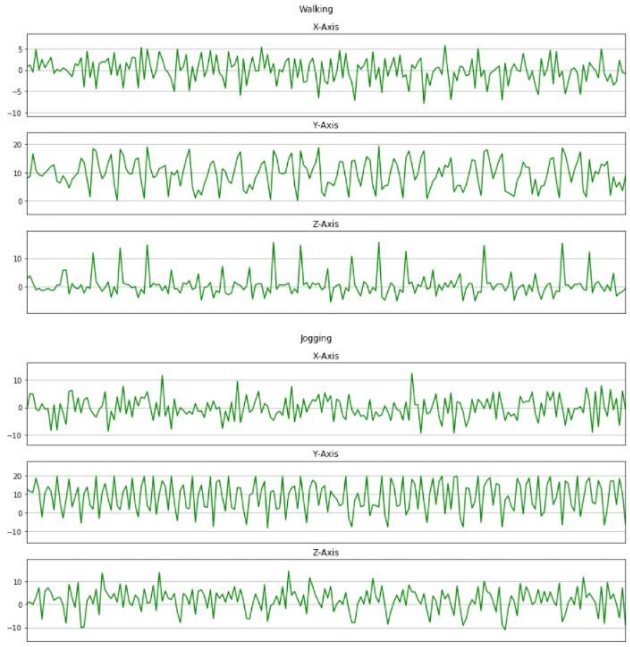

Figure 4: input sensor data sample

In this dataset, the number of examples collected over different activities was highly unbalanced, which is shown in the Table 1

Activities	# Of example	percentage
Walking	424400	38.6%
Jogging	342.177	31.2%
Upstairs	122869	11.2%
Downstairs	100472	9.1%
Sitting	59939	5.5%
Standing	48395	4.4%

Table 1: Class Distribution [16]

Since the unbalanced dataset can lead to a biased weight matrix in the C-LSTM network, the first step in our data preprocessing is to balance the dataset. The dimensions of each activity were reduced to the dimension of [3555, 3], which is the number of examples in the standing class. After that, to avoid magnitude differences, we apply mean and variance normalization on the z-score scale with a standard deviation of 0.5.

As indicated in Fig 5, we segment the three channels' data using a sliding window of set length T. The window was changed to produce fresh samples with a 50% overlap, which added redundancy during training and testing and accounted for some of the label changes being learned. Each window was referred to as a "segment," which will constitute the network's input. The length of the window is set to 4 seconds which contains 80 timeframes. The label that is most often used in each segment is selected as the label of each window. The label of each window is shown as the ground truth in Fig 5.

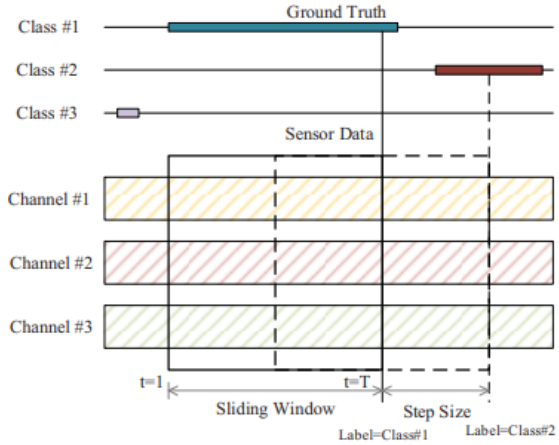

Figure 5: Sliding Window for Data Pre-processing [10]

For the optimization algorithms, we apply Adam optimization algorithm [17]. The Adam optimizer combined the gradient descent with momentum [18] and RMSprop [19]. The update rule of Adam optimization algorithms is shown in Eq 7.

$$w_t = w_{t-1} - \eta\left(\frac{\hat{m}}{\sqrt{\hat{v}_t} + \epsilon} + \lambda w_{t-1}\right) \quad (7)$$

Where m is the updated value computed by gradient descent with momentum and v is computed by RMSprop. By applying the Adam optimizer, the training cost of each iteration in the neural network is reduced [17].

B. *Experimental Result*

In this section the performance of the C-LSTM on the WISDM dataset is compared with other deep learning algorithms such as CNN and LSTM. the CNN model is build

based on the code from Kant *[14]*. The result is shown in Table 2, and all these deep learning all performed on the same dataset and using the same data preprocess method.

Deep Learning Architecture	**Accuracy**
LSTM	71.03%
CNN	87.6%
C-LSTM	92.5%

Table 2: Accuracy of each Deep Learning Architecture

From Table 2, it shows that the C-LSTM outperformance other deep learning models and improves the accuracy from 4.9 % to 21.2%.

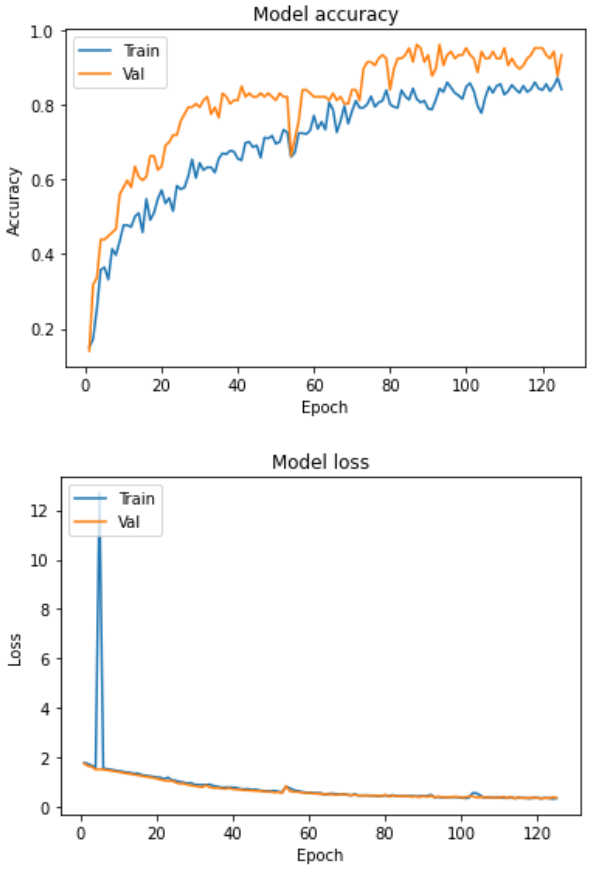

Figure 6: Training Session Progress over Iteration

Fig 6 depicts the progress of the training session across iterations, including the loss, and accuracy trend as well as the training and testing data for our C-LSTM network. The training outcomes were excellent, with high accuracies, and low training losses, as expected. In addition, the testing findings were quite good. When the training was complete, the testing accuracies oscillated around 91.5% and the best result is 92.5%.

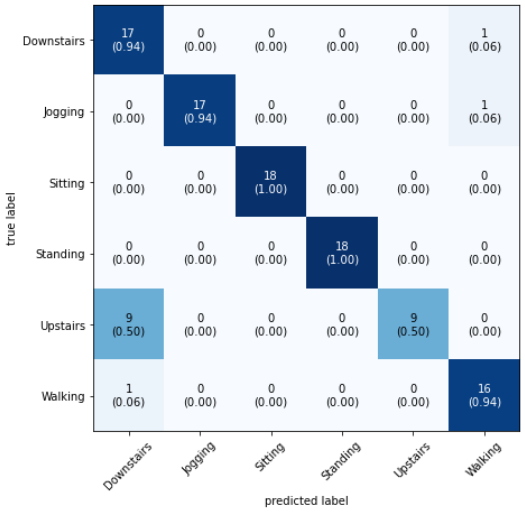

Figure 7: Confusion Matrix on testing dataset using C-LSTM

The confusion matrix on the validation dataset using C-LSTM is shown in Fig 7. Activities include downstairs, jogging, sitting, standing, upstairs, and walking. The confusion matrix shows the number of times that activity in the row is recognized as activity in the column. As shown by Fig 7, it is difficult for the C-LSTM to distinguish downstairs and upstairs. The reason could be the raw data from the accelerometer is similar when subjects perform upstairs and downstairs. To solve this problem, we might need data from different sensors such as inertial measurement units (IMU) which include accelerometers, gyroscopes, and magnetometers.

V. DISCUSSION

In this study, we implement the C-LSTM structure proposed by Zhou [1] on the human activity recognition task. In this C-LSTM architecture, it combines the convolutional neural network and LSTM. The convolutional layer in the structure can capture the temporal dynamics and the LSTM capable of capturing temporal dynamics. The performance of this architecture outperforms CNN and LSTM by improving the accuracy from 4.9 % and 21.2 %. One disadvantage of the C-LSTM architecture we found during the experiment is the computational cost of C-LSTM is much higher than the two-dimensional CNN and LSTM. It took L-STM around 100 iterations to reach accuracy around 80% which CNN can reach within 15 iterations.

However, the C-LSTM still can't distinguish downstairs and upstairs. To solve this problem, data from different sensors such as IMU might be needed. Nowadays, most wearable devices such as the apple watch that provide activity recognition not only include the accelerometer and gyroscope also contain heart rate sensor. For future human activity research, we could include the heart rate into our dataset to take advantage of various sensors in the state of art wearable devices and further improve the accuracy.

VI. References

[1] C. Zhou, C. Sun and Z. Liu, "A C-LSTM Neural Network for Text Classification," *Computer Science*, pp. 1(4):39-44, 2015.

[2] . Y. N. Hammerla, S. Halloran and T. Poeatz, "Deep, Convolutional, and Recurrent Models for Human Activity Recognition using Wearables," *Journal of Scientific Computing* , pp. 61(2): 454-476, 2016.

[3] A. Bulling and U. Blanke , "A tutorial on human activity recognition using body-worn inertial sensors," *Acm Computing Surveys*, pp. 46(3):1-33, 2014.

[4] Y. Chen and Y. Xue, "A Deep Learning Approach to Human Activity Recognition Based on Single Accelerometer," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, Hong Kong, 2015.

[5] J. Yang and M. Nguyen, "Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition," in *International Joint Conference on Artificial Intelligence*, 2015.

[6] A. Ignatov, "Real-time human activity recognition from accelerometer data using Convolutional Neural Networks," *Applied Soft Computing* , vol. 62, pp. 915-922, 2018.

[7] N. Natalia and W. Christian , "Learning human identity from motion patterns," *arXiv*, no. 1511.03908, 2015.

[8] O. Francisco and R. Daniel, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, p. 115, 2016.

[9] H. Seep, "Long short-term memory. Neural computation," *Neural Computation*, vol. 9(8), pp. 1735-1780, 1997.

[10] S. Yu, L. Qin and Q. Yin, "A C-LSTM Neural Network for Human Activity Recognition Using Wearables," in *2018 International Symposium in Sensing and Instrumentation in IoT Era (ISSI)*, Shanghai, 2018.

[11] D. Palaz and D. Nagimai, "Analysis of CNN-based Speech Recognition System using Raw Speech as Input," in *In Proceedings of the 16th Annual Conference of International Speech Communication Association (Interspeech)*, Dresden, 2015.

[12] M. Lv, W. Xu and T. Chen, "A hybrid deep convolutional and recurrent neural network for complex activity recognition using multimodal sensors," *Neurocomputing*, vol. 362, pp. 33-40, 2019.

[13] J. Yang, M. Nguyen and P. San, "Deep Convolutional Neural Networks On Multichannel Time Series For Human Activity Recognition," in *In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, Buenos Aires, 2015.

[14] L. Kant, "Human Activity Recognition Using Accelerometer Data," 5 September 2019. [Online]. Available: https://github.com/laxmimerit/Human-Activity-Recognition-Using-Accelerometer-Data-and-CNN. [Accessed 8 December 2021].

[15] N. Vinod and H. E. Geoffrey, "Rectified linear units improve restricted boltzmann machines," in *In Proceedings of the 27th International Conference on Machine Learning*, 2010.

[16] J. Kwapisz and G. Weiss, "Activity Recognition using Cell Phone Accelerometers," in *Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data* , Washington, 2010.

[17] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015.

[18] R. Sebastian, "Anoverview of gradient descent optimization algorithms," *arXiv:1609.04747v2 [cs.LG]*, 2017.

[19] A. Graves, "Generating Sequences With Recurrent Neural Networks," *arXiv:1308.0850v5 [cs.NE]*, 2014.