Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition

Aabhas Asawa	2017A1PS0829P
Aniruddha Mahajan	2017A7PS0145P
Kshitij Gupta	2017B3A70601P

Problem

The basic task that is being done in this case is Human Activity Recognition (HAR). It is based on the assumption that certain activities performed by humans would transform into certain characteristic signals by wearable sensors, which could be classified by machine learning techniques. Wearable sensors such as accelerometers, gyroscopes and magnetic field sensors are used in this case. Detection of human activities play an important role in smart home systems, in health support and in industrial settings. However, this can be further applied to memory prosthetics (to help people with dementia), inserting subtle cues in everyday life in the right context to support voluntary behavior change (e.g., to fight obesity),
or enabling natural human-robot interaction in everyday settings.

Data

We have used the **OPPORTUNITY dataset** here, which comprises of a set of complex naturalistic activities collected in a sensor-rich environment.

It is available on the UCIMachine Learning Repository. It contains recordings of four subjects in a daily living scenario, performing activities with sensors integrated on the body and in the surroundings.

During the recordings, each subject performed a session five times with activities of daily living (ADL) and one drill session. During each ADL session, subjects perform the activities without any restriction, by following a loose description of the overall actions to perform. During the drill sessions, subjects performed 20 repetitions of a predefined sorted set of 17 activities.

Tasks Performed

We have focused the models on two tasks defined in the OPPORTUNITY challenge:

Each task consist of two parts with/without NULL class i.e. classification of locomotion and gestures with or without NULL classes.

Task A: Recognition modes of locomotion and postures.

The goal of this task is to classify modes of locomotion from the full set of body-worn sensors. This is a 5-class segmentation and classification problem.

Task B: Recognition of sporadic gestures.

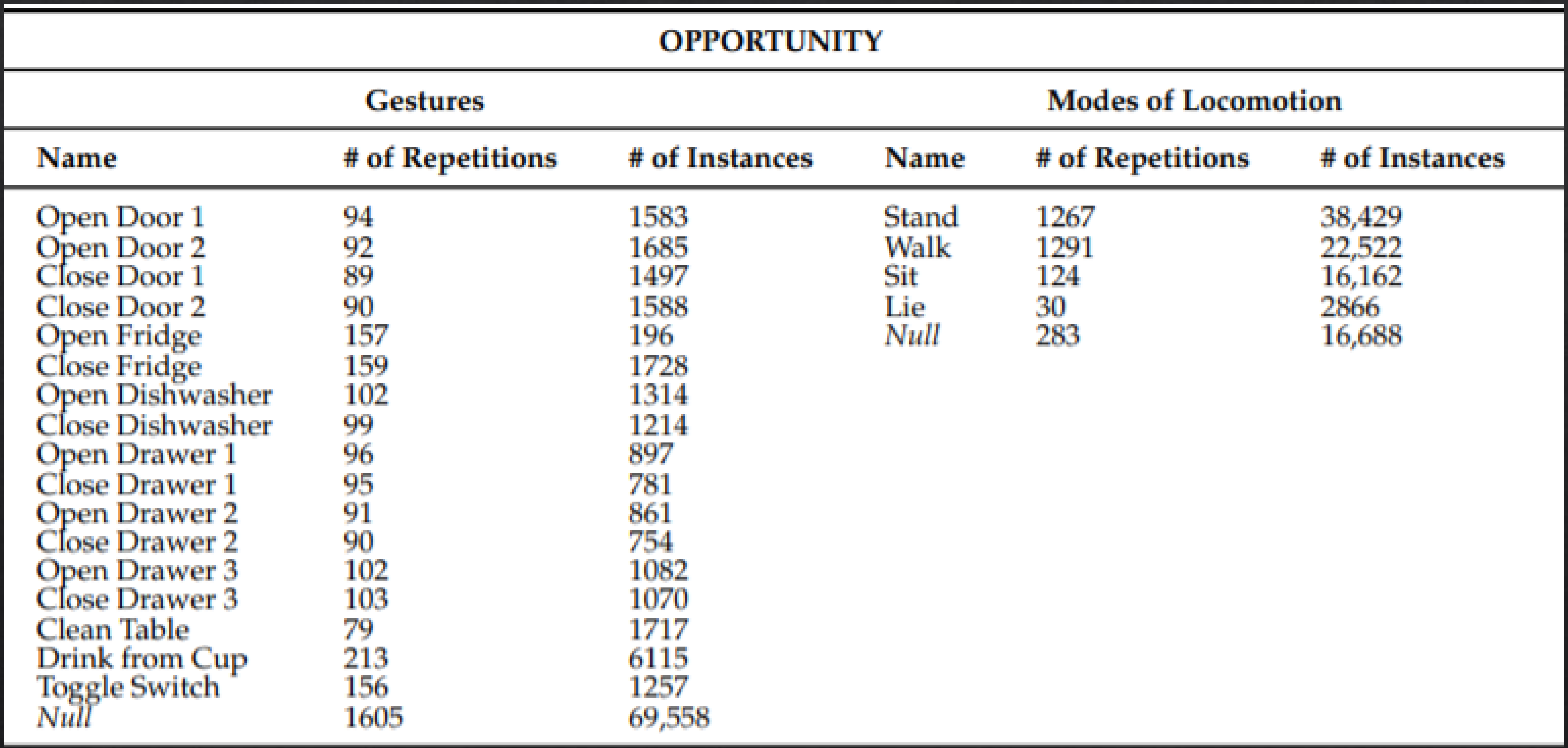
This task concerns recognition of the different right-arm gestures. This is an 18-class segmentation and classification problem.

Data Processing

The different values of sensors were normalized so that they lie between 0 and 1. The data was supervised and the labels for "locomotion" and "gestures" were added and the different activities were numbered accordingly.

The input to the model consists of a data sequence. The data sequence is a short time series which is extracted from the sensor data using a sliding window technique. The length of the window was chosen as 500 ms while the step size was chosen as 250 ms. The sliding window technique would help us understand better if the model is able to determine the start and end of sporadic gestures.

Classes in Opportunity dataset



OPPORTUNITY					
Gestures			**Modes of Locomotion**		
Name	**# of Repetitions**	**# of Instances**	**Name**	**# of Repetitions**	**# of Instances**
Open Door 1	94	1583	Stand	1267	38,429
Open Door 2	92	1685	Walk	1291	22,522
Close Door 1	89	1497	Sit	124	16,162
Close Door 2	90	1588	Lie	30	2866
Open Fridge	157	196	*Null*	283	16,688
Close Fridge	159	1728			
Open Dishwasher	102	1314			
Close Dishwasher	99	1214			
Open Drawer 1	96	897			
Close Drawer 1	95	781			
Open Drawer 2	91	861			
Close Drawer 2	90	754			
Open Drawer 3	102	1082			
Close Drawer 3	103	1070			
Clean Table	79	1717			
Drink from Cup	213	6115			
Toggle Switch	156	1257			
Null	1605	69,558			

Model

The model used here, **DeepConvLSTM**, is a dense neural network (DNN) made of convolutional, recurrent and softmax layers.

It comprises of four convolutional layers which process information only along the time axis. These layers act as feature extractors and provide abstract information about the raw sensor data in the feature maps. These layers make use of the rectified linear units (ReLu) to compute the feature maps.

The next two layers are recurrent layers which provide a useful dimension in recognizing activities. The units of these layers are LSTM recurrent cells which can change state according to the previous information processed.
Therefore, certain actions which differ by sequence of signals can be more accurately determined in this case.

The last layer is the softmax layer which provides the probability of an action belonging to a certain class.

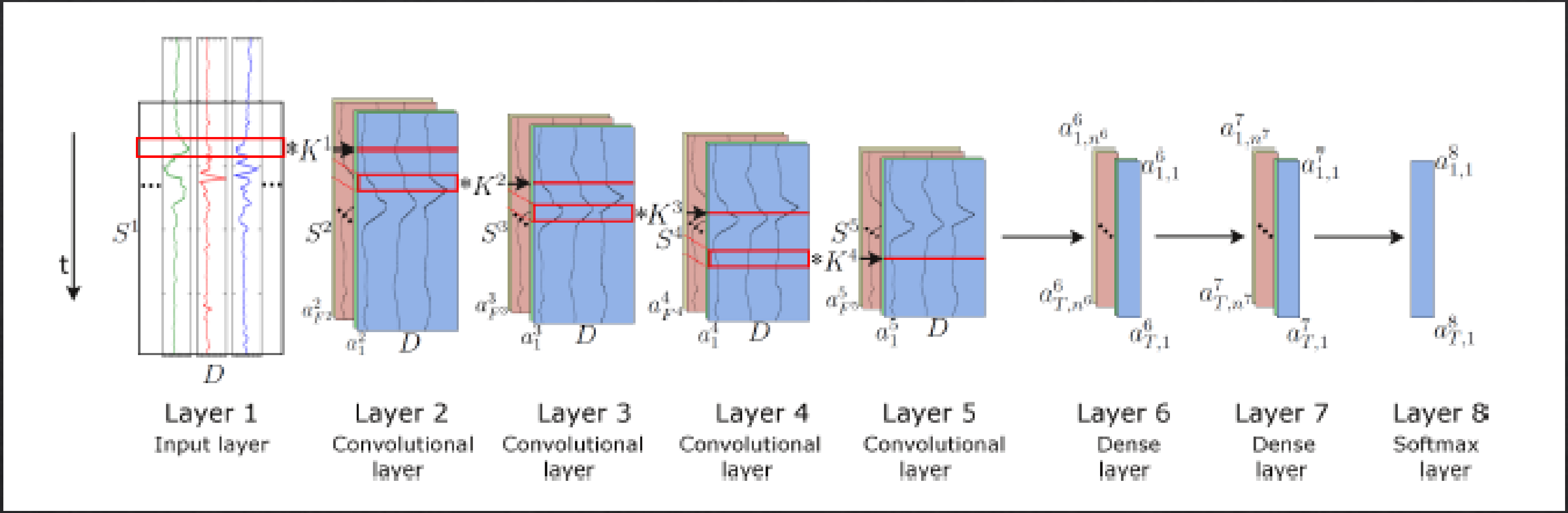
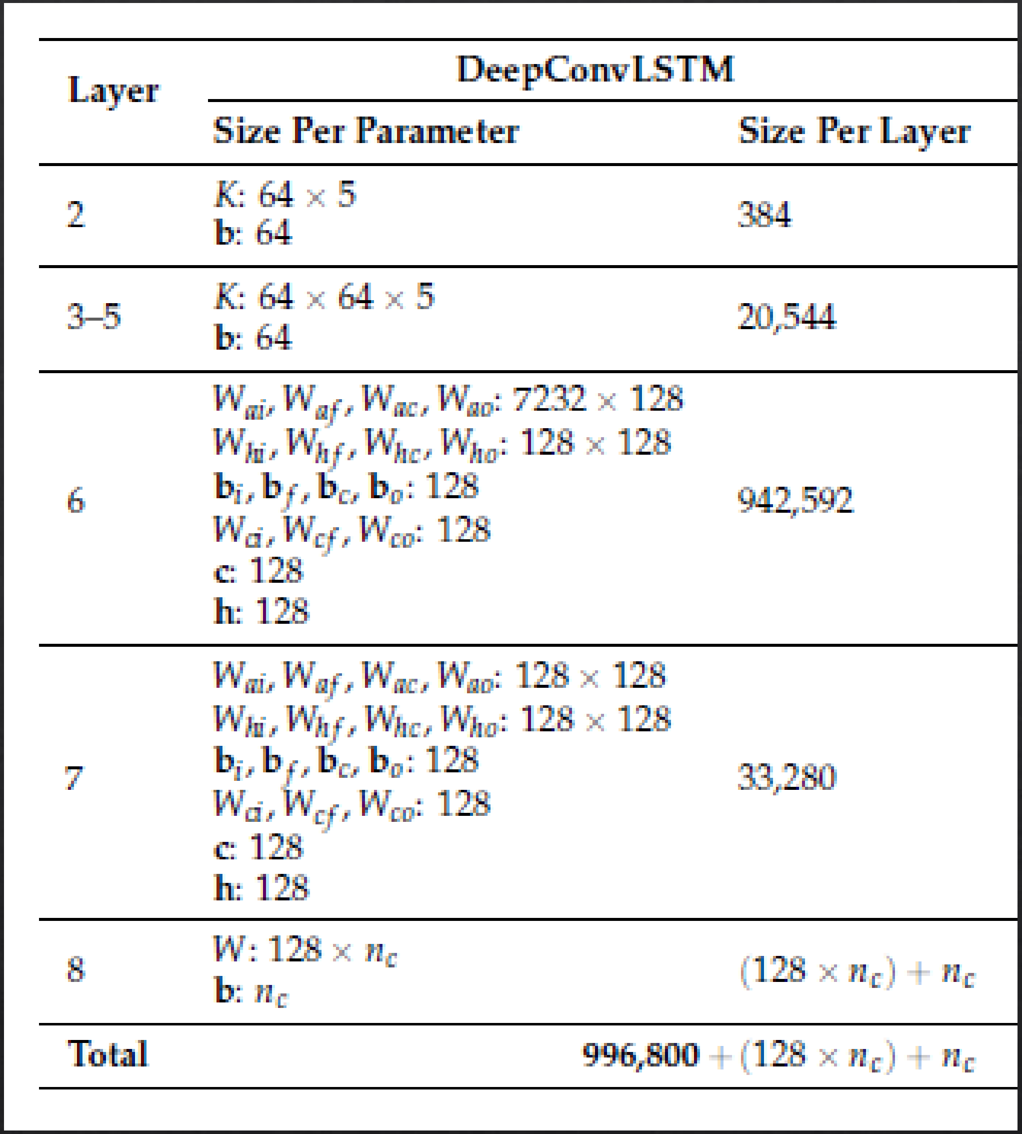


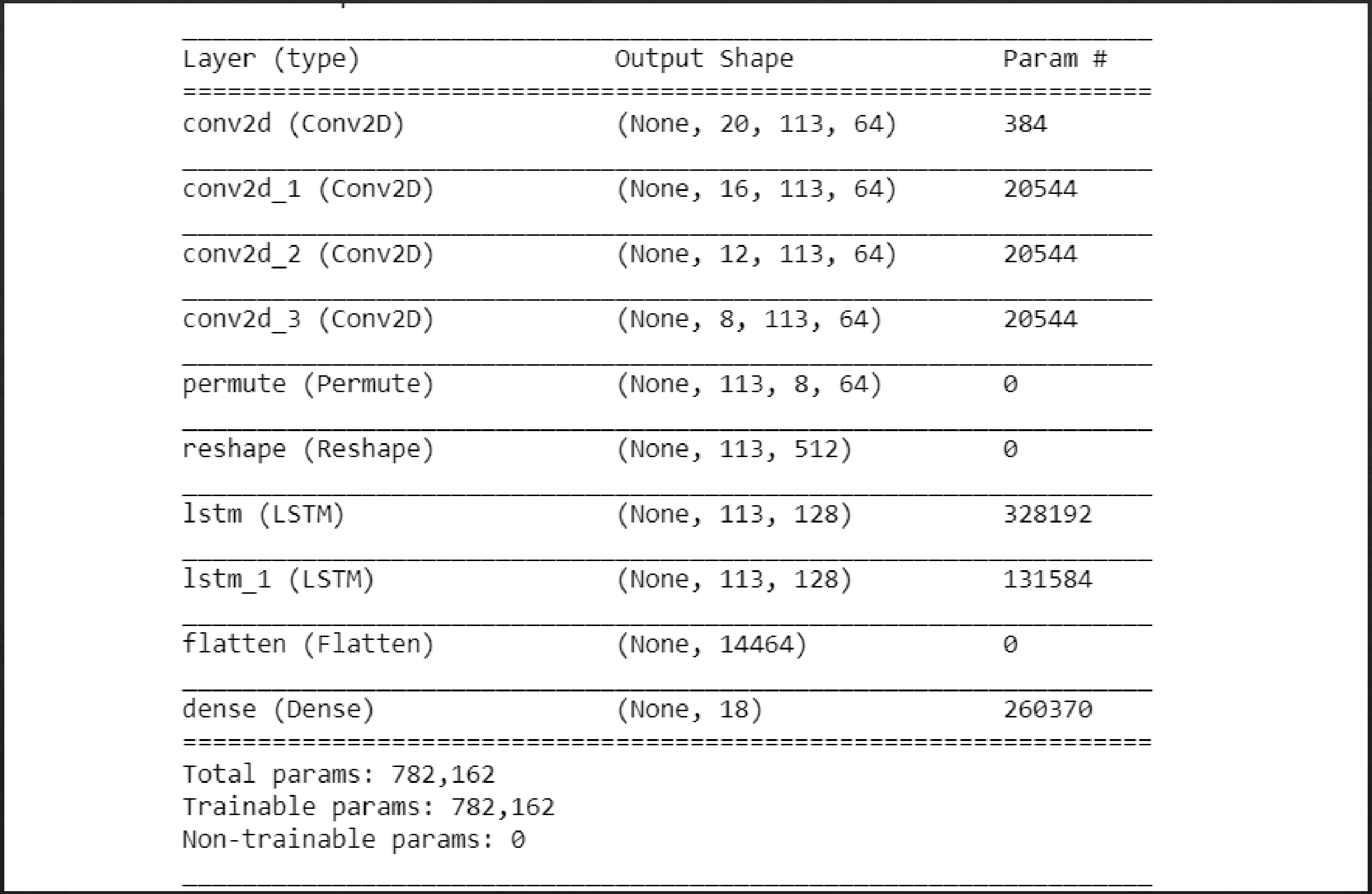
Figure: Architecture of the DeepConvLSTM (Conv, convolutional) framework for activity recognition. From the left, the signals coming from the wearable sensors are processed by four convolutional layers, which allow learning features from the data. Two dense layers then perform a non-linear transformation, which yields the classification outcome with a softmax logistic regression output layer on the right. Input at Layer 1 corresponds to sensor data of size D * S^1, where D denotes the number of sensor channels and S^l the length of features maps in layer l. Layers 2–5 are convolutional layers. K^l denotes the kernels in layer l (depicted as red squares). F^l denotes the number of feature maps in layer l. In convolutional layers, a^l_i denotes the activation that defines the feature map i in layer l. Layers 6 and 7 are dense layers. In dense layers, $a^l_{t,i}$ denotes the activation of the unit i in hidden layer l at time t. The time axis is vertical.



Layer	DeepConvLSTM	
	Size Per Parameter	Size Per Layer
2	K: 64×5 b: 64	384
3–5	K: $64 \times 64 \times 5$ b: 64	20,544
6	$W_{ai}, W_{af}, W_{ac}, W_{ao}$: 7232×128 $W_{hi}, W_{hf}, W_{hc}, W_{ho}$: 128×128 $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_c, \mathbf{b}_o$: 128 W_{ci}, W_{cf}, W_{co}: 128 c: 128 h: 128	942,592
7	$W_{ai}, W_{af}, W_{ac}, W_{ao}$: 128×128 $W_{hi}, W_{hf}, W_{hc}, W_{ho}$: 128×128 $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_c, \mathbf{b}_o$: 128 W_{ci}, W_{cf}, W_{co}: 128 c: 128 h: 128	33,280
8	W: $128 \times n_c$ b: n_c	$(128 \times n_c) + n_c$
Total		**996,800** $+ (128 \times n_c) + n_c$

Table: Number and size of parameters for the DeepConvLSTM architecture. The final number of parameters depends on the number of classes in the classification task, denoted as n_c.

Quantitative Results



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d (Conv2D)              (None, 20, 113, 64)       384       
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 113, 64)       20544     
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 12, 113, 64)       20544     
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 113, 64)        20544     
_________________________________________________________________
permute (Permute)            (None, 113, 8, 64)        0         
_________________________________________________________________
reshape (Reshape)            (None, 113, 512)          0         
_________________________________________________________________
lstm (LSTM)                  (None, 113, 128)          328192    
_________________________________________________________________
lstm_1 (LSTM)                (None, 113, 128)          131584    
_________________________________________________________________
flatten (Flatten)            (None, 14464)             0         
_________________________________________________________________
dense (Dense)                (None, 18)                260370    
=================================================================
Total params: 782,162
Trainable params: 782,162
Non-trainable params: 0
_________________________________________________________________
```

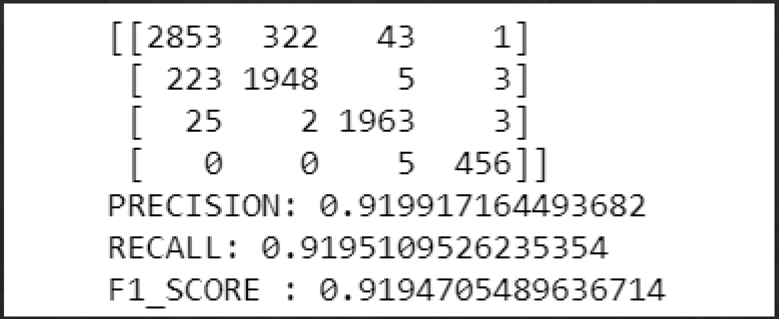
Fig. This is a snapshot of model during one of the cases of TASK B.

Quantitative Results

Task	F1 Score obtained	F1 Score in the paper
Task A with NULL class	0.8663	0.895
Task A without NULL class	0.9195	0.930
Task B with NULL class	0.8886	0.915
Task B without NULL class	0.92	0.930

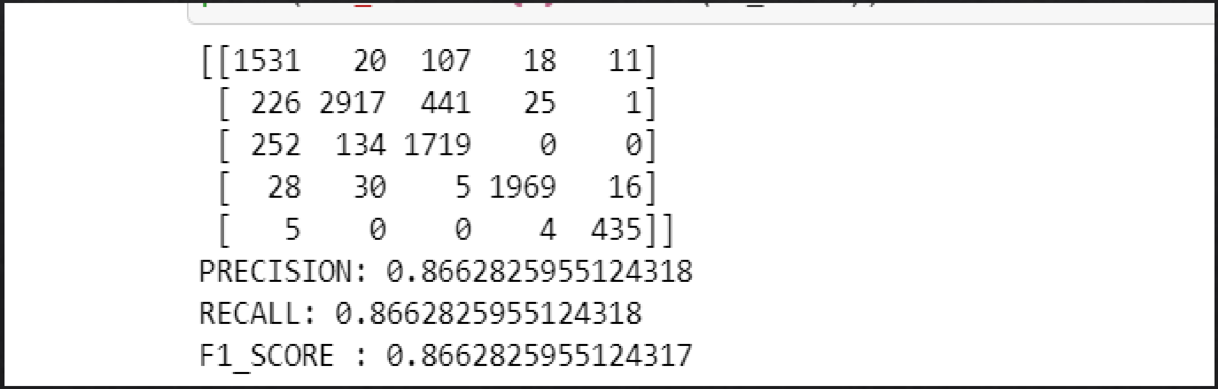
Confusion Matrix

TASK A with NULL class



```
[[2853  322   43    1]
 [ 223 1948    5    3]
 [  25    2 1963    3]
 [   0    0    5  456]]
PRECISION: 0.919917164493682
RECALL: 0.9195109526235354
F1_SCORE : 0.9194705489636714
```

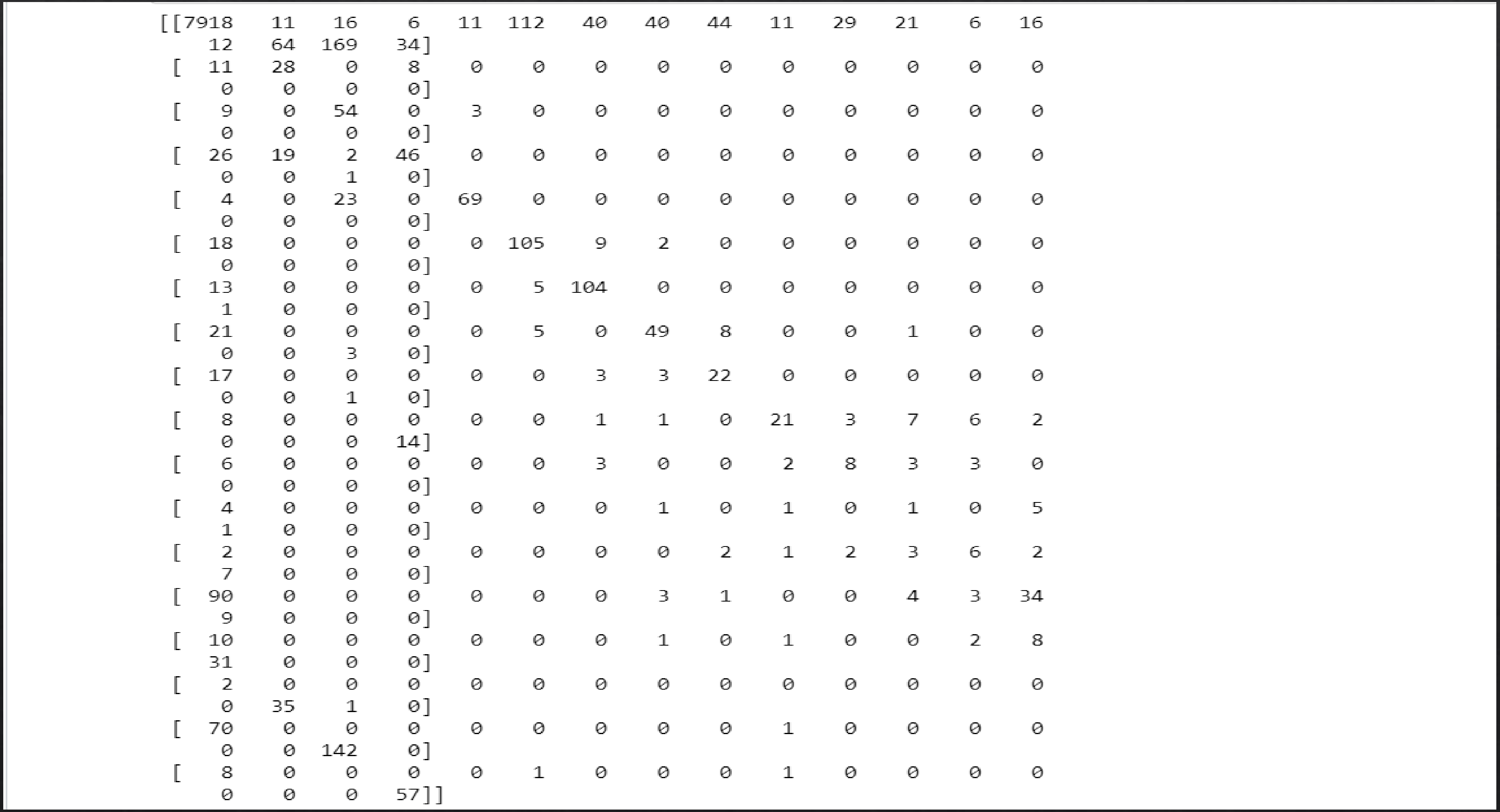
TASK A without NULL class



```
[[1531   20  107   18   11]
 [ 226 2917  441   25    1]
 [ 252  134 1719    0    0]
 [  28   30    5 1969   16]
 [   5    0    0    4  435]]
PRECISION: 0.8662825955124318
RECALL: 0.8662825955124318
F1_SCORE : 0.8662825955124317
```

Confusion Matrix

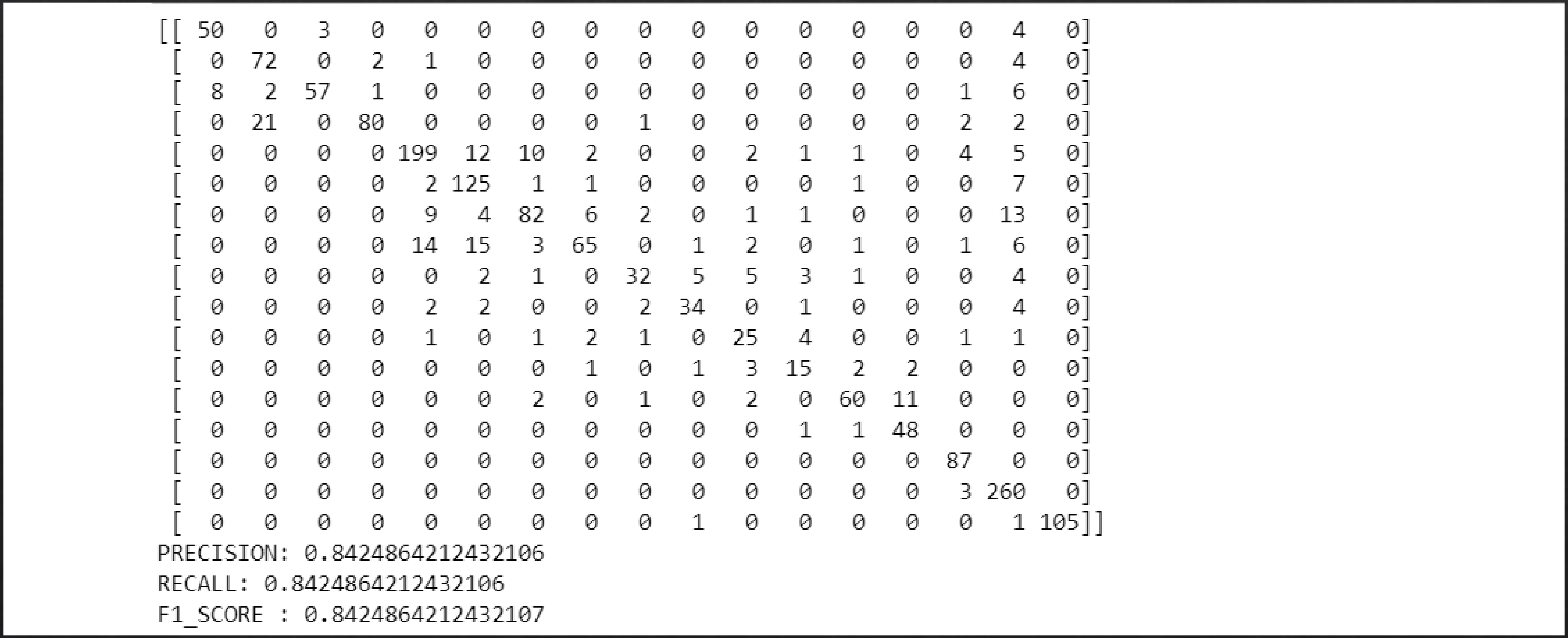
TASK B with NULL class



```
[[7918   11   16    6   11  112   40   40   44   11   29   21    6   16
    12   64  169   34]
 [  11   28    0    8    0    0    0    0    0    0    0    0    0    0
     0    0    0    0]
 [   9    0   54    0    3    0    0    0    0    0    0    0    0    0
     0    0    0    0]
 [  26   19    2   46    0    0    0    0    0    0    0    0    0    0
     0    0    1    0]
 [   4    0   23    0   69    0    0    0    0    0    0    0    0    0
     0    0    0    0]
 [  18    0    0    0    0  105    9    2    0    0    0    0    0    0
     0    0    0    0]
 [  13    0    0    0    0    5  104    0    0    0    0    0    0    0
     1    0    0    0]
 [  21    0    0    0    0    5    0   49    8    0    0    1    0    0
     0    0    3    0]
 [  17    0    0    0    0    0    3    3   22    0    0    0    0    0
     0    0    1    0]
 [   8    0    0    0    0    0    1    1    0   21    3    7    6    2
     0    0    0   14]
 [   6    0    0    0    0    0    3    0    0    2    8    3    3    0
     0    0    0    0]
 [   4    0    0    0    0    0    0    1    0    1    0    1    0    5
     1    0    0    0]
 [   2    0    0    0    0    0    0    0    2    1    2    3    6    2
     7    0    0    0]
 [  90    0    0    0    0    0    0    3    1    0    0    4    3   34
     9    0    0    0]
 [  10    0    0    0    0    0    0    1    0    1    0    0    2    8
    31    0    0    0]
 [   2    0    0    0    0    0    0    0    0    0    0    0    0    0
     0   35    1    0]
 [  70    0    0    0    0    0    0    0    0    1    0    0    0    0
     0    0  142    0]
 [   8    0    0    0    0    1    0    0    0    1    0    0    0    0
     0    0    0   57]]
```

Confusion Matrix

TASK B without NULL class



```
[[ 50   0   3   0   0   0   0   0   0   0   0   0   0   0   0   4   0]
 [  0  72   0   2   1   0   0   0   0   0   0   0   0   0   0   4   0]
 [  8   2  57   1   0   0   0   0   0   0   0   0   0   0   1   6   0]
 [  0  21   0  80   0   0   0   0   1   0   0   0   0   0   2   2   0]
 [  0   0   0   0 199  12  10   2   0   0   2   1   1   0   4   5   0]
 [  0   0   0   0   2 125   1   1   0   0   0   0   1   0   0   7   0]
 [  0   0   0   0   9   4  82   6   2   0   1   1   0   0   0  13   0]
 [  0   0   0   0  14  15   3  65   0   1   2   0   1   0   1   6   0]
 [  0   0   0   0   0   2   1   0  32   5   5   3   1   0   0   4   0]
 [  0   0   0   0   2   2   0   0   2  34   0   1   0   0   0   4   0]
 [  0   0   0   0   1   0   1   2   1   0  25   4   0   0   1   1   0]
 [  0   0   0   0   0   0   0   1   0   1   3  15   2   2   0   0   0]
 [  0   0   0   0   0   0   2   0   1   0   2   0  60  11   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   1   1  48   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0  87   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   3 260   0]
 [  0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1 105]]
PRECISION: 0.8424864212432106
RECALL: 0.8424864212432106
F1_SCORE : 0.8424864212432107
```

Qualitative Results

The results obtained showed that the model was working decently well.

Including the NULL class gives a higher score in gesture recognition case than not including it.

Whereas in case of locomotion recognition including NULL class gives a lesser score that not including it.

The scores obtained by our model were also close to that mentioned in the paper .