AssistNet: Transfer Learning for Training New Gestures to Recognize

Himanshu Sharma
Thasina Tabashum
Trevor Exley

Dr. Mark Albert
CSCE/BMEN 5280

Abstract

Gesture Recognition is an active field of research with applications such as automatic recognition of sign language, interaction of humans with robots, and for new ways of controlling video games. The objective of the project is to use transfer learning for training new human gestures using WISDM (Wireless Sensor Data Mining) Smartphone and Smartwatch Activity and Biometrics Dataset data and/or accelerometer data collected by mobile phone and smartwatch. Transfer learning allows us to train a model to perform new gesture recognition by leveraging the already existing labeled data of related tasks or domains. We try to store this knowledge gained in solving the source task in the source domain and apply it to our problem of interest. This application would allow Hannah's SPG (speech-generation device) to learn new specified gestures as she utilizes her device. This functionality will streamline the configuration process so that she can add gestures at her own will.

Data specification

The "WISDM Smartphone and Smartwatch Activity and Biometrics Dataset"includes data collected from 51 subjects, each of whom were asked to perform 18 tasks for 3 minutes each. Each subject had a smart-watch placed on his/her dominant hand and a smartphone in their pocket. The sensor data that was collected was from the accelerometer and gyroscope on both the smartphone and smartwatch, yielding four total sensors. The sensor data was collected at a rate of 20 Hz.

Number of subjects	51
Number of activities	18
Minutes collected per activity	3
Sensor polling rate	20 Hz
Number raw measurements	15,630,426

Table 1: Dataset Summary

Activity	Code
Walking	A
Jogging	B

Stairs	C
Sitting	D
Standing	E
Typing	F
Brushing Teeth	G
Eating Soup	H
Eating Chips	I
Eating Pasta	J
Drinking from cup	K
Eating Sandwich	L
Kicking (Soccer Ball)	M
Playing Catch	O
Dribbling a Basketball	P
Writing	Q
Clapping	R
Folding Clothes	S

Table 2: 18 Activities Represented in Dataset

Field name	Description
Subject-id	Uniquely identifies the subject. Range: 1600-1650
Activity code	Identifies a specific activity as listed in Table 2
Timestamp	Time in millisecond
X	Sensor value for x-axis (-ve or +ve)
Y	Sensor value for y-axis (-ve or +ve)

Z	Sensor value for z-axis (-ve or +ve)

Table 3: Raw Data Description

Following is the general outline for our project:

1. Pre-trained the CNN model on 9 out of 18 activities.
2. Freeze parameters (weights) in models' lower convolution layers.
3. Add a custom classifier with several layers of trainable parameters to model.
4. Train classifier layers on training data available for remaining 9 activities.
5. Fine-tune hyperparameters and unfreeze more layers as needed.

Data Set Up:

With all data science problems, formatting the data correctly will determine the success or failure of the project. I separated the data into training, validation, and testing sets with a 60%, 20%, 20% split.

Design and Milestone.

Language(s): Python.
Libraries: TensorFlow, Keras, sklearn, pandas, numpy, PyTorch, Matplotlib, TensorFlowLite
IDE: Jupyter Notebook, VS Code
Hardware: RAM: 8 GB or above, Processor: Minimum 1GHz; Recommended 2GHz or more, Hard Drive: Minimum 32GB

Milestone	Description	Date
Data gathering and data preparation	Found two IMU dataset WISDM	8/31
Determine best dataset to use for project	The team decided IMU data is the focus for our time with the project	9/3
Data pre-processing	Used the raw data included in the dataset	9/7
Building Initial model	Used WISDM data to build the architecture for the model, transfer learned with HAR	9/8
Initial training and accuracy validation	Plotted accuracy validation for both datasets	9/8

Standardize the freezing of layers/weights for other datasets	Program written to analyze dataset and freeze/unfreeze layers and weights based on the similarities of data	9/13
Prepare and split the WISDM data into two sets	Grouped the WISDM gestures into two groups based on similar characteristics	9/13

Dataset Insights:

Figure 1, depicts the training example with activities. Figure 2 demonstrates the user ID vs training examples.

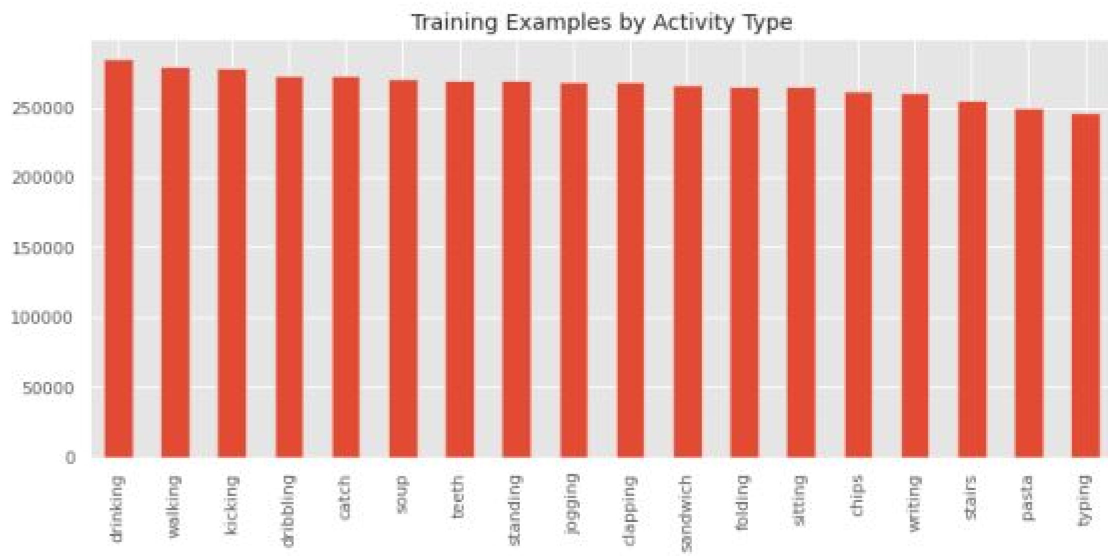

Figure 1 : Bar graph depicting number of training data for each activity/gesture

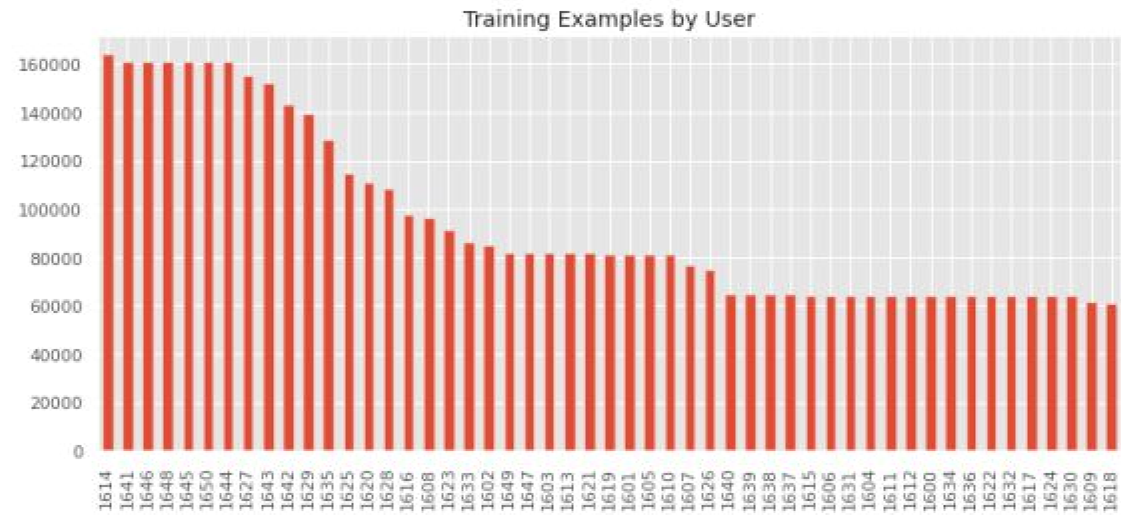

Figure 2: Bar graph depicting number of training data for each participant

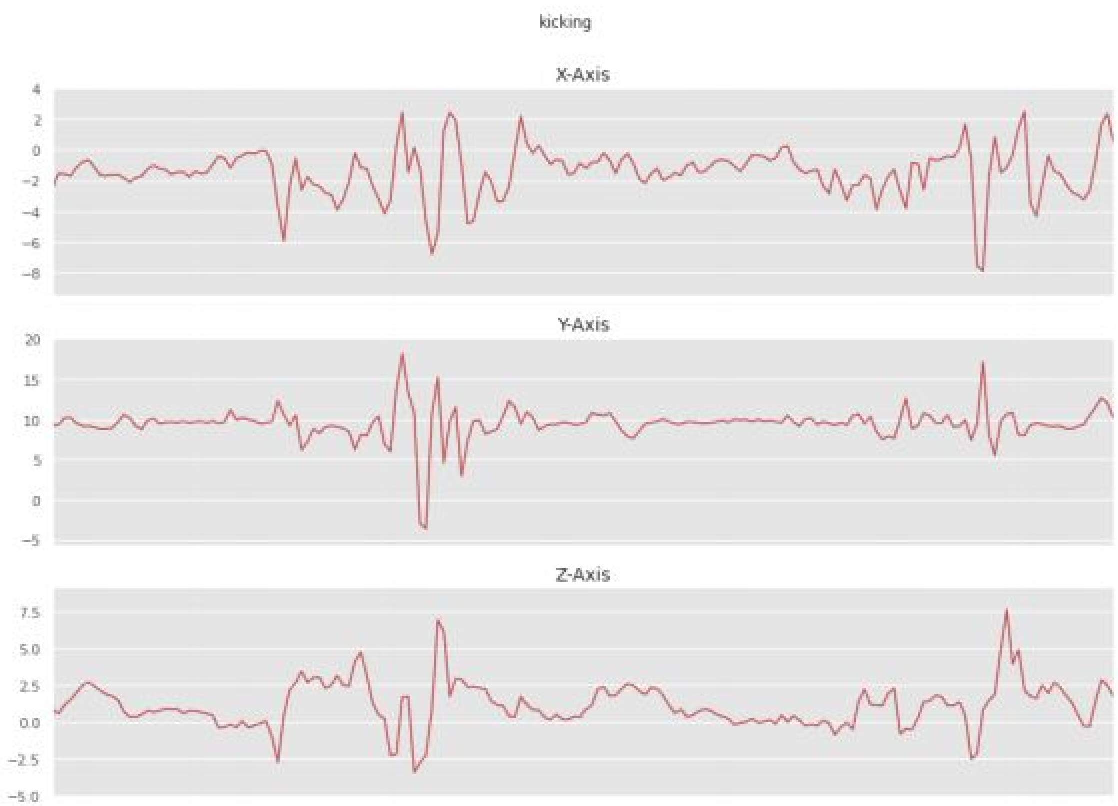

Figure 3: Graphed gesture (kicking a soccer ball) showing acceleration in x,y,z axes

We showed in figure 3 one example of activities.

Dataset Splitting for base model and transfer model:

Activity: We have split the activities into two parts 9 activities for base model and other 9 activities for transfer model.

For base model: Walking, Stairs, Standing, Brushing Teeth, Eating Chips, Drinking, Kicking a Soccer Ball, Dribbling a Basketball, Clapping.

For transfer learning: Jogging, Sitting, Typing, Eating Soup, Eating Pasta, Eating Sandwich, Catch a Baseball, Writing, Folding Clothes

Building a Base Model:

CNN model : Training the whole model with two splits of dataset

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 79, 2, 64)         320
_________________________________________________________________
dropout_3 (Dropout)          (None, 79, 2, 64)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 78, 1, 128)        32896
_________________________________________________________________
dropout_4 (Dropout)          (None, 78, 1, 128)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 9984)              0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                639040
_________________________________________________________________
dropout_5 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 9)                 585
=================================================================
Total params: 672,841
Trainable params: 672,841
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Summary of the CNN model used with our dataset

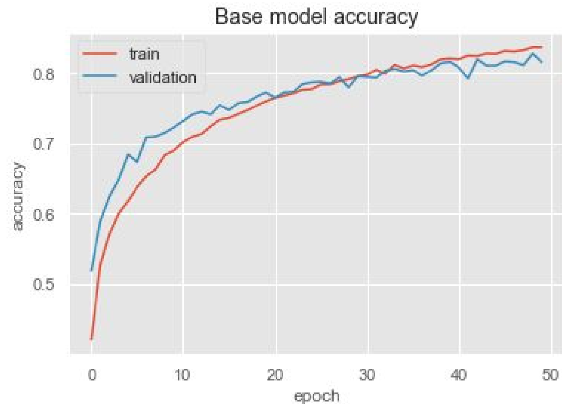

Figure 5:(Base CNN Model) Epochs vs Accuracy (above 84%)

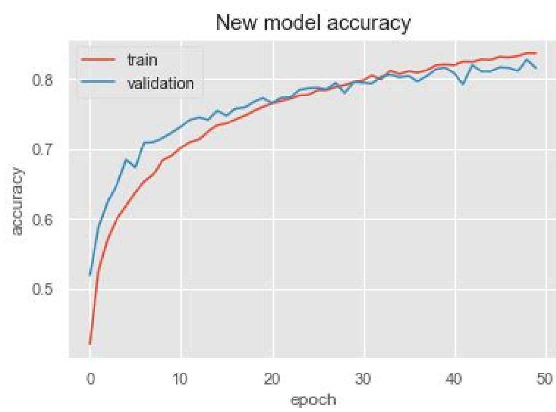

Figure 6: (Transferring new dataset training the whole previous CNN Model) Epochs vs Accuracy) (82%)

Results:

Fixed vs accuracy:

We have fixed each layer iteratively (1-8) and we ran thrice for each model then showed the mean and the standard deviation of accuracy in figure (7)

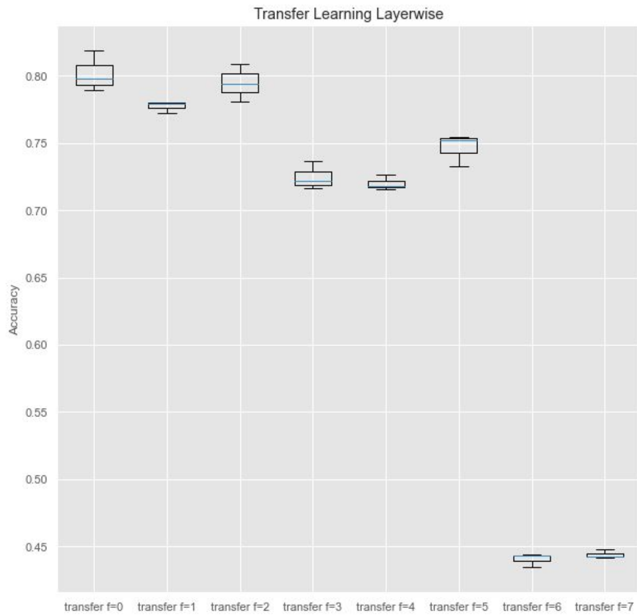

Figure 7: Fixed Layers vs Accuracy (with mean and std)

Fixed layers vs time

We have fixed each layer iteratively (1-8) and we ran thrice for each model then showed the mean of the running time of accuracy in figure (8)

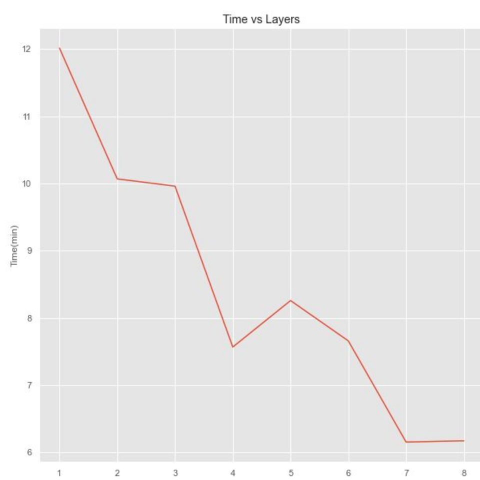

Figure 8: Fixed Layers vs Time (min)

Discussion :

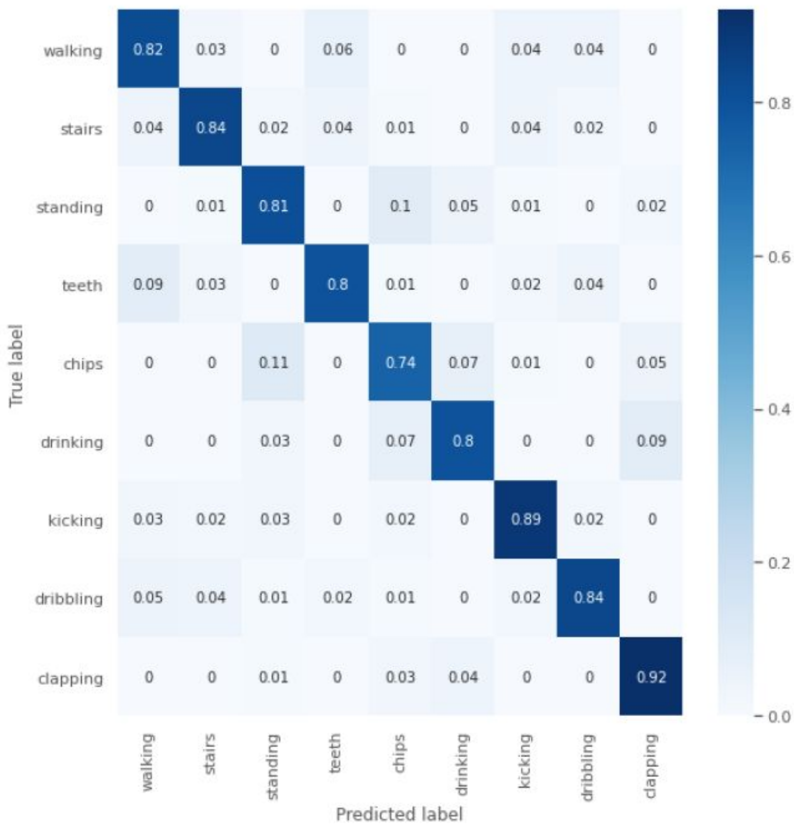

Figure 9: Confusion matrix for base model

Confusion matrix shown in Figure 9 is for the base model we have trained on the half of the activities (9 out of 18). And we can see that activities are correctly predicted 80% of the time. Eat chips and Standing are confusing most of the time.

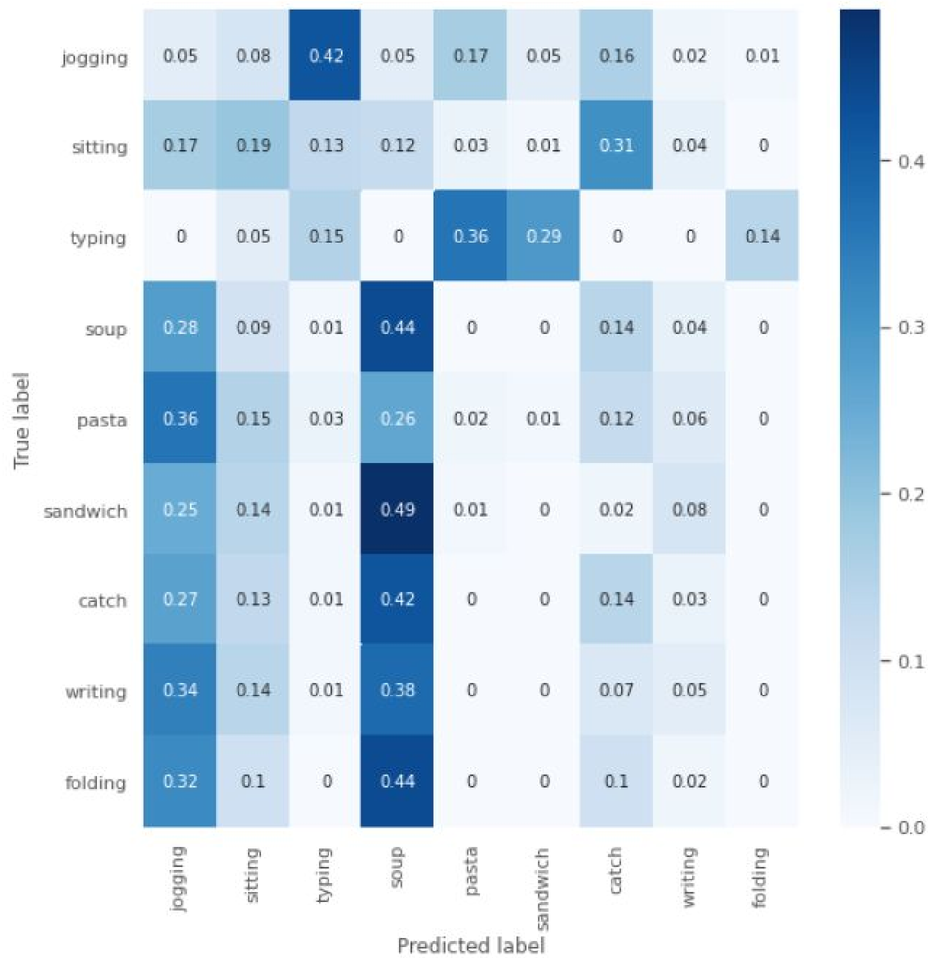

Figure 10: Confusion matrix for transfer learning model fixing all the layers (No-trained parameters on new dataset)

Confusion matrix shown in Figure 10 is for the rest of the 9 activities evaluated on the transfer model. In which the weights of all the layers are freezed. We can see that the transferred model is not doing well on the new activity with no trainable parameters and layers.

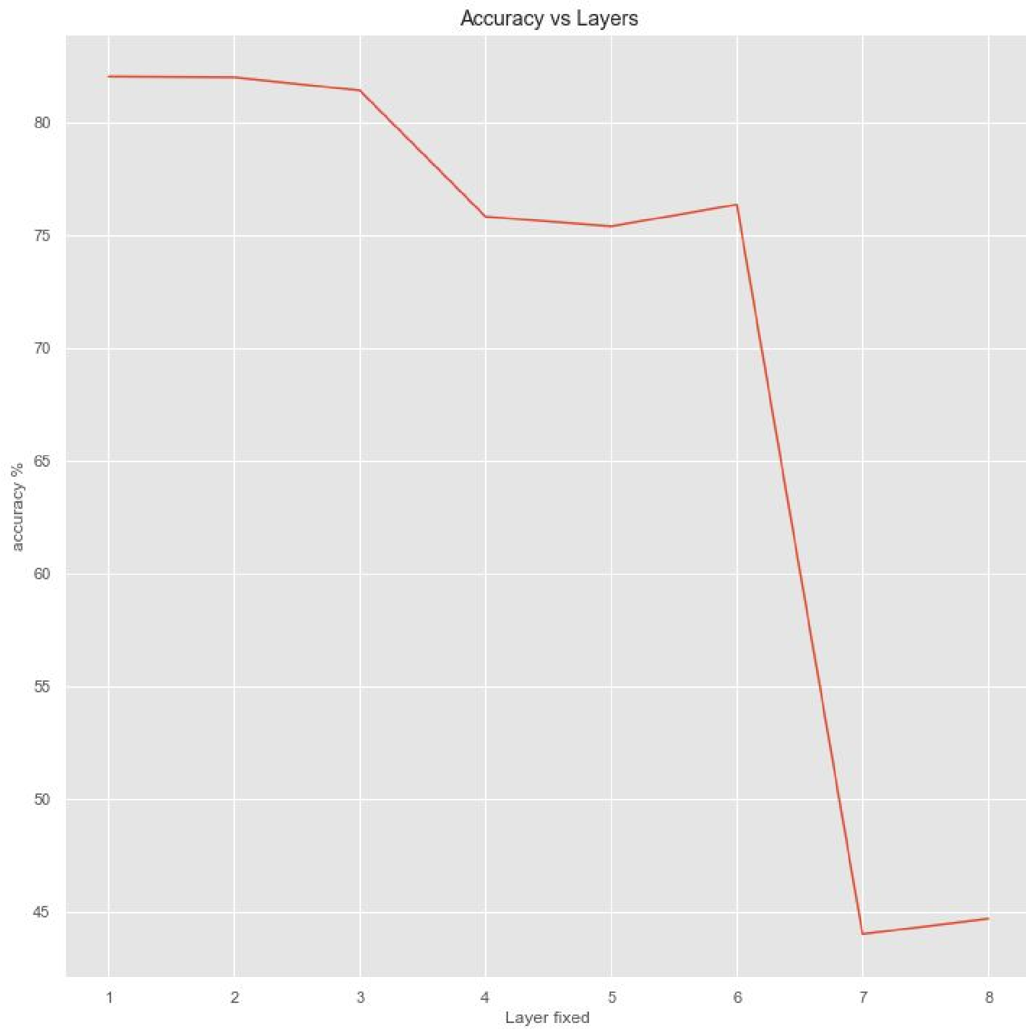

Figure 11: Layers fixed vs Accuracy

So, Figure 11 when we fix all the layers and transfer the new dataset to the older one, because of that the result is significantly bad. But from figure 3, we can observe that when we are freezing 3-4 layers, the model is performing well enough to identify new activity. In terms of time(figure 4) the training time also reduced.

Repository / Archive
https://github.com/yt-himanshu/AssistNet

Code
https://drive.google.com/file/d/1NLhv2WluAF0Jx2vg21LT8NctcVqNc60o/view?usp=sharing

Useful Links

1. Tutorial: Tensorflow Transfer learning and fine-tuning: https://www.tensorflow.org/tutorials/images/transfer_learning : *This tutorial outlines the general workflow of transfer learning. The tutorial outlines how to manipulate and augment small amounts of data if you don't have a large dataset (this pertains more to the image processing route, as IMUs would have less transformations that could be performed on data e.g.accelerations).* ***Freely available and open-source.***
2. On-device Learning of Activity Recognition Networks https://aqibsaeed.github.io/on-device-activity-recognition - *A project detailing how to collect data through a mobile phone and build a model.* ***Freely available and open-source.***
3. How to Improve Performance With Transfer Learning for Deep Learning Neural Networks *https://machinelearningmastery.com/how-to-improve-performance-with-transfer-learning-for-deep-learning-neural-networks/ ---- resource of transfer learning structured version*
4. *https://machinelearningmastery.com/how-to-improve-performance-with-transfer-learning-for-deep-learning-neural-networks/ ---- resource of transfer learning structured versions*
5. **Dataset**
    1. WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set *https://archive.ics.uci.edu/ml/datasets/WISDM+Smartphone+and+Smartwatch+Activity+and+Biometrics+Dataset+*