HUMAN ACTIVITY RECOGNITION USING DEEP LEARNING

ABSTRACT

Human activity recognition is the problem of classifying sequences of accelerometer data recorded by specialized harnesses or smart phones into known well-defined movements.

It is a challenging problem given the large number of observations produced each second, the temporal nature of the observations, and the lack of a clear way to relate accelerometer data to known movements.Classical approaches to the problem involve hand crafting features from the time series data based on fixed-size windows and training machine learning models, such as ensembles of decision trees. The difficulty is that this feature engineering requires deep expertise in the field.Recently, deep learning methods such as recurrent neural networks and one-dimensional convolutional neural networks or CNNs have been shown to provide state-of-the-art results on challenging activity recognition tasks with little or no data feature engineering.

RELATED WORK

The following are the deep learning baseline methods for time series classification in literature.

* MLP, FCN, ResNet, Encoder, MCNN, t-LeNet, MCDCNN, Time-CNN, TWIESN

These methods have been discussed in greater detail along with a comparative analysis in [2]. These methods are also compared with the state-of-the-art statistical time series classification methods like BOSS [4], COTE [5], DTW. ResNet and FCN are the models which have the highest number of wins on the benchmark datasets. We use FCN in our analysis.

INTRODUCTION

We have worked on two datasets - D1 and D2. D1 comprises of time series data belonging to 6 classes - Climbing Up, Climbing Down, Queue, Standing, Walking, In Train. D2 comprises of 3 classes - Climbing Up, Climbing Down, Walking. The paper is divided into two sections. Section A describes the datasets and the preprocessing techniques that we applied. We trained 3 models on parts of D1 and 1 model on D2. Section B describes the training and results of the 4 models.

Section A

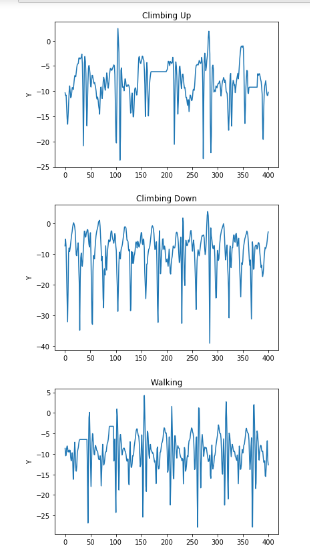
Dataset Description

The data in D1 and D2 have been collected by accelerometer sensors of a mobile phone placed in the pocket. On visualization, it is found that D1 has the following properties:

1. *Standing* and *In Train* have similar time series with accelerometer values close to 0 with a very less standard deviation
2. *Climbing up* and *Climbing down* can be differentiated from the rest of the time series by their high range of accelerometer values.
3. *Walking* class has some time series similar to *Queue* class whereas the rest resemble the *Climbing* classes.

D2 has been collected at a sampling frequency of 50 Hz. A sample climbing up, climbing down and walking time series of length 400 spanning upto 8 seconds is shown in the figure below.

Since the data was collected by a single person, the time series appear very uniform.



Data Preprocessing

Some of the important decisions related to data preprocessing for time series data includes how to slice the time series and the selection of the input window length.

In both D1 and D2, the data points have been generated by sliding window technique with a stride of 1. The train and validation sets were kept completely disjoint in both cases.

In [1], the problem of selecting an appropriate window length has been discussed. In D1, a window length of 24 and 35 have been selected. The size was also restricted due to the small size of the dataset. In D2, a window length of 100 was selected. With a sampling frequency of 50 Hz, this amounted to 2 seconds.

Section B

Model Architecture

There were 4 models trained. The architecture used in all 4 models is Fully Convolutional Network (FCN) [3]. In [2], the advantages of using a FCN for time series are discussed and its comparison with other time series models is made. The number of filters and the kernel size have been changed from the original architecture in [3]. The changes have been described in the following sections.

FCN Classifier

The FCN given in [3] was trained on 5 classes(excluding the In Train class) with a learning rate of 0.0001 (ReduceLROnPlateau callback of keras).

Results

The following confusion matrix was obtained on validation.

[[ 420, 23, 0, 55, 116],  
 [ 1, 298, 0, 31, 262],  
 [ 20, 91, 147, 7, 151],  
 [ 2, 18, 236, 872, 80],  
 [ 131, 186, 51, 85, 2101]]

The rows represent the actual values and the columns represent the predictions. An entry *a i,j* (i is the row and j is the column) corresponds to *a i,j* samples belong to class *i* were predicted as *j*.The following is the accuracy ratio for the classes:

0 **0.684** 1 **0.5** 2 **0.35** 3 **0.721** 4 **0.822**

The overall accuracy was **71.28** %.

Analysis

From the confusion matrix, we can infer that the overall accuracy is a misrepresentation of the model’s performance due to the following reasons:

1. The dataset was highly imbalanced with the walking class being the majority class and the 0.822 accuracy value can be attributed to that.
2. Queue shows an individual accuracy of 0.35 with more than half of Queue samples misclassified as walking.
3. Climbing down also has about half the samples misclassified as Walking.
4. Standing accuracy shows a decent value but that is perhaps because this data didn't include the In Train class, which has time series quite similar to the Standing class.

***Data belonging to different classes looks similar after visualization***

After visualizing the *X\_AXIS, Y\_AXIS, Z\_AXIS accelerometer* time series for a window length of 200 timesteps for all the 6 classes - Climbing Up - 1, Climbing Down - 2, Queue - 3, Standing - 4, Walking - 5, In Train - 6, the following can be inferred:

1. *Standing* and *In Train* have similar time series with accelerometer values close to 0 with a very less standard deviation
2. *Climbing up* and *Climbing down* can be differentiated from the rest of the time series by their high range of accelerometer values. What is surprising is that misclassification errors in the confusion matrix between these two classes are quite less. The data was trained on all features and we have visualized only the accelerometer values. Perhaps there was something in the other features to have differentiated between these classes. (Note: The features, however, didn't include barometer values).
3. About **42%** data in *Climbing up* and **28%** data in *Climbing down* class don't possess the characteristic high range value of these classes but are quite similar to other classes like Queue and Walking. These percentages are quite high to be considered as outliers and might hamper the training.
4. *Walking* class have some time series similar to *Queue* class and a few are similar to the *Climbing* classes.

**Length of time series**

The dataset was trained on a time series of length 24 and later it was decided to train it for a length of 35. The pitfalls of using such a length are as follows:

1. *Climbing* classes are characterized by peak and trough. Hence for small lengths, the model will only get to observe the peak and trough separately and is likely to misclassify. For the given data, a peak and a trough span over a length of 50 timesteps.
2. Depending on the waiting time in the queue, it is necessary to have a sufficiently long time series such that the model observes both, the moving and waiting, in the queue. In this case, it is difficult to estimate what can be an appropriate input length for a CNN based network.

**Number of training samples**

The dataset contains an uneven number of training samples per class. However, this can be countered by undersampling to attain class balance. But this will significantly reduce the number of training samples available for training. Also since we are using the windowing technique of generating data samples, there is a tradeoff between the number of samples and length of time series.

Cascading Classifier

Training, Results and Analysis

The dataset of *standing, walking,* and *queuing* was trained under the following conditions:

1. The first classifier was trained on standing and walking data. The architecture was an FCN + softmax. (89%)
2. The second classifier was trained on the previous data + queue data and had output as ‘queue’ and ‘not queue’. The output of the GAP layer of the first classifier was fed as input to the second and the second classifier was trained with the first classifier frozen. (92%) (See figure)
3. Together the model gives an accuracy of 86%. However, the accuracy for *queuing* is 69% and *not queuing* is 96% and hence 86% is a misrepresentation of the performance and this model needs to be re-trained and evaluated on balanced class data.

The aim of such a network was that the first classifier will specialize in classifying between Standing and Walking which is an easier problem. And when given Queue input, it won't fire up the cells corresponding to Standing and Walking data and this will make it easier for the second classifier to recognize Queue data.Thus, the network encodes them into distinct values and they can be segregated.

Results

The following are the confusion matrix for Standing - Walking and Queue - Not Queue validation data respectively.

[[2362, 374],  
 [ 123, 2012]]

[[ 466, 204],  
 [ 192, 4679]]



**Training, Results and Analysis of Walking Queuing Classifier**

Since *Standing* class is very different from *Walking* and *Queuing*, we decided to focus on the problem of *Walking* and *Queuing*. Since *Walking* data had significantly more samples than *Queue,* we undersampled the data to make the samples equal. We used FCN and also experimented with different architectures, but the model did not converge and hence, stating the accuracy values is pointless. This can be attributed to the following reasons:

1. The similarity between the two classes. As explained in the analysis of data above, about 20% queue data is similar to walking data.
2. The length of the window is small, making it difficult to capture the characteristic nature of any time series.
3. The amount of data is small leading to overfitting.

**Possible future approaches**

Keeping in mind the problems encountered during the course of this classification problem, the following alternatives can be explored:

1. **Less Data Problem**: We can use a pre-trained HAR classifier and finetune it on our dataset.
2. **Increasing Window Length**: The classifier can be evaluated for longer window length.
3. **Dropping out samples that are common between classes**: There was probably some discrepancy in labeling the data. There are long stretches of time series (about 200 and more timesteps) in Queue data that corresponds to Walking data. Even a time series of length 200 produces about 165 samples of length 35 by the windowing technique. Such data needs to be dropped as it hampers the learning process.
4. **Exploring the cascading classifier**: Training the cascading classifier with a balanced class dataset is one way to proceed.
5. **Exploring other datasets**: Trying to solve the problem for this specific dataset can possibly make this problem more complicated than it actually is. So, exploring other datasets could help in moving forward.

FCN for D2

In order to solve the above shortcomings, a second dataset D2 was created that comprised of 3 classes - Walking, Climbing up and Climbing Down. FCN architecture with kernel sizes of 16,8,4 and number of filters 8, 16, 24 was used to train the model. The input data used was a time series of length 100 (2 seconds).

Results

The following confusion matrix was obtained:

[[ 20, 38, 173],

[ 0, 351, 0],

[ 0, 0, 351]])

The class labels are Climbing Up - 0, Climbing Down - 1 and Walking - 2. Climbing Down and Walking have been perfectly classified. This complies with the fact that the time series belonging to these classes are very distinct whereas Climbing Up class has been highly misclassified as Walking. The overall accuracy obtained is 77.38%.

Other architectures and more data could be explored. One of the pitfalls of using D2 was that the data was collected in very controlled conditions, so there was very little noise. Another pitfall was that it was collected by only one person. This led to very uniform data which is not possible in real life scenario.

Comparing model sizes

The hdf5 model sizes for the first 2 models are 3.3 Mb and 3.4 Mb. For the walking and queuing classifier, the size was 23Kb. For the 4th model on D2 dataset, the size was 97.6Kb.

CONCLUSION

Given two datasets, we were able to attain reasonably good results using Neural Network Models while restricting the size of the model and the data used. Since the datasets are self generated, a transfer learning approach could be tried out where the model is finetuned on our datasets. This will counter the problem of small dataset size. Another point to keep in mind is selecting the window length carefully while keeping the training and inference time low and giving enough information to the neural network at the same time. The dataset should not contain too much noise and if it does, proper preprocessing should be done. All these points will help in improving the generalizing capacity of the neural network.

REFERENCES

[1] [A Survey on Human Activity Recognition using Wearable Sensors](https://ieeexplore.ieee.org/abstract/document/6365160)

[2] [Deep learning for time series classification: a review](https://arxiv.org/abs/1809.04356)

[3] [Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline](https://arxiv.org/abs/1611.06455)

[4] [BOSS](https://www2.informatik.hu-berlin.de/~schaefpa/boss.pdf)

[5] [COTE](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7069254)