Human Activity Recognition

Using SLP and RBF Network

By Prateek Jain(1801127)

Abstract

This work report discusses the problem of Human Activity Recognition based on UCI dataset Human Activity Recognition using Smartphone. Various Machine Learning Models have been proposed for the same. This report proposes two models: Single Layer Perceptron model and Radial Basis Function Network, for classifying the given data into various activity labels. A new kernel function is also introduced.

Contents

1. Introduction

Smartphones are becoming increasingly powerful in terms of computational abilities. Today most of them are equipped with motion sensors such as accelerometer and gyroscope sensors.

Recognition of human activities can be carried out by exploiting the data collected from these sensors. These activities include standing,walking, laying, walking, walking upstairs and walking downstairs.

The dataset is based on experiments that have been carried out with a group of 30 volunteers within an age bracket of 19-48 years. Each person performed six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz were captured. The experiments have been video-recorded to label the data manually. The obtained dataset has been randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data. University Of California (n.d.)

2. Exploring the Dataset

2.1. Preparation.

The sensor signals (accelerometer and gyroscope) were pre-processed by applying noise filters and then sampled in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window). The sensor acceleration signal, which has gravitational and body motion components, was separated using a Butterworth low-pass filter into body acceleration and gravity.

The gravitational force is assumed to have only low frequency components, therefore a filter with 0.3 Hz cutoff frequency was used. From each window, a vector of features was obtained by calculating variables from the time and frequency domain. This is how the data was prepared. University Of California (n.d.)

The dataset contains 562 features. There are total 6 classes (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING).

2.2. Visualization.

Fig1 per class number of samples. Fig2 shows scree plot describing amount of variation each principal component covers. Fig3 describes the PCA score plot for the test dataset.

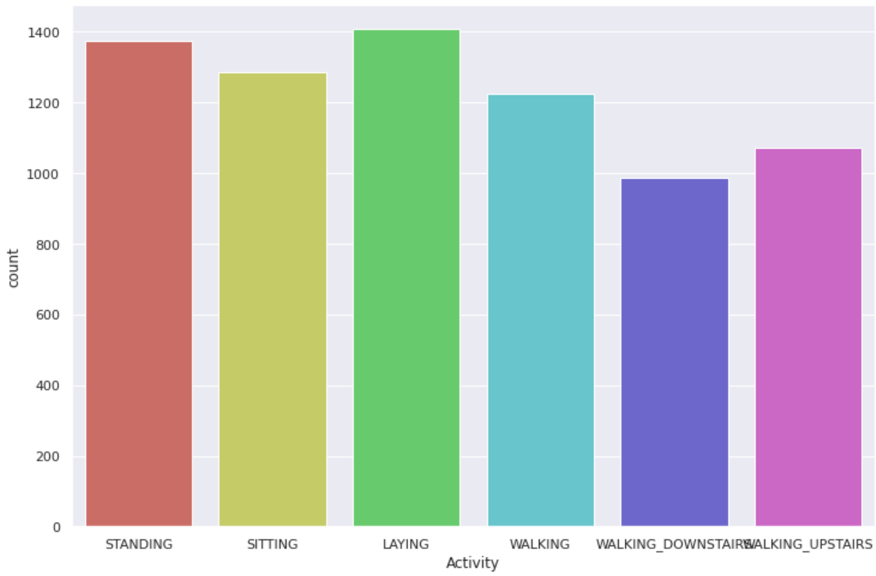

Figure 1. Number of samples per class

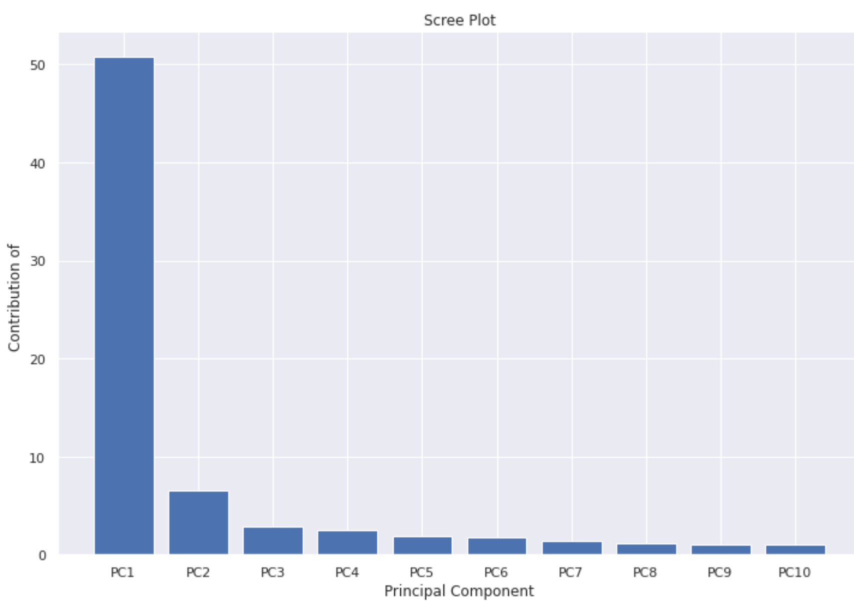

Figure 2. Number of samples per class

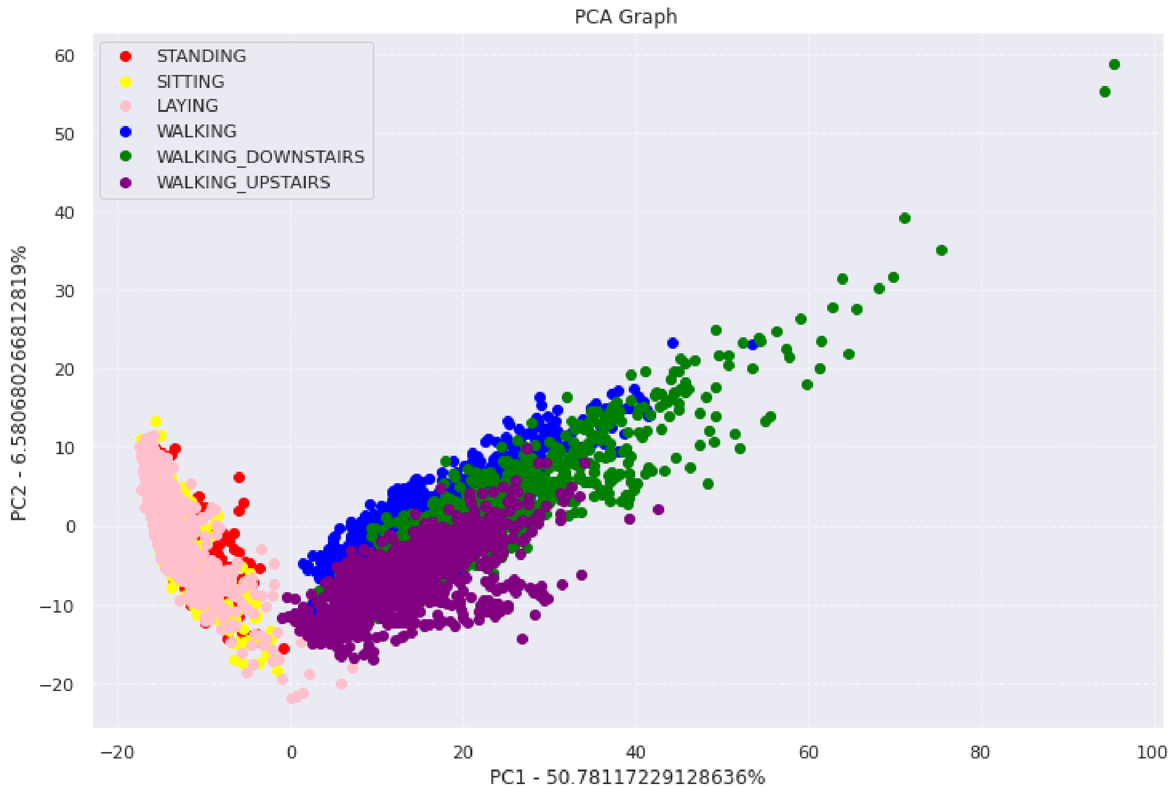

Figure 3. PCA plot

2.3. **Inference.** It can be inferred from the Fig1 is that class 'Laying' has the most number of samples present in the train set and class 'Walking Down' has the least number of samples.

From Fig2, we can infer that principal component 1 captures the most variation, around 51%. This percentage falls sharply for successive principal components.

Fig3 describes the projection of dataset on the principal components. From the graph, we can infer that on 2D plane, the projected samples are close to each other. Class 'WALKING DOWNSTAIRS' has some outliers and is more widely spread. Other than that, most of the points are clustered together. PC1 is able to separate 'LAYING' and 'STANDING' class samples from other class samples. Along PC2, much difference among classes is not there. Though the principal components are not able to capture much variations, they still give an idea about how the data might be spread.

3. Machine Learning Models Applied

This section discusses the machine learning models I have used on the given dataset, Single Layer Perceptron and Radial Basis Function Network. The architectures for the same have been described below.

Before applying the models, I removed the 'subject' feature from the dataset since it is irrelevant for model fitting and predictions. Also, I scaled the features such that the mean becomes zero and standard deviation becomes one for each of them.

Since there are no missing or null values in the dataset so downsizing of samples or approximation of missing values was not needed.

Based on the plot for the loading scores of the features, we can see that all the features are necessary, hence feature reduction is not performed here.

3.1. Single Layer Perceptron.

The model tries to approximate the following function:

$$y(X_j) = \sum_{i=1}^{n} (w_i x_i) + b$$

Here $y(X_j)$ is the output corresponding to $X_j = \{x_1, x_2, ..., x_n\}$ input sample. $W = \{w_1, w_2, .., w_n\}$ represent weights and b represent bias. For classification, this value is passed through an activation function to get a decision biundary that is less prone to outliers:

$$h(X_j) = Activation(y(X_j))$$

Sigmoid function is generally used as the activation function here:

$$h(x) = \frac{1}{1 + e^{-y(X)}}$$

The cost function used to update the weights is as follows:

$$J(W) = \sum_{k=1}^{m} (h_{X_k} - y'_k)^2$$

Here y'_k is the true value corresponding to k^{th} point and m is the total number of points. The weights are updated as follows:

$$w_i = w_i - \alpha(\frac{\delta(J(W))}{\delta w_i})$$

Here is the learning rate. Since there is only one output neuron in Single Layer Perceptron, it can be only be used for binary classification. A workaround for this is using multiple instances of the model ad training each one for one class label. The final output can be decided based on the outputs of individual models. This approach is called One-vs-All Classification model.

Here, one-vs-all SLP Classification model is used. There are 6 models in this one-vs-all classifier, each corresponding to one class label, hence 6 output neurons. Sigmoid function is used as activation function. The label predicted for an input sample is corresponding to the output neuron having the maximum value.

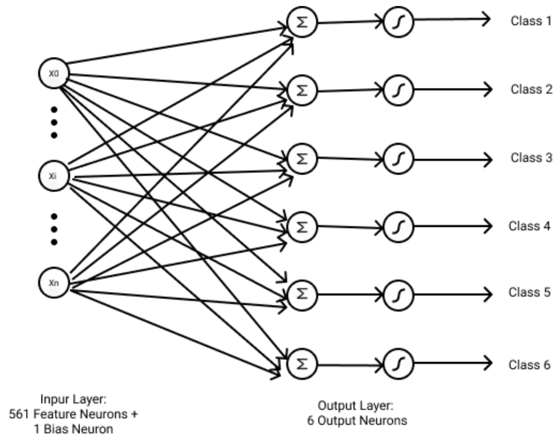

Figure 4. Single Layer Perceptron

3.2. **Radial Basis Function Network.** RBF Network solves the problem of classifying non-linearly separable patterns by proceeding in a hybrid manner in two stages:

1. In the first stage, the input features are transformed non-linearly to higher dimension space where they are likely to be linearly separable. This is based on Cover's theorem.
2. The second stage involves using the transformed features to train a classifier model.Haykin, 2009

Before transformation is applied, one more step is used which involves finding points that represent the regions in the dataset. For this step, K-Means algorithm is often used because it is fast and simple. However due to it's non-deterministic nature and tendency to getting stuck in local optima, it may not give the centroids that best represent the spread of the data. This problem is taken care of in the second stage of the RBF Network algorithm, where a classification model is trained to fit on the given data even though the centroids might not be the most optimum one.

Here, K-Means is used for finding centroids.

For non-linear transformation, two kernel functions have been explored:
One involves use of **traditional Gaussian RBF Kernel**:

$$h_i(X_j, C_i) = e^{-\gamma D(X_j, C_i)^2}$$

where

$$\gamma = \frac{1}{(2\sigma)^2}$$

where

$$\sigma = \frac{max(D(C_m, C_n))}{\sqrt{2TotalCenters}}$$

Here h_i is i^{th} transformed feature, X_j is j^{th} input sample, C_i is i^{th} centroid, and $D(x, y)$ represents distance function, here euclidean distance.

$$D(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

σ is calculated by finding the maximum distance between any two centroids.

Second involves a **new RBF Kernel**:

$$h_i(X_j, C_i) = e^{-\gamma D(X_j, C_i)}$$

All variables hold the same meaning, except for D(x,y) which represents manhattan distance here.

$$D(x, y) = \sum_{i=1}^{n} |y_i - x_i|$$

Here I have introduced a new kernel function. The intuition behind using this function comes from here Aggarwal et al., 2001. It describes the effect of high dimensionality on the effectiveness of distance metrics and prove that L1 norm i.e. Manhattan distance metric performs consistently better than L2 norm i.e. Euclidean distance metric in terms of measuring distances. Since the given data has high dimensionality, it is intuitive that the new kernel function will work atleast as good as the traditional one. This function can applied in other models as well which make use of distance metrics for finding similarities in case of high dimensionality data.

For the second stage, one-vs-all SLP model is used. The results have been documented for the same.

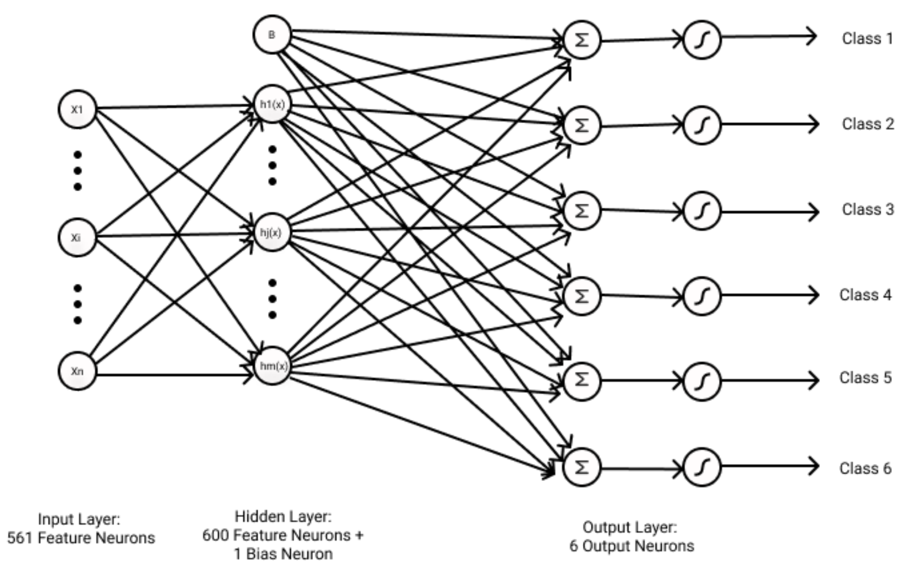

Figure 5. Radial Basis Function Network with traditional RBF kernel function

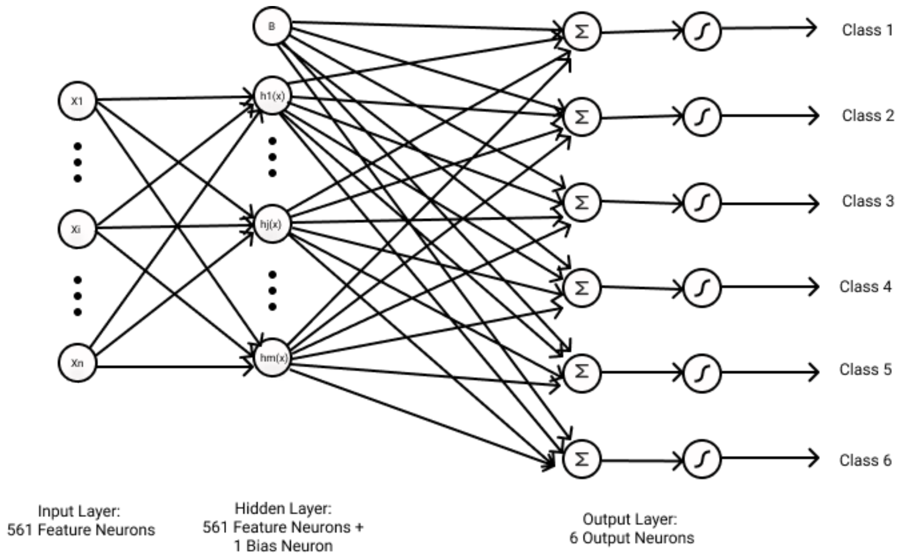

Figure 6. Radial Basis Function Network with new kernel function

4. Results

The results presented here corresponds to the best version for each model.

Best Hyperparameters			
Model	Learning Rate	Number of epochs	Number of centroids
One-Vs-All SLP	0.01	45	Not applicable
RBF Network with orginal kernel function	0.5	500	600
RBF Network with new kernel function	$1e-06$	500	561

Accuracy Scores		
Model	Train Accuracy	Test Accuracy
One-Vs-All SLP	99.06%	95.45%
RBF Network with traditional kernel function	98.81%	96.19%
RBF Network with new kernel function	98.85%	96.67%

5. Inference

One-vs-all SLP model fits the dataset very well and gives very high test accuracy indicating that the data is linearly separable since SLP draws linear hyperplane for distinguishing the classes. Results for RBF Network suggest that the new kernel function performed slightly better than the traditional function on the test dataset. Moreover, the new function is able to achieve this accuracy for less number of transformed features compared to traditional function.

6. Summary

The report presents two models for performing classification on the given dataset. Results show that the class samples are separable because a one-vs-all SLP model works is performing well on the given dataset. RBF Network is also used with variations in it's steps and the results are presented for the same. I have made an attempt to bring novelty in the implementation of RBF Network by introducing a new kernel function.

7. Future Works

The paper Aggarwal et al., 2001 discusses how Manhattan Distance metric and Fractional Distance metric can significantly improve the performance of models that calculate similarity between data points such as K-Means. This presents an opportunity to improve the performance of RBF Networks further by incorporating these distance metrics in the first stage where K-Means is used to find the centroids. I tried to implement it however the computation was too slow to derive any inferences. Hence I will try and improve it's performance by using different language and environment.

References

Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In J. Van den Bussche & V. Vianu (Eds.), *Database theory — icdt 2001* (pp. 420–434). Springer Berlin Heidelberg.

Haykin, S. (2009). *Neural networks and learning machines*. Pearson Edu.

University Of California, I. (n.d.). *Human activity recognition using smartphones data set*. https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones#