iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** XI **Month of publication:** November 2020

DOI: https://doi.org/10.22214/ijraset.2020.32065

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue XI Nov 2020- Available at www.ijraset.com

Comparative Analysis of Different Machine Learning Classifiers on Human Activity Recognition Applications

Hardik Kathuria[1], Poras Khetarpal[2], Arvind Rehalia[3], Saloni Gupta[4]
[1, 4]*Student,* [2]*Asst. Professor,* [3]*Associate Professor, Bharati Vidyapeeth's College of Engineering, New Delhi, India*

Abstract: ***Human Activity Recognition is an active field of research and is the basis for a wide range of modern day gadgets such as fitness bands, sleep-tracking devices etc.[6] In this research based project, we present our efforts of executing a performance comparison in order to determine the most effective machine learning algorithm for Human Activity Recognition applications. Our objective in this project is to train Artificial Neural Networks, Random Forest Classifiers, k-NN and Support Vector Machine algorithms over an HAR dataset (Human Activity Recognition using Smartphones Dataset) which is publicly available. We then graphically compare the accuracy achieved by each of the algorithms thereby finding out the approach that should be used by developers in Human Activity Recognition applications (such as fitness tracking platforms and home automation platforms) in order to improvise and contrive their platforms. It was observed that Support Vector Machines demonstrated the best performance with a staggering accuracy and is recommended for HAR applications.***

I. INTRODUCTION

Nowadays, the use of smartphones has become deeply intertwined in our day to day lives. With smartphones being excessively used for wide range of applications such as gaming, information transfer, instant communication; it is also giving rise to Human Activity Recognition applications such as health monitoring, fall detection, context-aware mobile applications, human survey system and home automation etc. Smartphone-based activity recognition system is an active area of research owing to implications for a modern lifestyle and an urban design. As for example, for health reasons, some people show keen interest in tracking the amount of time they usually spend sitting down or the number of stairs they have climbed each day. For home automation, walking up the stairs or sitting on the sofa could trigger the lights or television to turn on. Human Activity Recognition Devices find their current applications in rapidly evolving fitness bands, smartphone applications, smart watches and medical research.[10,11] Understanding human activities has created a demand in health-care domain, especially in rehabilitation assistance, physiotherapist assistance, elder care support services and cognitive impairment. In a world of today, where almost every single person owns a smartphone, gathering the information relevant for Human Activity Recognition has not remained a difficult task as these devices are already equipped with embedded motion sensors (e.g. accelerometer, gyroscope, magnetometer, GPS, etc.) that can be used for activity detection. Sensors will record and monitor the patient‘s activities and report automatically when any abnormality is detected, so, a huge amount of resources can be saved. Other applications such as human survey system and location indicator are greatly benefitting from this study. Training process is always necessary when a new activity is added in to the system. The same algorithm parameters are needed to be trained and fine-tuned when the algorithm runs on different devices with various built-in sensors. However, labeling a training data (time-series data) is a time consuming procedure and it is not always possible to label all the training data by the users. As a result, we present a performance comparison among four machine learning algorithms in order to determine the most efficient algorithm for Human Activity Recognition problems. Unlike speech recognition, there is no grammar and strict definition for human activities. This causes twofold confusions. On one hand, the same activity may vary from subject to subject, which leads to the intraclass variations. The performing speed and strength also increases the interclass gaps. On the other hand, different activities may express similar shapes (e.g., using a laptop and reading). This is termed as interclass similarity which is a common phenomenon in HAR. Accurate and distinctive features need to be designed and extracted from the dataset to deal with these problems.

The dataset we deal with in the project is 'Human Activity Recognition using Smartphones Dataset' taken from UCI Machine Learning Repository. It is a Classification dataset with 561 attributes/features, 10299 instances, and 6 classes. The dataset consists of elements based on experiments conducted on a group of 30 volunteers within an age group of 19-48 years. They performed a series of activities comprising of six basic activities: three static postures (standing, sitting, lying) and three dynamic activities (walking, walking downstairs and walking upstairs). All the participants were wearing a smartphone (Samsung Galaxy S II) on the waist during the experiment execution. The experiments were video-recorded to label the data manually. The obtained dataset was randomly divided into two sets, where 70% of the volunteers were selected for generating the training data and 30% to test the data.

A. Motivation

The main motivation of writing this paper is to show in detail the performance comparison of Artificial Neural Network, K-NN, Random Forest Classifiers and Support Vector Machine algorithms over an HAR dataset. Most of the papers in the literature have used only two or three types of algorithms, and the comparative analysis part is missing or not comprehensive enough.[13] Further, to the best of our knowledge there exists no paper in the literature which has done in detail the robustness comparison of the four machine learning algorithms (ANN, K-NN, SVM and RFCs) specifically for a smartphone dataset. Our analysis is tailored for smartphone applications which are in high demand today. Furthermore, many papers have employed new powerful algorithms for designing Human Activity Recognition applications, but many of these algorithms are quite complex which makes the analysis quite difficult and are also quite difficult to implement.

II. WORKING METHODOLOGY

We used multiple approaches to classify our dataset into the required classes. But, before doing so, we performed Principal Component Analysis (PCA) in order to understand our dataset better. From the studies implemented in [1], we observe that many features of the available 561 features can be reduced without drastically reducing the performance. However, removing the features randomly is not particularly useful for reducing the number of features used in the model. Since, the data only varies in a few dimensions, we can use PCA to project the data onto the plane of maximal variance and visualize the data. Figure1 shows this projection, in a scatter plot. The plot gives distribution of different data points scattered graphically. Each class label for transition is represented by a different color on the graph.

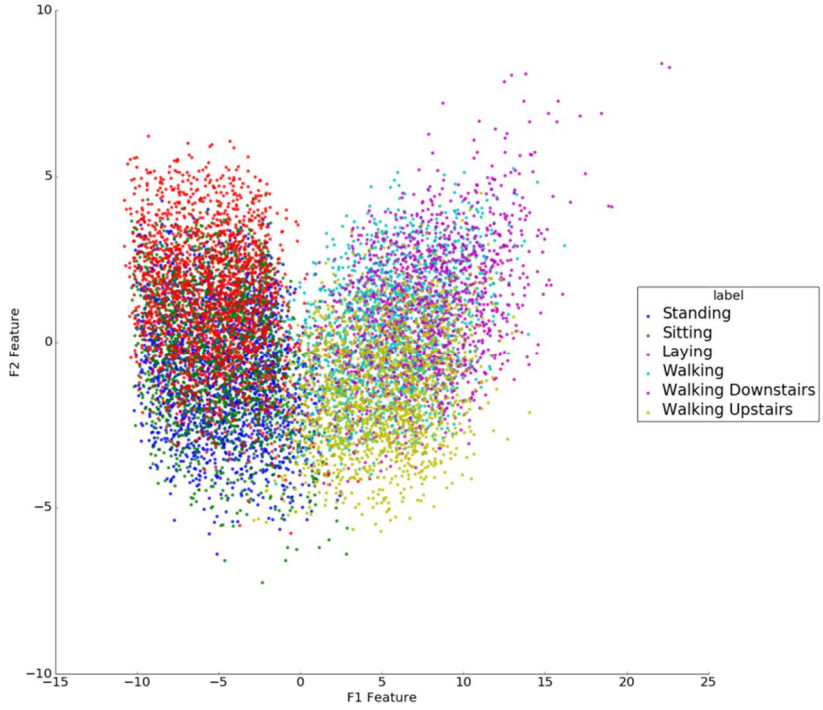


Figure1: Scatter Plot of class labels wrt. the features obtained using PCA

From Figure1, we observe that the class labels are fairly well grouped even in two dimensions and hence decision boundaries created using classification algorithms like Artificial Neural Networks(ANN), k-Nearest Neighbors(k-NN), Support Vector Machines(SVM) are expected to work well for the available dataset. While it is expected to achieve good performance using selected features only, we however used the raw dataset available to us to build our models. There are a few advantages of using the raw data, one of which being some of the features in the dataset available to us might be difficult to generate in real-world settings, especially given the memory, processor, and battery constraints of multitasking mobile devices. Creating the models based on selected features is left as a future scope to this paper. To establish a baseline, we first used ANN and k-NN on the raw data. We then moved on to comparatively complex algorithms namely SVM and Random Forest Classifier.

A. *Artificial Neural Networks*

Artificial Neural Network (ANN) are computing systems, which are inspired by the biological neural networks that are a constituent of animal brains. Such system learn (progressively learns to improve) to do various tasks by taking into account examples, generally without task specific programming. ANN operates on hidden states, which are similar to neurons. Each of the hidden state has a probabilistic behavior. A grid of such hidden state forms a bridge between the input and the output. A simple neural network algorithm tries to mimic the relationship between the input and the output variables and learn by the examples. The network designed was a four layered neural network:

1) Input layer with density equal to the number of features in our dataset i.e. **561**.
2) 3 hidden layers, 1 having density equal to **48**, the other having density equal to **24** and the last one having density equal to **12.**
3) The output layer with density equal to the number of classes in our dataset i.e. **6**.

ReLU also known as rectified linear unit was used as the activation while training our models for hidden layers. This results in much faster training for large networks. It has output 0 if the input is less than 0, and raw output otherwise i.e. if the input is greater than 0, the output is equal to the input. The mathematical definition of a ReLU is

$$h=max\ (0,a)$$
$$where\ a=Wx+b$$

Relu is used as the activation function as it is faster, and more biologically inspired. The other major benefits of ReLUs are sparsity and a reduced likelihood of vanishing gradient. One major benefit is the reduced likelihood of the gradient to vanish. This arises when a>0a>0. In this regime the gradient has a constant value. In contrast, the gradient of sigmoids becomes increasingly small as the absolute value of x increases. The constant gradient of ReLUs results in faster learning. The other benefit of ReLUs is sparsity. Sparsity arises when a≤0a≤0. The more such units that exist in a layer the more sparse the resulting representation. Sigmoids on the other hand are always likely to generate some non-zero value resulting in dense representations. Sparse representations seem to be more beneficial than dense representations. Adam optimizer was used in order to optimize the synaptic weights of the model. Adam makes use of stochastic gradient descent while carrying out optimization i.e.

$$\omega := \omega - \alpha \nabla Q(\omega) = \omega - \alpha \sum_{i=1}^{n} \nabla Q_i(\omega)/n$$

where the parameter ω which minimizes Q(ω) is to be estimated. Each summand function Qi is typically associated with the i-th observation in the data set that is used for training. α is the step size (also called the *learning rate* in machine learning). At the output layer we have used the softmax function. It makes the outputs of each unit to be between 0 and 1 also it divides each output such that the total sum of the outputs is equal to 1. The Artificial Neural Network was fitted to the dataset in 16 epochs i.e. the entire training dataset was passed to the algorithm 16 times so as to optimize the weights of the network and fit the network to the training dataset.

B. *K-Nearest Neighbors*

The k-Nearest Neighbors algorithm (k-NN) is a non- parametric method i.e. it does not make any underlying assumptions about the distribution of data and is used for classification purposes and regression. In k-NN classification algorithm, the output is a class label based on voting. An object is classified (i.e. assigned a class label) based on the class label of its nearest neighbors. In this algorithm, firstly the distance of the object is calculated from all the other points in the dataset. K is the number of neighbors which vote for the object classification. (k is always a positive integer and usually small). After that, the object is assigned the appropriate class label which is the most common or the most frequent one amongst its k nearest neighbors. The performance of a k-NN classifier is primarily determined by the choice of K as well as the distance metric applied [2-5]. The estimate is affected by the sensitivity of the selection of the neighborhood size K, because the radius of the local region is determined by the distance of the Kth nearest neighbor to the query and different K yields different conditional class probabilities. If K is very small, the local estimate tends to be very poor owing to the data sparseness and the noisy, ambiguous or mislabeled points. In order to further smooth the estimate, we can increase K and take into account a large region around the query. Unfortunately, a large value of K easily makes the estimate over smoothing and the classification performance degrades with the introduction of the outliers from other classes.[6] To deal with the problem of finding the optimal value of K, we trained 50 models, each with value of K varying from 1 to 50. While training our models, we used Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.

$$Euclidian\ Distance = \sqrt{\sum_{i=1}^{k} (Test\ Instance - Training\ Set[x])^2}$$

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue XI Nov 2020- Available at www.ijraset.com

For the optimal model the value of k was observed to be 23. A drawback of the "majority voting" classification occurs when the class distribution is skewed that is, examples of a more frequent class dominate the prediction of the new example, because they tend to be common among the *k* nearest neighbors due to their large number. To overcome this problem the classification can be weighted taking into account the distance from the test object to each of its *k* nearest neighbors. However the distribution of our dataset wasn't skewed, we didn't have any need to perform any such technique of adding weights.

C. Random Forest Classifier

Random Forest Classifier(RFC) is a bagging based ensemble learning method that finds its use in classification that operates by making a number of decision trees at the training time and the output obtained is the mode or the mean of the individual trees depending upon the operation it is performing (classification or regression). In this algorithm, a subset of the features and samples from the dataset (along with replacement) is chosen for training an individual decision tree classifier and the output class predicted for each data point is the mode of the classes (classification) predicted by all the individual trees. By taking a random subset of features, the algorithm tries to make sure that the predictions from all the sub-trees have got less correlation between them. Random decision forests are better than decision trees in the case of overfitting to their training set.[7,8]

The entire RFC model is like a black box and hence should be used judiciously. Hyperparameter tuning is one of the very important aspects of the model because in RFC as the number of trees, maximum depth of trees, and the number of features to be considered when looking for a spilt, all have to be specified beforehand.[9]

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable. Decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

The algorithm selection is also based on type of target variables. Let's look at the four most commonly used algorithms for making a split in decision tree are 'Gini index split', 'Chi square', 'Information gain' and 'reduction in variance'. We however used 'Gini index spilt' method which is based on the formula:

$$\boldsymbol{GINI\ (s,t)\ = GINI\ (t) - P_L\ GINI\ (t_L) - P_R\ GINI\ (t_R)}$$

Where s is split, t is node, GINI (t) is Gini Index of input node t, P_L is Proportion of observation in Left Node after split s, GINI (t_L) is Gini of Left Node after split s, P_R is Proportion of observation in Right Node after split s, GINI (t_R) is Gini of Right Node after split s. GINI index for a given node t is created as:

$$\boldsymbol{GINI\ (t)\ =\ probability\ for\ success^2\ +\ probability\ for\ failure^2}$$

In Random Forest classifier bias and variance are two main factors which should be taken care of. There should be a proper trade-off between bias and variance as randomization increases bias but makes it possible to reduce the variance of the corresponding ensemble model. So the problem is to find the right trade-off.

To train our model, GridSearchCV was used to find the optimal parameters for the Random Forest Model. The parameters used are n_estimators which tells about the number of trees to be used while creating the model, max_depth which defines the maximum depth for the tree. If mentioned none, then nodes of the trees are expanded until all leaves are pure, and max_features which is the number of features to consider when looking for the best split, The search for a split does not stop until at least one valid partition of the node samples is found. We trained 18 different models using different combinations of values of n_estimators, max_depth, and max_features.

The values of n_estimators considered were 10, 100, and 1000. The values of max_depth considered were 3, 6, and 9. The arguments considered for max_features were 'auto' and '$\log_2$' .The 'auto' feature simply take all the features which make sense in every tree, it does not put any restrictions on the individual tree. The '$\log_2$' feature on the other hand will only consider $log_2^{no.of\ features}$ in an individual run for an individual tree.

To train these 18 models, 2/3rd of dataset was used for training purpose and the remaining 1/3rd as validation dataset. Of the 18 models, the best model found used the following hyperparameter values:

1) n_estimators (Number of trees in the forest) = 1000
2) max_depth (Maximum depth of the tree) = 9
3) max_features (Number of features to consider when looking for the best split) = log2

D. Support Vector Machine

Support Vector Machine is a type of supervised learning model with associated learning algorithms that analyse data used for classifying and regression evaluation. An SVM model is a characterization of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as far away as possible. New examples are then mapped into the same space and predicted to belong to a category depending on which side of the gap they fall. Support vector machine is defined by the separating hyperplane which categorizes new examples. The algorithm outputs an optimal hyperplane which is a line in a two dimensional space, dividing a plane in two parts where in each class lay in either side. So the hyperplane depends upon the distribution of data. Finding perfect hyperplane for millions of training data set takes lot of time so there are parameters used in svm used to find the best hyperplane. So there are tuning parameters in svm classifier which are regularization parameter and gamma and varying these we can achieve non-linear classification line with more accuracy in reasonable amount of time. One more parameter kernel is used which defines whether a linear or non-linear separate is needed.

To train our model, GridSearchCV was used to find the optimal parameters for the Support Vector Machine. We trained 12 different models using different combinations of values of C, gamma, and kernel. 4 linear kernels were built using 4 different values of C. The values of C considered were 1,10,100, and 1000. 8 rbf models were built using 4 different values of C and 2 different values of gamma. The values of C considered were same as those considered while modelling linear kernels and the values of gamma considered were 0.001 and 0.0001.

To train these 12 models, 2/3rd of dataset was used for training purpose and the remaining 1/3rd as validation dataset. Of the 12 models, the best model found used the following hyperparameter values:

1) C = 1000

2) gamma = 0.001

3) kernel = rbf

In our model the kernel used is rbf which is used for non-linear separation. C is the regularization parameter which tells the SVM optimization how much you want to avoid misclassifying each training example. The use of gamma parameter is to define how far the influence of a single training example reaches.

Rbf kernel used by SVc classifier is,

$$K(x, x') = exp(\frac{-||x-x'||^2}{2\sigma^2})$$

Where $||x - x'||2$ is the squared Euclidean distance between two data points x and x′.

C and Gamma are the symbolic parameters for a nonlinear Support Vector Machine with a Gaussian radial basis function (rbf) kernel where ‘C’ represents the parameter for the soft margin cost function that controls the influence of each individual support vector. ‘Gamma’ is the parameter which controls variance of the model.

III. RESULTS AND ANALYSIS

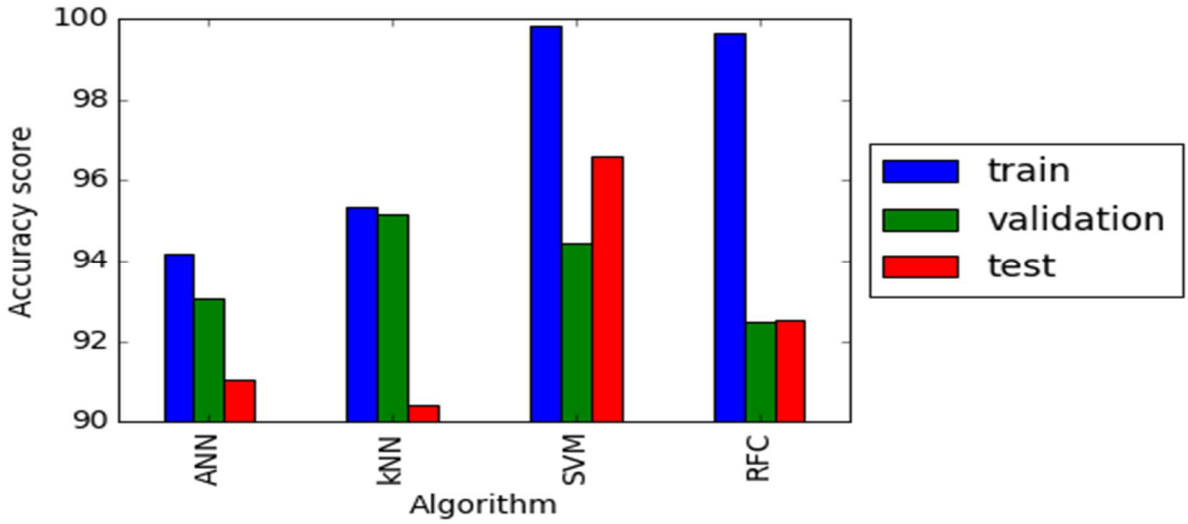


Figure2: Accuracy scores (in %) using various approaches

Algorithm	Train	Validation	Test
Artificial Neural Network	94.1928	93.0366	91.0417
K - Nearest Neighbors	95.3299	95.1380	90.3970
Support Vector Machines	99.8096	92.4505	96.5727
Random Forest Classifier	99.6599	92.4918	92.5008

Table1: Results on the dataset (in %) using various approaches

A. Artificial Neural Network

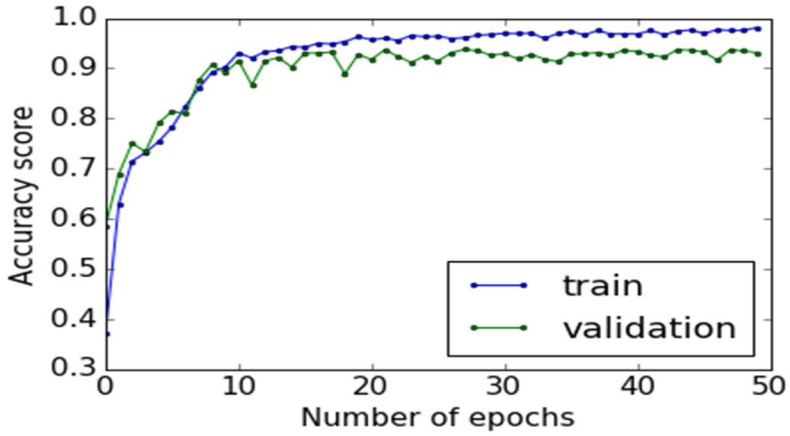


Figure3: Accuracy score for Artificial Neural Network

As can be seen from the model complexity curve in Figure3, the model didn't perform well till epochs less than 10 and started to overfit after 20 epochs. The model showed good accuracy results with no signs of overfitting, if trained between 10-20 epochs. Hence, the final model was trained in 16 epochs. The final Artificial Neural Network model was a 5 layered Neural Network of 561 X 48 X 24 X 12 X 6 architecture. Dropouts of 10% were also added in all 3 hidden layers to combat overfitting. The resultant model produced a training accuracy score of 94.1928%, a validation accuracy score of 93.0366% and a test accuracy score of 91.0417%.

B. k-Nearest Neighbors

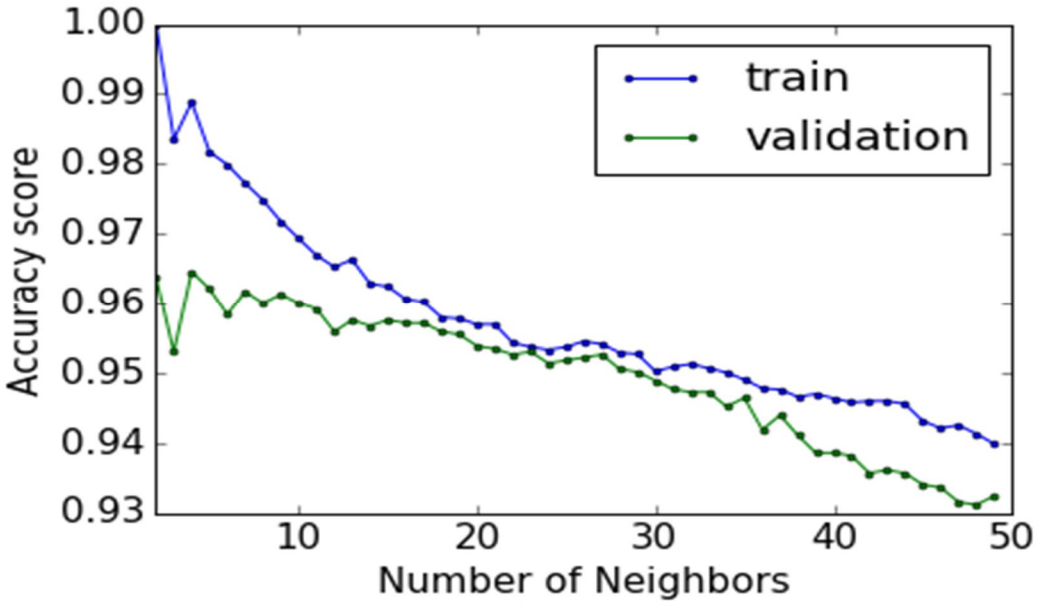


Figure4: Accuracy score for k-Nearest Neighbors

As can be seen from the model complexity curve in figure4, using less than 20 neighbors results in a complex model that works well for the training data but fails to work well for validation or testing data. Using more than 30 neighbors results in a simpler model with a smoother decision boundary that fails to work well for either of the training or validation data. This means, the model overfits if less than 20 neighbors are used to train the model and underfits if more than 30 neighbors are used to train the model. However setting the value of neighbors between 20-30 results in a model that works well for both training and validation data. Hence, the final model was trained using 23 neighbors to achieve optimal results. The model so trained produced a training accuracy of 95.3299%, a validation accuracy of 95.1380%, and a testing accuracy of 90.3970%.

Though ANN and k-NN gave good accuracy results, we moved on to RFC and SVM which are comparatively complex algorithms in order to improve our results further.

C. Random Forest Classifier

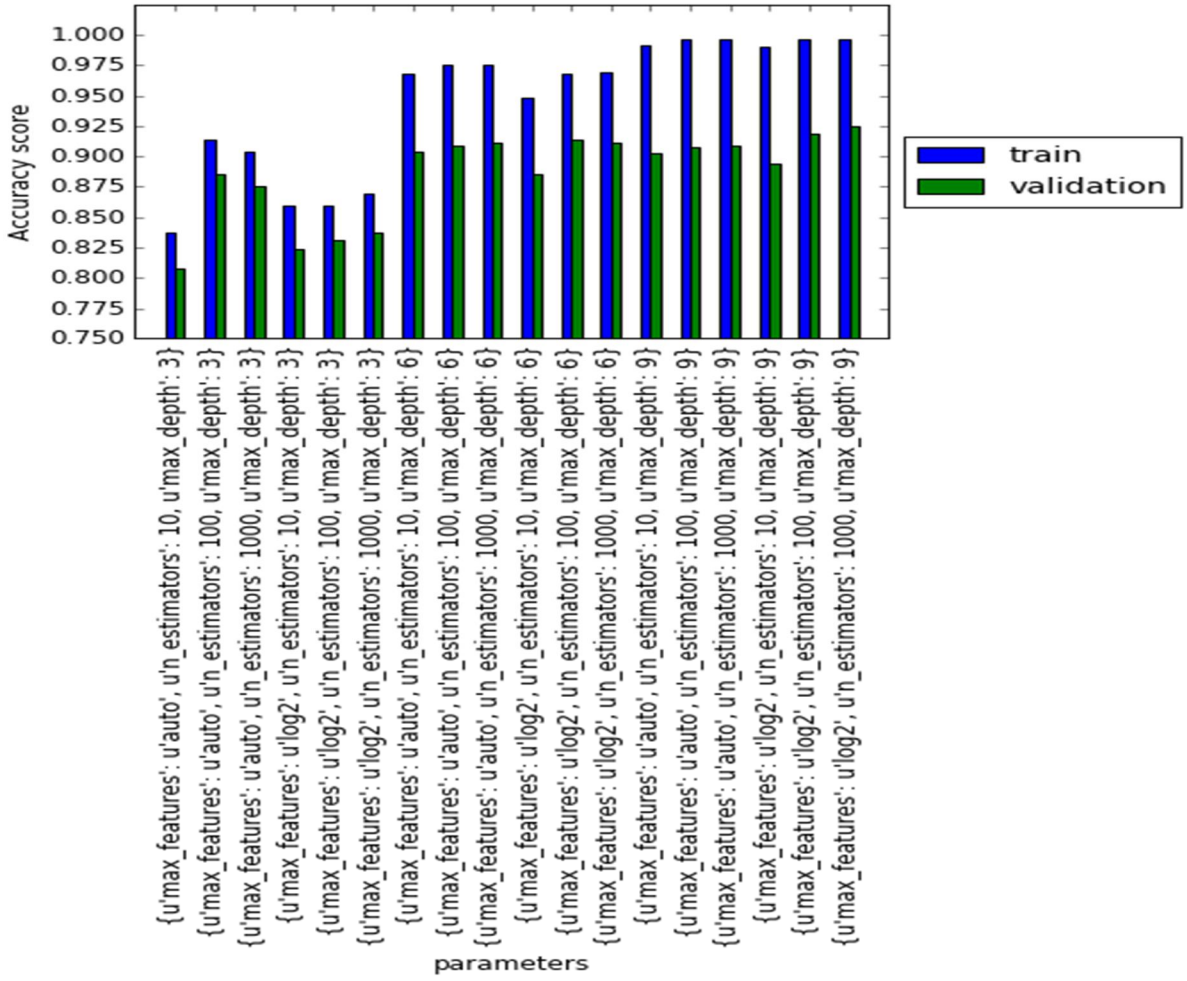


Figure5: Bar plot comparing different RFC models trained using different sets of hyperparameters

As can be seen from figure5, the model trained using n_estimators (number of trees) = 1000, max_depth (maximum depth for a tree) = 9, and max_features (number of features to be considered while building a tree) = $\log_2$ gave us the best results. The model so trained produced the training accuracy of 99.6599, a validation accuracy of 92.4918, and test accuracy of 92.5008.

One more observation that can be done form figure5 is that almost all of the models that produce more than 97% of accuracy on training set are barely able to cross validation accuracy of 92%. This is a clear depiction of overfitting. To overcome the problem of overfitting we need to increase the number of trees in the model or increase the maximum depth of existing tress. Both of these solutions might help in overcoming overfitting,but will make the model computationally expensive. Hence, we decided to move on to SVM in order to create a better model.

D. Support Vector Machine

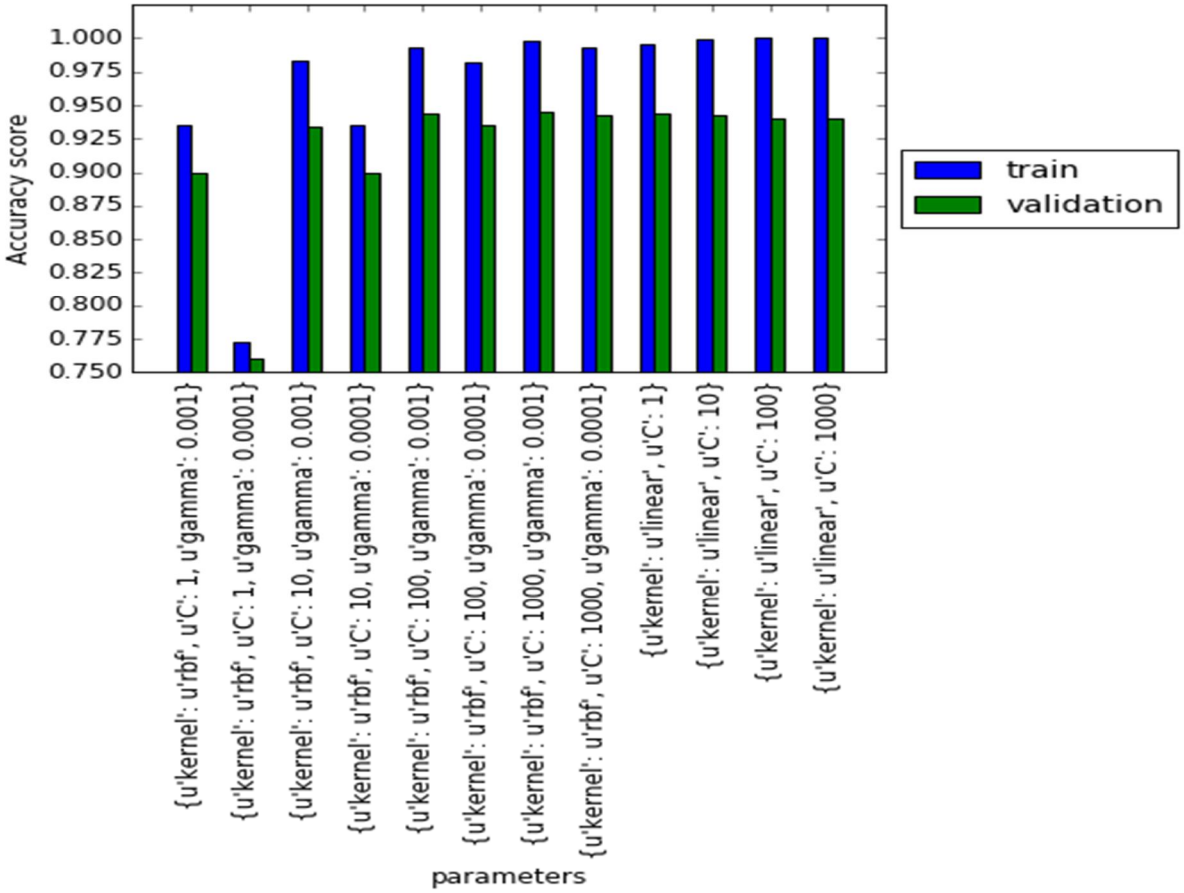


Figure6: Bar plot comparing different SVM models trained using different sets of hyperparameters

As can be seen from the bar plot in figure6, the model trained using rbf kernel with C= 1000 and gamma= 0.001gave us the best results. The model so trained produced a training accuracy of 99.8096%, validation accuracy of 92.4505% and test accuracy of 96.5727%. Achieving a test accuracy of 96.5727% is a very good result, and best among all of the algorithms used in the paper. After this point, we decided to conclude with our paper.

IV. CONCLUSION AND FUTURE SCOPE

ALGORITHM	Easy to understand ?	General Predictive Accuracy	Predictive Accuracy in our models	Training Speed	Amount of parameter tuning needed (excluding feature selection)
k-Nearest Neighbors	Yes	Lower	Lower	Fast	Minimal
Artificial Neural Network	Somewhat	High	Higher	Slow	Maximum
Random Forest Classifier	No	Higher	Higher	Slowest	Some
Support Vector Machines	No	Higher	Highest	Slower	Some

Table2: General trends observed about different machine learning classifiers studied in the paper

After comparing the results achieved by each of the algorithms performed in this paper, we reach to the conclusion that Support Vector Machines algorithm worked best for 'Human Activity Recognition using smartphones' dataset and hence should be used by developers in Human Activity Recognition applications (such as fitness tracking platforms and home automation platforms) in order to improvise and contrive their platforms.

Support Vector Machines algorithm worked extremely well for our 'Human Activity Recognition using smartphones' dataset, with a phenomenal accuracy of 96.572% but another fact worth noticing is that the algorithm took 4.5 minutes to complete a total of 48 fits. This might seem a short time span at first glance but in a world of today where we are dealing with petabytes of big data, a time span of around 5 minutes for only 48 fits is gigantic in itself.

Hence, more work is needed to be done in order to design and develop a faster and efficient model.

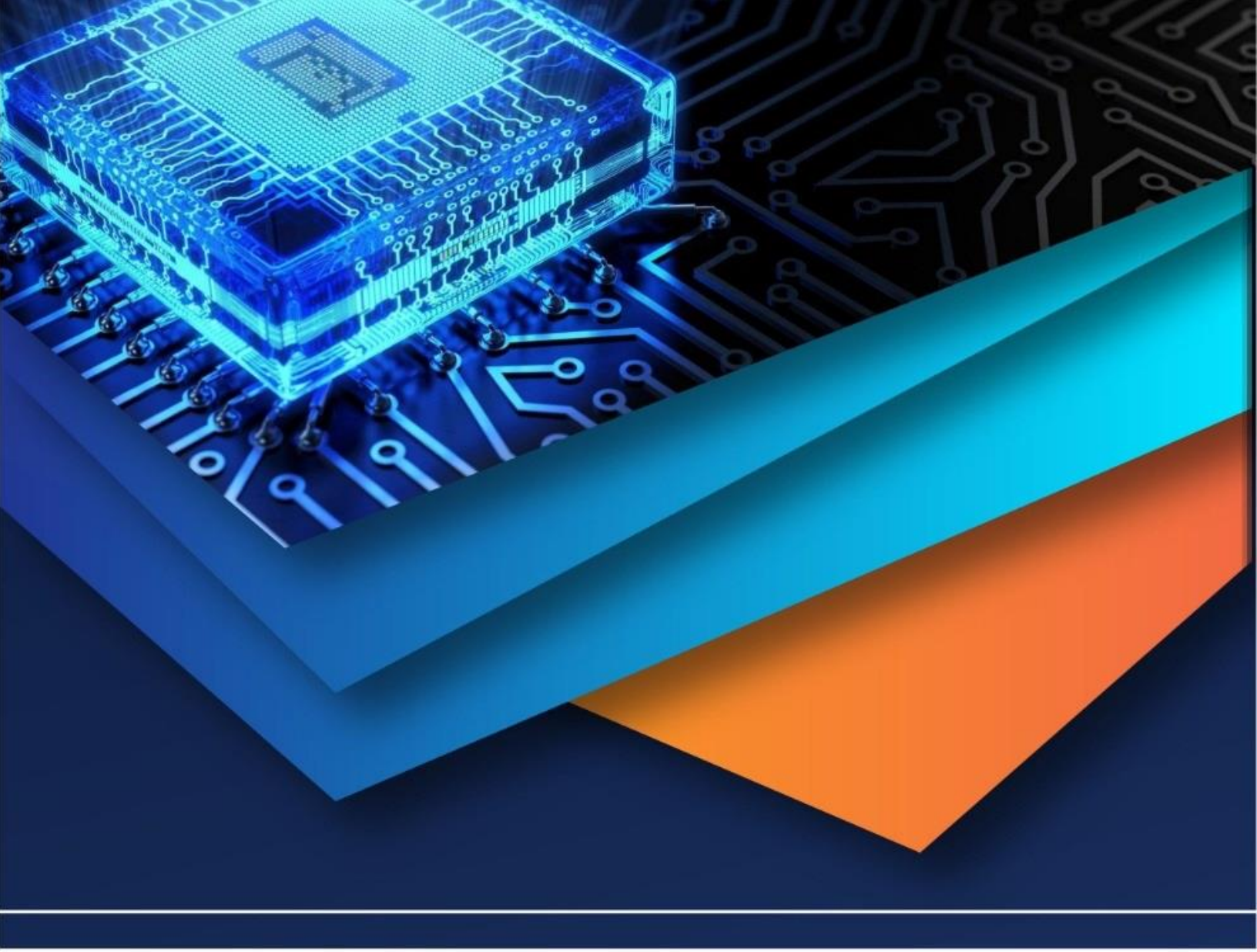
A Hidden Markov Model is said to be useful in capturing information in the transitions between activities and the time history of the estimates. K-means clustering algorithm can be used as the emission model for the HMM as it allows multiple clusters per activity. This approach might help us in creating a less computationally expensive algorithm without compromising much on the accuracy.

Another thing that can be tried in future is reducing the number of features to increase the computational efficiency. The data can be processed in various ways to find out the useful features among 561 available features and our models can be trained using those useful features only.

Postural Transitions (PTs) are transient movements that discuss the change of state from one static posture to the other. In several Human Activity Recognition (HAR) systems, these transitions cannot be sidelined owing to their noticeable incidence concerning the duration of other Basic Activities (BAs). In the future, we can vary the current models and put forward a smartphone-based HAR system that elucidates the occurrence of postural transitions. Doing so may raise the system accuracy by avoiding fluctuations in the classifier.

REFERENCES

[1] Activity Classification with Smartphone Data Matt Brown, Trey Deitch, and Lucas O'Conor
[2] Fritz, J. (1975) "Distribution-free exponential error bound for nearest neighbor pattern classification", IEEE Trans. Inform. Theory, 21: 552–557.
[3] Fukunaga, K. & Hostetler, L. (1975) "k-nearest-neighbor Bayes risk estimation", IEEE Trans. Information Theory, 21(3): 285-293.
[4] Gil-Garcia, R. & Pons-Porrata, A. (2006) "A New Nearest Neighbor Rule for Text Categorization", Lecture Notes in Computer Science 4225, Springer, New York, 814–823.
[5] Guo, G., Wang, H., Bell, D., Bi, Y. & Greer, K., (2006) "Using KNN Model for Automatic Text Categorization", Soft Computing –A Fusion of Foundations, Methodologies and Applications 10(5): 423–430.
[6] Application of K-Nearest Neighbor (KNN) Approach for Predicting Economic Events: Theoretical Background Sadegh Bafandeh Imandoust And Mohammad Bolandraftar
[7] Svetnik, A. Liaw, C. Tong, J. Christopher Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and QSAR modeling," Journal of Chemical Information and Computer Sciences, vol. 43, no. 6, pp. 1947–1958, 2003.
[8] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[9] Random Forest: A Review Eesha Goel*, Er. Abhilasha Computer Science & Engineering &GZSCCET Bhatinda, Punjab, India
[10] Energy Efficient Smartphone-Based Activity Recognition using Fixed-Point Arithmetic Davide Anguita, Alessandro Ghio, Luca Oneto (Universit`a degli Studi di Genova, Genoa, Italy {davide.anguita, alessandro.ghio, luca.oneto}@unige.it) Xavier Parra, Jorge L. Reyes-Ortiz (Universitat Polit`ecnica de Catalunya, Vilanova i la Geltr´u, Spain xavier.parra@upc.edu, jorge.luis.reyes@estudiant.upc.edu)
[11] Human Activity and Motion Disorder Recognition: Towards Smarter Interactive Cognitive Environments Jorge L. Reyes-Ortiz1,2, Alessandro Ghio1, Davide Anguita1, Xavier Parra2, Joan Cabestany2, Andreu Catal`a2 1- Universit`a degli Studi di Genova - DITEN. Via Opera Pia 11A, I-16145, Genoa, Italy. 2- Universitat Polit`ecnica de Catalunya - CETpD Rambla de l'Exposici´o 59-69, 08800, Vilanova i la Geltr´u, Spain.
[12] A Public Domain Dataset for Human Activity Recognition Using Smartphones Davide Anguita1, Alessandro Ghio1, Luca Oneto1, Xavier Parra2 and Jorge L. Reyes-Ortiz1,2 1- University of Genova - DITEN. Via Opera Pia 11A, I-16145, Genova, Italy. 2- Universitat Politecnica de Catalunya - CETpD ` Rambla de l'Exposicio 59-69, 08800, Vilanova i la Geltr ´ u, Spain.
[13] A Comparative Study of Random Forest & K – Nearest Neighbors on HAR dataset Using Caret Kella BhanuJyothi1 , K Hima Bindu2 and D. Suryanarayana3 1PG Scholar, CSE, Vishnu Institute of Technology, Bhimavaram, A.P, India 2,3Professor, CSE, Vishnu Institute of Technology, Bhimavaram, A.P, India
[14] Human Activity Recognition with Smartphones Rao Fu Computing Science Simon Fraser University raof@sfu.ca Yao Song Computing Science Simon Fraser University songyaos@sfu.ca Weipu Zhao Computing Science Simon Fraser University weipuz@sfu.ca



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY