Machine Learning Methods for Human Activity Recognition Using Smartphones

Ervin Pangilinan[*1] **Gomathi Lakshmanan**[*1] **Lauren Bailey**[*1]

In this paper, we introduce a full pipeline, demonstrating pre-processing, processing, and post-processing techniques for the samples in the Human Activity Recognition Dataset provided by the University of California, Irvine. The pre-processing phase consists of standardizing the initial dataset and then reducing the same dataset through Fisher's Linear Discriminant or Principal Component Analysis. We then visualize this dataset with tSNE. After dimensionality reduction, we perform classification through k-Nearest Neighbors with varying values of k, Multi-Layer Perceptron using grid search and Support Vector Machines with different kernel methods, in addition to ten-fold cross-validation. Post-processing consists of observing the classifiers' class-wise and overall precision and recall, then applying majority voting fusion and stacking separately. We also provide an exploration into feature selection as an alternative in the pipeline to provide pre-processing of the dataset through Cuckoo Search and Recursive Feature Elimination.

1. Introduction

Human activity recognition (HAR) using smartphones has extensive research potential in understanding the activities of daily life (ADL). These days, people rely on their smartphones for nothing short of everything, so having smartphones able to track daily activities allows for things like exercise trackers, fall detectors, or, with the newest Samsung, even theft detection to be possible. This area of research has immense potential to contribute to human activity trackers and assist in everyday living. With this, many datasets have been collected using smartphones with an embedded accelerometer and gyroscope. These datasets can be used in machine learning methods to see how well the algorithms can understand a person's activity.

In this case, we are working with a dataset hosted by the University of California, Irvine, the UCI HAR dataset [1]. This dataset looks at human activity recognition, and researchers at the University of Genova and Universitat Politècnica de Catalunya performed the experiments. The data collection experiment was done with 30 individuals, all between the ages of 19 and 48. They measured six everyday activities: walking, standing, sitting, lying down, walking upstairs, and downstairs. Each participant performed the activities twice, once with the smartphone on their left side and once in their preferred location [2]. The original data set had a dimension of 561. This data was initially analyzed by the researchers who ran the experiments where they preprocessed that data using the Gaussian kernel method and then used 10-fold validation to find hyperparameters for Support Vector Machine (SVM). Other researchers have since looked at the data, and SVM, kNN, and multilayer perceptron (MLP) are popular processing methods. This work introduces a complete pipeline and presents this dataset's pre-processing, processing, and post-processing techniques.

[*] Equal Contribution [1] Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, USA

1.1 Task Allocation

Table 1. Group Task Allocation

	Presentation	Report	Experiments
E. Pangilinan	Post-Processing, Performance Comparison	Abstract, Results	Majority Voting, Stacking
G. Lakshmanan	Processing, Feature Selection	Methodology, Discussion	SVM, MLP, Feature Selection
L. Bailey	Pre-Processing, Processing	Introduction, Related Work, Methodology	FLD, t-SNE, kNN

1.2 Contribution

We introduce the following new techniques in this pipeline:

- t-SNE
- Cuckoo Search
- Recursive Feature Elimination
- Grid Search
- Stacking
- General Additive Model

Each new technique applied in the pipeline is further explored in section 3 - Methodology.

2. Related Works

Various machine learning techniques have been implemented using this dataset or similar data sets working towards the same goal. In the original experimental paper [2], the data was pre-processed using the Gaussian kernel method and split 70-30 for training and testing data. The data was then processed using 10-fold validation to find the hyperparameters for support vector machine (SVM), and then a multiclass SVM was utilized for a one-versus-all approach. The recall and precision were laid out for the classification of each of the activities. Sitting had the lowest recall score of 88%, with standing having the lowest precision score of 90%. The recall and precision were in the upper 90s (96% and above) for all other actions. The overall accuracy of the classification results was 96%.

In additional work by Anguita in [3], the same dataset was used, but a different feature extraction approach resulted in reduced features. This was done by only considering the acceleration data from the smartphones. The same pre-processing technique was used as in the original paper, except that the RBF kernel was used instead of the Gaussian kernel. As for processing, the same cross-fold validation was used as the original paper, but a multiclass hardware-friendly SVM (MC-HF-SVM) was used to try to eliminate floating point arithmetic to reduce computation complexity. While less memory, processor time, and power consumption were used, the results were less accurate than the original paper, with the overall accuracy being 89%.

There was also work with the UCI HAR dataset using different neural network techniques. In [4], there is a focus on using multichannel convolutional neural networks. This paper extracts frequency and power features from the raw signals using the Fast-Fourier Transform and proposes a multi-channel CNN model. Two different CNN models are implemented: frequency and power features. The overall accuracy was 95.25%, as there was some misclassification due to overlapping features. Another paper that uses this data set for neural networks is [5], which uses Stacked Discriminative Feature Learning (SDFL). SDFL is a stacked learning network with multiple modular layers arranged serially. Each layer then applies discriminate analysis and a nonlinear activation function. The data is processed into time and frequency domain features in this work and fed to the model. The model progressively learned to discriminate features at multiple levels. This method led to an accuracy of 96.3%.

As previously mentioned, many datasets similar to the UCI HAR one have used ML analysis techniques, such as in [6]. In this paper, the same six actions were studied but with a smaller sample size of only nine people. With this, three classification methods were investigated: decision trees, SVM, and kNN. SVM had the highest accuracy at 99.4%. This was followed by kNN, where a k value of 1 gave a success rate of 97.1%, and the best k value was 3, which had a success rate of 97.5%. Finally, using a binary decision tree, the success rate was only 53.1%, but when a branching limit of 100 was set, an accuracy rate of 94.4% was achieved. While different from the exact data set used for this project, this paper shows more variety in the classification techniques that could be used.

Overall, across the literature, many similar methods are used for pre-processing, processing, and post-processing. Of the preprocessing methods, feature extraction and kernel methods were popular options. As for processing the data, support vector machines and neural networks were the most popular. However, some work was done using kNN, naive Bayes, decision trees, and logistic regression. As for post-processing recall, precision and accuracy were the most prevalent reported values. Many values were reported in the confusion matrix style, but no work was found in which a fusion matrix was presented.

Much work is being done to better understand daily human activity using smartphone accelerometers. This work can provide insight into how well a sensor can tell what action a person is taking at a specific time. This is handy for exercise or movement tracking and emergency services with elderly and fall-risk patients. With the vast work in this field, there is a wide variety of related work for machine learning methods, and different algorithms were used and analyzed.

3. Methodology

Our primary challenge was the high dimensionality of the dataset, which contains 561 features. To address this, we considered two approaches: dimensionality reduction and feature selection. However, feature selection can lead to a loss of interpretability. Therefore, we decided to explore both approaches: **Phase 1** focused on dimensionality reduction, and **Phase 2** involved feature selection.

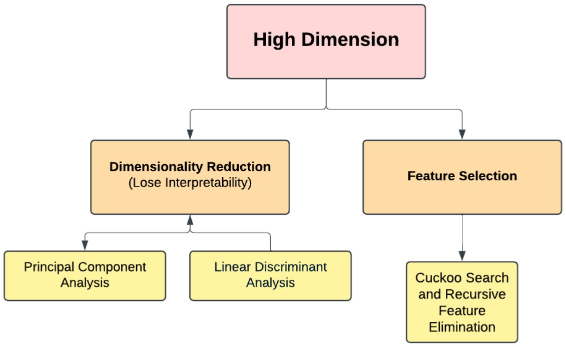
High Dimension
Dimensionality Reduction
(Lose Interpretability)
Feature Selection
Principal Component Analysis
Linear Discriminant Analysis
Cuckoo Search and Recursive Feature Elimination

Figure 1. Primary Challenge

3.1 Dimensionality Reduction

Within phase 1, this pipeline was further split into three phases: preprocessing, where the data was standardized and reduced; processing, where the machine learning algorithms were used; and post-processing, where the performance of the machine learning algorithms was evaluated. In the preprocessing phase, two-dimensionality reduction techniques, Principal Component Analysis (PCA), and Fisher's Linear Discriminant (FLD) were used. With both reduced data sets, t-SNE was applied. In the processing phase, kNN, SVM, and MLP were used. Confusion matrices were created in the post-processing phase with the information from the processing phase. In this phase, majority voting fusion and stacking were incorporated. The workflow of these three phases is displayed in Figure 1.

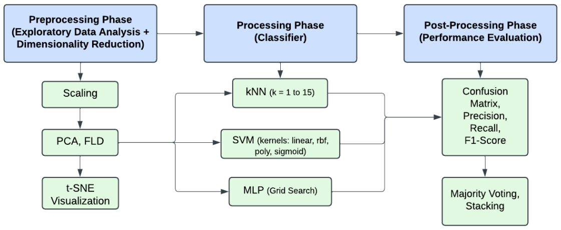
Preprocessing Phase
(Exploratory Data Analysis + Dimensionality Reduction)
Processing Phase
(Classifier)
Post-Processing Phase
(Performance Evaluation)
Scaling
PCA, FLD
t-SNE Visualization
kNN (k = 1 to 15)
SVM (kernels: linear, rbf, poly, sigmoid)
MLP (Grid Search)
Confusion Matrix, Precision, Recall, F1-Score
Majority Voting, Stacking

Figure 2. Workflow

Looking further into the preprocessing phase, the variance was solved at different numbers of components for PCA to determine the optimal number of dimensions to reduce to. The other dimensionality reduction method used was FLD. This technique was applied to the original (standardized) data in which the dimension was reduced to 5, or one less than the number of classes. t-SNE was then used on both the PCA and FLD reduced data to visualize the class separation better. t-SNE reduced the data for both initial methods further down to 3 components. This method, t-SNE, is a non-linear dimensionality reduction technique and is a way to visualize high dimensional data in lower dimensions better, usually 2 or 3 dimensions, 3 in this case.

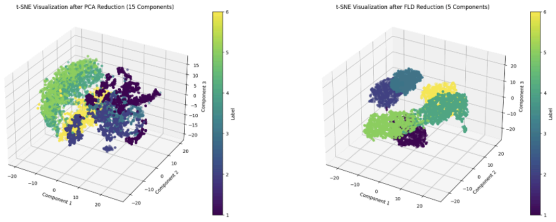

Figure 3. t-SNE Visualization - PCA vs FLD

Within the processing phase, the data was reduced through FLD, and t-SNE was used as it had the best class separation. First, kNN was implemented. This algorithm used standard kNN, and k values between 1 and 15 were tested to see which k value offered the highest accuracy. Next, SVM was implemented, and four kernel methods were used: linear, RBF, polynomial, and sigmoid. Finally, MLP was implemented. Within this method grid search was performed. Grid search is a method used for hyperparameter tuning. This method tries all combinations of parameters to determine the most optimal. This method is computationally expensive but allows for a clear understanding of the best MLP setup.

Going further into the post-processing methods used, majority voting was implemented. This is where each algorithm used presents its result for the classification, and the majority's decision determines the classification. Next, stacking was also implemented. Stacking is a technique in which the strengths of different learners are used, and their predictions are used to make a final prediction that is able to surpass any individual learner. To apply stacking, k-fold cross validation applied to one of the base learners and the training set. Next, the entire training set is fit to the classifier and is applied to the test set. This process was repeated for all six classes.

These methods were integrated to create a complete pipeline. This pipeline was divided into three phases, each of which involved implementing new techniques to further enhance its effectiveness.

3.2 Feature Selection

A two-phase feature selection process to optimize model performance and reduce dimensionality. The methodology employed combined Cuckoo Search (CS) and Recursive Feature Elimination (RFE) sequentially. A Support Vector Classifier (SVC) with a linear kernel was used to guide feature selection at each phase, leveraging its capability to evaluate the impact of selected features on classification accuracy.

In the first phase, Cuckoo Search, a metaheuristic optimization algorithm inspired by the brood parasitism behavior of certain cuckoo species, was employed for feature selection. The CS algorithm leverages the concept of Lévy flights, a random walk mechanism characterized by step lengths following a heavy-tailed probability distribution. Lévy flights allow CS to efficiently explore the search space by making occasional long jumps, avoiding local optima, and ensuring a global search for the optimal solution. The primary objective of CS in this study was to minimize 1−Accuracy(Model), with accuracy serving as the performance metric of the SVC model. The cost function included a penalty term to discourage empty feature subsets, ensuring the retention of meaningful features. In each iteration, candidate solutions represented feature subsets, and their fitness was evaluated by training an SVC model on the selected features and computing the classification accuracy. Poor-performing solutions were replaced by new ones generated through Lévy flights, balancing exploration and exploitation of the search space. The CS algorithm effectively reduced the feature set from 561 to 307 features.

In the second phase, the subset of 307 features from CS was refined using Recursive Feature Elimination (RFE). RFE aimed to maximize the importance of selected features, as measured by the linear SVC model's hyperplane coefficients. By iteratively removing the least important feature at each step (eliminating one feature per iteration), the set was reduced to 10 critical features, emphasizing impactful features with minimal redundancy. The reduced dataset was also visualized using t-SNE as shown in Figure 4, which revealed clear cluster separability between Activities 1, 2, and 3, and Activities 4, 5, and 6. This separation is logical, as the first three activities involve walking, while the latter activities—sitting, standing, and lying down—occur in stationary positions.

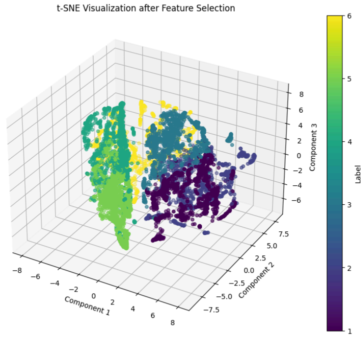

Figure 4. t-SNE Visualization - CS and RFE Applied

The sequential combination of CS and RFE provided a systematic way to reduce dimensionality while preserving critical information in the dataset. Cuckoo Search served as a broad filter to identify relevant features, and RFE further refined the selection by prioritizing feature importance. This approach successfully reduced the original feature set from 561 to 10, ensuring the development of an interpretable dataset.

To assess feature importance and their contribution to the target variable, three analyses were conducted: Spearman correlation analysis, Random Forest feature importance evaluation, and fitting variables as smoothed functions using a General Additive Model (GAM).

The combination of these methods confirms the importance of features 52, 41, and 57 as key contributors to the target variable, supported by both linear and non-linear perspectives. The Spearman correlation analysis highlights the linear relationships, identifying these features as having strong positive correlations with the target variable. The Random Forest feature importance provides a ranking based on their contribution to model accuracy. The GAM model identifies non-linear patterns and interactions, showing how these features exhibit complex relationships with the target through their high EDF. This multi-faceted approach integrates insights from correlation, feature ranking, and non-linear analysis, providing robust evidence for the importance and influence of these features in predictive modeling.

4. Experiments and Results

Our pipeline was developed after running multiple trials on each of our models and different reduced datasets. Hyperparameter tuning with 10-fold cross-validation found that a k-value of 10 for kNN, using an RBF kernel for the multiclass support vector machine, and two hidden layers of 100 nodes, each with stochastic gradient descent and an adaptive learning rate provided the best individual accuracy results. Preliminary testing of our prototype consisted of three experiments that tested the UCI HAR dataset reduced with PCA on kNN, SVM, and majority fusion of kNN and SVM, which yielded precision and recall between 68%-100%. Using the same models along with the stacking model but with FLD applied on the dataset yielded precision and recall between 87%-100%, with the majority of the class-wise precision and recall being in the 90% range.

4.1 Dataset

The UCI HAR dataset consists of 7352 training samples and 2947 testing samples, each containing 561 features. Further investigation into the dataset found an imbalance in the training set for the distribution of the samples, as shown in Table 2 below.

Table 2. Distribution of Training Samples

Class Label	Sample Count
Walking	1226
Walking Upstairs	1073
Walking Downstairs	986
Sitting	1286
Standing	1374
Laying Down	1407

The imbalance in class distribution influenced our decision to train multiple classifiers and perform ensemble methods as an attempt to minimize the effects of a skewed dataset. Our initial prototype utilized PCA for pre-processing, but results show that using a dataset reduced with FLD provided better performance.

4.2 Metrics

The overall goal of this pipeline was to minimize the amount of misclassified samples. Due to the skewed nature of the UCI HAR dataset, the primary metric for evaluating our models' performance was their F1 score. Each model's precision and recall scores were the secondary metrics of evaluation to identify which labels had misclassified samples. We took the weighted averages of the precision and recall to account for a skewed distribution.

4.3 kNN

In our preliminary testing, we found that a *k*-value of 15 provided the best results after applying 10-fold cross-validation using the PCA-reduced dataset. Class-wise precision and recall scores varied widely, but performance was not comparable to the results presented in [2]. Table 3 highlights the overall results of testing with PCA and kNN.

Table 3. Prototype Performance from kNN (k = 15)

Performance Metric	Score
F1-Score	0.83
Precision	0.84
Recall	0.83

In our finalized pipeline, we found that a *k*-value of 10 provided the best accuracy score. After using the FLD-reduced dataset, our kNN classifier yielded significantly higher performance results than our prototype. Table 4 shows the individual classifier performance after adjusting pre-processing and hyperparameter tuning.

Table 4. Finalized Performance from kNN (k = 10)

Performance Metric	Score
F1-Score	0.98
Precision	0.97
Recall	1.00

4.4 SVM

Similar to our experimentation with kNN, we first investigated the results of using a PCA-reduced dataset with an SVM, then used an FLD-reduced dataset in the final pipeline. After hyperparameter tuning, we found that the RBF kernel provided the best accuracy after 10-fold cross-validation, as shown in Figure 5.

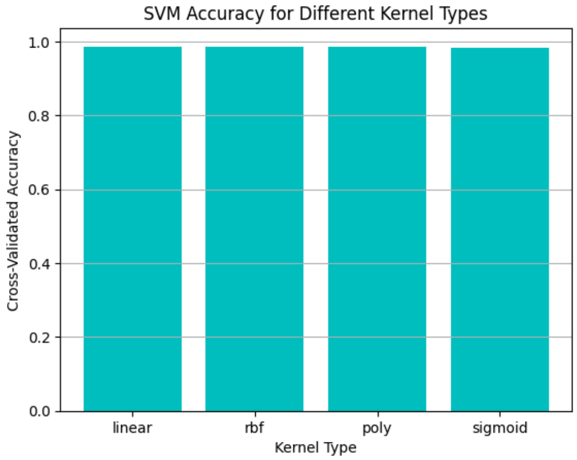

Figure 5. Accuracy Scores for Different Kernel Types

Similar to kNN, our preliminary results for the RBF-SVM saw that class-wise precision and recall varied dramatically. The recall for sitting was the lowest, which scored 0.68 while the precision for laying down scored the highest which was 1.00. Figure 8 shows the overall scores of the RBF-SVM prototype, which performed slightly better than the prototype kNN model. In our final pipeline, we saw improved results compared to the initial prototype, as shown in Table 6.

Table 5. Prototype Performance from RBF-SVM

Performance Metric	Score
F1-Score	0.86
Precision	0.87
Recall	0.86

Table 6. Finalized Performance from RBF-SVM

Performance Metric	Score
F1-Score	0.96
Precision	0.97
Recall	0.97

4.5 MLP

We implemented MLP as an additional classifier that can improve overall performance of our majority voting fusion and stacking models. A grid search was first performed for hyperparameter tuning with 10-fold cross-validation. Table 7 shows the parameter grid used for the MLP grid search.

Utilizing the grid search found that the most optimal hyperparameters were having 2 hidden layers, each with a size of 100 nodes, stochastic gradient descent solver (sgd), and having an adaptive learning rate.

Table 7. Parameter Grid for MLP Grid Search

Hidden Layers	(50,)	(100,)	(50,50)	(100,100)
Solver	adam	sgd		
Learning Rate	adaptive	constant		

The MLP performance on the FLD-reduced dataset was comparable to the other individual classifiers in our pipeline, as shown in Figure 8. There was less variance among the class-wise scores for precision and recall, which led to higher overall results for the F1 score, precision, and recall.

Figure 12. Finalized Performance from MLP

Performance Metric	Score
F1 Score	0.97
Precision	0.97
Recall	0.97

4.6 Ensemble Methods

Table 8. Performance of Ensemble Methods

Performance Metric	Majority Voting	Stacking
F1-Score	0.96	0.97
Precision	0.96	0.97
Recall	0.96	0.97

Both ensemble methods (majority voting fusion and stacking) yielded results comparable or better than the individual classifiers, as shown in Table 8.

4.7 Feature Selection

To explore alternatives to dimensionality reduction, we performed feature selection. A linear SVM was trained and then tested on the dataset reduced with a Cuckoo Search and RFE. The dataset with ten selected features yielded a precision, recall, accuracy, and F1-score of 0.88, 0.87, 0.88, and 0.88, respectively. While these metrics are noticeably lower than the results achieved with dimensionality reduction, they are still considered decent.

Table 9. SVM Performance after Feature Selection

Performance Metric	Score
F1-Score	0.88
Precision	0.88
Recall	0.87

4.8 Comparison to State-of-the-Art Model

Our pipeline produced overall and class-wise performance results comparable to the existing linear SVM model found in [2]. Since other models found in the literature tested against variations or subsets of the UCI HAR dataset, we compared our ensemble models against the linear SVM that used the entire dataset. Tables 10 and 11 show the class-wise precision and recall metrics of both ensemble methods compared to the multiclass SVM found in the existing literature.

Table 10. Comparison of Precision Metrics

	Majority Voting	Stacking	SVM [2]
Walking	98%	98%	96%
Walking Upstairs	97%	97%	98%
Walking Downstairs	100%	100%	99%
Sitting	95%	96%	97%
Standing	89%	90%	90%
Laying Down	100%	100%	100%
Overall	96%	97%	96%

Table 11. Comparison of Recall Metrics

	Majority Voting	Stacking	SVM [2]
Walking	99%	99%	99%
Walking Upstairs	98%	98%	96%
Walking Downstairs	98%	98%	98%
Sitting	87%	89%	88%
Standing	96%	96%	97%

Laying Down	100%	100%	100%
Overall	96%	97%	96%

5. Discussion

In this study, we applied machine learning methods to the UCI-HAR dataset, exploring both dimensionality reduction and feature selection techniques. Both approaches proved viable, but their suitability depends on the specific context. Dimensionality reduction, such as PCA, sacrifices interpretability for computational efficiency, while feature selection retains interpretability but can be more time-consuming. For studies where explainability is crucial, feature selection is the preferred choice. Our analysis revealed that SVM classifiers excelled in handling underrepresented classes, likely due to their ability to maximize margin and handle imbalanced data effectively. We also found that FLD outperformed PCA for this dataset, as FLD considers class separability, making it a better fit for classification tasks. We show that stacking (fusion) the results of three classifiers - kNN, SVM, and MLP proved effective, leveraging the strengths of each method. This approach benefited from kNN's reliance on proximity, SVM's margin maximization, and MLP's ability to capture complex, non-linear relationships. These findings emphasize the importance of method selection and ensemble techniques in achieving robust and reliable performance while balancing interpretability and computational constraints. Future work would include additional hyperparameter tuning for the MLP and a full pipeline of showing feature selection techniques with our implemented classifiers and ensemble methods.

6. References

[1] UC Irvine Machine Learning Repository, “Human Activity Recognition Using Smartphones.” Dec.2012. https://archive.ics.uci.edu/dataset/240/human+activity+recognition+using+smartphones

[2] Anguita, Davide, et al. “A Public Domain Dataset for Human Activity Recognition Using Smartphones.” The European Symposium on Artificial Neural Networks, Jan. 2013, pp. 437–42.

[3] Anguita, Davide,et al. “Human Activity Recognition on Smartphones Using a Multiclass Hardware-Friendly Support Vector Machine.” Lecture notes in computer science, 2012, pp. 216–23. https://doi.org/10.1007/978-3-642-35395-6_30.

[4] Ronao, Charissa Ann, and Sung-Bae Cho. “Human Activity Recognition With Smartphone Sensors Using Deep Learning Neural Networks.” Expert Systems With Applications, vol. 59, Apr. 2016, pp. 235–44. https://doi.org/10.1016/j.eswa.2016.04.032.

[5] Pang, Ying Han, et al. “Stacked Deep Analytic Model for Human Activity Recognition on a UCI Har Database.” F1000Research, vol. 10, 18 Feb. 2022, p. 1046, doi:10.12688/f1000research.73174.2.

[6] Bulbul, Erhan, et al. “Human Activity Recognition Using Smartphones.” 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Oct. 2018, pp. 1–6, doi:10.1109/ismsit.2018.8567275.