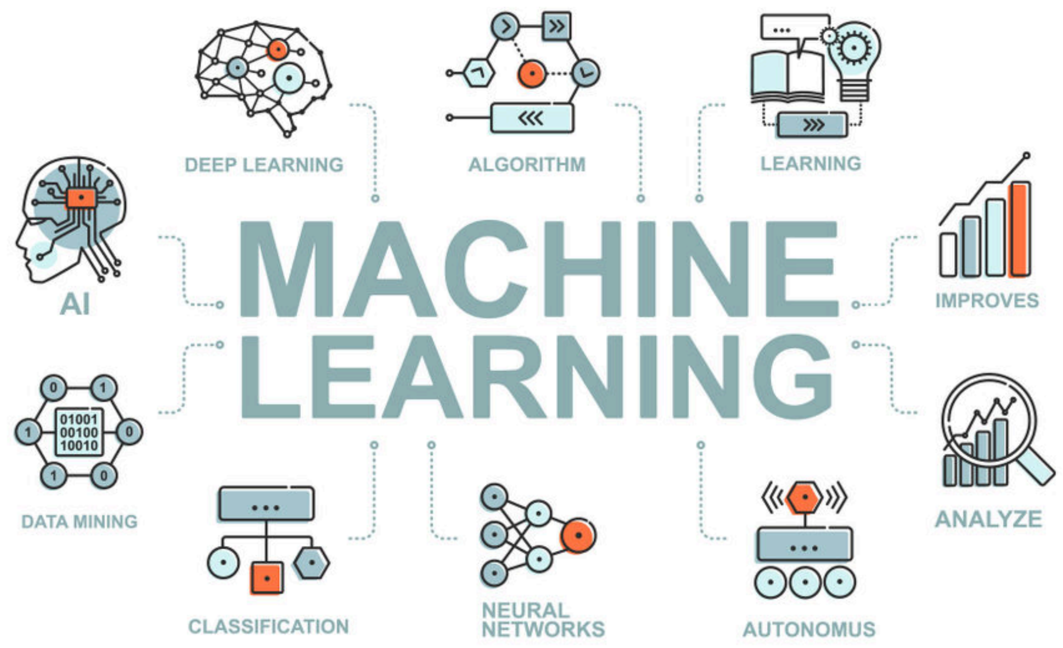
MACHINE LEARNING AND DATA MINING

HUMAN ACTIVITY RECOGNITION PREDICTION

METHODOLOGY, DATA EXPLORATION, MODELLING, AND RESULTS



AI-Assisted Technology

The incorporation of AI-assisted technology, specifically ChatGPT, proved to be highly advantageous in report writing. ChatGPT was used to improve the structure, readability, and language of the report, ultimately enhancing its overall quality.

1. Introduction

The widespread use of mobile devices with sensors provides an opportunity for data collection to study human behaviour and health. In this project, the task is to propose a solution using machine learning for human activity recognition (HAR) based on data collected from personal digital devices. The goal is to develop and test several machine learning models before suggesting a final approach to solve the task.

The dataset used for this project is generated from an extract of data collected by the WISDM Lab, consisting of data collected from 36 different users performing six types of human activities (ascending and descending stairs, sitting, walking, jogging, and standing) for specific periods of time. The data were acquired from accelerometers, which measure the acceleration along the three different dimensions, and collected using a sample rate of 20 Hz. The data was provided in both metadata form and signal time-series format, with the test data and training data provided in separate files in both formats. The use of personal digital devices such as smartphones allowed for the collection of detailed, continuous, and objective measurements on different aspects of the users' physical activities without the use of extra hardware or instruments.

2. Methodology

This project's objective is to create a machine learning model for identifying human activity from a dataset of metadata and signal data.

2.1 Approach

The approach taken for this task involved conducting an exploratory analysis of the metadata and signals data to gain insight into the data and extract relevant features. The results of the exploratory analysis led to the identification of outliers and incomplete data steps, which were filtered out. Feature extraction was also conducted from the signal data to optimize the training of machine learning models. The extracted metadata was standardized and split into 60% for training, 20% for validation, and 20% for testing. Grid Search Cross-validation and Random Search Cross-validation were performed to find the best hyperparameters for traditional machine learning models.

An array of machine learning algorithms were utilised for training the models. The base model, logistic regression, was chosen because of its effectiveness and simplicity in handling data that can be separated into linear categories, which is frequently the case in classification tasks. Random Forest and Support Vector Machines were then used as they are more useful for handling noisy and high-dimensional data. A KNN model was also applied to incorporate unsupervised learning in order to find groups of the expected activities. On the basis of the metadata, features were extracted, and a feature selection technique was used to choose features that would increase the models' accuracy.

According to research, Deep Learning models have shown more success in these problems, DNNs are a particular kind of neural network that is well recognized for processing complex and non-linear data. Hence, a DNN using the extracted features dataset was used. Furthermore, CNNs are a particular class of neural networks that are recognized for handling image and time-series data. Hyper-parameters were tweaked in the CNN training to increase accuracy. LSTM models were also trained due to their effectiveness in time series data handling. Finally, a hybrid model using both CNN and LSTM was developed, which gave good accuracy. Hyperparameter tuning was done on the CNN-LSTM model to achieve the highest accuracy.

2.2 Alternative Approach

The alternate method for solving the recognition of human activities uses both tabular and signal data to create a hybrid model. This is due to the fact that activities are frequently characterised by patterns of movement across time, which requires both temporal and spatial information to recognize human activity. This strategy makes use of both data kinds' characteristics to potentially enhance model performance. A more complete feature set can be produced by merging the features derived from the two datasets, which might improve the model's accuracy. The hybrid model might be trained using either the Random Forest or Gradient Boosting algorithms. This method could beat models trained just on tabular or signal data since it takes into account both temporal and spatial information.

3. Exploratory Analysis

3.1 Class Label Distribution

To gain a basic understanding of how the data is distributed in our dataset, we can plot a histogram of the class label distribution.

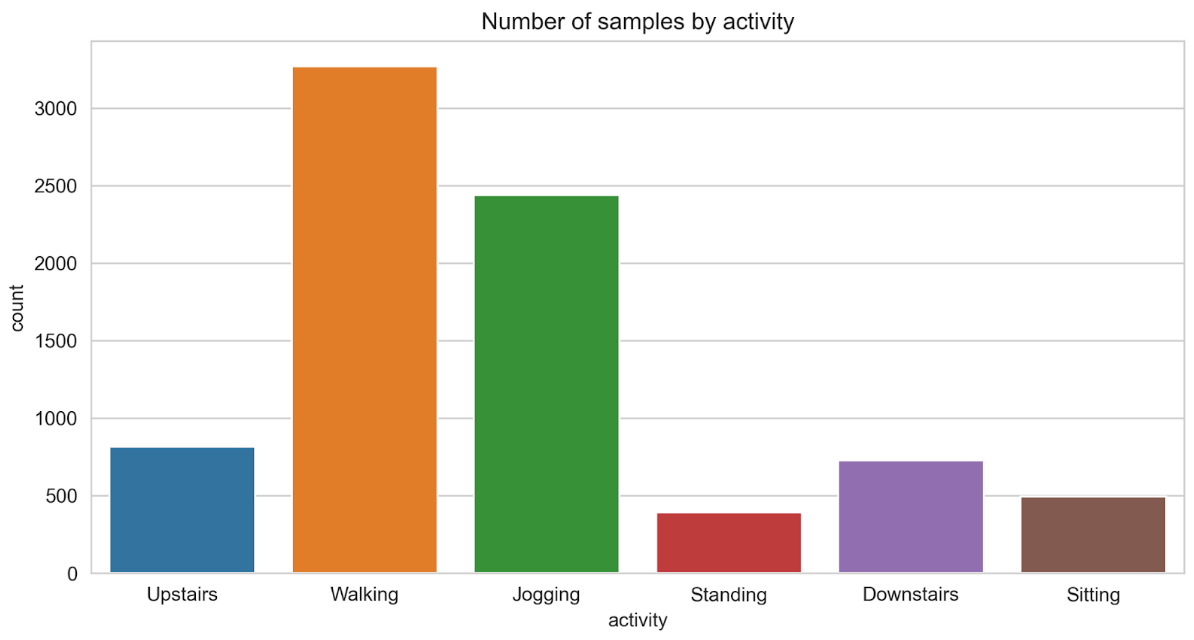


Fig 3.1 Histogram displaying the class label distribution

The distribution of classes in the dataset is skewed, as most of the samples belong to the 'Walking' and 'Jogging' classes, while the 'Standing' and 'Sitting' classes have very few samples. This is called an Imbalanced Dataset.

An imbalanced dataset can be a challenge for machine learning models as it can lead to poor performance in predicting certain classes. In the case of the human activity recognition problem, the skewed distribution of classes can result in poor performance when predicting activities such as 'Standing' and 'Sitting' due to the limited number of samples available for these classes. The models may struggle to differentiate between these classes and other classes that have a larger number of samples, leading to poor predictions. This is a common issue in many machine learning applications and must be addressed to ensure accurate predictions.

One way to address the imbalance issue is through oversampling the minority classes. This involves generating more samples from the original dataset to create a more balanced dataset. In this project, The Synthetic Minority Over-sampling Method (SMOTE) was chosen as the approach to address this class imbalance to provide more samples for the standing and sitting classes, which contain fewer samples than the walking and jogging groups, which could differ from user to user due to individual pace of movement.

After oversampling, we evaluated the performance of the base models using logistic regression both before and after the oversampling technique was applied. The results showed a significant improvement in the performance of the models, indicating that oversampling is an effective approach for dealing with the class imbalance in this dataset. It is important to address class imbalance to ensure accurate predictions and prevent bias in machine learning models.

3.2 Feature Engineering

Although the data came in metadata format with features such as axis-sum_values, axis-median, axis-mean, axis-length, axis-standard_deviation, axis-variance, axis-root_mean_square, axis-maximum, axis-absolute_maximum, and axis-minimum already extracted, we decided to further enhance the accuracy of our models by extracting additional features from the signal dataset. We carried out this process in three stages:

- Stage 1 Statistical Measures: We extracted 18 statistical features, including mean, standard deviation, median, skewness, kurtosis, energy, signal magnitude area, average resultant acceleration, and interquartile range among others from the signals dataset. According to research, these features were important in solving the problem statement at hand because they provide important details on the kind, degree, scope, and range of the activities engaged in, which can aid in differentiating across activities.
- Stage 2 Fast-Fourier Transform: We applied the Fast-Fourier Transform on the signal dataset to analyse the frequency components of the signals. This transformation is a powerful tool for identifying specific patterns or features in the data, which can help differentiate between different types of activities and improve the accuracy of the HAR models.

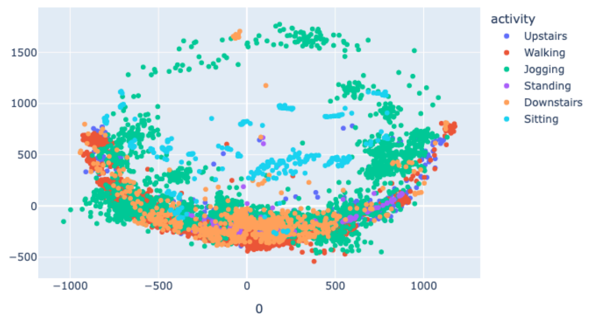
- Stage 3 Capturing Indices: We extracted the index values in the underlying data since they can provide crucial information about the timing and sequencing of events in the data. By analysing the time between successive index values, we can extract important temporal features, such as the duration of an activity or the frequency of transitions between activities. Additionally, index values can help track changes in the user's behaviour over time or identify periodic patterns in their activity.

After these three stages of feature engineering, we obtained a final dataset with 112 columns, which we exported into a separate CSV file called extracted-features.csv for use in training our machine learning models.

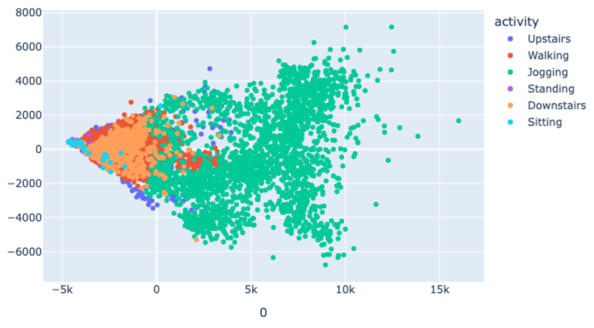
3.3 Dimensionality reduction in metadata

After extracting features from the dataset, we were faced with the challenge of selecting the metadata file or the extracted features file to use in training. To make an informed decision, we employed data mining techniques to visualise the metadata files using efficient dimensionality reduction algorithms. We wanted to determine which dataset would best distinguish between the six classes while retaining as much information as possible. We used three algorithms for this purpose: PCA (Principal Component Analysis), T-SNE (t-Distributed Stochastic Neighbour Embedding), and UMAP (Uniform Manifold Approximation and Projection). We applied these algorithms to both the original metadata.csv and the extracted-features.csv.

Principal Component Analysis



PCA on metadata.csv

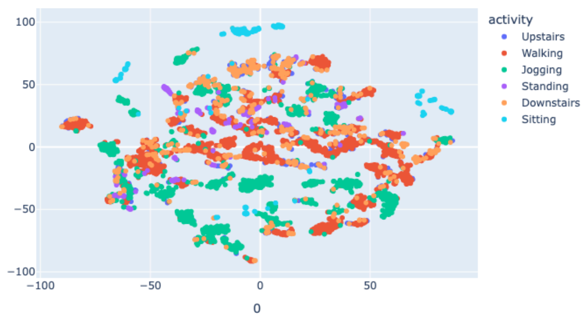


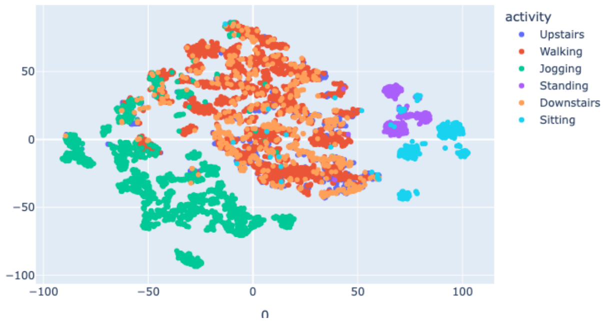
PCA on extracted-features.csv

Fig 3.3.1 Comparison between Principal Component Analysis on Metadata.csv and extracted-features.csv

The comparison between Principal Component Analysis (PCA) on Metadata.csv and extracted-features.csv reveals that the latter provides a better structure to the data. The PCA plot of the extracted-features.csv file shows a clear disparity between the 'sitting' and 'jogging' classes, as they have the highest variability. In contrast, the PCA plot of the metadata.csv does not provide a clear separation of classes as they are not fully distinguishable and are riddled with random clusters and overlaps. This analysis suggests that the extracted-features.csv is better at classifying the activities and placing them in visible single clusters compared to the metadata.csv. Overall, the extracted features are more informative and useful for developing machine learning models.

T-Distribution Stochastic Neighbor Embedding





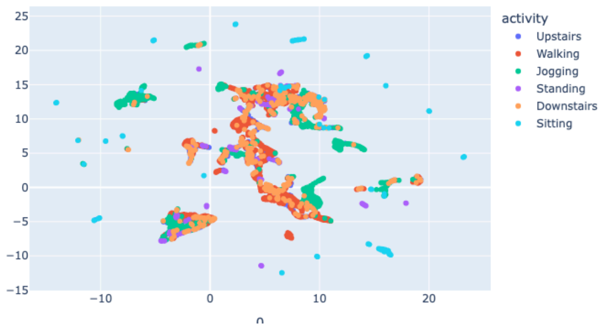
T-SNE on metadata.csv

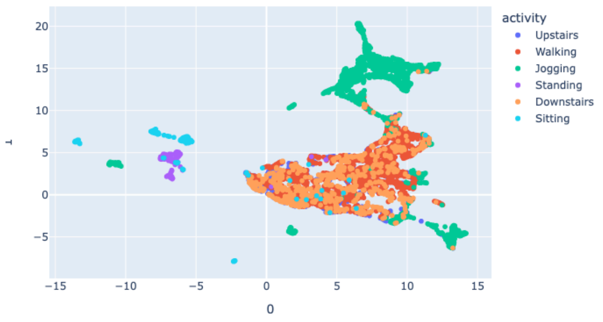
T-SNE on extracted-features.csv

Fig 3.3.2 Comparison between T-SNE on Metadata.csv and extracted-features.csv

In comparing the performance of T-SNE on both datasets, , it is evident that T-SNE on extracted-features.csv provides better class separation than T-SNE on metadata.csv. This can be attributed to the fact that T-SNE works by preserving the local structure of data and its neighbours. The extracted features dataset has clear distinctions between the classes, with most neighbours belonging to the same class and a major overlap between the 'walking' and 'downstairs' classes. However, the metadata.csv dataset shows poor performance with random clusters and no clear way to visualise class labels.

Uniform Manifold Approximation and Projection(UMAP)





UMAP on metadata.csv UMAP on extracted-features.csv

Fig 3.3.3 Comparison between UMAP on Metadata.csv and extracted features

The UMAP algorithm, when applied to the extracted features dataset, shows similar trends to those observed with the T-SNE and PCA techniques. Specifically, the UMAP visualisation reveals distinct clusters for most activities, with only minor overlaps between the 'walking' and 'downstairs' classes. This is indicative of a well-separated dataset, where the features are able to accurately capture the underlying patterns that distinguish between the different classes. In contrast, when UMAP is applied to the metadata, it does not provide a clear visualisation of the different classes, suggesting that the metadata features may not be sufficient to capture the complexity of the HAR problem.

Based on the analysis of the dimensionality reduction techniques, it is evident that the extracted features dataset has a better structure and separation between the classes. Therefore, it is recommended that the training should be done using the extracted features dataset as the input to the UMAP model. By doing so, the classification accuracy of the machine learning models is expected to increase, as the extracted features dataset provides a better representation of the data and improves the ability of the models to distinguish between the different classes. This will ultimately result in better performance of the models in predicting the activities performed by the users.

3.4 Preprocessing

To improve the performance of the machine learning models, several pre-processing techniques were applied to the data. The first technique involved standardising the data, which helped in improving the quality of the models. Next, label encoding was used to transform the activity labels into numerical labels, as deep learning models are unable to process string data. Once predictions were made, the numerical labels were transformed back into their original string format. Finally, to address issues encountered when using certain deep learning models, such as LSTM and CNN, a time-step filtering technique was applied. This involved removing any timesteps that were not equal to 100, to ensure that the data could be accurately processed by the models.

4. MODELLING

We aim to present the results of the machine learning models applied to predict human activity recognition using time series data. In this section, we will discuss the various machine learning algorithms implemented and provide insights into the performance of each model. The objective of this section is to evaluate the effectiveness of the models and highlight any limitations and assumptions that may have influenced the results. Through this process, we hope to provide a comprehensive understanding of the modelling methodology and its outcomes.

4.1 Model Algorithms

Logistic Regression

Logistic regression was chosen for this specific problem because it is a simple yet effective algorithm for binary classification tasks. Given the high dimensionality of the dataset, it served as a good starting point for building a baseline model, which could handle the noise in the data. It is also computationally efficient and easy to interpret, which makes it a practical choice for this problem. it was applied to the feature-extracted dataset.

Support Vector Machines (SVM)

SVM was chosen for this specific problem because it can effectively handle high-dimensional data and is robust to noise. The SVM was applied to the feature-extracted dataset using a radial basis function (RBF). The best C and gamma were 100 and 0.001 respectively.

Random Forest

Random Forest was chosen for this specific problem because it is a powerful ensemble learning method that can handle high-dimensional data with noisy features. It is robust to overfitting and can capture complex non-linear relationships between variables. The best Random forest model was trained using the following parameters: {'n_estimators': 600, 'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features': 'sqrt', 'max_depth': 90, 'bootstrap': False}.

K-nearest neighbours (KNN)

KNN was chosen for this specific problem because it is a non-parametric algorithm that can handle high-dimensional data and is robust to noisy features. We were hoping to see clusters connected to the different activities. We selected the number of clusters to be six, the same number of unique activities.

Deep Neural Network (DNN)

The model consists of a series of dense layers with rectified linear unit (ReLU) activation functions, followed by dropout layers to reduce overfitting. The last dense layer has 6 units and uses a softmax activation function. The total trainable parameters were 34,054.

Convolutional Neural Network (CNN)

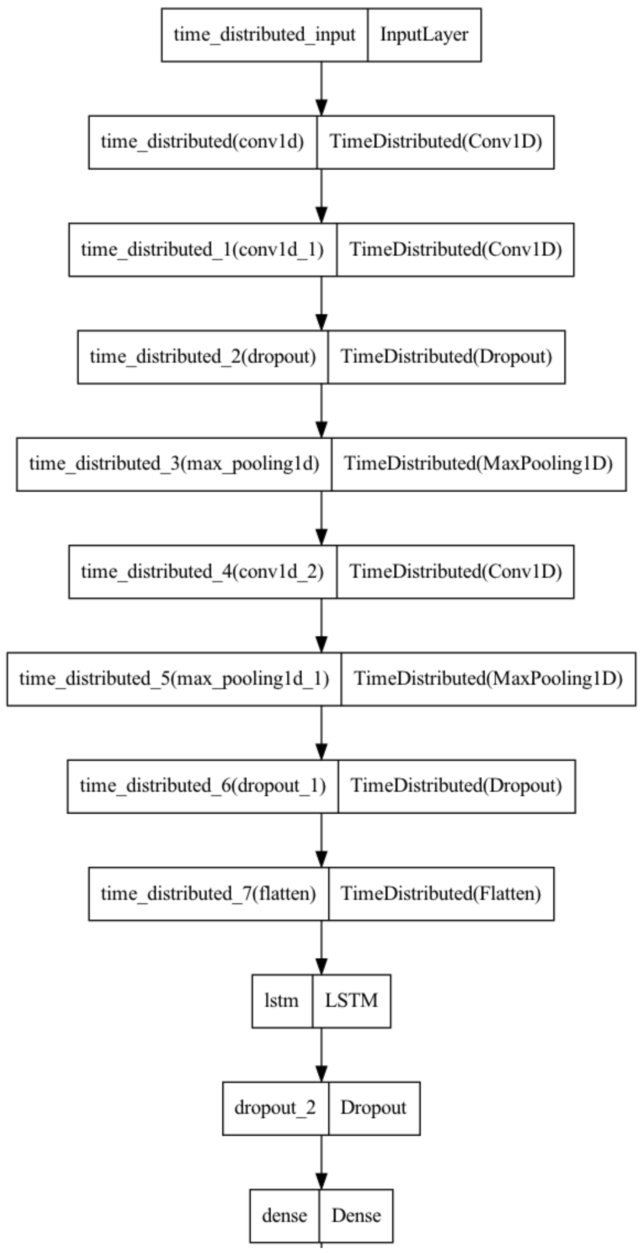
The model is a convolutional neural network (CNN) with four convolutional layers followed by max pooling, flattening, and three fully connected layers. The model uses the Adam optimizer with a learning rate of 0.0001, sparse categorical cross-entropy loss, and accuracy as the evaluation metric. The total trainable parameters were 2,133,062.

Long Short-Term Memory Network (LSTM)

This model is a sequential neural network with three layers: an LSTM layer, a dropout layer, and a dense layer with ReLu activation. It ends with a softmax layer to predict the output classes. It has a total of two dense layers, one LSTM layer, and one dropout layer. It uses categorical cross-entropy as the loss function, the Adam optimizer, and accuracy as the evaluation metric. The total trainable parameters were 132,038.

CNN-LSTM

CNN-LSTM was chosen for this specific problem because it combines the strengths of both CNN and LSTM to handle high-dimensional time series data with spatial structure, such as sensor data collected over time. The final pipeline was an ensemble of 5 neural network models that utilise a combination of convolutional and recurrent layers to classify human activities from time series accelerometer data. The model architecture includes two 1D convolutional layers, followed by dropout and max pooling layers, and another 1D convolutional layer. The output of this layer is fed to an LSTM layer, followed by dense layers with ReLU and softmax activation functions. The model uses the RMSprop optimizer and categorical cross-entropy loss function. The model was trained on 688,530 trainable parameters with the time-series dataset as input parameters. During training, predictions are created by each model, and the mode of the predictions from all 5 models is taken as the final prediction.



time_distributed_input | InputLayer
time_distributed(conv1d) | TimeDistributed(Conv1D)
time_distributed_1(conv1d_1) | TimeDistributed(Conv1D)
time_distributed_2(dropout) | TimeDistributed(Dropout)
time_distributed_3(max_pooling1d) | TimeDistributed(MaxPooling1D)
time_distributed_4(conv1d_2) | TimeDistributed(Conv1D)
time_distributed_5(max_pooling1d_1) | TimeDistributed(MaxPooling1D)
time_distributed_6(dropout_1) | TimeDistributed(Dropout)
time_distributed_7(flatten) | TimeDistributed(Flatten)
lstm | LSTM
dropout_2 | Dropout
dense | Dense

Fig 4.1 Model Architecture of the CNN-LSTM model with the highest accuracy.

4.2 Hyperparameter tuning

The process of hyperparameter tuning was crucial to find the optimal hyperparameters for each algorithm. The Grid Search Cross Validation method was employed to find the best combination of hyperparameters that maximises the performance of each algorithm.

For each algorithm, a range of hyperparameters was specified, and a grid of possible combinations of these hyperparameters was created. The models were then trained and validated using the different combinations of hyperparameters. The combination that resulted in the best performance was chosen as the optimal hyper-parameter.

The following hyperparameters were tuned for each algorithm:

Support Vector Machines: C, gamma

Random Forest: n_estimators, max_depth, min_samples_split, min_samples_leaf

Logistic Regression: C

CNN: number of filters, kernel size, the dropout rate

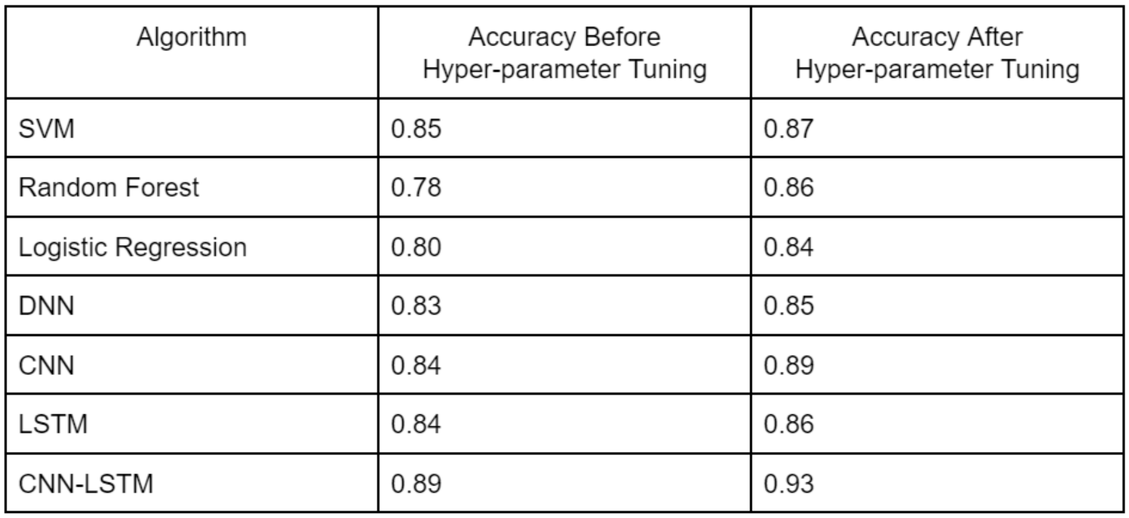
LSTM: number of hidden units, the dropout rate

CNN-LSTM: number of filters, kernel size, number of hidden units, the dropout rate

DNN: number of hidden layers, number of neurons in each layer, activation function.

The range of values searched for each hyperparameter varied depending on the algorithm and the hyperparameter in question. Before hyperparameter tuning, the models were trained and validated using default/random hyperparameters. After hyperparameter tuning, the models were retrained and validated using the optimal hyperparameters. The performance metrics, such as accuracy and F1-score, were then compared before and after hyperparameter tuning to assess the improvement in performance.

By implementing hyperparameter tuning, we were able to fine-tune each model to perform optimally on our high-dimensional and noisy human activity recognition dataset. Below is a summary of the accuracy scores of each model before and after hyperparameter tuning:



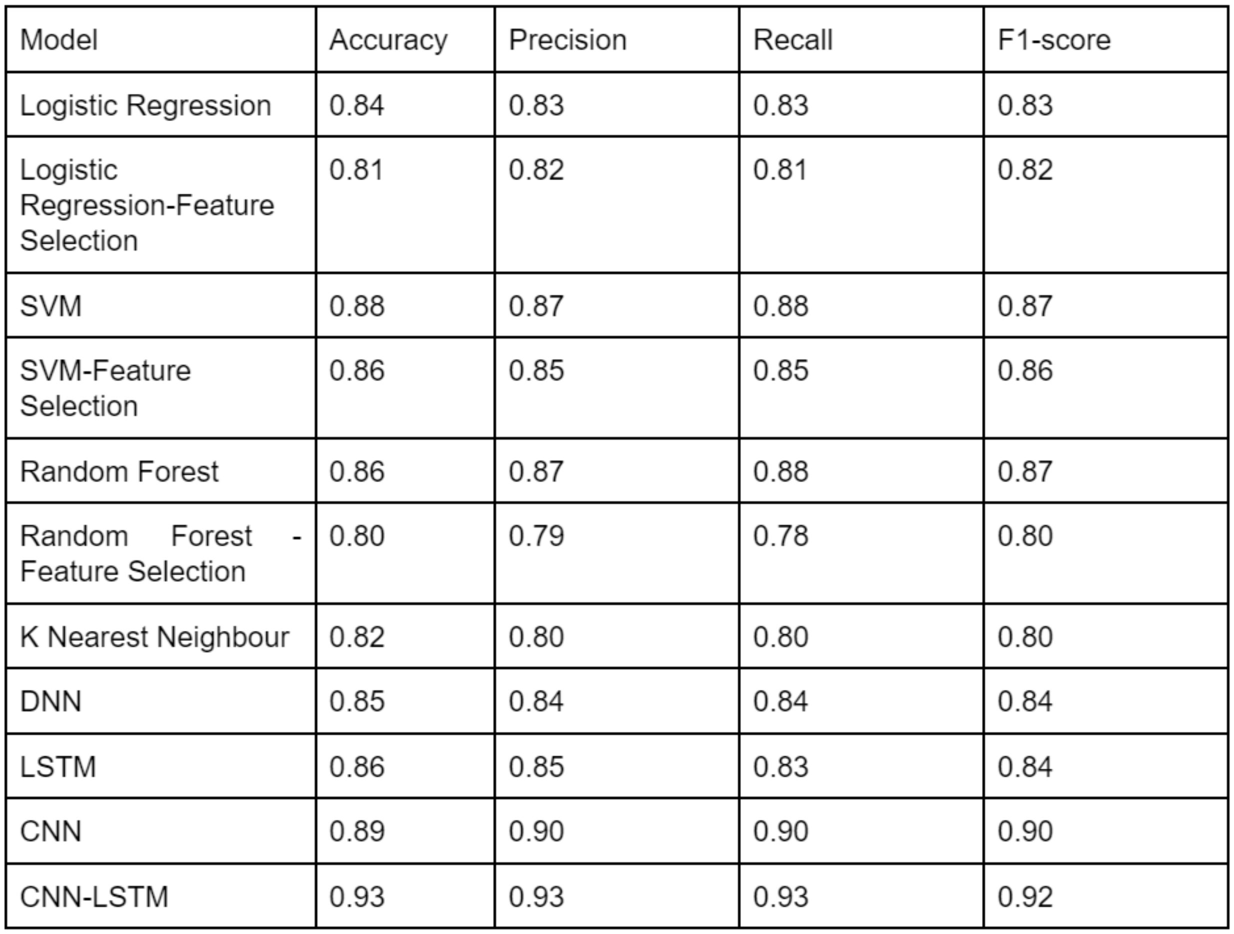
Algorithm	Accuracy Before Hyper-parameter Tuning	Accuracy After Hyper-parameter Tuning
SVM	0.85	0.87
Random Forest	0.78	0.86
Logistic Regression	0.80	0.84
DNN	0.83	0.85
CNN	0.84	0.89
LSTM	0.84	0.86
CNN-LSTM	0.89	0.93

Figure 4.2 Showing the accuracy of models before and after hyper-parameter tuning.

As we can see, hyperparameter tuning significantly improved the performance of each model, with CNN and CNN-LSTM showing the most improvement.

4.3 Model Performance

After a series of hyperparameter tuning, we also employed recursive feature elimination (RFE) to extract the best 90 features from the 112 features in the extracted features dataset, the performance of each model was evaluated using a variety of metrics, including accuracy, precision, recall, and F1-score. The final performance metrics of each model are summarised in the figure below:



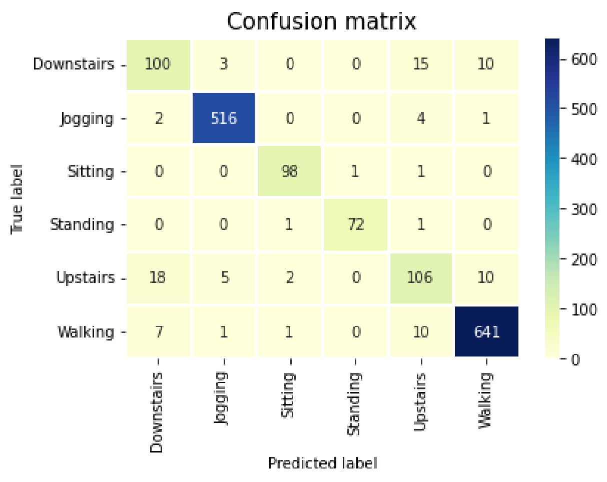
Model	Accuracy	Precision	Recall	F1-score
Logistic Regression	0.84	0.83	0.83	0.83
Logistic Regression-Feature Selection	0.81	0.82	0.81	0.82
SVM	0.88	0.87	0.88	0.87
SVM-Feature Selection	0.86	0.85	0.85	0.86
Random Forest	0.86	0.87	0.88	0.87
Random Forest - Feature Selection	0.80	0.79	0.78	0.80
K Nearest Neighbour	0.82	0.80	0.80	0.80
DNN	0.85	0.84	0.84	0.84
LSTM	0.86	0.85	0.83	0.84
CNN	0.89	0.90	0.90	0.90
CNN-LSTM	0.93	0.93	0.93	0.92

Figure 4.3 Showing the performance of models using Accuracy, precision, Recall, and F1-Score.

As shown in Fig 4.3, all models achieved relatively high performance, with the CNN-LSTM model achieving the highest accuracy of 0.92. The precision, recall, and F1-score metrics also indicate strong performance for each model.

4.4 Confusion Matrix and Epoch Curves

To better understand the performance of each model, confusion matrices and Epoch curves were generated.



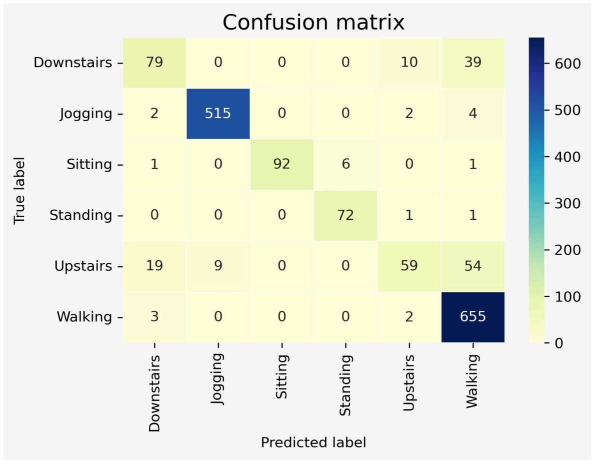
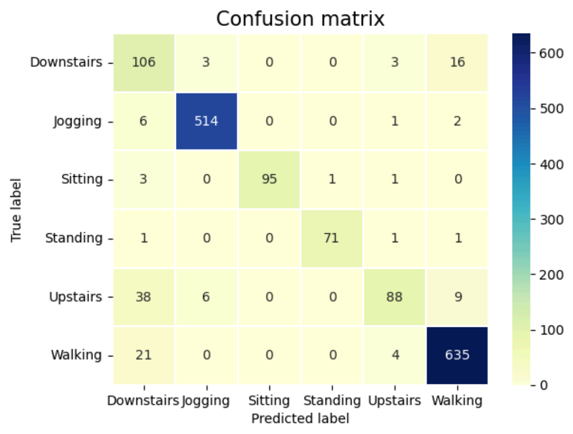


Fig 4.4.1 Confusion matrix of SVM Model   Fig 4.4.2 Confusion matrix of KNN Model



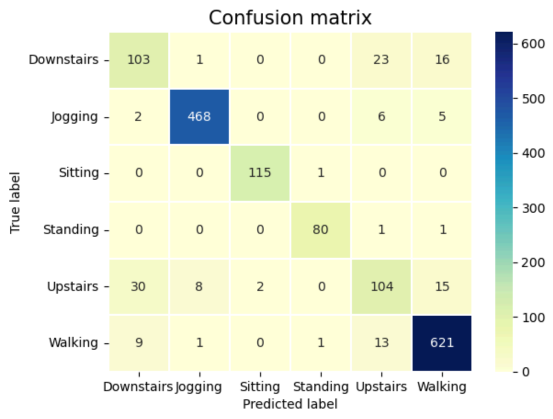


Fig 4.4.3 Confusion matrix of DNN Model   Fig 4.4.4 Confusion matrix of CNN Model

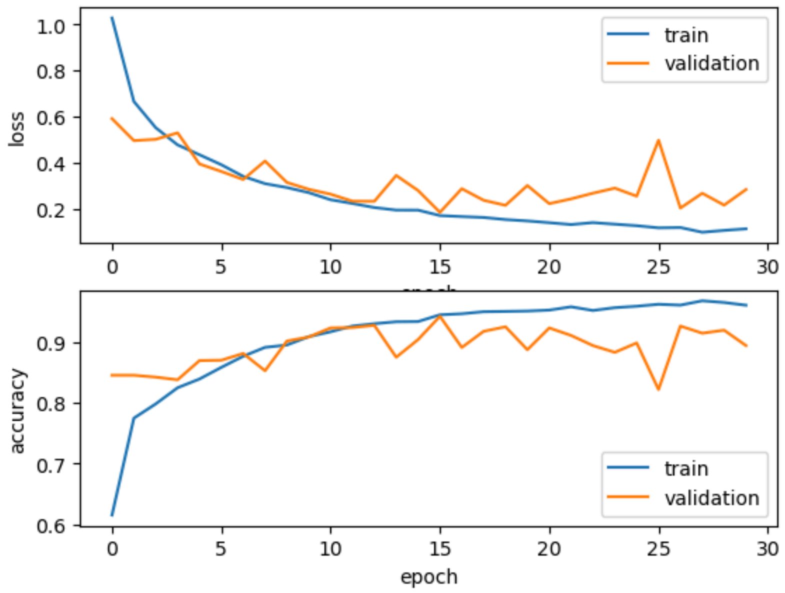


Fig 4.4.5 Epoch curve training the CNN-LSTM Model

Upon comparison of the different models, the CNN-LSTM model performed the best due to its ability to effectively capture both temporal and spatial features in the time series data.

5. Result Analysis

We explored human activity recognition using a dataset containing numerous columns and noise. We applied several machine learning algorithms to the data, including logistic regression, SVM, random forest, KNN, DNN, CNN, LSTM, and CNN-LSTM.

5.1 Exploratory Results

The exploratory analysis showed that the data used for the machine learning models was highly imbalanced, with some classes having significantly more data samples than others. This imbalance could negatively impact the performance of the models. To address this issue, we applied an oversampling method to the minority classes, which helped to balance the dataset.

Upon further exploration, we discovered that there was a need for feature engineering to extract better features from the dataset. We carried out this feature engineering process and verified the new extracted-features dataset by comparing the PCA, UMAP, and TSNE visualisations of the original and new datasets. The results of all three methods showed that our new dataset was a better representation of the original data and would be more useful for training the machine learning models.

To better improve the performance of our models, we applied a feature selection technique using scikit-learn RFE. The top 5 features in the metadata that were found to be important were the mean, standard deviation, average absolute deviation, minimum and maximum axis, and median axis. This indicates that these features were the most informative for distinguishing between the different human activities. Overall, our efforts to address the imbalanced data and extract better features have resulted in a more accurate and reliable machine learning model.

5.2 Modelling Results

After applying the various machine learning algorithms to the preprocessed data, we evaluated their performance using several metrics such as accuracy, F1-score, and

ROC-AUC. The results showed that the CNN-LSTM model achieved the highest accuracy score of 0.93, followed closely by the LSTM model with a score of 0.91. These models outperformed the other algorithms, which had accuracy scores ranging from 0.75 to 0.89.

The success of the CNN-LSTM and CNN models can be attributed to their ability to effectively capture the temporal relationships in the data. The CNN-LSTM model, in particular, was able to extract relevant features from the data using its convolutional layers, which were then fed into its LSTM layers to capture the temporal dependencies between the features. The SVM model was quite good in performance using the extracted features, getting an accuracy of 0.88. This could be due to its non-linear characteristics. We can observe the UMAP diagram in Fig 3.3.3 and TSNE in Fig 3.3.2 that the visualisation of the dataset using dimensionality reduction shows the data points are clustered together in a complex pattern, then it is likely that the problem is non-linear.

On the other hand, the poor performance of some of the other algorithms such as KNN can be attributed to their inability to handle high-dimensional data with noise, we can clearly observe that in the TSNE in Fig 3.3.2 that the extracted features metadata cannot be explicitly separated thus leading to inaccurate predictions among classes ‘walking’ and ‘downstairs’. Additionally, some of the algorithms such as logistic regression may not be able to capture the temporal dependencies in the data, which can limit their performance as they are primarily used for linear classification problems.

We used an ensemble methodology on the CNN-LSTM model, which involved training 5 models and taking the mode of all 5 models to make the final predictions. This helped to reduce overfitting and improve the accuracy of the model.

5.3 Limitations & Assumptions

A major assumption was that the features extracted are relevant to the problem. However, there could be some noise in the features extracted and do not accurately capture the underlying patterns and relationships in the data, the model may be less accurate. Another assumption is that the data provided is a representation of all possible activities. We could run into overfitting problems when we come across new data from new users.

5.4 Future Improvements

To improve the performance of the models, several strategies can be considered. One strategy is to further preprocess the data to reduce noise and feature redundancy. Another strategy is to explore more advanced deep learning architectures, such as attention-based models, which can effectively capture long-term dependencies in the data. Additionally, ensemble methods such as stacking and boosting can be applied to combine the strengths of multiple models and further improve performance.

6. Conclusion

In conclusion, this project demonstrated the effectiveness of deep learning models such as LSTM and CNN-LSTM for human activity recognition. However, there is still room for improvement, and future work can explore more advanced techniques to further enhance the performance of these models.