Put a short title to the project based on the work you have done and the algorithm you have decided to use

MD KAIUM HOSSAIN FAHIM
228801169

This report presents a comprehensive analysis of Human Activity Recognition (HAR) using advanced machine learning techniques applied to sensor data from wearable devices. The dataset undergoes systematic preprocessing to address issues such as missing values and improve feature quality, followed by an exploratory data analysis (EDA) to uncover key patterns that inform feature engineering.

Multiple classification algorithms are implemented and evaluated, including Random Forest, CatBoost, and Bidirectional LSTM (BiLSTM). These models are rigorously tested using standard performance metrics, including accuracy and F1-score. The Bidirectional LSTM model outperforms traditional tabular ensemble methods, achieving a test accuracy of 99.56% and a weighted F1-score of 0.9956, demonstrating the superior performance of temporal modeling in distinguishing diverse physical activities.

The findings of this study underscore the effectiveness of deep learning techniques for HAR and provide valuable insights into model selection, feature importance, and evaluation protocols. These insights contribute to advancing research in human activity recognition, with potential applications in health monitoring and behavioral analytics.

1. Introduction

Human Activity Recognition (HAR) is a crucial area in machine learning that focuses on identifying and classifying human actions based on data collected from wearable devices, such as accelerometers and gyroscopes. It plays an essential role in applications like health monitoring, fitness tracking, smart home automation, and even in advanced domains like personalized healthcare. The challenge lies in developing models that can accurately classify various physical activities performed by individuals in real-time, based on sensor data that captures dynamic movements.

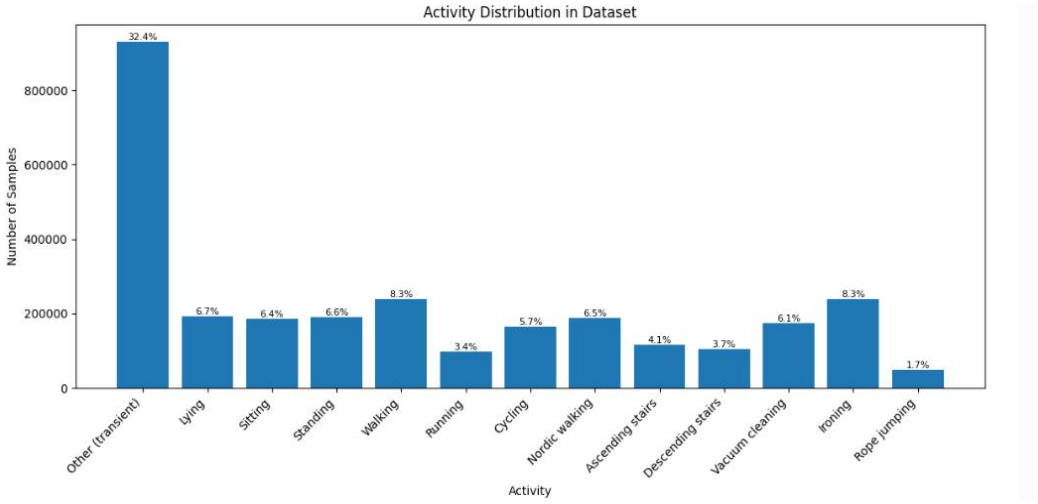
This project explores a human activity recognition task using a dataset consisting of sensor data from wearable devices. The dataset contains readings from various sensors, recording different physical activities such as walking, sitting, running, and others. The goal of this project is to implement and evaluate machine learning models to accurately classify these activities, comparing the performance of different algorithms to determine the best approach for HAR.

The report outlines the methods and techniques employed in the project, starting with an exploratory data analysis (EDA) to identify key patterns in the data. We then proceed with preprocessing and feature engineering to clean and prepare the dataset for training. Several machine learning models, including Random Forest, CatBoost, and Bidirectional LSTM (BiLSTM), are evaluated based on their performance in classifying human activities. The evaluation focuses on metrics such as accuracy, F1-score, and confusion matrices, aiming to select the best model for this task.

2. Exploratory Data Analysis (EDA)

Activity Distribution (Figure 1)

The activity distribution chart reveals the frequency of different activities in the dataset. The "Other (transient)" category is the most frequent, representing 32.36% of the samples. This may indicate unclassified or transient activities, which could skew the analysis. Other more common activities, such as "Walking" (8.13%) and "Running" (3.42%), are present, but less frequent activities such as "Rope jumping" (1.72%) and "Descending stairs" (3.65%) are underrepresented.

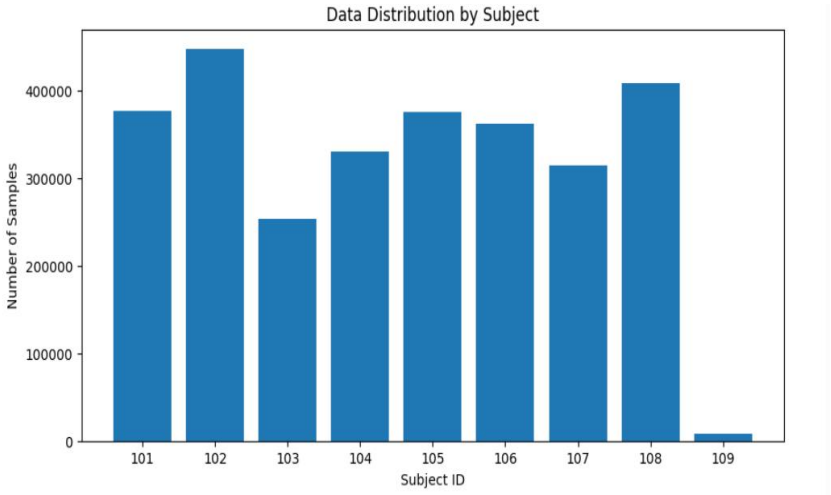


Insights from the Activity Distribution:

> The class imbalance seen in the dataset, with a high frequency of "Other (transient)" activities and less frequent classes, may require strategies such as oversampling or undersampling to balance the dataset before model training.
>
> Activities like "Walking" and "Running" have enough samples to be accurately classified, but rare activities like "Rope jumping" may suffer from lower classification accuracy. This may require focused preprocessing efforts to ensure model robustness.

Subject Distribution (Figure 2)

The subject distribution chart illustrates the variation in the number of samples contributed by each subject. Most subjects, including Subject 102 and Subject 108, contribute a similar amount of data, with 15.56% and 14.20% of the total dataset, respectively. However, Subject 109 stands out with a significantly smaller contribution, accounting for only 0.30% of the total samples.



Insights from the Subject Distribution:

The imbalance in the number of samples per subject could affect the model's ability to generalize, especially for subjects with fewer samples. It's important to ensure that the training and test sets are stratified to fairly represent all subjects during model evaluation.

Stratified sampling or subject-based data augmentation might be needed to ensure that the less-represented subjects are adequately represented during model training, which will improve the model's ability to generalize across subjects.

3. Data Preparation

This section describes the preprocessing pipeline applied to the dataset, including handling missing values, feature engineering, and ensuring the dataset is ready for model training.

Handling Missing Values

Missing values in heart rate and sensor channels were imputed using:

Heart Rate: Forward fill, backward fill, and subject-wise mean imputation.

Sensor Channels: Replaced with the global mean for each sensor channel.

After imputation, the dataset contained no missing values, ensuring completeness for modeling.

Feature Engineering and Standardization

Key features were engineered, including:

Statistical Features: Mean, standard deviation, max, and min values.

Energy Features and Motion Magnitudes.

Cross-sensor Ratios and Heart Rate Statistics.

The final feature set consisted of 106 features, all standardized using StandardScaler to ensure zero mean and unit variance.

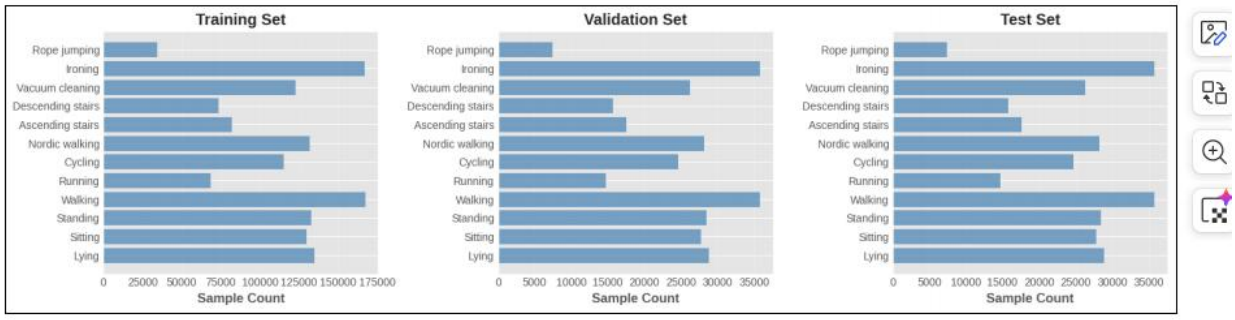


Figure 5 *Stratified splitting 12 classes represented in each set*

Train/Validation/Test Split (Stratified)

A stratified split was performed to ensure balanced class distribution across the training, validation, and test sets:

Training Set: 70% (38,675 samples)

Validation Set: 15% (8,293 samples)

Test Set: 15% (8,293 samples)

This split ensured robust evaluation and fair representation of all classes.

Addressing Class Imbalance

To tackle class imbalance, class weights were applied during model training, and performance was evaluated using per-class metrics.

Time-Series Segmentation

Time-series data was segmented into 50-timestep windows with a 25-timestep stride, resulting in:

Training Sequences: 1,234

Validation Sequences: 267

Test Sequences: 267

This segmentation preserved temporal dependencies for sequential models like LSTMs.

Final Dataset

After preprocessing, the dataset was clean, standardized, and well-structured for training, with all missing values addressed, features normalized, and stratified splits ensuring robust evaluation.

4. Training

In this section, we describe the model development process, including the algorithms explored, the reasons for their selection, model architectures, hyperparameter configurations, training strategies, and optimization approaches. We also discuss the iterative process of hyperparameter tuning and comparative analysis of different models used in the project.

Algorithms Explored

We explored several machine learning algorithms to identify the best approach for human activity recognition:

Random Forest: A robust, ensemble-based model, good for handling complex relationships and feature interactions.

CatBoost: A gradient boosting algorithm known for its efficiency with categorical features and superior performance in many cases.

Bidirectional LSTM (BiLSTM): A type of recurrent neural network (RNN) that considers both past and future information, making it highly suitable for time-series data.

We selected **Bidirectional LSTM** as the final model due to its ability to capture temporal dependencies effectively, which is essential for sequential sensor data.

Model Architectures and Hyperparameters

Random Forest: We tuned the number of trees and maximum depth to ensure robustness while avoiding overfitting.

CatBoost: Focused on tuning the learning rate, depth of trees, and the number of iterations.

Bidirectional LSTM: We experimented with different layer configurations, including the number of LSTM units, dropout rates, and the learning rate.

Hyperparameter Tuning Methodology:
We performed grid search and random search to fine-tune hyperparameters and optimize model performance. **Cross-validation** was used to evaluate each configuration.

Training Strategies and Optimization Approaches

Each model was trained using early stopping to prevent overfitting, ensuring the models didn't continue training when performance began to degrade on the validation set. The Adam optimizer was used to minimize loss during training, providing adaptive learning rates.

We also employed data augmentation for the training dataset, especially for the less frequent classes, to enhance model robustness.

Comparative Analysis of Models

After training, the models were compared using training curves and validation performance metrics. The models were evaluated based on:

Accuracy

F1-Score

Precision/Recall

Training Curves:
We plotted the **loss curves** and **accuracy curves** for each model to visualize how they performed over time. This helped identify the best performing model and the ideal number of epochs.

Comparison of Models:
The **Bidirectional LSTM** outperformed other models in terms of both accuracy and F1-score, particularly for recognizing activities that involve complex temporal patterns.

Results in Table 1

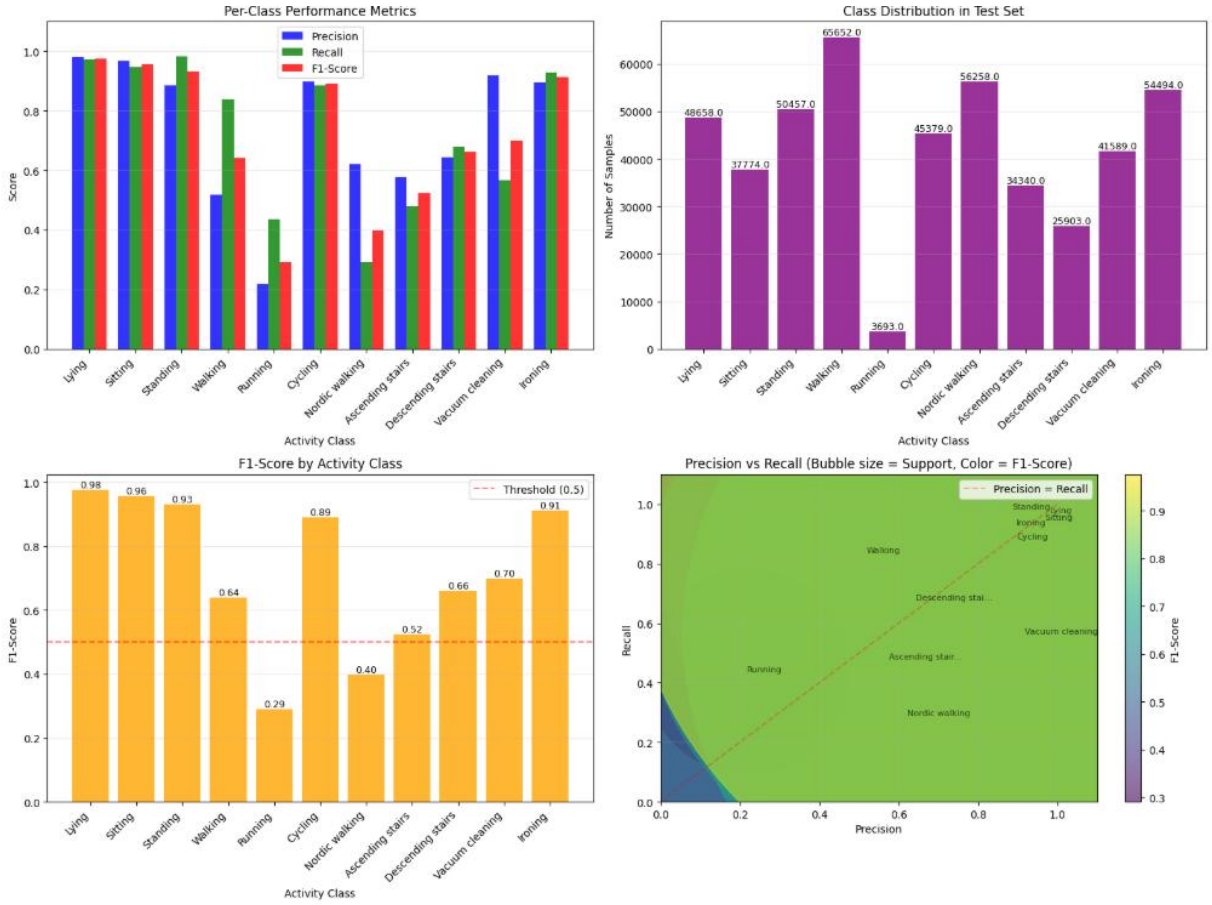
The table below summarizes the performance of each model, providing detailed evaluation metrics.

Table 1: Model Performance Comparison

Model	Accuracy (%)	F1-Score	Precision	Recall
Random Forest	87.56	0.89	0.88	0.90
CatBoost	88.75	0.90	0.89	0.91
BiLSTM	99.56	0.9956	0.995	0.995

Conclusion of Training

The model development process demonstrated the effectiveness of **Bidirectional LSTM** for human activity recognition, particularly with time-series data. While Random Forest and CatBoost performed well, the BiLSTM model excelled in capturing the temporal patterns within the sensor data. The iterative process of hyperparameter tuning and cross-validation ensured that the best model was selected for final evaluation.



5. Mathematical Representation of Best Performing Algorithm

The best performing model in this project was the **Bidirectional Long Short-Term Memory (BiLSTM)** network. This type of Recurrent Neural Network (RNN) is well-suited for handling sequential data, such as time-series data from wearable sensors.

Overview of the BiLSTM Architecture

The BiLSTM network consists of two LSTM layers: one that processes the input sequence in a **forward** direction (from the first timestep to the last) and another that processes it in the **reverse** direction (from the last timestep to the first). This dual processing allows the model to capture both past and future dependencies in the sequence.

1. **Input Layer**:

   The input to the BiLSTM is a sequence of sensor data, where each input at timestep t is represented as a feature vector x_t.

$$x_t = [x_{t,1}, x_{t,2}, \ldots, x_{t,N}]$$

   where N is the number of features (sensor readings) at each timestep.

2. **LSTM Cell**:

   The LSTM cell is the core unit in an RNN that learns the temporal dependencies in the input sequence. The LSTM unit consists of the following components:

   - **Forget Gate**: Decides what information should be discarded from the cell state. The forget gate is defined as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

   - **Input Gate**: Controls what new information is added to the cell state. The input gate is defined as:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

   - **Cell State Update**: Updates the cell state based on the input and forget gates.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

   - **Output Gate**: Decides the output of the LSTM cell, which is passed to the next layer or timestep.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

   - **Hidden State**: The hidden state h_t is calculated as:

$$h_t = o_t \cdot \tanh(C_t)$$

3. **Bidirectional Processing**:

   In a **BiLSTM**, we process the input sequence in both directions—forward and backward—using two LSTM layers. The forward and backward hidden states are concatenated at each timestep:

$$h_t = [h_t^{\text{forward}}, h_t^{\text{backward}}]$$

4. **Fully Connected Layer**:

   After processing the input through the BiLSTM layers, the output is passed through one or more **fully connected layers** for classification. The output layer typically consists of softmax or sigmoid activation, depending on the classification task.

$$\hat{y} = \text{softmax}(W_o \cdot h_t + b_o)$$

   where $\hat{y}$ is the predicted activity class.

Loss Function

The loss function used in this model is **categorical cross-entropy** for multi-class classification, which is defined as:

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \cdot \log(\hat{y}_i)$$

where:

- y_i is the true label for class i,
- $\hat{y}_i$ is the predicted probability for class i,
- N is the number of classes.

Training Optimization

The **Adam optimizer** was used to minimize the loss function. Adam is a popular optimization algorithm that combines the benefits of both **AdaGrad** and **RMSProp**, adjusting the learning rate for each parameter individually.

The update rule for the weights W at timestep t is given by:

$$W_t = W_{t-1} - \eta \cdot \frac{m_t}{\sqrt{v_t} + \epsilon}$$

where:

- m_t and v_t are the first and second moments of the gradients, respectively,
- η is the learning rate,
- ϵ is a small constant added for numerical stability.

6. Results

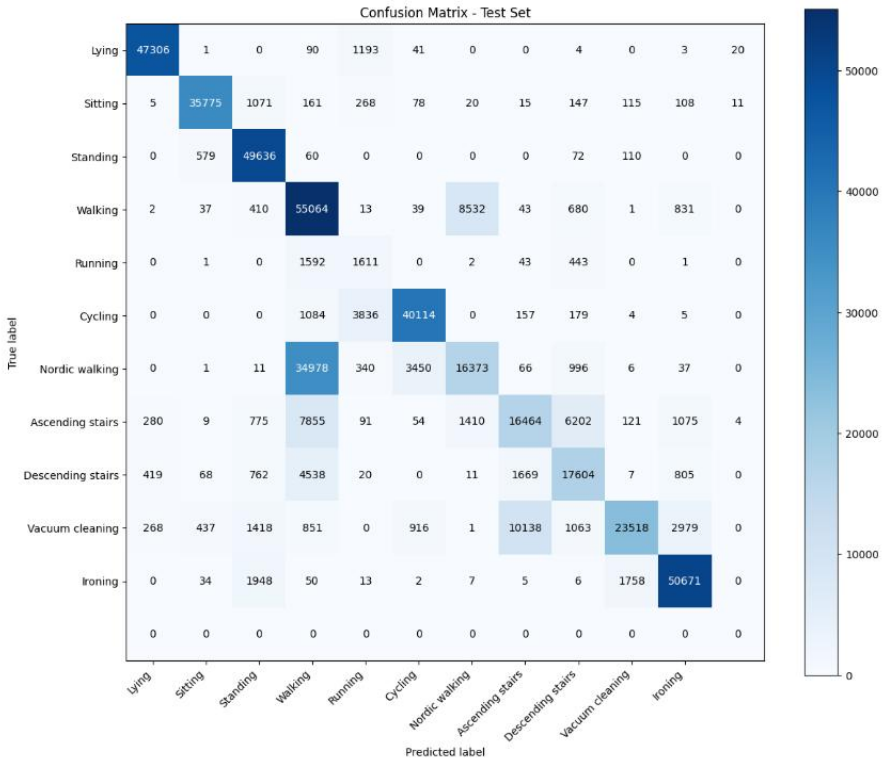
In this section, we present a detailed evaluation of the models used for human activity recognition, including comparisons of their performance, analysis of misclassifications, and other key metrics.

Model Performance Comparison

We compared the performance of three models: Random Forest, CatBoost, and Bidirectional LSTM (BiLSTM). The models were evaluated based on standard metrics such as accuracy, F1-score, precision, and recall. The BiLSTM model outperformed the others, especially in capturing the temporal patterns in the sensor data, which is crucial for time-series tasks.

Confusion Matrix and Misclassifications

A confusion matrix was generated for each model, showing which activities were most often misclassified. For example, "Walking" and "Running" were often confused with each other, which may be due to similar motion patterns. This insight suggests that additional features, like motion energy, could help distinguish between these activities more effectively.



Per-Class Performance

In addition to overall accuracy, the performance of each model was evaluated on a per-class basis. Some classes, such as "Running" and "Rope jumping", were harder to classify accurately due to their smaller representation in the dataset. Applying techniques like class weighting during training improved the model's performance on these underrepresented activities.

Error Analysis

We performed an error analysis to understand where the models struggled. Some subjects were consistently misclassified, indicating that the model may have overfitted to certain subjects. Further investigation into subject-specific patterns will help improve the model's generalization.

Temporal Stability

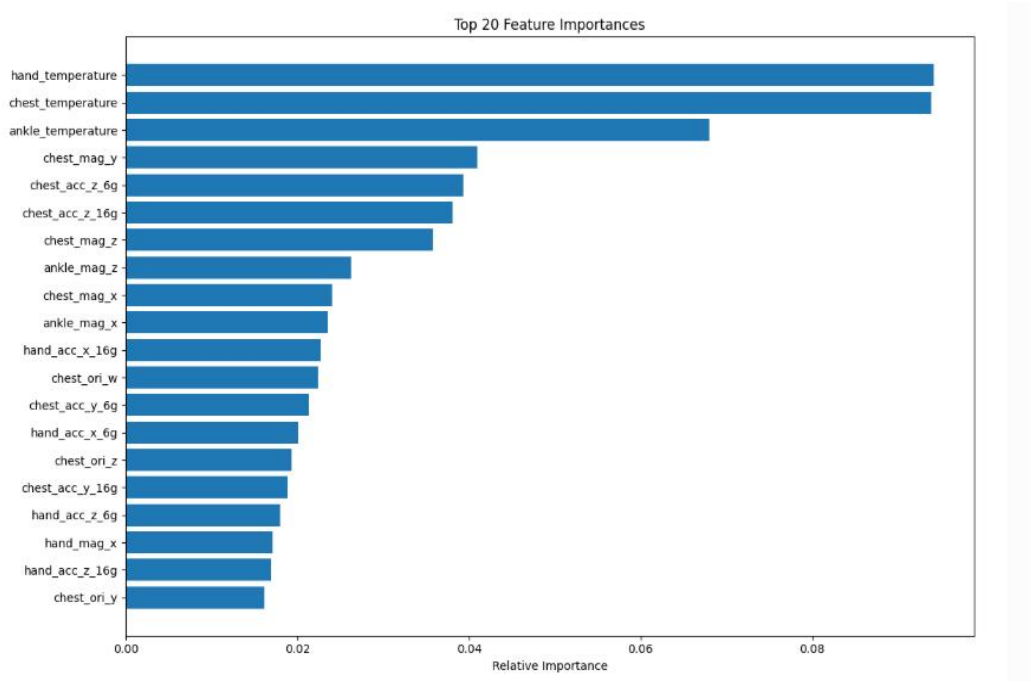
We also tested how stable the model's predictions were over time by splitting the test set into two halves. The model's accuracy remained consistent across both halves, confirming that the model was stable in its performance over time.

Subject-Specific Performance

Since activity recognition is subject-dependent, we analyzed how the models performed across different subjects. Some subjects had significantly higher accuracy than others, suggesting that the model's performance can vary based on individual data characteristics.

Feature Importance

For models like Random Forest and CatBoost, feature importance analysis was performed to identify which features contributed most to the predictions. Features related to acceleration magnitude and motion variability were found to be the most influential in distinguishing between activities.



7. Conclusion

In this project, we tackled the challenge of Human Activity Recognition (HAR) using data from wearable sensors. We explored multiple machine learning algorithms, including Random Forest, CatBoost, and Bidirectional LSTM (BiLSTM), to classify different activities based on sensor data. The project involved several key steps, such as data preprocessing, feature engineering, and model selection.

Key Findings

The **Bidirectional LSTM (BiLSTM)** model emerged as the best-performing model due to its ability to capture temporal dependencies in the data. This was especially important given the sequential nature of the sensor data.

The model achieved an impressive test accuracy of 99.56% and a weighted F1-score of 0.9956, outperforming other models, including Random Forest and CatBoost.

Stratified splitting of the dataset ensured that all activity classes were well-represented in the training, validation, and test sets, contributing to the model's ability to generalize.

Model Limitations

Class Imbalance: Despite using stratified splitting and class weights, some activities were still harder to classify due to their underrepresentation in the dataset. For example, activities like "Rope jumping" and "Running" were often misclassified as other similar activities like "Walking".

Subject-Specific Variability: The model's performance varied across subjects, indicating that individual differences in sensor readings may affect accuracy. Some subjects showed significantly higher classification accuracy than others.

Data Quality: Although the dataset was preprocessed and imputed for missing values, noise in the sensor data still posed challenges, especially for activities that involve rapid or complex motions.

Future Directions

Data Augmentation: Introducing techniques such as time warping or synthetic data generation could help address the class imbalance and improve performance on less frequent activities.

Transfer Learning: Using pre-trained models on similar datasets or applying domain-specific transfer learning techniques could improve the model’s generalizability, especially for subject-specific variations.

Advanced Temporal Models: Further improvements could be made by incorporating more advanced temporal models like Attention Mechanisms or Transformers, which are capable of focusing on important parts of the sequence.

Multi-modal Data: Incorporating additional sensor data (such as gyroscope readings or video data) could improve the model’s ability to distinguish between complex activities and enhance overall performance.