ML Project

Prepared by:
Kwanele Mabanga (MBNKWA005)
Zolile Zoko (ZKXZOL001)

Prepared for:
EEE4114F
Department of Electrical Engineering
University of Cape Town

May 30, 2025

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

K. Mabanga

May 30, 2025

Kwanele Mabanga

Date

Zolile Zoko

May 30, 2025

Zolile Zoko

Date

Contents

Chapter 1

Introduction

Human Activity Recognition (HAR) is an important area in computing and artificial intelligence that focuses on identifying human movements using data from sensors. It can detect actions like walking, sitting, and jogging, and has many useful applications. These include monitoring patients' wellbeing in healthcare, tracking fitness in wearable devices, such as Fitbit Charge 6, Apple Watch Ultra 2, Garmin Fenix, to name a few; improving smart home systems, and ensuring safety in workplaces.

Traditionally, HAR relied on external infrastructure like cameras. Today, smartphones and wearables with built-in motion sensors—called inertial measurement units (IMUs) make it easier to collect activity data simply and continuously. These sensors include accelerometers and gyroscopes, which measure motion and rotation.

The main challenge in HAR lies in processing noisy and complex time-series data and turning it into useful features that machine learning models can learn from. For this project, we use the MotionSense dataset, which was generated by an accelerometer and a gyroscope. The dataset was collected with an iPhone 6s kept in the participant's front pocket using Sensing Kit. The MotionSense dataset consists of IMU data (attitude, gravity, user acceleration, and rotation) collected from 24 participants performing 6 activities (walking, jogging, standing, sitting, climbing stairs, and going down stairs) in 15 trials

The goal of this project is to design, implement, and evaluate a machine learning model that can recognize the above-mentioned activities using the MotionSense data. This involves exploring different data pre-processing techniques, selecting appropriate features, training various machine learning models, and evaluating how well they perform. This report will detail the design choices made during the project, present the achieved results, and provide an analysis of these results, including any limitations and what could be improved in the future.

The rest of this report is organized as follows: Chapter 2 reviews related work on HAR, focusing on data processing and machine learning techniques. Chapter 3 details the design of the proposed HAR system, including data pre-processing, resampling methods, and the machine learning algorithms used. Chapter 4 presents the results of our experiments, followed by a discussion and analysis of these results in Chapter 5. Chapter 6 concludes the report with a summary of our findings and recommendations for future work.

Chapter 2

Literature Review

This chapter reviews existing research related to Human Activity Recognition (HAR), with a focus on key components relevant to our project. Specifically, it explores input data domain, data pre-processing techniques, feature selection methods, model selection and training, and evaluation strategies. The aim is to identify best practices and insights that can inform the design of a machine learning model capable of effectively utilizing the MotionSense dataset to recognize the activities outlined in Chapter 1. Our goal is to achieve high classification accuracy while maintaining acceptable generalization performance.

2.1 Input Data Domain: Time domain vs Frequency domain

The way data is expressed or represented has a significant impact on which features appear more prominent in the input. This section explores common input data domains. Whether input data is sampled from a stationary or non-stationary signal is an important consideration that must be evaluated before applying any transformation.

Naidu et al. point out that applying a Fast to a non-stationary signal results in a loss of time-varying properties. Thus, frequency domain transformations should only be applied to stationary signals. Furthermore, they evaluated the use of time-frequency representations as a means to retain timing information in non-stationary signals through transformations such as the Short-Time Fourier Transform (STFT), Continuous Wavelet Transform (CWT), and Wigner-Ville Distribution (WVD). Among these, the CWT showed superior time-frequency resolution for heart rate time series analysis.[1]

2.2 Data Pre-processing Techniques

Data pre-processing is a fundamental step in the development of machine learning models. Before feeding data into a model, data must be cleaned, transformed, and structured to meet the model's input requirements. Since different modeles have varying input expectations, the pre-processing techniques used can vary widely. For instance, some models require fixed-length input, while others handle sequential data. This section explores common data pre-processing methods and how they align with different model types.

A key challenge is aligning input dimensions with model requirements. Malekzadeh et al. note recurrent neural networks (RNNs) require input streams with a fixed structure, while feedforward neural networks (FNNs) require fixed-dimensional input. To address misaligned input sizes, they discuss using upsampling or downsampling during pre-processing, though these may reduce classification accuracy and risk removing important patterns. As an alternative, they explore global dimension pooling, which summarizes each CNN filter map into a one-dimensional vector independent of input size. While this avoids resizing, it ignores the spatial structure of the data, leading to accuracy loss [2].

Common pre-processing techniques include noise filtering, where methods like the Butterworth low-pass filter are used to remove unwanted high-frequency components while preserving important motion signals. Normalization or standardization ensures that no single feature dominates the learning process. Popular approaches include Z-score normalization and min-max scaling. Another standard method is segmentation, where data is split into fixed-size overlapping time windows to create structured inputs for machine learning models, and allows control over how much context the model sees[3].

In addition to these basics, more advanced techniques have emerged. For instance, Nouriani et al. use a physics-based model with a high-gain nonlinear observer to estimate the device's orientation (roll and pitch). They then convert this data into spectrograms, a visual representation of the frequency content of a signal over time [3].This transformation of time-series data into images allows the use of powerful pre-trained computer vision models for activity classification. Altunkaya et al. review similar techniques such as Gramian Angular Fields and Markov Transition Fields [4].

Real-world HAR must also handle variable input conditions, such as different sampling rates or missing sensors. Malekzadeh et al. address this with the Dimension-Adaptive Neural Architecture (DANA), which uses a Dimension-Adaptive Pooling (DAP) layer to allow neural networks to process inputs of varying sizes without needing complex resampling [2].

Finally, self-supervised learning (SSL) approaches, like those by Saeed et al., apply signal transformations to unlabeled data as pretext tasks. The model learns useful patterns through these tasks, which improves its performance when trained later on labeled data [5].

2.3 Feature Selection

Feature selection is essential to HAR performance. Traditional approaches rely on manual feature extraction, where time-domain features (such as mean, standard deviation, etc) and frequency-domain features (for example, from FFT) are manually computed.

With deep learning, models can automatically learn features. CNNs can learn spatial patterns from raw data, while RNNs (e.g., LSTMs and GRUs) capture how these patterns change over time[5][3]. Furthermore, self-supervised learning enhances representation learning by using pretext tasks. Saeed et al. show that models trained to recognize signal transformations on unlabeled data can learn features that perform well on HAR tasks, especially when labeled data is limited [5].

Ghasemzadeh et al. emphasize that feature selection must account for the limited power resources of wearable devices. Their work introduces a power-aware feature selection method that incorporates the energy cost of computing each feature in real time. By modeling the correlation and computational complexity of features using a graph-based approach, they formulate the selection problem as an integer programming task and propose a greedy approximation algorithm. Their experimental evaluation on activity data demonstrates over 30% energy savings while maintaining 96.7% classification accuracy[6].

2.4 Model selection

A variety of machine learning models are used in HAR. Classical models like SVMs (Support Vector Machines), Random Forests, KNNs (K-Nearest Neighbors), etc., are popular, especially when using hand-crafted features.

Deep learning models, however, offer more flexibility and often superior performance. 1D CNNs are effective for detecting local patterns in sequential sensor data, while 2D CNNs, such as ResNet and VGG, can be used when the data is transformed into images like spectrograms [3]. RNNs, particularly LSTMs (Long Short-Term Memory) and GRUs (Gated Recurrent Units), are great at modeling long-term dependencies in time-series data and are often used with CNNs to combine spatial and temporal learning.

For scenarios involving variable input dimensions, adaptive models like DANA (Malekzadeh et al.) have been developed. DANA introduces DAP layers that allow the model to adapt to missing sensor channels or different sampling rates, making models more flexible in real-world scenarios [2].

2.5 Model Training

How a model is trained greatly affects its performance. Supervised learning, where models are trained directly on labeled sensor data, remains the most common approach. However, it often requires large labeled datasets, which may not always be available. Self-supervised learning (SSL) helps address this issue. Saeed et al. use SSL first on unlabeled data and then fine-tune the model with limited labeled data, boosting the performance, especially in low-data or transfer learning settings [5]. Nouriani et al. apply transfer learning by fine-tuning image-based CNNs pre-trained on ImageNet to classify spectrograms of sensor data. [3] Similarly, Saeed et al. use features learned on one dataset to improve results on others with less labeled data. [5]

To improve generalization across varying input configurations, Dimension-Adaptive Training (DAT), proposed by Malekzadeh et al., introduces random variations in sensor availability and sampling rates during training, making the model more robust to different real-world scenarios. [2]

2.6 Model Evaluation

Evaluating HAR models requires both standard and advanced practices. Common metrics include accuracy, precision, recall, F1-score, and confusion matrices. Cohen's Kappa is sometimes used to better assess performance in cases of class imbalance [5]

Proper cross-validation is essential. User-independent validation, such as leave-one-subject-out, is recommended to test how well a model generalizes to unseen users. Random train-test splits can produce overly optimistic results if data from the same user is used in both sets [5].

2.7 Conclusion

This chapter has reviewed key aspects of Human Activity Recognition (HAR) relevant to the development of an effective machine learning model using the MotionSense dataset. It explored how different input data domains affect feature representation, highlighted essential data pre-processing techniques, and examined both traditional and deep learning-based approaches to feature selection and model training. Chapter 3 builds on this foundation by detailing the design of the proposed HAR system, including data pre-processing, resampling methods, and the machine learning algorithms used.

Chapter 3

Design

This chapter describes the methodology used for developing and implementing the Human Activity Recognition (HAR) system. It covers the dataset used, data pre-processing steps, feature selection approaches, the machine learning algorithms used, and the experimental setup for training and evaluation.

3.1 Data Description

The primary dataset used in this project is the MotionSense dataset. This dataset contains inertial measurement unit (IMU) data collected from 24 participants performing six distinct activities: walking, jogging, upstairs, downstairs, sitting, and standing. For each participant and activity, the dataset provides time-series data from the iPhone 6s's IMU, including:

- `attitude (roll, pitch, yaw)`
- `gravity (x, y, z)`
- `userAcceleration (x, y, z)`
- `rotationRate (x, y, z)`

The data was collected at a sampling rate of 50 Hz.

For this project, `userAcceleration` and `rotationRate` were selected as the primary input features for activity recognition. These two sensor streams directly capture the dynamics of human motion:

- `userAcceleration` provides information about voluntary movements such as steps, strides or posture changes, making it highly informative for distinguishing between locomotor activities (e.g., walking vs jogging) and stationary ones (e.g., standing vs sitting)
- `rotationRate`: captures the angular motion of the device and is particularly useful in detecting changes in orientation during dynamic actions like climbing stairs, or going down the stairs.

The `attitude` and the `gravity` sensor streams were excluded for the following reasons:

- `attitude` represents the orientation of the device, which can vary significantly depending on the phone's exact placement in the pocket and may introduce inconsistency across users.
- `gravity` provides a static reference of the gravitational vector. While potentially useful for postural orientation, it contributes less directly to capturing movement dynamics and is partly

redundant with user acceleration.

3.2 Model Design Pipeline Block diagrams

The following models were implemented to predict user activity based on input time-series data.

CWT + CNN Pipeline Model Block Diagram

Figure 3.1 below presents the block diagram of the implemented CWT + CNN pipeline. This model was chosen because the use of the CWT enables the network to leverage both temporal and frequency information from the input signal.

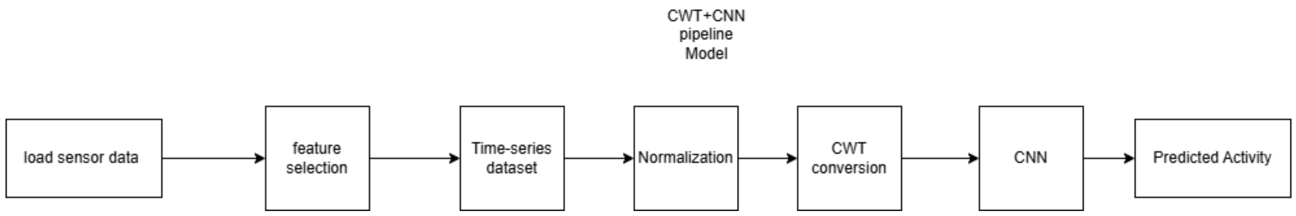

Figure 3.1: CWT + CNN Pipeline Implementation High Level Block Diagram

CNN Model Block Diagram

Figure 3.2 below shows the block diagram of the implemented CWT and CNN pipeline. This design was chosen to investigate the impact of focusing solely on temporal data and how this affects the model's performance.

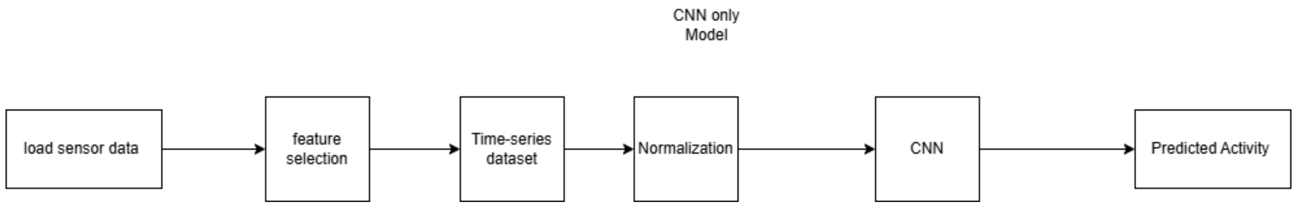

Figure 3.2: CNN Model Implementation High Level Block diagram

3.3 Feature Selection

As previously mentioned in Section 3.1, the `userAcceleration` and `rotationRate` raw data were selected as the primary input features for activity recognition. This early-stage decision was based on their ability to capture both translational and rotational motion, which are key to distinguishing between physical activities. These selected sensor types were used to extract relevant feature groups from the dataset during the initial pre-processing phase.

3.4 Data Pre-processing

After selecting the desired sensor raw data, the dataset is loaded and labeled based on trial activity mappings. The raw feature values are then standardized using `StandardScaler` to ensure uniform scaling. For the CWT + CNN model, the standardized time-series is segmented into overlapping windows, and each window is transformed using the Continuous Wavelet Transform (CWT) to capture both temporal and frequency-domain features. These CWT representations are reshaped to match the input format expected by the CNN. Finally, the data is split into training and testing subsets using stratified sampling to preserve label distribution across splits.

3.5 Machine Learning Algorithms

A Convolutional Neural Network (CNN) was used in both pipelines, as described in Sections 3.2 and 3.2. In the CWT + CNN model, the CNN is applied to windowed Continuous Wavelet Transform (CWT) images to learn discriminative patterns corresponding to different activities. Once trained, the model was able to predict the user's activity based on these learned features.

3.6 Experimental Setup

To evaluate the effectiveness of the two proposed activity recognition pipelines, a Convolutional Neural Network (CNN) model was implemented using PyTorch.

The CNN architecture consists of three 1D convolutional layers with increasing channel depths. Each convolutional layer is followed by batch normalization, ReLU activation, max pooling, and dropout regularization. This hierarchical structure enables the model to progressively extract high-level temporal features from the input. After feature extraction, the output is flattened and passed through two fully connected layers with dropout before being mapped to the final activity classes via a classification layer.

To thoroughly assess model performance, both the two-way and three-way data splits were employed during training and evaluation. For each model design, accuracy and loss curves were generated to monitor training progress. Additionally, confusion matrices were produced to provide detailed insights into the class-wise prediction performance.

Model evaluation metrics included overall accuracy as well as precision, recall, and F1-score for each activity class, summarized in classification reports. Visualization of the confusion matrices further elucidated strengths and weaknesses in distinguishing between specific activity categories.

Chapter 4

Results

This chapter presents the performance results of the implemented Convolutional Neural Network (CNN) models on the human activity recognition task. The evaluation is performed using both two-way and three-way data splits, and later with Continuous Wavelet Transform (CWT) images as inputs.

The following visualizations and tables are included for each experiment:

- **Accuracy-Loss Plots**: These plots show the training and validation accuracy and loss across epochs. Accuracy indicates the proportion of correctly predicted samples, while loss reflects the model's error. These plots help assess how well the model is learning and whether overfitting or underfitting is occurring.
- **Classification Report**: This includes key performance metrics for each activity class:
  - **Precision**: The proportion of correctly predicted instances among all instances predicted for a class.
  - **Recall**: The proportion of correctly predicted instances among all actual instances of that class.
  - **F1-score**: The harmonic mean of precision and recall, providing a balance between the two.
  - **Support**: The number of true instances for each class in the dataset.

  Macro and weighted averages are also provided, summarizing overall model performance.
- **Confusion Matrix**: This matrix visualizes the number of correct and incorrect predictions for each class. The diagonal elements represent correct predictions, while off-diagonal elements indicate misclassifications.

4.1 CNN (Two-way Split)

Accuracy-Loss Plots

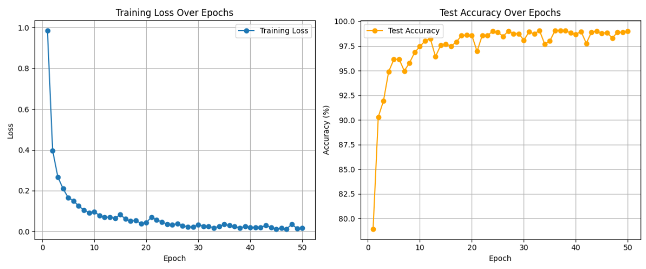

Figure 4.1: Accuracy-Loss Plots: Two-way Split

Classification Report

	precision	recall	f1-score	support
dws	1.00	0.96	0.98	264
ups	0.96	1.00	0.98	314
wlk	0.99	0.99	0.99	689
jog	1.00	0.99	0.99	268
std	1.00	0.99	0.99	614
sit	0.99	1.00	0.99	677
accuracy		0.99		2826
macro avg		0.99	0.99	2826
weighted avg		0.99	0.99	2826

Table 4.1: Classification Report Metrics

Confusion Matrix

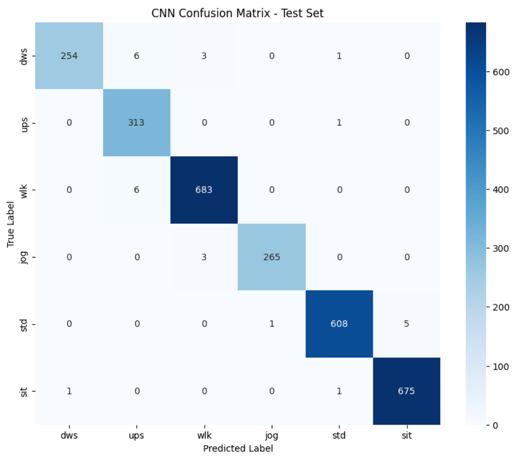

Figure 4.2: Confusion Matrix: Two-way Split

4.2 CNN (Training-Validation-Test Split)

Accuracy-Loss Plots

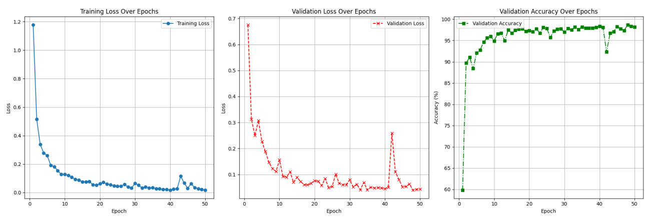

Figure 4.3: Accuracy-Loss Plots: Three-way Split

Classification Report

Classification Report

	precision	**recall**	**f1-score**	**support**
dws	0.98	0.98	0.98	264
ups	0.97	0.99	0.98	314
wlk	1.00	0.99	0.99	689
jog	1.00	0.99	0.99	268
std	0.99	0.99	0.99	614
sit	0.99	0.99	0.99	677
accuracy: 0.99				
macro avg	0.99	0.99	0.99	2826
weighted avg	0.99	0.99	0.99	2826

Table 4.2: Classification Report Metrics

Confusion Matrix

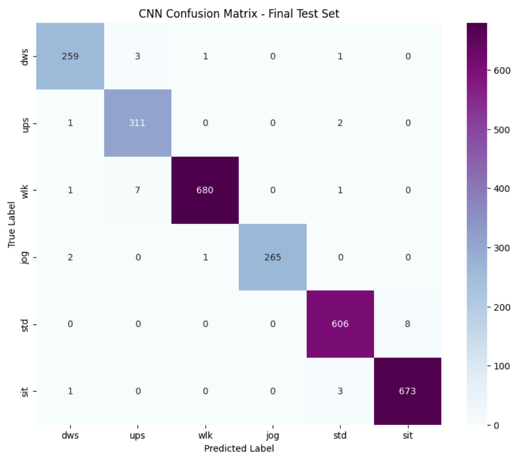

Figure 4.4: Confusion Matrix: Three-way Split

4.3 CWT + CNN (Training-Test Split)

Accuracy-Loss Plots

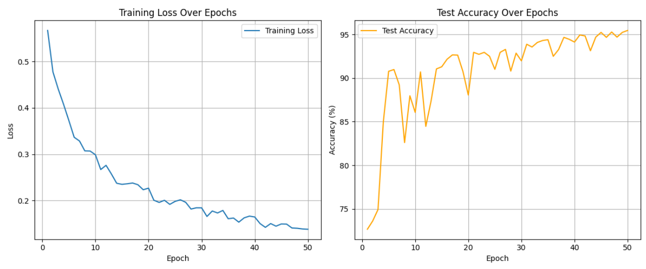

Figure 4.5: Accuracy-Loss Plots: CWT-CNN

Classification Report

	precision	recall	f1-score	support
dws	0.97	0.96	0.97	411
ups	0.97	0.97	0.97	492
wlk	0.98	0.99	0.98	1076
jog	0.99	0.98	0.98	419
std	0.93	0.91	0.92	958
sit	0.92	0.94	0.93	1059
accuracy	-	-	0.95	4415
macro avg	0.96	0.96	0.96	4415
weighted avg	0.95	0.95	0.95	4415

Table 4.3: Classification Report Metrics

Confusion Matrix

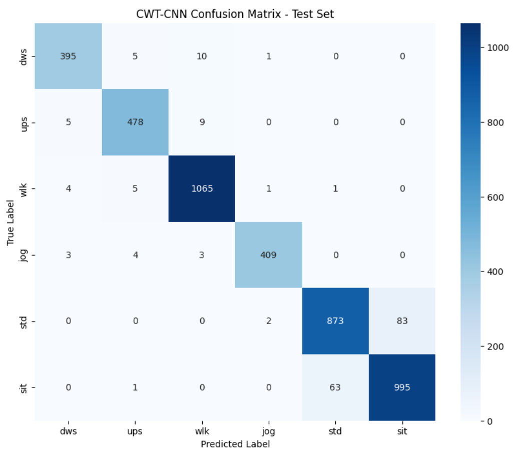

Figure 4.6: Confusion Matrix: CWT + CNN

Chapter 5

Discussion

This chapter provides an analysis and interpretation of the results presented in Chapter 4.

5.1 CNN only implementation

Based on the results in Table 4.1, the CNN-only model demonstrates strong performance across all activities with high precision, recall, and F1-scores, indicating effective activity recognition.

The accuracy and loss plots for the two-way split (Figure 4.2) show consistent improvement: training loss decreases toward zero, and test accuracy rises above 90%, reflecting good generalization. Minor accuracy fluctuations likely stem from activity complexity or slight overfitting, but the overall trend is positive.

Results from the three-way split (Figure 4.3) confirm these findings, showing similar learning patterns and validating the model's reliability. The confusion matrix (Figure 4.4) further supports accurate predictions, with most results along the main diagonal.

5.2 CWT+CNN Implementation Summary

The CWT+CNN pipeline enhances the input representation by transforming sensor time-series data into time-frequency images using Continuous Wavelet Transform, allowing the model to learn richer temporal and frequency features. Despite increased input complexity causing greater training variability and slower loss convergence, the model achieves strong classification performance with an overall accuracy of 95% and a macro average F1-score of 0.96 (Table **??**).

Some slight performance drops occur for the 'stand' and 'sit' classes, with F1-scores of 0.92 and 0.93 respectively. Accuracy-loss plots (Figure 4.5) reflect fluctuations likely due to higher input dimensionality and potential feature redundancy. The confusion matrix (Figure 4.6) confirms reliable predictions across all classes.

Overall, the CWT+CNN approach effectively captures complex activity patterns and offers a valuable enhancement over simpler models when richer input features are needed.

Chapter 6

Conclusion

In this project, we developed and tested machine learning models for Human Activity Recognition (HAR) using the MotionSense dataset. By using sensor data from smartphones specifically user acceleration and rotation rate,we were able to train CNN-based models to accurately recognize six daily activities.

The basic CNN model performed very well, showing high accuracy with time-series data. When we used the Continuous Wavelet Transform (CWT) to convert the data into images, the model captured more detailed patterns, improving performance for some activities, though training became slightly more complex.

Overall, both models worked effectively, each with its own strengths. The CNN-only model is simple and fast, while the CWT + CNN model provides richer information for better accuracy. This shows that with the right data and model design, HAR systems can be accurate and useful for real-world applications.

Bibliography

[1] V. Naidu and P. Mahalakshmi, “Time-frequency analysis of heart rate time series,” in *2004 IEEE Region 10 Conference TENCON 2004*. IEEE, 2004, pp. 215–218.

[2] M. Malekzadeh, R. Clegg, A. Cavallaro, and H. Haddadi, “Dana: Dimension-adaptive neural architecture for multivariate sensor data,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 3, pp. 1–27, 2021.

[3] A. Nouriani, R. McGovern, and R. Rajamani, “Activity recognition using a combination of high gain observer and deep learning computer vision algorithms,” *Intelligent Systems with Applications*, vol. 18, p. 200213, 2023.

[4] D. Altunkaya, F. Y. Okay, and S. Ozdemir, “Image transformation for iot time-series data: A review,” *arXiv preprint arXiv:2311.12742*, 2023.

[5] A. Saeed, T. Ozcelebi, and J. Lukkien, “Multi-task self-supervised learning for human activity detection,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 2, pp. 1–30, 2019.

[6] H. Ghasemzadeh, N. Amini, R. Saeedi, and M. Sarrafzadeh, “Power-aware computing in wearable sensor networks: An optimal feature selection,” *IEEE Transactions on Mobile Computing*, vol. 14, no. 4, pp. 800–812, 2014.