EFFICIENT HUMAN ACTIVITY RECOGNITION USING FEATURE-OPTIMIZED DEEP LEARNING APPROACH

A project report submitted by

RHENIUS A – URK21CS1073

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

under the supervision of

Dr. J. ANITHA M.E., Ph. D

Associate Professor

DIVISION OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTER SCIENCE AND TECHNOLOGY

KARUNYA INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed to be University -under Sec-3 of the UGC Act, 1956)

Karunya Nagar, Coimbatore - 641 114. INDIA

APRIL 2025

Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES

(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)

MoE, UGC & AICTE Approved

NAAC A++ Accredited

DIVISION OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"EFFICIENT HUMAN ACTIVITY RECOGNITION USING FEATURE-OPTIMIZED DEEP LEARNING APPROACH"** is the bonafide work of **RHENIUS A (Reg. No: URK21CS1073)** who carried out the project work under my supervision.

SIGNATURE

Dr. J. Immanuel Johnraja

Head of the Division

Division of Computer Science and Engineering

SIGNATURE

Dr. J. Anitha

Supervisor

Associate Professor

Division of Computer Science and Engineering

Submitted for the Project Viva Voce held on **29.04.2025**

Examiner

Karunya INSTITUTE OF TECHNOLOGY AND SCIENCES
(Declared as Deemed to be University under Sec.3 of the UGC Act, 1956)
MoE, UGC & AICTE Approved; NAAC Accredited A++
Karunya Nagar, Coimbatore - 641 114, Tamil Nadu, India.

School of Computer Science and Technology

DIVISION OF COMPUTER SCIENCE AND ENGINEERING

KITS/CSE/PCC/2025/58 23.04.2025

CERTIFICATE

This is to certify that RHENIUS A [URK21CS1073] carried out the project titled EFFICIENT HUMAN ACTIVITY RECOGNITION USING FEATURE-OPTIMIZED DEEP LEARNING APPROACH from December 09, 2024 to April 23, 2025 in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Karunya Institute of Technology and Sciences.

Guide	Project Coordinator	Deputy Head – CSE	HoD-CSE

ACKNOWLEDGEMENT

First and foremost, I praise and thank **ALMIGHTY GOD** for giving me the will power and confidence to carry out my project.

I am grateful to my beloved founders Late **Dr. D.G.S. Dhinakaran**, **C.A.I.I.B, Ph.D.,** and **Dr. Paul Dhinakaran**, **M.B.A., Ph.D.,** for their love and always remembering me in their prayers.

I extend my thanks to **Dr. G. Prince Arulraj, Ph.D.,** Vice chancellor, **Dr. R. Elijah Blessing, Ph.D.,** Pro-Vice Chancellor, and **Dr. S. J. Vijay, Ph.D.,** Registrar for giving me the opportunity to carry out this project.

I would like to place my heart-felt thanks and gratitude to **Dr. J. Immanuel Johnraja, Ph.D.,** HOD, Division of Computer Science and Engineering and **Dr. G. Jaspher W. Kathrine, Ph.D.,** Deputy Head, Division of Computer Science and Engineering for their encouragement and guidance.

I am grateful to my guide, **Dr. J. Anitha, M.E., Ph. D** Associate Professor, Division of Computer Science and Engineering for her valuable support, advice and encouragement.

I also thank all the staff members of the Division for extending their helping hands to make this project work a success.

I would also like to thank my parents and all my friends who have prayed and helped me during the project work.

ABSTRACT

Human Activity Recognition (HAR) is a critical area in healthcare, fitness tracking, and intelligent environments, requiring accurate and efficient classification of human movements from sensor data. Traditional machine learning models often struggle with high-dimensional features, noisy data, and similar activity patterns, leading to reduced performance and increased computational costs. To address these challenges, an Efficient Human Activity Recognition approach is proposed, integrating LASSO Regression for feature selection and Temporal Convolutional Networks with Attention Mechanism (TCN + AM) for enhanced classification accuracy. For reliable data preprocessing, Winsorization is implemented for outlier handling, maintaining data integrity and minimizing the influence of extreme values. LASSO Regression is employed to rank and extract the most relevant features, reducing redundancy while preserving essential information. The Attention Mechanism refines temporal dependencies, allowing the model to focus on key patterns, thereby improving classification precision. Results highlight the effectiveness of integrating optimized feature selection with deep learning models for HAR. Multiple deep learning architectures, including TCN, TCN + AM, Bidirectional TCN (Bi-TCN), and Bi-TCN + AM, are evaluated on different feature subsets (256, 150, 100, and 50). Among these, TCN + AM with 150 features outperforms all other models, achieving an accuracy of 98.74%, demonstrating superior generalization and computational efficiency. The proposed approach provides a robust, efficient, and scalable solution for HAR, making it well-suited for real-time applications in wearable devices, healthcare monitoring, and intelligent surveillance systems. Future work will explore additional model optimizations and real-world deployment strategies.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	BONAFIDE CERTIFICATE	**ii**
	PROJECT COMPLETION CERTIFICATE	**iii**
	ACKNOWLEDGEMENT	**iv**
	ABSTRACT	**v**
	TABLE OF CONTENTS	**vi**
	LIST OF FIGURES	**viii**
	LIST OF TABLES	**ix**
	LIST OF ABBREVIATIONS	**x**
1	**INTRODUCTION**	**1**
	1.1 Motivation of the Work	1
	1.2 Problem Statement	1
	1.3 Objective	2
	1.4 Chapter wise Summary	2
2	**LITERATURE SURVEY**	**4**
	2.1 Existing System	4
	2.2 Inference from the Survey	6
3	**PROPOSED METHODOLOGY**	**8**
	3.1 Proposed System	8
	3.2 Dataset	8
	3.3 Train Test Split	9
	3.4 Pre-Processing Data	10
	3.4.1 Outlier Detection	10
	3.4.2 Data Imputation	10
	3.5 Feature Selection	11
	3.5.1 LASSO Regression	12
	3.6 TCN Model Description	13

3.6.1 TCN 14
3.6.2 Activation Function: ReLU 14
3.6.3 Attention Mechanism (AM) 15
3.6.4 TCN + AM 16
3.6.5 Bi-TCN 17
3.6.6 Bi-TCN + AM 17
3.7 Explainable AI 18
3.7.1 SHAP 19
4 EXPERIMENTAL RESULTS AND DISCUSSIONS 20
4.1 Implementation 20
4.2 Performance Metrics 23
4.2.1 Accuracy 23
4.2.2 Precision 24
4.2.3 Recall 24
4.2.4 F1-Score 24
4.2.5 Specificity 25
4.2.6 ROC-AUC 25
4.2.7 Training Time 26
4.2.8 Inference Time 26
4.3 Results and Discussions 27
4.4 Inference from SHAP Plots 33
5 CONCLUSION AND FUTURE SCOPE 38
5.1 Conclusion 38
5.2 Future Scope 39
ANNEXURE 1 - SOURCE CODE 41
REFERENCES 50
PROJECT EVALUATION FORM 53

LIST OF FIGURES

FIG NO	TITLE	PAGE NO
3.1	Proposed Architecture for Human Activity Recognition System	8
3.2	Dataset Class Distribution	9
3.3	Representation of Outlier	10
3.4	Winsorization for Outlier Replacement	11
3.5	Importance Feature after LASSO Regression	13
3.6	TCN Model Block Diagram	14
3.7	Activation Function – ReLU	15
3.8	Attention Mechanism (AM)	16
3.9	TCN + AM Model Block Diagram	16
3.10	Bi-TCN Model Block Diagram	17
3.11	Bi-TCN + AM Model Block Diagram	18
4.1	Accuracy & Loss Graph for TCN + AM with 256 Features	28
4.2	Accuracy & Loss Graph for TCN + AM with 150 Features	28
4.3	Model Accuracy Chart for all Models with 150 Features	30
4.4	Feature Importance Analysis using SHAP Values	33
4.5	SHAP Dependency Plot for tGravityAcc-energy()-X	34
4.6	SHAP Waterfall Plot for Feature Contribution	35
4.7	SHAP Summary Plot for Feature Contribution	36

LIST OF TABLES

TABLE NO	TITLE	PAGE NO
4.1	Performance of Time Series Models with Different Number of Features	27
4.2	Performance of TCN Variants with Different Number of Features	29
4.3	Performance Comparison of TCN-Based Models with 150 Features	31

LIST OF ABBREVIATIONS

Abbreviation	Full Form
HAR	Human Activity Recognition
SVM	Support Vector Machine
kNN	k-Nearest Neighbors
TCN	Temporal Convolutional Networks
AM	Attention Mechanism
GAP	Global Average Pooling
Bi-TCN	Bidirectional Temporal Convolutional Network
LASSO	Least Absolute Shrinkage and Selection Operator
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
XAI	Explainable Artificial Intelligence
SHAP	SHapley Additive exPlanations
LIME	Local Interpretable Model-Agnostic Explanations
PCA	Principal Component Analysis
IoT	Internet of Things
UCI	University of California, Irvine
ReLU	Rectified Linear Unit
ROC-AUC	Receiver Operating Characteristic - Area Under Curve
TPR	True Positive Rate
FPR	False Positive Rate
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives
CPU	Central Processing Unit
GPU	Graphics Processing Unit
ADAM	Adaptive Moment Estimation (optimizer)

CHAPTER 1

INTRODUCTION

Human Activity Recognition (HAR) refers to the automated identification of physical activities such as walking, sitting, standing, or running, using data from sensors embedded in smartphones or wearable devices [1], [8]. It has significant applications in fields such as healthcare, elderly care [25], smart homes, and fitness monitoring. With advancements in sensor technology and deep learning [5], HAR systems have become increasingly sophisticated. However, the challenges posed by high-dimensional feature spaces [2], noisy data [3], and overlapping activity patterns remain pressing concerns in achieving accurate and efficient classification. This project aims to address these challenges by combining effective feature selection [2] and optimized deep learning models [5] to improve the overall performance of HAR systems.

1.1 MOTIVATION OF THE WORK

Human Activity Recognition plays a vital role in modern applications where real-time monitoring and context awareness are crucial [22]. In healthcare, accurate activity recognition supports remote patient monitoring and fall detection [7]. In fitness, it enables activity tracking and personalized coaching [8]. Despite the rapid development in deep learning [5], current models often fail to meet the demands for accuracy, computational efficiency, and real-time deployment [22]. Most HAR datasets include hundreds of extracted features, many of which are redundant or irrelevant, causing unnecessary computational load and reduced model performance [2].

These challenges motivated the development of a feature-optimized deep learning framework that integrates LASSO Regression for effective feature selection [2] and Temporal Convolutional Networks (TCNs) [23] with an Attention Mechanism (AM) [4] for enhanced temporal modeling [6]. The motivation lies in achieving a HAR system that is not only accurate but also efficient enough for real-world, real-time applications on edge devices.

1.2 PROBLEM STATEMENT

Current HAR systems face a range of limitations that hinder their effectiveness and practical deployment. Noisy sensor readings [3], similar activity patterns, and a high-dimensional feature space [2] introduce complexities that reduce classification accuracy.

Traditional machine learning models often struggle with redundant features, which leads to inefficient training and limited scalability [24]. On the other hand, deep learning models like RNNs, LSTMs, and GRUs [11], while capable of learning sequential dependencies, are plagued by issues such as vanishing gradients and long training times.

Additionally, the standard HAR datasets, such as UCI HAR [28], contain up to 561 extracted features, making real-time deployment difficult due to excessive computational requirements. These issues necessitate a new approach that reduces dimensionality and enhances feature prioritization [2]. To address this, LASSO Regression is proposed for selecting the most significant features [2] and integrating TCN [15] with an Attention Mechanism [4] to model time-based dependencies more effectively [6]. This results in a lightweight, accurate, and scalable HAR system suitable for real-time environments [21].

1.3 OBJECTIVE

The goal of this work is to develop an optimized Human Activity Recognition (HAR) system by integrating LASSO Regression for feature selection [2] and Temporal Convolutional Networks (TCN) [15] with an Attention Mechanism (AM) [4] for improved classification accuracy. HAR is essential in applications like healthcare [7], intelligent environments [22], and fitness tracking [8], where accurate classification of human activities based on sensor data is crucial. Traditional models face challenges such as high feature dimensionality [2], computational inefficiencies [24], and overlapping activity patterns [21], reducing their effectiveness. To address these limitations, a feature-optimized deep learning approach is proposed, ensuring both accuracy and scalability. The model's performance is evaluated using key metrics, including accuracy, precision, recall, specificity, F1-score, and ROC-AUC, to validate its effectiveness.

1.4 CHAPTER-WISE SUMMARY

Chapter 1: Introduction

This chapter introduces the concept of Human Activity Recognition (HAR) and highlights its importance in domains such as healthcare, smart environments, and fitness tracking. It outlines the motivation behind the work, identifies the key challenges in existing HAR models such as high-dimensional feature space and noisy data, and presents the objectives of developing a feature-optimized deep learning framework using LASSO Regression and Temporal Convolutional Networks with Attention Mechanism (TCN + AM).

Chapter 2: Literature Survey

This chapter reviews existing work in the field of HAR, analyzing traditional machine learning and modern deep learning techniques. It highlights the limitations of current models, such as computational inefficiency and limited temporal modeling capabilities. The chapter infers the need for integrated solutions that combine feature selection, temporal modeling, and attention mechanisms to improve classification performance.

Chapter 3: Proposed Methodology

This chapter details the design and methodology of the proposed HAR system. It describes the dataset used, the train-test split, and preprocessing steps including outlier detection using Z-score, data imputation using Winsorization, and feature normalization. It introduces LASSO Regression for feature selection and explains the architecture of four deep learning models: TCN, TCN + AM, Bi-TCN, and Bi-TCN + AM. It also presents the use of SHAP (SHapley Additive exPlanations) for model interpretability.

Chapter 4: Experimental Result and Discussion

This chapter discusses the implementation of the proposed models and evaluates their performance using various metrics such as accuracy, precision, recall, F1-score, specificity, ROC-AUC, training time, and inference time. Comparative analysis is presented for different model variations and feature subsets. The effectiveness of the attention mechanism and feature selection is validated, and insights from SHAP visualizations are included to interpret model predictions.

Chapter 5: Conclusion and Future Scope

This chapter concludes the project by summarizing the results and emphasizing the effectiveness of the TCN + AM architecture combined with LASSO-selected features. It confirms the proposed framework's suitability for real-time applications and outlines future directions including multi-sensor fusion, hybrid deep learning models, and deployment on edge devices for enhanced scalability and performance.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Human Activity Recognition (HAR) has gained significant attention for applications in healthcare, fitness tracking, and intelligent environments, requiring accurate classification of human movements from sensor data. Traditional HAR systems have primarily relied on machine learning techniques like k-Nearest Neighbors (kNN), Support Vector Machines (SVM), and Decision Trees [1], [13], [24]. While effective in controlled settings, these models often underperform with real-time, high-dimensional sensor data. Their limitations stem from manual feature engineering, inability to capture temporal patterns, and sensitivity to noisy or redundant data inputs.

Feature selection is crucial in HAR systems, as sensor data is often high-dimensional, redundant, and noisy, which can impact model performance and training time. While LASSO (Least Absolute Shrinkage and Selection Operator) regression is commonly used for feature selection [2], techniques like Principal Component Analysis (PCA) [26] and Mutual Information-based feature selection [8] have also been applied to improve model performance. Alternative outlier detection methods, such as Isolation Forest [27] and Local Outlier Factor (LOF), are used, alongside imputation techniques like K-Nearest Neighbors imputation and mean or mode imputation to handle missing data. These methods improve model robustness by reducing redundancy and preserving key information.

As wearable technology and IoT sensors have become more widespread, the volume and complexity of data have increased, necessitating a transition to more advanced models. Deep learning approaches have since become a dominant choice in HAR systems. Convolutional Neural Networks (CNNs) are widely used to capture spatial dependencies in sensor data, especially from accelerometers and gyroscopes [10], [14]. These models automatically extract relevant features and remove the need for hand-crafted features. However, CNNs alone cannot effectively model the temporal nature of activity data.

These limitations are addressed by employing Recurrent Neural Networks (RNNs), including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures [9], [11], [23]. These models are capable of capturing time-based sequences and

dependencies, which are essential for recognizing patterns in human activities. However, RNNs face challenges such as long training times and vanishing gradient problems, especially with long input sequences.

Temporal Convolutional Networks (TCNs) have emerged as an efficient and scalable alternative to RNNs. TCNs use dilated convolutions and causal structures to capture long-range dependencies in time-series data, allowing them to process sequences with better parallelism and stable gradient flow [15], [23]. These models outperform RNNs in both accuracy and computational efficiency for HAR tasks. In addition, bidirectional variants of TCNs (Bi-TCNs) have been proposed to capture both forward and backward temporal dependencies, enhancing the context understanding of sequential data [21], [22].

While TCNs offer substantial improvements, the uniform treatment of all input features can be a limitation. Attention mechanisms have been introduced to resolve this by dynamically assigning importance to specific time steps during training. This allows the model to focus on the most relevant parts of the data while downplaying irrelevant or noisy inputs. Studies show that integrating attention mechanisms with TCNs improves classification accuracy and enhances the model's ability to distinguish between similar or overlapping activities [4], [5], [6].

Outlier detection and handling is another critical part of preprocessing in HAR systems. Sensor readings are susceptible to noise caused by motion artifacts, sensor displacement, or environmental interference. The Z-score method is commonly used for outlier detection, identifying extreme values that deviate significantly from the mean [12], [20]. Winsorization complements this by capping outliers at set percentiles, preserving the structure of the dataset while mitigating the influence of anomalies [3], [17]. Outlier detection with IQR (Interquartile Range) is also used as an alternative approach for handling outliers in HAR systems [18] [19].

The UCI HAR dataset is one of the most frequently used benchmarks in the field. It contains sensor readings from accelerometers and gyroscopes collected from smartphones placed at the waist of 30 participants while performing six physical activities [28]. The dataset has been instrumental in testing and comparing various HAR models, including CNNs, RNNs, TCNs, and hybrid models.

Recent advancements have also focused on deploying HAR models on edge devices

such as smartphones and wearables. Lightweight architectures, particularly those based on TCNs, have been developed for real-time performance with limited computational resources [9], [23]. These models maintain high classification accuracy while reducing memory usage and latency, making them suitable for real-world applications.

Along with performance, model explainability has become a key focus in recent research. Explainable AI (XAI) techniques such as SHAP (SHapley Additive exPlanations) are used to interpret deep learning models in HAR applications. SHAP assigns importance values to each input feature, enabling researchers and practitioners to understand which features contribute most to model predictions [22]. This transparency is especially valuable in healthcare applications [7], where understanding model decisions is critical for trust and adoption.

Lastly, multimodal data fusion has also been explored to enhance HAR accuracy. Combining data from multiple sensors—such as accelerometers, gyroscopes, magnetometers, and environmental sensors—improves robustness and helps capture more complex activity patterns [4], [21]. Attention mechanisms, when applied to multimodal sensor data, further improve model performance by weighting the contribution of each sensor channel based on its relevance to the current activity.

In summary, the evolution of HAR systems has moved from traditional machine learning to advanced deep learning techniques that integrate robust preprocessing, feature selection, and attention mechanisms. TCNs combined with attention mechanisms and LASSO-based feature optimization have shown superior performance in terms of accuracy, computational efficiency, and scalability, making them the most promising direction for future HAR developments.

2.2 INFERENCE FROM THE SURVEY

The survey of existing Human Activity Recognition (HAR) systems highlights key challenges and opportunities for improvement in future models. Traditional machine learning methods, such as Support Vector Machines (SVM) and k-Nearest Neighbors (kNN), struggle with high-dimensional, sequential sensor data due to their reliance on manual feature engineering and their limited ability to capture temporal dependencies in time-series data. These approaches perform well in simpler settings but fail in real-time applications

where data complexity and noise are prevalent.

In contrast, deep learning approaches like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) offer significant improvements by automating feature extraction and learning complex sequential patterns. However, RNN-based models, such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), face limitations like the vanishing gradient problem and high training times, which hinder their real-time deployment for HAR tasks.

Temporal Convolutional Networks (TCNs) present an efficient alternative to RNNs, capturing long-range dependencies without the vanishing gradient issue. TCNs use dilated convolutions and causal structures to efficiently process sequential data, enabling better parallel computation and scalability. However, TCNs treat all time steps equally, which may result in suboptimal performance, as some time steps are more relevant than others for specific activities. This limitation can be addressed by integrating Attention Mechanisms, which dynamically assign importance to specific time steps, helping the model focus on critical activity segments. Despite the advantages, attention mechanisms are not consistently applied across existing HAR models, indicating an area for further research.

Feature selection remains an underexplored area in deep learning-based HAR models. High-dimensional sensor data often contains redundant and irrelevant features that degrade model performance. Techniques like LASSO Regression (Least Absolute Shrinkage and Selection Operator) can help optimize feature sets by reducing dimensionality, improving both model accuracy and computational efficiency.

Outlier detection is another area that requires attention. Sensor data is susceptible to noise from various sources, such as motion artifacts and environmental interference. Methods like Winsorization and Z-score are effective for outlier handling but are often absent in many HAR systems, despite their ability to ensure data quality and improve model performance.

Finally, model explainability remains critical, especially in healthcare and safety-critical applications. Explainable AI (XAI) techniques, such as SHAP (SHapley Additive exPlanations), provide valuable insights into how model predictions are made, ensuring transparency and trust, which is essential in real-world deployments.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 PROPOSED SYSTEM

The system architecture of the proposed Human Activity Recognition (HAR) framework is indicated in Fig. 3.1. The process begins with collecting the sensor data, followed by preprocessing, which includes detecting outliers, imputing them, encoding the data, and splitting it into train and test sets. LASSO Regression selects the most relevant features, which are then used for model compilation. The accuracy and loss metrics are used to optimize the performance of HAR by training and evaluating four models: TCN, TCN + AM, Bi-TCN, and Bi-TCN + AM.

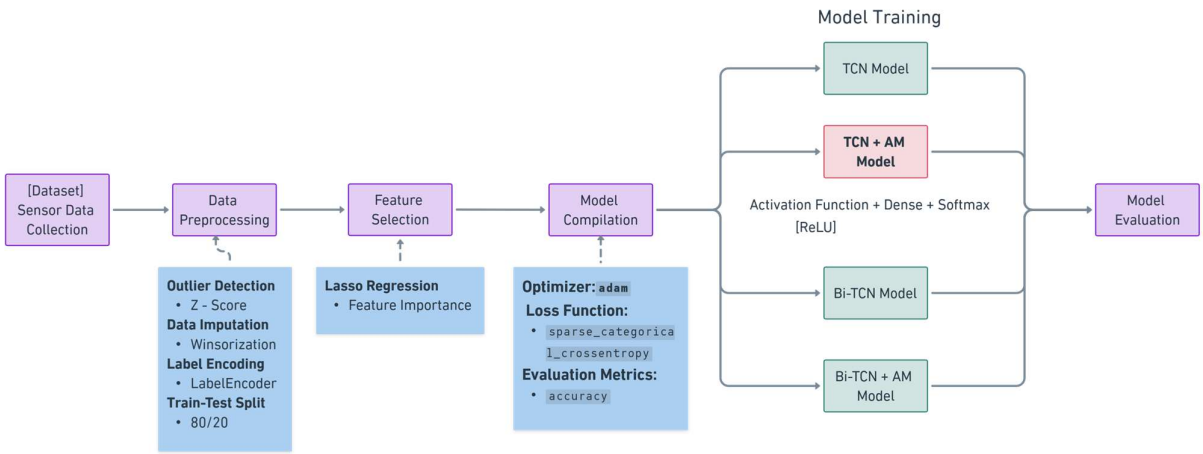


Fig 3.1. Proposed Architecture for Human Activity Recognition System

3.2 DATASET

The Human Activity Recognition Using Smartphones dataset is a widely used benchmark for evaluating machine learning and deep learning models in activity classification. It contains time-series sensor data collected from 30 volunteers aged 19 to 48 years, each performing six activities: walking, walking downstairs, walking upstairs, standing, sitting, and laying. The Samsung Galaxy S II smartphone has been used to record the data, placed at the waist, capturing motion data via its embedded accelerometer and gyroscope at 50Hz [28].

The dataset includes triaxial acceleration from the accelerometer and angular velocity from the gyroscope. A Butterworth low-pass filter has been applied to separate gravitational

and body motion components. The signals have been segmented into 2.56-second windows with 50% overlap, resulting in 561 time and frequency domain features. The dataset consists of 10,299 instances and has been split into 80% for training and 20% for testing, making it well-suited for deep learning-based human activity recognition applications. Fig 3.2 illustrates the distribution of the dataset, showcasing how data points are spread across different classes or features.

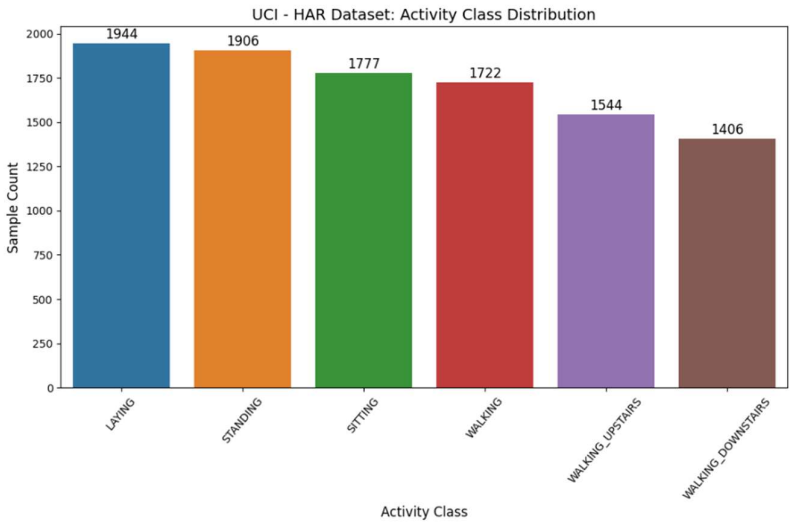


Fig 3.2. Dataset Class Distribution

3.3 TRAIN TEST SPLIT

Train-test splitting is a crucial step in machine learning to evaluate model performance on unseen data. The dataset is divided into 80% training data and 20% testing data, ensuring a balanced and effective evaluation. A stratified split is applied to maintain the original class distribution, preventing data imbalance that could affect the model's learning.

After feature selection using LASSO regression, the selected features are standardized using z-score normalization, ensuring consistency in data scaling. The training data is used to fit the model, while the test data helps assess its generalization ability. To make the data suitable for Temporal Convolutional Networks (TCN), it is reshaped into a format that allows efficient temporal pattern recognition. This approach prevents overfitting, ensuring that the model performs well on new, real-world data while maintaining high classification accuracy in Human Activity Recognition tasks.

3.4 PRE-PROCESSING DATA

Pre-processing ensures data consistency and enhances model performance. The raw sensor data undergoes normalization using Z-score standardization to maintain uniform scaling. Winsorization is applied for outlier handling, preserving essential patterns while reducing distortions. LASSO Regression is utilized for feature selection, retaining the most relevant 150 features. Finally, the dataset is segmented into fixed-size windows, ensuring temporal dependencies are maintained for effective classification in HAR tasks.

3.4.1 Outlier Detection

The boxplot in Fig. 3.3 visualizes the distribution of 20 features before outlier replacement. Anomalies, represented by extreme values outside the whiskers, are identified as potential outliers that could adversely affect the model's performance. The presence of these extreme values indicates noise in the data. This visualization helps in identifying the features with significant outliers, which will be handled during the data preprocessing stage to improve the reliability of the HAR model.

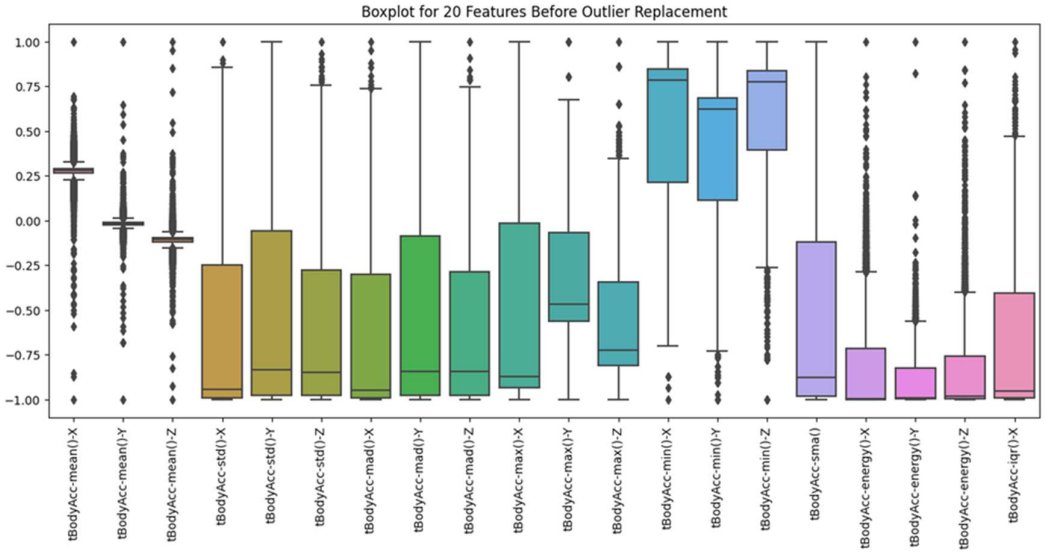


Fig 3.3. Representation of Outlier

3.4.2 Data Imputation

Outlier detection and handling are crucial for dataset quality. Various methods, such as removing extreme values or replacing them with statistical measures, can be applied to manage outliers. Winsorizing has been used to cap extreme values at specific percentiles, specifically

the 10th and 90th, eliminating outliers while preserving the main data pattern. This technique has been primarily applied to numerical features to maintain uniform distributions.[3], [17], [18], [19] Rather than deleting extreme values, Winsorization has capped them, preventing adverse effects on model training and improving model reliability. Fig 3.4 illustrates a boxplot after Winsorization.

The Winsorization formula can be expressed as:

$$X_{winsorized} = \begin{cases} X_{min} & if\ X < X_{min} \\ X & if\ X_{min} \leq X \leq X_{max} \\ X_{max} & if\ X > X_{max} \end{cases} \tag{3.1}$$

Explanation:

- X → Original data point
- X_{min} → Lower percentile threshold (e.g., 5th percentile for 10% Winsorization)
- X_{max} → Upper percentile threshold (e.g., 95th percentile for 10% Winsorization)

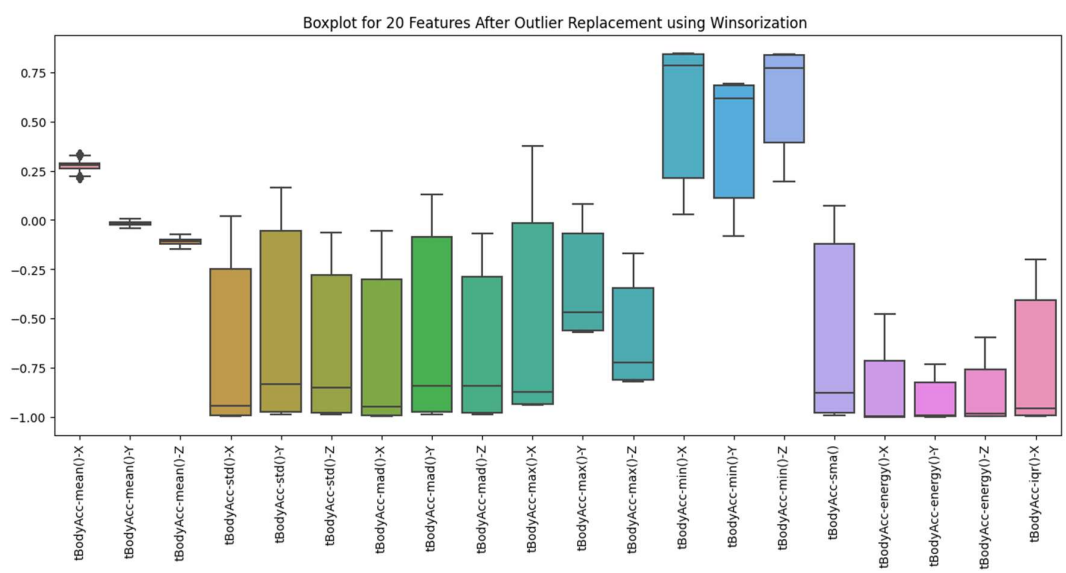


Fig 3.4. Winsorization for Outlier Replacement

3.5 FEATURE SELECTION

Feature selection is a critical step in Human Activity Recognition (HAR) to enhance model accuracy and efficiency. Sensor data in HAR tasks often include redundant, irrelevant, or noisy features, which can increase model complexity and reduce computational efficiency. By selecting only the most relevant features, the model becomes more focused, improves

generalization, and reduces the risk of overfitting [18] [19].

Techniques like LASSO Regression (Least Absolute Shrinkage and Selection Operator) are commonly employed for feature selection. LASSO uses L1 regularization to shrink the coefficients of less important features to zero, effectively removing them from the model. This process helps reduce dimensionality while maintaining critical information. By optimizing the feature set, LASSO enhances model performance, particularly in real-time HAR applications where speed and accuracy are essential. This ensures that the model remains computationally efficient without sacrificing classification performance, making it suitable for real-time deployment in areas like healthcare and smart environments.

3.5.1 LASSO Regression

LASSO (Least Absolute Shrinkage and Selection Operator) regression is a linear regression technique that incorporates an L1 regularization term to the loss function, enabling feature selection by forcing some coefficients to shrink to zero. This results in the elimination of irrelevant or redundant variables, thus simplifying the model. The L1 penalty in LASSO adds a constraint to the optimization process, where the sum of the absolute values of the coefficients is minimized, promoting sparsity in the model. By applying LASSO, only the most influential features are retained, which helps in reducing overfitting, especially in high-dimensional datasets. The regularization parameter, λ, controls the strength of the penalty; larger λ values increase shrinkage, leading to more coefficients being zeroed out. Fig 3.5 represents the top 20 of the selected features by using LASSO Regression.

The LASSO regression cost function is:

$$Loss = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{3.2}$$

Where:

- y_i = actual values
- $\hat{y}_i$ = predicted values
- β_j = regression coefficients
- λ = regularization parameter that controls shrinkage
- n = number of observations
- p = number of features

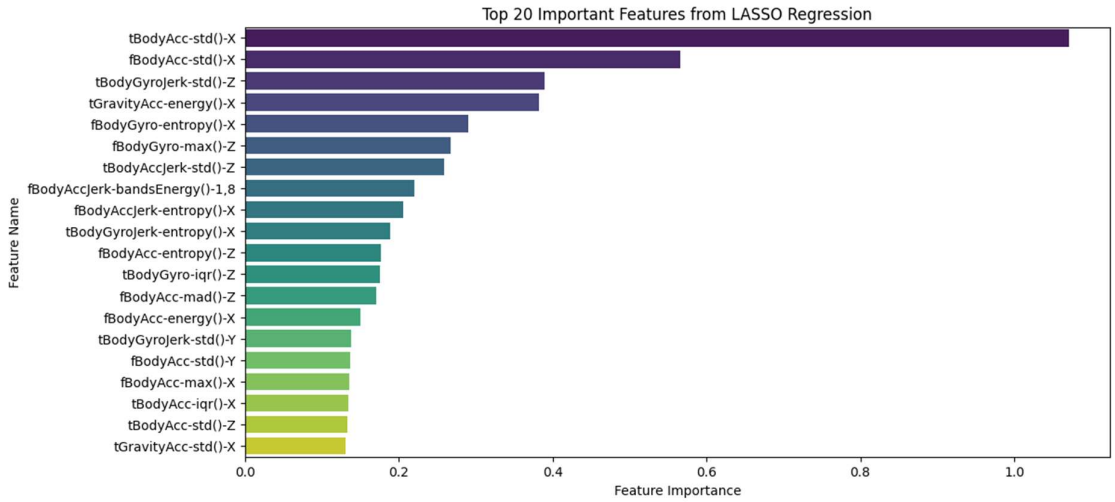


Fig 3.5. Importance Feature after LASSO Regression

3.6 TCN MODEL DESCRIPTION

Temporal Convolutional Networks (TCNs) are a powerful and efficient model for processing sequential data in machine learning, offering a notable improvement over traditional Recurrent Neural Networks (RNNs). While RNNs process data sequentially, which can lead to issues like vanishing gradients, TCNs tackle these challenges by utilizing one-dimensional dilated causal convolutions. Dilated convolutions allow TCNs to capture long-range dependencies in the data, enabling them to process sequences efficiently without losing important temporal information.

The core advantage of TCNs lies in their ability to parallelize computations, unlike RNNs, which require sequential data processing. This parallelism makes TCNs much faster and more scalable, especially when dealing with large datasets. The use of causal convolutions in TCNs ensures that the model does not leak future data into the past, preserving the temporal structure and maintaining the integrity of the sequence during learning.

Because TCNs excel at capturing long-term dependencies and their ability to work efficiently with time-series data, they have found significant applications in various domains. For instance, TCNs are widely used in speech recognition systems, where they process audio signals sequentially, and in financial forecasting, where they model time-based trends and stock price movements. In the domain of Human Activity Recognition (HAR), TCNs are highly effective at analyzing sensor data collected over time, such as data from wearable devices, enabling accurate and real-time activity classification.

By integrating the power of convolutional layers and the principles of sequential data processing, TCNs provide a robust framework for time-series analysis, making them a preferred choice in numerous industries that require efficient and reliable temporal data modeling.

3.6.1 Temporal Convolutional Network (TCN)

Temporal Convolutional Network (TCN) as depicted in Fig. 3.6 is a deep learning model designed for time-series data, using one-dimensional convolutions instead of recurrent connections to capture long-term dependencies efficiently. It consists of convolutional layers for feature extraction, dilated convolutions for expanding the receptive field, ReLU for stable training, and a Softmax output layer for classification.

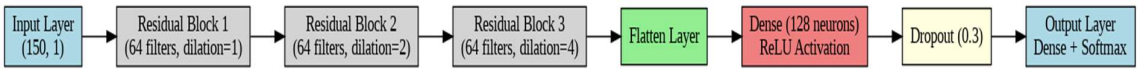


Fig 3.6. TCN Model Block Diagram

TCN supports parallel computation, making training faster than recurrent networks while effectively handling sequential sensor data. Its ability to model long-term dependencies without vanishing gradients makes it ideal for human activity recognition tasks requiring real-time efficiency and accuracy.

3.6.2 Activation Function: ReLU

Activation functions are essential in deep learning models as they introduce non-linearity, enabling the network to capture complex patterns. ReLU has been compared with Leaky ReLU and Swish, and it has been found to be the most effective activation function for TCN-based models. Its ability to enhance training speed and gradient flow ensures stable learning. Due to its efficiency in handling sequential dependencies, ReLU is particularly well-suited for HAR applications, where accurate time-series classification is essential [16].

The Rectified Linear Unit (ReLU) activation function, as shown in Fig. 3.7, is defined mathematically in eq (3.3) as:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{3.3}$$

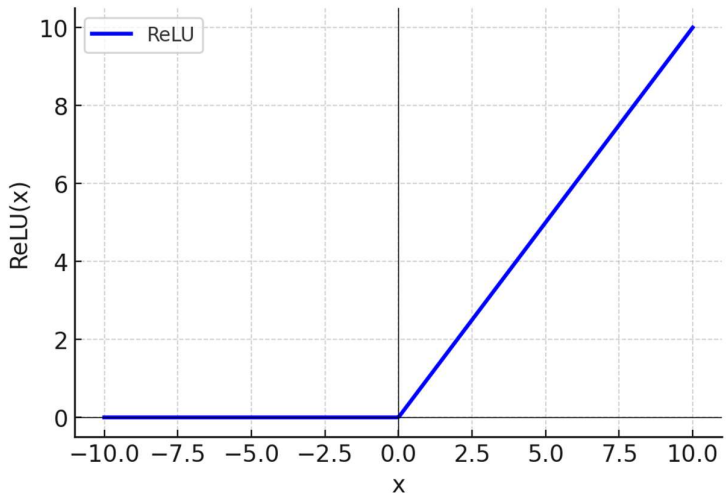


Fig 3.7. Activation Function – ReLU

ReLU's simplicity and efficiency make it an ideal activation function for Human Activity Recognition (HAR) tasks. In HAR, where sequential sensor data must be processed efficiently, ReLU enables the model to learn complex temporal dependencies without introducing unnecessary computational overhead. Its non-linearity allows the network to capture intricate patterns in human activity data, while its ability to address the vanishing gradient problem ensures that gradients do not diminish as they propagate through the layers.

This results in more stable training, which is essential for real-time applications. Additionally, ReLU's ability to allow faster training speeds, especially with deep architectures, enhances the model's efficiency, making it highly suitable for large-scale HAR datasets with high-dimensional features. Its robustness and efficiency in handling noisy or sparse data contribute significantly to improving classification accuracy and real-time response in HAR systems.

3.6.3 Attention Mechanism (AM)

As illustrated in Fig. 3.8, The Attention Mechanism (AM) improves deep learning models by allowing them to focus on the most relevant features in the input data. In the context of Human Activity Recognition (HAR), it enhances temporal dependencies, leading to improved classification accuracy. The process begins with Global Average Pooling (GAP), which reduces the feature map's dimensionality by averaging the features over time. This is followed by a Fully-Connected Layer with ReLU activation, which captures non-linear relationships in the data. A second Fully-Connected Layer with Sigmoid activation is then

applied to generate refined weights for each feature. These weights are multiplied element-wise with the feature map, highlighting important features while suppressing less relevant ones. By dynamically adjusting feature importance, the attention mechanism allows the model to focus on critical time steps, improving its ability to distinguish between complex human activities. This enhances both model performance and interpretability in HAR tasks.

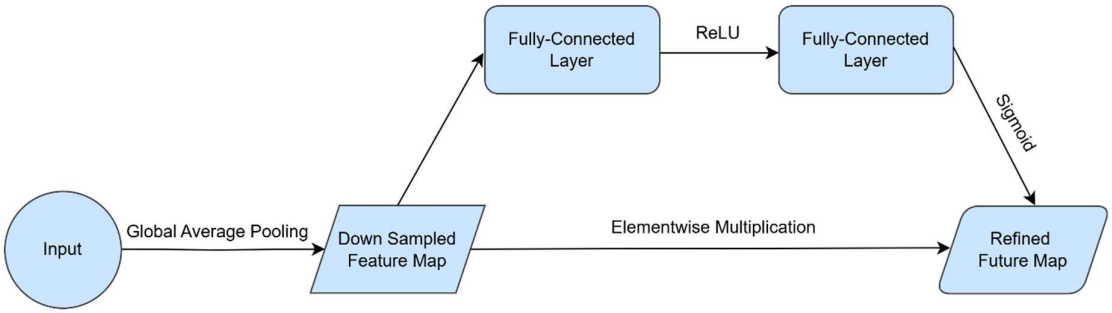


Fig 3.8. Attention Mechanism (AM)

3.6.4 TCN + Attention Mechanism (TCN + AM)

The TCN + Attention Mechanism (TCN + AM) model, shown in Fig. 3.9, combines the strengths of Temporal Convolutional Networks (TCN) and Attention Mechanisms to enhance performance in Human Activity Recognition (HAR). TCNs efficiently process time-series data by using dilated convolutions to capture long-range dependencies, essential for identifying complex human activity patterns.

Fig 3.9. TCN + AM Model Block Diagram

The Attention Mechanism enables the model to focus on critical time steps, prioritizing relevant features. By dynamically adjusting feature importance, it enhances feature representation, improves classification accuracy, and helps distinguish between similar activities while suppressing irrelevant data.

Achieving 98.74% accuracy with 150 features, the TCN + AM model demonstrates improved robustness and reliability, especially in handling noisy data. This hybrid approach is highly effective for real-time HAR applications, offering a superior solution for accurate and efficient human activity classification.

3.6.5 Bi-TCN Bidirectional TCN (Bi-TCN)

Bi-TCN (Bidirectional Temporal Convolutional Network), as shown in Fig. 3.10, is an extension of the standard Temporal Convolutional Network (TCN) designed to enhance temporal modeling by processing data in both forward and backward directions. This bidirectional processing allows the model to capture both past and future dependencies in sequential data, improving its ability to understand and predict complex activity patterns, especially in dynamic settings like human activity recognition (HAR).

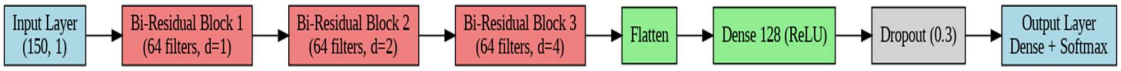


Fig 3.10. Bi-TCN Model Block Diagram

The use of dilated convolutions further strengthens the model's capability to maintain long-term dependencies within the data. This ensures that even distant time steps within the sequence can influence predictions, making Bi-TCN especially suitable for analyzing time-series data where long-range relationships are crucial.

By incorporating bidirectional processing, Bi-TCN enhances the model's understanding of complex human activities, such as recognizing overlapping or simultaneous motions. This added context allows Bi-TCN to outperform traditional TCNs in identifying intricate movement patterns, making it a more comprehensive and effective choice for HAR applications.

3.6.6 Bidirectional TCN + Attention Mechanism (Bi-TCN + AM)

Bidirectional TCN + Attention Mechanism (Bi-TCN + AM), as shown in Fig. 3.11, integrates the advantages of Bidirectional Temporal Convolutional Networks (Bi-TCN) and the Attention Mechanism (AM) to enhance the model's performance in Human Activity Recognition (HAR). The Bidirectional TCN component captures both past and future dependencies by processing the data in both forward and backward directions. This capability allows the model to consider the entire temporal context when classifying activities, making it more robust for recognizing complex and overlapping human movements.

By adding the Attention Mechanism, the model can dynamically adjust the importance of different time steps, focusing more on relevant features and reducing the influence of noisy

or less informative data. The attention mechanism emphasizes key motion patterns, allowing the model to differentiate between similar activities, which is crucial for accurate classification in HAR tasks.

Fig 3.11. Bi-TCN + AM Model Block Diagram

With 98.45% accuracy, the Bi-TCN + AM model excels in precision, recall, and specificity, making it highly effective for real-time HAR applications. The combination of bi-directional processing and attention ensures that the model not only captures long-range dependencies but also prioritizes the most important features for improved activity recognition. This makes it particularly suitable for applications in healthcare, fitness tracking, smart homes, and security, where high accuracy and efficiency are essential.

3.7 EXPLAINABLE AI

Explainable Artificial Intelligence (XAI) is crucial for enhancing the transparency and interpretability of machine learning models, particularly in Human Activity Recognition (HAR), where decisions impact human lives. In HAR, deep learning models classify activities based on sensor data, and understanding how these predictions are made is vital. XAI techniques help researchers and practitioners interpret the decision-making process, fostering trust and transparency. This is especially important in applications like healthcare and fitness tracking, where wrong decisions can affect well-being. XAI reduces bias, increases trust, and improves model accountability, ensuring ethical and reliable outcomes in critical areas.

One of the key benefits of XAI is that it allows stakeholders, such as healthcare providers, patients, and end-users, to understand the reasoning behind model decisions. In healthcare, for example, knowing how a model arrived at a diagnosis or treatment recommendation is vital for adoption and patient safety. When users can interpret these decisions, they are more likely to trust and effectively use the system. This transparency also ensures alignment with ethical guidelines, as healthcare decisions can significantly impact patient care, reinforcing the importance of ethical and reliable model outputs.

Another key advantage of XAI is its ability to uncover biases in the model. By examining how features influence the model's predictions, XAI techniques can reveal hidden

biases that might affect certain groups of individuals, such as gender, age, or race. Identifying these biases allows researchers to correct them, ensuring that the model's predictions are fair and unbiased. Moreover, XAI helps enhance model performance by enabling debugging and error correction. By understanding which features are most important for predictions, researchers can improve feature selection, tune model parameters, and correct errors in the model's behavior, thus increasing its reliability and accuracy.

3.7.1 SHAP

SHapley Additive exPlanations (SHAP) is a powerful method within XAI that offers a game-theoretic approach to feature attribution. It assigns an importance value to each feature based on its contribution to the model's prediction. In the context of HAR, SHAP identifies which sensor readings—such as accelerometer and gyroscope data—have the most significant impact on classifying human activities. This process makes the model's decision-making more interpretable, enhancing its transparency.

SHAP provides several visualization techniques to illustrate feature importance. Feature importance plots are commonly used to highlight the most influential features in activity classification, making it easy to identify which data points drive predictions. Dependency plots, on the other hand, show how the value of a feature affects the model's output, offering insight into the relationship between feature values and predicted outcomes. The waterfall plot, another visualization provided by SHAP, illustrates how each feature contributes positively or negatively to the final prediction, offering a more granular view of the decision-making process.

Integrating SHAP into HAR models enhances both the interpretability and trustworthiness of the system. It allows researchers to better understand the contributions of different sensor data, ensuring that the model's predictions align with real-world expectations. This transparency is especially valuable in sensitive areas like healthcare, where the model's decisions can directly impact patient care. Additionally, SHAP's ability to break down the model's decision-making process makes it easier to identify areas for improvement, ensuring that the system is continuously refined and optimized.

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSIONS

This chapter provides an in-depth analysis of the experimental evaluation of the proposed Human Activity Recognition (HAR) system. It discusses the implementation methodology, the various performance metrics used for evaluating the models, and the results obtained. The chapter also highlights the importance of accuracy, precision, recall, F1-score, specificity, ROC-AUC, training time, and inference time, all of which serve as key indicators of model effectiveness and performance.

4.1 IMPLEMENTATION

Step 1: Dataset Acquisition

The implementation of the HAR system begins with the acquisition of the UCI HAR dataset, widely regarded as one of the benchmark datasets for evaluating human activity recognition models. This dataset contains sensor data recorded from accelerometers and gyroscopes embedded in smartphones. The data is collected while 30 individuals perform six different human activities: walking, walking downstairs, walking upstairs, standing, sitting, and laying. The data acquisition process ensures that a diverse range of activities is captured, which is crucial for the robustness of the model. Each instance in the dataset corresponds to a time-series sequence of sensor readings at a 50Hz sampling rate.

Step 2: Data Preprocessing

Once the raw sensor data is acquired, it undergoes a series of preprocessing steps to ensure that the data is ready for analysis and model training. One of the first tasks is to detect and handle outliers, which could otherwise adversely affect model performance. Winsorization, a statistical method, is employed to cap extreme values at predefined percentiles, ensuring that outliers are managed effectively without discarding valuable data points. Additionally, feature scaling is applied to normalize the data, making sure all features have the same scale and improving model performance during training.

Step 3: Feature Selection Using LASSO Regression

Feature selection is an essential step in improving the efficiency of the model. Given

that sensor data is often high-dimensional and contains many irrelevant or redundant features, it is critical to select the most relevant features for training the model. LASSO (Least Absolute Shrinkage and Selection Operator) regression is applied to perform feature selection. By using LASSO, the top 150 most relevant features are selected from the original 561, significantly reducing the dimensionality of the data while preserving essential information related to human activity recognition. This reduction in feature space not only accelerates training but also improves the model's generalization ability.

Step 4: Train-Test Split

To evaluate the performance of the HAR models, the dataset is split into training and testing subsets. A stratified sampling technique is applied to ensure that the distribution of activity classes is maintained across both the training and testing sets. This step guarantees that both subsets contain representative samples of each activity, preventing class imbalances that could skew the evaluation process. 80% of the data is used for training, while the remaining 20% is reserved for testing, ensuring that the model is evaluated on unseen data to assess its generalization ability.

Step 5: Time-Series Models Selection

In this step, different time-series models are selected to evaluate their suitability for the HAR task. Models such as Temporal Convolutional Networks (TCN), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN), and Gated Recurrent Units (GRU) are considered due to their ability to handle sequential data. The selection of these models is driven by the need to identify the most effective architecture for accurately classifying human activities based on time-series sensor data. The goal is to compare the performance of these models in terms of accuracy, computational efficiency, and real-time deployment capabilities.

Step 6: Attention Mechanism Integration

To further improve the performance of the selected time-series models, an Attention Mechanism (AM) is integrated into both the TCN and Bi-TCN models. The attention mechanism dynamically assigns weights to different time steps, allowing the model to focus on the most relevant parts of the input data while suppressing less important information. This mechanism enhances the model's ability to capture complex patterns and dependencies within the time-series data, improving classification accuracy, especially for activities that involve

subtle changes in motion.

Step 7: Model Development

Four different deep learning models are developed for evaluating the effectiveness of various configurations. The models include TCN, TCN with Attention Mechanism (TCN + AM), Bidirectional TCN (Bi-TCN), and Bidirectional TCN with Attention Mechanism (Bi-TCN + AM). Each model utilizes dilated causal convolutions to capture long-term dependencies within the time-series data, which is crucial for identifying human activities accurately over time. The models are designed to handle large-scale time-series data while ensuring efficient computation, making them suitable for real-time applications in wearable devices and smart environments.

Step 8: Model Training and Optimization

The training of the models is carried out using the Adam optimizer with a learning rate of 0.001. Sparse categorical cross-entropy is used as the loss function to optimize the classification performance. Batch normalization and dropout layers are applied to prevent overfitting and ensure stable learning. During the training process, the models are continuously adjusted to minimize the loss and improve the classification accuracy. Hyperparameter tuning is performed to find the best configurations for each model.

Step 9: Performance Evaluation

Each model is evaluated using several performance metrics, including accuracy, precision, recall, F1-score, specificity, ROC-AUC, training time, and inference time. These metrics provide a comprehensive evaluation of how well the models classify human activities and how efficient they are in terms of computational resources. The evaluation is carried out using the test dataset, and the results are used to compare the performance of different model variants.

Step 10: Comparative Analysis

All model variations are compared based on their performance across various evaluation metrics. This analysis identifies the optimal configuration for the HAR system, highlighting the best trade-off between accuracy, stability, and computational efficiency. The TCN + AM model with 150 features is found to provide the best balance and demonstrates

superior performance in real-time HAR applications.

Step 11: Explainability with SHAP

To increase the interpretability of the model, SHAP (SHapley Additive exPlanations) is applied to explain the decision-making process of the HAR system. SHAP assigns importance values to each feature, showing how much each feature contributes to the final prediction. This interpretability is essential in sensitive applications like healthcare, where understanding the reasoning behind the model's decisions is crucial. SHAP helps enhance the transparency of the HAR system, making it more reliable and trustworthy for real-world deployment.

4.2 PERFORMANCE METRICS

The model's performance has been assessed using standard machine learning evaluation metrics, including accuracy, precision, recall, F1-score, specificity, ROC-AUC, training time, and inference time. These metrics help measure the effectiveness of the HAR system in accurately classifying human activities.

4.2.1 Accuracy

Accuracy represents the ratio of correctly classified activities to the total number of activities, ensuring a reliable performance measure for the model. It is computed using eq (4.1)

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

where:

- **TP** (True Positives): Correctly identified activity instances.
- **TN** (True Negatives): Correctly rejected non-activity instances.
- **FP** (False Positives): Incorrectly identified activity instances.
- **FN** (False Negatives): Missed activity instances.

A high accuracy score indicates that the model is proficient in distinguishing between different activities. However, while accuracy provides a general measure of performance, it is essential to consider other metrics like precision and recall, especially in imbalanced datasets

where high accuracy could mask the model's true performance.

4.2.2 Precision

Precision measures the proportion of correctly predicted positive observations to total predicted positive observations, as defined in eq (4.2)

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

A high precision value indicates that the model is accurate in identifying positive instances and minimizes false positives. In Human Activity Recognition (HAR), this is crucial to ensure that the model only recognizes actual activities and avoids mistakenly classifying non-activity instances as activities. High precision ensures that the model's positive activity detections are reliable, making it ideal for applications like healthcare or fitness tracking, where false positives could lead to unnecessary interventions or misinterpretations of a person's state.

4.2.3 Recall

Recall, also known as sensitivity, measures the proportion of true positive instances that are correctly identified by the model. It is defined by the formula in Equation (4.3).

$$\text{Recall} = \frac{TP}{TP+F} \tag{4.3}$$

A high recall score indicates that the model is effective at detecting the majority of true activities. In Human Activity Recognition (HAR), this means the model is proficient in identifying all relevant activity patterns, minimizing false negatives. A higher recall value is particularly important in applications like healthcare and security, where failing to identify an activity (such as detecting a fall or emergency) can have significant consequences. Balancing recall with precision is essential to ensure the model accurately detects activities while avoiding excessive false alarms.

4.2.4 F1-Score

The F1-score is the harmonic mean of precision and recall, balancing both metrics, as defined in eq (4.4)

$$F1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

A high F1-score indicates that the model maintains a good balance between precision and recall, effectively managing the trade-off between the two. This is particularly important in applications like healthcare, where both false positives (misclassifying non-activity as activity) and false negatives (failing to detect an activity) can have significant consequences. Thus, a high F1-score ensures that the model does not overly favor precision at the cost of recall, or vice versa, making it a reliable indicator of overall classification performance.

4.2.5 Specificity

Specificity quantifies the ability of the model to correctly identify negative instances, as defined in eq (4.5)

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{4.5}$$

Specificity measures the model's ability to correctly identify negative instances, ensuring that non-activity data is accurately classified as such. In the context of our Human Activity Recognition (HAR) project, high specificity is crucial to minimize false alarms, ensuring that the model does not mistakenly identify non-activity situations as activities. This is particularly important in real-time applications like healthcare and smart environments, where false positive detections could lead to unnecessary interventions or alerts, compromising system reliability and efficiency.

4.2.6 ROC-AUC

The Receiver Operating Characteristic - Area Under the Curve (ROC-AUC) assesses the model’s ability to distinguish between different activity classes. A higher AUC value, approaching 1, signifies superior classification performance. The True Positive Rate (TPR) is defined in eq (4.6), while the False Positive Rate (FPR) is given in eq (4.7), both contributing to the ROC curve's evaluation of model effectiveness.

$$\text{TPR} = \frac{TP}{TP + FN} \tag{4.6}$$

$$\text{FPR} = \frac{FP}{FP + TN} \tag{4.7}$$

4.2.7 Training Time

Training time represents the total duration taken to train the HAR model on the dataset. It depends on, as defined in eq (4.8)

$$Training\ Time = Model\ Training\ End\ Time - Model\ Training\ Start\ Time \quad (4.8)$$

Model complexity, dataset size, and hardware resources impact training time. Complex models and larger datasets require more computation, increasing training duration. GPUs, with their parallel processing capabilities, accelerate training significantly compared to CPUs, reducing overall time. Efficient hardware utilization helps optimize the training process in deep learning models.

4.2.8 Inference Time

Inference time refers to the duration it takes for a trained model to generate a prediction for new input data. It is the time from when the input is fed into the model until the output is produced, as outlined in Equation (4.9).

$$Inference\ Time = Model\ Inference\ End\ Time - Model\ Inference\ Start\ Time \quad (4.9)$$

Minimizing inference time is vital for real-time applications, ensuring fast predictions. In healthcare, it enables quick responses to patient conditions. In smart homes, it supports automation like adjusting temperature. In industrial automation, low inference time aids in anomaly detection and predictive maintenance, enhancing system responsiveness and reliability.

In conclusion, the performance metrics—accuracy, precision, recall, F1-score, specificity, ROC-AUC, training time, and inference time—offer a comprehensive evaluation of the Human Activity Recognition (HAR) system. High accuracy reflects overall performance, while precision, recall, and F1-score balance false positives and false negatives, ensuring reliability. Specificity minimizes false alarms, enhancing efficiency in critical applications like healthcare and security. ROC-AUC gauges the model's ability to distinguish activity classes, while training and inference times are crucial for real-time applications. Optimizing these metrics ensures the HAR system is accurate, efficient, and suitable for time-sensitive environments, making it reliable for diverse, real-world use cases.

4.3 RESULTS AND DISCUSSIONS

The experimental evaluation of various time-series models in Human Activity Recognition (HAR) reveals that different model architectures, particularly variations of Temporal Convolutional Networks (TCN), show promising results. Table 4.1 presents the performance of several models with different numbers of features, highlighting the impact of feature selection on model accuracy.

Table 4.1: Performance of Time Series Models with Different Number of Features

Model Variation	**ACCURACY %**			
	256 Features	**150 Features**	**100 Features**	**50 Features**
TCN	**97.96**	**98.06**	**97.48**	**95.53**
LSTM	67.28	88.30	86.65	86.50
GRU	77.86	87.18	90.78	90.83
RNN	63.64	80.68	92.72	94.27

The results demonstrate that TCN consistently outperforms traditional recurrent models such as LSTM, GRU, and RNN across all feature sets. Specifically, at 150 features, TCN achieves an accuracy of 98.06%, significantly surpassing LSTM (88.30%), GRU (87.18%), and RNN (80.68%). This indicates that TCN is more efficient in capturing temporal patterns and managing high-dimensional data, which is crucial for HAR tasks.

LSTM, GRU, and RNN models face challenges with lower-dimensional feature sets, where they show performance degradation. In contrast, TCN maintains high accuracy, even with fewer features, due to its ability to capture long-term dependencies in time-series data. This reinforces the advantages of using TCN over traditional recurrent models, particularly for HAR applications where high-dimensional, real-time sensor data needs to be processed efficiently.

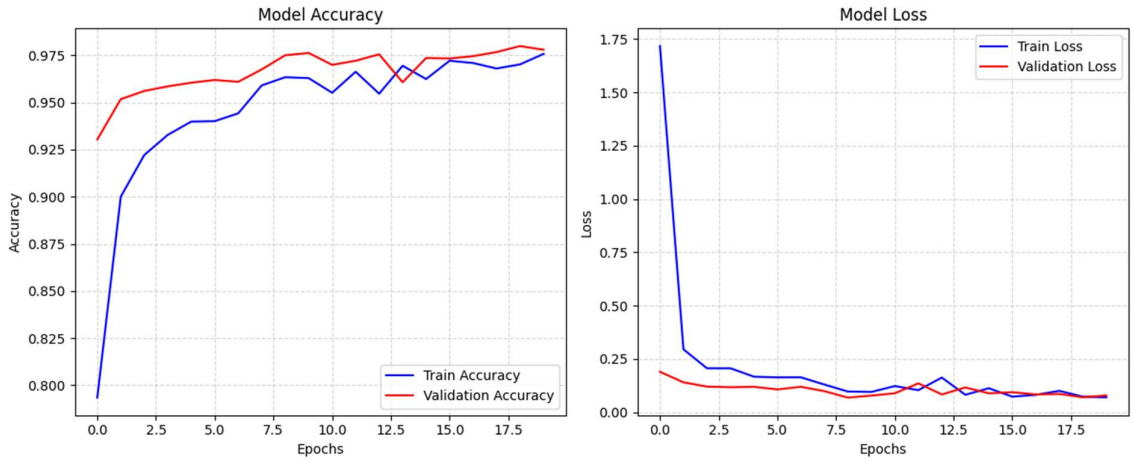


Fig 4.1. Accuracy & Loss Graph for TCN + AM with 256 Features

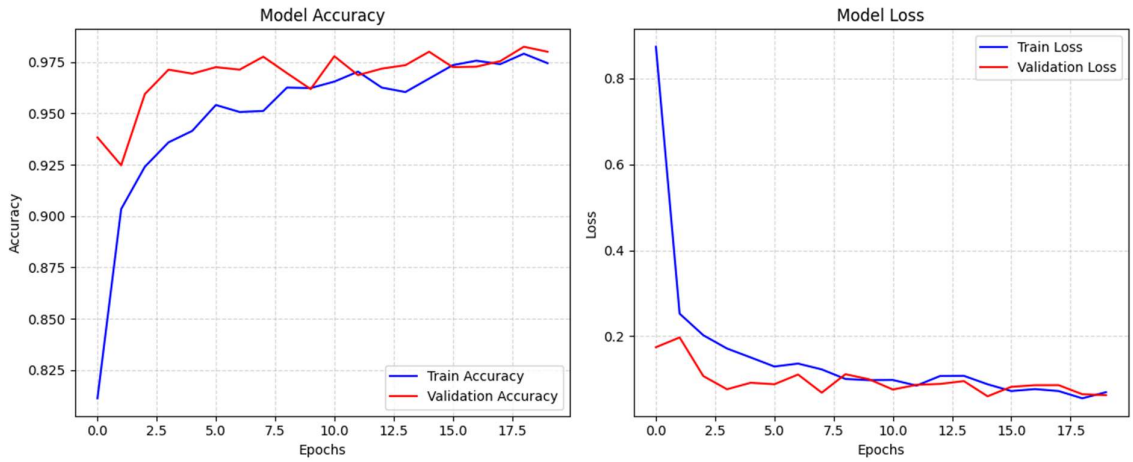


Fig 4.2. Accuracy & Loss Graph for TCN + AM with 150 Features

The accuracy and loss graphs for TCN + AM with 256 and 150 features (Fig 4.1 and Fig 4.2) offer valuable insights into the impact of feature selection on the model's performance. In Fig. 4.1, the model with 256 features achieves a high accuracy of 98.20%, but the validation accuracy fluctuates significantly, suggesting some degree of overfitting. Overfitting occurs when the model learns the training data too well, capturing noise and irrelevant patterns, which negatively affects its performance on unseen data.

In contrast, Fig. 4.2, with 150 features, shows a higher accuracy of 98.74%, with a more stable and smooth learning curve. This indicates better generalization and reduced overfitting. The model's performance remains consistent over time, which is desirable for real-time applications like Human Activity Recognition (HAR), where model adaptability and reliability are crucial. Additionally, the loss graph for the 256-feature model shows greater variations in validation loss, which is a clear sign of memorization rather than learning general patterns. On

the other hand, the 150-feature model maintains lower and more stable loss values throughout the training process.

By reducing the number of features from 256 to 150, the model becomes more efficient and stable, without sacrificing its accuracy. This emphasizes the importance of feature selection in optimizing model performance, improving both computational efficiency and generalization, making the 150-feature model the superior choice for HAR tasks.

Table 4.2: Performance of TCN Variants with Different Number of Features

Model Variation	**ACCURACY %**			
	256 Features	**150 Features**	**100 Features**	**50 Features**
TCN	97.96	98.06	97.48	95.53
TCN + AM	98.20	**98.74**	97.96	95.78
Bi-TCN	98.25	97.14	97.52	95.63
Bi-TCN + AM	98.45	98.40	97.62	96.26
LSTM	67.28	88.30	86.65	86.50
GRU	77.86	87.18	90.78	90.83
RNN	63.64	80.68	92.72	94.27

Table 4.2 provides a comprehensive comparison of various Temporal Convolutional Network (TCN) models with different numbers of features, highlighting their performance across multiple configurations. The table shows that the TCN + Attention Mechanism (TCN + AM) model achieves the highest accuracy of 98.74% when using 150 features, significantly outperforming both the standard TCN model (98.06%) and the Bi-TCN model (97.14%). This demonstrates the notable improvement gained from integrating the attention mechanism, which allows the model to focus on the most relevant time steps in the data, thereby improving its ability to capture complex human activity patterns.

Moreover, the TCN + AM model continues to excel across various feature configurations, achieving a consistent and stable performance. It not only leads in accuracy but also offers better generalization, as indicated by its performance with fewer features (150

versus 256 features). The traditional deep learning models, such as LSTM, GRU, and RNN, while effective, do not perform as well as TCN-based models. For example, LSTM achieves 88.30% accuracy with 150 features, which is significantly lower than TCN + AM. The use of fewer features in TCN + AM also highlights the importance of feature selection in improving both accuracy and computational efficiency.

In conclusion, the TCN + AM model with 150 features provides the best combination of performance, stability, and efficiency, making it the optimal choice for Human Activity Recognition tasks. It demonstrates that feature reduction, combined with attention mechanisms, significantly enhances the model's ability to recognize human activities accurately while maintaining efficiency in real-time applications.

The bar chart in Fig. 4.3 presents the accuracy performance of various deep learning models when applied to Human Activity Recognition (HAR) tasks using 150 selected features. Among the models evaluated, the TCN + Attention Mechanism (TCN + AM) stands out with the highest accuracy of 98.74%, closely followed by the Bi-TCN + AM model, which achieves 98.40%. Both of these models demonstrate the effectiveness of integrating the attention mechanism to enhance feature extraction and improve the model’s ability to focus on the most relevant time steps, thereby increasing overall classification accuracy.

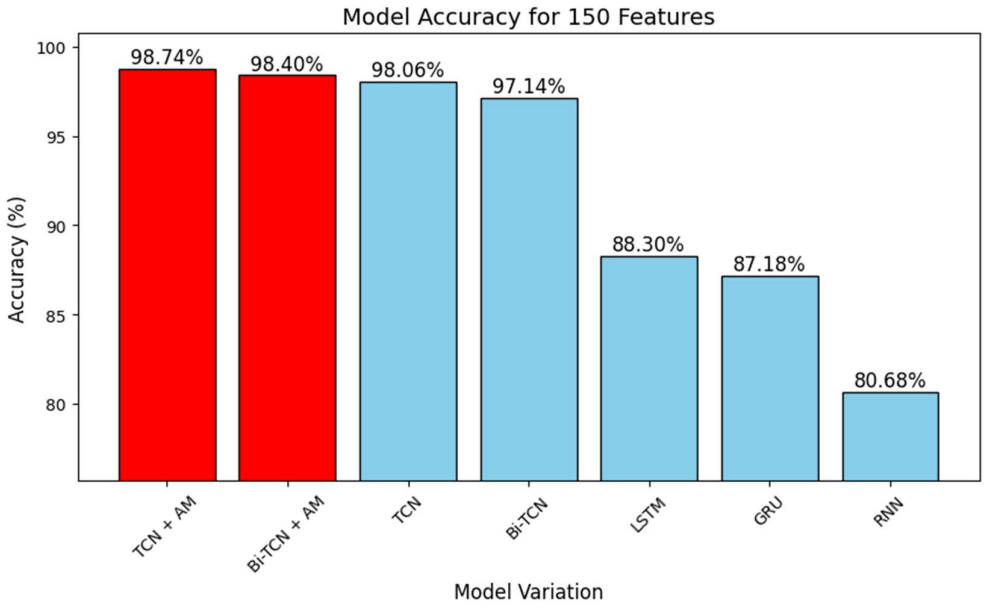


Fig 4.3. Model Accuracy Chart for all Models with 150 Features

The standard TCN model also shows excellent performance with an accuracy of

98.06%, outperforming the Bi-TCN model, which achieves 97.14%. This suggests that TCNs, which capture sequential dependencies effectively, offer a stronger framework for HAR tasks than the bidirectional version. The slight drop in performance for Bi-TCN could indicate that bidirectional processing does not necessarily lead to better outcomes in this context.

On the other hand, traditional recurrent models such as LSTM, GRU, and RNN show relatively lower accuracy, with values of 88.30%, 87.18%, and 80.68%, respectively. These models, while capable of capturing temporal patterns, face challenges like slower convergence and vanishing gradients, which impact their efficiency in HAR tasks. The results from the chart underscore the superior performance of TCN-based models, particularly those enhanced with attention mechanisms, in achieving higher accuracy and efficiency for real-time HAR applications.

Table 4.3: Performance Comparison of TCN-Based Models with 150 Features

Model Variation	Accuracy	Precision	Recall	Specificity	F1-Score	ROC AUC	Training Time (s)	Inference Time (s)
TCN	98.06	98.10	98.06	98.07	98.06	99.94	32.95	1.52
TCN + AM	**98.74**	98.74	98.74	98.80	98.74	99.96	41.15	2.06
Bi-TCN	97.14	97.15	97.14	97.28	97.13	99.89	38.83	1.55
Bi-TCN + AM	98.40	98.40	98.40	98.46	98.40	99.95	44.59	1.98

Table 4.3 compares the performance of various Temporal Convolutional Network (TCN)-based models using 150 features, including the standard TCN, TCN with Attention Mechanism (TCN + AM), Bi-TCN, and Bi-TCN with Attention Mechanism (Bi-TCN + AM). The table highlights the effectiveness of integrating Attention Mechanisms (AM) into the TCN architecture, demonstrating improvements across several performance metrics, such as accuracy, precision, recall, specificity, F1-score, and ROC AUC.

The TCN + AM model achieves the highest accuracy of 98.74%, outperforming the standard TCN model by 0.69%. This performance gain is attributed to the Attention Mechanism's ability to focus on important time steps in the data, enhancing feature extraction and optimizing sequence learning. The attention mechanism dynamically assigns weights to the most relevant time steps, thereby improving

the model's ability to capture critical activity patterns in human activity recognition (HAR). This result underscores the importance of incorporating attention mechanisms in deep learning models for HAR, where precision and recall are essential.

When comparing TCN + AM with Bi-TCN, the TCN + AM model demonstrates a significant 1.60% higher accuracy. While Bi-TCN captures temporal dependencies in both forward and backward directions, the TCN + AM model, which combines attention-based feature weighting, proves to be more efficient. Bidirectional processing, though useful in some contexts, does not always guarantee better performance, and in this case, TCN + AM achieves superior accuracy by focusing on the most relevant features. This suggests that attention mechanisms provide a more effective approach to improving model performance than bidirectional processing alone.

Additionally, when compared to Bi-TCN + AM, the TCN + AM model still achieves a 0.34% higher accuracy, further demonstrating its superior efficiency. The results show that incorporating attention mechanisms not only enhances accuracy but also allows the model to focus on essential temporal features, thereby improving its robustness and generalization.

A key factor contributing to the model's efficiency is the use of feature selection techniques, particularly LASSO Regression. Feature selection is crucial in high-dimensional sensor data, where redundancy and noise can negatively impact model performance. By applying LASSO, irrelevant features are removed, and the most significant features are retained. The TCN + AM model achieves 98.74% accuracy with 150 selected features, ensuring that the model is computationally efficient while maintaining high performance.

The performance results also highlight the advantages of TCN-based models over traditional deep learning models like LSTM, GRU, and RNN. For instance, LSTM achieves an accuracy of 88.30%, GRU achieves 87.18%, and RNN achieves 80.68%, all of which are significantly lower than the TCN-based models. These results emphasize the superior ability of TCNs to capture long-range dependencies in time-series data, outperforming RNN-based models, which often suffer from long training times and vanishing gradient problems.

In conclusion, integrating Attention Mechanisms with TCN-based models enhances performance in Human Activity Recognition (HAR). The TCN + AM model outperforms other variations in accuracy and efficiency, effectively handling high-dimensional sensor data. Feature selection techniques like LASSO Regression further optimize the model, ensuring robustness and improved generalization. This combination offers an ideal solution for real-time HAR applications, especially in healthcare, fitness tracking, and smart environments, where computational efficiency and accuracy are crucial.

4.4 INFERENCE FROM SHAP PLOTS

Explainable AI – SHAP

To promote interpretability and transparency in Human Activity Recognition (HAR) models, SHapley Additive exPlanations (SHAP) is implemented as an explainability method. SHAP provides contribution values to each feature, which is a measure of its contribution to model predictions. Through the examination of these attributions, deep learning model decision-making processes are more interpretable, facilitating trust and adoption in important applications like healthcare and intelligent environments. SHAP integration enables debugging, bias identification, and model decisions to be in line with domain expectations.

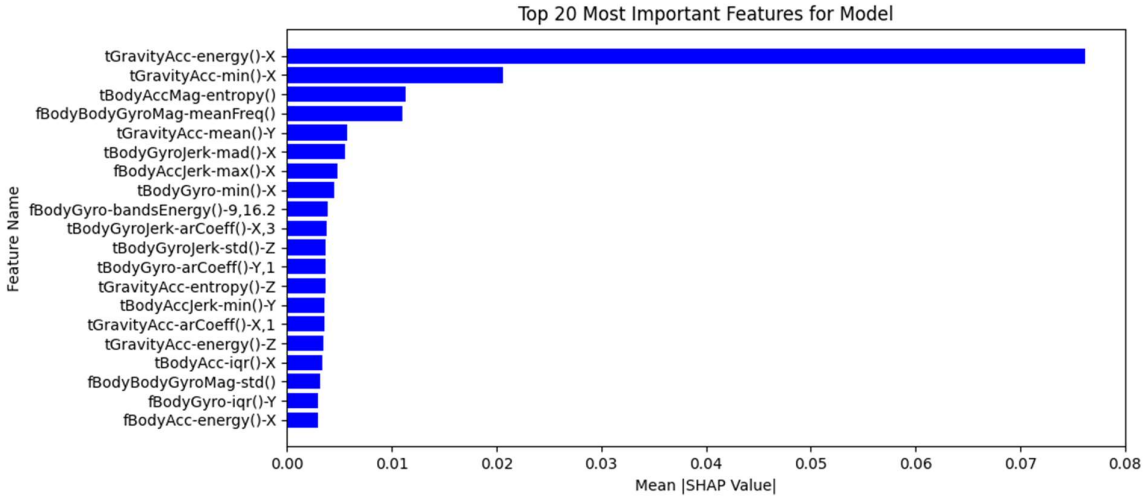


Fig 4.4. Feature Importance Analysis using SHAP Values

Fig 4.4 displays the top 20 most important features in the model as determined by SHAP values. The feature "tGravityAcc-energy()-X" stands out as the most influential, followed by "tGravityAcc-min()-X" and "tBodyAccMag-entropy()". These features play a significant role in driving human activity classification decisions, emphasizing the importance of specific sensor readings in accurately predicting activities. This feature importance analysis helps validate the model's focus on relevant sensor measurements, ensuring effective activity recognition.

The SHAP dependency plot in Fig 4.5 visually represents the relationship between the SHAP value of tGravityAcc-energy()-X and its feature values, highlighting its influence on model predictions. The x-axis shows the feature value for tGravityAcc-energy()-X, while the

y-axis represents its corresponding SHAP value, indicating how much this feature contributes to the model's output. Each point on the plot represents a data sample, helping to understand how changes in the feature value affect the model's prediction.

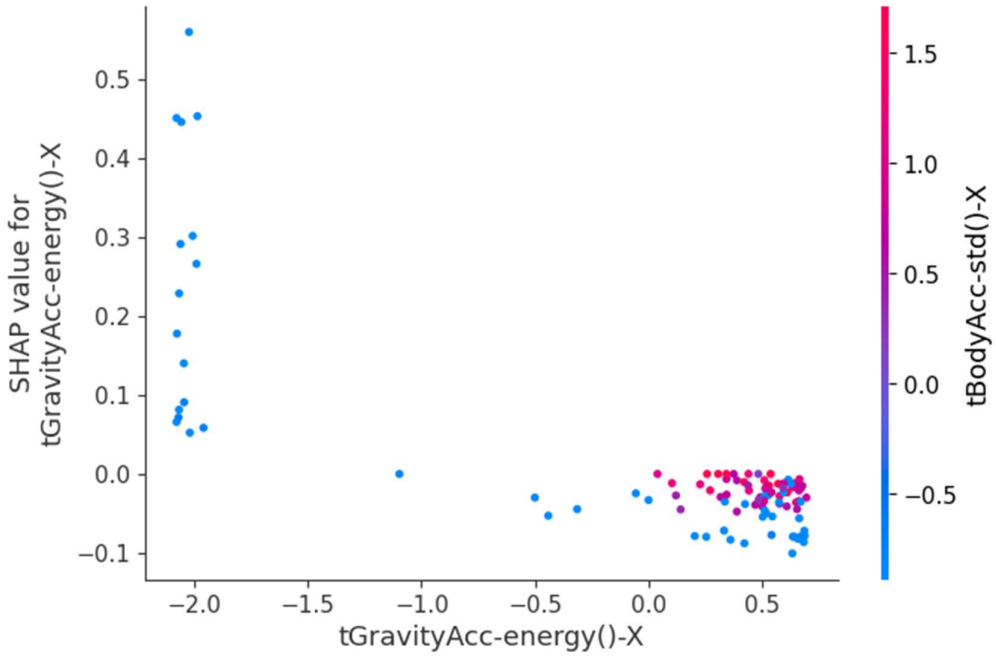


Fig 4.5. SHAP Dependency Plot for tGravityAcc-energy()-X

The plot also uses a color gradient to display the values of another feature, tBodyAcc-std()-X, showing how its interaction with tGravityAcc-energy()-X influences the model's decision-making process. Higher values of tGravityAcc-energy()-X lead to stronger SHAP values, indicating a greater influence on the prediction. This plot provides valuable insights into how feature interactions affect the model's predictions, offering a more interpretable view of the model's behavior in the context of Human Activity Recognition (HAR).

The SHAP dependency plot for tGravityAcc-energy()-X illustrates the intricate relationship between the feature's value and its impact on the model's output. By visualizing how this feature interacts with tBodyAcc-std()-X, it becomes clear that the combination of these features plays a crucial role in the model's predictions. The plot helps identify how variations in tGravityAcc-energy()-X contribute to the model's understanding of human activity, and how the interaction with other features can either strengthen or weaken this effect. This analysis is essential for understanding the model's decision-making process and can guide feature engineering for improved performance.

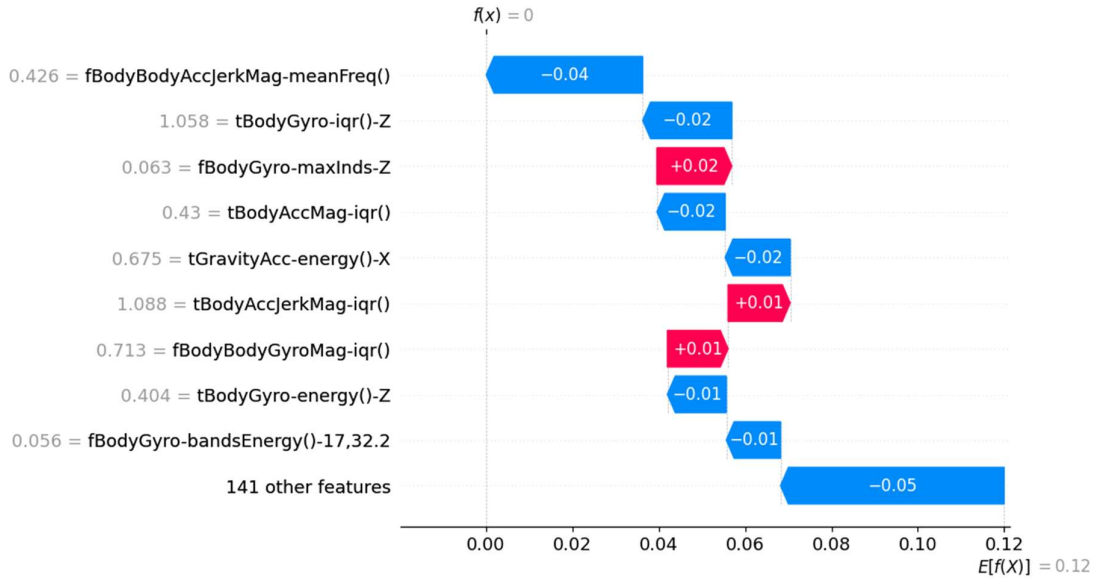


Fig 4.6. SHAP Waterfall Plot for Feature Contribution

The SHAP waterfall plot in Fig 4.6 provides a detailed breakdown of how each feature contributes to the model's prediction. The plot visualizes the cumulative effect of individual features on the model's output, starting from the expected value (baseline) and moving towards the final prediction. Positive and negative contributions are depicted using red and blue bars, respectively, indicating how each feature increases or decreases the output. The larger the magnitude of the bar, the greater the influence of that feature on the final prediction.

In this particular plot, fBodyBodyAccJerkMag-meanFreq() and tGravityAcc-energy()-X are the most significant contributors, moving the model's output in specific directions. These features have the largest impact, with fBodyBodyAccJerkMag-meanFreq() contributing negatively, while tGravityAcc-energy()-X contributes positively. The remaining features, though influential, have a smaller impact on the prediction. This analysis helps identify the key drivers of the model's decision-making process, providing valuable insights into which features are most critical in classifying human activities. It further aids in optimizing feature selection and improving model interpretability.

The SHAP waterfall plot is crucial for understanding the model’s internal workings and how different feature values impact predictions. It helps ensure transparency, allowing practitioners to better align model decisions with domain knowledge and trust the outputs, especially in sensitive applications like healthcare.

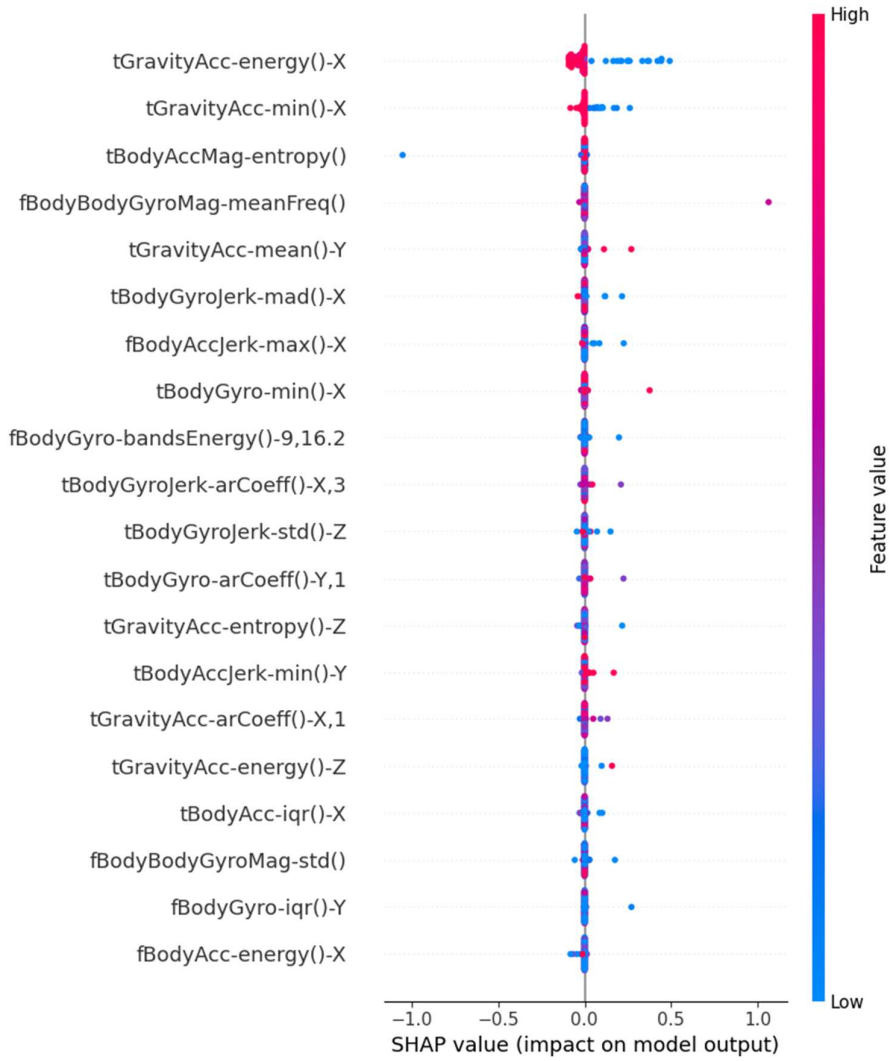


Fig 4.7. SHAP Summary Plot for Feature Contribution

The Fig 4.7 shows the SHAP summary plot, which visualizes the contribution of the top features to the model's prediction. Each point in the plot represents a SHAP value, showing how each feature influences the model’s output. High feature values are depicted in red, while low values are shown in blue. The SHAP value indicates the direction of influence, with higher SHAP values positively affecting the model's prediction and lower values having the opposite effect.From the plot, it is evident that the most influential features are "tGravityAcc-energy()-X" and "tBodyAccMag-entropy()", which contribute significantly to the model's output. These features are key in driving the model's decision-making process, ensuring accurate human activity classification. The plot also highlights how different feature values (high or low) interact with the model, showcasing their importance in predicting the human activity being recognized.

In conclusion, the integration of SHAP in the Human Activity Recognition (HAR) model significantly enhances the interpretability and transparency of the decision-making process. The SHAP plots—feature importance analysis, dependency plots, waterfall plots, and summary plots—provide valuable insights into how different sensor readings contribute to the model's predictions. By visualizing the relationships between features and their impact on the model’s output, SHAP helps identify the most influential features, such as "tGravityAcc-energy()-X" and "tBodyAccMag-entropy()", which drive accurate activity classification. Additionally, SHAP helps improve model interpretability, fostering trust and ensuring alignment with domain expectations, particularly in critical applications like healthcare. This transparency allows for better decision-making, debugging, and optimization, ensuring that the model performs effectively and ethically in real-world scenarios.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

Efficient Human Activity Recognition (HAR) using a feature-optimized deep learning approach has successfully evaluated the performance of various models, including Temporal Convolutional Networks (TCN), TCN with Attention Mechanism (TCN + AM), Bidirectional TCN (Bi-TCN), and Bi-TCN with Attention Mechanism (Bi-TCN + AM), in classifying time-series data obtained from wearable sensors. Through extensive training and testing, the study has yielded key insights into model performance, outlier handling, feature selection, and practical applicability, which are crucial for deploying the system in real-world scenarios.

Among the models evaluated, the TCN + AM configuration has achieved the highest classification accuracy of 98.74% when trained with 150 selected features. This result demonstrates that the integration of the Attention Mechanism significantly enhances model performance, making it the most efficient choice for real-time HAR applications. The attention mechanism enables the model to focus on the most critical time steps in the sensor data, which is key to distinguishing subtle differences in human activity patterns.

Bi-TCN + AM also showed strong performance when trained on larger feature sets, indicating that the model benefits from the bidirectional processing of sensor data, which allows it to capture both past and future dependencies in the input sequences. However, standard TCN, while not yielding the highest accuracy, offered the fastest training time, albeit with a slight trade-off in accuracy. This highlights the trade-off between computational efficiency and model performance in real-time applications.

Proper data preprocessing has emerged as an essential aspect of the proposed HAR system. Winsorization, a method used for handling outliers, has proven to be the most effective in ensuring stable and reliable sensor readings, preserving data integrity while reducing the influence of extreme values. Feature selection using LASSO Regression has successfully reduced the high-dimensional sensor data by retaining only the most relevant features, thus optimizing computational efficiency without compromising classification accuracy. This optimization enables the models to handle HAR data more effectively by eliminating redundancy and focusing on critical information.

The models are assessed using a range of performance metrics, including accuracy, precision, recall, specificity, F1-score, ROC-AUC, training time, and inference time. These metrics provide a comprehensive evaluation of the model's efficiency in both classification performance and real-time execution. The findings suggest that TCN-based models, particularly those enhanced with attention mechanisms, are well-suited for real-time applications in healthcare, fitness tracking, smart environments, and sports analytics, where high accuracy and low latency are critical.

Furthermore, the study has demonstrated the effectiveness of combining deep learning techniques with feature optimization for efficient Human Activity Recognition. The results confirm that TCN-based models, especially the TCN + AM configuration, outperform traditional machine learning and deep learning architectures in terms of accuracy and computational efficiency. The successful integration of LASSO Regression for feature selection and Winsorization for outlier handling has further optimized model performance, providing a robust solution for real-time HAR tasks. These findings contribute valuable insights into the ongoing development of deep learning models for HAR, emphasizing the importance of feature selection, attention mechanisms, and temporal modeling in achieving high-performance systems.

5.2 FUTURE SCOPE

While the proposed approach demonstrates high accuracy and efficiency, several areas offer potential for further research and improvements. A key enhancement is integrating multi-sensor fusion techniques. By combining data from accelerometers, gyroscopes, magnetometers, and other sensors like temperature and pressure sensors, the model’s robustness and accuracy can be significantly improved. This fusion will help the model handle complex activities across various environmental conditions.

Another promising direction is exploring hybrid deep learning architectures, such as combining Temporal Convolutional Networks (TCN) with Recurrent Neural Networks (RNNs) like LSTM and GRU. These hybrid models could leverage TCN’s spatial feature extraction and RNN’s temporal learning capabilities, enhancing the model’s ability to capture both spatial and temporal dependencies effectively.

Real-time deployment on edge devices like smartphones and wearables presents

challenges in optimizing for low power consumption and minimal latency. Lightweight models that maintain performance while being computationally efficient will enable scalability, especially in resource-constrained environments like mobile and wearable devices.

Improving model explainability is crucial, particularly in sensitive domains like healthcare. While SHAP is used to interpret decision-making, more advanced techniques like LIME could offer more detailed insights, enhancing transparency and trust in the system.

Future advancements in fine-grained activity recognition could expand HAR's practical applications, such as recognizing activities involving multiple people in social, elder care, or professional settings. Integrating HAR systems with IoT devices and smart environments could enable real-time decision-making in healthcare facilities, smart homes, and industrial settings.

Finally, expanding the model's ability to recognize abnormal activities, such as falls or sudden health events, could greatly enhance its impact in healthcare and safety applications. By integrating intelligent alert systems, HAR could provide timely responses in critical situations, particularly for elderly care and patient monitoring.

In conclusion, while the current approach shows strong performance, expanding it to include multi-sensor fusion, hybrid architectures, real-time edge deployment, explainability, and fine-grained activity recognition will increase its real-world applicability. Future research in these areas will enhance the robustness, efficiency, and adaptability of HAR systems, enabling broader adoption in healthcare, fitness, smart environments, and beyond.

ANNEXURE 1

SOURCE CODE & SCREEN SHOTS

DEEP LEARNING APPROACH

```
# This code imports necessary libraries and lists all files in the /Kaggle /input directory

import numpy as np

import pandas as pd

import os

for dirname, filenames in os.walk('/kaggle/input'):

for filename in filenames:

print(os.path.join(dirname, filename))
```

TCN + AM – 98.74

Code:

```
import numpy as np
import pandas as pd
import tensorflow as tf
from tensorflow.keras.models import Model
from tensorflow.keras.layers import (
Input, Conv1D, BatchNormalization, Dropout, ReLU, Flatten, Add, Dense, Multiply,
GlobalAveragePooling1D, Reshape
)
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LassoCV
from sklearn.metrics import (
accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, roc_auc_score
)
import time
import matplotlib.pyplot as plt
```

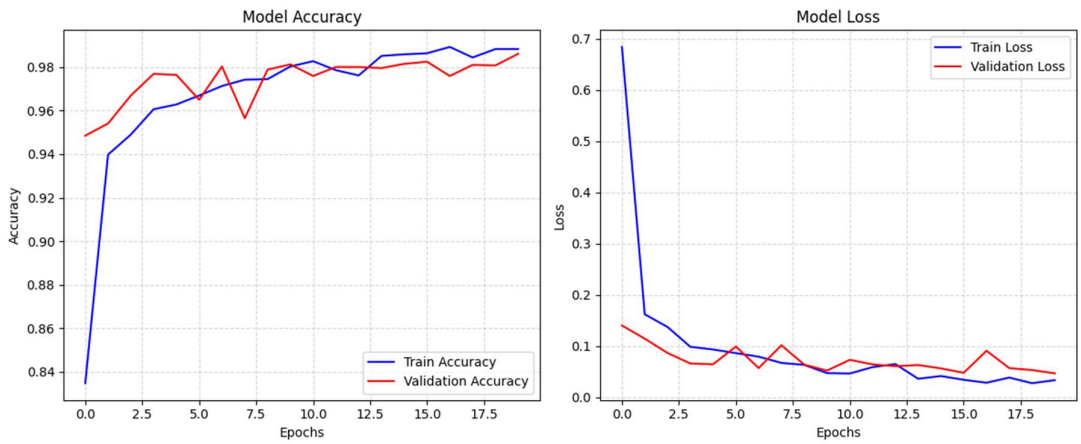
```
from tqdm import tqdm  # Progress bar

# Load dataset
df=pd.read_csv('/kaggle/input/har-
dataset/Human_Activity_Recognition_Using_Smartphones_Data.csv')

# Encode target labels
label_encoder = LabelEncoder()
df['Activity'] = label_encoder.fit_transform(df['Activity'])

# Separate features and target
X = df.drop(columns=['Activity']).values
y = df['Activity'].values

# === Outlier Detection & Replacement using Winsorizing ===

winsor_limit = 0.1  # 10% Winsorizing (both sides)

for col in X.select_dtypes(include=[np.number]):  # Only numeric columns

X[col] = winsorize(X[col], limits=[winsor_limit, winsor_limit])  # Apply Winsorizin

# Feature Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Feature Selection using LASSO
lasso = LassoCV(cv=5, random_state=42)
lasso.fit(X_scaled, y)

# Extract absolute coefficients from the best alpha model
feature_importance = np.abs(lasso.coef_)

# Get feature names
feature_names = df.drop(columns=["Activity"]).columns
```

```
# Select top 150 features based on importance
sorted_idx = np.argsort(feature_importance)[::-1][:150]  # Select top 150 features
selected_features = np.array(feature_names)[sorted_idx]
num_features = len(selected_features)  # Get number of selected features

# Print the number of selected features
print(f"Number of features used for model training: {num_features}")

# Select corresponding columns from scaled dataset
X_selected = X_scaled[:, sorted_idx]

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=0.2,
random_state=42)

# Reshape for TCN input
X_train = X_train[..., np.newaxis]
X_test = X_test[..., np.newaxis]

# Define Attention Mechanism
def attention_block(inputs):
"""Attention Mechanism"""
avg_pool = GlobalAveragePooling1D()(inputs)
dense1 = Dense(units=inputs.shape[-1], activation="relu")(avg_pool)
dense2 = Dense(units=inputs.shape[-1], activation="sigmoid")(dense1)
attention = Multiply()([inputs, Reshape((1, inputs.shape[-1]))(dense2)])
return attention

# Define TCN Residual Block with Dilated Convolutions
def residual_block(x, filters, dilation_rate):
"""TCN Residual Block"""
conv1 = Conv1D(filters=filters, kernel_size=3, padding='causal',
dilation_rate=dilation_rate)(x)
bn1 = BatchNormalization()(conv1)
```

```
relu1 = ReLU()(bn1)
dropout1 = Dropout(0.3)(relu1)
conv2 = Conv1D(filters=filters, kernel_size=3, padding='causal',
dilation_rate=dilation_rate)(dropout1)
bn2 = BatchNormalization()(conv2)
relu2 = ReLU()(bn2)
# Residual Connection
res = Add()([x, relu2]
return res
# Cross-validation setup
kf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
best_accuracy = 0
best_model = None
best_training_time = None

# Cross-validation loop
with tqdm(total=kf.get_n_splits(), desc="Running Cross-Validation", unit="split") as pbar:
for train_index, val_index in kf.split(X_train, y_train):
X_tr, X_val = X_train[train_index], X_train[val_index]
y_tr, y_val = y_train[train_index], y_train[val_index]

# Define TCN Model
input_layer = Input(shape=(X_train.shape[1], 1))

# TCN Residual Blocks with Dilated Convolutions
res_block1 = residual_block(input_layer, filters=64, dilation_rate=1)
res_block2 = residual_block(res_block1, filters=64, dilation_rate=2)
res_block3 = residual_block(res_block2, filters=64, dilation_rate=4)

# Attention Mechanism
attention_output = attention_block(res_block3)

flatten = Flatten()(attention_output)
dense1 = Dense(128, activation='relu')(flatten)
```

```
dropout2 = Dropout(0.3)(dense1)
output_layer = Dense(len(np.unique(y)), activation='softmax')(dropout2)

model = Model(inputs=input_layer, outputs=output_layer)
model.compile(optimizer=Adam(learning_rate=0.001),
loss='sparse_categorical_crossentropy', metrics=['accuracy'])
# Train the model and track training time
start_train_time = time.time()
history = model.fit(X_tr, y_tr, epochs=20, batch_size=32, validation_data=(X_val, y_val),
verbose=0)
training_time = time.time() - start_train_time

# Evaluate the model
val_accuracy = model.evaluate(X_val, y_val, verbose=0)[1]
if val_accuracy > best_accuracy:
best_accuracy = val_accuracy
best_model = model
best_training_time = training_time
pbar.update(1)  # Update progress bar
# Evaluate the best model
start_time = time.time()
y_pred_prob = best_model.predict(X_test)
inference_time = time.time() - start_time
y_pred = np.argmax(y_pred_prob, axis=1)

# Metrics Calculation
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')
cm = confusion_matrix(y_test, y_pred)
roc_auc = roc_auc_score(tf.keras.utils.to_categorical(y_test), y_pred_prob, multi_class='ovr')

# Specificity Calculation
```

```
specificity = np.mean(np.diag(cm) / np.sum(cm, axis=1))

# Print Metrics
print(f"Features used for model training: {num_features}")
print(f"Model Test Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall (Sensitivity): {recall:.4f}")
print(f"Specificity: {specificity:.4f}")
print(f"F1-Score: {f1:.4f}")
print(f"ROC AUC: {roc_auc:.4f}")
print(f"Training Time (Best Model): {best_training_time:.2f} seconds")
print(f"Inference Time: {inference_time:.2f} seconds")
```



Accuracy & Loss Graph for TCN + AM

```
Running Cross-Validation: 100%|██████████| 5/5 [03:22<00:00, 40.52s/split]

65/65 ━━━━━━━━━━━━━━━━━━━━ 2s 17ms/step
Features used for model training: 150
Model Test Accuracy: 0.9874
Precision: 0.9874
Recall (Sensitivity): 0.9874
Specificity: 0.9880
F1-Score: 0.9874
ROC AUC: 0.9996
Training Time (Best Model): 41.15 seconds
Inference Time: 2.06 seconds
```

SHAP Codes:

```
# === SHAP Visualizations ===
import shap

import numpy as np

import matplotlib.pyplot as plt

# Ensure correct input shape for SHAP

X_test_2d = X_test.squeeze(axis=-1)  # Convert from (samples, features, 1) → (samples, features)

# Define SHAP prediction function

def model_predict(X):

    X = np.expand_dims(X, axis=-1)  # Convert back to (samples, features, 1)

    return best_model.predict(X).astype(np.float64)

# Use a small subset of training data as background for SHAP

background = X_train.squeeze(axis=-1)[:50]  # Use only 50 background samples

# Initialize SHAP KernelExplainer

explainer = shap.KernelExplainer(model_predict, background)

# Compute SHAP values for 100 test samples

shap_values = explainer.shap_values(X_test_2d[:100], nsamples=100)

# Select SHAP values for a specific class (e.g., class 0)

shap_values_class = shap_values[0]  # Change index if needed for another class

# Compute mean absolute SHAP values for feature ranking

mean_shap_values = np.mean(np.abs(shap_values_class), axis=0)  # Mean over samples
```

```
# Get top 20 most important features

top_20_indices = np.argsort(mean_shap_values)[-20:][::-1]  # Get indices of top 20 features

top_20_features = selected_features[top_20_indices]  # Use feature names

top_20_importance = mean_shap_values[top_20_indices]  # Get importance values

# Print Top 20 Features and Their Importance

print("\n 🔹 Top 20 Most Important Features:")

for i, (feature, importance) in enumerate(zip(top_20_features, top_20_importance), 1):

    print(f"{i}. {feature}: {importance:.5f}")

# 1 Summary Plot - Feature importance for one class

plt.figure(figsize=(10, 6))

shap.summary_plot(shap_values_class, X_test_2d[:100], feature_names=selected_features)

# 2 Bar Chart of Top 20 Features

plt.figure(figsize=(10, 5))

plt.barh(top_20_features[::-1], top_20_importance[::-1], color="blue")

plt.xlabel("Mean |SHAP Value|")

plt.ylabel("Feature Name")

plt.title("Top 20 Most Important Features for Model")

plt.show()

--

import shap

import matplotlib.pyplot as plt
```

```
# Ensure SHAP values are computed

shap_values_class = shap_values[0]  # Select SHAP values for class 0

# === 1 Beeswarm Plot (Alternative to Summary Plot) ===

plt.figure(figsize=(10, 6))

shap.summary_plot(shap_values_class, X_test_2d[:100], feature_names=selected_features)

# === 2 Dependence Plot (Feature Interaction Effects) ===

# Select an important feature (e.g., the most important feature)

feature_to_plot = top_20_indices[0]  # Change index for different feature

shap.dependence_plot(feature_to_plot,          shap_values_class,          X_test_2d[:100],
feature_names=selected_features)

# === 3 Waterfall Plot (SHAP Contribution Breakdown for One Sample) ===

sample_idx = 0  # Choose a test sample index

shap.plots.waterfall(shap.Explanation(

    values=shap_values_class[sample_idx],

    base_values=explainer.expected_value[0],

    data=X_test_2d[sample_idx],

    feature_names=selected_features))

plt.show()
```

REFERENCES

[1] Bulbul, E., Cetin, A., & Dogru, I. A. (2018, October). Human activity recognition using smartphones. In *2018 2nd international symposium on multidisciplinary studies and innovative technologies (ismsit)* (pp. 1-6). IEEE.

[2] S. Thangapriya and N. J. Goldena, "Comparison of Feature Selection Algorithms for Human Activity Recognition," *Department of Computer Applications and Research Centre, Sarah Tucker College (Autonomous), affiliated to Manonmaniam Sundaranar University, Tirunelveli*, 2021.

[3] Ahmad Mahir, R., & Al-Khazaleh, A. M. H. (2009). New method to estimate missing data by using the asymmetrical winsorized mean in a time series. *Applied Mathematical Sciences*, *3*(35), 1715-1726.

[4] Bai, Y., Tao, Z., Wang, L., Li, S., Yin, Y., & Fu, Y. (2022). Collaborative attention mechanism for multi-modal time series classification. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)* (pp. 495-503). Society for Industrial and Applied Mathematics.

[5] Brauwers, G., & Frasincar, F. (2021). A general survey on attention mechanisms in deep learning. *IEEE Transactions on Knowledge and Data Engineering*, *35*(4), 3279-3298.

[6] Yu, Y., Xia, N., & Li, J. (2021, May). Application of attention mechanism for multi-time series in fault diagnosis. In *2021 33rd Chinese Control and Decision Conference (CCDC)* (pp. 1646-1651). IEEE.

[7] Basu, S. (2023, June). Machine learning and visualizations for time-series healthcare data. In *2023 IEEE 11th International Conference on Healthcare Informatics (ICHI)* (pp. 485-485). IEEE.

[8] Nguyen, T. M., & Motoki, T. (2024). Using a Semi-supervised Learning Model for Recognition of Human Daily Activities from Wearable Sensor Data. *International Journal of Artificial Intelligence Research*, *8*(1), 57-64.

[9] Sekaran, S. R., Han, P. Y., & Yin, O. S. (2023). Smartphone-based human activity recognition using lightweight multiheaded temporal convolutional network. *Expert Systems with Applications*, *227*, 120132.

[10] Kaya, Y., & Topuz, E. K. (2024). Human activity recognition from multiple sensors data using deep CNNs. *Multimedia Tools and Applications*, *83*(4), 10815-10838.

[11] Ahmad, T., Wu, J., Alwageed, H. S., Khan, F., Khan, J., & Lee, Y. (2023). Human activity recognition based on deep-temporal learning using convolution neural networks features and bidirectional gated recurrent unit with features selection. *IEEE access*, *11*, 33148-33159.

[12] Anusha, P. V., Anuradha, C., Murty, P. C., & Kiran, C. S. (2019). Detecting outliers in high dimensional data sets using Z-score methodology. *International Journal of Innovative Technology and Exploring Engineering*, *9*(1), 48-53.

[13] Sanal Kumar, K. P., & Bhavani, R. (2019). Human activity recognition in egocentric video using PNN, SVM, kNN and SVM+ kNN classifiers. *Cluster Computing*, *22*(Suppl 5), 10577-10586.

[14] Sikder, N., Chowdhury, M. S., Arif, A. S. M., & Nahid, A. A. (2019, September). Human activity recognition using multichannel convolutional neural network. In *2019 5th International conference on advances in electrical engineering (ICAEE)* (pp. 560-565). IEEE.

[15] Lara-Benítez, P., Carranza-García, M., Luna-Romera, J. M., & Riquelme, J. C. (2020). Temporal convolutional networks applied to energy-related time series forecasting. *applied sciences*, *10*(7), 2322.

[16] Li, X., Liu, C., & He, Y. (2021, August). Efficient Time Series Predicting with Feature Selection and Temporal Convolutional Network. In *2021 IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET)* (pp. 141-145). IEEE.

[17] Ahmad Mahir, R., & Al-Khazaleh, A. M. H. (2009). New method to estimate missing data by using the asymmetrical winsorized mean in a time series. *Applied Mathematical Sciences*, *3*(35), 1715-1726.

[18] Hamdard, M. S., & Lodin, H. (2023). Effect of feature selection on the accuracy of machine learning model. *Int. J. Multidiscipl. Res. Anal*, *6*(9), 4460-4466.

[19] Lazaros, K., Tasoulis, S., Vrahatis, A., & Plagianakos, V. (2022, December). Feature selection for high dimensional data using supervised machine learning techniques. In *2022 IEEE International Conference on Big Data (Big Data)* (pp. 3891-3894). IEEE.

[20] Anusha, P. V., Anuradha, C., Murty, P. C., & Kiran, C. S. (2019). Detecting outliers in high dimensional data sets using Z-score methodology. *International Journal of Innovative Technology and Exploring Engineering*, *9*(1), 48-53.

[21] Djenouri, Y., & Belbachir, A. N. (2023). A hybrid visual transformer for efficient deep human activity recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 721-730).

[22] Basly, H., Ouarda, W., Sayadi, F. E., Ouni, B., & Alimi, A. M. (2022). DTR-HAR: deep temporal residual representation for human activity recognition. *The Visual Computer*, *38*(3), 993-1013.

[23] Li, Z., Jiang, T., Yu, J., Ding, X., Zhong, Y., & Liu, Y. (2021, July). A lightweight mobile temporal convolution network for multi-location human activity recognition based on wi-fi. In *2021 IEEE/CIC International Conference on Communications in China (ICCC Workshops)* (pp. 143-148). IEEE.

[24] Shekhar, R., Tomar, D. S., Pateriya, R. K., & Sharan, B. (2023, March). Human activity recognition with smartphone using classical machine learning models. In *2023 10th International conference on computing for sustainable global development (INDIACom)* (pp. 85-90). IEEE.

[25] Jing, L., & Cheng, Z. (2017, October). Recognition of daily routines and accidental event with multipoint wearable inertial sensing for seniors home care. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 2324-2389). IEEE.

[26] Jolliffe, I. T. (2002). *Principal component analysis for special types of data* (pp. 338-372). Springer New York.

[27] Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In *2008 eighth ieee international conference on data mining* (pp. 413-422). IEEE.

[28] Reyes-Ortiz, J., Anguita, D., Ghio, A., Oneto, L., & Parra, X. (2013). Human Activity Recognition Using Smartphones [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C54S4K.

Project Report Evaluation form

Criteria	Excellent (5)	Good (4)	Satisfactory (3)	Needs Improvement (2)	Poor (1)
Report Formatting	Fully formatted as per guidelines	Minor formatting issues	Moderate formatting issues	Several formatting issues	Not formatted
Grammar and Language	No grammatical/spelling errors	Few minor errors	Noticeable errors	Frequent errors	Poor grammar, hard to understand
Referencing, Tables & Figures	All cited properly, visuals are clear	Minor citation/visual issues	Some missing citations or unclear visuals	Many missing or incorrect citations	No citations; visuals poorly presented
Technical Content Quality	In-depth, innovative, and well-explained with original content	Clear and relevant	Meets basic expectations	Lacks depth or clarity	Incomplete or incorrect technical info

Name	RHENIUS A
Reg. No	URK21CS1073
Report Correctly formatted	
No Grammatical Errors	
All References are Cited Inside; Tables and Figures Represented Properly	
Technical Content	
Total	

Guide Signature