

Democratic and Popular Algerian Republic
Ministry of Higher Education and Scientific Research
Faculty of Sciences
Departement of Mathematics and Computer Science

YAHIA FARES UNIVERSITY OF MEDEA

FINAL MASTER PROJECT

Human Activity Recognition Using Deep Learning Approach-Application for Patient and Elderly Person Supervision

---

Authors:
Mohammedi Oussama
Mazighi Oussama

Supervisor:
Rahim Messaoud

Co-Supervisor:
Boudjema Ali

2022/2023

Dedication

I dedicate this work:

To ALLAH Almighty who opens my mind before my eyes to guide me, enlighten my heart, and show me clearly the right way so I follow it, and show me clearly the wrong way so I avoid it.

To the person who is the source of inspiration, wisdom, knowledge, understanding, mercy, peace and the person who encourages people to be patient, our Prophet Mohammad.

To my dear parents, whose unconditional love, sacrifices, and unwavering belief in my abilities have been the cornerstone of my success. Your steadfast presence, encouragement, and endless sacrifices have shaped me into the person I am today.

To my beloved family, with whom I have shared countless days and nights filled with joy, laughter, tears, and bittersweet moments. Your constant love, support, and enduring presence have been the foundation of my strength and resilience.

to my amazing friends (can't mention all of you here, but each of you knows the special place you hold in my heart), who have been my constant companions on this remarkable journey. From the joyful moments to the challenging times, we have stood together, creating memories that will forever hold a special place in my heart.

"My love for you all can never be quantified. God bless you all"

Mazighi Oussama

Dedication

I dedicate this thesis :

to my dear parents and all my family for their sacrifices, love, affection, and support throughout my studies.

To my brothers and sisters: Islam, Faras, Somia.

to My dear friends, whether near or far, for their constant encouragement and moral support.

to all those who love me.

"I thank you all."

Mohammedi Oussama

Acknowledgment

We would like to extend our heartfelt gratitude to the individuals who have played pivotal roles in the successful completion of this thesis and have been instrumental in my academic journey.

First and foremost, we would like to express our deep appreciation to our thesis advisor, Rahim Messaoud. Their exceptional guidance, unwavering support, and profound expertise have shaped this research and have been invaluable throughout the entire process.

We are equally grateful to our co-advisor, Boudjema Ali, for their valuable contributions, insightful feedback, and collaborative spirit. Their dedication and support have been invaluable in ensuring the success of this research endeavor.

We would also like to thank the members of the jury for their time, effort, and expertise in evaluating this thesis. We sincerely appreciate their constructive feedback, valuable insights, and critical evaluation.

We are deeply grateful to our family for their unconditional love, encouragement, and continuous belief in our abilities.

Finally, we want to extend our sincere thanks to all the individuals who have supported us, whether through their kind words, encouragement, or understanding. Your belief in our abilities and your presence in our lifes have been a source of inspiration and motivation.

Abstract

Given the rising number of elderly individuals and growing demands on healthcare systems, Human Activity Recognition (HAR) has emerged as a vital tool for improving the care and well-being of elderly individuals and patients. By accurately identifying and classifying activities, HAR enables caregivers and medical professionals to closely monitor movements, detect anomalies, and provide timely assistance. In this study, we propose a Multi-Branch Convolutional LSTM Network (MBCLNet), a state-of-the-art deep learning model for HAR. Through a systematic approach, we develop and evaluate it using two benchmark datasets WISDM and PAMAP2. Our model achieves impressive accuracy rates of 99.98% and 99.95% on the respective datasets. By integrating MBCLNet into an API we enable real-time processing through a user interface, we enhance monitoring and supervision in healthcare environments, leading to improved control, care, and well-being for patients and elderly individuals. This research contributes to advancing healthcare outcomes and patient safety, with the potential for widespread implications in healthcare settings.

Keywords— *Human activity recognition, Deep learning, CNN, LSTM, WISDM, PAMAP2, API*

ملخص

بالنظر إلى زيادة عدد كبار السن والطلب المتزايد على أنظمة الرعاية الصحية، ظهر التعرف على النشاط البشري (HAR) كأداة حيوية لتحسين رعاية ورفاهية كبار السن والمرضى. من خلال تحديد وتصنيف الأنشطة بدقة، يمكن للمقدمين الرعاية والمحترفين الطبيين مراقبة حركات المرضى بشكل وثيق، واكتشاف الانحرافات، وتقديم المساعدة في الوقت المناسب. في هذه الدراسة، نقترح شبكة متعددة الفروع للتعلم العميق باستخدام الشبكة العصبية المتكاملة (MBCLNet) كنموذج حديث للتعرف على النشاط البشري. من خلال نهج منهجي، نطور ونقيم النموذج باستخدام مجموعتي بيانات المقاييس الأساسية WISDM و PAMAP2. يحقق نموذجنا معدلات دقة مذهلة تبلغ 99.98% و 99.95% على التوالي في المجموعات البيانية المقابلة. من خلال دمج MBCLNet في واجهة برمجة تطبيقات (API)، نمكّن المعالجة في الوقت الحقيقي من خلال واجهة المستخدم، مما يعزز المراقبة والإشراف في بيئات الرعاية الصحية، مما يؤدي إلى تحسين السيطرة والرعاية ورفاهية المرضى وكبار السن. تسهم هذه البحث في تقدم نتائج الرعاية الصحية وسلامة المرضى، مع إمكانية آثار واسعة النطاق في إعدادات الرعاية الصحية.

الكلمات المفتاحية--- التعرف على أنشطة الإنسان، التعلم العميق، CNN ، LSTM، WISDM ، PAMAP2 ، API

Resumé

Étant donné le nombre croissant de personnes âgées et les exigences croissantes des systèmes de santé, la Reconnaissance d'Activité Humaine (RAH) est devenue un outil essentiel pour améliorer les soins et le bien-être des personnes âgées et des patients. En identifiant et classifiant précisément les activités, la RAH permet aux soignants et aux professionnels de la santé de surveiller de près les mouvements, de détecter les anomalies et de fournir une assistance en temps opportun. Dans cette étude, nous proposons un réseau Multi-Branch Convolutional LSTM (MBCLNet), un modèle de deep learning de pointe pour la RAH. Grâce à une approche systématique, nous le développons et l'évaluons à l'aide de deux ensembles de données de référence, WISDM et PAMAP2. Notre modèle atteint des taux de précision impressionnants de 99,98% et 99,95% sur les ensembles de données respectifs. En intégrant MBCLNet dans une API, nous permettons un traitement en temps réel à travers une interface utilisateur, améliorant ainsi la surveillance et la supervision dans les environnements de soins de santé, ce qui conduit à un meilleur contrôle, des soins et un bien-être améliorés pour les patients et les personnes âgées. Cette recherche contribue à l'avancement des résultats de santé et à la sécurité des patients, avec des implications potentiellement généralisées dans les contextes de soins de santé.

Mots Clés— *Reconnaissance d'activité humaine, Apprentissage profond, CNN, LSTM, WISDM, PAMAP2, API*

List of Acronyms

AI : Artificial Inteligence
ANN : Artificial Neural Networks
API : Application Programming Interface
Bi-LSTM : Bidirectional LSTM
CNN : Convolutional Neural Network
CSS : Cascading Style Sheets
DL : Deep Learning
DNN : Deep Neural Network
GAP : Global Average Pooling
GPU : Graphics Processing Unit
HAR : Human Activity Recognition
HTML : Hypertext Markup Language
HTTP : Hypertext Transfer Protocol
LSTM : Long Short Term Memory
ML : Machine Learning
MSE : Mean Squared Error
NLP : Natural Language Processing
PAMAP2 : Physical Activity Monitoring of Older Adults, version 2
PDO : Php Data Objects
RAM : Random Access Memory
ReLU : Rectified Linear Unit
SNN : Simulated Neural Network
REST API : Representational State Transfer Application Programming Interface
RNN : Recurrenl Neural Network
SGD : Stochastic Gradient Descent
SQL : Structured Query Language
SVM : Support vector machine
WISDM : Wireless Sensor Data Mining

Contents

1 Introduction **1**

1.1 Introduction 2
1.2 Research Problem 4
1.3 Research Objectives 5
1.4 Research Methodology 6
1.5 Thesis Outline 6

2 Artificial Intelligence **8**

2.1 Introduction 9
2.2 Artificial Neural Network (ANN) 11
2.2.1 Components Of ANN 12
2.2.1.1 Artificial Neuron 12
2.2.1.2 Parameters 12
2.2.1.3 Activation Function 12
2.2.1.4 Layers: 15
2.2.2 ANN Architecture HyperParameters 16
2.2.2.1 Network Hyper parameters 16
2.2.2.2 Training Hyper Parameter 18
2.2.3 Training Phase in ANN 19
2.2.4 Challenges of Training 21

2.2.5 Types of Artificial Neural Networks 21
2.2.5.1 Deep Neural Network (DNN) 22
2.2.5.2 Convolutional Neural Networks (CNN) 23
2.2.5.3 Recurrent Neural Networks (RNN) 24
2.2.5.4 Long Short-Term Memory (LSTM) 25
2.2.5.5 Autoencoder 26
2.2.5.6 Transfer Learning 26
2.3 Deep Learning for HAR 28
2.4 Conclusion 33

3 Conception **34**
3.1 Introduction 35
3.2 Overall Architecture 35
3.3 AI-API for HAR 37
3.3.1 Datasets 37
3.3.1.1 WISDM 38
3.3.1.2 PAMAP2 40
3.3.2 Dataset Preparation 43
3.3.2.1 Data Preprocessing 43
3.3.2.2 Data Splitting 44
3.3.3 Our Approach 45
3.3.3.1 Autoencoder Model 46
3.3.3.2 Proposed Model 49
3.4 Results and Discussion 52
3.4.1 Key Classification Metrics 52
3.4.1.1 Accuracy and losses curves 53
3.4.1.2 Classification Report 55
3.4.1.3 Confusion Matrix 57
3.4.2 Comparison 60
3.5 Conclusion 61

4 Realisation & Implementation **63**
4.1 Introduction . . . 64
4.2 Environment & Development Tools . . . 64
4.2.1 Programming Language, Framework & API . . . 64
4.2.1.1 Python . . . 64
4.2.1.2 JavaScript . . . 65
4.2.1.3 Django Framework . . . 65
4.2.1.4 REST API . . . 66
4.2.2 Environment . . . 66
4.2.2.1 COLAB . . . 66
4.2.2.2 Google Drive . . . 67
4.2.2.3 Node.js . . . 67
4.2.3 Libraries . . . 67
4.2.3.1 TensorFlow . . . 67
4.2.3.2 Keras . . . 67
4.2.3.3 Pandas . . . 68
4.2.3.4 NumPy . . . 68
4.2.3.5 Matplotlib . . . 68
4.2.3.6 Scikit-learn . . . 69
4.3 Junction Layer (Node-Red) . . . 69
4.3.1 Nodes . . . 70
4.3.2 Flow . . . 71
4.4 Junction Layer Implementation & Simulation . . . 71
4.5 Web Application . . . 73
4.6 Conclusion . . . 76

List of Figures

1.1 Broad Classification of Human Activity Recognition. 3

2.1 The Relation Between AI, ML and Deep Learning [3]. 10
2.2 ANN's Basic Structure. 11
2.3 Different Training Parameters of ANN. 13
2.4 Different Activation Functions. 14
2.5 Different Types of ANN Layers. 15
2.6 General Structure of Artificial Neural Network [48]. 16
2.7 Different Network HyperParameters. 17
2.8 Different Training HyperParameters. 18
2.9 Feed forward back propagation mechanism in artificial neural network [2]. 19
2.10 Essential Steps in Neural Network Training. 20
2.11 Different Types of ANN Architecture. 22
2.12 Architecture Of Deep Neural Network. 23
2.13 Illustration of Convolutional Neural Network [29]. 24
2.14 Difference Between Max Pooling And Average Pooling [43]. 24
2.15 Schema of RNN. 25
2.16 Architecture of LSTM. 26
2.17 Presenting Architecture Of Autoencoder [54]. 27
2.18 Transfer Learning Concept. 27

3.1 Display the Way Our Architecture Work 36
3.2 Sequence Diagram of Our Architecture 37
3.3 Placement of Inertial Sensors in WISDM Dataset 38
3.4 Class Distribution of WISDM Dataset 39
3.5 Placement of Inertial Sensors in PAMAP2 Dataset 41
3.6 Class Distribution of PAMAP2 Dataset 42
3.7 The approach we adopted to train our model 46
3.8 The Autoencoder Model Used 47
3.9 The Loss Value of the Autoencoder Model 48
3.10 The Presented Figure Displays the Utilized Architecture 50
3.11 Accuracy and Loss Training Curves on WISDM Dataset 54
3.12 The Training Process of Our Model on WISDM Dataset 54
3.13 Accuracy and Loss Training Curves on PAMAP2 Dataset 54
3.14 The Training Process of Our Model on PAMAP2 Dataset 55
3.15 Evaluation Model of WISDM Dataset 56
3.16 Evaluation Model of PAMAP2 Dataset 57
3.17 Confusion Matrix of WISDM Dataset 58
3.18 Confusion Matrix of PAMAP2 Dataset 59

4.1 The Node-Red Platform 70
4.2 Some of the nodes that are available in Node-Red 71
4.3 First Part of Simulation and Implementation of Junction Layer 72
4.4 First Part of Simulation and Implementation of Junction Layer 73
4.5 Web Application Authentication 74
4.6 Web Application Dashboard 74
4.7 State of Patient Safe in the Bed 75
4.8 State of Patient Safe Out of the Bed 75
4.9 State of Patient Danger Out of the Bed 75

List of Tables

3.1 General Properties of WISDM Dataset 39
3.2 Structure of WISDM Dataset 40
3.3 General Properties of PAMAP2 Dataset 41
3.4 Structure of PAMAP2 Dataset 42
3.5 The Different Number of Parameters 51
3.6 Different Key Classification Metrics 53
3.7 Results of Training and Testing on Different Parameters 60
3.8 Table Represent Comparison of Classification Results for Different Methods an Our Method on WISDM and PAMAP2 Datasets 61

Chapter 1

Introduction

Unlocking the Rhythm of Human Activity Recognition: Prelude and Methodological Harmonies

1.1 Introduction

Human Activity Recognition (HAR) is the task of classifying a physical activity performed by one or more users. Human Activity Recognition plays an important role in people's daily life because it has the ability to learn deeply advanced knowledge of human activity [59].

Communities around the world have shown a keen interest in the field of HAR as part of their commitment to enhancing healthcare systems and improving the well-being of individuals. They recognize the importance of utilizing medical and assistive technologies to develop innovative systems that enable long-term care and support independence for people. By embracing HAR, communities aim to create suitable living conditions that prioritize the quality of life and promote the overall well-being of individuals [12].

Human activity recognition has a wide range of potential applications that are diverse and extensive. These applications include areas such as virtual education, entertainment, sport injury detection, healthcare, rehabilitation, and monitoring smart home environments [21]. HAR technology has the ability to contribute to these fields by accurately detecting and analyzing human activities, enabling various benefits and advancements in different domains [61]. Healthcare field has multiple potential applications of HAR, making it a valuable and promising area of research.

Firstly the recovery process after a stroke extends beyond the hospital setting and continues in the home environment. HAR enables healthcare professionals to capture motor performance characteristics and gain valuable insights into patients' daily routines and activity levels. By continuously monitoring these factors, HAR can identify any changes or trends that might indicate health issues or complications. This objective and comprehensive monitoring can be done both at home and in community settings, providing a more holistic approach to stroke patient care and enhancing the ability to detect and address potential problems in a timely manner [40].

Moreover, HAR can be utilized to monitor the movements of surgical instruments

and the actions of surgical teams and provide real-time feedback to the surgical team to reduce the risk of errors and improving patient safety [42].

HAR can also be used to detect falls, which is a significant concern for elderly individuals and those with mobility problems. especially when falls are the second leading cause of unintentional injury deaths worldwide [31]. HAR algorithms can analyze the data collected to identify the distinct patterns of falls and send alerts or notifications to healthcare professionals or family members.

When it comes to human activity recognition tasks, the usual procedure involves gathering data that is commonly categorized as either vision-based or sensor-based as shown in Figure 1.1.

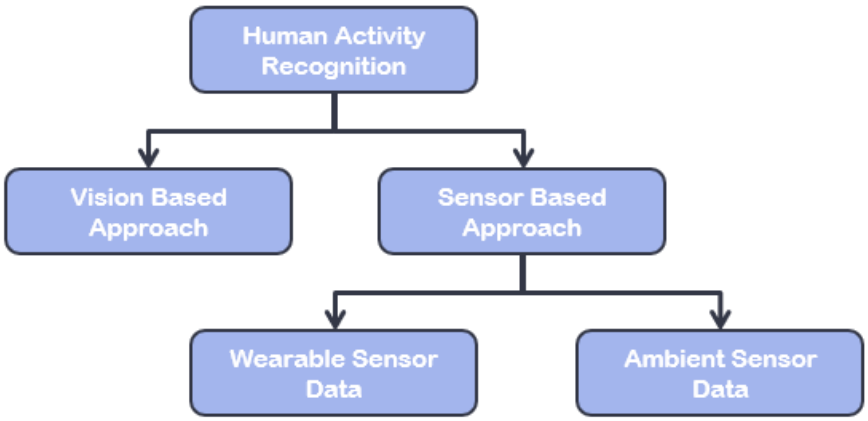


Figure 1.1: Broad Classification of Human Activity Recognition.

The first category is based on the use of visual sensing facilities, such as video cameras, to monitor an actors behavior and changes in the environment. The generated sensor data are video sequences or digitized visual data. The approaches in this category exploit computer vision techniques, including feature extraction, structural modeling, movement segmentation, action extraction, and movement tracking to analyze visual observations for pattern recognition [8].

The second category has two classes, wearable and ambient sensors. the Ambient sensors are designed to detect and measure various environmental parameters of the

area. These sensors can include a variety of different types of sensors, such as temperature sensors, humidity sensors, light sensors, motion sensors, and sound sensors, among others [44].

the Wearable sensors are based on the use of real-time signals from one or multiple wearable inertial sensors are used to recognize human actions. Sensors such as accelerometers, gyroscopes, and others are used in smart devices (often smartphones). Activity recognition is useful because it allows individuals to track their daily activities. Mobile phones have become an integral part of our lives and are equipped with sensors that can monitor people's movements and helps us recognize activities such as walking, sleeping, running, sitting, going downstairs, upstairs, etc [4].

In systems based on external or visual sensors, devices are installed at fixed locations where activity recognition is limited in the camera viewing angle. Two similar activities may not be detectable when we use the camera, privacy is also violated, also one significant limitation of cameras is their fixed position, which limits their field of view and restricts their ability to capture a complete scene [50]. The various benefits that wearable sensors offers it's a small gadgets that users can take around with them while going about their regular activities for the cost reduction, placing multiple sensors in the body can certainly lead to more accurate and efficient use of the signal in the human body and also could collect a human's bodily movement signals at any time and from any location [10]. Wearable technology appears to be a practical step towards achieving the goal of monitoring patients at home [35].

1.2 Research Problem

In healthcare applications for human activity recognition, supervising for patients in hospital settings or elderly individuals. Within this context, there are several specific research challenges related to the development of effective Human Activity Recognition systems. These challenges encompass issues such as selecting the attributes and metrics in this process [55], minimizing false alarms through picking the right algorithm to perform predictions is crucial for a successful outcome, in order to provide

timely alerts to healthcare providers, HAR systems must be able to process sensor data quickly and accurately in real-time. Developing systems that can process sensor data in real-time is an important research problem and managing ethical considerations such as privacy and autonomy concerns [22], researchers need to investigate methods to ensure minimizing the risk of data breaches and misuse. By addressing these challenges, researchers can help create Human Activity Recognition systems that are more accurate, reliable and useful in reducing the occurrence and negative impacts of falls in healthcare settings.

1.3 Research Objectives

Controlling and supervising patients in hospitals or elderly individuals requires careful attention to their activities. Addressing this challenge in one hand, to develope algorithms and models to identify and classify diverse human actions, and in other hand, addressing real-time processing challenges to provide alerts and notifications to supervisors and healthcare staff.

In this study, our primary objective is to design a deep learning architecture that achieves high levels of precision, robustness, and efficiency. This architecture should be applicable to a broad range of activity types and environmental conditions. To accomplish this, we focus on creating and evaluating machine learning algorithms, deep learning models and other techniques for the extraction and selection of features for activity classification. Additionally, we aim to propose real-time processing techniques capable of swiftly and accurately recognizing different activities. These techniques will provide instantaneous feedback or alerts to users. To achieve this, developing a proficient architecture capable of collecting and processing large quantities of sensor data with precision is a must. Furthermore, we integrate the proposed model into an API, which includes a user interface for presenting the results in a comprehensive and user-friendly format.

1.4 Research Methodology

our reserch methodology employs a systematic approach to achieve the objectives set in the present study, which involves the development and evaluation of a novel deep learning model capable of accurately recognizing Human Activities. We carefully selected benchmark datasets and performed data preprocessing to ensure data quality. Our unique model was then trained, evaluated and compared with existing approaches using appropriate metrics. Experimental setup and statistical analysis were conducted to ensure robustness and validate the models performance. To further enhance the practical applicability of our research, the proposed model was exposed as an API than integrated to a complete platform that enables real-time processing and supervision of elderly individuals or patients.
By accomplishing these objectives, we aim to enhance the control, care, and well-being of patients and elderly individuals in healthcare environments.

1.5 Thesis Outline

Chapter 1 Provides an introduction to the thesis. It begins by discussing Human Activity Recognition and the various branches and applications within HAR. The chapter then highlights the research problems and challenges faced in the field of HAR. The objectives of the thesis are clearly stated, outlining the specific goals to be achieved. The chapter concludes with a description of the research methodology employed, from the datasets used to the analysis techniques used throughout the study.

Chapter 2 Describes a comprehensive background study for the research, The chapter delves into theoretical foundations, and key concepts that form the basis of the research. we provide insights into related work, shedding light on the advancements and research conducted in this field. Establishing a solid foundation and understanding of the subject matter, paving the way for the subsequent chapters of the thesis.

Chapter 3 Provides an overview of the methodology used in the thesis, outlining the steps and experiments designed to address the research questions. It defines the methods, libraries, and tools employed to test the hypothesis and presents the metrics necessary to evaluate the experimental results as well as a comparison between the other approaches. The chapter serves as a road-map for the implementation of the research methodology, aiming to answer the research questions effectively.

Chapter 4 Explores the practical implementation of the Junction Layer and the development of a web application, two integral components of the HAR system. The chapter begins with an overview of the environment and development tools used. It then focuses on the detailed implementation and simulation of the Junction Layer, showcasing its role in processing and analyzing human activity data. The development of the web application is also discussed, highlighting its user-facing features and real-time insights.

Chapter 2

Artificial Intelligence

The Artistry of Intelligence: Painting the Future with Neural Networks and Deep Learning

2.1 Introduction

Artificial Intelligence (AI) is a big area of research that involves making machines able to do things that usually only humans can do, like visual perception, information processing, automated programming, robotics, scenario analysis, game playing, intelligent systems and scientific theorem proving, AI systems are programmed to learn and get better as they encounter more situations, which means they can handle new challenges and solve difficult tasks with great precision [26]. Various computer systems have significantly improved the quality of human life by providing an array of tools and devices that significantly reduce the physical and mental effort required to accomplish a wide range of tasks. Artificial intelligence represents the next phase of this evolution, one that promises to further augment these benefits by incorporating a suite of rational, analytical, and more efficient technologies that will enable us to achieve unprecedented levels of effectiveness and efficiency in both our personal and professional lives [24].

AI includes methods like Machine Learning (ML), which aims to design algorithms that enable machines to learn from data without being programmed step-by-step. In fact, ML algorithms use various techniques to find patterns in data, so they can predict outcomes or make decisions. ML has been very useful in many areas, such as computer vision (CV), speech recognition, natural language processing (NLP), fraud detection and recommendation systems. However, ML methods need a lot of work in the early stages to choose the right information from the data, so they can achieve good results [52].

On the other hand, Deep Learning (DL) is a type of machine learning that uses artificial neural networks (ANN) which are inspired by the human brain's working mechanism, to automatically extract features from data, without the need for human intervention. DL methods are utilized to model complex patterns in data, it takes a set of data as an input and transforms it through many layers, This involves machine learning, where machines learn and develop skills through analyzing data and past experiences [28]. A regular neural network usually has one or two hidden layers,

whereas a deep neural network can have many hidden layers. Each hidden layer in a deep learning neural network is responsible for training a specific set of features, based on how well the previous layer performed. As the number of hidden layers increases, so does the complexity and abstraction of the data. This allows the deep learning algorithm to solve more complex problems that involve a large number of nonlinear transformations, which would be impossible for a human to solve.

Overall, AI, ML and DL (Figure 2.1), are all rapidly evolving fields with numerous applications in various industries, including healthcare, finance, transportation and more. As these technologies continue to advance, they have the potential to revolutionize the way we live and work. Also there are those who think that if AI keeps progressing as quickly as it is, it could be dangerous for humans. On the other hand, some people believe that unlike previous technological advances, AI would lead to a lot of people being out of work [24].

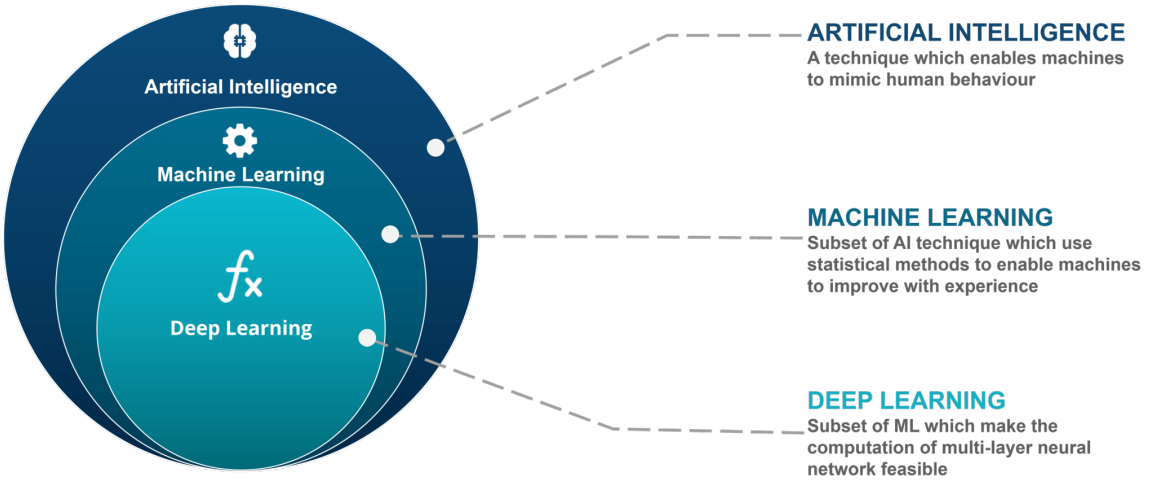


Figure 2.1: The Relation Between AI, ML and Deep Learning [3].

2.2 Artificial Neural Network (ANN)

Artificial Neural Network (ANN) is a computer program that mimics the way the human brain works. It's also called a Simulated Neural Network (SNN). Figure 2.2 shows the basic structure of ANN. This type of machine learning is used for recognizing patterns and sorting data, and it's essential in the field of deep learning. ANNs have been successfully used in various fields, including finance, computer vision, speech recognition and natural language processing. Deep learning relies heavily on ANNs, which are built using layers of neurons to learn and classify data in a hierarchical manner. They are also a fundamental component of deep learning, which uses multiple layers of neurons to learn hierarchical representations of input data. ANN applies several techniques during its execution process, including supervised learning, unsupervised learning, or reinforcement learning. Currently, research on neural network learning algorithms is still being explored and accepted by communities [14].

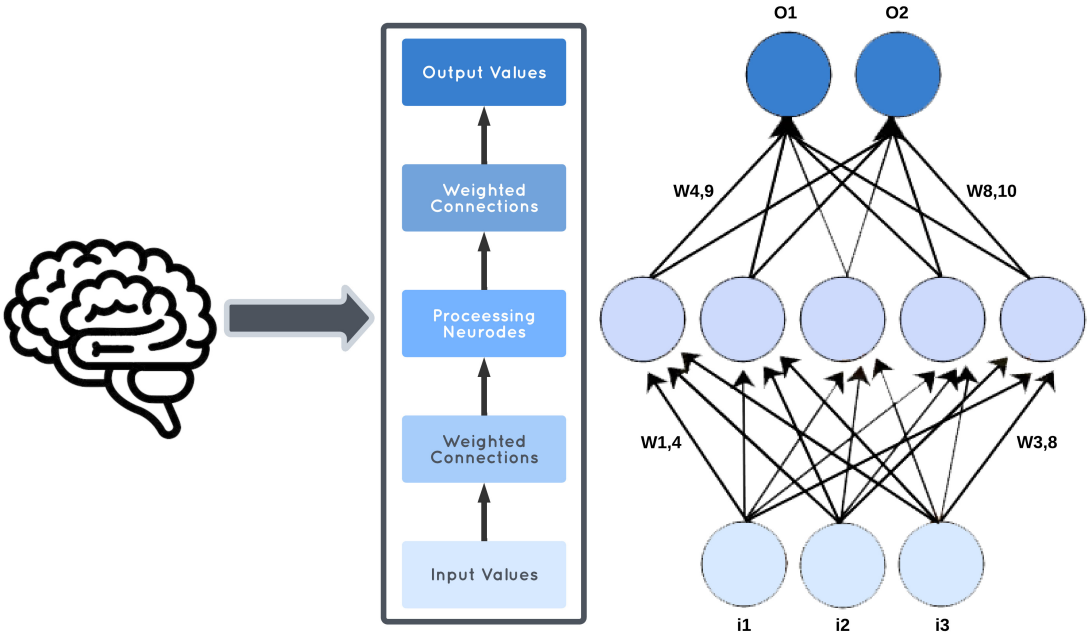


Figure 2.2: ANN's Basic Structure.

2.2.1 Components Of ANN

2.2.1.1 Artificial Neuron

Artificial neurons are essential components of ANNs. The neurons process inputs and pass on outputs to other nodes in the network, making them vital and indispensable component within the realm of artificial neural networks [62].

2.2.1.2 Parameters

The parameters of an ANN play a crucial role in determining its behavior and performance. These parameters include the weights, biases and others associated with the connections between neurons, (Figure 2.3) shows the details of each parameter.

2.2.1.3 Activation Function

In Artificial Neural Networks, the activation function is a mathematical function that is applied to the output of each neuron in the network. The purpose of the activation function is to introduce nonlinearity into the output of the neuron, allowing the network to model complex, nonlinear relationships between inputs and outputs [36]. (Figure 2.4) shows different types of activation functions used in ANNs, followed by their right Formula (2.1 to 2.4).

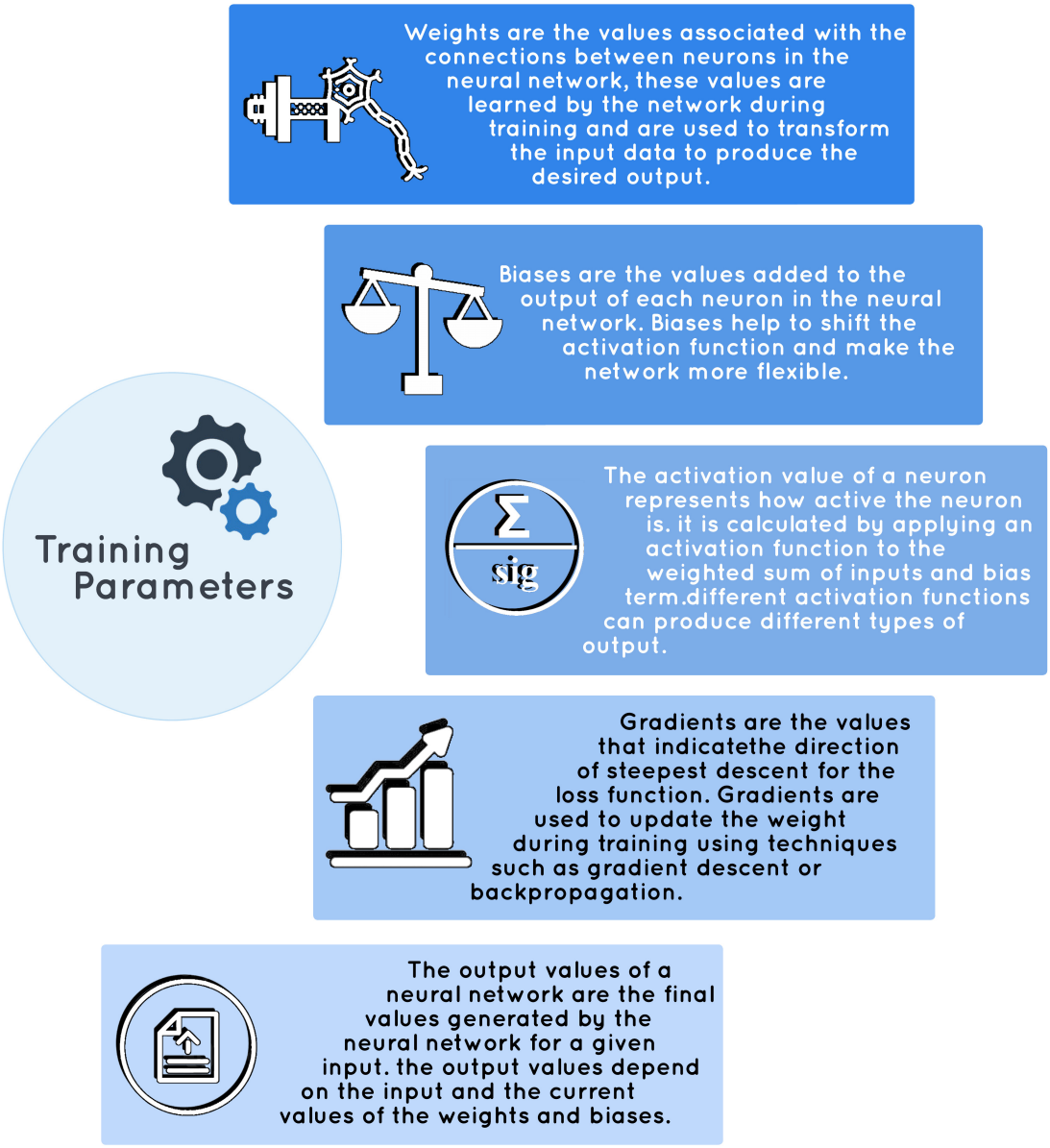


Figure 2.3: Different Training Parameters of ANN.

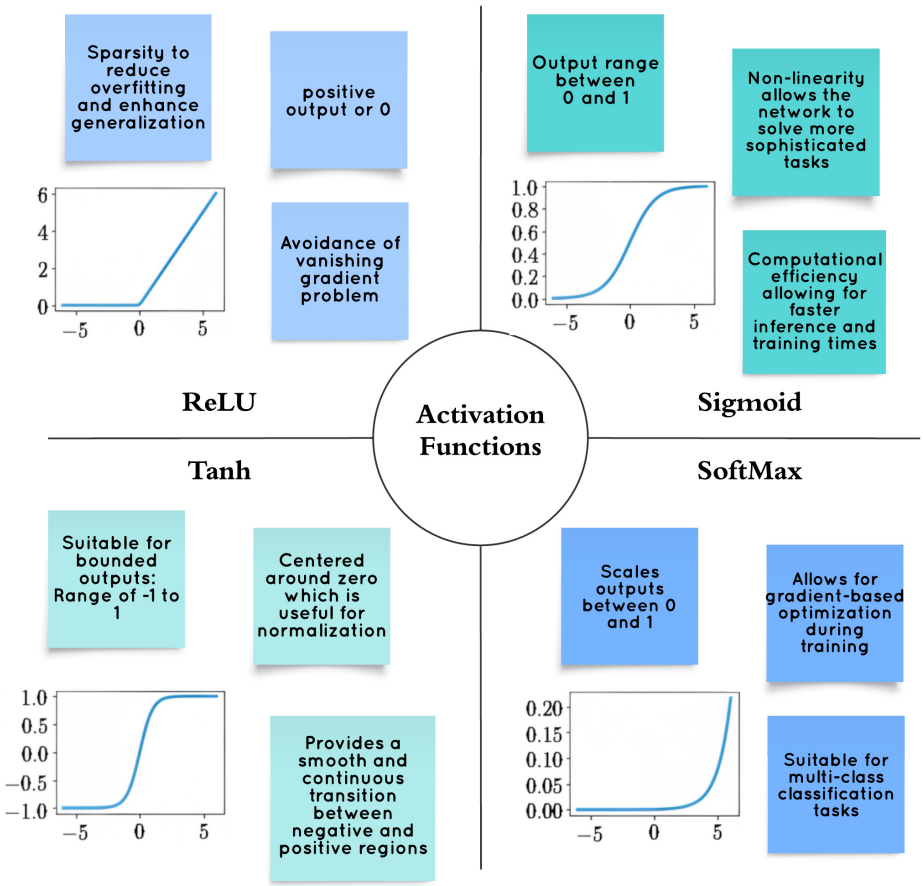


Figure 2.4: Different Activation Functions.

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

$$ReLU(z) = \max(0, z) \tag{2.2}$$

$$Tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.3}$$

$$Softmax(\mathbf{z_i}) = \frac{e^{z_i}}{\sum j = 1^{K} e^{z_j}} \tag{2.4}$$

2.2.1.4 Layers:

An ANN is composed of several interconnected layers of nodes that carry out specific computations while processing data. Within a neural network, each neuron receives input from neurons in the preceding layer, processes the input data in a unique way, and transmits the output to the next layer's neuron [48][34]. ANN contains three types of layers shown in (Figure 2.5).

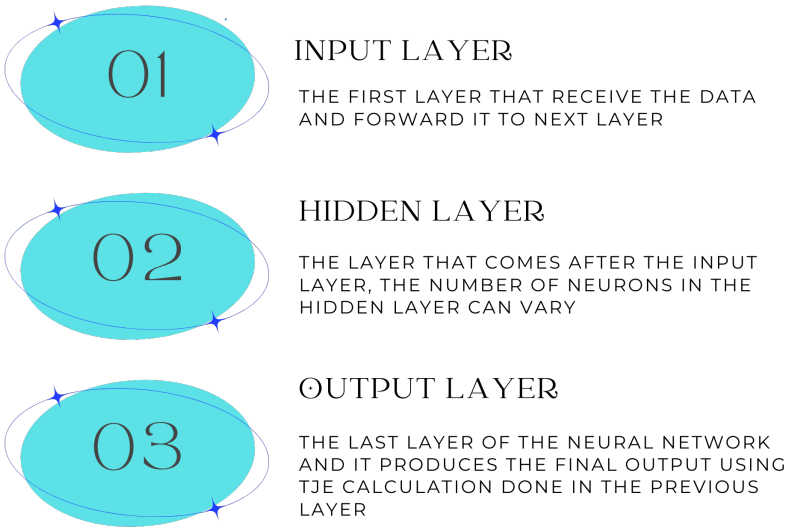


Figure 2.5: Different Types of ANN Layers.

A neural network structure is created by combining these layers, in which each neuron in a layer is linked to every neuron in the preceding layer and the following layer, except for the neurons in the input layer that only obtain input data. The strength of each connection between neurons is determined by an associated weight, and these weights are modified during training to minimize the loss function and improve the network's performance [48].

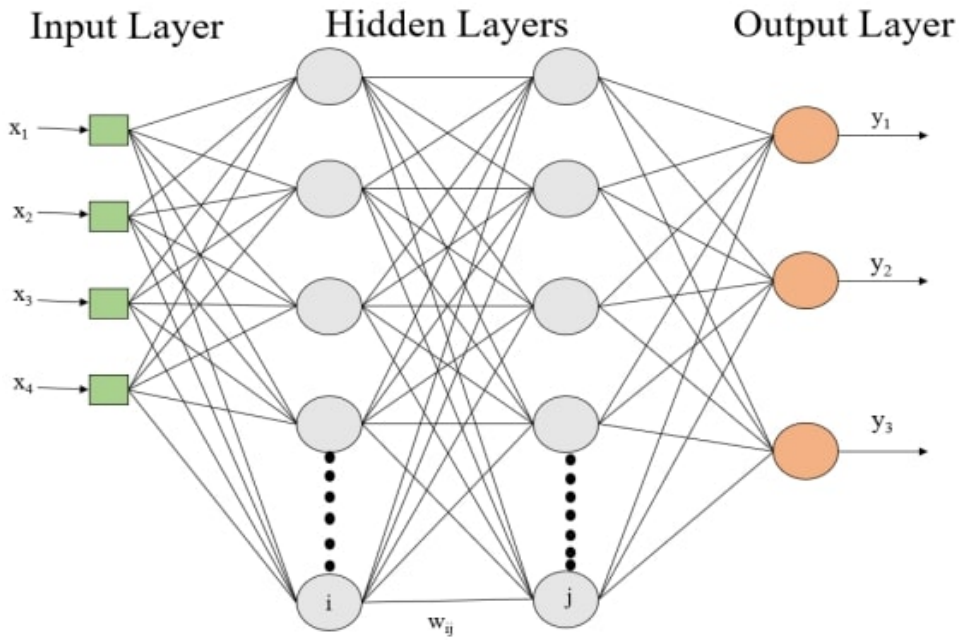


Figure 2.6: General Structure of Artificial Neural Network [48].

2.2.2 ANN Architecture HyperParameters

Hyperparameters are important settings in deep learning that are manually set by the user in order to determine how the model is trained, They can have a significant impact on improving the performance of the model [27].

2.2.2.1 Network Hyper parameters

Network hyperparameters are a type of hyperparameter that determine the architecture of a neural network. Some of the major hyperparameters in deep learning shwon and explained in (Figure 2.7).

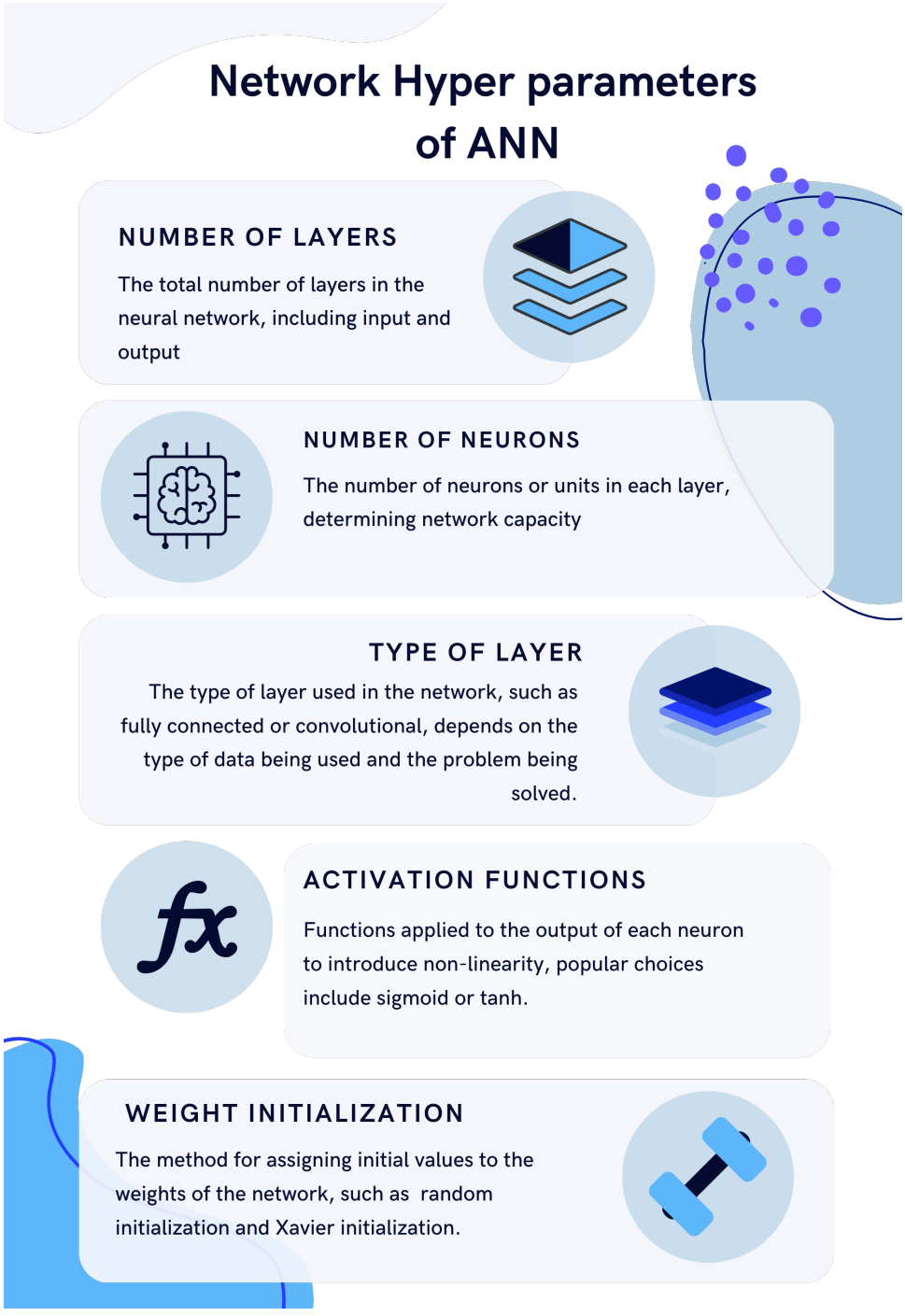


Figure 2.7: Different Network HyperParameters.

Choosing the right network hyperparameters is essential for building an effective neural network. the choice of hyperparameters can affect the speed and accuracy of training, the generalization ability of the network, and its ability to handle new data. grid search and random search are commonly used methods for finding optimal network hyperparameters.

2.2.2.2 Training Hyper Parameter

Training hyperparameters are a set of parameters that define how a neural network is trained or optimized during the learning process. These hyperparameters directly impact the training dynamics and convergence of the model. They include parameters such as the learning rate, batch size, number of epochs and optimizer choice as shown in (Figure 2.10).

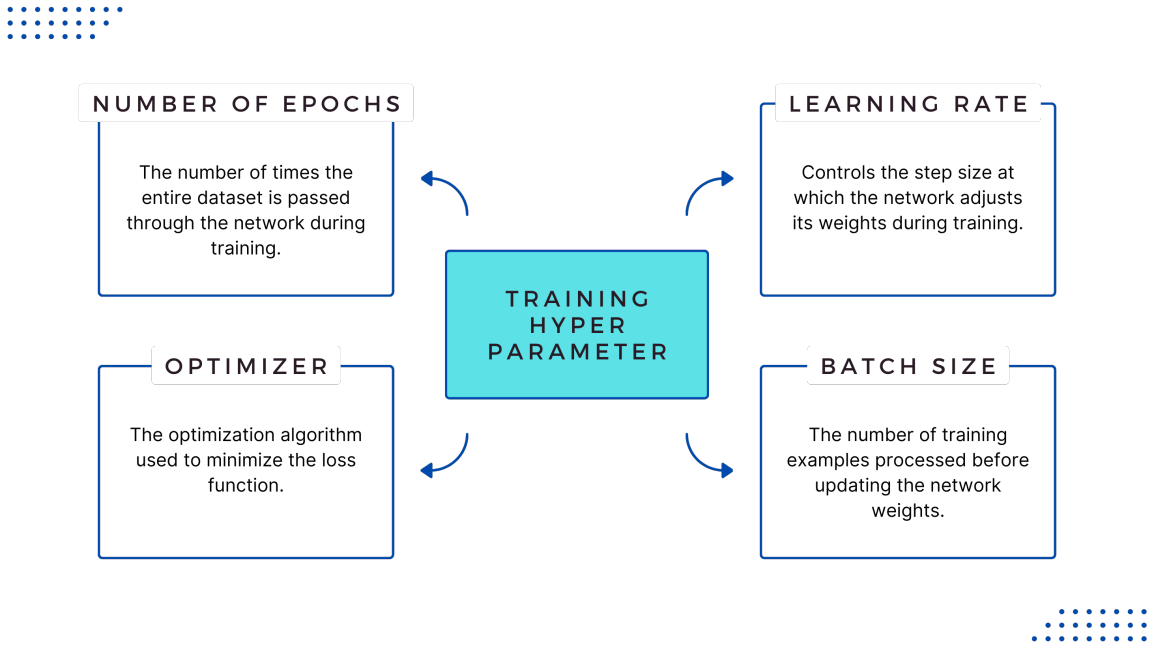


Figure 2.8: Different Training HyperParameters.

Properly tuning these training hyperparameters is essential for achieving optimal model performance and preventing issues like overfitting or underfitting. It often

requires experimentation and fine-tuning to find the best combination of hyperparameter values for a specific task or dataset.

2.2.3 Training Phase in ANN

Training is the process of using a dataset to teach a deep learning model how to predict or classify new, unseen data. During training, the model learns to recognize patterns and relationships in the training data by adjusting its parameters, or weights. The goal of training is to minimize the deviation between the model's predictions and the actual values in the training data.
The process of training a deep learning model typically involves several steps shown in (Figure 2.10).

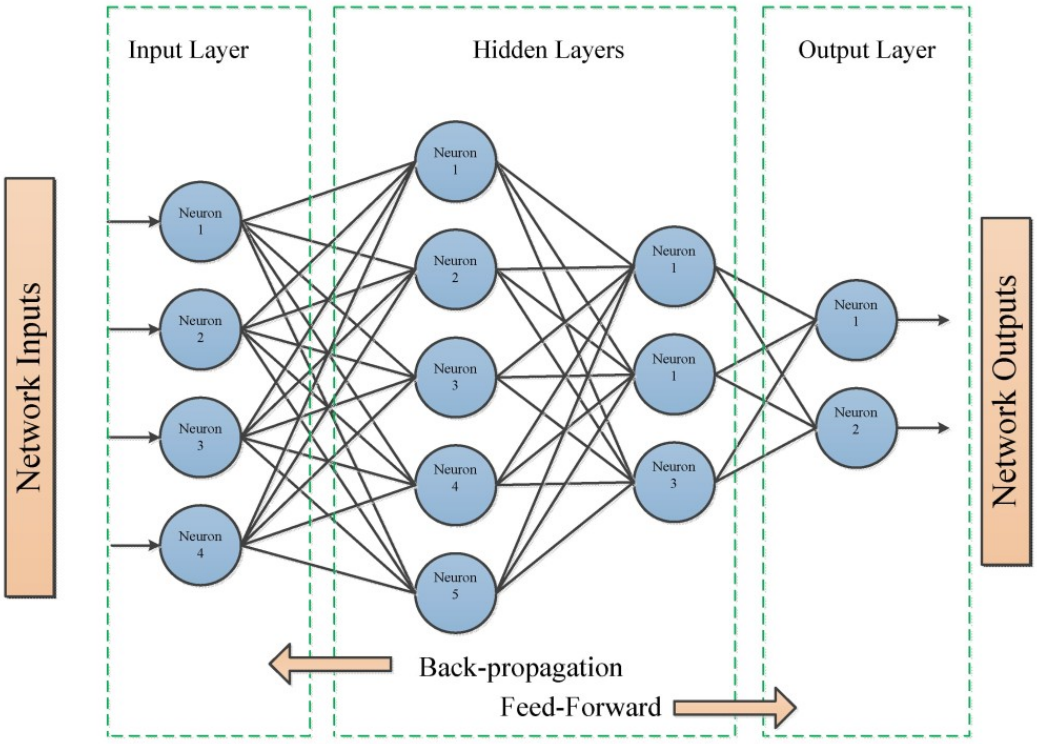


Figure 2.9: Feed forward back propagation mechanism in artificial neural network [2].

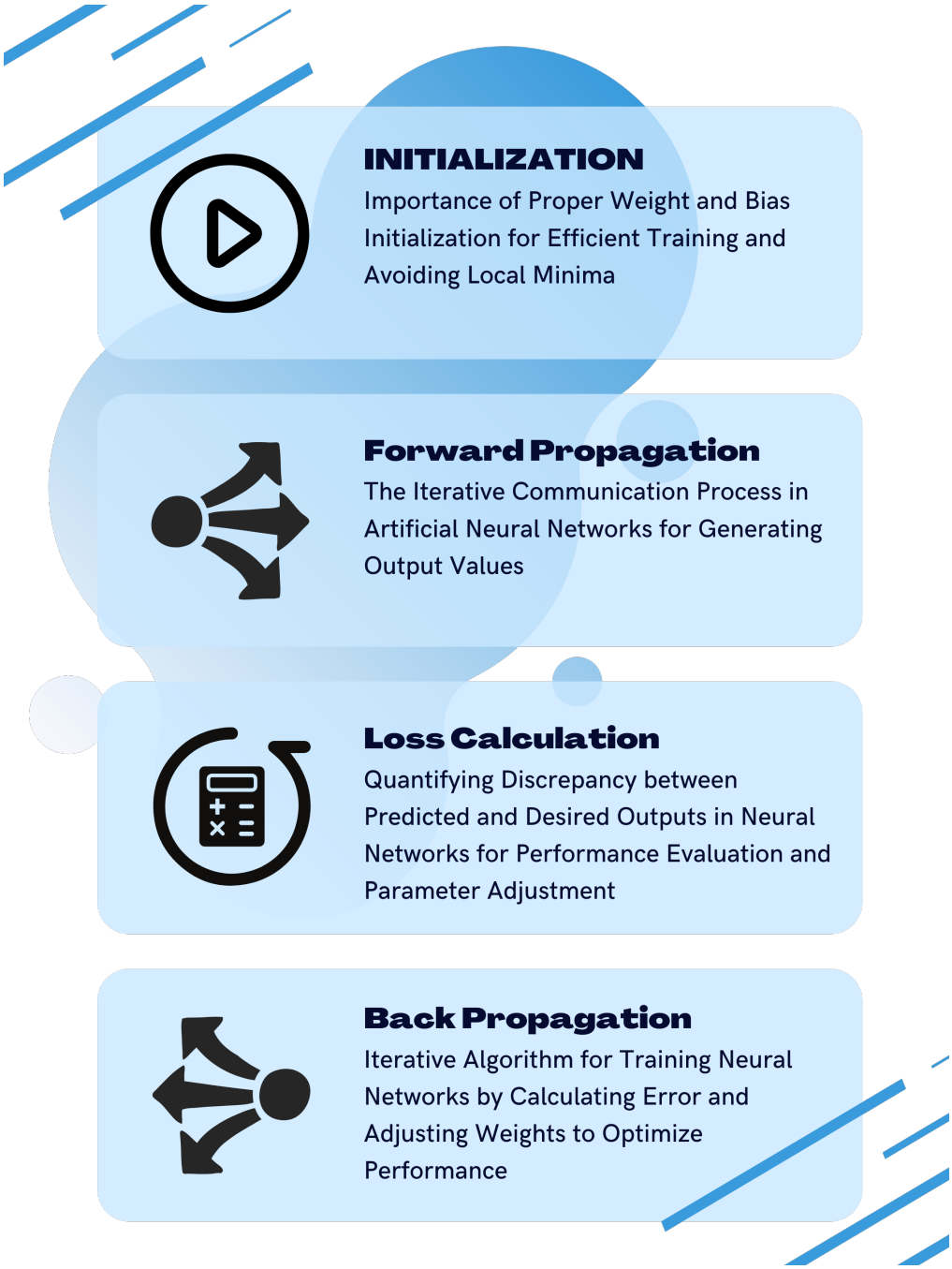


Figure 2.10: Essential Steps in Neural Network Training.

The process of training a machine learning model can be time-consuming and computationally intensive, especially for large datasets and complex models. However, a well-trained model possesses the ability to make accurate predictions on new, unseen data, which is essential for various modern applications.

2.2.4 Challenges of Training

Over-fitting: Is a significant issue in training neural networks, occurring when the model becomes overly familiar with the training data and begins to memorize the noise and outliers present instead of understanding the fundamental patterns. This phenomenon indicates that the neural network reaches a point during training where it no longer enhances its problem-solving capability. It can be observed that the error on the test set reaches its lowest point, indicating the network's optimal generalization ability, after which the error begins to rise again [25].

Under-fitting: Occurs when a model fails to capture the data's variations and fundamental patterns. It is the opposite of over-fitting and typically happens when the model is too simple or when the training data is complex. Underfit models have poor performance on both training and unseen data, with high bias and low variance. They oversimplify the problem and struggle to capture the data intricacies, leading to limited predictive capabilities [25].

2.2.5 Types of Artificial Neural Networks

In artificial neural networks, there are many different designs, each intended for a specific type of task. One of the fundamental structures is the deep neural network (DNN), and many other structures are developed based on its principles. The following Figure 2.11 illustrates different types of ANN structures.

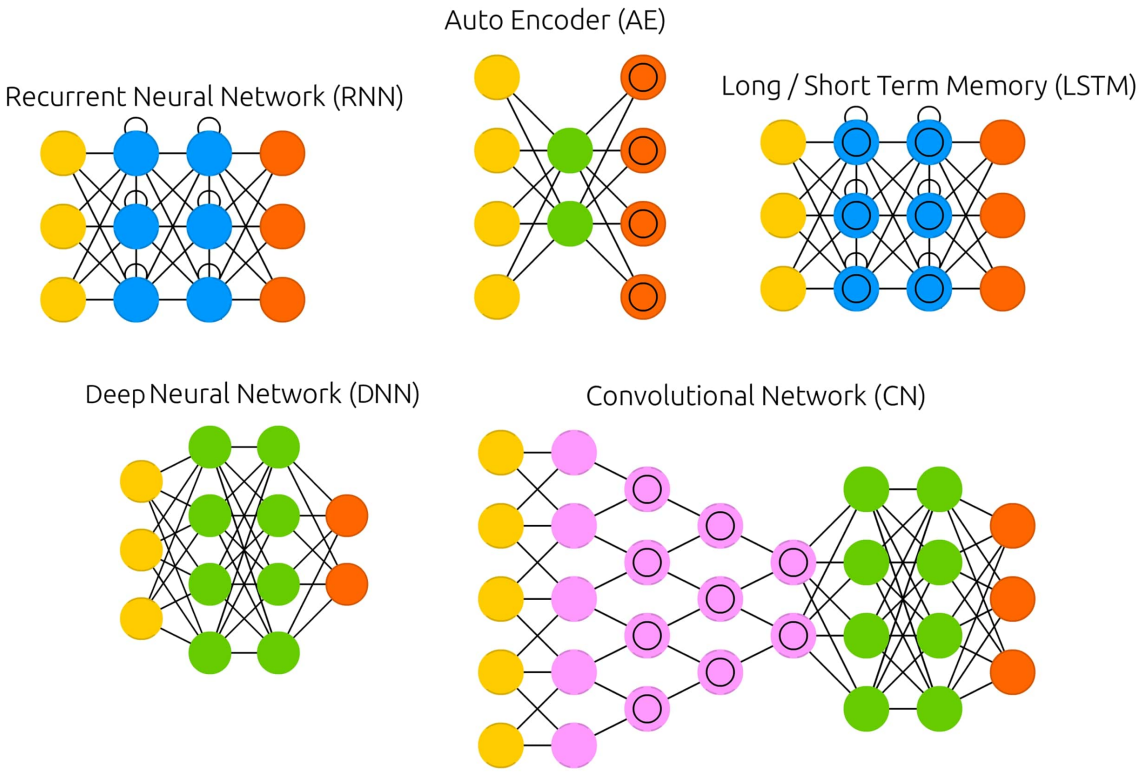


Figure 2.11: Different Types of ANN Architecture.

2.2.5.1 Deep Neural Network (DNN)

A deep neural network (DNN) is a special type of ANN that has many hidden layers situated between the input and output layers as shown in (Figure 2.12). Because of their depth, they are able to learn more complex patterns in data, resulting in better performance than other machine learning methods. Despite being designed in the early 1980s, DNNs have only gained significant attention from researchers and industry experts in recent years, largely due to advancements in computational resources and training techniques [30][19]. DNNs are now widely used in various areas, such as speech recognition, image recognition and machine translation. They are also a fundamental technology in the field of deep learning.

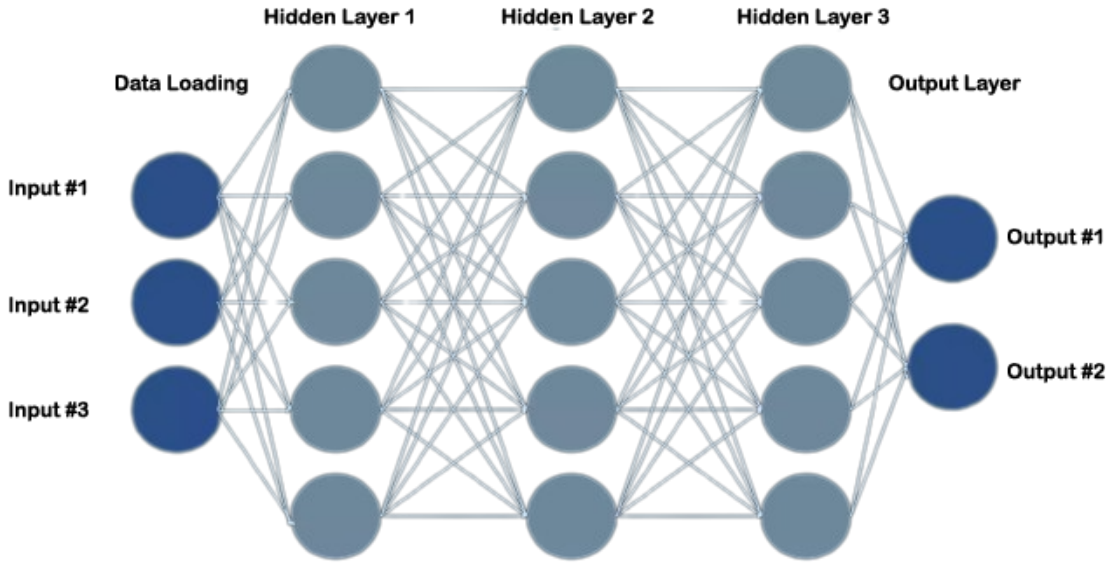


Figure 2.12: Architecture Of Deep Neural Network.

2.2.5.2 Convolutional Neural Networks (CNN)

Convolutional neural networks (CNN) are a type of artificial neural network. They are specifically designed to analyze and process image data and are commonly used for tasks such as image recognition and processing [7][30]. The architecture of a CNN includes several important components such as pooling layers, fully connected layers and convolutional layers, which work together to identify the most significant features in the image. Besides image processing, CNNs are effective in handling speech or time series signal inputs [38]. CNNs are particularly useful because they can automatically learn features from raw data, making them a powerful tool in computer vision and machine learning.

The convolutional layer, applies learnable filters to extract features from the input data, enabling the network to recognize lines, colors, and changes in brightness see (Figure 2.13), the pooling layer downsamples the input, reducing parameters and spatial dimensions. Two common pooling methods are max-pooling and average pooling as shown in Figure 2.14 and the fully-connected layers integrate information and

normalize the output using a softmax activation function. These layers collectively enable the CNN to extract features and make accurate predictions.

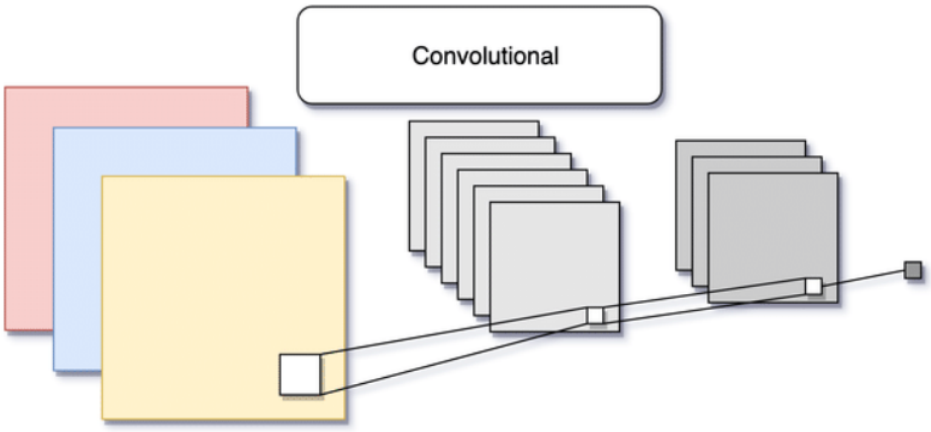


Figure 2.13: Illustration of Convolutional Neural Network [29].

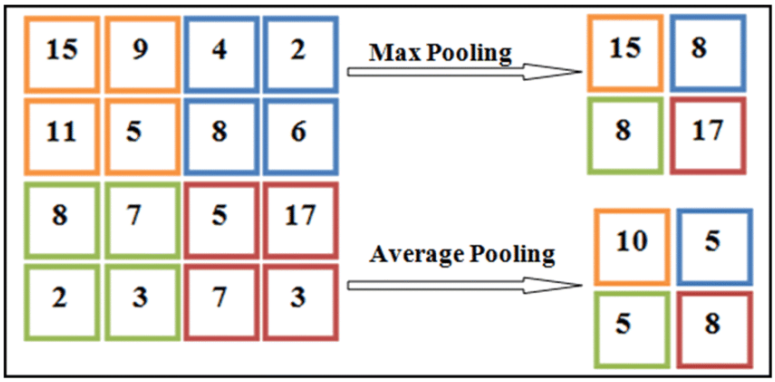


Figure 2.14: Difference Between Max Pooling And Average Pooling [43].

2.2.5.3 Recurrent Neural Networks (RNN)

Recurrent neural networks (RNN) are a type of deep learning network that uses loops to allow information to persist (Figure 2.15). This gives them the ability to remember

previous data and use it to predict the next value in a sequence. They are particularly useful in recognizing patterns in sequential data, such as predicting the next item in a series or sequence. Unlike traditional neural networks, RNNs have a memory component that allows them to make use of previous calculations [47]. RNNs are a popular choice for dealing with data that comes in sequences. But, RNNs can be tough to train because of two issues: the gradient vanishing and exploding problems. These issues make it difficult for the network to learn patterns that are spread out over a long period of time [32]. RNNs are used in various applications, including speech recognition, machine translation, and analyzing stock market trends

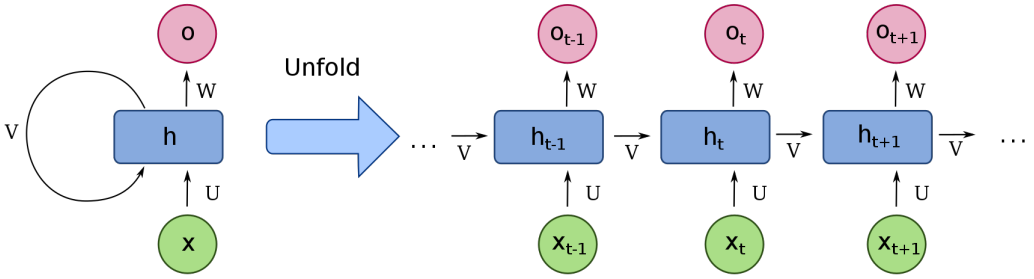


Figure 2.15: Schema of RNN.

2.2.5.4 Long Short-Term Memory (LSTM)

Long short-term memory (LSTM) networks are a type of RNN that can avoid common issues related to error back-flow. Traditional RNNs may have trouble training due to problems like gradient vanishing and exploding, which makes it hard to learn long-term patterns. LSTMs solve this problem by allowing the network to remember important information from earlier steps and forget irrelevant information, making it possible to detect long-term patterns. The architecture of LSTM shown in (Figure 2.16) They are particularly useful for processing noisy and complex input sequences and can be trained using gradient-based learning algorithms [20][11].

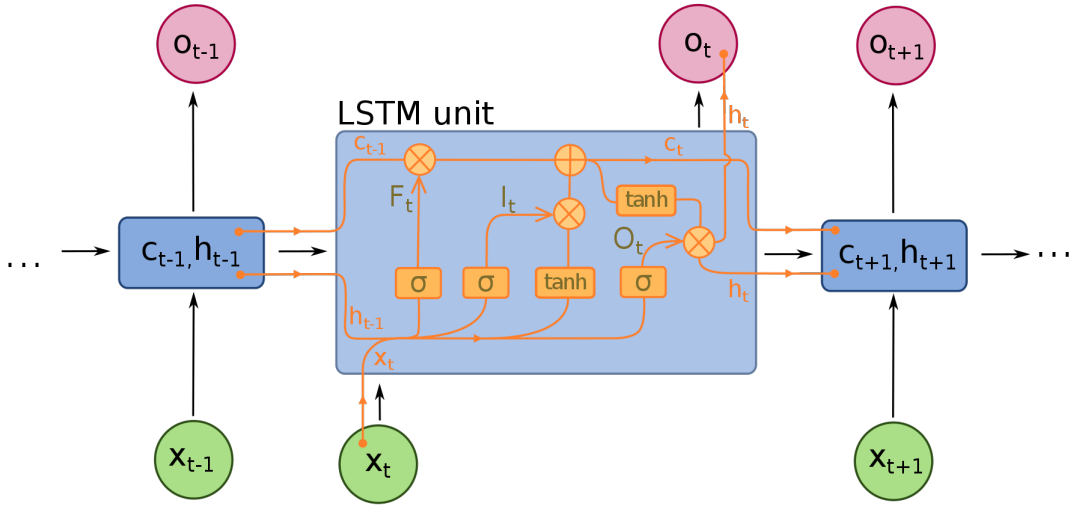


Figure 2.16: Architecture of LSTM.

2.2.5.5 Autoencoder

Autoencoders are a special kind of neural networks that do not require any labels and are part of the unsupervised learning category [9], the Autoencoders typically used for reducing the dimensionality of data, the goal is to regenerate the input at the output stage, with the network being trained to do so. by removing redundant information. the network is trained to replicate the input at the output layer [26]. There are three components to make an Autoencoder: the encoder function, the code and the decoder function (Figure 2.17). The encoder function takes the input data and converts it into a compressed representation in a low-dimensional space, producing the code. During this transformation, the data gets compressed. After that, the decoder network receives the compressed representation and changes it back to its original input space [56].

2.2.5.6 Transfer Learning

Transfer learning is a technique that allows pre-trained deep learning models to be used with different data distributions see (Figure 2.18) [39]. The use of transfer

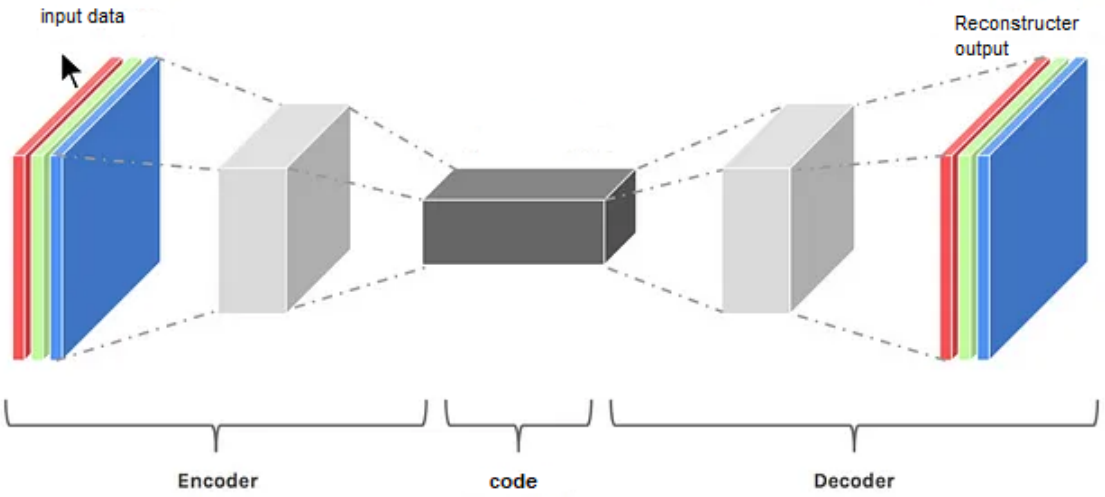


Figure 2.17: Presenting Architecture Of Autoencoder [54].

learning in time series classification and activity recognition has been reviewed in different studies. Recent advances in transfer learning for deep learning have also improved adaptation to new settings, including the development of meta-learning algorithms. These algorithms train a model on various learning tasks, allowing it to solve new learning tasks with only a few training samples [5][46].

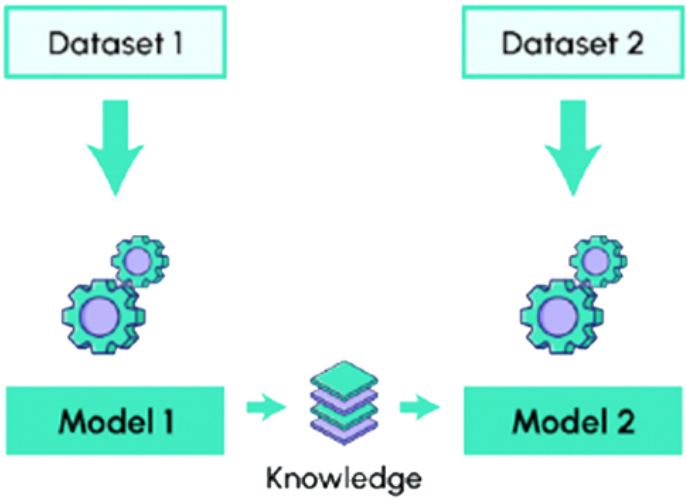


Figure 2.18: Transfer Learning Concept.

2.3 Deep Learning for HAR

There are some existing review papers in the literature for deep learning approaches for sensor-based human activity recognition.

- Haojie Ma et al. [33] in 2019 proposed a new type of neural network model called AttnSense. this model combines a type of attention mechanism with a structure that uses both convolutional neural networks and recurrent neural networks. by doing this, the model is able to capture information about how the sensor data is related in both spatial and temporal domains. this results in improved comprehensibility and prioritized sensor selection. the paper evaluates the proposed AttnSense model on three publicly available HAR datasets: Heterogeneous, Skoda, and PAMAP2. AttnSense outperforms several state-of-the-art methods such as Random forest (RF), SVM, CNN, LSTM, DeepConvLSTM, and DeepSense on all three datasets.

- Qi Tenget et al. [53] in 2020 introduced a new layer-wise convolutional neural network (CNN) model for sensor-based HAR tasks. the model uses a combination of similarity matching loss and prediction loss, which is denoted as predsim loss, in a local error model algorithm for activity recognition. this approach enables training layers one-at-a-time, reducing the number of parameters from 5.81M to 0.55M by using local loss and improving memory efficiency compared to traditional whole-network training methods. the backward gradient flow is detached from the computation graph to prevent the use of local loss. the proposed method was evaluated on five public benchmark datasets: UCI HAR, Opportunity, UniMib-SHAR, PAMAP2, and WISDM. the results show that the layer-wise CNN with local loss is effective in improving classification performance on various HAR datasets compared to traditional whole-network training methods or other state-of-the-art approaches. the proposed method achieved high accuracy on each dataset, outperforming the baseline model and

other state-of-the-art methods.

- Jessica Sena et . [49] in 2021 Suggested a multimodal DCNN ensemble (MDE) model which is a type of deep learning model used for human activity recognition that uses sensor data. It combines data from multiple sensors and temporal scales to improve accuracy. The model has been evaluated using several datasets, including WHARF, USCHAD, UCI-HAR, OPPORTUNITY, UniMiB-SHAR, PAMAP2 and UTD-MHAD1. MDE uses an ensemble of convolutional neural networks, each with different kernel sizes to identify patterns at different time scales, and processes data from individual sensors. The approach generates diversity in feature extraction by combining scores of all streams using meta-learning for fusion. MDE achieved state-of-the-art performance on several datasets under both Leave-One-Subject-Out (LOSO) and Leave-One-Trial-Out (LOTO) protocols when compared to previous methods.

- Andrey Ignatov et al. [23] in 2018 presented a deep learning method for classifying human activities in real-time using accelerometer data from mobile devices. The model uses Convolutional Neural Networks (CNNs) for capturing local features and statistical features for extracting global characteristics of the time series. The CNN architecture comprises of convolutional layers with ReLU activation function and a max-pooling layer, followed by fully-connected layers for classification. The model does not require manual feature engineering and has good generalization, which makes it appropriate for real-time activity recognition on mobile devices. The proposed model is tested on two public datasets, WISDM and UCI HAR, and it outperforms existing solutions in the field of human activity recognition, achieving state-of-the-art results on both datasets.

- KUN XIA et al. [60] in 2020 have proposed a new deep learning model for

human activity recognition using sensors in mobile devices. The model combines convolutional layers and long short-term memory (LSTM) to automatically identify activity features and classify them using few model parameters. The raw data collected by sensors is fed into a two-layer LSTM followed by convolutional layers. To reduce the number of parameters, a global average pooling layer (GAP) is applied instead of a fully connected layer after convolution, and a batch normalization layer (BN) is added after the GAP layer to speed up convergence. The model was tested on three publicly available datasets (UCI, WISDM, and OPPORTUNITY) and showed high accuracy in all three, demonstrating its superior performance.

- Sravan Kumar Challa et al. [6] in 2022 introduced a deep neural network for human activity recognition using data collected by mobile sensors. The model combines convolutional layers with long short-term memory (LSTM) to automatically extract activity features and classify them with few model parameters. It achieved high accuracy on three public datasets (WISDM, UCI-HAR, and PAMAP2) and outperformed some reported results in terms of robustness and activity detection capability. The model consists of eight layers, including two LSTM layers for temporal feature extraction and two convolutional layers for spatial feature extraction. A global average pooling layer (GAP) replaces the fully-connected layer behind the convolutional layer to reduce model parameters while maintaining high accuracy.

- Yu Guan et al. [17] in 2017 proposed a deep learning model using Long Short Term Memory (LSTM) network, which is a type of recurrent neural network that can capture temporal dependencies of sensor data streams for human activity recognition. The LSTM network has specific gate parameters and network output layer parameters that can learn data representations and classifiers. The paper focuses on combining multiple LSTM networks into ensemble classifiers to

improve recognition performance and robustness. Modified training procedures that include epoch-wise bagging and probabilistic selection approaches are used to determine problematic data subsets. The resulting ensembles of deep LSTM learners have been shown to outperform individual LSTM networks in three standard benchmark datasets (Opportunity, PAMAP2, Skoda) for human activity recognition using wearable devices.

- The article [58] presented by Shaohua Wan et al. in 2020 discusses the use of smartphones and wearable sensors in recognizing human activities. The article proposes a real-time classification method for human activity recognition using a smartphone's inertial accelerometer-based architecture. The authors conducted experiments using two distinct datasets, namely UCI and Pamap2, and employed various algorithm models, including convolutional neural network (CNN), long short-term memory (LSTM), bidirectional LSTM (BLSTM), multilayer perceptron (MLP), and support vector machine (SVM). Among the different models tested, the proposed CNN model showcased superior performance compared to traditional approaches, demonstrating state-of-the-art results on both datasets. The CNN model leverages convolutional layers and pooling layers to extract meaningful features from the data. Subsequently, a fully connected layer computes scores for each class, enabling the generation of accurate classification results.

- Saedeh Abbaspour et al. [1] in 2020 conducted a research study that explores the integration of convolutional neural networks (CNNs) with recurrent neural networks (RNNs) for human activity recognition. The article discusses four hybrid models that combine CNNs with four powerful RNN variants, namely LSTMs, BiLSTMs, GRUs, and BiGRUs. The purpose of integrating CNNs with RNNs is to leverage the strengths of both architectures in human activity recognition tasks. CNNs excel in extracting relevant features from sensor

data by capturing local dependencies and scale invariance. They are capable of extracting spatial and temporal dependencies from the sensor data. Pooling layers in CNNs play a crucial role in reducing dimensionality and extracting dominant features. On the other hand, RNNs, such as LSTMs, are effective in modeling long-term dependencies by incorporating an internal memory mechanism. BiLSTMs, which consist of two LSTMs trained on data sequences in normal and reverse time order, further enhance the model's ability to capture temporal dependencies. GRUs address short-term memory challenges in RNNs by utilizing two gates but do not include a cell state. BiGRUs, similar to BiLSTMs, combine two GRUs that process data in normal and reverse time order. The proposed hybrid models aim to capitalize on the advantages of both CNNs and RNNs. By utilizing CNNs for feature extraction and RNNs for modeling temporal dependencies, the hybrid models achieve higher accuracy rates compared to traditional machine learning models in human activity recognition tasks. The integration of CNNs and RNNs proves beneficial in enhancing the accuracy and performance of the models for recognizing and classifying human activities.

- Nidhi Dua et al. [13] in 2021 proposed a research paper that discusses various models used for HAR and proposes a multi-input CNN-GRU model for HAR. The model uses convolutional layers for feature extraction and Gated Recurrent Unit (GRU) layers for time series data classification, and the proposed model outperformed other architectures in terms of classification performance. Overall, the results indicate that the use of GRU layers in the proposed model makes it superior when compared to its LSTM counterpart, especially on datasets containing various features such as those used in this study.

2.4 Conclusion

Chapter 2 serves as a comprehensive introduction to artificial neural networks and their essential components, necessary for developing well-trained models capable of accurate predictions on unseen data. Additionally, the chapter explores various types of artificial neural networks commonly used in applications, such as LSTM, CNN and Autoencoder, also encompasses a collection of articles that explore the application of deep learning techniques specifically for HAR. The relevance of deep learning in the context of HAR is emphasized, paving the way for further exploration of deep learning techniques in HAR. By providing a solid foundation of knowledge, Chapter 2 sets the stage for the subsequent chapters' in depth exploration of deep learning methodologies in the field of HAR.

Chapter 3

Conception

Painting Methodological Masterpieces:
Exploring Experiments and
Evaluative Brushstrokes

3.1 Introduction

In this chapter, we present the Holistic Structure that underpins our research on Human Activity Recognition. Our architecture encompasses a systematic arrangement of interconnected components, with a primary focus on the creation and integration of our powerful deep learning model. This model serves as the cornerstone of our research, enabling accurate recognition and real-time monitoring of human activities. We commence by mentioning the datasets selected. Next, we delve into the integration of an Application Programming Interface (API) that facilitates efficient processing and analysis of human activities using the deep learning model. Additionally, our architecture includes a developed platform for real-time result presentation, empowering nurses and healthcare professionals with prompt monitoring capabilities. Throughout this chapter, we provide a detailed exposition of each component, highlighting their functionalities, interactions, and significance in achieving accurate recognition and real-time monitoring.

3.2 Overall Architecture

Our System Architecture follows a systematic approach to accomplish the defined objectives in this study, which encompasses the creation and assessment of an innovative deep learning model with the ability to accurately identify human activities and its use for detecting patient activities in real time. The overall architecture of our solution (see Figure 3.1) revolves around our proposed deep learning model delivered as an API, which serves as the centerpiece of our architecture. Our solution ensures a seamless workflow from data input to result presentation. It consists of several key components working together to accomplish the desired outcome. Firstly, we employ a junction layer, which acts as middleware for the entire system. The data flow begins by feeding data from sensors implanted on the patient to the junction layer. Serving as a bridge, the junction layer role is to transfer data to the AI API that implements our deep learning model. The AI API handles the received data

and processes it using our deep learning model to recognize patient activities. Once the model has processed the data, the API sends the prediction back to the junction layer. From there, the junction layer transfers the results after analysing the prediction sent by the AI API with the other collected data (patient position in our case) to the user interface. This user interface display the results and presents them in a user-friendly manner, we refer that in addition to its function as middle-ware ensuring connectivity services, the junction layer ensure in our case the simulation of the physical system (the patient, sensors implanted on it and its position)(Figure 3.2).

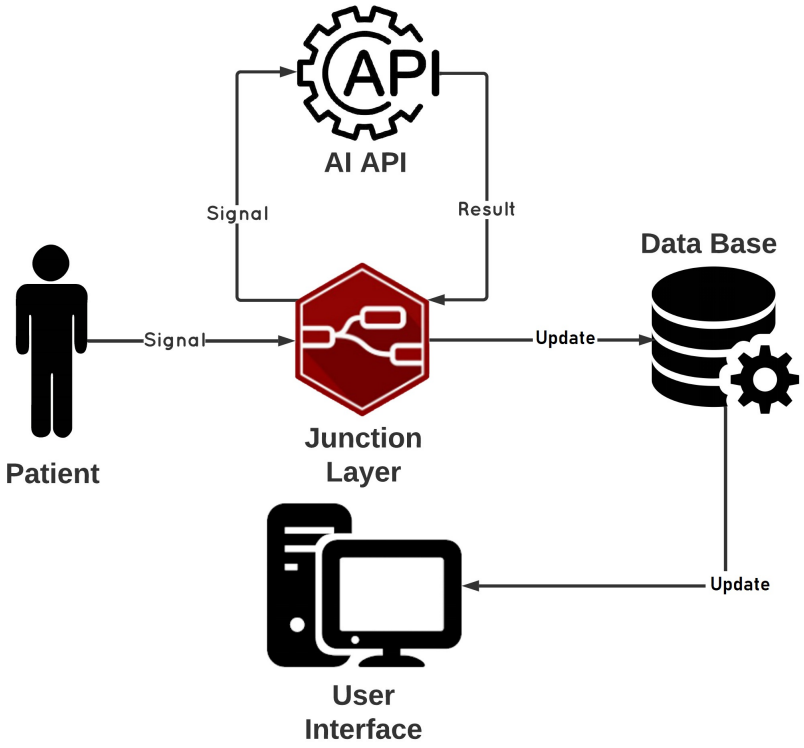


Figure 3.1: Display the Way Our Architecture Work

This architecture ensures a smooth exchange of data and results, allowing us to develop and evaluate our novel deep learning model effectively. The Junction Layer platform, as a simulation tool, enables us to test and refine our system, ensuring its

robustness and performance.

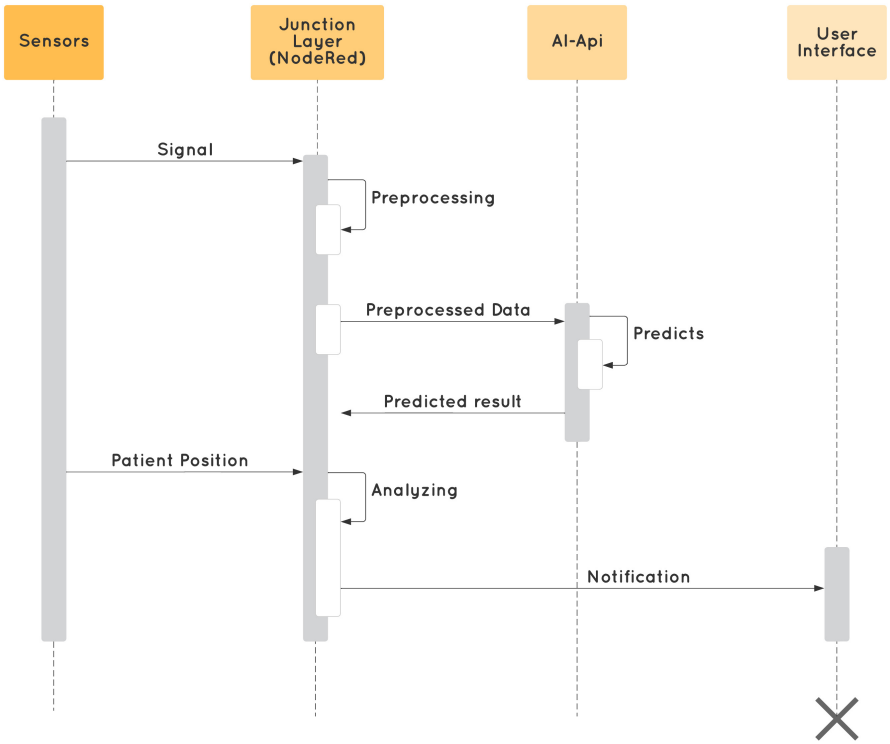


Figure 3.2: Sequence Diagram of Our Architecture

3.3 AI-API for HAR

3.3.1 Datasets

The dataset plays a crucial role in the development and evaluation of HAR systems. it provides the necessary input for training and testing the algorithms and serves as a representation of real-world activities. this section introduces the dataset utilized in this thesis, highlighting its characteristics and the relevance to the research objectives. in our research we have carefully selected and utilized two publicly available datasets: the WISDM dataset and the PAMAP2 dataset. these datasets have been widely used in the research community and cover a wide range of human activities.

3.3.1.1 WISDM

The WISDM (Wireless Sensor Data Mining) dataset is a well-known and widely used dataset in the field of human activity recognition. It was collected by researchers at the Wireless Sensor Data Mining (WISDM) Lab at Fordham University., which was collected using an Android smartphone app (see Figure 3.3). 36 participants were instructed. Each person had a smartwatch on their dominant hand and a smartphone in their pocket. Each of them was asked to perform 6 activities of 3 minutes each. The accelerometer sensor sampling rate was set to 20 Hz during activities. The WISDM HAR dataset contains detailed information as well as raw accelerometer time series data for three axes [57].

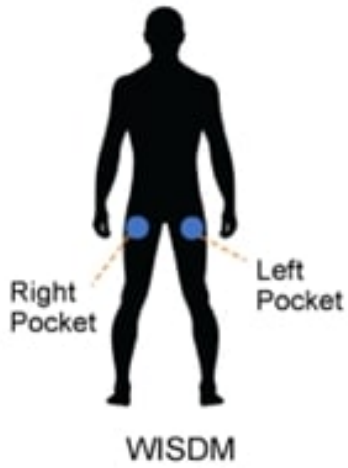


Figure 3.3: Placement of Inertial Sensors in WISDM Dataset

Information summaries	
Number of individuals	36 users
Number of activities	6 activities
Sensor frequency	20 Hz
Smartphone used	Google Nexus 5/5x Samsung Galaxy S5
Number of rows	1086466
Missing values ?	NO

Table 3.1: General Properties of WISDM Dataset

The activities represented in the dataset are : "walking", "standing", "sitting", "jogging", "upstairs", "downstairs". These activities represent common daily movements and provide a diverse set of motion patterns for training and evaluating activity recognition models. each data sample in the WISDM dataset includes a timestamp and the corresponding sensor readings. The dataset is labeled with the corresponding activity performed during each recording. This labeling allows for supervised learning approaches to train machine learning models for activity recognition tasks.

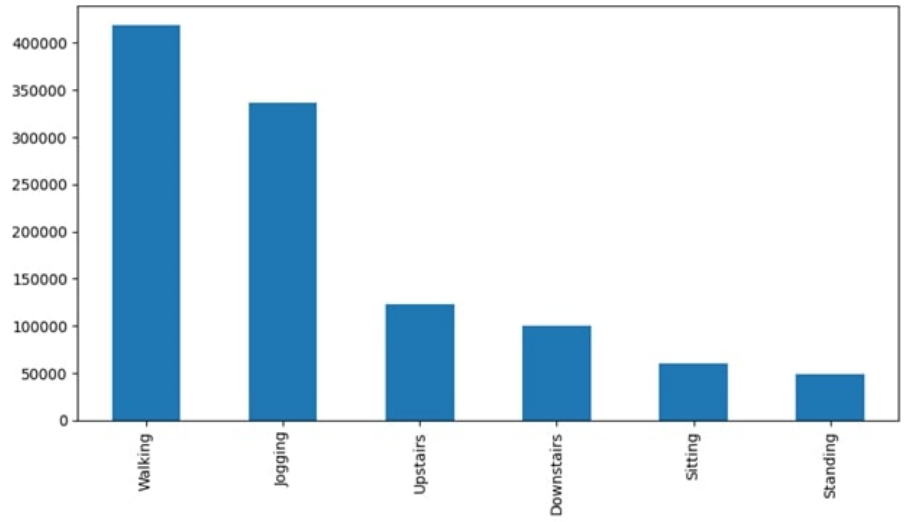


Figure 3.4: Class Distribution of WISDM Dataset

columns	Description
User	An integer identifier from 1 to 36
Activity	Object type activity name (int64)
Timestamp	Integer type (int64)
X-axis	Type real (float64), can be positive or negative
Y-axis	Type real (float64), can be positive or negative
Z-axis	Type real (float64), can be positive or negative

Table 3.2: Structure of WISDM Dataset

One advantage of the WISDM dataset is its large size, which provides a significant amount of training data for building robust activity recognition models. The dataset has been extensively used by researchers to develop and evaluate various machine learning algorithms and deep learning architectures for human activity recognition. due to its popularity and availability, the WISDM dataset has become a benchmark for activity recognition research, allowing for fair comparisons between different approaches. Researchers can directly access the dataset and utilize it in their studies, enabling reproducibility and facilitating advancements in the field.
In our thesis, we utilize the WISDM dataset to evaluate the performance of our proposed deep neural network model for human activity recognition. By leveraging this dataset, we aim to demonstrate the effectiveness and generalization capability of our model in accurately classifying various activities.

3.3.1.2 PAMAP2

The PAMAP2 (Physical Activity Monitoring of Older Adults, version 2) dataset is a publicly available dataset used for activity recognition research and human movement analysis. Collected by the Wearable Computing Lab at ETH Zurich

It was derived from three wireless inertial sensors (IMU) consisting of three-axis accelerometers, gyroscopes, and magnetometers located at the ankle, chest, and wrist

area for 18 household activities (see Figure 3.5), which were captured by 9 different subjects, 8 men and 1 woman . Data were collected at an average Sampling is 100 Hz and includes information about people's activities, such as walking, running, cycling and various household chores [45].

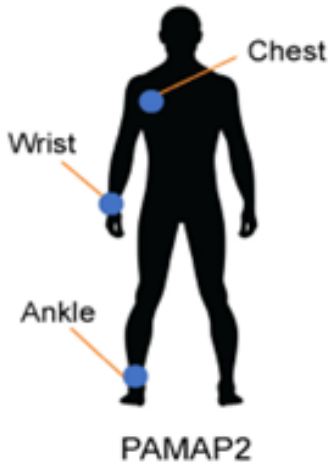


Figure 3.5: Placement of Inertial Sensors in PAMAP2 Dataset

Information summaries	
Number of individuals	9 users
Number of activities	18 activities
Sensor frequency	100 Hz
Number of Attributes	54
Number of rows	2872533
Missing values ?	YES

Table 3.3: General Properties of PAMAP2 Dataset

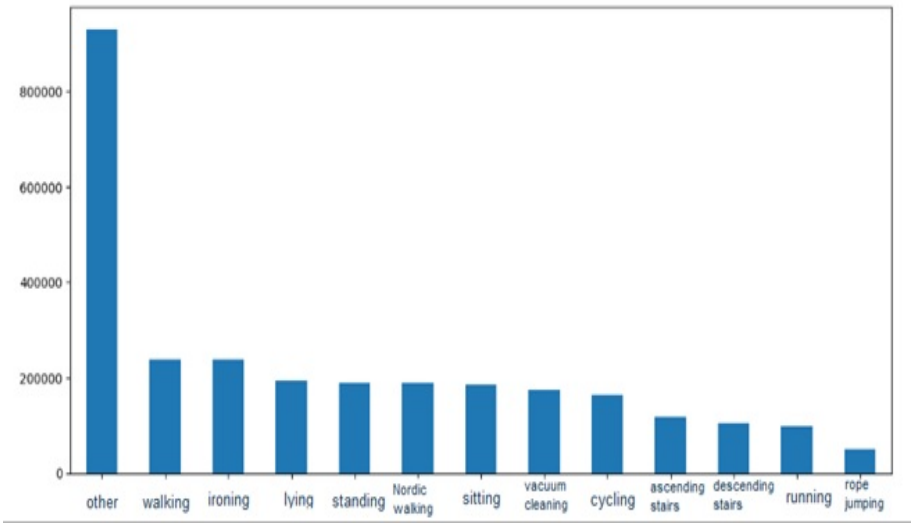


Figure 3.6: Class Distribution of PAMAP2 Dataset

The PAMAP2 dataset offers several benefits, including a wide range of diverse activities. This dataset contains recordings of various physical activities such as walking, running, cycling, and household tasks like vacuum cleaning and ironing. The inclusion of different types of activities allows researchers to investigate a broad spectrum of movements and assess the effectiveness of activity recognition models in different scenarios. This encourages the development of adaptable models capable of handling a variety of real-world activities.

columns	Description
1 Timestamp	Integer type (int64)
2 ActivityID	Object type activity name (int64)
3 Heart rate (bpm)	Type real (float64)
4-20 IMU hand	Type real (float64), can be positive or negative
21-37 IMU chest	Type real (float64), can be positive or negative
38-54 IMU ankle	Type real (float64), can be positive or negative

Table 3.4: Structure of PAMAP2 Dataset

In our thesis, we employ WISDM and PAMAP2 datasets to assess the performance of our proposed deep neural network model for recognizing human activities. By utilizing the dataset's extensive sensor data and diverse activity recordings, our objective is to showcase the effectiveness and ability of our model to accurately classify different physical activities based on wearable sensor measurements.

3.3.2 Dataset Preparation

3.3.2.1 Data Preprocessing

Once the data set is collected, it is presented to the users after undergoing some preprocessing. However, there is still a need for some cleaning before the pre-processing. By removing some undesirable symbols ,there may be instances where data was missing due to breaks during data collection. In such cases, we calculate and compensate for the missing data to ensure it does not affect our analysis. It is best to use an interpolation that creates a new data point from a set of known data points.also we can eliminate unnecessary columns that do not contribute to our work and do not provide any additional value.

Finally, we obtain a dataframe that meets all the data requirements and necessary properties for the deep learning model we are going to use. After preparing the dataframe, the next step is to divide it into separate parts, with each part containing data from a specific user in the data collection process. In the second step, we apply the sliding window protocol, (check algorithm 1). This algorithm is designed to segment time series data, specifically sensor readings data in the context of HAR. The algorithm takes as input the sensor readings data, activities, window length (90), and stride (40). It then iterates through the data, extracting windows of a specified length with a certain stride. Each window represents a portion of the data, and it is stored in a list along with its corresponding sub-activity. This segmentation allows for a more fine-grained analysis of the data, enabling the development of HAR models and systems that can accurately recognize and understand human activities. The algorithm

outputs the segmented windows and their corresponding sub-activities, which can be further utilized in subsequent processing and analysis. by applying this algorithm on each user activities individually, segments of shape of (N, 90)are obtained, where N represents the number of features or channels. Specifically, the WISDM dataset has 3 features, while the PAMAP2 dataset has 36 features.

Algorithm 1 Sliding Windows for HAR

Require: Sensor readings data X with N data points
Require: Activities Y
Require: Window length W and stride S
Ensure: Segmented windows $W_{segments}$
Ensure: Sub-Ativities $Y_{segments}$

```
function SlidingWindows(X, Y, N, W, S)
    Initialize empty list W_segments
    for i = 0 to N − W with step size S do
        Extract window W_i from X[i : i + W]
        Append W_i to W_segments
        Append Y_i to Y_segments
    end for
    return W_segments, Y_segments
end function
```

3.3.2.2 Data Splitting

Once the dataset is preprocessed for training and evaluation, ott its crucial step to split the it into two distinct parts: the training set and the test set. The training set comprises 80% of the dataset and is used to adjust the DL model weights. This training process allows the model to learn and extract meaningful patterns from the data. The remaining 20% of the dataset is allocated as the test set. The purpose of the test set is to evaluate the predictive accuracy of the developed model. By assessing

its performance on unseen data, we can gauge how well the model generalizes to new instances.
It ensures that the model is capable of making accurate predictions on new instances, indicating its effectiveness and suitability for real-world applications.

3.3.3 Our Approach

In this work , a Multi-Branch Convolutional LSTM Network (MBCLNet) is proposed, it work on 1D data generated from sensors . Firstly, we incorporate an autoencoder as a preprocessing step to rebuild dataset. The autoencoder compresses the input data into a lower-dimensional latent space and then reconstructs it. These extracted features are then fed into the MBCLNet model. By leveraging the learned representations from the autoencoder, our model can effectively analyze and predict complex temporal patterns in the data. This integration enhances the model's performance and improves its ability to capture important patterns and dependencies in the dataset. (Figure 3.7) represents the different steps to follow to train the network.

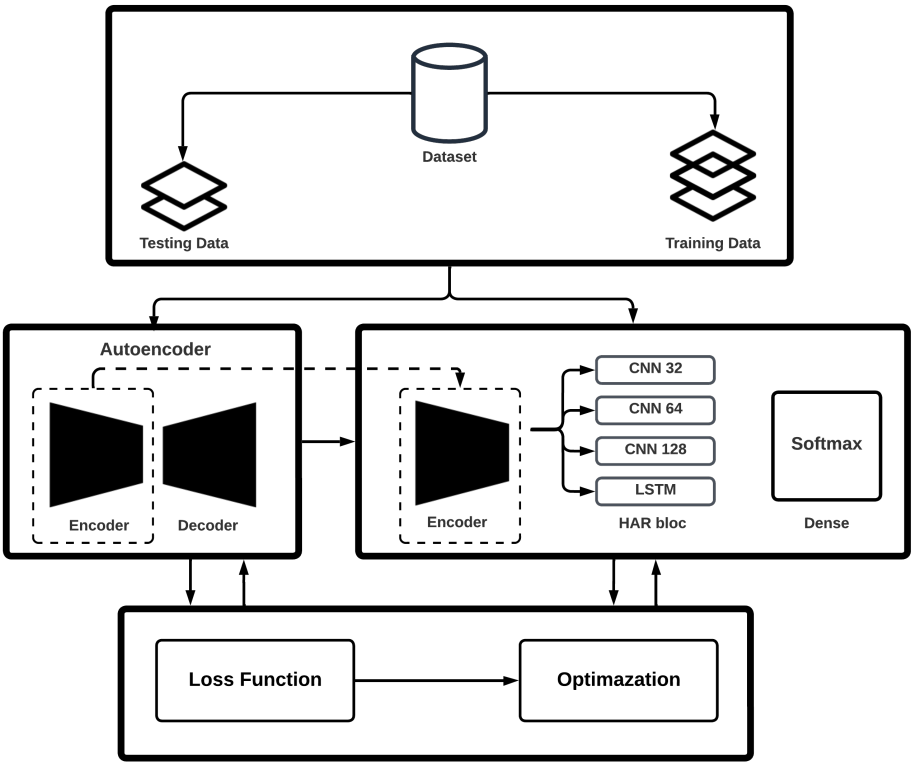


Figure 3.7: The approach we adopted to train our model

3.3.3.1 Autoencoder Model

The autoencoder model consists of three Conv1D encoder layers, each with a decreasing number of filters. This design choice allows data to be progressively compressed as it passes through layers. Specifically, the three encryption layers operate on (64, 32, 16) filters respectively (see Figure 3.8). These layers play an important role in extracting meaningful representations from the input data. In order to reconstruct the original input from the compressed representation, the autoencoder model incorporates three corresponding Conv1D decoder layers. These decoder layers work in reverse order, with an increasing number of filters. The first decoder layer uses 32 filters, followed by the second layer with 64 filters, and the final decoder layer with 90 filters. Their purpose is to restore the compressed data back to its original size

and shape. To make the model more efficient at learning meaningful representations and to further reduce the size of the input data, we make max pooling layers between every two convolutional layers. However, the small number of features in the WISDM dataset made it difficult to apply max-pooling. As for the pamap2 dataset it was easy to use max pooling, we used a max-pooling layer with a filter size of 2. In the final convolution layer of the model, a sigmoid activation function is used to output values between 0 and 1.

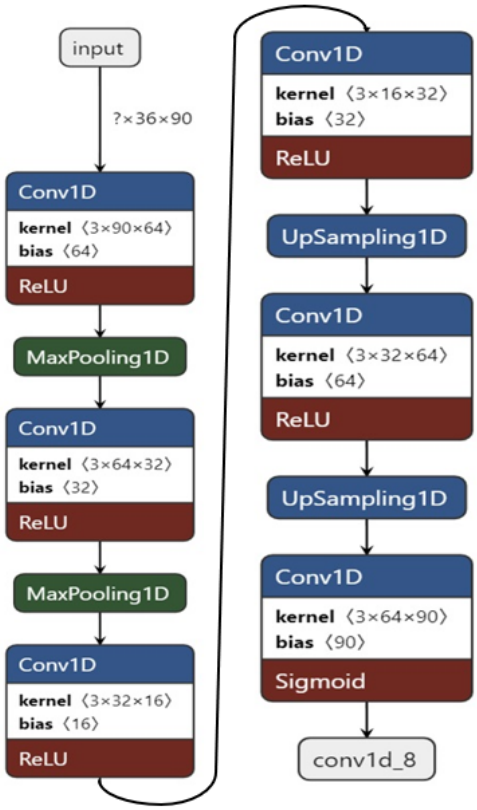


Figure 3.8: The Autoencoder Model Used

Training Autoencoder During the training process of the autoencoder, it is crucial to configure various parameters to ensure effective learning. One such parameter is the optimizer, where we set it to Adam, a widely utilized algorithm for adjusting neural network weights during training. This choice enables the optimizer to optimize

the autoencoder's weights efficiently. To measure the quality of the autoencoder's performance, we employ the mean squared error (MSE) as the chosen loss function. The MSE quantifies the average squared difference between the original input and the reconstructed output. By minimizing this reconstruction error, the autoencoder aims to faithfully reproduce the input data.

The training process involves iterating through the dataset multiple times, and in this case, we iterate five times. Each iteration includes a batch size of 256 samples, allowing the autoencoder to process and learn from a subset of data at a time. This iterative approach facilitates the gradual optimization of the autoencoder's weights, enhancing its ability to reconstruct the input data accurately. To assess the reconstruction accuracy, we rely on metrics such as the MSE. Lower MSE values indicate a higher level of fidelity in reconstructing the input data, highlighting the effectiveness of the autoencoder in capturing the underlying patterns and features. By striving for lower MSE values, the autoencoder aims to achieve superior reconstruction performance and faithfully reproduce the original input data.

```
Epoch 1/5
3820/3820 [==============================] - 33s 6ms/step - loss: 49.3755
Epoch 2/5
3820/3820 [==============================] - 20s 5ms/step - loss: 49.1030
Epoch 3/5
3820/3820 [==============================] - 21s 6ms/step - loss: 49.0517
Epoch 4/5
3820/3820 [==============================] - 21s 5ms/step - loss: 49.0316
Epoch 5/5
3820/3820 [==============================] - 21s 6ms/step - loss: 49.0253
<keras.callbacks.History at 0x7f3644082b60>
```

```
30555/30555 [==============================] - 63s 2ms/step
48.992725
```

Figure 3.9: The Loss Value of the Autoencoder Model

3.3.3.2 Proposed Model

After conducting a comprehensive analysis of the existing literature, we have devoted considerable effort to crafting an optimized architecture that consistently yields impressive results. Our goal was to design a model that can effectively process input data with a shape of (N, 90), where 90 indicates time steps and N number of features per time step.

The architecture of our model encompasses various layers (see Figure 3.10), each serving a specific purpose in the overall learning process. Among these layers are Conv1D, MaxPooling1D, Dropout, Dense, LSTM, and concatenation layers. Together, they work in harmony to extract meaningful features from the input data and facilitate accurate classification. To achieve this, we employed the categorical crossentropy loss function, which measures the dissimilarity between the predicted probability distribution and the true distribution of the classes. The Adam optimizer, known for its efficiency and effectiveness in training neural networks, was utilized to adjust the weights and biases of the model during the learning process. After successfully training the autoencoder model and achieving satisfactory results, we proceeded to extract the trained weights from the first part of the encoder. These weights were then used by integrating them into another model for further processing or as a starting point for engineering subsequent models. By reusing the representations gained from the autoencoder, we aim to reduce the number of parameters to not train those layers again. Building upon this foundation, we introduce four parallel branches within the model architecture. Each branch specializes in handling the obtained features, applying specific operations to further refine the information. The first branch incorporates two consecutive 1D convolutional layers with 32 filters of size N. Similarly, the second and the third branch employs two consecutive 1D convolutional layers with 64 and 128 filters of size N, respectively, providing a broader scope for feature extraction. Lastly, the fourth branch introduces an LSTM layer with 64 units, allowing the model to capture temporal dependencies within the data. The outputs from these parallel branches are then concatenated, facilitating the integration of diverse features. To obtain a fixed-size feature vector, we employ a global average pooling

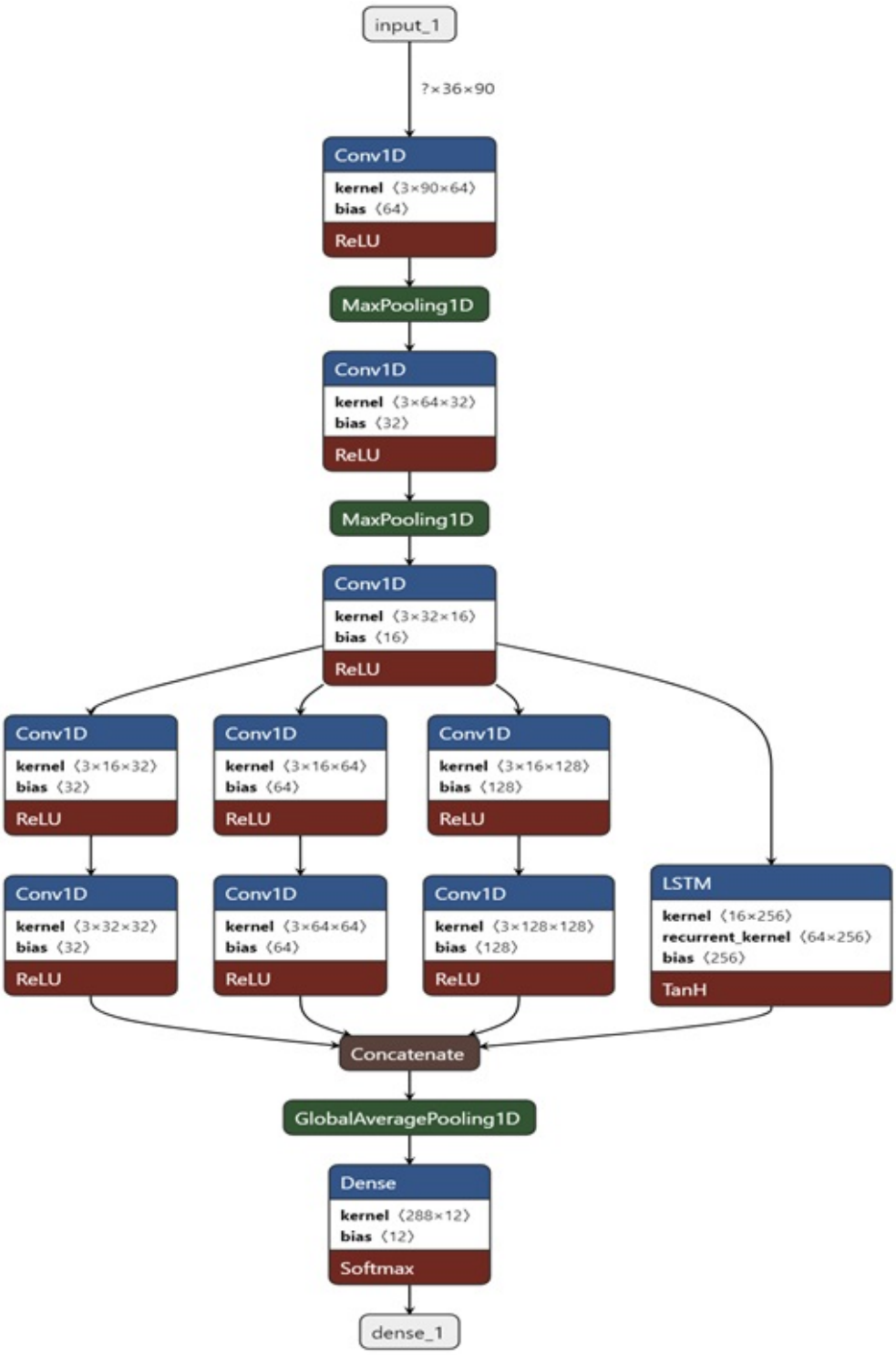


Figure 3.10: The Presented Figure Displays the Utilized Architecture

layer. This layer aggregates the outputs and produces a condensed representation, summarizing the most salient features from the parallel branches. The condensed feature vector is then passed through a fully connected layer with 288 units of size N, enabling deeper processing and extraction of higher-level representations. To prevent overfitting and enhance model generalization, we incorporate a dropout layer with a rate of 0.2. This layer randomly masks a portion of the inputs during training, effectively reducing the models reliance on specific features and promoting robustness. To ensure that we capture the best performing model during the training process, we incorporated the Model Checkpoint callback. This callback periodically saves the weights of the model with the lowest validation loss. By doing so, we can retrieve the model with the optimal performance for further evaluation and deployment. Finally, the processed data flows into a dense layer with A units, corresponding to the number of classes in the dataset. Activated by a SoftMax function, this layer produces the models output, generating a probability distribution over the classes, representing the models prediction for the given input. Through this carefully designed architecture, we aim to capture the inherent com-plexities and patterns present in the input data, enabling accurate classification and prediction. By leveraging convolutional, pooling, and recurrent operations, as well as transfer learning techniques.

	Parametere	WISDM	PAMAP2
Auto-Encoder	Trainable Parameters	50,218	50,218
	Non-Trainable Parameters	0	0
Proposed Model	Trainable Parameters	113,030	99,916
	Non-Trainable Parameters	25,072	25,072

Table 3.5: The Different Number of Parameters

3.4 Results and Discussion

In this section, we present the comprehensive results of our experiments, highlighting the performance and effectiveness of our proposed methodology. The obtained results provide valuable insights into the capabilities of our deep learning model in accurately recognizing Human Activities. We present a detailed analysis of the achieved accuracy, precision, recall, and other relevant metrics, showcasing the model's ability to successfully classify and identify different activities. Additionally, we discuss the comparative performance of our model against existing approaches, shedding light on its superiority and contributions to the field.

3.4.1 Key Classification Metrics

Precision: Precision is the ratio between the true positives and the sum of the true and false positives. where TP represents true positives (examples correctly identified as positives) and FP represents false positives (examples incorrectly identified as positives).

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

Recall: Recall is the ratio of true positives over the sum of true positives and false negatives, where TP represents true positives and FN represents false negatives (examples incorrectly identified as negative).

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

Support: Support is the number of signals of each class in the database.

Accuracy: Accuracy is a metric used to assess the performance of a classification model by measuring the proportion of correctly classified instances out of the total

number of instances.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.3}$$

F1-score: F1-score is a measure of precision and recall that combines these measures into a single value. The F1 score is a general measure used as the primary measure of classification performance.

$$F1\ score = \frac{2 \times (Precision \times Recall)}{Precision + Recall} \tag{3.4}$$

Parameters	Wisdm	Pamap2
Number of layers	14	16
Activation function	Softmax / ReLU / TanH	Softmax / ReLU / TanH
Epochs	10	10
Batch size	256	64
Learning rate	0.0001	0.0001

Table 3.6: Different Key Classification Metrics

To present the outcomes achieved by our model, we have provided a comprehensive analysis of the results based on the following aspects: Accuracy, Loss, Classification report, Confusion matrix.

3.4.1.1 Accuracy and losses curves

The following Figures (from 3.11 to 3.14 display the graphical representations of the accuracy and loss during both training and validation phases, We note that the validation accuracy increases proportionally with the number of periods in addition to training. On the other hand, loss refers to a decrease inversely proportional to the number of epochs.

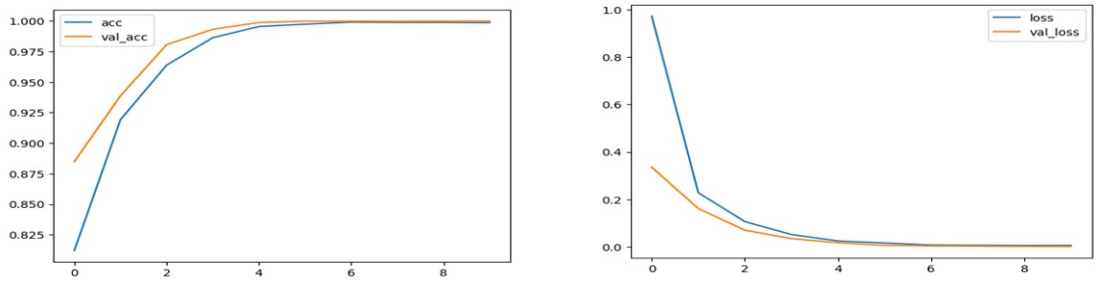


Figure 3.11: Accuracy and Loss Training Curves on WISDM Dataset

```
Epoch 1/10
3056/3056 [==============================] - 36s 10ms/step - loss: 0.9867 - acc: 0.8262 - val_loss: 0.2807 - val_acc: 0.9041
Epoch 2/10
3056/3056 [==============================] - 28s 9ms/step - loss: 0.2053 - acc: 0.9276 - val_loss: 0.1344 - val_acc: 0.9514
Epoch 3/10
3056/3056 [==============================] - 32s 10ms/step - loss: 0.1016 - acc: 0.9666 - val_loss: 0.0734 - val_acc: 0.9769
Epoch 4/10
3056/3056 [==============================] - 28s 9ms/step - loss: 0.0537 - acc: 0.9859 - val_loss: 0.0318 - val_acc: 0.9951
Epoch 5/10
3056/3056 [==============================] - 30s 10ms/step - loss: 0.0271 - acc: 0.9955 - val_loss: 0.0200 - val_acc: 0.9976
Epoch 6/10
3056/3056 [==============================] - 28s 9ms/step - loss: 0.0158 - acc: 0.9978 - val_loss: 0.0098 - val_acc: 0.9993
Epoch 7/10
3056/3056 [==============================] - 28s 9ms/step - loss: 0.0104 - acc: 0.9984 - val_loss: 0.0048 - val_acc: 0.9998
Epoch 8/10
3056/3056 [==============================] - 28s 9ms/step - loss: 0.0069 - acc: 0.9989 - val_loss: 0.0041 - val_acc: 0.9999
Epoch 9/10
3056/3056 [==============================] - 29s 9ms/step - loss: 0.0060 - acc: 0.9989 - val_loss: 0.0041 - val_acc: 0.9996
Epoch 10/10
3056/3056 [==============================] - 30s 10ms/step - loss: 0.0054 - acc: 0.9988 - val_loss: 0.0027 - val_acc: 0.9999
```

Figure 3.12: The Training Process of Our Model on WISDM Dataset

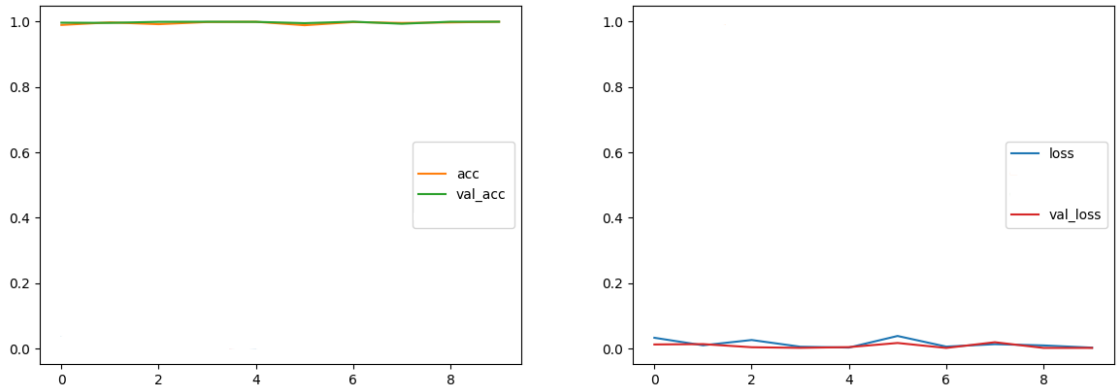


Figure 3.13: Accuracy and Loss Training Curves on PAMAP2 Dataset

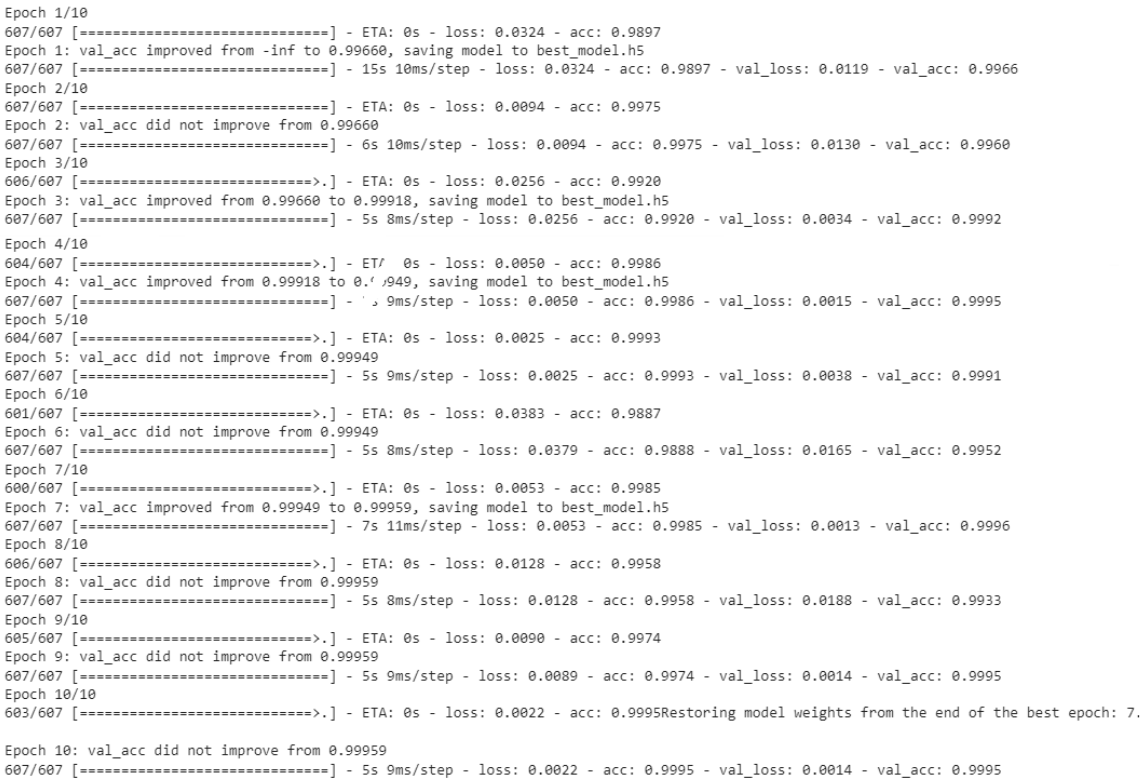


Figure 3.14: The Training Process of Our Model on PAMAP2 Dataset

3.4.1.2 Classification Report

A classification report is an evaluation tool used to measure the performance of a classification model. It displays the precision, recall, f1-score1 and support of our model. Thus, it allows us to better understand the overall performance of our model.

The performance of the classification model was assessed using a dataset comprising a total of 195,552 signals that were categorized into six different classes. To evaluate the model's effectiveness, precision, recall, and F1 score were calculated for each individual class. These metrics provide insights into the model's ability to accurately classify instances from each class. Additionally, both the weighted and unweighted means of precision, recall, and F1 score were computed to gauge the overall performance of the model across all classes.

WISDM

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18152
1	1.00	1.00	1.00	60665
2	1.00	1.00	1.00	10696
3	1.00	1.00	1.00	8757
4	1.00	1.00	1.00	22055
5	1.00	1.00	1.00	75227
accuracy			1.00	195552
macro avg	1.00	1.00	1.00	195552
weighted avg	1.00	1.00	1.00	195552

Figure 3.15: Evaluation Model of WISDM Dataset

Upon analyzing the results, it is noteworthy that the model achieved exceptional performance, with a 100% precision, recall, and F1 score for all classes. This outcome is indicative of the model's high accuracy and reliability in correctly identifying instances belonging to each class. Such outstanding results signify the effectiveness of the classification model in accurately categorizing the signals and reflect the model's capability to make robust predictions across all classes.

Similar observations were made when evaluating the classification model on a separate dataset comprising 9,712 signals distributed among 12 distinct classes. Once again, precision, recall, and F1 score were calculated for each individual class, providing valuable insights into the model's performance across various categories.

PAMAP2

	precision	recall	f1-score	support
lying	1.00	1.00	1.00	1400
sitting	1.00	1.00	1.00	921
standing	1.00	1.00	1.00	872
walking	1.00	1.00	1.00	858
running	1.00	1.00	1.00	709
cycling	1.00	1.00	1.00	866
Nordic walking	1.00	1.00	1.00	707
ascending stairs	1.00	1.00	1.00	613
descending stairs	1.00	1.00	1.00	580
vacuum cleaning	1.00	1.00	1.00	893
ironing	1.00	1.00	1.00	965
rope jumping	1.00	1.00	1.00	328
accuracy			1.00	9712
macro avg	1.00	1.00	1.00	9712
weighted avg	1.00	1.00	1.00	9712

Figure 3.16: Evaluation Model of PAMAP2 Dataset

Remarkably, the model achieved a perfect score of 100% in precision, recall, and F1 score for all classes within this dataset. This exceptional outcome further highlights the model's robustness and accuracy in accurately classifying signals across diverse categories.

3.4.1.3 Confusion Matrix

A confusion matrix is a metric used to evaluate the performance of a classification model. It helps us understand how accurate and effective the model is. It provides a detailed summary by comparing the predicted class labels with the actual class labels. The confusion matrix is displayed as a table. The rows represent the actual class labels, while the columns represent the predicted class labels. Each cell in the matrix represents the count or proportion of observations that belong to a specific

combination of predicted and actual class labels. in the Wisdm dataset, we have a total of six classes, while in the PAMAP2 dataset, we have twelve classes. As a result, our confusion matrix for the Wisdm dataset is a 6x6 matrix, and for the PAMAP2 dataset, it is a 12x12 matrix.

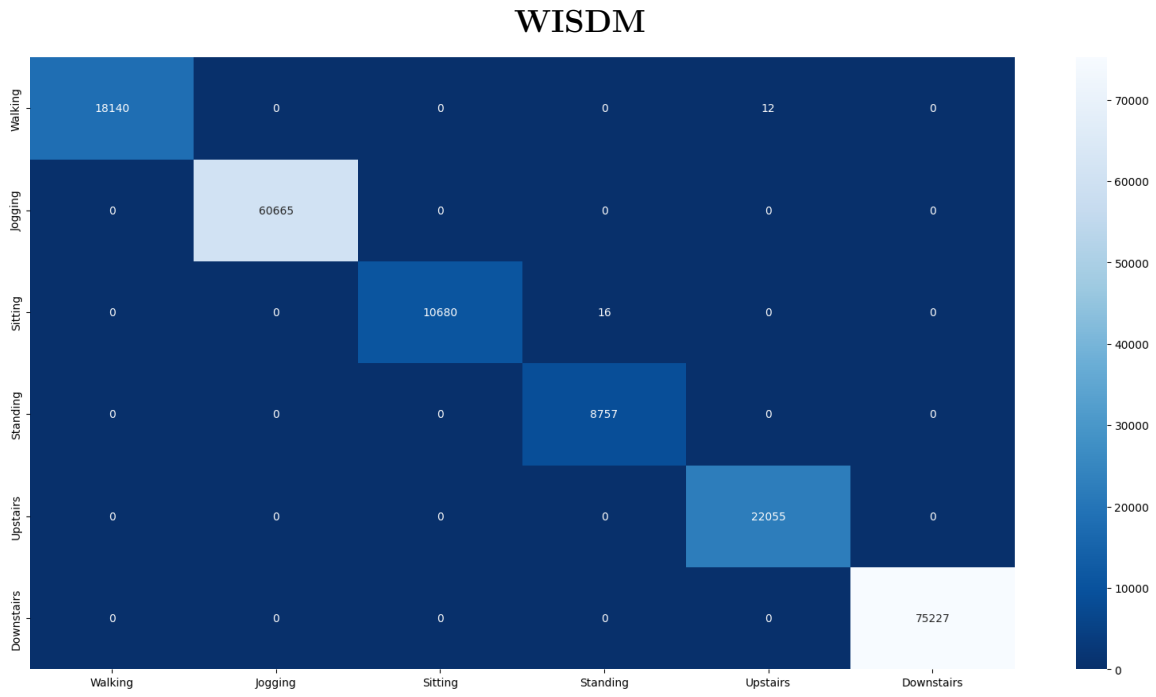


Figure 3.17: Confusion Matrix of WISDM Dataset

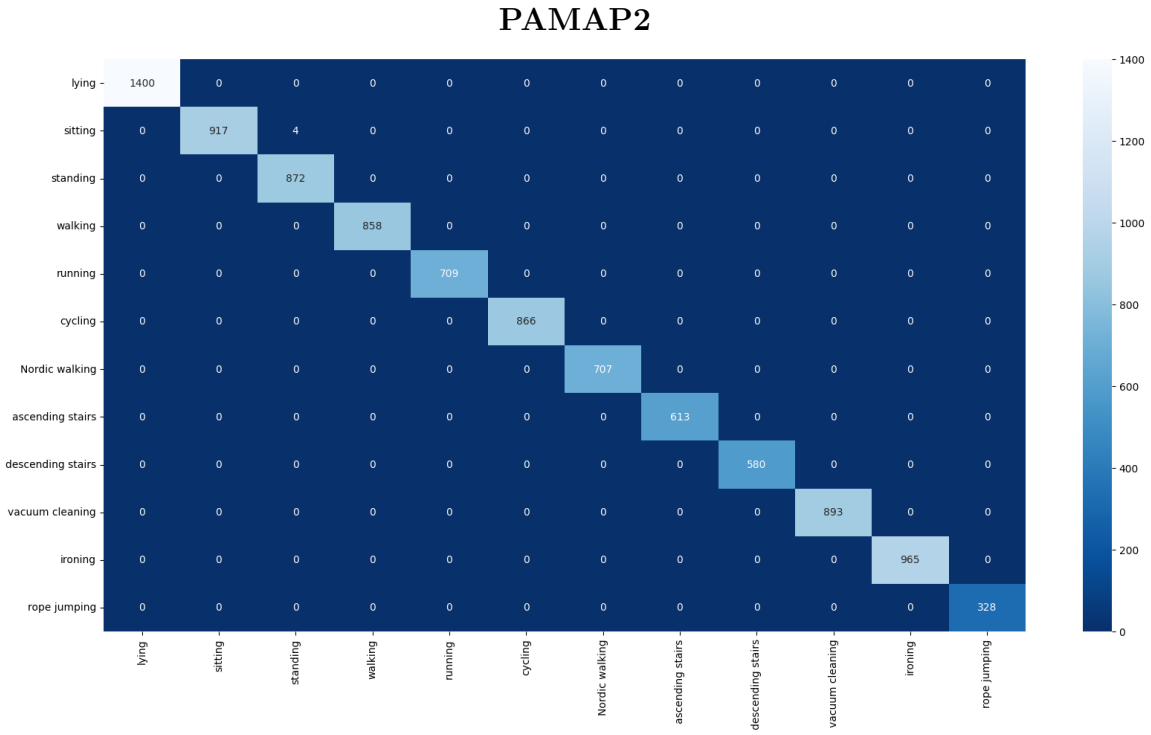


Figure 3.18: Confusion Matrix of PAMAP2 Dataset

The accuracy metric indicates how well the model performs in predicting the correct label for each input sample in the test set. Overall, the results obtained from training our model on the WISDM and PAMAP2 datasets show that our model achieves high accuracy on both datasets, indicating its effectiveness in accurately classifying human activities.

Performance Results of Model Training The following table (Table 3.7) presents the results obtained from training our model on two different datasets: WISDM and PAMAP2.

Dataset	Sliding Window	Preprocess Before	Preprocess After	Train/Test Ratio	Batch Size	Accuracy
WISDM	70	1086466	977796	70/30	256	99.27
PAMAP2	70	1942872	48563	70/30	128	98.12
WISDM	90	1086466	977760	80/20	256	**99.98**
PAMAP2	90	1942872	48557	80/20	64	**99.95**
WISDM	120	1086466	977724	80/20	128	99.45
PAMAP2	120	1942872	48549	80/20	64	99.23

Table 3.7: Results of Training and Testing on Different Parameters

The table presents the results of training and testing on different sliding window sizes for Human Activity Recognition. Among the evaluated window sizes (70, 90, and 120), a window size of 90 consistently achieves the highest accuracy on both the WISDM and PAMAP2 datasets. With an accuracy of 99.98% for WISDM and 99.95% for PAMAP2, a sliding window size of 90 proves to be the most effective in capturing patterns and temporal dependencies in the data.

3.4.2 Comparison

In this section, we compare our approach with some of related work in recent years mentioned in (Section2.3) The selected works are based on the WISDM and PAMAP2 datasets As shown in the (Table 3.8). our proposed methodology yields the highest accuracy for HAR than all the previous experiments. Our model has several advantages that contribute to its superior performance. Firstly, it benefits from transfer learning by using pre-trained weights in the encoder layers and freezing their training. This reduces the number of parameters that need to be learned and improves the model's ability to learn from limited labeled data. Additionally, our model incorporates parallel branches, which allow for the exploration of diverse features and expand the range of information that can be extracted. This helps in capturing a broader understanding of the data and improving the model's accuracy. Furthermore, the

global average pooling layer in our model summarizes the most important features from the parallel branches into a fixed-size feature vector. This pooling mechanism enables the model to focus on relevant and discriminative information, enhancing its robustness and ability to make accurate predictions.
Overall, the well-designed architecture of our model, contributes to its superior performance in accurately recognizing and classifying human activities.

Author	Method	Year	WISDM Results %	PAMAP2 Results %
Qi Tenget[53]	layer-wise CNN	2020	acc=98.82	acc=92.97
Andrey Ignatov[23]	CNN	2018	acc=90.42 - 93.32	acc=97.47
KUN XIA[60]	LSTM-CNN	2020	acc=95.85	*
Sravan Kumar[6] Challa	CNNBiLSTM	2022	acc=96.05	acc=94.29
Yu Guan[17]	LSTM	2017	*	F1-Score=0.854 (+-) 0.026
haohua Wan[58]	BLSTM LSTM CNN	2020	*	acc=98.40(BLSTM) acc=89.01(LSTM) acc=92.71(CNN)
Saedeh Abbaspou[1]	CNN-LSTM CNN-GRU CNN-BiLstm Cnn-BiGru	2020	*	acc= 99.57(CNN-LSTM) acc=99.06(CNN-GRU) acc=99.65(CNN-BiLstm) acc= 99.80(Cnn-BiGru)
Nidhi Dua[13]	CNN-GRU	2021	acc=97.21	acc=95.27
Our Model	**Multi-Branch Convolutional LSTM Network (MBCLNet)**	**2023**	**acc=99.98**	**acc=99.95**

Table 3.8: Table Represent Comparison of Classification Results for Different Methods an Our Method on WISDM and PAMAP2 Datasets

3.5 Conclusion

Chapter 3 unveils the holistic structure of our research on Human Activity Recognition. Through the integration of interconnected components, including our powerful deep learning model, we have established a robust framework for accurate activity

recognition and real-time monitoring. By carefully selecting datasets, developing an API, and creating a user interface for result presentation, our architecture empowers healthcare professionals with prompt monitoring capabilities. This research contributes to the advancement of healthcare systems by enhancing the quality of care provided to individuals through comprehensive activity recognition and real-time monitoring.

Chapter 4

Realisation & Implementation

Discover a world where the symphony of seamless data processing and intuitive engagements mesmerizes, crafted with meticulous precision and a keen eye for user-centricity.

4.1 Introduction

Chapter 4 explores the practical implementation of the Junction Layer and the development of a web application, two integral components of our HAR system. These elements are essential for ensuring accurate recognition, seamless data processing, and an intuitive user experience.

The chapter begins with a detailed exposition of the environment and development tools utilized in our research. We then shift our focus to the implementation and simulation of the Junction Layer using Node-RED, a powerful flow-based programming tool. By leveraging the WISDM dataset, we construct a flow that orchestrates the processing and analysis of human activity data, including the determination of a person's location. Next, we delve into the development of the web application, which serves as the user-facing component of our HAR system. This application provides an intuitive and interactive platform for users to access and effectively leverage the system's features. It offers real-time insights into the system's output, allowing users to interact with panels, visualize activity data, and gain a comprehensive understanding of recognized human activities.

Throughout this chapter, we explore the practical implementation of the Junction Layer and the development of the web application, highlighting their significance in achieving accurate recognition, seamless data processing, and enhancing the user experience.

4.2 Environment & Development Tools

4.2.1 Programming Language, Framework & API

4.2.1.1 Python

python is a high-level interpreted programming language widely used in various fields such as scientific computing, data analysis, and rapid application development. It was first released by Guido van Rossum in 1991 and has since become one of the most

popular programming languages in the world. Python is a powerful, easy-to-learn, and versatile programming language. The language is supported by the Python Software Foundation (PSF) as it helps maintain and develop the Python programming language by providing various resources, including development funding and community projects.
Python is a powerful and easy-to-learn programming language. It incorporates efficient high-level data structures and a simple but effective approach to object-oriented programming. We also use it to build web applications with frameworks like Django and Flask, and several libraries like NumPy, Pandas, and TensorFlow for data analysis and machine learning [15].

4.2.1.2 JavaScript

JavaScript is a versatile programming language used for web scripting on both the client and server sides. It enables website creators to execute code when users visit their sites. It is supported by all modern web browsers and can be used both on the client side (in the browser) and on the server side (with the help of platforms such as Node.js) [51].

4.2.1.3 Django Framework

Django, a Python-based web framework, empowers developers to efficiently and rapidly create web applications. It follows the Model-View-Controller (MVC) architectural pattern, promoting the separation of concerns. Django integrates a built-in URL routing system that associates URLs with views, providing a means to organize and handle diverse request types within the application. Moreover, Django incorporates a testing framework, facilitating the creation of unit tests and automated testing for applications. With its comprehensive set of features and robust foundation, Django simplifies web development, enabling developers to focus on implementing core functionalities with ease [16].

4.2.1.4 REST API

A REST API, also known as a Representational State Transfer Application Programming Interface, embodies an architectural style for designing networked applications. It establishes a framework of rules and conventions for constructing web services that facilitate communication between different systems via the internet. By leveraging standard HTTP methods such as GET, POST, PUT, and DELETE, REST APIs enable the manipulation of resources identified by URLs. They adhere to fundamental principles including statelessness, resource-centric design, support for CRUD operations, a consistent interface, and stateless communication. Data representation in REST APIs commonly employs JSON or XML formats. Developers interact with these APIs by utilizing HTTP clients to send requests to specific endpoints and receive responses containing data or operation outcomes. REST APIs find extensive application in web, mobile, and system integration scenarios, providing a flexible and scalable mechanism for exchanging data and functionality across the internet [41].

4.2.2 Environment

4.2.2.1 COLAB

Google Colab is a cloud service provided by Google that allows users to create and share Jupyter notebooks containing executable code, text, and multimedia content such as images, HTML, and LaTeX. Users can store Colab notebooks in their Google Drive account and easily share them with co-workers or friends for collaboration purposes.

Google Colab allows its users to access a free virtual environment to run code that includes access to GPUs and TPUs, it provides 12GB of RAM and a Tesla K80 NVIDIA GPU2, it allows anyone to write and execute arbitrary Python code Through the browser, making it a popular choice among machine learning practitioners and researchers for running complex deep learning models.

4.2.2.2 Google Drive

Google Drive is a widely used and popular cloud-based file storage and synchronization service developed by Google. Users can also access Google Drive through the web interface or by using the Google Drive mobile app for Android or iOS. The service allows users to upload and store different types of files. of files, with a capacity of 15 GB free for individual users, and provides easy access to files and collaboration tools from anywhere with an Internet connection.

4.2.2.3 Node.js

Node.js is an open source server-side runtime environment that allows developers to create and execute JavaScript code outside of a web browser, is free and runs on various platforms (Windows, Linux, Unix, Mac OS X, etc..). Node.js enables server-side scripting with JavaScript, allowing developers to create dynamic web pages and perform server-side tasks. It can be used to develop various types of applications, including web servers, command-line tools, desktop applications, and even Internet of Things (IoT) devices.

4.2.3 Libraries

4.2.3.1 TensorFlow

TensorFlow is an open-source machine learning library developed by Google that allows users to build cutting edge deep learning and machine learning applications. It provides a flexible and scalable framework for creating a wide variety of models, including deep neural networks.

4.2.3.2 Keras

Keras is a popular open-source Python library for building and training deep learning models. It offers a simple and easy-to-use interface for creating different types of neural networks. Keras supports multiple backends, such as TensorFlow, Theano,

and CNTK, making it flexible in terms of hardware and software configurations. Its simplicity and versatility make it an excellent tool for exploring and developing deep learning models.

4.2.3.3 Pandas

Pandas is an open-source data analysis and manipulation library for the Python programming language. it provides a fast, flexible, and expressive way to work with structured data, such as spreadsheets and databases. it provides Panda with a set of main features that represent data input and output, its structure and processing so that it can filter, select, merge, aggregate and transform data. It can also integrate with other libraries such as Matplotlib and Seaborn to create high-quality visualizations of data.

4.2.3.4 NumPy

NumPy or (Numerical Python) is a free and open-source Python library for numerical computation. It is used for higher level math functions. It is the global standard for working with numerical data in Python and adds support for large multidimensional arrays and arrays. NumPy is a powerful tool for numerical computation in Python. They are widely used in scientific computing, data analysis, and machine learning.

4.2.3.5 Matplotlib

Matplotlib is a Python library for creating static, animated, and interactive visualizations in Python. It is an alternative to the MATLAB application, Matplotlib provides a variety of functions for creating different types of plots, including line plots, scatter plots, bar plots, and histograms, as well as backends for exporting plots to various file formats, such as PNG, PDF, and SVG. Matplotlib is a powerful and flexible library for creating high-quality visualizations in Python, and is widely used in scientific computing, data analysis, and machine learning.

4.2.3.6 Scikit-learn

Scikit-learn (also known as sklearn) is a popular Python machine learning library that providesa wide range of tools for implementing various machine learning algorithms. It is built on NumPy, SciPy and matplotlib. Sklearn It is utilized for determining precision and recall, as well as evaluating and calculating the confusion matrix, Precision, sensitivity, specificity and F1-score.

4.3 Junction Layer (Node-Red)

Node-RED, an IBM-developed tool for IoT application programming, is a visual programming tool that facilitates the connection of devices, APIs, and services over the Internet through flow creation. It is constructed on the foundation of Node.js and employs JavaScript for scripting purposes. Utilizing a web-based editor, users can easily drag and drop nodes onto a canvas and establish connections to construct a flow. Each individual node within the flow corresponds to a particular function or task, including tasks such as sensor data reading, data processing, API requests, or hardware control. Node-RED is highly flexible and expandable, offering an extensive collection of pre-built nodes contributed by the community, which enable seamless interaction with diverse devices, services, and protocols. Moreover, users have the capability to generate their own custom nodes to further enhance the functionality of Node-RED [37][18].

Node-RED provides a palette of nodes that can be interconnected to create flows. (Figure 4.1) displays a screenshot of the Node-RED platform, which consists of three main sections: the Nodes bar, workspace, and sidebar. The Nodes bar offers a diverse selection of nodes that can be connected to construct flows. Creating a flow is as simple as dragging nodes from the Nodes bar to the workspace and connecting them together.

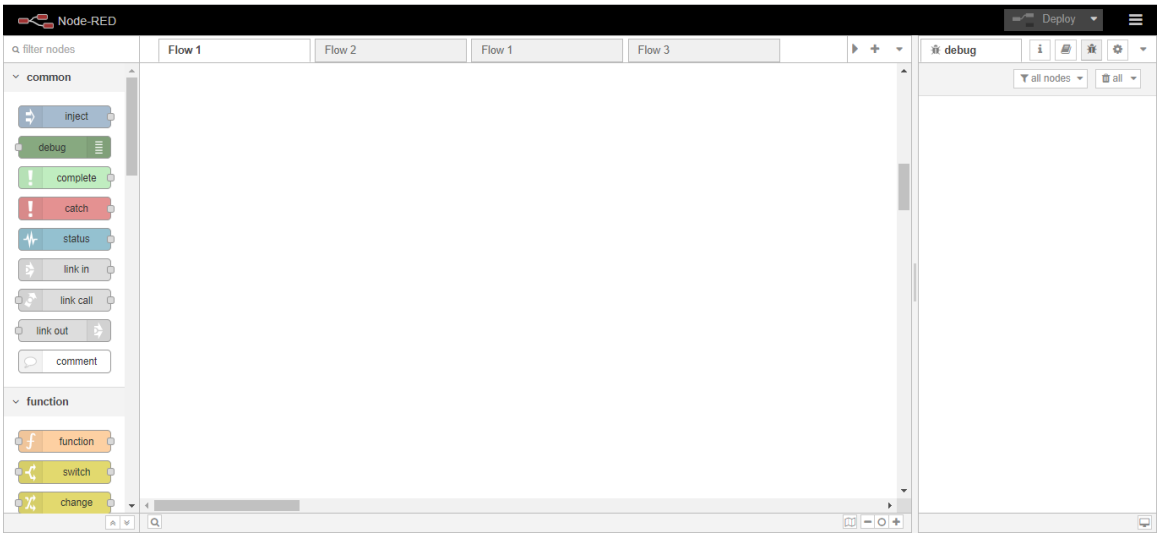


Figure 4.1: The Node-Red Platform

4.3.1 Nodes

Nodes in Node-RED are building blocks that perform specific tasks. They can be connected together to create flows for automation and data processing. Nodes represent different functionalities.

The input nodes in the Node-RED application are denoted by grey boxes on the right-hand side of the node, allowing them to receive input. The process nodes have boxes on both sides, enabling them to take input data and produce output. On the other hand, the output nodes only have boxes on the left side. (Figure 4.2) illustrates the various types of nodes.

By combining nodes, you can easily design and develop applications without traditional coding

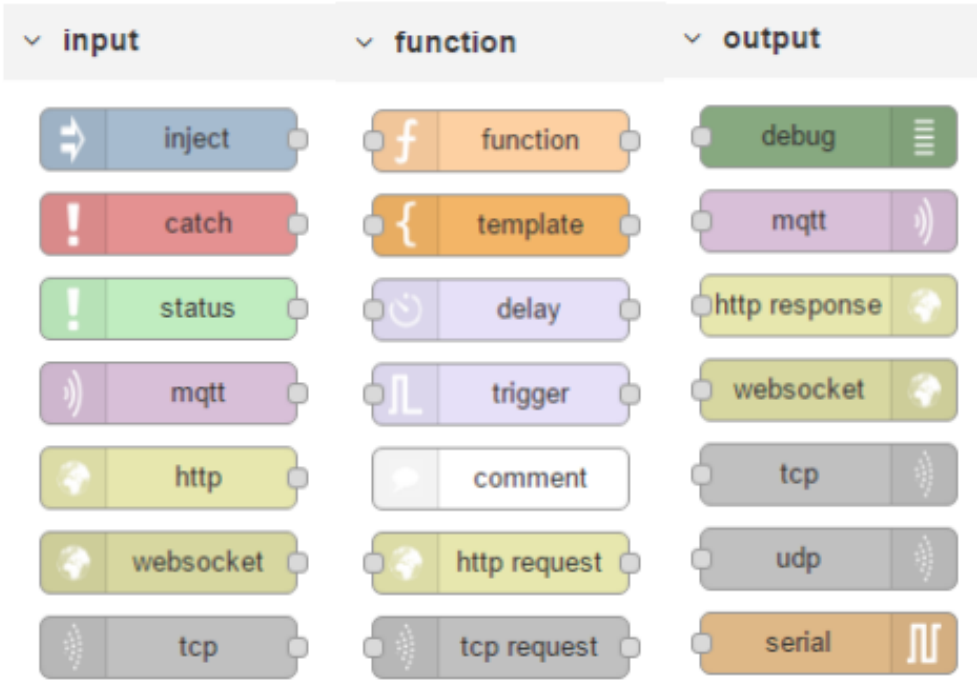


Figure 4.2: Some of the nodes that are available in Node-Red

4.3.2 Flow

A flow is defined as a set of nodes that have their own independent work cycle, that run in parallel with other flows. This combined with globally shared data, makes it possible to communicate with nodes and a flow without having a connection between them.

4.4 Junction Layer Implementation & Simulation

In the first part, we utilized data from the WISDM dataset and incorporated several nodes in our Node-RED flow. The **"timestamp"** node triggers the flow execution at specified intervals. The **"function 2"** node generates a random number for switching the flow path, which is handled by the **"switch"** node. The **"file in"** node reads CSV files from specific paths, while the **"function"** node formats the header and payload

for an HTTP request. The **"http request"** node sends the formatted payload as an HTTP POST request to a local URL. The **"debug"** node displays the response from the HTTP request.
Overall, this flow involves reading CSV files, formatting the data, and sending it as an HTTP request payload, with the response being displayed in the debug sidebar.

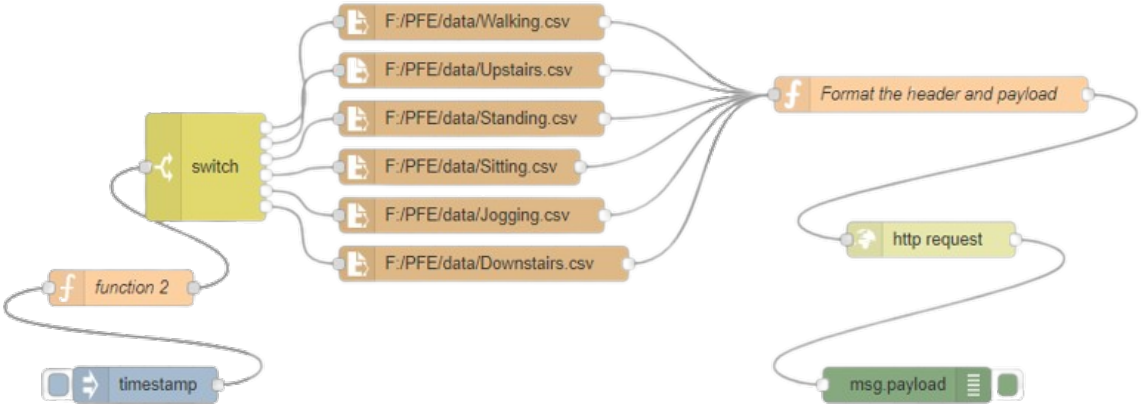


Figure 4.3: First Part of Simulation and Implementation of Junction Layer

And for the second part of the simulation, the person's location (either on the bed or outside the bed) is chosen randomly. The human movement and location data are visualized and sent to a pre-existing database for storage. The results are displayed on the sidebar along with an image. The **"Image Viewer"** node is responsible for displaying the image based on the input data. The **"Join"** node combines the outputs of different branches into a single message. The **"MySQL database"** node configures the connection to the MySQL database, specifying details such as host, port, and database name. It updates the data every 5 seconds. The **"function 1"** node generates a random number between (0 and 1) whether the patient is in bed or not. The **"function 3"** node contains JavaScript code to determine the person's location and activity. It sets properties like "msg.image", "msg.payload", and "msg.topic" to update the database's patient-statup table. The **"function 4"** node prepares an SQL

query to update the Activity table based on a specific idps.
Finally, the **"debug"** node displays the output payload for debugging purposes, showing the person's activity, the result of the **"http request"** node, and the corresponding location, indicating whether it's safe or dangerous, and the **"telegram sender** to send a message to the nurse or the responsible person when there is danger at the patient health. The image reflects the normal or dangerous state, and all these results are sent to and updated in the database.

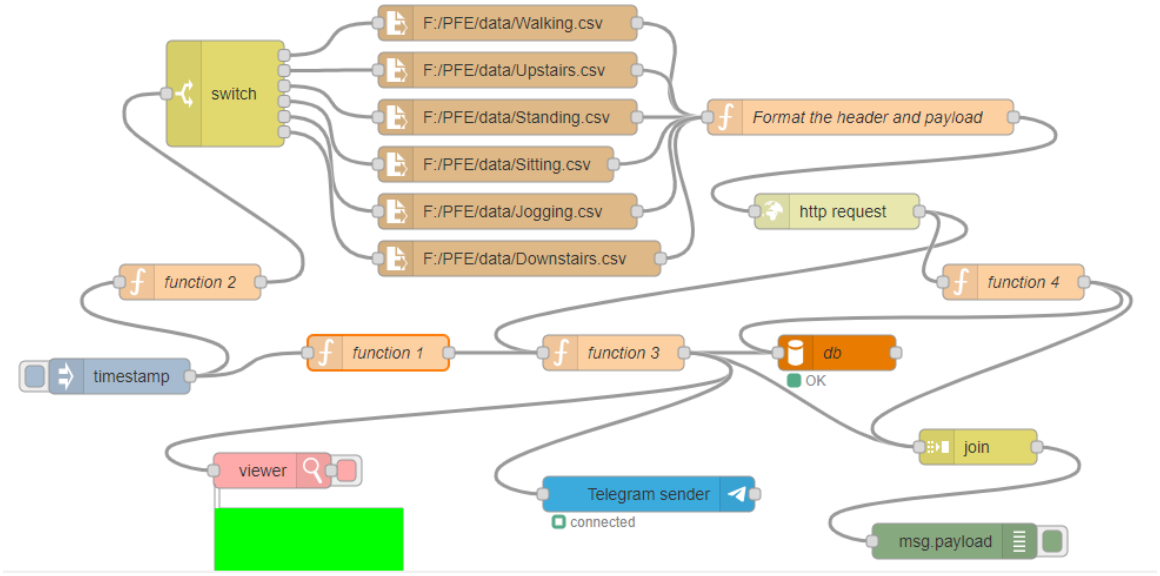


Figure 4.4: First Part of Simulation and Implementation of Junction Layer

4.5 Web Application

In the web application section, we present a comprehensive overview of the web interface developed for our system. The web application serves as the user-facing component, providing an intuitive and interactive platform are designed using HTML5 and styled with CSS3, for users to access and utilize the system's features. It includes features such as user authentication (Figure 4.5), dashboard (Figure 4.6) and result display panels (Figs from 4.7 to 4.9). These panels are dynamically generated using

PHP and MySQL.

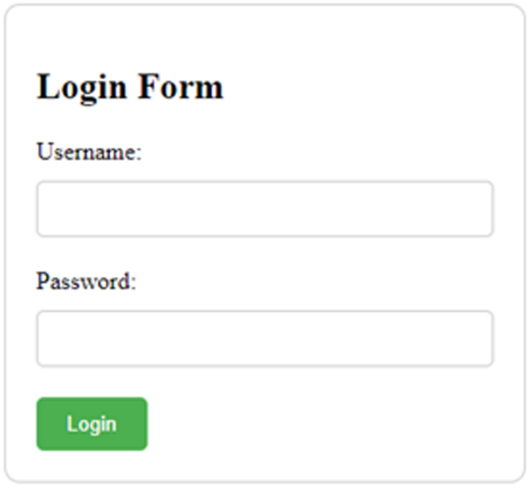


Figure 4.5: Web Application Authentication

User authentication is implemented to ensure secure access and personalized experiences for users. User authentication involves verifying the identity of users before granting them access to the application's features and resources.



Figure 4.6: Web Application Dashboard

The dashboard in the web application provides an overview of different patients, displaying their individual status to facilitate easy access and monitoring. This dashboard serves as a centralized hub where users can view and manage multiple patient profiles simultaneously.

The dashboard interface presents a visual representation of each patient, typically in the form of tiles or cards. Each tile contains relevant information about the patient, such as their name, profile picture, and current status.

Different Statuses of Patient



Figure 4.7: State of Patient Safe in the Bed

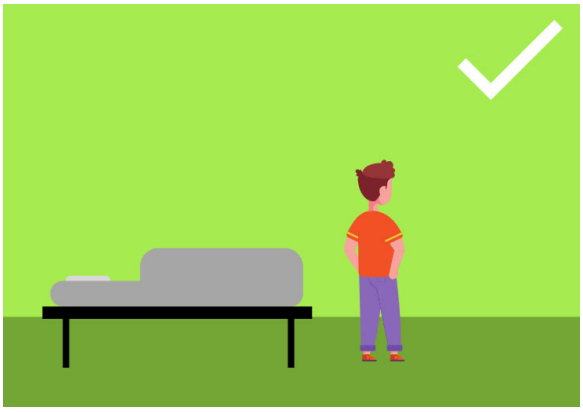


Figure 4.8: State of Patient Safe Out of the Bed



Figure 4.9: State of Patient Danger Out of the Bed

Overall, the web application section provides an in-depth exploration of the interface that enables users to interact with the system effectively.

4.6 Conclusion

Chapter 4 showcases the practical implementation of the Junction Layer and the development of a web application within our HAR system. By seamlessly processing data and fostering intuitive engagements, these components play a crucial role in achieving accurate recognition and enhancing the user experience. Through meticulous precision and a focus on user-centricity, our system creates a captivating environment where human activity is analyzed and visualized effectively. The successful integration of these elements brings us closer to our goal of advancing the field of Human Activity Recognition.

General Conclusion

This thesis has focused on the development and evaluation of the Multi-Branch Convolutional LSTM Network (MBCLNet) as a powerful deep learning model for HAR. Through extensive experimentation and analysis using benchmark datasets, namely WISDM and PAMAP2, our model has demonstrated exceptional accuracy rate of 99.98% on the WISDM dataset and 99.95% on the PAMAP2 dataset. These remarkable accuracy rates validate the effectiveness of the MBCLNet model in accurately identifying and classifying activities. Moreover, the integration of MBCLNet into an application programming interface (API) enables real-time processing and user interface interactions. This integration enhances monitoring and supervision in healthcare environments, providing caregivers and medical professionals with valuable insights and timely assistance. The utilization of HAR technologies in healthcare settings has the potential to improve control, care, and overall well-being for patients and elderly individuals.

In anticipation of future research directions, there exist numerous promising avenues to explore. One such avenue involves expanding the range and diversity of datasets, as doing so will greatly contribute to the generalizability and robustness of Human Activity Recognition models. By incorporating data from various sources, including sensor fusion from multiple sensors and wearable devices, there is substantial potential to enhance the accuracy and effectiveness of HAR models. This, in turn, enables continuous and unobtrusive monitoring of activities, allowing for a more comprehensive understanding of individuals' behaviors. Additionally, investing in the development and augmentation of the user interface can further improve the usability and accessibility of HAR systems, ensuring a seamless and intuitive experience for caregivers, medical professionals, and end-users. By focusing on these areas, future research

endeavors in HAR can pave the way for innovative advancements in healthcare and contribute to the overall well-being of individuals in various settings.

Bibliography

[1] Saedeh Abbaspour, Faranak Fotouhi, Ali Sedaghatbaf, Hossein Fotouhi, Maryam Vahabi, and Maria Linden. A comparative analysis of hybrid deep learning models for human activity recognition. *Sensors*, 20(19):5707, 2020.

[2] Maher GM Abdolrasol, SM Suhail Hussain, Taha Selim Ustun, Mahidur R Sarker, Mahammad A Hannan, Ramizi Mohamed, Jamal Abd Ali, Saad Mekhilef, and Abdalrhman Milad. Artificial neural networks based optimization techniques: A review. *Electronics*, 10(21):2689, 2021.

[3] Atul. Edureka. `https://www.edureka.co/blog/ai-vs-machine-learning-vs-deep-learning/`. Accessed: May 31, 2023.

[4] Debarshi Bhattacharya, Deepak Sharma, Wonjoon Kim, Muhammad Fazal Ijaz, and Pawan Kumar Singh. Ensem-har: An ensemble deep learning model for smartphone sensor-based human activity recognition for measurement of elderly health monitoring. *Biosensors*, 12(6):393, 2022.

[5] Jason Brownlee. A gentle introduction to transfer learning for deep learning. *Machine Learning Mastery*, 20, 2017.

[6] Sravan Kumar Challa, Akhilesh Kumar, and Vijay Bhaskar Semwal. A multibranch cnn-bilstm model for human activity recognition using wearable sensor data. *The Visual Computer*, 38(12):4095–4109, 2022.

[7] Jing Chen, Qingcai Chen, Xin Liu, Haijun Yang, Daohe Lu, and Buzhou Tang. The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 4946–4951, 2018.

[8] Liming Chen, Jesse Hoey, Chris D Nugent, Diane J Cook, and Zhiwen Yu. Sensor-based activity recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):790–808, 2012.

[9] Shuangshuang Chen and Wei Guo. Auto-encoders in deep learninga review with new perspectives. *Mathematics*, 11(8):1777, 2023.

[10] Nurul Amin Choudhury, Soumen Moulik, and Diptendu Sinha Roy. Physique-based human activity recognition using ensemble learning and smartphone sensors. *IEEE Sensors Journal*, 21(15):16852–16860, 2021.

[11] N Computation. Long short-term memory. *Neural Comput*, 9:1735–1780, 2016.

[12] Ha Manh Do, Minh Pham, Weihua Sheng, Dan Yang, and Meiqin Liu. Rish: A robot-integrated smart home for elderly care. *Robotics and Autonomous Systems*, 101:74–92, 2018.

[13] Nidhi Dua, Shiva Nand Singh, and Vijay Bhaskar Semwal. Multi-input cnn-gru based human activity recognition using wearable sensors. *Computing*, 103:1461–1478, 2021.

[14] AG Farizawani, M Puteh, Y Marina, and A Rivaie. A review of artificial neural network learning rule based on multiple variant of conjugate gradient approaches. In *Journal of Physics: Conference Series*, volume 1529, page 022040. IOP Publishing, 2020.

[15] Python Software Foundation. Python. `https://www.python.org/doc/essays/blurb/`. Accessed: May 31, 2023.

[16] Himanshu Gore, Rakesh Kumar Singh, Ashutosh Singh, Arnav Pratap Singh, Mohammad Shabaz, Bhupesh Kumar Singh, and Vishal Jagota. Django: Web development simple & fast. *Annals of the Romanian Society for Cell Biology*, 25(6):4576–4585, 2021.

[17] Yu Guan and Thomas Plötz. Ensembles of deep lstm learners for activity recognition using wearables. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, 1(2):1–28, 2017.

[18] Taiji Hagino. *Practical Node-RED Programming: Learn powerful visual programming techniques and best practices for the web and IoT*. Packt Publishing Ltd, 2021.

[19] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[21] Zawar Hussain, Michael Sheng, and Wei Emma Zhang. Different approaches for human activity recognition: A survey. *arXiv preprint arXiv:1906.05074*, 2019.

[22] Duy Tâm Gilles Huynh. Human activity recognition with wearable sensors. *Technische Universität Darmstadt*, pages 59–65, 2008.

[23] Andrey Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922, 2018.

[24] Md Islam et al. A deep study of artificial intelligence: Machine learning in the browser using tensorflow. 2021.

[25] H Jabbar and Rafiqul Zaman Khan. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). *Computer Science, Communication and Instrumentation Devices*, 70:163–172, 2015.

[26] Ameet V Joshi. Machine learning and artificial intelligence. 2020.

[27] Harcharan Kabbay. Artificial neural network concepts and examples. 2022.

[28] J Kaur Gill. Automatic log analysis using deep learning and ai. *Accès https://www. xenonstack. com/blog/log-analytics-deep-machine-learning*, 2018.

[29] Talia B Kimber, Yonghui Chen, and Andrea Volkamer. Deep learning in virtual screening: recent applications and developments. *International Journal of Molecular Sciences*, 22(9):4435, 2021.

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[31] Hong-Lam Le, Duc-Nhan Nguyen, Thi-Hau Nguyen, and Ha-Nam Nguyen. A novel feature set extraction based on accelerometer sensor data for improving the fall detection system. *Electronics*, 11(7):1030, 2022.

[32] Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5457–5466, 2018.

[33] Haojie Ma, Wenzhong Li, Xiao Zhang, Songcheng Gao, and Sanglu Lu. Attnsense: Multi-level attention mechanism for multimodal human activity recognition. In *IJCAI*, pages 3109–3115, 2019.

[34] Sonali B Maind, Priyanka Wankar, et al. Research paper on basic of artificial neural network. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(1):96–100, 2014.

[35] Sakorn Mekruksavanich and Anuchit Jitpattanakul. Multimodal wearable sensing for sport-related activity recognition using deep learning networks. *Journal of Advances in Information Technology*, 2022.

[36] Srinath Obla, Xinghan Gong, Asma Aloufi, Peizhao Hu, and Daniel Takabi. Effective activation functions for homomorphic evaluation of deep neural networks. *IEEE Access*, 8:153098–153112, 2020.

[37] OpenJS Foundation, Contributors. Node-RED. `https://www.nodered.org`. Accessed: 2023-05-23.

[38] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[39] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[40] Shyamal Patel, Richard Hughes, Todd Hester, Joel Stein, Metin Akay, Jennifer Dy, and Paolo Bonato. Tracking motor recovery in stroke survivors undergoing rehabilitation using wearable technology. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 6858–6861. IEEE, 2010.

[41] AA Prayogi, M Niswar, M Rijal, et al. Design and implementation of rest api for academic information system. In *IOP Conference Series: Materials Science and Engineering*, volume 875, page 012047. IOP Publishing, 2020.

[42] Wen Qi, Hang Su, Fei Chen, Xuanyi Zhou, Yan Shi, Giancarlo Ferrigno, and Elena De Momi. Depth vision guided human activity recognition in surgical procedure using wearable multisensor. In *2020 5th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 431–436. IEEE, 2020.

[43] Rinku Datta Rakshit, Dakshina Ranjan Kisku, Phalguni Gupta, and Jamuna Kanta Sing. Cross-resolution face identification using deep-convolutional neural network. *Multimedia Tools and Applications*, 80:20733–20758, 2021.

[44] Raúl Gómez Ramos, Jaime Duque Domingo, Eduardo Zalama, Jaime Gómez-García-Bermejo, and Joaquín López. Sdhar-home: A sensor dataset for human activity recognition at home. *Sensors*, 22(21):8109, 2022.

[45] Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th international symposium on wearable computers*, pages 108–109. IEEE, 2012.

[46] Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Tutorials*, pages 15–18, 2019.

[47] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[48] Balaji Sathelly. *An Artificial Neural Network Approach to Predict Liver Failure Likelihood*. The University of Toledo, 2018.

[49] Jessica Sena, Jesimon Barreto, Carlos Caetano, Guilherme Cramer, and William Robson Schwartz. Human activity recognition based on smartphone and wearable sensors using multiscale dcnn ensemble. *Neurocomputing*, 444:226–243, 2021.

[50] Fatemeh Serpush, Mohammad Bagher Menhaj, Behrooz Masoumi, and Babak Karasfi. Wearable sensor-based human activity recognition in the smart healthcare system. *Computational intelligence and neuroscience*, 2022, 2022.

[51] Victor RL Shen, Chin-Shan Wei, and Tony Tong-Ying Juang. Javascript malware detection using a high-level fuzzy petri net. In *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 2, pages 511–514. IEEE, 2018.

[52] Alex Smola and SVN Vishwanathan. Introduction to machine learning. *Cambridge University, UK*, 32(34):2008, 2008.

[53] Qi Teng, Kun Wang, Lei Zhang, and Jun He. The layer-wise training convolutional neural networks using local loss for sensor-based human activity recognition. *IEEE Sensors Journal*, 20(13):7265–7274, 2020.

[54] Yu-Ru Tsai. Medium. `https://medium.com/ai-academy-taiwan/what-are-autoencoders-175b474d74d1`. Accessed: May 27, 2023.

[55] Haritha Vellampalli. Physical human activity recognition using machine learning algorithms. 2017.

[56] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[57] Kishor H Walse, Rajiv V Dharaskar, and Vilas M Thakare. A study of human activity recognition using adaboost classifiers on wisdm dataset. *The Institute of Integrative Omics and Applied Biotechnology Journal*, 7(2):68–76, 2016.

[58] Shaohua Wan, Lianyong Qi, Xiaolong Xu, Chao Tong, and Zonghua Gu. Deep learning models for real-time human activity recognition with smartphones. *Mobile Networks and Applications*, 25:743–755, 2020.

[59] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern recognition letters*, 119:3–11, 2019.

[60] Kun Xia, Jianguang Huang, and Hanyu Wang. Lstm-cnn architecture for human activity recognition. *IEEE Access*, 8:56855–56866, 2020.

[61] Shoujiang Xu, Qingfeng Tang, Linpeng Jin, and Zhigeng Pan. A cascade ensemble learning model for human activity recognition with smartphones. *Sensors*, 19(10):2307, 2019.

[62] Jure Zupan. Introduction to artificial neural network (ann) methods: what they are and how to use them. *Acta Chimica Slovenica*, 41:327–327, 1994.