People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Amar Telidji - Laghouat
Faculty Of Science and Technology

Department of Electronics
Specialty: Instrumentation

Final-year project
(Master's degree)

Medical images classification using CNN

Presented by:
Ziane Saif Elddine

Supervised by:
Zitouni Abdelkader

Academic Year: 2024 / 2025

Dedication

"To my beloved family, mentors, and friends, whose unwavering support and encouragement have been my greatest source of strength."

Acknowledgments

I wish to express my sincere gratitude to my supervisor, Dr. Abdelkader Zitouni , for his unwavering support, insightful guidance, and continuous encouragement throughout the course of this research. His expertise and constructive feedback have been invaluable at every stage of this work.

I am also deeply thankful to the faculty members and staff of the Department of Electronics at Ammar Telidji University for creating a productive academic environment and for providing the necessary resources that made this study possible.

My heartfelt appreciation goes to my family for their patience, sacrifices, and constant belief in me. Their love and encouragement have been a source of strength during the most challenging moments of this journey.

Finally, I would like to extend my thanks to my friends and colleagues, whose discussions, advice, and moral support greatly enriched this experience.

ملخص

تبحث هذه المذكرة في تطبيق التعلم العميق للتشخيص الطبي الحيوي الآلي من خلال الشبكات العصبية التلافيفية (CNNs)، حيث تدمج الأسس النظرية مع التطبيق العملي. تبدأ بمراجعة مبادئ الذكاء الاصطناعي والتعلم العميق الأساسية، مع تركيز خاص على بنية الشبكة العصبية التلافيفية وفعاليتها في استخراج الميزات من كل من بيانات الصور والإشارات. بالنسبة للجانب العملي، تم تنفيذ نموذجين مخصصين: نموذج 1D-CNN لتصنيف خمسة أنواع من عدم انتظام ضربات القلب من إشارات تخطيط القلب الكهربائي (ECG) في قاعدة بيانات MIT-BIH ، ونموذج CNN لتصنيف الأورام من صور الرنين المغناطيسي للدماغ. بعد المعالجة المسبقة الشاملة والتدريب لمعالجة عدم التوازن الكبير في الفئات، أظهر النموذجان أداءً عاليًا. حقق مصنف تخطيط القلب دقة و f1-score مقدارها **0.99**، بينما حقق مصنف أورام الدماغ دقة مقدارها **0.91**. تسلط هذه النتائج الضوء على متانة نماذج التعلم العميق في التحليل الطبي الحيوي وإمكاناتها الكبيرة في دعم عمليات التشخيص السريري.

الكلمات المفتاحية: التعلم العميق، الشبكات العصبية التلافيفية، تخطيط القلب الكهربائي، الأورام الدماغية، التصنيف الطبي.

Abstract

This report explores the application of deep learning for automated biomedical diagnosis through Convolutional Neural Networks (CNNs), integrating theoretical foundations with practical application. It begins with a review of essential AI and deep learning principles, with particular emphasis on CNN architecture and its effectiveness in feature extraction for both image and signal data. For the practical component, two dedicated models were implemented: a 1D-CNN to classify five types of cardiac arrhythmias from ECG signals in the MIT-BIH database, and a CNN to classify tumors from brain MRI scans. Following comprehensive preprocessing and training to address significant class imbalance, the models demonstrated high performance. The ECG classifier achieved an accuracy and F1-score of **0.99**, while the brain tumor classifier achieved an accuracy of **0.91**. These results highlight the robustness of deep learning models in biomedical analysis and their significant potential in supporting clinical diagnostic processes.

Keywords: Deep Learning, Convolutional Neural Networks, ECG, Brain Tumor, Medical Image Classification.

Résumé

Ce rapport explore l'application de l'apprentissage profond pour le diagnostic biomédical automatisé à travers les réseaux de neurones convolutionnels (CNN), en intégrant des fondements théoriques à une application pratique. Il commence par une revue des principes essentiels de l'IA et de l'apprentissage profond, en mettant un accent particulier sur l'architecture des CNN et son efficacité pour l'extraction de caractéristiques depuis des données both d'images et de signaux. Pour la composante pratique, deux modèles dédiés ont été implémentés : un réseau 1D-CNN pour classer cinq types d'arythmies cardiaques à partir de signaux ECG de la base de données MIT-BIH, et un CNN pour classer les tumeurs à partir d'IRM cérébrales. Suite à un prétraitement complet et un entraînement visant à résoudre un déséquilibre de classes significatif, les modèles ont démontré des performances élevées. Le classificateur d'ECG a atteint une exactitude (accuracy) et un score F1 de **0.99**, tandis que le classificateur de tumeurs cérébrales a atteint une exactitude de **0.91**. Ces résultats soulignent la robustesse des modèles d'apprentissage profond dans l'analyse biomédicale et leur potentiel significatif pour soutenir les processus de diagnostic clinique.

Mots-clés: Apprentissage profond, Réseaux de neurones convolutionnels, ECG, Tumeurs cérébrales, Classification médicale.

Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
ECG	Electrocardiogram
MRI	Magnetic Resonance Imaging
VGG16	Visual Geometry Group 16-layer network
GPU	Graphics Processing Unit
SSD	Solid State Drive
NIH	National Institutes of Health
AAMI	Association for the Advancement of Medical Instrumentation
PVC	Premature Ventricular Contraction
ReLU	Rectified Linear Unit
L2	L2 Regularization (Weight Decay)
Adam	Adaptive Moment Estimation (Optimizer)
N	Normal beat (sinus rhythm)
L	Left bundle branch block beat
R	Right bundle branch block beat

V	Premature ventricular contraction beat
/	Paced beat (artificial pacemaker activation)

Chapter 1

Foundations and Core Concepts of Artificial Intelligence

1.1 Introduction

In recent decades, artificial intelligence (AI) has evolved from a conceptual ambition into a transformative force across numerous industries, including healthcare [24, 3]. AI refers to the ability of machines to perform tasks that typically require human intelligence, such as reasoning, learning, and problem-solving [25]. This technological evolution has been fueled by advances in algorithms, the exponential growth of computational power, and the availability of vast datasets interconnected through high-speed networks [13].

A major subfield of AI is machine learning (ML), which enables systems to recognize patterns in data and improve performance over time without explicit programming [3]. Within ML, deep learning (DL) has emerged as a particularly powerful approach, capable of extracting complex features automatically from raw data using multi-layered artificial neural networks [25]. Convolutional Neural Networks (CNNs), among the most successful DL architectures, have demonstrated exceptional performance in fields such as computer vision, natural language processing, and biomedical signal processing .

In the medical field, AI — and DL in particular — has enabled significant advances in diagnostic support [2]. Automated analysis of biomedical signals, including electrocardiograms (ECG), provides an opportunity to improve efficiency, reduce human error, and ensure consistent detection of abnormalities [20, 21]. However, biomedical data present inherent challenges such as variability across patients, signal noise, and the need for highly precise interpretation to detect subtle pathological patterns.

This thesis focuses on the classification of ECG signals using deep learning techniques, with the aim of developing an accurate and reliable automatic classification model that minimizes manual intervention and improves diagnostic quality.

The objectives of this work can be summarized as follows:

- Provide an overview of Artificial Intelligence, Machine Learning, and Deep Learning concepts, emphasizing their relevance to biomedical signal analysis [24, 25].
- Design and implement a deep learning model, based on Convolutional Neural Networks (CNNs), to classify ECG signals effectively .
- Evaluate the model using standard performance metrics such as accuracy, precision, recall, and F1-score .
- Compare the proposed model with traditional machine learning approaches to highlight its advantages and limitations .
- Discuss the findings and identify potential improvements for future research.

1.2 Overview

Artificial Intelligence (AI) has evolved from an abstract idea into a transformative force influencing nearly every sector of modern life [24]. It represents a multidisciplinary field that combines computer science, mathematics, logic, and cognitive science to design machines capable of performing tasks that require human-like reasoning, decision-making, and problem-solving [3]. This chapter provides a broad understanding of AI, including its key principles, definition, major application areas, and current limitations.

1.3 Fundamental Principles of AI

The development of AI systems is guided by several foundational principles that enable them to perform intelligently:

- **Perception:** acquiring and interpreting information from the environment through sensors or data inputs [24].
- **Reasoning:** processing available information to draw logical conclusions and make decisions.
- **Learning:** improving performance by analyzing previous experiences and adapting to new situations [25].
- **Autonomy:** operating with minimal human intervention while maintaining accuracy and reliability [5].
- **Goal Orientation:** focusing actions toward achieving defined objectives while optimizing resources [1].

1.4 Definition and Scope

Artificial Intelligence can be defined as the capability of a system to perform functions that would normally require human intelligence [24]. It encompasses diverse techniques, from symbolic rule-based systems to probabilistic reasoning methods [3], all designed to solve problems in dynamic and uncertain environments.

1.5 Major Applications of AI

Artificial Intelligence has widespread applications across multiple industries. Some of the most impactful areas include:

1.5.1 Healthcare and Medical Diagnosis

AI is used to analyze diagnostic images, predict disease progression, recommend treatment plans, and process electronic medical records. Intelligent systems assist healthcare professionals by providing faster and more accurate decisions [2].

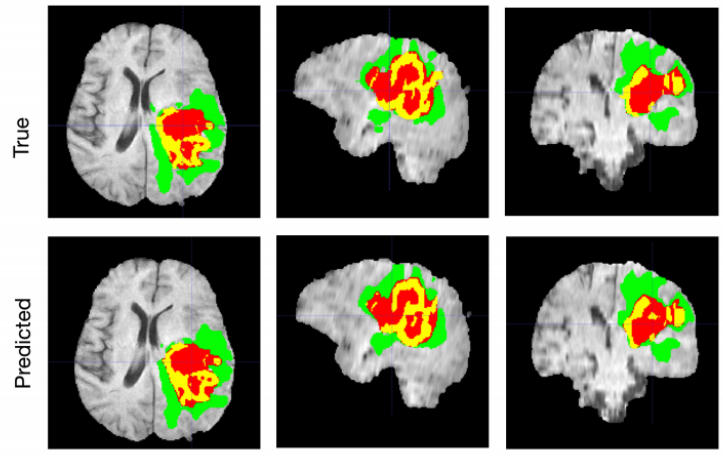

Figure 1.1: Example of artificial intelligence applied to brain tumor detection [26]

1.5.2 Banking and Financial Services

In finance, AI supports fraud detection, credit risk assessment, investment portfolio management, and automated trading. These systems process large datasets in real time, improving both security and efficiency [4].

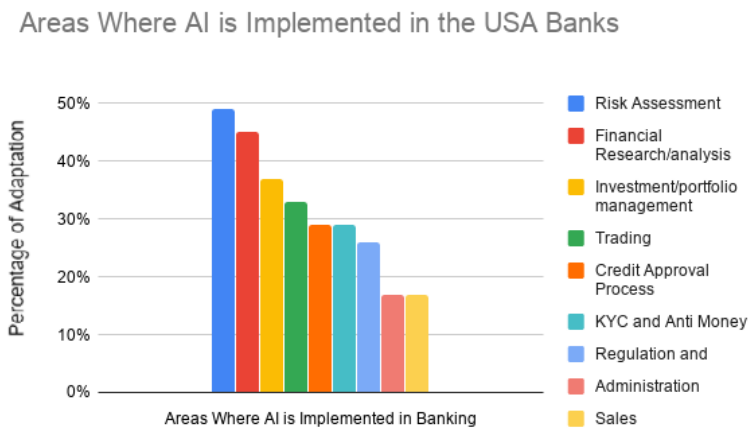

Figure 1.2: Distribution of AI applications in banking [27]

1.5.3 Intelligent Transportation and Autonomous Vehicles

Self-driving systems rely on AI to perceive surroundings, recognize objects, and make safe navigation decisions. This technology integrates data from cameras, radar, and sensors to ensure reliability and efficiency on the road [5].

Figure 1.3: Amazon Zoox self-driving robotaxi [28]

1.5.4 Commerce and Retail Systems

Retailers use AI to forecast demand, manage inventory, personalize product recommendations, and analyze customer behavior. These applications improve service quality and operational efficiency [6].

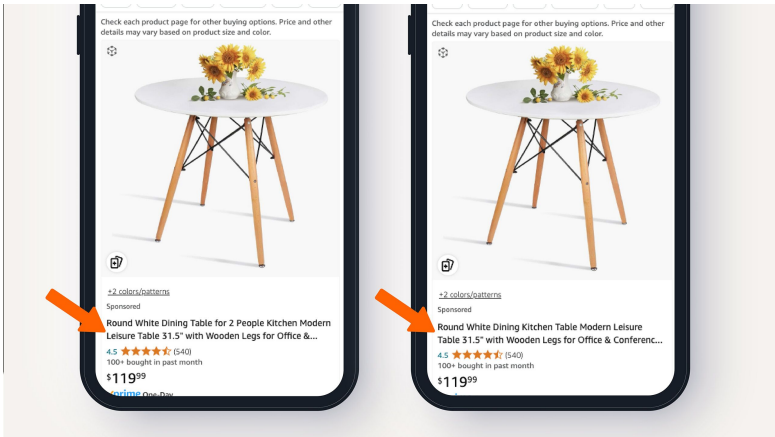

Figure 1.4: Amazon AI personalizes product recommendations and descriptions [29]

1.5.5 Industrial Production and Automation

In manufacturing, AI improves quality control, predictive maintenance, supply chain optimization, and production planning. Smart automation systems increase productivity while minimizing downtime [8].

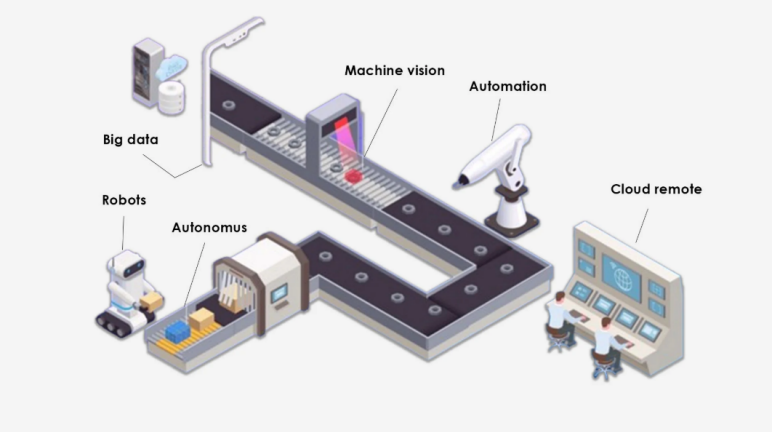

Figure 1.5: Manufacturing with Artificial Intelligence [30]

1.6 Key Challenges and Limitations

Despite its rapid growth, AI faces several obstacles:

- **Data Dependence:** high-quality and large datasets are required for accurate performance.
- **Lack of Transparency:** many AI models function as "black boxes," making their decision-making process difficult to interpret.
- **Limited Generalization:** AI systems often perform well only in specific tasks and fail to adapt to new domains.

- **Ethical and Privacy Concerns:** issues related to fairness, accountability, and the protection of sensitive information remain unresolved.
- **Resource Intensive:** developing and deploying advanced AI models requires substantial computational power and energy.

1.7 Concluding Remarks

Artificial Intelligence represents a cornerstone of modern technology, providing innovative solutions in medicine, finance, agriculture, manufacturing, and transportation. Understanding its principles, scope, and applications helps to evaluate both its potential and its current constraints. This chapter establishes a foundation for the subsequent exploration of more specialized concepts and techniques.

Chapter 2

Tools and Algorithms Used for Artificial Intelligence

Introduction

Artificial Intelligence (AI) has evolved beyond a theoretical concept to become a key technological foundation in solving real-world problems across multiple disciplines [24]. However, the capabilities of AI systems rely not only on theoretical concepts but also on concrete tools and advanced algorithms that enable machines to learn from data, adapt to new situations, and make decisions [3].

This chapter delves into the essential components required to implement AI solutions effectively. We begin by exploring the main learning paradigms, including machine learning and its different types: supervised, unsupervised, and reinforcement learning [25, 11]. These learning models constitute the core of most intelligent systems, allowing machines to analyze data, detect patterns, and make predictions.

We will then shift our focus to deep learning — a subfield of machine learning — which has significantly transformed areas like image classification, natural language processing, and medical diagnostics [13]. The chapter will explain the structure and functioning of artificial neural networks, especially deep neural architectures, which emulate the behavior of the human brain in processing information [23].

In addition, this chapter introduces widely-used programming tools and software libraries, such as TensorFlow, Keras, and PyTorch, that facilitate the development and deployment of AI models. Cloud-based platforms like Google Colab also play an important role in democratizing access to computational resources.

Overall, this chapter provides the foundational knowledge necessary for understanding and building intelligent systems using both classical and modern AI tools.

2.1 Machine Learning Techniques

Machine learning is a fundamental sub-field of artificial intelligence that aims to develop algorithms capable of learning from data and making decisions without being explicitly programmed for every specific task [3]. This section delves into the core concepts and types of machine learning, which are foundational for many AI systems today [24].

2.1.1 Definition of Machine Learning

Machine Learning (ML) is defined as a branch of artificial intelligence that focuses on building systems that learn from data and improve their performance with experience. Rather than being explicitly programmed, ML systems use statistical techniques to infer patterns and make predictions or decisions.[1]

2.1.2 Types of Learning

Machine Learning (ML) techniques can be broadly categorized into several types based on how the algorithm learns from the data it is provided. The main categories are supervised learning, unsupervised learning, and reinforcement learning [3, 25].

Supervised learning is a machine learning approach in which models are trained on labeled datasetshe structure can be observed in Figure 2.1 he structure can be observed in Figure, meaning that each input is paired with a known output. The goal is to learn a mapping from inputs to outputs that can generalize to unseen data. This method is primarily used for classification tasks, where data are assigned to discrete categories, and regression tasks, where continuous values are predicted. By minimizing the difference between predicted and actual results, supervised learning enables accurate and reliable decision-making in applications such as spam filtering, medical diagnosis, and financial forecasting [9].

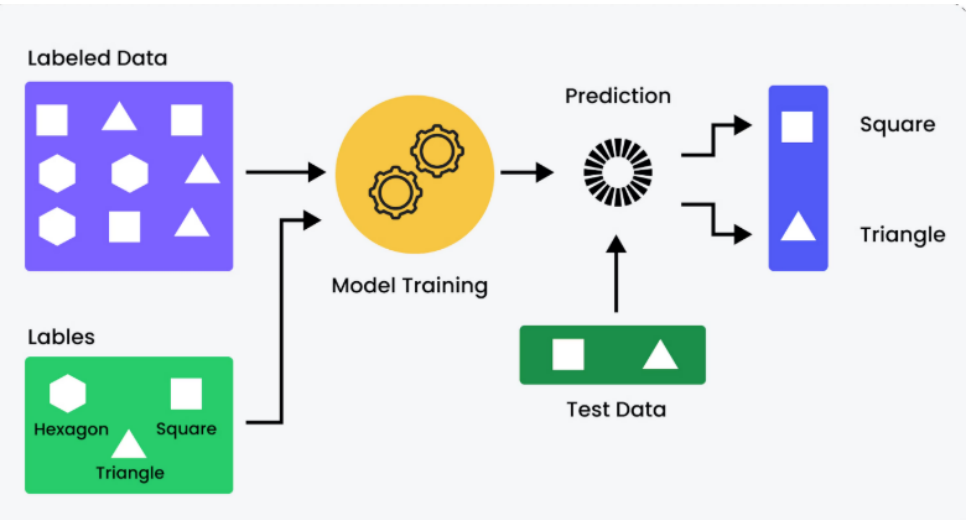

Figure 2.1: Supervised learning overview [31]

Applications: Supervised learning is commonly used for classification tasks (e.g., spam detection, image recognition) and regression tasks (e.g., predicting house prices).

Unsupervised learning : is a machine learning approach where algorithms are trained on unlabeled data he structure can be observed in Figure 2.2 , meaning no predefined outputs are provided. The aim is to identify hidden structures, patterns, or groupings within the data without external guidance. This method is commonly used for tasks such as clustering, where similar data points are grouped together, and dimensionality reduction, where data is simplified while preserving essential information. Unsupervised learning is widely applied in areas like customer segmentation, anomaly detection, and exploratory data analysis [10].

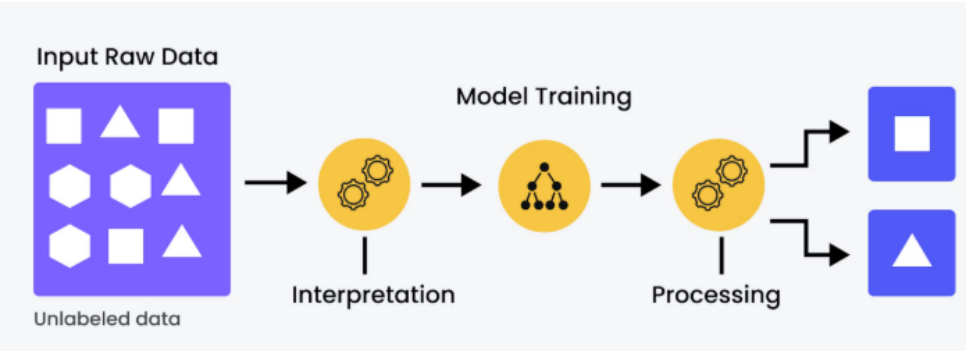

Figure 2.2: Unsupervised learning overview [31]

Goals of Unsupervised Learning:

- Discover hidden patterns
- Group similar data points
- Reduce data dimensionality for visualization

Reinforcement learning is a machine learning approach in which an agent learns to make decisions by interacting with an environment. Instead of being trained on labeled data, the agent receives feedback in the form of rewards or penalties based on its actions. Over time, it learns a strategy, or policy, that maximizes cumulative reward. This method is widely used for applications such as robotics, game playing, and autonomous systems, where sequential decision-making is critical [11].

Applications: Used in game playing (e.g., AlphaGo), robotics, and autonomous systems.

Learning Type	**Input Data**	**Output/Feedback**	**Use Cases**
Supervised	Labeled	Explicit feedback	Classification, Regression
Unsupervised	Unlabeled	No feedback	Clustering, Dimensionality Reduction
Reinforcement	Environment interaction	Reward signals	Game playing, Robotics

Table 2.1: Comparison of Machine Learning Types

2.2 Deep Learning

Deep learning represents a subfield of machine learning that focuses on learning hierarchical representations from data through neural networks composed of many layers [25]. It has become one of the most significant advances in artificial intelligence in recent years, especially due to its ability to automatically learn features from raw data, bypassing the need for manual feature engineering [13].

2.2.1 Artificial Neural Networks

Deep learning models are built upon networks composed of layers of interconnected processing units, commonly called neurons As shown in Figure 2.4, which perform non-linear transformations of input data [25]. The output of each layer is passed to the next, forming an *Artificial Neural Network* (ANN), a concept inspired by the structure and operation of biological neurons in the human brain [13]. An ANN consists of a large number of artificial neurons linked by weighted connections, and its depth grows with the number of intermediate layers and neurons involved.

A typical deep learning architecture can be described as follows:

- **Input layer:** Responsible for receiving the raw data (e.g., pixel intensities of an image) [23].

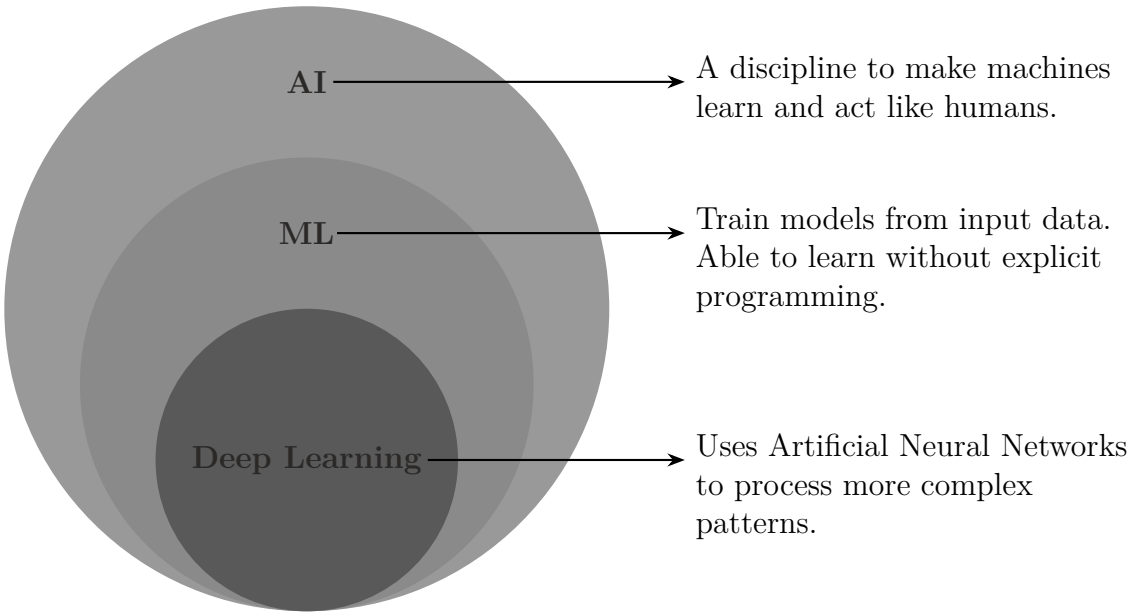

Figure 2.3: Relationship between AI, ML, and Deep Learning

- **Hidden layers:** One or more intermediate layers where neurons transform data through weighted connections, progressively extracting more complex features [25].
- **Output layer:** Generates the final predictions (e.g., class label or probability scores) [13].

Learning in an ANN involves adjusting the connection weights to minimize a loss function, typically using optimization methods such as backpropagation combined with gradient descent [12].

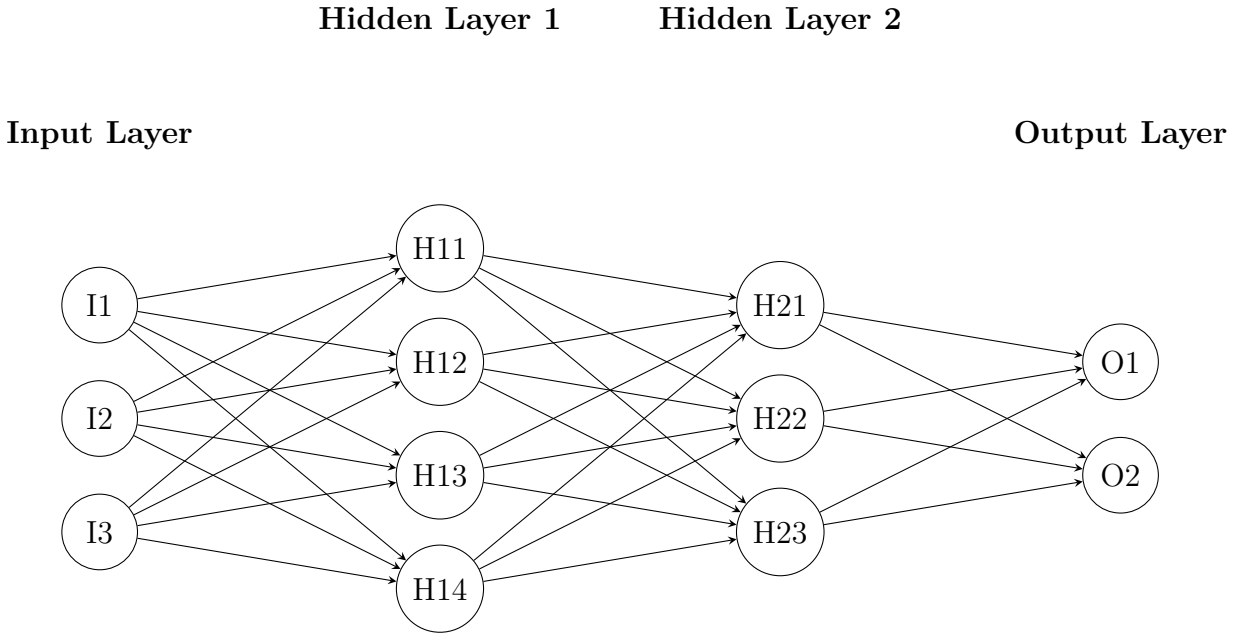

Figure 2.4: Structure of a typical Artificial Neural Network showing labeled input, hidden, and output neurons.

2.2.2 Artificial Neuron

An artificial neuron is the fundamental computational unit of an artificial neural network (ANN) As shown in Figure 2.5 , inspired by biological neurons in the human brain. It processes one or more input signals, each associated with a weight indicating its importance. The neuron calculates a weighted sum of its inputs, adds a bias term to adjust the activation threshold, and then applies an activation function—often nonlinear—to produce an output. This mechanism allows artificial neurons to model complex relationships between inputs and outputs, enabling networks to approximate nonlinear functions and generalize to unseen data. Artificial neurons are typically organized into layers, where the outputs of one layer become the inputs to the next, forming deep architectures capable of hierarchical feature extraction and decision-making [23].

Mathematically, the output y of an artificial neuron can be expressed as:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

where:

- x_i are the input signals,
- w_i are the corresponding weights,
- b is the bias term,
- f is the activation function.

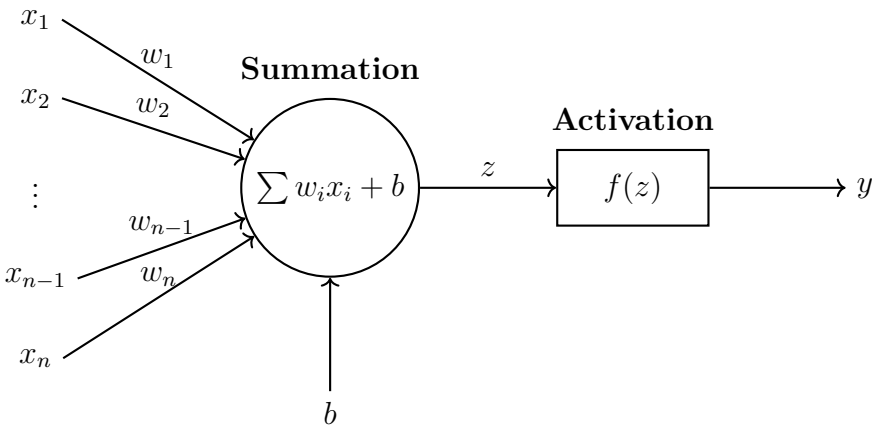

Figure 2.5: Mathematical Structure of a Artificial Neuron

2.3 Architectures of Deep Learning Neural Networks

Deep Learning neural networks are organized into different architectures, each designed to address specific types of problems by leveraging their structural properties. These architectures

consist of multiple layers of interconnected neurons, where each layer progressively extracts higher-level features from the input data. The selection of an appropriate architecture is a critical step, as it determines the network's ability to model complex relationships, generalize to unseen data, and achieve efficient performance. Below are some of the most widely used architectures:

Feedforward Neural Networks (FNNs): Also known as fully connected or dense networks, FNNs are the simplest type of neural network architecture. Data flows in one direction—from the input layer through one or more hidden layers to the output layer—without any feedback connections. These networks are primarily used for basic classification and regression tasks, but they are less efficient for high-dimensional structured data such as images or sequential signals.

Convolutional Neural Networks (CNNs): CNNs are specifically designed to process data with a grid-like topology, such as images. By employing convolutional layers, these networks automatically learn spatial hierarchies of features using shared weights (kernels), reducing the number of parameters compared to fully connected networks. Pooling layers further downsample the feature maps, while fully connected layers at the end aggregate learned features for classification. CNNs are the dominant architecture in computer vision tasks, including image recognition, medical image analysis, and object detection [13].

Recurrent Neural Networks (RNNs): RNNs are designed to handle sequential data by maintaining internal memory through recurrent connections. This enables the network to model temporal dependencies, making RNNs suitable for natural language processing, speech recognition, and time-series prediction. Variants such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) address the vanishing gradient problem, allowing for the learning of long-range dependencies [14].

Autoencoders: Autoencoders are unsupervised neural networks that learn to encode input data into a compressed latent representation and reconstruct it back to its original form. They are widely used for dimensionality reduction, denoising, and anomaly detection. Variants such as Variational Autoencoders (VAEs) introduce probabilistic components to generate new data samples.

Generative Adversarial Networks (GANs): GANs consist of two competing neural networks: a generator, which attempts to produce realistic data, and a discriminator, which distinguishes between real and generated samples. Through adversarial training, GANs have achieved remarkable success in image synthesis, data augmentation, and other generative tasks [15].

Transformer Architectures: Transformers leverage self-attention mechanisms to capture contextual relationships without recurrent connections. Initially proposed for machine translation, transformers have become state-of-the-art in natural language processing and are increasingly applied to vision tasks (Vision Transformers, ViTs). They provide parallel processing advantages and scalability for large datasets [16].

In practice, these architectures are often combined or modified to meet application-specific requirements. Hybrid approaches—such as combining convolutional and recurrent layers—are widely adopted in domains like video analysis, biomedical signal processing, and autonomous systems.

Conclusion

Deep learning neural networks encompass a range of architectures, each developed to solve specific computational problems and application needs. Convolutional Neural Networks (CNNs) have proven highly effective for image recognition and computer vision tasks by automatically extracting spatial hierarchies of features. Generative Adversarial Networks (GANs) have enabled unsupervised learning approaches that can generate realistic synthetic data, while Transformer-based models have achieved significant success in tasks requiring the processing of complex data relationships, particularly in natural language processing.

Selecting the appropriate architecture depends on the nature of the task, the computational resources available, and the characteristics of the dataset. While these models have achieved impressive performance across many domains, they still face challenges such as high data requirements, computational complexity, and a lack of interpretability. Understanding these core architectures provides a strong foundation for applying deep learning methods effectively and paves the way for exploring their practical implementation in subsequent chapters.

Chapter 3

Data Classification Using CNN

3.1 Introduction

Convolutional Neural Networks (CNNs) represent one of the most powerful and widely adopted deep learning architectures for data classification tasks. Unlike traditional methods that rely on handcrafted features, CNNs automatically learn spatial hierarchies of features directly from raw data through layers of convolution, activation, and pooling. Originally inspired by the organization of the visual cortex, CNNs have achieved state-of-the-art performance in image recognition, medical image analysis, speech processing, and other domains requiring high-dimensional pattern recognition. Their ability to extract both low-level details and high-level semantic features makes them exceptionally robust and scalable. In classification problems, CNNs transform input data into a structured representation that is then processed by fully connected layers to produce accurate predictions. This chapter introduces the foundations of data classification using CNNs, examines their architecture, highlights their biological inspiration, and discusses the performance metrics used to evaluate their effectiveness in real-world applications.

3.1.1 Definition

A Convolutional Neural Network (CNN) is a specialized class of deep neural networks designed to automatically and adaptively learn spatial hierarchies of features from input data. Unlike fully connected networks, CNNs employ convolutional layers that use learnable filters (kernels) to scan input data, detecting local patterns such as edges, textures, and shapes. These features are then progressively combined through additional convolution, non-linear activation functions, and pooling operations to form high-level abstractions. The resulting feature maps are typically passed through fully connected layers to perform classification or regression tasks. CNNs are particularly effective for structured data such as images, audio, or time series, where spatial or temporal correlations are important [13].

3.1.2 Biological Motivation for CNNs

Convolutional Neural Networks (CNNs) are inspired by the organization of the visual cortex in the human brain. In biological vision systems, individual neurons in the visual cortex respond to specific regions of the visual field, forming a pattern of local connectivity. This concept, introduced by Hubel and Wiesel in their seminal work on cat and monkey visual cortices, demonstrated that certain neurons, called simple cells, respond selectively to edges and orientations within a localized receptive field, while complex cells combine these responses to detect more intricate patterns. This biologically inspired design allows CNNs to efficiently extract spatial features from images while reducing the number of trainable parameters compared to fully connected networks [17].

3.2 General Architecture of Convolutional Neural Networks

The architecture of a Convolutional Neural Network (CNN) is designed to automatically and adaptively learn hierarchical features from input data [13]. Originally developed for image analysis, CNNs have also proven highly effective for processing other types of structured data, such as audio signals. The fundamental concept remains the same: CNNs extract local patterns and progressively combine them to form higher-level representations, enabling accurate classification and prediction tasks.

For image data, CNNs exploit the spatial structure of images. The process begins with an **input layer**, which receives raw pixel values organized in a grid format. This is followed by one or more **convolutional layers**, where filters (kernels) slide across the image to capture local spatial features such as edges, corners, or textures. Each convolutional operation is typically followed by an **activation function**, such as ReLU, introducing non-linearity to capture complex relationships. After convolution, **pooling layers** (commonly max-pooling) reduce the spatial dimensions of feature maps, making the network more computationally efficient and robust to small variations in the input.

For audio data, CNNs are usually applied to a time-frequency representation of the signal, such as a spectrogram. This transforms audio from the time domain into a two-dimensional grid, where the horizontal axis represents time, the vertical axis represents frequency, and the intensity of each cell reflects energy. Applying convolution on spectrograms enables CNNs to learn both temporal and spectral patterns, making them highly effective for speech recognition and sound classification. Similar to image CNNs, convolutional layers extract low-level features (e.g., frequency components) and progressively combine them into higher-level representations (e.g., phonemes or sound events). Pooling layers reduce dimensionality, while activation functions add non-linearity.

After feature extraction, CNNs use one or more **fully connected layers** to integrate the learned features. The final **output layer** typically applies a softmax function to generate class probabilities for multi-class classification. To enhance generalization and prevent overfitting, regularization methods such as dropout or weight decay are often employed.

This hierarchical design—from low-level feature detection to high-level pattern understanding—mirrors biological vision and auditory systems, making CNNs highly effective for image recognition, object detection, and audio analysis [13, 18].

3.2.1 Feature Extraction Module

The feature extraction module of a CNN includes convolutional layers, activation functions, and pooling layers [13]. These layers progressively learn hierarchical feature maps: early layers capture simple patterns (e.g., edges/textures in images or basic waveform morphologies in signals), while deeper layers capture complex structures (e.g., anatomical object parts or arrhythmic sequences). Convolution operations with learnable kernels remove the need for manual feature engineering for both spatial and temporal data, and pooling reduces spatial or temporal dimensions to make the model efficient and robust.

Convolutional Layers

Convolutional layers are the core components of CNNs. They apply learnable filters (or kernels) that slide across local regions of the input to detect spatial or temporal patterns as show in figure 3.1. In image processing, early convolutional layers capture low-level features such as edges or corners, while deeper layers extract more abstract features like shapes or objects. For audio signals (e.g., spectrograms), these layers detect frequency components and temporal structures. The convolution operation significantly reduces the need for manual feature engineering, allowing the network to learn representations directly from data [13].

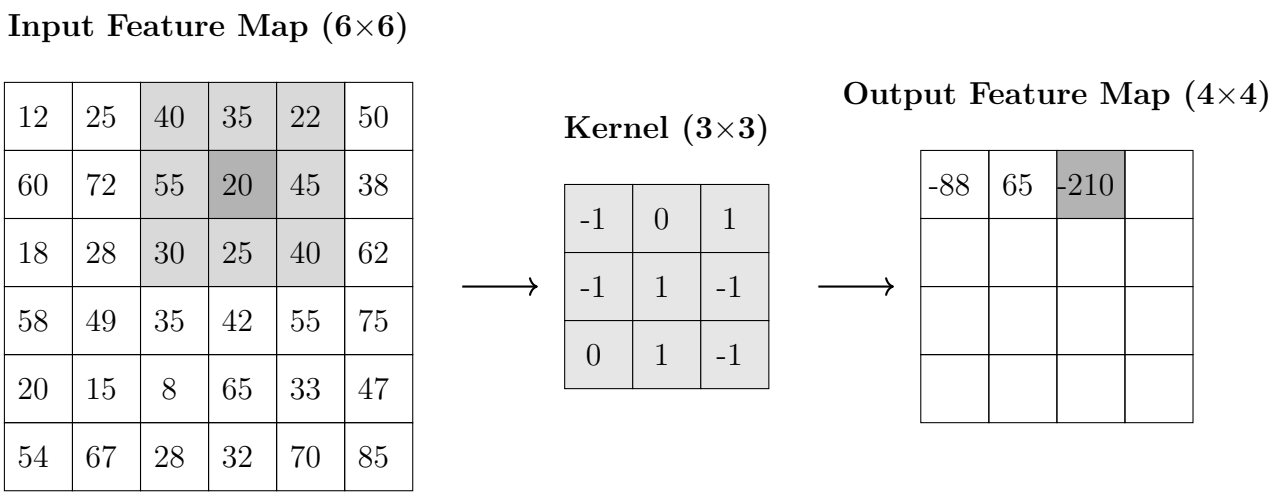

Figure 3.1: Convolution operation using a 3×3 kernel over a 6×6 input feature map

Activation Functions

The activation function Figure 3.2 introduces non-linearity to the model, enabling it to learn complex patterns . Typically, the Rectified Linear Unit (ReLU) is used, but other activation functions like sigmoid or tanh can be employed depending on the specific use case [25].

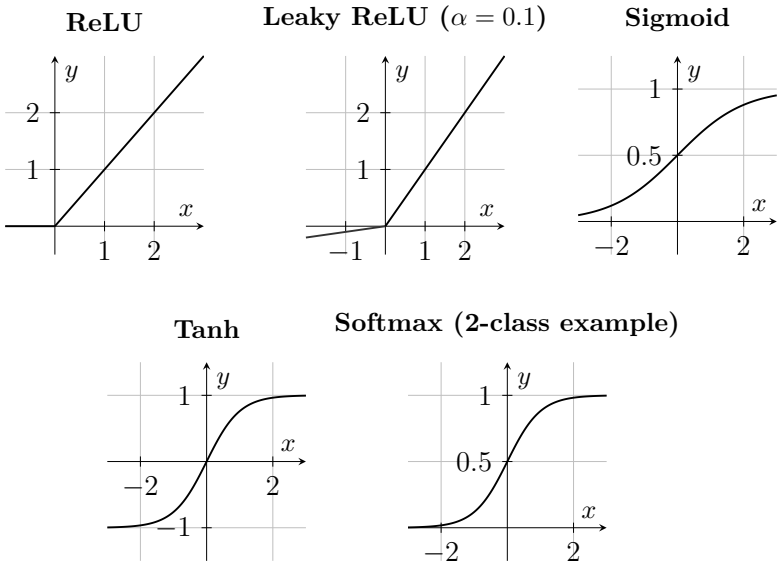

Figure 3.2: Common activation functions used in Convolutional Neural Networks (CNNs)

Pooling Layers

Pooling layers reduce the spatial or temporal dimensions of the feature maps while preserving the most important information. Max-pooling, the most common approach, selects the maximum value within a small window, making the network more robust to small input variations and reducing computational complexity. Average pooling, which computes the average value within the window, is sometimes used as an alternative. Pooling layers help control overfitting by introducing a form of translation invariance [19].

Together, these three components form the hierarchical feature extractor of CNNs, enabling the automatic discovery of patterns in both images and audio signals as show in Figure 3.3.

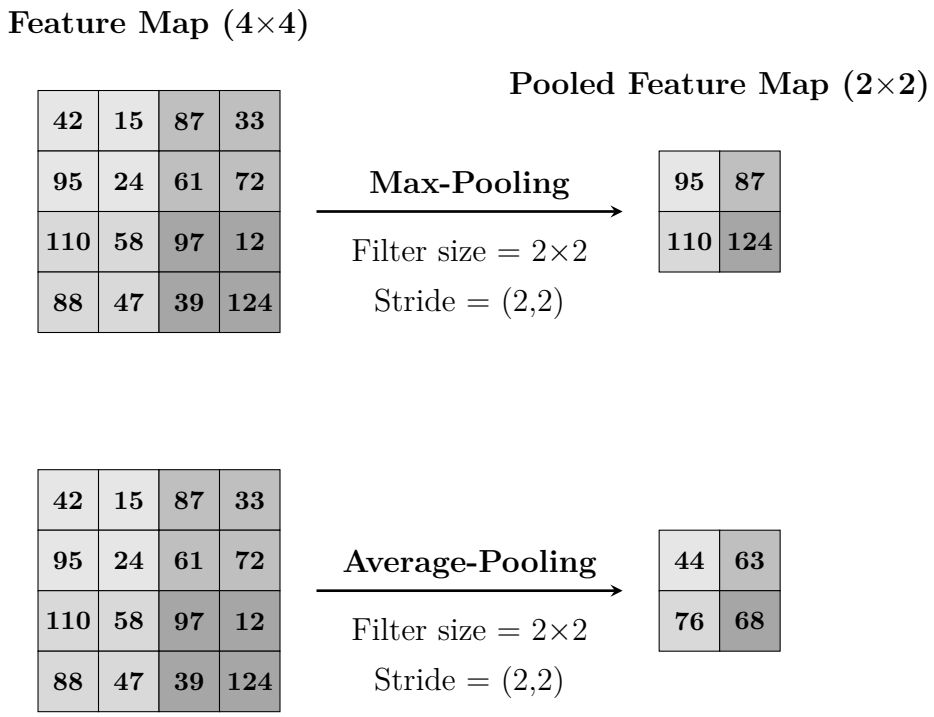

Figure 3.3: Pooling operations in CNNs: Max-Pooling (top) and Average-Pooling (bottom).

3.2.2 Classification Module

After the convolutional and pooling layers have progressively extracted hierarchical features, the CNN transitions to the **classification module**. This part of the architecture is responsible for mapping the high-level feature representations into output predictions [13].

Flattening Layer

The first step in the classification module is the **flattening operation**, which converts the multidimensional feature maps into a single one-dimensional vector as show in figure 3.4. This transformation makes the data compatible with fully connected layers.

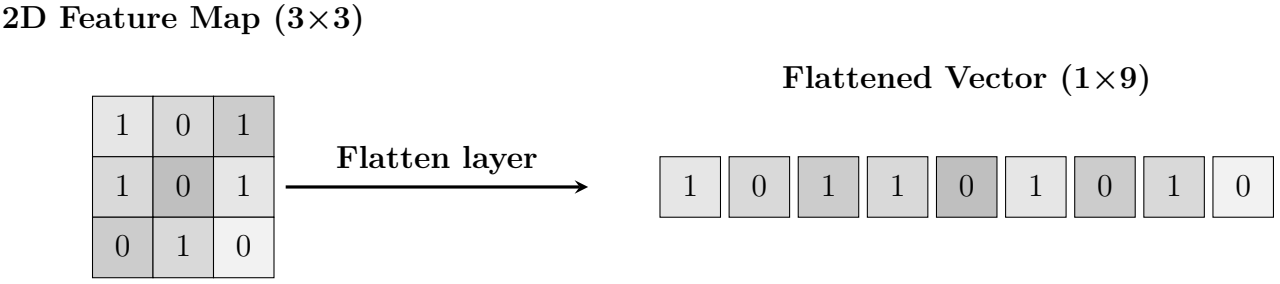

Figure 3.4: Flatten layer reshaping a 2D feature map into a 1D vector.

Fully Connected Layers (Dense Layers)

Once flattened, the vector passes through one or more **fully connected (dense) layers** as show in Figure 3.5. Each neuron in these layers is connected to every neuron in the previous layer, allowing the network to combine all extracted features to make decisions [13, 25]. These layers perform a non-linear mapping and are typically followed by activation functions such as ReLU or Leaky ReLU [25, 23].

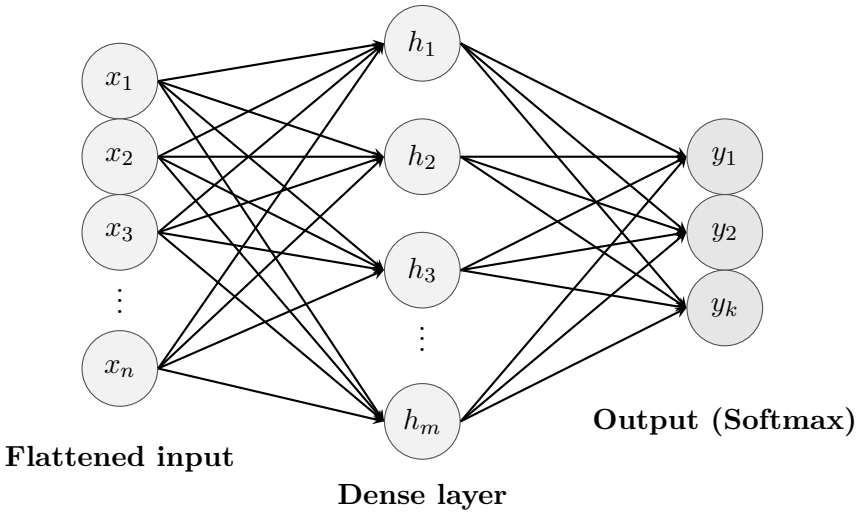

Figure 3.5: Fully Connected (Dense) Layers after flattening in a CNN. Each input is connected to every neuron in the dense layer, which then maps features to output class probabilities.

Output Layer

The final dense layer is usually followed by a **softmax activation function** as show in Figure 3.6 (for multi-class classification) or a **sigmoid activation function** (for binary classification) [25, 23]. These functions transform raw output scores into normalized class probabilities:

$$P(y = i \mid \mathbf{x}) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

where C is the total number of classes and z_i is the output logit corresponding to class i.

The class with the highest probability is selected as the network's prediction [13].

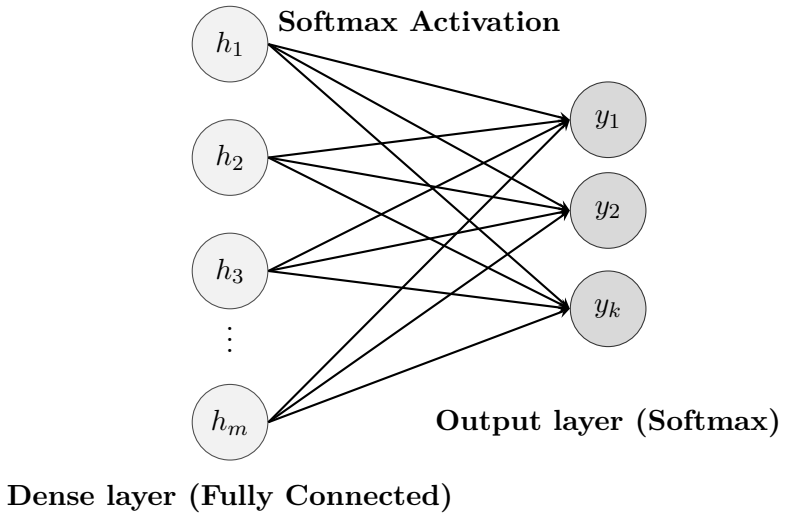

Figure 3.6: Fully connected dense layer directly mapped to output probabilities through a Softmax layer.

Regularization Techniques

To prevent overfitting, **dropout layers** or **weight decay (L2 regularization)** can be inserted between dense layers [25, 23]. Dropout randomly deactivates a fraction of neurons during training, forcing the network to learn redundant representations and improving its ability to generalize [13].

Overall, the classification module integrates all features extracted by the convolutional backbone and produces the final decision about the input class.

3.3 Evaluating CNN Classification Performance

Evaluation metrics are critical to determine how well a Convolutional Neural Network (CNN) model generalizes to unseen data. Relying solely on classification accuracy can be misleading, particularly when working with imbalanced datasets, where some classes have many more samples than others. In such cases, a model may appear to perform well simply by predicting the majority class more frequently, even though it fails to correctly identify minority classes [22].

3.3.1 Limitations of Accuracy as a Sole Metric

In deep learning–based vision systems, accuracy alone often fails to provide a complete picture of model performance. When datasets are imbalanced, meaning that some classes contain far more samples than others, a model can achieve high accuracy simply by predicting the majority class more frequently, while performing poorly on minority classes [22].

For example, in medical image classification, if 95% of the samples are normal and only 5% show a disease, a classifier that always predicts "normal" will achieve 95% accuracy yet completely fail to detect the disease [22].

To address this limitation, additional metrics such as **precision**, **recall (sensitivity)**, and the **F1-score** are required. These metrics evaluate how well the model detects minority classes, balances false positives and false negatives, and provides a more reliable assessment of real-world performance in vision tasks [22].

3.3.2 Overall Classification Accuracy

Overall classification accuracy is defined as the ratio between the number of correctly classified samples and the total number of samples in the dataset:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of samples}}$$

This metric provides a quick, global indication of a CNN model's performance in image classification tasks. However, while accuracy summarizes how often predictions are correct, it does not indicate *which* classes are being correctly or incorrectly classified. In highly imbalanced vision datasets, a high accuracy can be misleading if the model predominantly predicts the majority class. Therefore, accuracy should always be complemented with more granular metrics such as precision, recall, and F1-score to ensure robust evaluation of CNN classifiers [22].

3.3.3 Recall (Sensitivity)

Recall, also known as sensitivity or true positive rate, measures the proportion of actual positive instances that are correctly identified by a model. It is defined as:

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \tag{3.1}$$

A high recall indicates that the model successfully captures most of the positive cases. This metric is especially important in scenarios where missing positive cases can have serious consequences, such as in medical diagnoses, where failing to detect a disease may lead to delayed or inadequate treatment.

Recall is often used alongside precision to provide a more comprehensive evaluation of a model's performance, particularly in cases of imbalanced datasets common in classification tasks [22].

3.3.4 Precision

Precision, also known as positive predictive value, measures the proportion of instances predicted as positive that are actually positive. It is defined as:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{3.2}$$

A high precision indicates that the model makes few false positive predictions, ensuring that positive predictions are reliable. This metric is particularly important in applications where false alarms can lead to unnecessary actions, costs, or anxiety, such as in medical testing or automated detection systems.

Precision is often considered together with recall to provide a balanced evaluation of model performance, especially when dealing with imbalanced datasets [22].

3.3.5 F-measure (F1-score)

The F1-score, also known as the F-measure, is the harmonic mean of precision and recall. It provides a single metric that balances both false positives and false negatives, making it particularly useful when an equal importance is placed on precision and recall. The F1-score is defined as:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

A high F1-score indicates that the model achieves both high precision and high recall, which is especially important in classification tasks where both types of errors can have significant consequences, such as in medical diagnosis or fault detection in automated systems [22].

3.4 Conclusion

Convolutional Neural Networks (CNNs) provide a powerful framework for automatically learning both spatial and temporal features from data. Their hierarchical architecture enables the extraction of increasingly complex representations, making them highly effective for a wide range of applications, including image and audio classification.

Proper evaluation metrics, such as accuracy, precision, recall, and F1-score, are essential for assessing model performance comprehensively and ensuring reliable results. Understanding these concepts lays a strong foundation for the practical implementation of deep learning techniques, which will be explored in detail in the subsequent chapters.

Chapter 4

Implementation

4.1 Introduction

This chapter presents the practical implementation of the proposed deep learning approach for medical image and signal classification. After discussing the theoretical foundations of Convolutional Neural Networks (CNNs) in previous chapters, this section focuses on how these concepts are applied to real datasets, including ECG and brain tumor scans. The goal is to transform the theoretical design into a functional system capable of learning discriminative features and performing accurate classification.

The implementation process begins by describing the computational environment, including the hardware setup, programming tools, and deep learning frameworks that enable efficient model development. The datasets used in this study are then analyzed to provide insights into their structure, class distribution, and preprocessing requirements.

Following this, CNN architectures are developed and trained for each classification task. The training phase involves hyperparameter tuning, optimization strategies, and techniques to reduce overfitting, such as data augmentation and dropout regularization. The chapter also emphasizes the visualization of results, including performance metrics and learning curves, to evaluate the robustness of the models.

Finally, the obtained results are analyzed in detail, highlighting the strengths and limitations of the proposed models. This systematic implementation not only validates the theoretical framework introduced earlier but also demonstrates its applicability to real-world medical imaging problems.

4.2 Tools and Implementation Environment

4.2.1 Hardware Setup

The implementation and experimentation were carried out on a local personal computer [25]. Table 4.2.1 summarizes the system specifications used for developing and testing the Convolu-

tional Neural Network (CNN) models.

Component	**Specification**
Processor (CPU)	Intel Core i3, 11^{th} generation
RAM	8 GB
Graphics	Integrated GPU (Intel UHD)
Storage	512 GB SSD
Operating System	Windows 10 (64-bit)

Table 4.1: Hardware configuration used for implementation

Although this configuration is modest compared to dedicated deep learning workstations with high-end GPUs (e.g., NVIDIA RTX or Tesla) [13], it was sufficient to develop, train, and evaluate CNN models on the target datasets. Resource constraints were addressed by reducing batch sizes, optimizing training epochs, and applying efficient data preprocessing techniques. This demonstrates that deep learning models for research and educational purposes can be implemented successfully on standard computing hardware without requiring costly specialized equipment.

4.2.2 Python Programming Language

Python was chosen as the primary programming language for implementing the Convolutional Neural Network (CNN) models due to its simplicity, extensive community support, and comprehensive ecosystem for machine learning and deep learning applications [25, 13]. Its clean syntax and readability accelerate the development process, making it suitable for research and educational purposes.

Feature	**Benefit**
Ease of Use	Simple syntax reduces development time and facilitates debugging.
Community Support	A large and active community provides extensive tutorials, documentation, and open-source projects.
Rich Libraries	Availability of powerful frameworks such as TensorFlow, Keras, and PyTorch for building deep learning models.
Cross-Platform	Runs seamlessly on Windows, Linux, and macOS systems.
Integration	Easily integrates with tools for visualization, data analysis, and deployment.

Table 4.2: Advantages of Python for Deep Learning Implementation

In this work, Python served as the foundation for data preprocessing, model development, training, and evaluation, ensuring both flexibility and scalability.

4.2.3 Libraries and Frameworks

Several Python libraries and frameworks were used to facilitate the implementation of deep learning models. These tools streamline dataset processing, model development, and performance evaluation [25, 13]. Below is a description of the main tools adopted in this work.

TensorFlow

TensorFlow is a powerful open-source library developed by Google for building and deploying deep learning models [25]. It provides both low-level operations for fine-grained control and high-level APIs to simplify model training. TensorFlow supports GPU acceleration and efficient computation, which makes it ideal for deep learning applications.

Keras

Keras is a high-level deep learning API built on top of TensorFlow [25]. It allows rapid model prototyping using simple, modular building blocks. With Keras, complex neural network architectures such as Convolutional Neural Networks (CNNs) can be constructed with minimal code, while maintaining flexibility for customization.

NumPy

NumPy is a fundamental Python library for numerical computations [25]. It provides support for multidimensional arrays and efficient matrix operations, which are essential for handling input data and performing mathematical calculations required during neural network training.

Matplotlib

Matplotlib is a comprehensive visualization library used to create plots and figures [25]. In this work, it was employed to visualize training curves, model accuracy, and loss evolution, as well as to present prediction results in a clear and interpretable way.

scikit-learn

scikit-learn is a widely used library for machine learning tasks [22]. It was used in this work to split datasets into training, validation, and testing sets, as well as to compute performance metrics such as accuracy, precision, recall, and F1-score for evaluating the models.

Pandas

Pandas is a powerful library for handling structured datasets [25]. It provides flexible data manipulation tools, making it easier to analyze and preprocess datasets before feeding them into the deep learning models.

4.3 Brain Tumor Classification

4.3.1 Data Description and Analysis

The dataset used in this work is the **Brain Tumor MRI Dataset**, publicly available on Kaggle[1], which is commonly employed for evaluating deep learning models in medical image classification [13, 25]. It contains more than **7,000 T1-weighted brain MRI images** divided into four categories: **No Tumor**, **Glioma Tumor**, **Meningioma Tumor**, and **Pituitary Tumor**. The images are organized into separate folders for training, validation, and testing, with subdirectories corresponding to each class to ensure a clear separation between model development and evaluation [13].

Although all images are stored in three-channel JPEG format for compatibility with convolutional neural network architectures such as VGG16, the original scans are essentially grayscale. Image dimensions are not uniform, ranging approximately from 150×150 to 512×512 pixels, and the intensity distribution varies due to differences in MRI acquisition settings. The dataset is slightly imbalanced, with around 3,000 images of glioma tumors, 2,500 of meningioma tumors, 1,500 of pituitary tumors, and about 1,000 normal brain scans without tumors. Before training, an exploratory analysis was conducted to verify data quality, visualize representative samples from each class, and study pixel intensity histograms, which revealed class-dependent contrast variations that may aid automated feature extraction [25].

A small number of corrupted or mislabeled images were removed, and data augmentation techniques were later applied to mitigate class imbalance and improve the robustness of the models. This thorough understanding of the dataset provides a reliable foundation for the deep learning experiments presented in the next sections [13, 25].

No Tumor

This class contains brain MRI scans showing no evidence of tumor growth. The brain structures appear normal with no abnormal mass or irregularity, serving as a baseline for the classification model.

[1] https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset

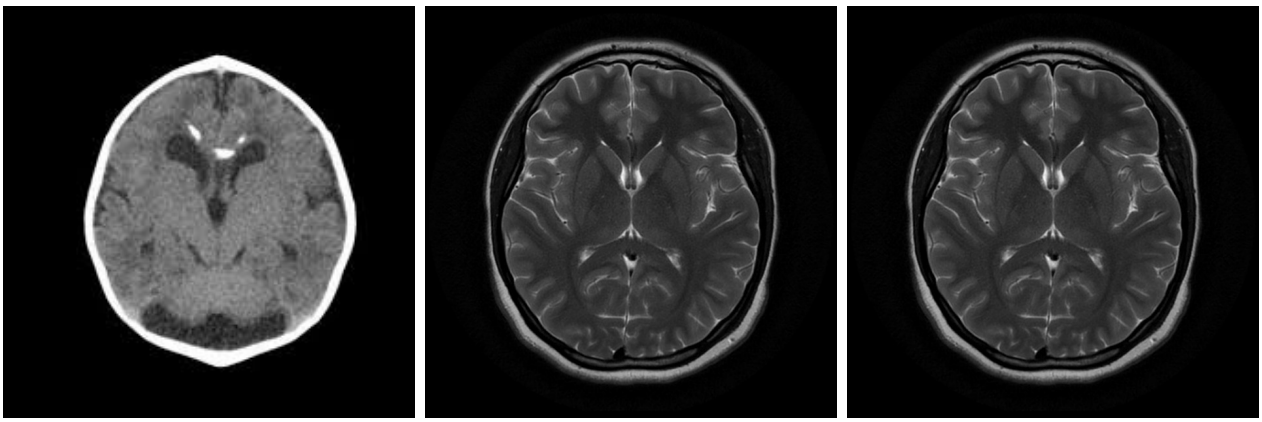

Figure 4.1: Sample MRI images from the **No Tumor** class.

Glioma Tumor

Gliomas are tumors that originate from glial cells in the brain. They typically appear as irregular masses with blurred boundaries and varying intensity on MRI scans.

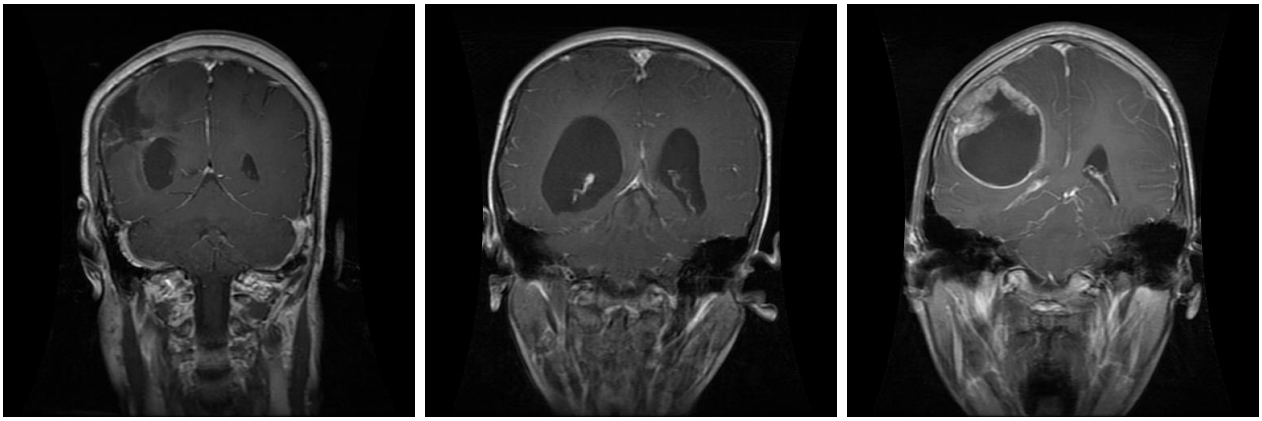

Figure 4.2: Sample MRI images from the **Glioma Tumor** class.

Meningioma Tumor

Meningiomas are tumors that develop in the meninges, the protective membranes surrounding the brain and spinal cord. They usually have well-defined, uniform structures on MRI scans.

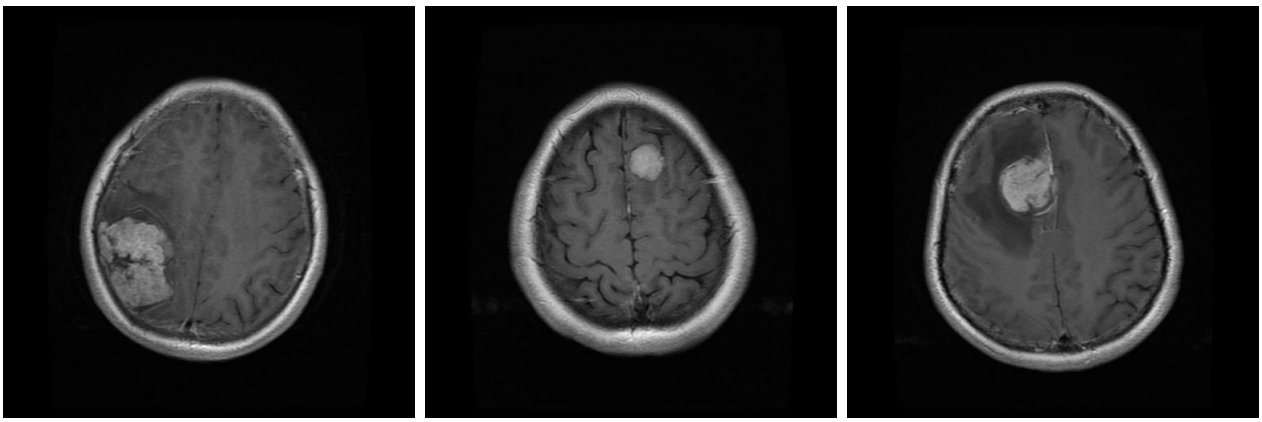

Figure 4.3: Sample MRI images from the **Meningioma Tumor** class.

Pituitary Tumor

Pituitary tumors arise from the pituitary gland located at the base of the brain. They often appear as small, dense masses in the sellar region on MRI images.

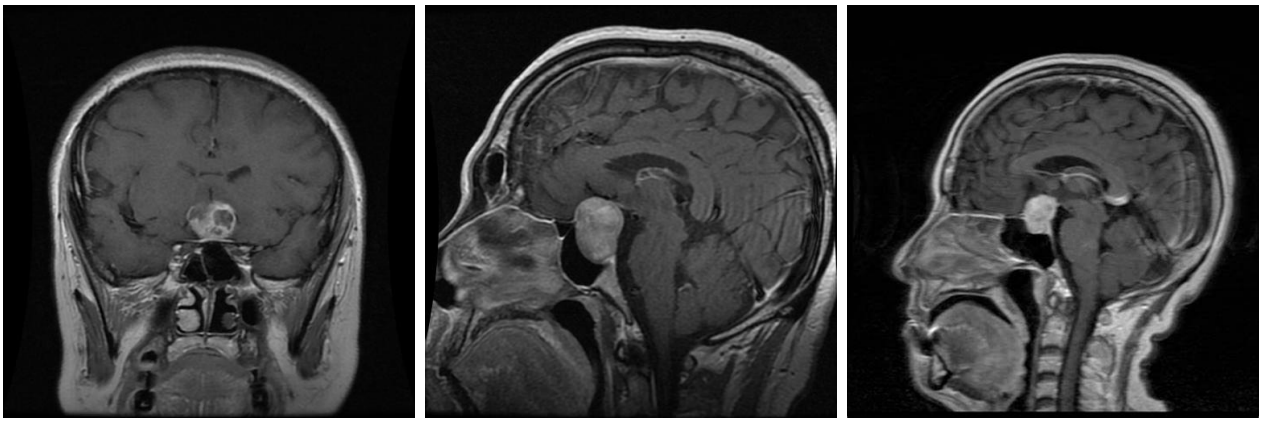

Figure 4.4: Sample MRI images from the **Pituitary Tumor** class.

4.3.2 Model Development and Training

Data Processing

The MRI brain tumor dataset was prepared using a custom preprocessing and augmentation pipeline implemented in Python [25]. All images were resized to 128×128 pixels to ensure consistent input dimensions for the CNN [13]. Pixel values were normalized to the range $[0, 1]$ by dividing by 255 [25].

Custom Image Augmentation

Rather than relying on standard augmentation tools, a manual function was implemented using the `Pillow` library to enhance data variability [25]:

- **Brightness adjustment:** Random factor between 0.8 and 1.2,

- **Contrast adjustment:** Random factor between 0.8 and 1.2.

These transformations help the model learn features robust to illumination and contrast changes in MRI scans [13].

Label Encoding

Class labels were encoded into integers by mapping each class folder name to an index [25]. This ensured that the network output layer (Softmax) could directly predict one of the four encoded classes: *No Tumor*, *Glioma*, *Meningioma*, or *Pituitary*.

Custom Data Generator

A batch generator was implemented to load and augment data efficiently during training [25]:

- Loads images in mini-batches (batch size = 12),
- Applies random augmentation on-the-fly,
- Encodes labels dynamically,
- Yields image–label pairs to the training loop for the specified number of epochs.

This approach avoids memory overflow and enables training on large datasets without precomputing augmented images [13].

CNN Architecture and Transfer Learning

The classification model is based on the **VGG16** architecture pre-trained on the ImageNet dataset [25, 13]. This approach leverages transfer learning by using VGG16 as a feature extractor while fine-tuning its deeper layers to adapt to brain tumor MRI classification [25].

Base Network

- **Input shape:** $128 \times 128 \times 3$ RGB images.
- **Pre-trained VGG16:** with top classification layers removed (`include_top=False`) [13].
- All convolutional layers are frozen, except for the last three, which are trainable to allow domain-specific feature adaptation [25].

Custom Classification Head

- **Flatten layer:** converts extracted feature maps into a one-dimensional vector [25].
- **Dense layer:** 128 neurons with ReLU activation for nonlinear feature mapping [13].

- **Dropout layers:** with rates 0.3 and 0.2 to reduce overfitting [25].
- **Softmax output layer:** 4 neurons corresponding to the classes (*No Tumor*, *Glioma*, *Meningioma*, *Pituitary*) [25].

Training Configuration

- Optimizer: **Adam** with a learning rate of 10^{-4}.
- Loss function: **Sparse categorical cross-entropy**.
- Evaluation metric: **Sparse categorical accuracy**.

Layer	Description
Input	$128 \times 128 \times 3$
VGG16 Base	Pre-trained on ImageNet (last 3 layers trainable)
Flatten	Converts feature maps to vector
Dense (128)	ReLU activation
Dropout	Rate = 0.3
Dense (Softmax)	4 output classes

Table 4.3: Summary of CNN Model Architecture

4.3.3 Visualisation of Results

Training and validation accuracy and loss were plotted throughout the training epochs to monitor the learning behavior of the VGG16-based model [25, 13]. These visualizations assist in identifying issues such as overfitting or underfitting during training [25].

In addition, a confusion matrix provides an intuitive overview of the classification performance across the four brain tumor classes: No Tumor, Glioma, Meningioma, and Pituitary [22]. This matrix highlights how well the model distinguishes between the different categories and where misclassifications occur [22].

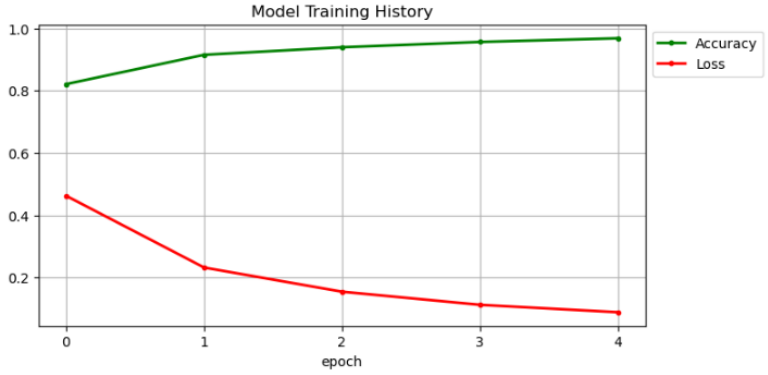

Figure 4.5: The progress of training and validation accuracy and loss

Classification Report

The classification report summarizes the precision, recall, and F1-score for each class as show in Table 4.4, along with the overall accuracy [22]. Precision measures the correctness of positive predictions, recall measures the model's ability to detect all samples of a class, and the F1-score balances both metrics [22].

Class	Precision	Recall	F1-Score	Support
No Tumor	1.00	0.69	0.81	300
Glioma	0.73	0.99	0.84	306
Meningioma	0.99	0.98	0.99	405
Pituitary	0.99	0.96	0.97	300
Accuracy		**0.91**		1311
Macro Avg	0.93	0.90	0.90	1311
Weighted Avg	0.93	0.91	0.91	1311

Table 4.4: Classification report of the CNN model on the test dataset.

Confusion Matrix

The confusion matrix (Figure 4.6) shows the distribution of predictions across the four classes. The diagonal elements represent correct predictions, and the off-diagonal elements indicate misclassifications.

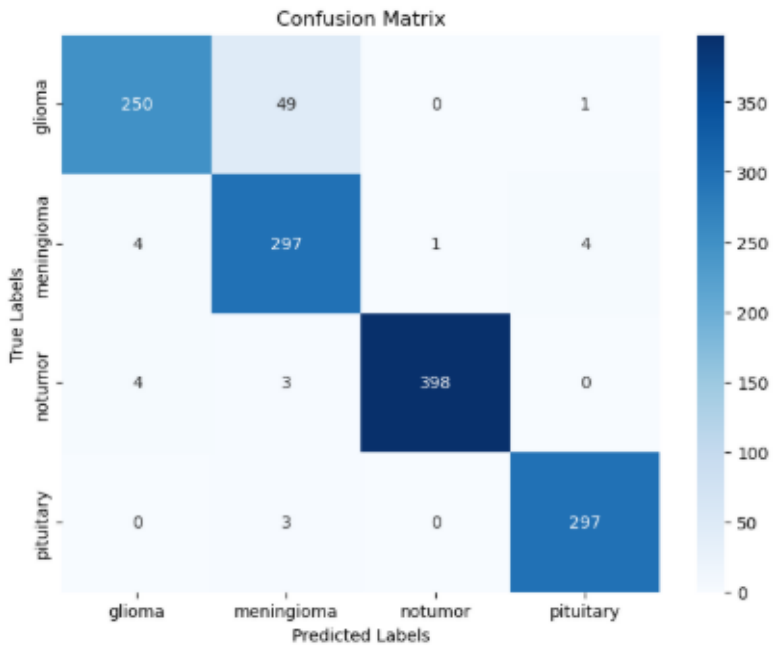

Figure 4.6: Confusion matrix for brain tumor classification model

Analysis of Results

- The model performed very well on Meningioma and Pituitary with recall 96% and 98%.

- No Tumor achieved perfect precision but relatively lower recall (69%), indicating missed detections.
- The high F1-score across all classes shows balanced performance with minimal bias.
- The overall accuracy of 91% confirms the robustness of the VGG16-based transfer learning approach.

4.4 ECG Classification

This section details the implementation and experimental results for the classification of electrocardiogram (ECG) heartbeats using the MIT-BIH Arrhythmia Database [21, 20]. The process encompasses data description, preprocessing, a balanced dataset creation, model architecture, training, and a comprehensive analysis of the classification performance.

4.4.1 Electrocardiogram Overview and Heartbeat Morphology

The electrocardiogram (ECG) is a non-invasive diagnostic tool that records the heart's electrical activity over time using surface electrodes [23]. Each heartbeat produces an electrical impulse that propagates through the atria and ventricles, triggering contraction and relaxation. These impulses are captured as a voltage-time trace, allowing clinicians and researchers to evaluate cardiac rhythm, conduction pathways, and detect abnormalities such as arrhythmias or ischemia. Standard clinical ECGs are recorded with 12 leads for a comprehensive spatial view of the heart's activity, while research databases such as MIT-BIH often use two-lead or single-lead recordings, which are sufficient for rhythm and beat classification tasks [20]. In the MIT-BIH Arrhythmia Database, signals are sampled at 360 Hz to preserve fine morphological details of each beat [21].

A single heartbeat is composed of several characteristic waves and intervals that correspond to physiological events [23]:

- **P wave:** Atrial depolarization initiating atrial contraction.
- **QRS complex:** Ventricular depolarization, the most prominent feature of the ECG waveform.
- **T wave:** Ventricular repolarization as the ventricles recover for the next beat.
- **PR interval:** Time from the onset of atrial depolarization to ventricular depolarization, indicating atrioventricular conduction.
- **ST segment:** Period between ventricular depolarization and repolarization, analyzed for ischemic changes.

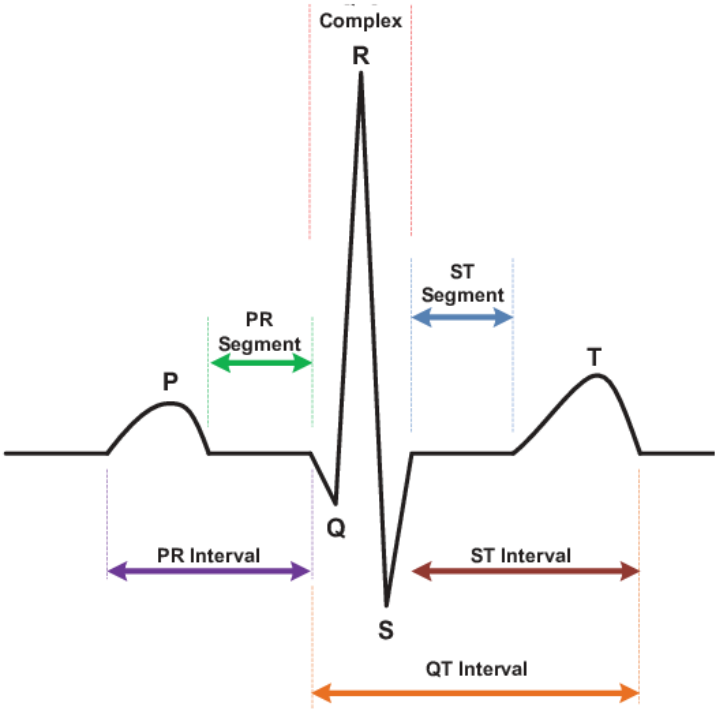

Figure 4.7: ECG heartbeat with waves and segments.

In this study, beats are annotated according to the AAMI standard heartbeat classes [21]. The dataset specifically includes:

- **N:** Normal beat (sinus rhythm)
- **L:** Left bundle branch block beat
- **R:** Right bundle branch block beat
- **V:** Premature ventricular contraction (PVC)
- **/ :** Paced beat (artificial pacemaker activation)

The shape, amplitude, and duration of these waveforms vary according to cardiac condition as show in figure 4.8 . For example, bundle branch blocks (**L** and **R**) are characterized by widened QRS complexes due to delayed ventricular conduction, while PVCs (**V**) appear as early, abnormally shaped QRS complexes without a preceding P wave. Paced beats (/) typically exhibit sharp pacing spikes followed by altered ventricular depolarization patterns. Automated ECG classification systems and deep learning models use these morphological and temporal features to discriminate between normal and abnormal beats [20, 21].

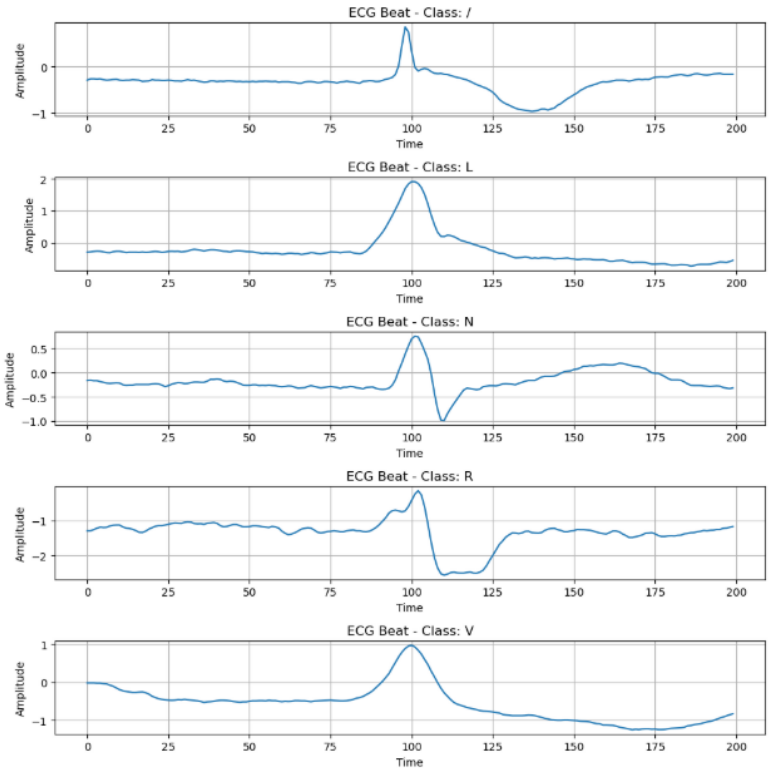

Figure 4.8: Sample waveforms for each of the five heartbeat classes used in this study [21, 20].

4.4.2 Data Description and Analysis

Data Source and Acquisition

The experiments in this study were conducted using the **MIT-BIH Arrhythmia Database** [20], freely available on PhysioNet[2], a web-based resource initiated by the NIH in 1999 that provides free access to physiologic signals (like ECGs) and related open-source software to support medical and biomedical research [20]. This database is one of the most widely used and authoritative benchmark datasets for the development and evaluation of arrhythmia detection algorithms.

The MIT-BIH database was originally created by the Massachusetts Institute of Technology to support research in automated ECG analysis and has since become a reference standard in both academia and industry [20]. It consists of 48 half-hour two-lead ECG recordings acquired from 47 subjects, including both inpatients and outpatients, representing a diverse range of arrhythmias and normal sinus rhythms. The signals were sampled at 360 Hz with 11-bit resolution over a 10 mV range, ensuring high temporal and amplitude precision for accurate waveform analysis. All recordings were carefully reviewed and annotated beat-by-beat by certified cardiologists, providing reliable labels for each heartbeat corresponding to its underlying rhythm or arrhythmic event. In total, the database contains 112,572 labeled heartbeats distributed across 23 rhythm classes, including normal beats, ventricular ectopic beats, supraventricular ectopic beats, fusion beats, and other less common arrhythmias, offering a rich and hetero-

[2] https://physionet.org/content/mitdb/1.0.0/

geneous dataset for machine learning tasks. The original unfiltered signals preserve baseline wander, muscle noise, and powerline interference, making this dataset particularly valuable for testing algorithms under realistic clinical conditions [20]. Such detailed annotation and high-quality acquisition have established the MIT-BIH Arrhythmia Database as the gold standard for evaluating ECG classification models and comparing results across different studies.

Dataset Preprocessing and Balancing

The raw ECG signals were processed to extract individual heartbeats. Each heartbeat was segmented to a fixed window of 200 samples (approximately 0.56 seconds) centered around the R-peak, resulting in an input shape of $(200, 1)$ [20].

The original class distribution was highly imbalanced, dominated by the normal beat class (‘N‘) as show in Figure 4.9 . The initial counts were:

```
Counter({'N': 75028, 'L': 8073, 'R': 7257, 'V': 7129, '/': 7026, 'A': 2546,
                                        ...})
```

To mitigate the bias introduced by class imbalance, a balanced subset was created. The five most populous classes were selected: ‘N‘, ‘L‘, ‘R‘, ‘V‘, ‘/‘. Each class was then down-sampled to 7,026 samples as show in Figure 4.10 , matching the size of the smallest class (‘/‘), resulting in a final balanced dataset of 35,130 samples.

The dataset was subsequently split into training and testing sets with an 80/20 ratio, ensuring an equal distribution of all classes in both sets:

- **Train set:** 28,104 samples
- **Test set:** 7,026 samples

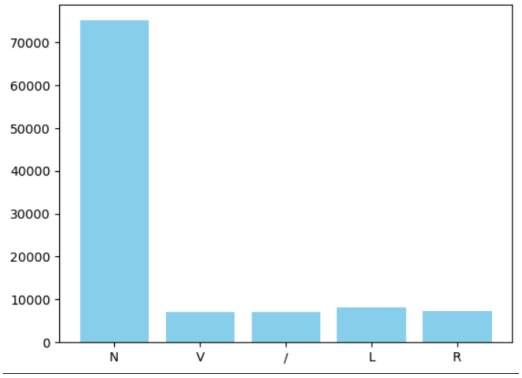

Figure 4.9: Imbalanced class distribution of the MIT-BIH Arrhythmia Database [20].

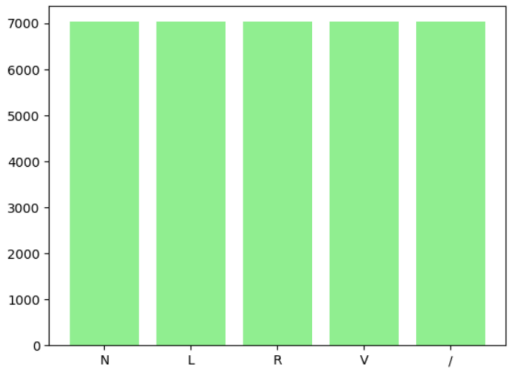

Figure 4.10: Balanced class distribution of the MIT-BIH Arrhythmia Database [20].

4.4.3 Model Development and Training

Proposed Model Architecture

A sequential Convolutional Neural Network (CNN) model was designed for this 1D signal classification task, leveraging principles of deep learning for biomedical signal analysis [23, 22]. The architecture, summarized in Table 4.5, is as follows:

- A first **1D convolutional layer** with 32 filters, a kernel size of 5, and ReLU activation, which extracts local temporal features from the ECG signal [23].
- A **max-pooling layer** with a pool size of 2 to reduce dimensionality and computation while preserving important features [23].
- A second **1D convolutional layer** with 64 filters, a kernel size of 5, and ReLU activation.
- Another **max-pooling layer** with a pool size of 2.
- A **flatten layer** to convert the feature maps into a 1D vector, preparing it for fully connected layers [23].
- A **dense (fully connected) layer** with 64 units and ReLU activation, allowing nonlinear combinations of extracted features.
- A **dropout layer** with a rate of 0.5 to prevent overfitting by randomly deactivating neurons during training [23].
- A final **dense output layer** with 5 units and a softmax activation function to output class probabilities [22].

Layer (Type)	Output Shape	Param #	Kernel Size
InputLayer	(None, 200, 1)	0	-
Conv1D	(None, 196, 32)	192	5
MaxPooling1D	(None, 98, 32)	0	2
Conv1D	(None, 94, 64)	10,304	5
MaxPooling1D	(None, 47, 64)	0	2
Flatten	(None, 3008)	0	-
Dense	(None, 64)	192,576	-
Dropout (0.5)	(None, 64)	0	-
Dense (Output)	(None, 5)	325	-
Total params:		203,397	
Trainable params:		203,397	

Table 4.5: Summary of the proposed 1D-CNN architecture for ECG classification.

Training Configuration

The model was compiled using the Adam optimizer with a default learning rate of 0.001 [22]. The loss function was set to *categorical crossentropy*, appropriate for multi-class classification [23]. The primary metric for evaluation was *accuracy*. The model was trained for 20 epochs with a batch size of 32. A validation split of 20% from the training set was used to monitor for overfitting during training [22].

4.4.4 Visualization of Results

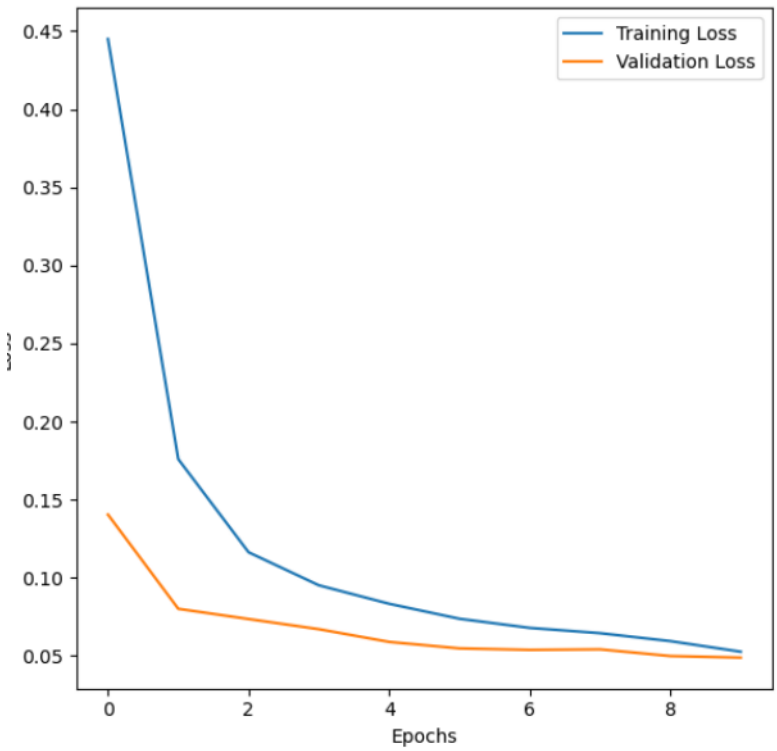

Figure 4.11: Training and validation loss over epochs.

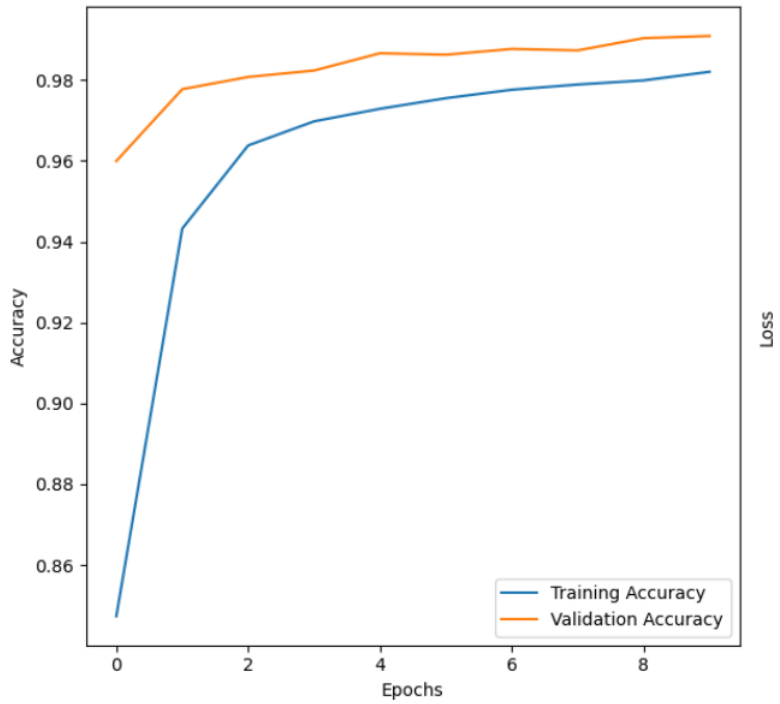

Figure 4.12: Training and validation accuracy over epochs.

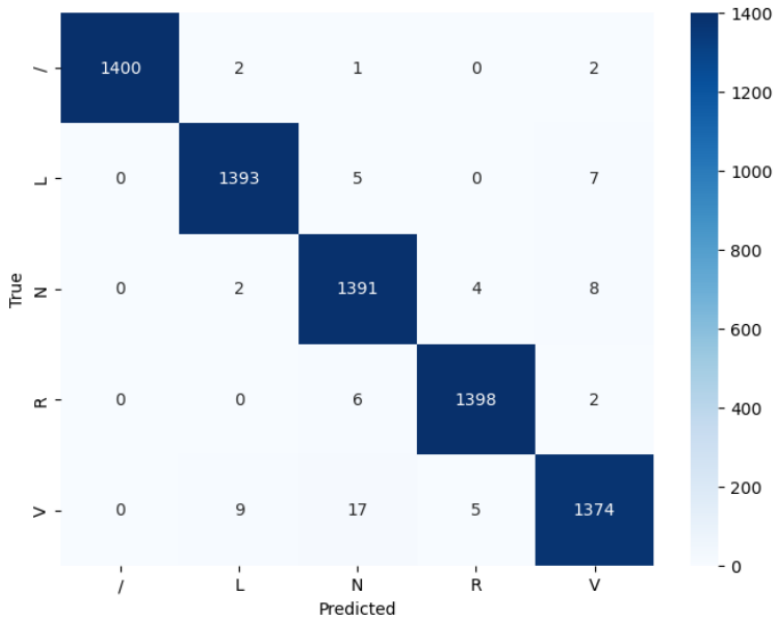

Figure 4.13: Normalized confusion matrix for the model's predictions on the test set. The diagonal elements represent the correct predictions.

4.4.5 Model Performance and Result Analysis

The trained model was evaluated on the held-out test set of 7,026 samples. The overall test accuracy achieved was **99.0%**, demonstrating the model's high efficacy in classifying ECG heartbeats.

A detailed classification report, providing precision, recall, and F1-score for each class, is

presented in Table 4.6. The macro-averaged F1-score of 0.99 confirms that the model performs consistently well across all five classes, with no single class being disproportionately misclassified. The high precision values (0.98–1.00) indicate that when the model predicts a class, it is highly likely to be correct. The high recall values (0.98–1.00) show that the model successfully finds nearly all samples of a given class.

The confusion matrix in Figure 4.13 provides further insight. The strong diagonal dominance confirms the high accuracy. The minor misclassifications, such as a few 'N' beats being predicted as 'R', are expected as these classes can sometimes share similar morphological characteristics in the ECG signal that are challenging to distinguish.

These results strongly validate the chosen 1D-CNN architecture and the data preprocessing pipeline for the task of automated ECG arrhythmia classification.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
/	1.00	1.00	1.00	1405
L	0.99	0.99	0.99	1405
N	0.98	0.99	0.98	1405
R	0.99	0.99	0.99	1406
V	0.99	0.98	0.98	1405
Accuracy				0.99
Macro Avg				0.99
Weighted Avg				0.99

Table 4.6: Detailed classification report on the test set (7,026 samples).

4.5 General Conclusion

Through the development of this thesis, substantial knowledge and hands-on experience were gained in applying deep learning techniques to diverse biomedical data challenges. By building and evaluating classification systems for both ECG arrhythmias and brain MRI tumors, we explored the complete lifecycle of a deep learning workflow. This included data acquisition, preprocessing, addressing class imbalance, model design and training, evaluation, and a detailed interpretation of the results.

The implemented models proved to be highly effective in their respective tasks. The 1D-CNN architecture successfully distinguished between five distinct types of cardiac arrhythmias (N, L, R, V, /) in ECG signals with exceptional performance. Concurrently, the CNN model for brain MRIs accurately identified and classified four different tumor categories, demonstrating strong overall capabilities. Visual tools such as training/validation curves and confusion matrices provided invaluable insight into each model's learning dynamics and classification behavior, clearly highlighting their strengths and precisely identifying specific areas for potential improvement.

This work reinforced core theoretical concepts and underscored critical practical considerations for biomedical AI. The importance of rigorous data preprocessing and the decisive impact of mitigating class imbalance were clearly demonstrated in the ECG project. The brain tumor task highlighted the necessity of moving beyond aggregate metrics to perform a nuanced, per-class analysis to truly understand model performance and trustworthiness. Key challenges, including overfitting, were mitigated through strategies like dropout regularization.

Looking ahead, the path for future work is clear. For the ECG model, the logical next step involves integration into a prototype diagnostic application to demonstrate real-world clinical utility. For the brain tumor classifier, future efforts should focus on improving performance for the confused classes by exploring more sophisticated architectures (e.g., Attention mechanisms), advanced data augmentation, or acquiring more diverse data. Ultimately, this thesis provides a robust foundation and a clear demonstration of the potential for deep learning to serve as a powerful tool in the future of medical diagnosis.

Bibliography

[1] T. M. Mitchell, "Machine Learning," *McGraw-Hill*, New York, 1997.

[2] J. Esteva, A. Robicquet, B. Ramsundar et al., "A guide to deep learning in healthcare," *Nature Medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[3] T. M. Mitchell, *Machine Learning*, McGraw-Hill, New York, 1997.

[4] M. Bracke, "Artificial intelligence in finance," *Bank of England Staff Working Paper No. 816*, 2019.

[5] S. Thrun, "Toward robotic cars," *Communications of the ACM*, vol. 53, no. 4, pp. 99–106, 2010.

[6] K. Grewal, M. Roggeveen, and J. Nordfält, "The future of retailing," *Journal of Retailing*, vol. 93, no. 2, pp. 135–140, 2017.

[7] L. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, "Machine learning in agriculture: A review," *Sensors*, vol. 18, no. 8, pp. 2674–2697, 2018.

[8] K. Lee, "Artificial intelligence and industrial applications," *Engineering*, vol. 6, no. 4, pp. 418–423, 2020.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, 2018.

[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. doi: 10.1038/nature14539.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, 2014.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

[17] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex," *Journal of Physiology*, vol. 195, no. 1, pp. 215–243, 1968.

[18] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[19] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," *International Conference on Artificial Neural Networks*, pp. 92–101, Springer, 2010.

[20] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH Arrhythmia Database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[21] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "MIT-BIH Arrhythmia Database," *PhysioNet*, 2000. Available: https://physionet.org/content/mitdb/1.0.0/ (Accessed: Sept. 2, 2025).

[22] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[23] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed., Upper Saddle River, NJ, USA: Pearson, 2009.

[24] Stuart Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, 3rd Edition, Pearson, 2016.

[25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

List of Figure References

[26] HealthImaging. *Example of artificial intelligence applied to brain tumor detection.* https://developer.nvidia.com/blog/automatically-segmenting-brain-tumors-with-ai/

[27] Areas Where AI is Implemented in Banking. *Example ai used in bank.* https://nuvento.com/blog/artificial-intelligence-in-banking-trends-in-usa/

[28] zoox. *Self-Driving Car Technology for robotaxi.* https://zoox.com/

[29] amazon. *AI to improve product recommendations and descriptions.* https://www.aboutamazon.com/news/retail/amazon-generative-ai-product-search-results

[30] *AI Based Solution.* https://www.aippals.com/ai-based-solutions.html

[31] "Supervised and Unsupervised Learning," LogicMojo, Accessed August 2025. Available: https://logicmojo.com/supervised-and-unsupervised-learning

[32] " ECG heart beat with waves and segments. Available: https://www.researchgate.net/figure/An-ECG-heart-beat-with-waves-and-segments_fig1_305875074