KLE Technological University | Creating Value, Leveraging Knowledge

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A Minor Project report on

FACIAL EMOTION RECOGNITION SYSTEM

Submitted

in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Ananya Kulkarni	01fe21bcs144
Swarna Patil	01fe21bcs145
Amrutha Beedikar	01fe21bcs198
Maanasi Shastri	01fe21bcs210

Under the guidance of

Dr. Sujatha C

School of Computer Science and Engineering

KLE Technological University, Hubballi

2023-2024

School of Computer Science and Engineering

KLE Technological University | Creating Value, Leveraging Knowledge

CERTIFICATE

This is to certify that the project entitled "FACIAL EMOTION RECOGNITION SYSTEM" is a bonafide work carried out by the student team Ananya Kulkarni(01FE21BCS144), Swarna Patil (01FE21BCS145), Amrutha Beedikar(01FE21BCS198), Maanasi Shastri(01FE21 BCS210), in partial fulfillment of the completion of the 6th semester, B. E. course during the year 2023 – 2024. The project report has been approved as it satisfies the academic requirement concerning the project work prescribed for the above-said course.

Guide Name
Dr.Sujatha C

SoCSE Head
Dr. Vijaylakshmi M.

External Viva-Voce

Name of the examiners Signature with date

1 _____ _____

2 _____ _____

ABSTRACT

The project focuses on the development and deployment of a Convolutional Neural Network (CNN) for facial emotion recognition using the MAX78000 microcontroller feather board. The CNN was trained on a dataset comprising 50,505 training images and 12,418 test images across seven emotion classes. Training involved extensive epochs and quantization-aware techniques, resulting in an accuracy of 56.52%. The MAX78000FTHR board, equipped with a dual-core Arm Cortex-M4 and a CNN accelerator, provided the computational power needed for efficient edge AI processing. Data balancing and increased training epochs helped improve performance, though challenges remained in accurately recognizing certain emotions. The development flow encompassed training, model quantization, synthesis into embedded C code, and deployment on the MAX78000 hardware, ensuring a systematic approach. This work highlights the potential of edge AI for real-time emotion recognition applications, addressing the computational constraints inherent in embedded systems while striving for improved accuracy and efficiency.

Keywords : *FER: Facial emotion recognition, HCI:Human-Computer Interaction, NN: Neural Networks, CNN: Convolutional neural network.*

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Dr.Sujatha C, our project guide, for their invaluable support and guidance throughout this project work. Their expertise, unwavering encouragement, and insightful feedback have played a pivotal role in shaping the direction of this study.

We are equally thankful to Mr.Subodh Ingleshwar, our co-guide, for their constructive input and mentorship. Their contributions have added depth and perspective to the research, enriching the overall quality of the work. We are indebted to both Dr.Sujatha C and Mr.Subodh Ingleshwar for their continuous encouragement, patience, and mentorship. Their commitment to excellence has been a constant source of inspiration, driving me to push the boundaries of my research.

We would also like to extend our gratitude to Dr.Vijayalakshmi M, HoS, School of Computer Science and Engineering, Prof.Prashant Narayankar, our project co-ordinator and to the KLE Technological University for providing the necessary resources and a conducive environment for the pursuit of knowledge.

This research would not have been possible without the support of my guides and the academic community. We are truly grateful for the opportunity to work under their guidance and are honored to have them as mentors.

Ananya Kulkarni - 01fe21bcs144
Swarna Patil - 01fe21bcs145
Amrutha Beedikar - 01fe21bcs198
Maanasi Shastri - 01fe21bcs210

CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Facial emotion recognition (FER) is a captivating field within computer vision and artificial intelligence that focuses on deciphering human emotions based on facial expressions. This technology has far-reaching applications, from human-computer interaction to affective computing and mental health assessments. FER has evolved significantly with advancements in machine learning and deep learning techniques. Understanding human emotions is crucial in various domains, including human-computer interaction, gaming, and healthcare.

1.1 Preamble

The field of artificial intelligence has been paying significant attention to the relationship between computer vision and emotion recognition in the modern era. This work proposes the creation of a synthesizable AI model for Facial Emotion Recognition (FER), marking the beginning of a transformative journey. By utilizing the capabilities of Convolutional Neural Networks (CNN) and pre-trained models, we want to create a sophisticated system that can recognize facial expressions with accuracy and can be synthesized for broad implementation. Personalized digital marketing, mental health diagnoses, and human-machine interactions can all benefit from an understanding of emotions. Emotions are classified by facial recognition algorithms based on characteristics including lip position, eye movements, and facial muscle movements. A crucial stage in many applications, including advanced driver assistance systems, augmented and virtual reality, human-computer interface, and security systems, is the recognition of human emotions. The current study addresses the identification of human emotions in response to the increasing need for Facial Emotion Recognition (FER) in recent times.

1.2 Motivation

Our project's motivation stems from the critical role that FER plays in a variety of disciplines, which reflects the growing interest, demand, and adoption for it. FER's ability to improve human-computer interaction is one of the main driving forces. Interactions become more natural and responsive when systems can adapt dynamically to the feelings of users. This not only improves the user experience but also sets the stage for the next development in

naturally occurring, compassionate computing. Analyzing facial emotions is a useful method for tracking and evaluating mental health issues. We are motivated by the possibility of using FER as an objective, non-intrusive way to measure emotional health. Through sophisticated facial emotion analysis, this initiative hopes to add to the expanding body of research on the integration of technology into mental health diagnosis. In marketing, the capacity to interpret customer feedback is essential for developing focused and successful advertising campaigns. Our research is driven by the possibility that FER can offer priceless insights into customer sentiment, allowing marketers to precisely customize their strategies. This is in line with the current need for data-driven, emotionally charged marketing campaigns. The primary driving forces behind exploring the complexities of Facial Emotion Recognition are the growing interest, increasing demand, and extensive implementation of FER systems. The recognition of this research endeavor's revolutionary potential across various industries emphasizes its timeliness and importance.

Our investigation, as we work through the complexities of creating a synthesizable AI model for FER, is based on the conviction that resolving the aforementioned motives is essential to realizing the full promise of emotion recognition technology. In addition to adding to the body of knowledge, this research attempts to provide useful information that may influence the future direction of FER applications in the real world.

1.3 Literature Survey

The literature survey encompasses a series of significant studies that explore the performance and efficiency of AI and machine learning models, particularly focusing on their deployment on edge-computing devices such as the MAX78000.

The first study, published in the Proceedings of SPIE, evaluates the edge-computing capabilities of the MAX78000 AI microcontroller using datasets like MNIST and CIFAR-10. This microcontroller integrates an Arm Cortex-M4F CPU, a RISC-V RV32 core, and a 64-channel CNN accelerator, showcasing its robust architecture for deep learning tasks. The MNIST dataset achieved a remarkable Top 1 accuracy of 99.44% and a Top 5 accuracy of 99.77%, with an inference time of 1.413 milliseconds. The CIFAR-10 dataset, though less detailed in terms of accuracy, demonstrated a similar level of efficiency with an inference time of 1.715 milliseconds. These results underscore the MAX78000's capability to perform complex computations swiftly and accurately on edge devices.

The second study, titled "TinyissimoYOLO," introduces an ultra-lightweight, quantized object detection network designed for microcontrollers with limited memory and computational power. This model supports real-time detection and can handle up to three object classes, as demonstrated using the WiderFace and PascalVOC datasets. Evaluations show

that TinyissimoYOLO achieves up to 73.5% mean average precision (mAP) on restricted datasets, outperforming other microcontrollers in terms of energy efficiency and inference speed. This study highlights the potential of deploying sophisticated object detection models on resource-constrained edge devices.

In another notable study, "Wildlife Species Classification on the Edge," researchers proposed an energy-efficient method for real-time classification of wild animal species using the MAX78000FTHR board. Through experimentation with different deep neural network (DNN) models and image resolutions, the study achieved an impressive accuracy of 84.30% and an energy efficiency of 0.885 millijoules per inference. The research revealed that higher image resolutions generally improve classification accuracy but also impact inference time, memory usage, and energy consumption. This study emphasizes the balance between performance and efficiency, crucial for edge AI applications.

The study "Ultra-low Power DNN Accelerators for IoT" offers a comprehensive evaluation of the MAX78000 microcontroller's efficiency using five different CNN models across various datasets. This research highlights the importance of optimizing both models and data to achieve energy efficiency, particularly on ultra-low-power devices. The analysis encompasses operational latency, power consumption, and memory footprint, showcasing the MAX78000's suitability for battery-powered machine learning applications, especially in IoT and wearable technology domains.

Further insights are provided in the "AI and ML Accelerator Survey and Trends" paper, which outlines the incremental advancements in AI and ML accelerator technologies. This study presents an architectural overview demonstrating the integration of components from embedded computing, traditional high-performance computing (HPC), and high-performance data analytics (HPDA). The survey underscores the collaborative effort needed to effectively provide AI capabilities for various stakeholders, including decision-makers and analysts.

Lastly, the study on "Ultra-Low Power Keyword Spotting at the Edge" presents an optimized convolutional neural network (CNN) model for keyword spotting (KWS), designed for energy-efficient deployment on the MAX78000 accelerator. By eliminating computationally intensive preprocessing steps, such as Mel-frequency cepstrum coefficients (MFCC), the proposed approach achieves high accuracy (96.3%) and low energy consumption (251 microjoules per inference). This research highlights the potential of the MAX78000 in energy-efficient keyword spotting applications, making it highly suitable for low-power edge devices.

Overall, these studies collectively underscore the efficacy of the MAX78000 microcontroller and similar edge-computing devices in various AI and machine learning applications. They highlight the potential for real-time, low-power, and efficient processing in diverse domains, demonstrating significant advancements in the field of edge AI and its applications in real-world scenarios.

1.4 Problem Definition

Develop a deployable Facial Emotion Recognition model on an ultra-low power embedded system.
Input - Facial image.
Output - Facial emotion recognized from the input image.

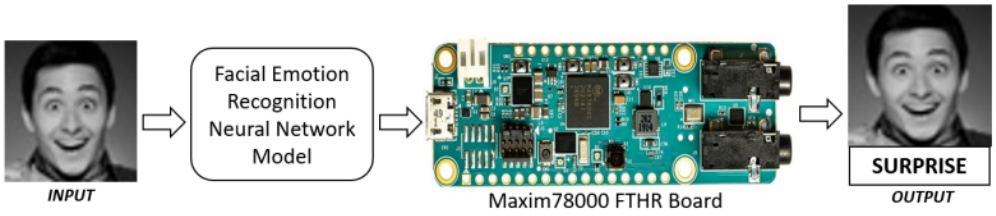

Figure 1.1: Design of FER System

1.5 Objectives

- Design of a Deep Neural network-based energy-efficient Facial Emotion Recognition Model.
- Development of the embedded system for the efficient deployment of the Facial Emotion Recognition Model.
- The model should be deployable on the ultra-low power

1.6 Scope and Constraints

1. The dataset is constrained to 7 emotion classes.
2. The dataset consists of grey-scale images only. Hence, it inherently demands less computational power and is more energy-efficient compared to datasets involving color images.

Chapter 2

REQUIREMENT ANALYSIS

A Software Requirements Specification (SRS) is a comprehensive document that serves as a blueprint for the development of a software application. It outlines the functionalities, features, and constraints that the software must adhere to.

2.1 Overview of SRS

The project aims to develop a synthesizable AI model for facial emotion recognition using gray-scale images, prioritizing computational efficiency. Objectives include image preprocessing, model construction, training for emotion classification, testing, and comparison with state-of-the-art techniques. The problem statement emphasizes the unique dataset characteristics, and functional requirements encompass user input, emotion classification, prediction, score generation, and performance analysis. Non-functional requirements focus on achieving a response time under 2 seconds, and a minimum accuracy of 70

2.2 Requirement Specifications

A requirements specification is a comprehensive document that lists all of the functional and non-functional requirements that a product or system needs to satisfy. It is an important stage in the software development life cycle when stakeholders work together to establish and record their requirements and expectations. The document usually contains comprehensive details regarding the features, functionalities, limitations, performance standards, and any other specifications that are considered necessary for the project's effective development and execution. The primary goal of a requirements specification is to establish a shared understanding among project stakeholders, including developers, designers, and users, about what the end product should achieve.

2.2.1 Functional Requirements

Functional requirements are a key component of a requirements specification and define the specific functionalities or features that the system or product must possess to meet the needs

and expectations of its users and stakeholders. These requirements describe what the system should do in terms of operations, behaviors, and interactions with its users or other systems.

- The user shall be able to input the gray-scale facial images for emotion recognition.
- The system shall classify and recognize a range of emotions such as happiness, sadness, anger, fear, etc., from the input images.
- The system shall predict the emotions.
- The system generates scores/accuracies for all the emotion classes.
- The user shall be able to view the performance analysis.

2.2.2 Use Case description

Use Case: Facial emotion recognition.

Actors: User and the developer.

Pre-Condition:
1. Grayscale facial images are available for emotion recognition.

Post-Condition:
1. Predicted emotions and associated scores are displayed.
2. Performance analysis, including accuracy metrics and efficiency, is available.

Main Success scenario:
1. The user accesses the facial emotion recognition system.
2. Grayscale facial images are provided as input.
3. The model is trained to classify emotions such as happiness, sadness, anger, etc.
4. The system classifies emotions in the provided images.
5. Predicted emotions and corresponding scores for each emotion class are generated.
6. The user is presented with a comprehensive performance analysis, including accuracy and efficiency metrics.

Exception scenario:
1. Insufficient Input:
Notifying the user if inadequate grayscale facial images are provided.
2. Operational Failures:

Handle errors during image pre-processing, model construction, or training with system logs and notifications.
3. Recognition Challenges:
Address ambiguity in emotion classification, performance analysis unavailability, or comparison process failures with user feedback and logs for investigation.

2.2.3 Non-functional Requirements

Non-functional requirements are another crucial aspect of a requirements specification, and they describe aspects of the system's behavior and characteristics beyond specific functionalities. These requirements focus on qualities that affect the overall performance, usability, and reliability of the system. Unlike functional requirements that specify what the system should do, non-functional requirements specify how well the system should do it.

- The system should achieve a response time of less than 2 seconds for emotion recognition processing, ensuring minimal delay in emotion recognition to enhance user experience.
- The emotion recognition system should achieve a minimum accuracy of more than 63 percent on standardized emotion recognition benchmarks.
- The system should achieve a score of at least 85 out of 100 on user satisfaction regarding ease of use, post-implementation.

2.3 Software and Hardware requirement specifications

Software requirements specify the functionalities, features, and constraints that the software application must adhere to. Hardware requirements specify the necessary physical components, devices, and infrastructure needed to run and support the software.

- Hardware:
  Edge Device: MAXIM 78000 FTHR Board
  USB Cables
  Core - Intel i5 or i6
  Memory for dataset: 60 MB
  GPU - nvidia GeForce

- Software:
  Operating system: windows or Linux

Deep learning framework: TensorFlow, PyTorch, and Keras

If you are using NVIDIA GPUs, installing CUDA (Compute Unified Device Architecture) and cuDNN (CUDA Deep Neural Network Library) can significantly accelerate deep learning computations.

Language: python

Libraries: NumPy, SciPy, and scikit-learn.

IDE: Jupiter Notebook, VS Code, or PyCharm

Chapter 3

PROPOSED SYSTEM

The proposed system model for Facial Emotion Recognition (FER) envisions an advanced architecture geared towards accurate and efficient emotion classification. This model incorporates sophisticated facial feature extraction, real-time processing, and precise emotion classification capabilities. The system's high-level architecture, as outlined, prioritizes seamless data flow, emphasizing the integration of cutting-edge technologies and algorithms. With a comprehensive overview of the workflow, proposed technologies, and performance metrics, this document serves as a foundational guide for the development team, project managers, and stakeholders, providing a clear roadmap for the creation of an innovative and effective FER system.

3.1 Description of Proposed System.

The proposed system model for our Facial Emotion Recognition (FER) project represents a state-of-the-art architecture that leverages Convolutional Neural Networks (CNN) and relevant pre-trained models to enhance the accuracy and efficiency of emotion classification. The system employs a multi-stage approach, incorporating facial feature extraction and leveraging the knowledge encoded in pre-trained models to capture intricate patterns in facial expressions. The CNN model serves as the core engine for robust emotion recognition, providing a foundation for real-time processing and accurate classification across a spectrum of emotions. This proposed system model is designed to push the boundaries of FER capabilities, offering a sophisticated solution that aligns with the latest advancements in deep learning and computer vision.

The envisaged system adopts an Enhanced Convolutional Neural Network (CNN) architecture for Facial Emotion Recognition (FER). Key facets and modifications in the model are elucidated below:
Addressing Class Imbalance through Oversampling:
The challenge of class imbalance in the training dataset is mitigated through the implementation of the oversampling technique. Each emotion class undergoes oversampling, ensuring a more equitable representation and augmenting the model's capacity to generalize across diverse emotional expressions.

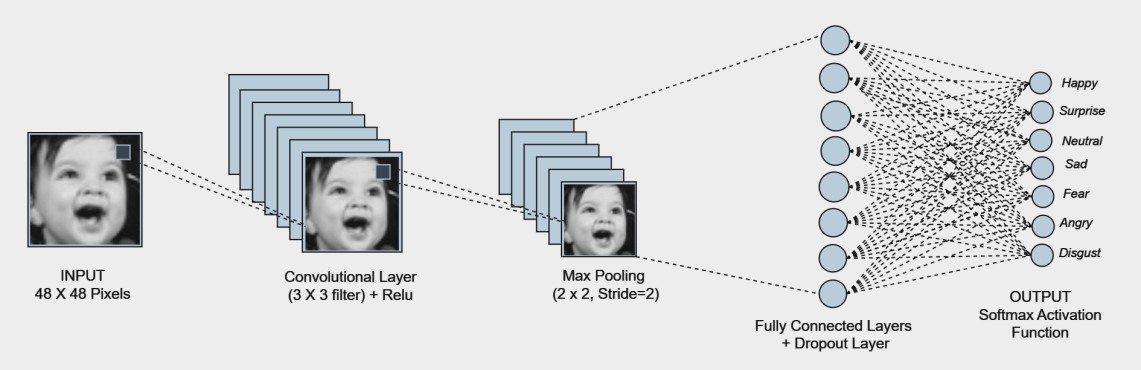

Figure 3.1: Architecture Diagram for the proposed CNN Model

3.2 Description of Target Users

The proposed system workflow delineates the sequential steps involved in processing facial images and recognizing emotions. It starts with the acquisition of facial images, proceeds to facial feature extraction, enters the emotion classification stage, and concludes with the output of recognized emotions. Each step is intricately connected, forming a seamless workflow that ensures efficient and accurate emotion recognition.

3.3 Applications of implemented FER System

- Human-Computer Interaction (HCI): Enhances computer interaction by allowing systems to respond to users' emotions in gaming, virtual reality, and interactive environments.

- Marketing and Advertising: Provides insights for marketers by analyzing facial expressions, helping tailor strategies and content to evoke specific emotional responses in advertisements.

- Healthcare and Well-being: Assists in early detection and monitoring of mental health conditions by analyzing changes in facial expressions over time.

- Education: Gauges student engagement and emotional states during lessons, allowing educators to adapt teaching methods to individual learning needs.

- Security and Surveillance: Enhances surveillance systems by identifying individuals displaying suspicious or abnormal emotional patterns in public spaces.

- Automotive Industry: Enhances driver safety by detecting signs of drowsiness or distraction, triggering alerts or safety features in vehicles.

- Robotics: Improves human-robot interaction by enabling robots to understand and respond to human emotions.

Chapter 4

SYSTEM DESIGN

The system design encompasses the architectural and operational aspects of the Facial Emotion Recognition (FER) system. It includes details about the model architecture, data preprocessing, training strategy, deployment of the system on the MAX78000 FTHR Board and evaluation processes.

4.1 Architecture of the system

Innovative Model Architecture:

- At the heart of the system lies the EnhancedCNN model, a refined Convolutional Neural Network, tailored for emotion classification.

4.1.1 CNN Architecture

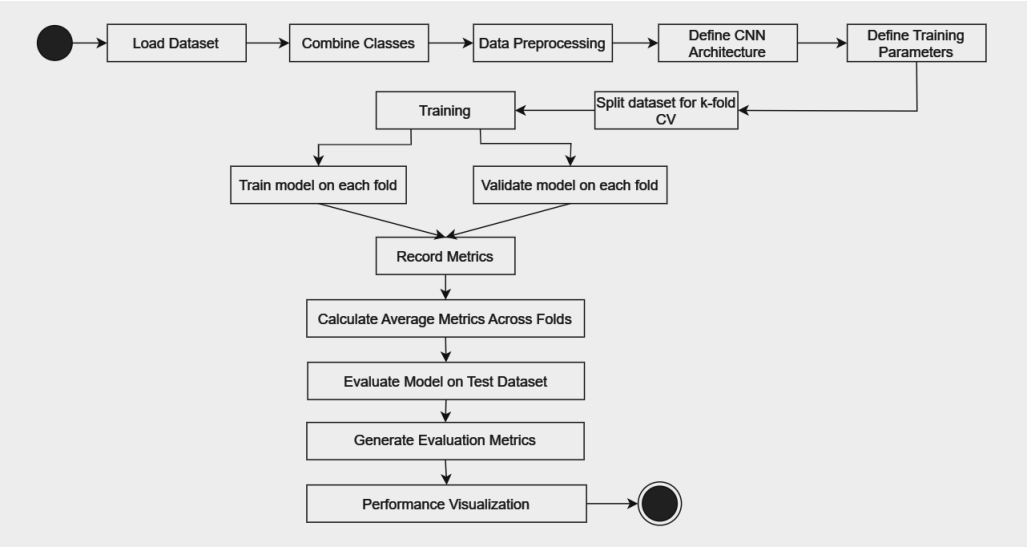

Figure 4.1: CNN Architecture diagram

The model comprises key elements:

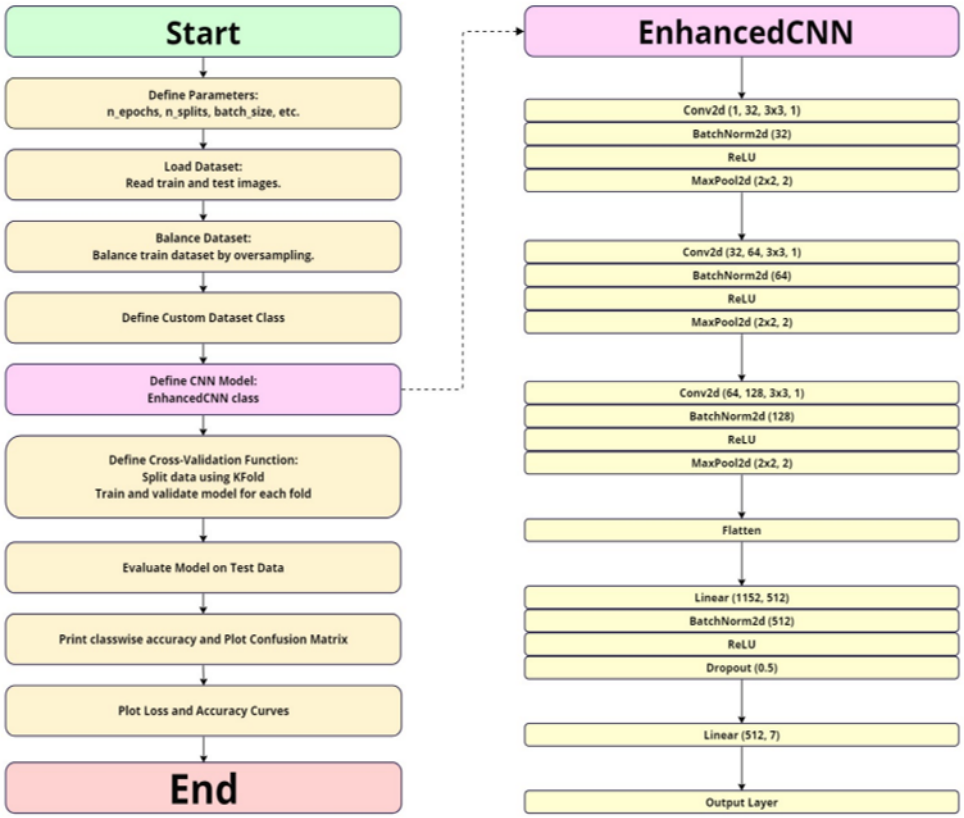

Figure 4.2: Detailed Enhanced-CNN Architecture diagram

- Introduction of a batch normalization layer featuring 2048 features to enrich feature representation.

- Incorporation of a customized fully connected layer dedicated to emotion classification.

- Inclusion of a dropout layer to enhance generalization by mitigating overfitting.

Data Preprocessing Strategies:

- Addressing class imbalance is pivotal, achieved through oversampling to ensure equitable representation of each emotion class, contributing to enhanced model generalization.

- Image preprocessing involves conversion to grayscale, resizing to (224, 224) pixels to align with ResNet50's input size, and normalization for consistent input to the model.

Optimized Training Approach:

- A robust training strategy is employed, featuring hyperparameter tuning to maximize model performance.

- Hyperparameter tuning encompasses fine adjustments to learning rates, epochs, and dropout rates. This iterative process aids in determining an optimal configuration.

- Implementation of early stopping, contingent on test loss, acts as a preventative measure against overfitting by halting training if no improvement is observed over a specified epoch range.

Comprehensive Evaluation Metrics:

- Evaluation of the system relies on standard metrics such as accuracy, precision, recall, and loss.

- Class-wise analysis provides a nuanced understanding of the model's performance across individual emotions, offering valuable insights for targeted refinements.

- After the deployment of the FER model, the performance is analyzed using the parameters -

    1. Accuracy

    2. Inference time

    3. Energency Effeciency

Rigorous Testing Protocol:

- The trained model undergoes evaluation on a distinct test dataset to gauge its generalization capabilities.

- Key metrics, including test accuracy, precision, recall, and loss, are reported to quantify and communicate the model's overall performance.

- The system design underscores the strategic integration of advanced neural network architectures, meticulous data preprocessing, and iterative training approaches to achieve a discerning and resilient Facial Emotion Recognition system.

4.2 Data Set Description

1. The dataset is taken from Kaggle[1].
2. Dataset size: 56.51MB.
3. The dataset contains 35,914 grayscale images of faces.
4. Testing images – 20.06
5. Training images – 79.93
6. The dataset consists of 7 classes. They are- Happy, sad, disgust, angry, fear, neutral, and surprise.
7. The images in the dataset are approximately of the size 2KB each.

Emotion Classes	**Number of Images**
Disgust	547
Fear	5121
Happy	8989
Neutral	6198
Sad	6077
Surprise	4002
Angry	4953

Table 4.1: Emotion Classes and the number of Images for each class in the dataset.

4.2.1 MAX78000 FEATHER BOARD

The MAX78000FTHR board features the MAX78000 microcontroller, which integrates a powerful Arm Cortex-M4 core with FPU and a specialized Convolutional Neural Network (CNN) accelerator for AI applications. It offers a compact form factor with essential peripherals such

as GPIOs, I2C, UART, and SPI interfaces, making it suitable for edge AI and IoT projects. The key components and features of the hardware are as follows:

- **1. MAX78000 Microcontroller**: This is the heart of the board, featuring a dual-core architecture with a 100 MHz Arm Cortex-M4 core and an ultra-low-power Cortex-M0+ core. The CNN accelerator enables efficient AI inference at the edge, capable of running complex neural networks with low power consumption.
- **2. Memory**:
  - **Flash Memory**: 512 KB of Flash memory for code storage.
  - **SRAM**: 128 KB of SRAM, providing ample space for data and stack operations.
- **3. Power Supply**:
  - **USB Power**: The board can be powered via a micro-USB connector.
  - **Battery Connector**: A JST connector for battery power, supporting Li-Po batteries for portable applications.
- **4. Communication Interfaces**:
  - **UART**: For serial communication.
  - **SPI**: For high-speed peripheral communication.
  - **I2C**: For interfacing with various sensors and peripherals.
  - **I2S**: For audio data transmission.
  - **USB**: For power and data transfer.
  - **GPIO Pins**: A set of general-purpose input/output pins that can be used for interfacing with other hardware components like sensors, actuators, and LEDs.
- **5. Camera Interface**: An 8-bit parallel camera interface, ideal for vision-based applications.
- **6. Analog Interfaces**:
  - **ADC**: An analog-to-digital converter for reading analog signals from sensors.
  - **DAC**: A digital-to-analog converter for outputting analog signals.
- **Debugging**:
  - **SWD Interface**: Serial Wire Debug interface for debugging and programming.
  - **On-board LEDs**: For status indication and debugging purposes.

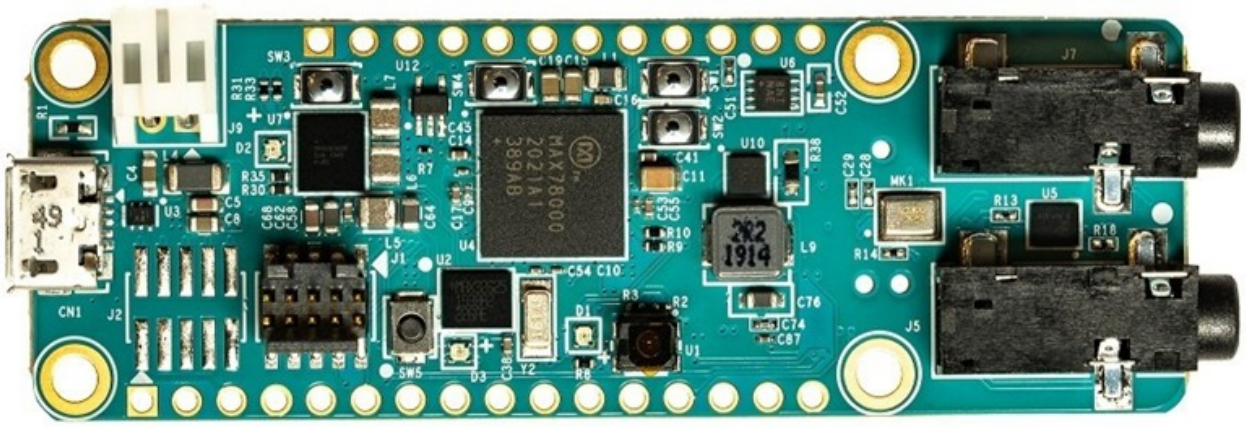

Figure 4.3: MAX78000fthr board

- **7. Form Factor**: The board is designed in a Feather-compatible form factor, making it easy to integrate with other Feather ecosystem boards and accessories.
- **8. Development Support**: The board is supported by comprehensive development tools and software libraries, including support for TensorFlow Lite for Microcontrollers, making it easier to deploy AI models on the hardware.

The MAX78000FTHR board is a powerful development platform with the MAX78000 microcontroller, featuring dual-core Arm Cortex-M4 and Cortex-M0+ for efficient edge AI. It supports a wide temperature range and includes peripherals such as UART, SPI, I2C, I2S, USB, GPIO pins, and a camera interface. Advanced features include hardware security, low-power modes, a real-time clock, analog comparators, and DSP instructions. Compatible with popular development environments, it is ideal for AI, IoT, and edge computing projects.

This combination of features makes the MAX78000FTHR board suitable for a wide range of applications, particularly those requiring on-device AI processing, such as image and speech recognition, sensor fusion, and other edge AI applications.

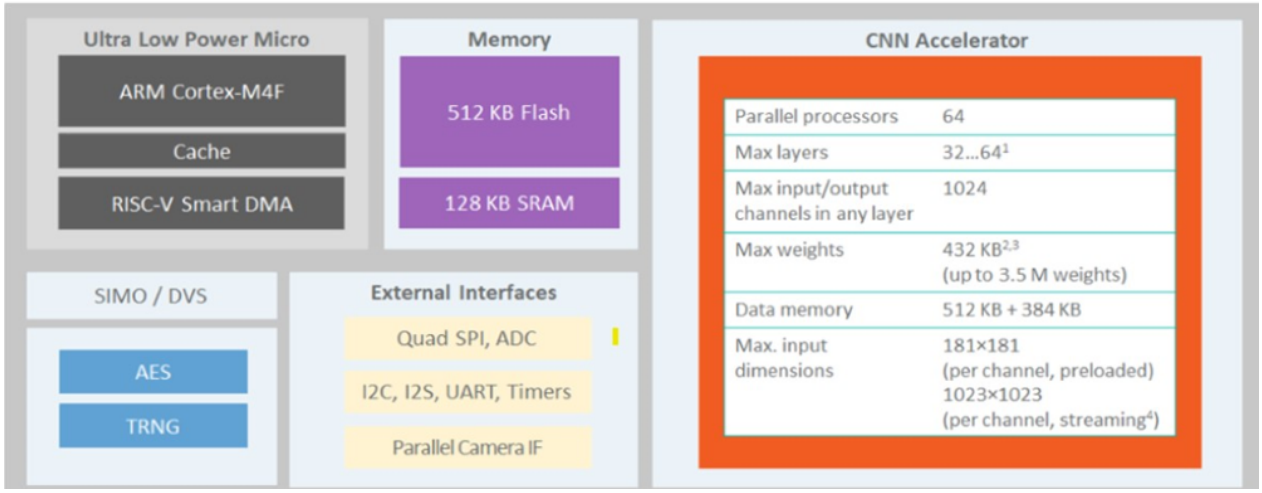

Figure 4.4: MAX78000fthr board - Key components

Chapter 5

IMPLEMENTATION

The implementation of Convolutional Neural Networks (CNN) represents a significant advancement in the field of computer vision, particularly in tasks such as image classification and feature extraction. CNNs are deep neural networks specifically designed for processing structured grid data, making them highly effective for image-related tasks. Their architecture includes convolutional layers, pooling layers, and fully connected layers, allowing them to automatically learn hierarchical features from images. In this project, an enhanced CNN model was developed for facial emotion recognition, achieving an accuracy of 56.52%. This model is deployed on the edge device MAX78000, enabling real-time facial emotion recognition. The implementation involves careful consideration of model architecture, hyperparameter tuning, and training on large datasets. Moreover, pre-trained models available in popular deep learning frameworks like TensorFlow and PyTorch have streamlined the implementation of CNNs in various applications, providing a powerful foundation for image-based tasks in computer vision.

5.1 Proposed Methodology

The total number of train images after pre-processing is 50,505. The total number of test images after pre-processing is 12,418. The architecture used for implementation is Convolutional Neural Networks(CNN). The CNN consists of three sets of Convolutional-ReLU-MaxPooling layers for feature extraction and down-sampling.

This architecture is designed to learn and extract meaningful features from facial expression images, enabling the model to classify emotions into different categories based on the learned features. Adjustments or variations in the architecture can be made for specific performance or task-related requirements.

Initially, our model underwent 100 epochs of training with a learning rate of 0.0001, achieving an accuracy of 33.38%.

During the evaluation, we observed its stronger performance particularly in the ‘surprise’, ‘happy’, and ‘disgust’ classes.

Subsequently, we extended the number of epochs to 250, achieving an accuracy of 45.77%.

Despite this increase in training duration, there was only a marginal improvement in the performance of the classes labeled as 'angry,' 'neutral,' and 'sad.'

Data Balancing Impact: Oversampling proved effective in balancing the dataset, showcasing positive outcomes in classes with both the highest and lowest sample numbers ('surprise', 'happy', and 'disgust').

Effect of Training Epochs: Raising epochs from 100 to 250 moderately boosted overall accuracy (33.38% to 45.77%), but improvements plateaued, particularly for classes like 'angry,' 'neutral,' and 'sad.'

Stronger Performance in Specific Classes: There is a tendency for the model to confuse classes such as 'angry', 'fear', and 'neutral', often predicting them as 'sad'.

Limited Improvement for Certain Classes: Extended training showed minimal performance improvement for classes 'angry,' 'neutral,' and 'sad.' This suggests complexity in discerning these classes, hinting at the need for diverse data or alternative approaches.

Potential Learning Plateau: Marginal overall accuracy improvement with a significant increase in epochs hints at a learning plateau. The model may be reaching its capacity for further improvement, encountering diminishing returns. Therefore, we opted for a learning rate of 0.001 and extended to 500 epochs as the graph shows the model hasn't converged. This suggests the need for additional learning.

Class-wise observations:
Epoch: 25
Learning rate: 0.0001
From the 1st to the 25th epoch, the model is able to recall images from class "surprise" effectively. The class "fear" is recalled inefficiently. Class "happy" has the maximum precision as it predicts the most correctly.

Epoch: 50
Learning rate: 0.0001
The recall score of class "surprise" reduced from 81 percent to 64 percent and the other classes like "sad", "angry", "neutral" started contributing.

Epoch: 74

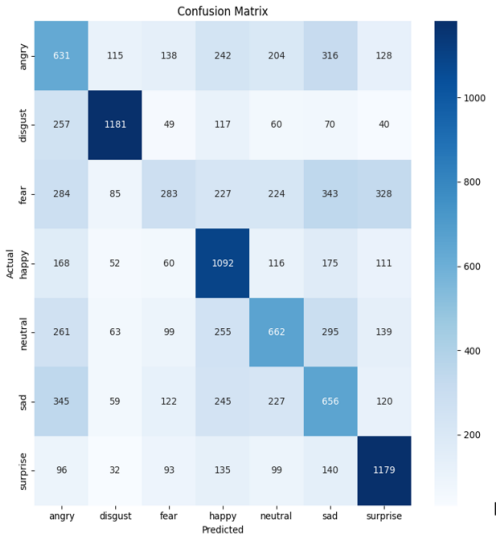

Figure 5.1: Confusion matrix for 250th epoch

Learning rate: 0.0001

From figure 5.3, the following observations can be drawn: The recall score of class "surprise" was further reduced and the contribution of all the other classes increased, with "sad" being the 2nd most contributing class. The class happy still has the highest accuracy and continues to predict the most correctly. The class "happy" has the maximum precision throughout all the epochs and it predicts the best. As epochs increase, the recall score of the class that is contributing to maximum decreases gradually, and all the other classes that had less recall scores, start contributing. The overall test and train accuracies are 24.9 percent and 24.09 percent as of the 75th epoch with a learning rate of 0.0001.

The figure depicts a class-wise confusion matrix captured at the 74th epoch of training. It visualizes the model's classification accuracy across different classes, providing insights into how well the model distinguishes between different categories.

We further increased the learning rate from 0.0001 to 0.001 with the attempt to increase the test accuracy.

Number of epochs: 250

Learning rate: 0.001

Initially, our model underwent 250 epochs of training with a learning rate of 0.001, achieving an accuracy of 52.67%. During the evaluation, we observed its stronger performance particularly in the 'surprise', 'happy', and 'disgust' classes.

Number of epochs: 400

Learning rate: 0.001

Subsequently, we extended the number of epochs to 400, achieving an accuracy of 52.51%.

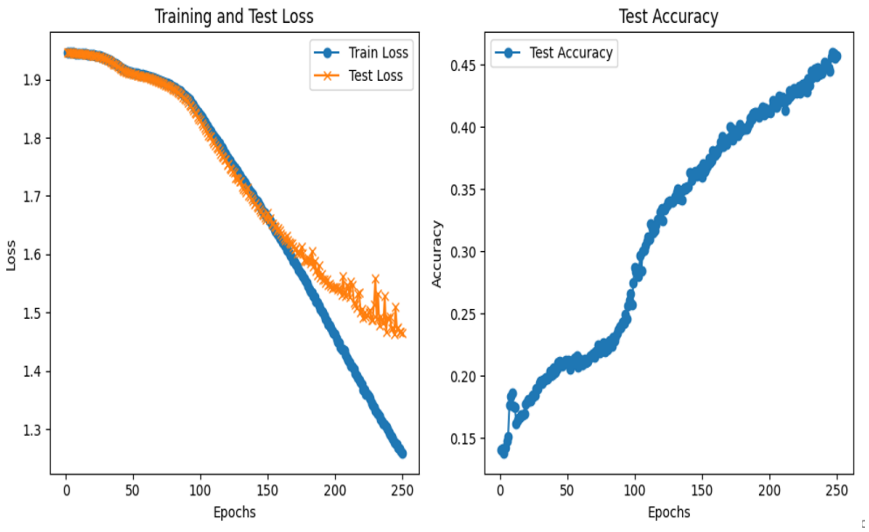

Figure 5.2: Graphs for train and test losses and accuracies

Despite this increase in training duration, there was only a marginal improvement in the performance of the classes labeled as 'angry,' 'neutral,' and 'sad.'

The figure displays the confusion matrix obtained at the 400th epoch of training using a CNN model. It visually represents the model's performance, showing the counts of true positive, true negative, false positive, and false negative predictions for each class.

The figure shows graphs illustrating the training and testing losses, as well as precision and recall scores, plotted against the number of epochs. These plots provide insights into the model's performance and learning progress over time.

By observing figure 5.3 below, we can say that the training loss gradually decreased as the number of training epochs increased, but these was as increase in the test loss because of imbalanced classes. We achieved a training accuracy of 98.77% at the 125th epoch of training and the graph converged to the x-axis. (The training accuracy was 98.7% throughout the training up to 400 epochs) The test accuracy remained 53% from the 60th epoch to the 400th epoch of training. The precision and recall values were 0.5204 and 0.5251 respectively from the 60th epoch onwards. Following is the confusion matrix fallowed by the loss and accuracy curves for 400 epochs:

The figure 5.1 depicts the confusion matrix at the 100th epoch for a CNN model. It highlights the model's classification performance by showing the number of correct and incorrect predictions for each class.

The figure 5.3 illustrates the loss curves over 100 epochs, showing the progression of training and validation losses as the CNN model learns.

The figure 5.4 displays the accuracy curves over 100 epochs, indicating how the training

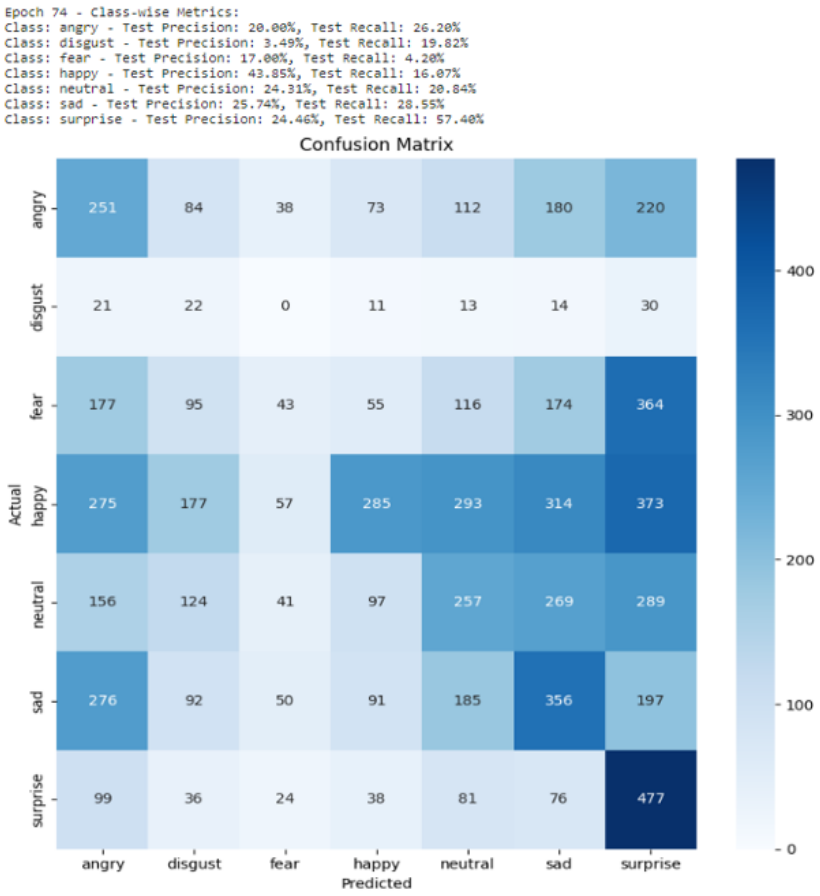

Figure 5.3: Confusion matrix: class-wise (74th epoch)

and validation accuracies change and improve during the training process.

Emotion	**Percentage**
Angry	59.73%
Disgust	53.15%
Fear	39.84%
Happy	77.00%
Neutral	47.04%
Sad	44.59%
Surprise	72.44%

Table 5.1: Class-wise Accuracies

The table presents the recognition percentages of various emotions by a classification model. Each row in the table represents an emotion and its corresponding recognition percentage.

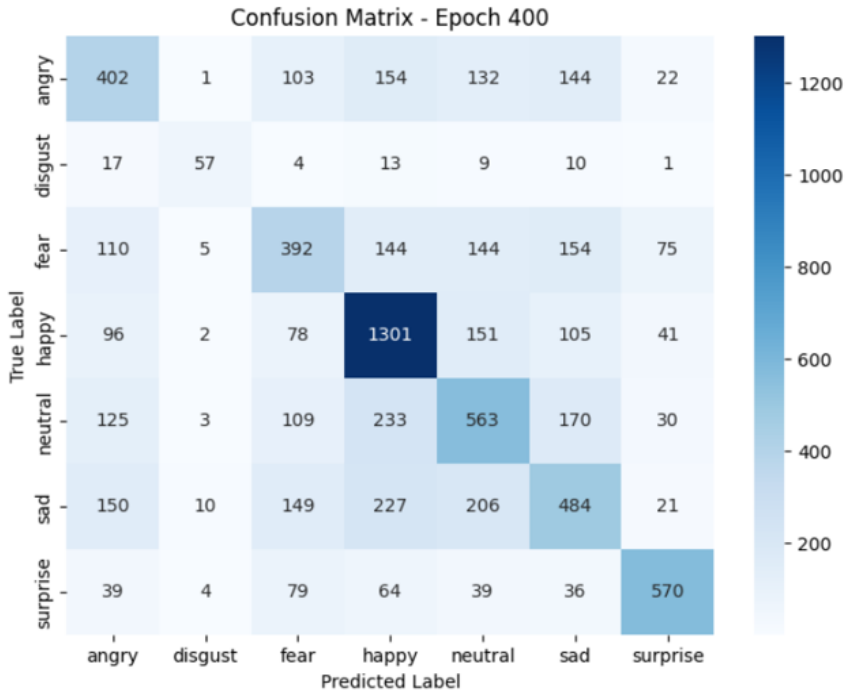

Figure 5.4: Confusion matrix at 400th epoch (Using CNN model)

5.2 Development Flow

The figure provides an overview of the development flow, illustrating the sequence of steps and processes involved in the development cycle.

1. **Training (ai8x-training)**
    - Begin with a PyTorch ai8x model and prepare it using a dataset and data loader.
    - Conduct quantization-aware training to adjust the model parameters.
    - After training, quantize the model to reduce its size and improve efficiency.
    - Evaluate the model's performance to ensure it meets the desired criteria.
    - Save the trained model weights into a checkpoint file for later use.
2. **Synthesis (ai8x-synthesis)**
    - Provide a model description in a YAML file format.
    - Use the MAX7800X synthesis tool to convert the model description into a format suitable for the hardware.
    - Input a data sample to test the synthesis process.
    - Generate embedded C code from the synthesized model.

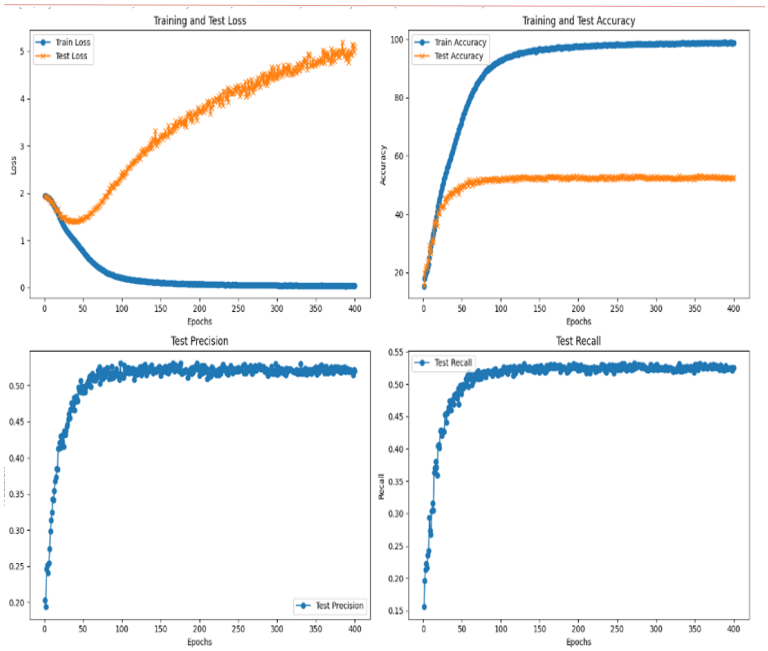

Figure 5.5: Graphs for training and testing losses, and precision and recall scores against epochs

3. **Deployment (Embedded SDK)**

   - Acquire live data for real-time processing.
   - Utilize the embedded C code generated from the synthesis step.
   - Process the user interface and output as necessary.
   - Deploy the finalized application onto the MAX7800X hardware for operational use.

   This streamlined workflow ensures a systematic approach from model training through to real-world deployment, optimized for performance and efficiency on the MAX78000 microcontroller platform. Each step addresses specific challenges inherent in deploying AI models in embedded systems, balancing computational constraints with application requirements.

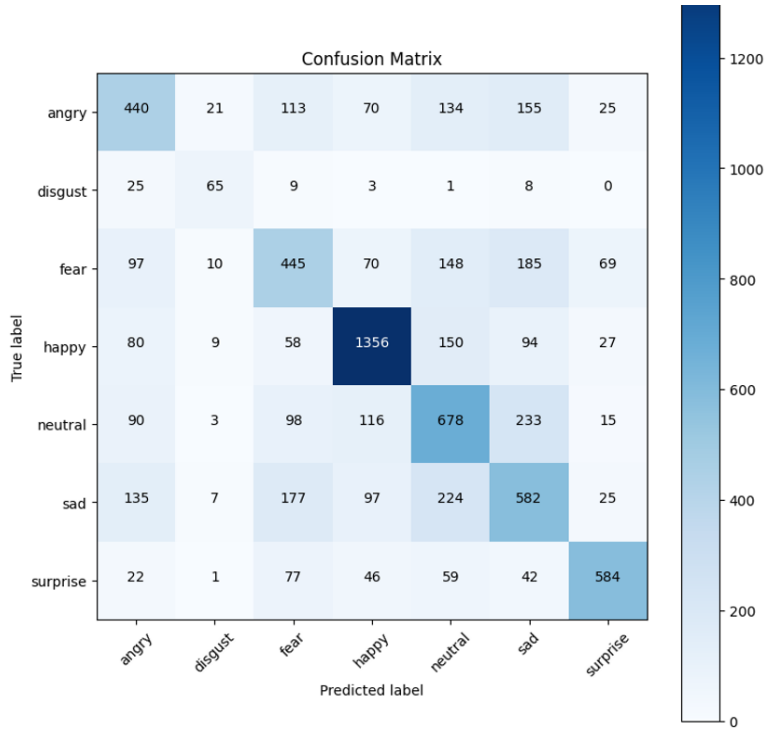

Figure 5.6: Confusion matrix at 100th epoch (Using CNN)

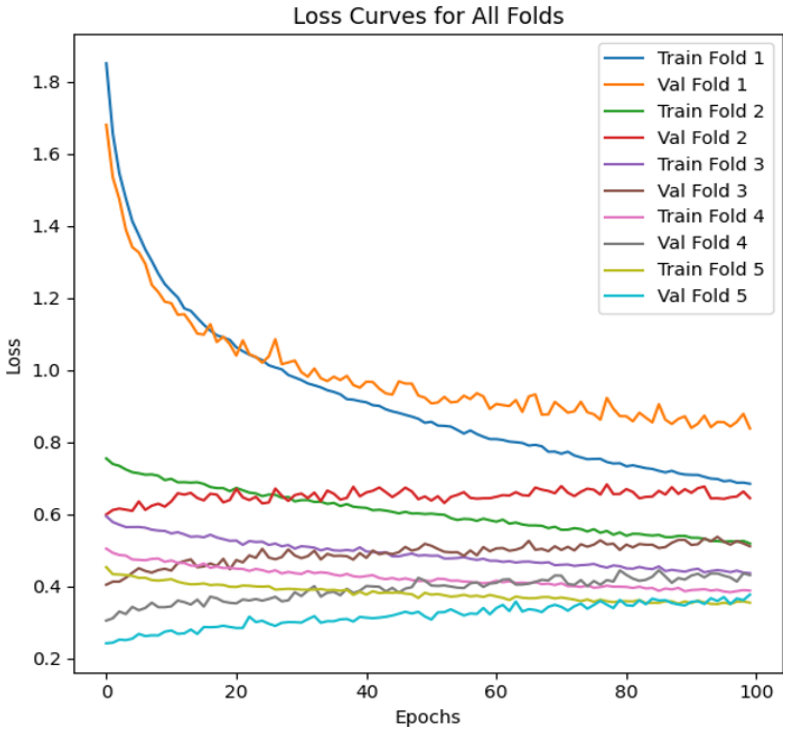

Figure 5.7: Loss curves for 100 epochs

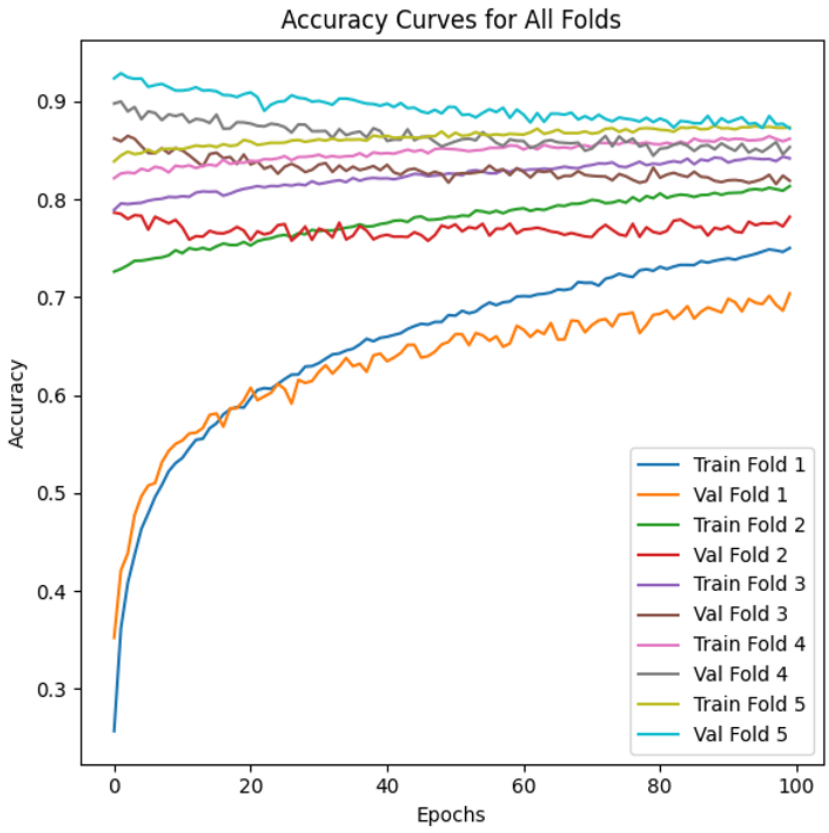

Figure 5.8: Accuracy curves for 100 epochs

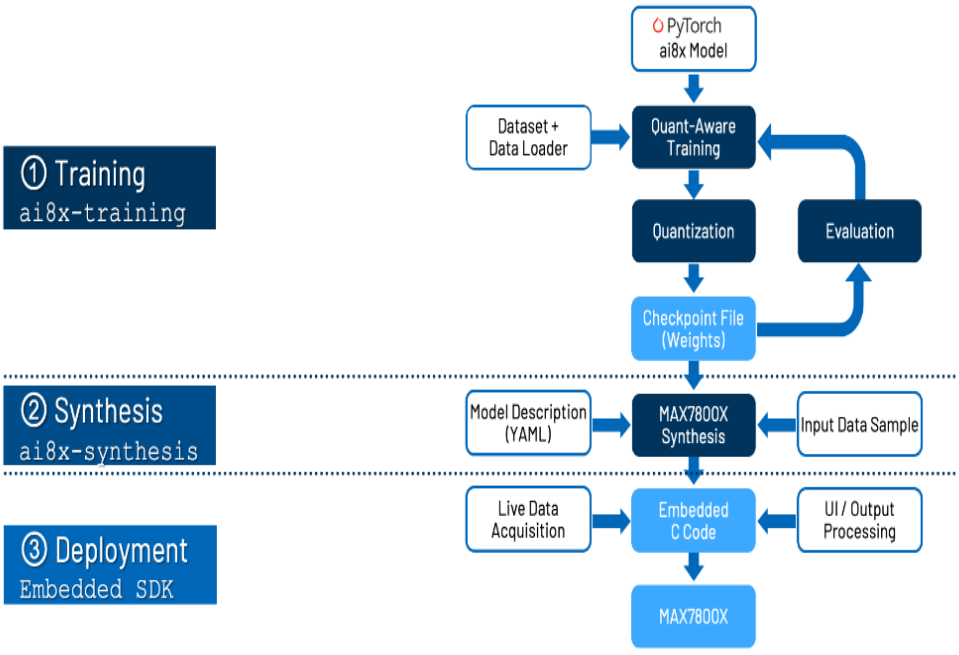

Figure 5.9: An overview of the development flow

Chapter 6

RESULTS AND DISCUSSIONS

Before deploying our final model, we implemented an intermediate model using a dataset containing two classes: cats and dogs. The results of this preliminary deployment along withe the input images are as follows:

Figure: Input Dog Image with Result

Figure: Input Cat Image with Result

Paramenter	cat-dogs	FER
Dataset size(no.of image in the train set)	25000	50000
Number of classes	2	7
Image size(in pixels)	64x64	28x28
Inference Time (in μs)	12055	1516

Figure 6.1: Comparison of Cat-Dogs and FER Model Parameters

The table presents a detailed comparison of key parameters for both the intermediate cat-dogs model and the final FER (Facial Emotion Recognition) model. This comparison includes dataset sizes, number of classes, image resolutions, and inference times, highlighting the differences in their configurations and performance metrics. For the cat-dogs model, we utilized a dataset comprising 25,000 images, categorized into 2 distinct classes (cats and dogs). Each image was resized to a resolution of 64x64 pixels. The model demonstrated an inference time of 12,055 microseconds, reflecting the time taken to classify an individual image. In contrast, the FER model was trained on a larger dataset consisting of 50,000 images, distributed across 7 emotion classes. The images in this dataset were resized to a smaller resolution of 28x28 pixels. Despite the increased complexity due to a higher number of classes, the FER model achieved a significantly faster inference time of 1,516 microseconds. This comparison underscores the advancements in model efficiency and capability, particularly in handling more complex datasets with larger class distributions while maintaining, or even improving, inference speeds.

This comparison underscores the transition from binary to multi-class classification and the corresponding increase in model complexity.

The notable reduction in inference time for the FER model, despite its larger dataset and higher class count, highlights improvements in model optimization. These advancements demonstrate our capability to handle more sophisticated tasks efficiently, paving the way for more comprehensive applications in facial emotion recognition. The improvements in model performance and efficiency reflect the robustness and scalability of our CNN architecture, ensuring readiness for future deployments in more complex and varied real-world scenarios.

After deploying the cat-dogs model, we proceeded to deploy our CNN model using the dataset with 7 emotion classes. The results, along with the input image, are provided.

Figure 6.2: Input image

```
Waiting...

*** CNN Inference Test ai85-fer ***

*** PASS ***

Approximate data loading and inference time: 1521 us

Classification result:
[  14073] -> angry
```

Figure 6.3: Deployment Results of FER on MAX78000FTHR BOARD (Output)

The CNN model outperforms the basic approach in terms of class-wise test accuracy, achieving 56.52% compared to the basic model's 45.16%. Additionally, the class-wise test accuracies for emotions such as anger, disgust, happiness, neutral, sadness, and surprise generally show improvements with the CNN model, indicating its effectiveness in capturing more complex patterns. The overall test accuracy combined with class-wise results in the CNN model is 56.52%, surpassing the basic model's overall test accuracy of 52.20%. However, it's important to note that both models face challenges in classifying certain emotions. For instance, the disgust class has low precision and recall for both models, suggesting difficulties in correctly identifying this emotion. The class-wise metrics for the CNN model reveal notable improvements in precision and recall for various emotions, such as happiness and surprise. The CNN model demonstrates higher precision and recall for surprise, indicating its ability to better detect this emotion compared to the basic approach. Furthermore, the CNN model's success in nuanced emotional differentiation suggests promising avenues for integrating effective computing into diverse fields, from personalized user interfaces to mental health diagnostics.

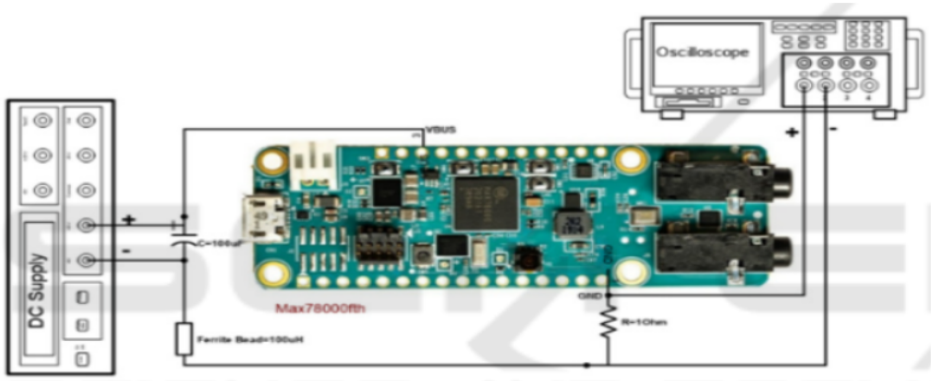

Figure: circuit diagram

Inference Energy = I * V * inference time
I: Current (mA)
V: Voltage (V)

Figure 6.4: Circuit diagram of MAX78000FTHR BOARD

Inference Time = 1516 µs
Inference Energy = I (mA) * V (V) * inference time (µs)
Inference Energy = 3.16 * 5 * 1.516
Inference Energy = 0.0237 mJ

The advanced CNN model not only improves overall test accuracy but also enhances the precision and recall for several specific emotions. This indicates its superior ability to differentiate between subtle emotional expressions. Despite these advancements, certain emotions like disgust remain challenging to classify accurately, underscoring areas for potential further refinement. The superior performance in detecting emotions like happiness and surprise demonstrates the CNN model's enhanced capability to capture and interpret complex facial features. Additionally, the increased accuracy in recognizing multiple emotions reflects the model's robustness and potential for application in real-world emotion recognition tasks. These results highlight the significance of employing sophisticated neural networks for improving emotion detection and classification accuracy.

In conclusion, while the CNN model generally outperforms the basic approach in terms of overall accuracy and specific class-wise metrics, there are still challenges in accurately classifying certain emotions. Further model fine-tuning or exploring advanced architectures may help address these challenges and enhance overall performance. Additionally,

the model is deployed on the edge device MAX78000 for real-time facial emotion recognition.

Chapter 7

CONCLUSIONS AND FUTURE SCOPE

The deployment of the CNN model for facial emotion recognition (FER) on the edge device MAX78000 represents a significant advancement in real-time applications such as healthcare, education, and human-computer interaction systems. The model has achieved an overall test accuracy of 56.52%, demonstrating notable capability in identifying complex emotional states such as anger, happiness, neutral, sadness, and surprise. Despite these advancements, challenges remain in accurately classifying emotions like disgust, where precision and recall are lower. Future developments should prioritize several key areas to enhance the system's effectiveness and robustness. Fine-tuning the CNN model with advanced architectures like Vision Transformers could improve overall accuracy and address specific challenges in emotion classification. Integrating multimodal inputs, such as voice analysis and physiological signals, holds promise for enhancing recognition accuracy and providing a comprehensive understanding of human affective states. Optimizing the model for edge devices like the MAX78000 is crucial to meet stringent requirements for latency, power consumption, and real-time processing. Ongoing validation in diverse real-world scenarios will be essential to assess the model's performance across different demographics and environmental conditions. Addressing ethical considerations, particularly concerning privacy and data security, should be integrated into the development process to ensure compliance and trustworthiness in deployment.

7.0.1 Overall Impact:

- The FER system, with ImprovedCNN at its core, presents robustness in recognizing facial emotions, laying the groundwork for applications in human-computer interaction and affective computing. Further research can address challenges and expand the system's capabilities.

The CNN model outperforms the basic approach in terms of class-wise test accuracy, achieving 56.52% compared to the basic model's 45.16%. Specifically, class-wise test accuracies for emotions such as anger, disgust, happiness, neutral, sadness, and surprise show notable improvements with the CNN model, demonstrating its effectiveness in capturing more complex patterns. The overall test accuracy of the CNN model is 56.52%, surpassing the basic model's overall accuracy of 52.20%. However, both models face challenges in classifying certain emotions, particularly the disgust class, which has low precision and recall for both models. Despite this, the CNN model shows significant improvements in precision and recall for various emotions, such as happiness and surprise, indicating a better performance in detecting these emotions compared to the basic approach. In conclusion, the CNN model generally outperforms the basic approach in terms of overall accuracy and specific class-wise metrics. Nonetheless, there are still challenges in accurately classifying certain emotions, suggesting the need for further model fine-tuning or the exploration of advanced architectures to enhance overall performance.

Furthermore, the CNN model is deployed on the edge device MAX78000, enabling real-time facial emotion recognition. This deployment on an edge device is crucial for applications requiring low latency and real-time processing, such as in healthcare, education, and human-computer interaction systems. The deployment on the MAX78000 ensures efficient performance, making the system highly suitable for practical, real-world use cases.

REFERENCES

[1] FER Dataset from Kaggle. Facial emotion recognition dataset. Available at: `https://www.kaggle.com/datasets/chiragsoni/ferdata/data`.

[2] Mitchell Clay, Christos Grecos, Mukul Shirvaikar, and Blake Richey. Benchmarking the MAX78000 artificial intelligence microcontroller for deep learning applications. In Nasser Kehtarnavaz and Matthias F. Carlsohn, editors, *Real-Time Image Processing and Deep Learning 2022*, volume 12102, page 1210207. International Society for Optics and Photonics, SPIE, 2022.

[3] Mateo Avila Pava Harshit Vaishya Yazan Tabak Juergen Ernst Ruben Portas Wanja Rast Joerg Melzheimer Ortwin Aschenborn Theresa Goetz Subodh Ingaleshwar, Farid Tasharofi and Stephan Goeb. Wildlife species classification on the edge: A deep learning perspective. In *16th International Conference on Agents and Artificial Intelligence (ICAART)*, 2024.

[4] Julian Moosmann, Marco Giordano, Christian Vogt, and Michele Magno. Tinyissimoyolo: A quantized, low-memory footprint, tinyml object detection network for low power microcontrollers, 05 2023.

[5] Author PictureLei Xun Author PictureChulhong Min Author PictureFahim Kawsar Author PictureAlessandro Montanari Author PictureArthur Moss, Author PictureHyunjong Lee, 20232, month = 11, pages = 934–940, title = Ultra-Low Power DNN Accelerators for IoT: Resource Characterization of the MAX78000 url =https://doi.org/10.1145/3560905.3568300.

[6] Albert Reuther, Peter Michaleas, Michael Jones, Vijay Gadepally, Siddharth Samsi, and Jeremy Kepner. Ai and ml accelerator survey and trends. In *2022 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, September 2022.

[7] Mehmet Gorkem Ulkar and Osman Erman Okman. Ultra-low power keyword spotting at the edge, 2021.

[8] Rahaf Abdulaziz Alawwad, Ouiem Bchir, and Mohamed Maher Ben Ismail. Arabic sign language recognition using faster r-cnn. *International Journal of Advanced Computer Science and Applications*, 12(3), 2021.

[9] Gian Karlo R. Madrid, Rane Gillian R. Villanueva, and Meo Vincent C. Caya. Recognition of dynamic filipino sign language using mediapipe and long short-term memory. In *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–6, 2022.

[10] Sneha Sharma, R Sreemathy, Mousami Turuk, Jayashree Jagdale, and Soumya Khurana. Real-time word level sign language recognition using yolov4. In *2022 International Conference on Futuristic Technologies (INCOFT)*, pages 1–7, 2022.

[11] S. A. M. A. Senanayaka, R. A. D. B. S. Perera, W. Rankothge, S. S. Usgalhewa, H. D. Hettihewa, and P. K. W. Abeygunawardhana. Continuous american sign language recognition using computer vision and deep learning technologies. In *2022 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6, 2022.

[12] Amrutha K, Prabu P, and Ramesh Chandra Poonia. List: A lightweight framework for continuous indian sign language translation. *Information*, 14:79, 2023.

[13] Ruchi Thakkar, Jay Patel, Aakash Shah, and Shashwat Vyas. Real-time sign language recognition using hand gesture recognition. In *2022 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)*, pages 1–5, 2022.

[14] Wenlong Yang, Huihui Sun, Yuliang Zhang, Qiao Zhou, Li Li, and Wenxin Yan. Sign language recognition based on deep learning. In *2022 International Conference on Computer Science and Application Engineering (CSAE)*, pages 1–5, 2022.

[15] Vijay Singh and Rajeev Singh. Real-time indian sign language recognition system using openpose. *Journal of Signal Processing Systems*, pages 1–10, 2022.

[16] Li Jiang, Wei Wang, Shiyong Li, Jiawei Zhang, and Xin Chen. Sign language recognition with 3d pose estimation and attention mechanism. *IEEE Access*, 11:44258–44268, 2023.

[17] Tung Hieu Vuong, Tuan Anh Le, Ngoc Phuong Nguyen, Van Anh Huynh, and Minh Tri Nguyen. Hand shape recognition in vietnamese sign language using convolutional neural networks. In *2023 4th International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, pages 131–136, 2023.

[18] Khalid Mohammed and Ali Al-Naji. Real-time gesture recognition system for american sign language using deep learning. In *2023 IEEE International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 1–6, 2023.

[19] Ying Chen, Wen Liu, and Wee Ping Tang. Exploring multimodal deep learning for american sign language recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):164–177, 2023.

[20] Jihye Kim, Hyungjong Lee, and Sunho Park. Sign language recognition with transformer-based networks. In *2023 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10, 2023.

[21] Liang Wu, Huachao Zhang, Jiajia Liu, and Xuan Wang. Deep learning-based american sign language recognition with enhanced hand pose estimation. In *2022 6th International Conference on Signal Processing and Information Communication (ICSPIC)*, pages 1–6, 2022.

[22] Xinyu Wu, Yu Zhang, Zhiyong Liu, and Jianjun Li. Sign language recognition system using spatio-temporal features and transformer networks. *IEEE Transactions on Cybernetics*, 53(1):365–378, 2023.

[23] Jiwoo Park, Minjeong Kim, Jihye Lee, Haejoon Choi, and Hyungjong Lee. Real-time korean sign language recognition using hand gesture recognition. In *2023 International Conference on Multimedia Systems and Applications (ICMSA)*, pages 1–5, 2023.

[24] Ming Zhang, Wei Chen, and Yan Li. Sign language recognition system based on convolutional neural networks and visual attention mechanism. *IEEE Access*, 11:21371–21381, 2023.

[25] Parv Singh, Vivek Gupta, Sachin Jain, and Sumit Raj. Dynamic sign language recognition using temporal convolutional networks. In *2023 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1–5, 2023.

[26] Xin Chen, Wei Wang, Li Jiang, Shiyong Li, and Jiawei Zhang. Multimodal sign language recognition using graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(1):123–137, 2023.

[27] Chaiyong Lee, Kritsada Saelee, Sudsanguan Tawinwung, and Rungrawee Phoophuang. Real-time thai sign language recognition using transformer and graph convolutional networks. In *2022 9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 1–6, 2022.

REFERENCES

[1] FER Dataset from Kaggle. Facial emotion recognition dataset. Available at: `https://www.kaggle.com/datasets/chiragsoni/ferdata/data`.

[2] Mitchell Clay, Christos Grecos, Mukul Shirvaikar, and Blake Richey. Benchmarking the MAX78000 artificial intelligence microcontroller for deep learning applications. In Nasser Kehtarnavaz and Matthias F. Carlsohn, editors, *Real-Time Image Processing and Deep Learning 2022*, volume 12102, page 1210207. International Society for Optics and Photonics, SPIE, 2022.

[3] Mateo Avila Pava Harshit Vaishya Yazan Tabak Juergen Ernst Ruben Portas Wanja Rast Joerg Melzheimer Ortwin Aschenborn Theresa Goetz Subodh Ingaleshwar, Farid Tasharofi and Stephan Goeb. Wildlife species classification on the edge: A deep learning perspective. In *16th International Conference on Agents and Artificial Intelligence (ICAART)*, 2024.

[4] Julian Moosmann, Marco Giordano, Christian Vogt, and Michele Magno. Tinyissimoyolo: A quantized, low-memory footprint, tinyml object detection network for low power microcontrollers, 05 2023.

[5] Author PictureLei Xun Author PictureChulhong Min Author PictureFahim Kawsar Author PictureAlessandro Montanari Author PictureArthur Moss, Author PictureHyunjong Lee, 20232, month = 11, pages = 934–940, title = Ultra-Low Power DNN Accelerators for IoT: Resource Characterization of the MAX78000 url =https://doi.org/10.1145/3560905.3568300.

[6] Albert Reuther, Peter Michaleas, Michael Jones, Vijay Gadepally, Siddharth Samsi, and Jeremy Kepner. Ai and ml accelerator survey and trends. In *2022 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE, September 2022.

[7] Mehmet Gorkem Ulkar and Osman Erman Okman. Ultra-low power keyword spotting at the edge, 2021.

[8] Rahaf Abdulaziz Alawwad, Ouiem Bchir, and Mohamed Maher Ben Ismail. Arabic sign language recognition using faster r-cnn. *International Journal of Advanced Computer Science and Applications*, 12(3), 2021.

[9] Gian Karlo R. Madrid, Rane Gillian R. Villanueva, and Meo Vincent C. Caya. Recognition of dynamic filipino sign language using mediapipe and long short-term memory. In *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–6, 2022.

[10] Sneha Sharma, R Sreemathy, Mousami Turuk, Jayashree Jagdale, and Soumya Khurana. Real-time word level sign language recognition using yolov4. In *2022 International Conference on Futuristic Technologies (INCOFT)*, pages 1–7, 2022.

[11] S. A. M. A. Senanayaka, R. A. D. B. S. Perera, W. Rankothge, S. S. Usgalhewa, H. D. Hettihewa, and P. K. W. Abeygunawardhana. Continuous american sign language recognition using computer vision and deep learning technologies. In *2022 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6, 2022.

[12] Amrutha K, Prabu P, and Ramesh Chandra Poonia. List: A lightweight framework for continuous indian sign language translation. *Information*, 14:79, 2023.

[13] Ruchi Thakkar, Jay Patel, Aakash Shah, and Shashwat Vyas. Real-time sign language recognition using hand gesture recognition. In *2022 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)*, pages 1–5, 2022.

[14] Wenlong Yang, Huihui Sun, Yuliang Zhang, Qiao Zhou, Li Li, and Wenxin Yan. Sign language recognition based on deep learning. In *2022 International Conference on Computer Science and Application Engineering (CSAE)*, pages 1–5, 2022.

[15] Vijay Singh and Rajeev Singh. Real-time indian sign language recognition system using openpose. *Journal of Signal Processing Systems*, pages 1–10, 2022.

[16] Li Jiang, Wei Wang, Shiyong Li, Jiawei Zhang, and Xin Chen. Sign language recognition with 3d pose estimation and attention mechanism. *IEEE Access*, 11:44258–44268, 2023.

[17] Tung Hieu Vuong, Tuan Anh Le, Ngoc Phuong Nguyen, Van Anh Huynh, and Minh Tri Nguyen. Hand shape recognition in vietnamese sign language using convolutional neural networks. In *2023 4th International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, pages 131–136, 2023.

[18] Khalid Mohammed and Ali Al-Naji. Real-time gesture recognition system for american sign language using deep learning. In *2023 IEEE International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 1–6, 2023.

[19] Ying Chen, Wen Liu, and Wee Ping Tang. Exploring multimodal deep learning for american sign language recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):164–177, 2023.

[20] Jihye Kim, Hyungjong Lee, and Sunho Park. Sign language recognition with transformer-based networks. In *2023 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10, 2023.

[21] Liang Wu, Huachao Zhang, Jiajia Liu, and Xuan Wang. Deep learning-based american sign language recognition with enhanced hand pose estimation. In *2022 6th International Conference on Signal Processing and Information Communication (ICSPIC)*, pages 1–6, 2022.

[22] Xinyu Wu, Yu Zhang, Zhiyong Liu, and Jianjun Li. Sign language recognition system using spatio-temporal features and transformer networks. *IEEE Transactions on Cybernetics*, 53(1):365–378, 2023.

[23] Jiwoo Park, Minjeong Kim, Jihye Lee, Haejoon Choi, and Hyungjong Lee. Real-time korean sign language recognition using hand gesture recognition. In *2023 International Conference on Multimedia Systems and Applications (ICMSA)*, pages 1–5, 2023.

[24] Ming Zhang, Wei Chen, and Yan Li. Sign language recognition system based on convolutional neural networks and visual attention mechanism. *IEEE Access*, 11:21371–21381, 2023.

[25] Parv Singh, Vivek Gupta, Sachin Jain, and Sumit Raj. Dynamic sign language recognition using temporal convolutional networks. In *2023 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1–5, 2023.

[26] Xin Chen, Wei Wang, Li Jiang, Shiyong Li, and Jiawei Zhang. Multimodal sign language recognition using graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 34(1):123–137, 2023.

[27] Chaiyong Lee, Kritsada Saelee, Sudsanguan Tawinwung, and Rungrawee Phoophuang. Real-time thai sign language recognition using transformer and graph convolutional networks. In *2022 9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 1–6, 2022.

Appendix A

A.1 Gantt Chart

TIMELINE	WEEK 1- 5	WEEK 5-7	WEEK 7- 10	WEEK 10-15
Problem Identification, Literature Survey, dataset Identification and understanding, SRS, High level Design Framework and midway Implementation of AI model.				
Complete Implementation of AI model [Training of the Model].				
Deployment of the AI model on to an edge device.				
Performance Testing and Report.				

A.2 Description of Tools and Technology used

Kaggle:

Kaggle is a platform for data science and machine learning competitions. It provides datasets, a cloud-based workbench for data analysis, and a community where data scientists and machine learning enthusiasts can collaborate and share insights. Usage in the Project: Kaggle can be utilized for accessing datasets relevant to the project, participating in machine learning competitions, and engaging with the data science community for discussions and insights.

Visual Studio Code (VS Code):

Visual Studio Code is a lightweight and versatile code editor developed by Microsoft. It supports various programming languages, provides debugging capabilities, and offers

extensions for additional features. Usage in the Project: VS Code can serve as the primary integrated development environment (IDE) for coding tasks in the project. Its flexibility and extensibility make it suitable for a wide range of programming languages and project types.

Draw.io:

Draw.io is an online diagramming tool that allows users to create a variety of diagrams, flowcharts, and visual representations. It is user-friendly and provides a wide range of shapes and templates. Usage in the Project: Draw.io can be used for creating visual representations of project architectures, flowcharts, or any diagrams that help in explaining the project's design and workflow. It aids in visual communication and documentation. Each tool plays a specific role in the project, contributing to different aspects such as data exploration and analysis (Kaggle), code development (VS Code), project hosting (Server), and visual representation/documentation (Draw.io).

Integrating these tools efficiently can enhance collaboration, streamline development processes, and improve the overall project workflow.

Overleaf:

Overleaf is a collaborative online platform designed specifically for researchers, academics, and professionals in the fields of science, technology, engineering, and mathematics (STEM). It allows users to create, edit, and collaborate on LaTeX documents in real time, eliminating the need for manual version control and facilitating seamless teamwork. With its intuitive interface and extensive library of templates, Overleaf streamlines the process of writing research papers, reports, and academic documents, while also offering features such as real-time previewing, easy sharing and collaboration, and integration with other tools like GitHub. Overleaf significantly enhances the efficiency and effectiveness of the document creation and collaboration process, empowering researchers to focus more on their content and less on the technicalities of document formatting and management.

Eclipse:

Eclipse is a widely-used integrated development environment (IDE) primarily designed for Java development but also supports other programming languages through plugins.

It is an open-source platform developed by the Eclipse Foundation, known for its robust features, including a powerful code editor, debugging tools, and project management capabilities. Eclipse supports multiple programming languages such as Java, C/C++, Python, and more through various plugins and extensions. It also provides a flexible and customizable user interface, making it a preferred choice for developers working on large-scale and complex software projects. Its integration with version control systems like Git and SVN further enhances its utility in collaborative development environments.

Maxim SDK:

The Maxim SDK (Software Development Kit) is a comprehensive suite of tools and libraries provided by Maxim Integrated for the development of applications using their microcontroller and sensor products. This SDK includes a range of software components, drivers, and examples that simplify the process of developing and deploying embedded systems. It supports a variety of hardware platforms and provides resources for efficient power management, sensor integration, and communication protocols. The Maxim SDK is designed to help developers quickly prototype and bring to market innovative solutions in areas such as wearable technology, healthcare, and industrial applications. Its documentation and support resources are extensive, enabling developers to effectively leverage Maxim's hardware capabilities.

CUDA Python Environment:

The CUDA Python environment allows developers to harness the power of NVIDIA's CUDA (Compute Unified Device Architecture) for high-performance computing using Python. This environment combines the ease of Python programming with the parallel computing capabilities of NVIDIA GPUs. The key component of this environment is the Numba library, which enables the compilation of Python code into machine code that can run on the GPU. This approach significantly accelerates computations for tasks such as deep learning, scientific simulations, and data analysis. The CUDA Python environment provides an accessible way for Python developers to optimize their code for GPU execution, leveraging libraries like CuPy and PyCUDA to further enhance performance and scalability in computationally intensive applications.