Kaunas University of Technology

Faculty of Electrical and Electronics Engineering

Development and Research of Human Emotion, Age and Gender Recognition System for InMoov Humanoid Robot

Master’s Final Degree Project

Nicolás Augusto Pava Roldán

Project author

Assoc. Prof. Dr. Gintaras Dervinis

Supervisor

Kaunas, 2025

Kaunas University of Technology

Faculty of Electrical and Electronics Engineering

Development and Research of Human Emotion, Age and Gender Recognition System for InMoov Humanoid Robot

Master's Final Degree Project

Control Technologies (6211EX014)

Nicolás Augusto Pava Roldán
Project author

Assoc. Prof. Dr. Gintaras Dervinis
Supervisor

Prof. Dr. Vidas Raudonis
Reviewer

Kaunas, 2025

Kaunas University of Technology

Faculty of Electrical and Electronics Engineering

Nicolás Augusto Pava Roldán

Development and Research of Human Emotion, Age and Gender Recognition System for InMoov Humanoid Robot

Declaration of Academic Integrity

I confirm that the final project of mine, Nicolás Augusto Pava Roldán, on the topic "Development and Research of Human Emotion, Age and Gender Recognition System for InMoov Humanoid Robot" is written completely by myself; all the provided data and research results are correct and have been obtained honestly. None of the parts of this thesis have been plagiarised from any printed, Internet-based or otherwise recorded sources. All direct and indirect quotations from external resources are indicated in the list of references. No monetary funds (unless required by Law) have been paid to anyone for any contribution to this project.

I fully and completely understand that any discovery of any manifestations/case/facts of dishonesty inevitably results in me incurring a penalty according to the procedure(s) effective at Kaunas University of Technology.

Nicolás Augusto Pava Roldán

Aprooved electronically

Nicolás Augusto Pava Roldán. Development and Research of Human Emotion, Age, and Gender Recognition System for InMoov Humanoid Robot. Master's Final Degree Project / Assoc. Prof. Dr. Gintaras Dervinis; Faculty of Electrical and Electronics Engineering, Kaunas University of Technology.

Study field and area (study field group): Electronics Engineering, Engineering Science.

Keywords: Computer Vision, Robotics, HRI, Real-Time, Artificial Intelligence.

Kaunas, 2025. 77.

Summary

This work explores the challenge of real-time facial attribute recognition (emotion, age, gender) for Human-Robot Interaction (HRI), focusing on deployment in resource-limited embedded systems. A full pipeline was developed and tested, including comparisons between deep learning architectures (CNNs, transfer learning models, and a Hybrid CNN-Transformer), optimization with TensorRT, and real-time evaluation on a Jetson Xavier NX using both public datasets and a live experiment with 36 participants.

The Hybrid CNN-Transformer offered a good trade-off between performance and efficiency. With TensorRT FP16 optimization, the system reached real-time inference (64 FPS) for all three tasks, achieving up to 26× speedups without sacrificing accuracy.

However, the real-time experiment revealed challenges, especially in emotion recognition, where performance dropped due to class imbalance, subtle expressions, and pose variation. Gender and age tasks also showed some biases depending on head pose and demographics.

Overall, the thesis shows that real-time multi-task facial analysis on edge devices is feasible, but further work is needed on improving robustness, handling dataset imbalance, and mitigating bias – key steps to enable more perceptive and reliable social robots.

Nicolás Augusto Pava Roldán. Žmogaus emocijų, amžiaus ir lyties atpažinimo sistemos InMoov humanoidiniam robotui sukūrimas ir tyrimas. Magistro baigiamasis projektas / vadovas doc. dr. Gintaras Dervinis; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Elektronikos inžinerija, inžinerijos mokslai.

Reikšminiai žodžiai: Kompiuterinė rega, HRI, Robotika, Realus laikas, Dirbtinis intelektas.

Kaunas, 2025. Puslapių sk. 77 p.

Santrauka

Šiame darbe nagrinėjamas žmogaus veido atributų (emocijų, amžiaus, lyties) atpažinimo realiuoju laiku iššūkis žmogaus ir roboto sąveikai (ŽRI, angl. HRI (Human-Robot Interaction)), daugiausia dėmesio skiriant diegimui ribotus išteklius turinčiose įterptinėse sistemose. Buvo sukurta ir ištestuota keletas algoritmų, atlikti gilaus mokymosi architektūrų palyginimai (CNN, mokymosi perkeliamaisiais modeliais ir hibridinius CNN-Transformerius), atliktas optimizavimas naudojant „TensorRT“ir realiojo laiko vertinimas panaudojant „Jetson Xavier NX“bei naudojant tiek viešuosius duomenų rinkinius, tiek 36 dalyvių tiesioginio eksperimento duomenis.

Hibridinis CNN-Transformeris leido gauti pakankamai gerą rezultatą/kompromisą tarp efektyvumo ir našumo. Optimizavus su TensorRT FP16“, sistema patiekė rezultatus realiu laiku (64 FPS) visoms trims užduotims, pasiekdama iki 26 kartų didesnį greitį be tikslumo praradimo.

Tačiau realaus laiko eksperimentas atskleidė iššūkių, ypač emocijų atpažinimo srityje, kur našumas sumažėjo dėl klasės disbalanso, subtilių išraiškų ir pozų skirtumų. Lyties ir amžiaus užduočių rezultatai taip pat parodė tam tikrą priklausomybę nuo galvos padėties ir demografinių rodiklių.

Apskritai darbas parodė, kad realaus laiko daugiafunkcinė veido analizė periferiniuose įrenginiuose yra įmanoma, tačiau reikia toliau dirbti gerinant patikimumą, tvarkant duomenų rinkinių disbalansą ir mažinant tam tikrų faktorių įtaką – tai yra pagrindiniai žingsniai, siekiant sukurti perspektyvesius ir patikimesnius socialinius robotus.

Table of contents

List of figures

List of tables

List of abbreviations and terms

Abbreviations:

HRI - Human Robot Interaction;
AI - Artificial Intelligence;
HW - Hardware;
FPS - Figures Per Second;
FER - Facial Expression Recognition;
CNN - Convolutional Neural Networks;
MAE - Mean Absolute Error;
RMSE - Root Mean Squared Error;
SVM - Support Vector Machines;
KNN - K-Nearest Neighbors;
LDA - Linear Discriminant Analysis;
LBP - Local Binary Patterns;
HOG - Histogram of Oriented Gradients;
SIFT - Scale-Invariant Feature Transforms;
TOPS - Trillion Operations Per Second;
FP32 - 32-bit Floating-point numbers;
FP16 - 16-bit Floating-point numbers;
INT8 - 8-bit integers;
PTQ - Post-Training Quantization;
QAT - Quantization-Aware Training;
ONNX - Open Neural Network Exchange;

Introduction

Human-robot interaction (HRI) has gained significant attention in recent years, particularly with the rise of social robots. These robots, built to interact with humans in meaningful and intuitive ways, are gradually being integrated into numerous sectors of daily life, including healthcare and entertainment. However, the effectiveness of such interactions heavily relies on the robot’s ability to understand and respond to human emotions, age, and gender, all of which are fundamental components of social communication [3].

The ability to recognize emotions, age, and gender allows robots to better adapt to human needs, fostering more natural and empathetic interactions. These recognition capabilities enable robots to respond appropriately to emotional states, tailor their behavior to specific age groups, and ultimately create a more personalized user experience [4].

For example, in geriatric care, a social robot interacting with possibly solitary individuals may detect negative feelings such as melancholy or distress. Recognizing these signs, together with the user's projected age, may cause the robot to initiate diverting activities, play favored music, or engage in age-appropriate discussion [5]. Similarly, in educational settings, a robot tutor could detect signs of boredom or disengagement in a student. This awareness, combined with age recognition, allows the robot to adjust its teaching methods, perhaps by simplifying concepts or modifying task difficulty, to recapture attention and optimize learning [6]. Furthermore, as Ritschel et al. [7] point out, interaction personalization based on perceived user attributes has a positive impact; for example, detecting gender may allow robots communicating in grammatically gendered languages (such as Spanish or French) to use correct grammatical forms, increasing interaction naturalness.

The practicality of constructing models capable of executing these sophisticated recognition tasks has improved dramatically as artificial intelligence (AI) advances, notably in deep learning and computer vision. However, achieving a reliable system capable of recognizing these attributes in real- world and real-time scenarios presents significant challenges through only the face. Factors such as variable lighting, contrast, dynamic head poses, partial occlusion, and the inherent ambiguity of facial expressions, commonly referred to as ’in-the-wild’ conditions, drastically affect the performance of models compared to results obtained in controlled laboratory environments [8].

Moreover, social robots often operate on hardware (HW) that present significant constraints. This HW typically consists of embedded platforms with limited computational resources (e.g., Raspberry PI boards, ARM-based boards or Arduino boards) [9]. As a result, recognition models must be not only accurate but also computationally efficient in order to work within these resource constraints and meet the stringent requirements of real-time systems.

These real-time requirements are directly influenced by the capture speed of the robot’s camera. Social robots may use cameras that capture images at varying frame rates, typically between 15 and 60 frames per second (FPS) [10]. This imposes strict limits on the inference time available to process each frame. For example, with a 60 FPS camera, the total time budget each frame is merely 16.6 milliseconds (ms), however at 15 FPS, it jumps to 66 ms. Recognition models must operate within these time constraints to ensure smooth and continuous interaction.

Although an alternative solution could be to run recognition models on external servers (cloud computing) while delegating only local control tasks to the robot’s onboard hardware, this approach introduces significant and unpredictable issues related to network latency and dependency on Internet connectivity. This latency often exceeds acceptable time thresholds, even at lower frame rates such as 15 FPS [11]. In addition to latency concerns, processing sensitive data such as facial images of individuals (including children and adults) on external servers raises serious privacy and security issues [12]. These factors reinforce the need to develop efficient solutions that can run locally on the robot’s hardware (edge computing).

This thesis tackles the issues of improving human-robot interaction in social robots by creating and optimizing AI models for emotion, age, and gender recognition. The experimental platform used is the InMoov robot, an open-source, 3D-printed humanoid constructed by Gael Langevin [13], renowned for its accessibility and modular design.

In this arrangement, the NVIDIA Jetson Xavier NX, an embedded edge-AI computing device, seamlessly integrates into the robot's framework to power its perceptual and decision-making capabilities. This device uses real-time visual input from a Logitech C270 HD webcam to perform recognition tasks. Additionally, a Kinect sensor is employed for complementary recognition tasks such as pose and gesture detection.

Low-level tasks, including motor control for movement and gestures, are managed by auxiliary microcontrollers such as Raspberry Pi or Arduino, which communicate with the Jetson Xavier. A high-level overview of this distributed system architecture is presented in Figure 0.1. The primary goal of this thesis is to build, construct, and evaluate AI models that run on the Jetson platform, trying to strike a compromise between identification accuracy and the efficiency required for seamless, real-time HRI.

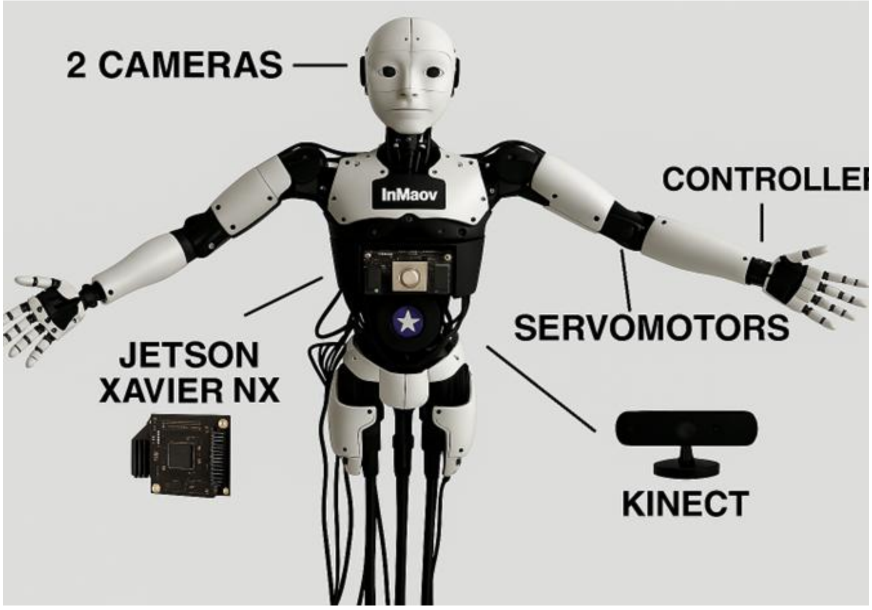


Fig. 0.1. InMoov architecture.

This study intends to improve the perceptive abilities of social robots. It focuses on conducting a comparative study of recognition models, developing novel architecture, and implementing optimization for emotion, age, and gender recognition, ensuring their efficiency for embedded systems like the Jetson Xavier NX. A key part of the methodology involves evaluating the optimized system through experiments simulating real-world interaction scenarios. This practical validation provides a realistic assessment of performance and helps identify current limitations and specific challenges encountered. An analysis of these findings informs a discussion on the system's shortcomings and outlines potential avenues for future refinement. This comprehensive process: encompassing design, optimization, real-world testing, and analysis, contributes towards the main goal of creating robots capable of more engaging, supportive, and personalized experiences in applications ranging from personal assistance to therapy and education. For this reason, this thesis aims to address this gap by: (1) conducting a comparative evaluation of selected standard, lightweight, and custom-designed deep learning architectures for emotion, age and gender recognition; (2) optimizing these models using TensorRT FP16 quantization; (3) rigorously evaluating their performance on the target Jetson Xavier NX platform; and (4) assessing the system's effectiveness in a simulated real-time interaction experiment.

1. Literature Review

1.1. Human robot interaction

HRI is an emerging field of study that analyzes complex relationships between people and machines, expanding its scope beyond physical manipulation of the environment to the realm of social interaction [14]. In this context, the concept of social robots is introduced, which is an important field in current robotics. Although scientific literature has not reached a definitive consensus on its definition [15], it is widely assumed that a social robot is intended to interact with humans in an intuitive and natural manner [3]. These robots are increasingly found in environments such as museums, receptions, homes, educational centers, and medical facilities [16]. The key characteristics that distinguish a social robot from a conventional robot include:

- Bidirectional interaction: Social robots must be able to respond to humans in a socially appropriate way, acting as companions rather than as mere tools [3]. This includes the ability to communicate, answer inquiries, and convey emotions. The interaction should be a dynamic and engaging dialogue, going beyond simple pre-programmed exchanges [17].
- Embodiment: According to Sarrica et al., physical presence is essential for a social robot [15]. Its form, whether humanoid, animal, caricatured, or functional, has a huge impact on how people perceive it and the quality of interaction [17]. The robot's appearance sets expectations about its social capabilities and the type of possible interaction [14].
- Emotions: Social robots must be able to communicate and recognize emotions using gestures, facial expressions, and words. This helps humans understand the robot's intention and its interaction [14]. Furthermore, artificial emotions can serve as feedback for the user, indicating the robot's internal state and its intentions [17].
- Autonomy: A social robot requires a degree of autonomy to make decisions and act independently [18]. This capability enables the robot to actively participate in social interactions, rather than merely reacting passively to commands [3].

While all these characteristics are fundamental to HRI, this thesis focuses specifically on how enhancing the robot's perceptual abilities, particularly its ability of the system to distinguish human emotions, age, and gender can considerably increase the quality and effectiveness of bidirectional social interactions. Recognizing human emotions is particularly important for creating more intuitive and responsive HRI systems [4]. By enabling robots to interpret and appropriately respond to human affective states, incorporating context-specific emotional cues, the quality of these exchanges can be significantly enhanced [19, 16]. Furthermore, recognizing the user's age and gender adds another layer of personalization, allowing interactions to be more tailored and potentially more effective [20]. These perceptual capabilities are critical preconditions for truly adaptive bidirectional interactions, in which the robot dynamically alters its behavior depending on its real-time comprehension of the user's status and features.

1.2. Emotion Recognition

Recognizing human emotions is a complex task that has been approached from two main perspectives: the discrete and the multidimensional approaches [21]. The theory of discrete emotions holds that emotions are clearly defined categories, each characterized by specific cognitive, psychological, and behavioral factors [22]. According to this theory, six primary emotions are widely recognized: happiness, sadness, anger, surprise, fear, and disgust [23]. Other categories, such as anticipation and trust, have now been added to this framework [22].

In contrast, the multidimensional approach considers emotions as complex phenomena influenced by personal experiences and cultural contexts. According to this theory, emotions are shown in two- or three-dimensional spaces with dimensions including dominance (perceived control), arousal (emotional intensity), and valence (degree of positive or negativity) [24]. This method allows for a more nuanced understanding of human emotions, emphasizing their dynamic and multidimensional nature.

To recognize human emotions, various techniques have been developed, including questionnaires, physical signals, and physiological signals.

Questionnaires are assessment tools that use affective scales to undertake periodic measures, usually monthly, and collect self-reports on emotions experienced [22].

Physical signals include observable aspects such as facial expressions, speech, text, gestures, and body postures. Among these, speech and facial expressions are the most commonly used methods for identifying emotions due to their accessibility and effectiveness [25].

Alternatively, physiological signals provide the advantage of being activated involuntarily, making them less susceptible to conscious control by the subject. This allows for more authentic emotion detection. Commonly used methods in this field include electrocardiograms, electroencephalograms, galvanic skin responses, respiration measurement, body temperature, and eye tracking [25].

The project utilizes a social robot for emotion recognition, and the identification methods are constrained by the capabilities of the available hardware. Cameras are among the most common sensors in social robots [3], detection based on physical signals, particularly facial expressions, emerges as the most viable and relevant approach. Consequently, this work focuses specifically on recognizing emotions through facial expression analysis.

Facial expression recognition (FER) focuses on identifying emotions by analyzing facial expressions, typically aligned with the discrete model of basic emotions [22]. Techniques in this field range from traditional computer vision algorithms to advanced AI models. In scientific research, AI models routinely attain the greatest accuracy rates among these. Notably, Convolutional Neural Networks (CNNs), which are a type of deep learning architecture designed to capture features specific to images and thus particularly effective for tasks centered around visual data [26], have proven particularly effective for FER due to their powerful representational capabilities [27].

As is well known, training AI models requires high-quality datasets. In the case of emotion recognition, available datasets are broadly divided into two main categories: in-the-wild datasets and controlled datasets [27]. This distinction is crucial when selecting a dataset for real-world FER applications, such as those involving social robots.

Controlled datasets, commonly used in laboratory settings, are often poorly suited for real-world applications. Consequently, the focus shifts to in-the-wild datasets, which better capture the variability of real-life scenarios. However, these datasets present considerable issues. Some, such as AffectNet, are not freely available, but others, especially video-based, need large storage space. Furthermore, even state-of-the-art models achieve only about 70% accuracy on these datasets [27].

Considering the need to balance realism with practical factors such as accessibility, size, and format, the FER2013 dataset was chosen for this study [28]. Although larger and more diverse ’in- the-wild’ datasets like AffectNet, RAF-DB, and ExpW exist (see Table 1.1), FER2013 remains a widely adopted benchmark due to its public availability and compact structure, comprising over 30,000 48×48 grayscale images. Despite its known limitations, it continues to serve as a common baseline for emotion recognition research.

Table 1.1: Facial emotion recognition datasets.

Dataset Name	Type	Scenario	Number of Images/Videos	Emotions
JAFFE [29]	Static Images	Controlled	213	6 basic + neutral
FER2013 [28]	Static Images	In-the-wild	35,887	6 basic + neutral
Extended Cohn-Kanade (CK+) [30]	Images/Videos	Controlled	920	6 basic + neutral+ contempt
AffectNet [31]	Static Images	In-the-wild	∼1 million	6 basic, valence, and arousal
RAF-DB [32]	Static Images	In-the-wild	29,672	Basic and compound
MultiPIE [33]	Static Images	Controlled	755,370	Various
CAER [34]	Videos	In-the-wild	13,201	6 basic
DFEW [35]	Videos	In-the-wild	12,059	6 basic
FERV39k [36]	Videos	In-the-wild	38,935	6 basic
MAFW [37]	Videos	In-the-wild	10,045	6 basic + 4 compound
ExpW [38]	Static Images	In-the-wild	91,793	6 basic
Oulu-CASIA [39]	Videos	Controlled	2,880	6 basic
Bosphorus [40]	Static Images	Controlled	4,652 3D	Various

The FER2013 dataset, whose distribution breakdown is shown in Figure 1.1, presents a slight imbalance in the number of images for each emotion, which could influence the performance of models trained on it. In particular, the emotions of disgust and happy are the most imbalanced, with only 547 images for disgust compared to 8,989 images for happy. This difference in the number of samples may affect the model’s ability to learn balanced representations of all emotions.
Furthermore, Figure 1.2 depicts exemplary instances of the photos connected with the emotions in the dataset, demonstrating the variety of face expressions utilized to represent each emotion.

After selecting the dataset to be used, it is essential to identify the deep learning models employed for this task. However, studies that rely exclusively on the FER2013 dataset report limited performance, with accuracy reaching only 69% [41]. As a result, studies investigating ways that integrate FER2013 with other methodologies and datasets to improve accuracy were examined.

Among these, the work of Zang et al. [42] stands out, as they achieve an average accuracy of 88.56% by combining the FER2013 dataset with the LFW dataset. Yet, the procedure for developing this new dataset is not specified. Taking a different approach, Chowdary et al. [43] apply transfer learning techniques on the CK+ dataset, utilizing pretrained architectures such as VGG19, ResNet50, InceptionV3, and MobileNet, achieving accuracies of 96%, 97.7%, 94.2%, and 98.5%, respectively. These investigations propose interesting solutions, including dataset fusion [42] and transfer learning [43]. However, the approach of combining datasets often lacks sufficient reproducibility details, limiting its practical application. On the controlled CK+ dataset, the transfer learning approach shows excellent accuracy; however, its performance on edge devices or in "in-the-wild" situations is yet unknown and might not translate well to real-world situations.

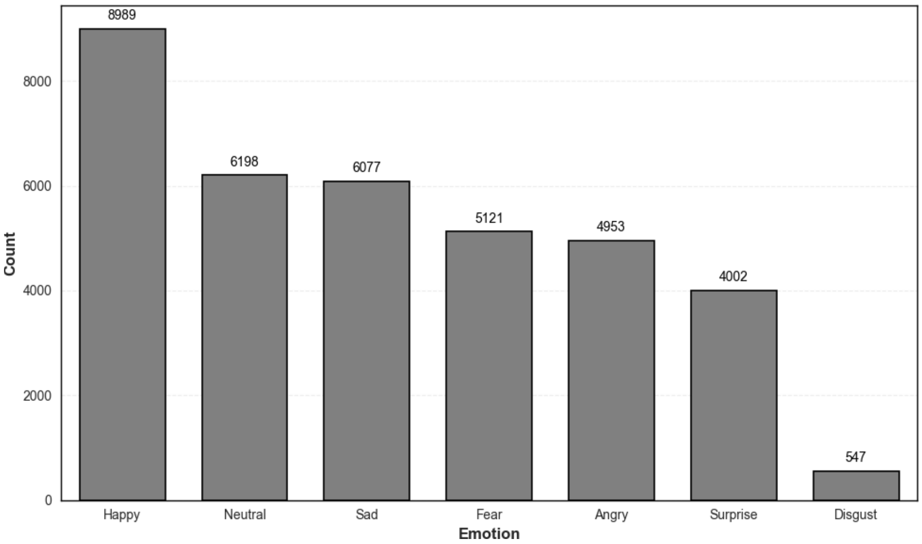


Fig. 1.1. FER2013 emotion distribution.

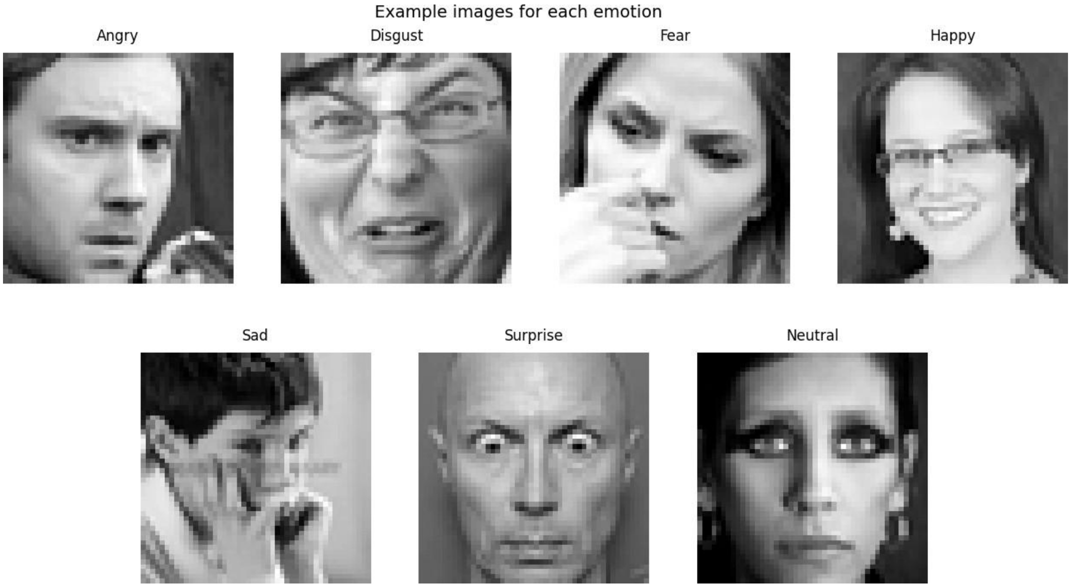


Fig. 1.2: FER2013 example images.

Similarly, Pascual et al. [44] trained a model based on transfer learning using the Xception architecture, achieving an accuracy of 69.87% on FER2013. Additionally, they applied this model on a Jetson Nano, which makes their methodology extremely pertinent to the goals of this thesis. Importantly, they also optimized the model for edge execution, achieving 6 FPS. While this frame rate may be insufficient for some real-time applications, their research represents a significant step toward putting FER models on embedded devices.

On the other hand, Seringel et al. [2] proposes an open-source framework with an emotion model achieving an accuracy of 57.42% on FER2013. In contrast the accuracy is minimal, this model is specifically built for edge devices, and its open-source nature makes it a helpful reference point for comparison. Finally, Priyadarshini V et al, [1] develops her own light CNN architecture trained on the CK+ dataset, achieving an accuracy of 97.79%. This highlights the potential of custom lightweight designs, although its performance on ’in-the-wild’ data or edge hardware remains unevaluated in their study.

The reviewed studies, along with broader surveys [27], reveal a recurring pattern: many high performing FER models are either evaluated primarily in controlled laboratory settings (e.g., using datasets like CK+) or consist of large, computationally intensive architectures designed for maximum accuracy, often neglecting deployment constraints. While some research, like that of Pascual et al. [44] and Seringel et al. [2], explicitly targets edge devices, a clear gap remains. Specifically, there is a scarcity of comprehensive comparative research analyzing the trade-offs between accuracy and real-time inference performance for various FER architectural methods, particularly on competent edge AI platforms like the NVIDIA Jetson Xavier NX. Achieving this optimal balance is critical for effective HRI [45], yet remains an underexplored challenge addressed by this thesis.

To close this gap, this thesis conducts a comparative study of many unique architectural techniques for FER on the Jetson Xavier NX platform. Based on the literature analysis, the following approaches were chosen for inquiry:

- **A Baseline Edge-Oriented Model:** The open-source framework by Seringel et al. [2], despite its modest reported accuracy on FER2013, serves as a relevant baseline due to its specific design for edge implementation.
- **Standard High-Performance CNNs via Transfer Learning:** Architectures such as ResNet50 [46], which uses residual connections to train much deeper networks, and InceptionV3 [47], which has Inception modules that perform convolutions at multiple scales concurrently, are widely used models known for achieving a strong balance between accuracy and computational cost on standard benchmarks such as ImageNet [48]. Although not originally designed for extreme efficiency, they are less demanding than previous designs (e.g., VGG) and are widely employed as effective backbones for transfer learning in a variety of visual tasks [43]. However, their baseline performance and potential for optimization specifically for real-time FER on the target Jetson Xavier NX platform remain to be quantified, justifying their inclusion in this comparative study.
- **A Custom Lightweight CNN:** The architecture proposed by Priyadarshini V et al. [1], which achieved high accuracy on CK+, represents the potential of custom-designed lightweight models, although its performance on ’in-the-wild’ data and edge hardware needs assessment.
- **Custom Hybrid Architecture:** To explore contemporary approaches, a novel architecture combining CNN blocks, for local feature extraction, with mechanisms inspired by Transformers, specifically self-attention layers [49], was designed. This helps to analyze if introducing layers capable of modeling long-range interdependence and global context improves FER within the restrictions of edge deployment.

The detailed implementation of these selected architectures, along with the methodology employed for their training, optimization, and comparative evaluation, is presented in the Methodology section.

1.3. Age and Gender Recognition from Facial Images

Beyond emotion detection, determining a person's age and gender from facial photos are two critical perceptual abilities for improving HRI. The task presents significant challenges, particularly in age estimation, which is commonly addressed using two primary approaches: continuous regression and classification by ranges [50].

In the regression approach, the goal is to predict the exact age by treating it as a continuous variable. This method allows for precise evaluations using metrics such as the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), making it especially useful when an exact prediction is required. Furthermore, it can handle a variety of data distributions with more flexibility because it does not require predetermined age ranges. Common techniques used for this approach include CNN and Support Vector Machines (SVM) [51]. However, regression can be more sensitive to noisy data, such as images with poor lighting or complex facial expressions.

In contrast, the classification approach organizes ages into predefined intervals or categories. This method is effective when an exact age is not needed, but rather a general age group. Additionally, it is more resilient to noisy or imprecise data, which makes it ideal for real-world uses like limiting access to content that is restricted to certain age groups. Algorithms such as K-Nearest Neighbors (KNN), SVM, and CNN are commonly used for this task [52].

Moreover, hybrid approaches have emerged that combine the strengths of both regression and classification. For instance, a regression model can be used inside a certain age range that has been identified using classification models to predict an age with greater precision. An additional approach involves framing classification as a multiclass problem, where each category corresponds to a particular age group. These hybrid models offer greater flexibility and adaptability, enabling them to perform effectively across a variety of different scenarios and requirements [51].

Choosing the most suitable approach depends on several factors, such as the quality of available data, the specific needs of the application, and whether an exact or categorical prediction is desired. External factors such as lighting, facial posture, and distinct individual aging processes can all have an impact on model performance and should be carefully studied [52].

The recommended strategy for a social robot is categorization, as establishing a conversation does not require a specified age. Instead, an age range is adequate to encourage communication and interaction.

On the other hand, gender estimation is commonly approached as a binary classification problem, which simplifies the task. The goal is to assign a person to one of two categories: male or female [50], although it is important to acknowledge that this binary classification based on visual appearance may not capture the full spectrum of gender identity. This approach is the most widely used due to the availability of data [51]. To carry out this classification, various machine learning methods are employed, among which SVM, CNN, and Linear Discriminant Analysis (LDA) stand out [50]. Furthermore, feature extraction from facial pictures employing techniques such as Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), and Scale-Invariant Feature Transform (SIFT) supplements existing classification approaches and improves their accuracy [50].

Age and gender recognition research rely largely on different datasets, which frequently include labels for both traits. These resources vary significantly in scale, annotation type, capture conditions, and demographic balance. For instance, IMDB-WIKI [50] offers an immense scale with over 500,000 images, but it suffers from known label inaccuracies, limiting its reliability for rigorous training. On the other hand, datasets captured under uncontrolled, "in-the-wild" conditions, such as Adience [52], which provides age group labels, and UTKFace [53], which offers precise age and ethnicity labels, add valuable diversity to the dataset. Another example is MORPH [50], which is widely utilized in academic research due to its high-quality photographs collected under highly regulated conditions. More recently, databases such as FairFace [53] have gained popularity. FairFace focuses on addressing racial bias through fair demographic representation, gives age and gender labels for a large volume of photographs (over 100,000), and, most importantly, captures these images "in-the-wild". Table 1.2 presents a comprehensive overview comparing these and other relevant datasets, detailing their specific characteristics. Understanding these distinctions is crucial for selecting appropriate datasets for training and evaluation based on application requirements.

Table 1.2: Public datasets for facial age and gender recognition

Dataset Name	Scenario	Images	Age	Gender
IMDB-WIKI [54]	In-the-wild	523,051	Exact age	Binary (male/female)
FairFace [53]	In-the-wild	108,501	Group age	Binary (male/female)
MORPH [55]	Controlled	55,134	Exact age	Binary (male/female)
Adience [56]	In-the-wild	26,528	Group age	Binary (male/female)
UTKFace [57]	In-the-wild	20,000	Exact age	Binary (male/female)
AgeDB [58]	In-the-wild	16,488	Exact age	Binary (male/female)
MSU-LFW+ [59]	In-the-wild	15,699	Exact age	Binary (male/female)

Note: The term "binary (male/female)" refers to the gender labels available in these datasets. Other gender identities are not represented.

Regarding age recognition, some of the best results in literature are achieved with Adience, where an accuracy of 91.8% is reported using AlexNet, and with UTKFace, where 71.84% accuracy is achieved using ResNet50 [50]. For gender recognition, MUS-LFW+ achieved 97.31% accuracy with a CNN based model, while a ResNet model achieved 96.26% [50].

For this particular application, which involves a social robot interacting in potentially various real-world circumstances, using datasets recorded in uncontrolled contexts is strongly recommended to assure the model's relevance and resilience. This requirement significantly narrows the choices among the available datasets (Table 1.2). Furthermore, the study by Karkkainen and Joo [53] compellingly highlights the critical importance of ethnic diversity in training data, not only because mitigates potential biases, a crucial consideration for social robots, but also to enhance overall model robustness and generalization. Their study produced remarkable results with their suggested dataset, prompting the use of FairFace as the principal dataset for the age and gender recognition tasks in this thesis. As a reference, the original study reported accuracies of 59.7% for age (9-class classification) and 94.2% for gender using a ResNet34 backbone trained on FairFace [53].

FairFace [53] contains 108,501 photos derived from the YFCC-100M dataset, expertly chosen and balanced among seven ethnicity groups (White, Black, Indian, East Asian, Southeast Asian, Middle Eastern, Latino). It provides labels for gender (Male, Female) and age, categorized into 9 groups: 0-2, 3-9, 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, and 70+. The dataset is officially divided into training (86,744 images) and validation (21,757 images) sets. The age and gender distributions within the dataset are illustrated in Figure 1.3 and Figure 1.4, respectively. As observed in the age distribution (Figure 1.3), there is a significant imbalance across age categories. Specifically, classes such as ’70+’ (with only 842 images) and ’0-2’ (with 1,792 images) are heavily underrepresented compared to more populated groups like ’20-29’ (with 25,598 images). This severe imbalance may have an influence on the model's capacity to develop robust representations for all age groups, as well as overall age classification accuracy. In contrast, the gender distribution (Figure 1.4) is relatively well-balanced, with only approximately 5,000 more images labeled as female than male.

The images in Figure 1.5 demonstrate the dataset's diversity and the 'in-the-wild' conditions under which they were taken.

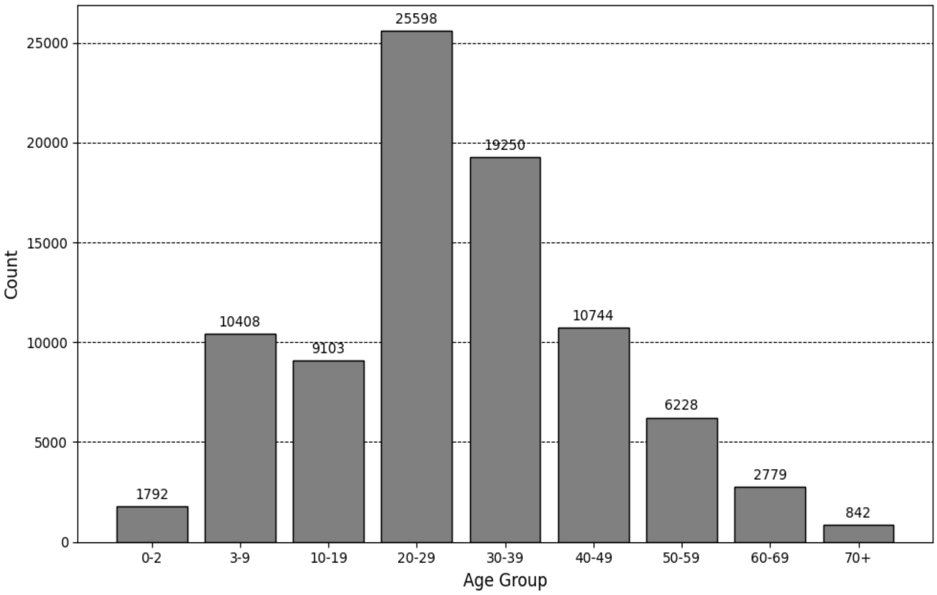


Fig. 1.3: Fair Face age distribution (Training data).

Concurrently with the emotion recognition task, and building on insights gained from that initial investigation, this thesis evaluates several architectural approaches for age and gender classification using the FairFace dataset on the Jetson Xavier NX. The goal is to assess the trade-offs between accuracy and efficiency, and to determine whether architectures perform consistently across different facial attribute recognition tasks. The selected models represent various points along the complexity-efficiency spectrum, including:

- **MobileNetV2 [60]:** Selected as a representative lightweight architecture, it is well known for its effectiveness on edge and mobile devices using methods like linear bottlenecks and inverted residuals. Its performance on FairFace serves as an important benchmark for highly optimized models designed for resource-constrained contexts such as the intended Jetson platform.
- **ResNet50 [46]:** Included as a standard, high-performance backbone, leveraging residual connections to train deeper networks effectively. Its inclusion matches with architectures utilized in relevant age and gender research [50, 53] and provides as a comparison point between the lightweight and custom models, assessing the performance achievable with a more complex, but potentially more accurate, model before optimization.

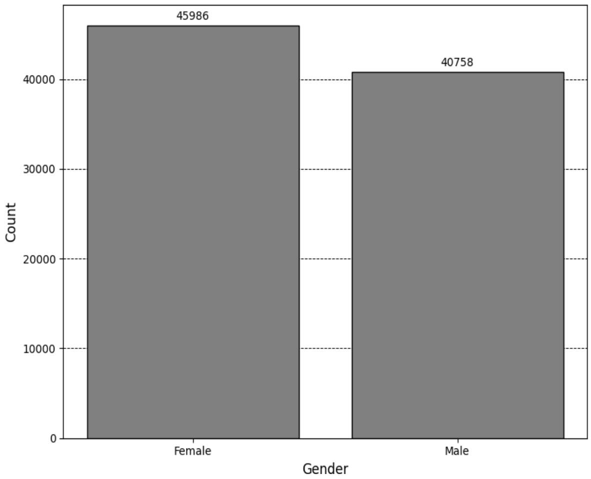


Fig. 1.4: Fair Face gender distribution (Training data).

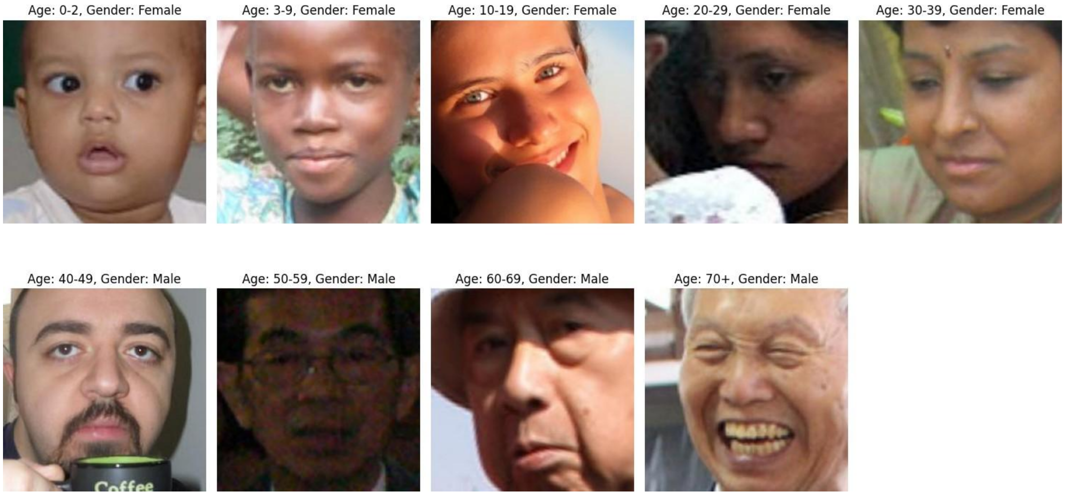


Fig. 1.5: Fair Face example images.

- **The Custom Hybrid Architecture:** The custom CNN-Attention architecture developed primarily for emotion recognition is also evaluated on age and gender tasks. This cross-task evaluation examines the architecture's flexibility and appropriateness for analyzing multiple facial attributes simultaneously. It assesses how well the model can generalize across various prediction tasks while maintaining a focus on balancing high accuracy with computational efficiency, particularly for deployment at the edge.

The methodology section provides a detailed approach for customizing these structures, as well as specifics on training, optimization, and comparative evaluation.

1.4. Nvidia Jetson Xavier

The deployment of real-time AI models for social robotics requires sufficient edge computing hardware. This thesis utilizes the NVIDIA Jetson Xavier NX platform [61], a compact yet powerful system-on-module (SOM) readily available for this research and specifically designed to deliver significant AI performance at the edge. It has a processing capacity of 21 trillion operations per second (TOPS) for INT8 accuracy, allowing for the parallel execution of many neural networks as well as the processing of input from high-resolution sensors used in robotics. Key components include an integrated NVIDIA Volta architecture GPU, featuring 384 CUDA cores and 48 Tensor Cores optimized for deep learning tasks, alongside a 6-core 64-bit ARM Carmel CPU for general purpose processing. The module is equipped with 8 GB of LPDDR4x RAM and 16 GB of eMMC storage [61].

Beyond the core CPU and GPU, the Jetson Xavier NX architecture incorporates several features pertinent to robotics and AI applications. It incorporates a high-throughput 128-bit LPDDR4x memory subsystem, which is critical for data-intensive activities. For handling sensor data, especially video streams, the module offers dedicated high-definition video processing capabilities, including multi standard hardware decoders and encoders. Furthermore, its rich set of communication interfaces (including PCIe Gen3/4, USB 3.1/2.0, Gigabit Ethernet, SPI, I2C, UART, and CAN) facilitates integration with various sensors, actuators, and network infrastructure [61].

The Jetson Xavier NX includes dedicated hardware accelerators known as Deep Learning Accelerators (DLAs), which are specifically designed to improve AI inference efficiency. These engines are optimized for common neural network operations and can offload tasks from the main GPU, potentially improving overall throughput or reducing power consumption. The platform also provides advanced power management features, including configurable power modes allowing developers to balance performance and energy efficiency based on application needs. Standard General-Purpose Input/Output (GPIO) pins are available for direct interfacing with low-level hardware components [61].

Although the Jetson Xavier NX has impressive processing capacity for an edge device, installing large deep learning models for real-time inference frequently necessitates further optimization beyond conventional framework execution. To address this, NVIDIA provides TensorRT, a high-performance deep learning inference optimizer and runtime library [62]. TensorRT integrates seamlessly with models trained in popular frameworks (like TensorFlow or PyTorch) and makes several optimizations for NVIDIA GPUs, particularly the Volta/Turing architecture found in the Xavier NX. This framework uses several optimization approaches, including layer and tensor fusion, precision calibration (which allows for optimized INT8 and FP16 quantization), kernel autotuning (which selects the fastest implementations for the target GPU), and dynamic tensor memory management [63]. The major purpose of these optimizations is to reduce inference delay while increasing throughput on the target device.

Several research have shown that TensorRT is effective at accelerating inference. For example, Zuo and Yang [63] showed significant speedups (1.6x to 2x) for efficient networks like MobileNet and SqueezeNet when using TensorRT compared to unoptimized framework execution. Using optimization frameworks like TensorRT is not only advantageous but maybe necessary for this thesis' emotion, age, and gender detection tasks, which must be completed in real-time on the social robot's Jetson Xavier NX. Therefore, TensorRT will be employed to optimize the selected models to meet the stringent real-time performance requirements. The specific optimization techniques applied are detailed further in Section 1.5.

1.5. Model Optimization Techniques

Achieving real-time performance for deep learning models on resource-constrained edge devices like the Jetson Xavier NX often requires specific optimization techniques. These strategies seek to minimize the model's computational complexity, memory footprint, and storage capacity, lowering inference latency and power consumption while preserving job accuracy. Common techniques include quantization and pruning [64, 65]. Pruning involves removing redundant model parameters but often requires complex retraining cycles. Given the significant performance gains observed preliminary with the direct optimizations offered by the target deployment framework (NVIDIA TensorRT), and considering the project scope, this work focused primarily on quantization.

1.5.1. Quantization

Quantization is the process of reducing the numerical precision used to represent values within a neural network, typically weights and activations [65]. Deep learning models are usually trained using 32-bit floating-point numbers (FP32), which offer a wide dynamic range and high precision. Quantization maps these FP32 values to lower-precision data formats like 16-bit floating-point (FP16), 8-bit integers (INT8), and even less bit widths.

The most common approach to quantization is uniform quantization, which maps real-valued floating-point numbers into a lower-precision numerical range [65]. A commonly used form of the quantization function is defined as follows:

$$Q(r) = Int\left(\frac{r}{S}\right) - Z \tag{1.1}$$

Where Q represents the quantization operator, r is a real-valued input (such as an activation or a weight), S is a real-valued scaling factor, and Z is an integer zero-point. The function Int maps a real number to an integer via a rounding operation [65].

Conversely, it is possible to recover an approximation of the original value r from the quantized values $Q(r)$ through the dequantization operation:

$$\bar{r} = S \cdot (Q(r) + Z) \tag{1.2}$$

Note that $\bar{r}$ does not exactly match the original value r due to the rounding applied during quantization.

The scaling factor S and zero-point Z are critical factors that are determined using the range of the original FP32 values. This determination can happen per-tensor or per-channel, and symmetrically or asymmetrically around zero [65].

Quantization yields several benefits:

- **Reduced Memory Footprint**: The memory needed to hold intermediate activations and model weights is decreased with lower-precision representations.
- **Faster Computation:** Hardware designed for low-precision arithmetic, such as dedicated accelerators like DLAs and Tensor Cores in NVIDIA GPUs, can perform these operations far more effectively [63].
- **Lower Power Consumption:** Faster computation and reduced memory access generally lead to lower energy usage.

Two main strategies exist for applying quantization:

- **Post-Training Quantization (PTQ):** The trained FP32 model is then quantized. This is simpler to implement, but it may result in accuracy reduction, particularly when quantizing to very low precision, such as INT8. A representative calibration dataset is often required by PTQ to estimate suitable activation ranges and establish the corresponding quantization parameters, such as scale (S) and zero-point (Z) [65].
- **Quantization-Aware Training (QAT):** Quantization effects (the clamping and rounding) are simulated during the model training or fine-tuning process. This allows the model to adapt to the reduced precision, often resulting in better accuracy preservation compared to PTQ, albeit at the cost of a more complex training procedure [65].

Both FP16 and INT8 precision conversion are supported by TensorRT, the optimization framework used in this investigation. A calibration step is usually necessary when applying INT8 precision utilizing PTQ capabilities. This involves providing a representative dataset to allow TensorRT to analyze the distribution of activation values and determine the optimal scaling factors to minimize accuracy loss during the FP32- to-INT8 conversion [66]. Considering the additional intricacy and data needs related to this calibration procedure, and the goal of streamlining the optimization workflow, FP16 quantization was selected as the primary optimization strategy for this thesis. FP16 conversion in TensorRT generally does not require a separate calibration dataset, simplifying the deployment pipeline while still offering significant performance benefits on compatible hardware.

1.5.2. Optimization workflow with TensorRT

The NVIDIA TensorRT framework, introduced earlier on section 1.4, provides a practical work- flow to apply several optimization techniques, including quantization, targeting NVIDIA hardware like the Jetson Xavier NX. The typical process involves:

1. **Model Conversion:** A pre-trained model from a framework like as TensorFlow or PyTorch is usually translated to an intermediate format, most notably Open Neural Network Exchange (ONNX).
2. **TensorRT Engine Building:** The TensorRT builder applies several graph optimizations to the ONNX model or straight from some frameworks. This comprises:
   - **Layer Fusion:** Combining multiple layers into a single optimized kernel to reduce overhead.
   - **Precision Optimization:** Enabling lower precision like FP16 or INT8. TensorRT must carry out a calibration step in which it uses a representative dataset (calibration set) to observe the distribution of activation values to determine the optimal scaling factors for INT8 PTQ while minimizing information loss [66]. It can perform quantization per-tensor or per-channel.
   - **Kernel Auto-Tuning:** Deciding which CUDA kernel implementation on the particular target GPU is the most effective for each operation.
   - **Memory Optimization:** Optimizing memory allocation and reuse.

3. **Engine Serialization:** The optimized execution plan is serialized into a deployable file called a TensorRT engine.
4. **Inference Deployment:** The TensorRT runtime loads this engine and executes inference efficiently on the target NVIDIA hardware.

This method dramatically streamlines the deployment of high-performance deep learning inference, allowing developers to take advantage of hardware optimizations without manually implementing low-level CUDA kernels [66].

1.6. Synthesis and research gap

In summary, this chapter has reviewed the state of the art in facial emotion, age, and gender recognition, highlighting the importance of these tasks for human-robot interaction. Common deep learning approaches, benchmark datasets, and the unique problems of deploying these models on resource-constrained edge devices such as the NVIDIA Jetson Xavier NX. Additionally, the need for model optimization was covered, including the TensorRT framework's capabilities and methods like quantization.

Despite notable advances, there is still a lack of practical, comparative evaluations that assess the trade-offs between accuracy, inference speed, and model size for facial attribute tasks such as emotion, age, and gender, when optimized for edge devices like the Jetson Xavier NX.

2. Methodology

This section describes the methodology used to create, test, and optimize a real-time facial attribute detection system for the social robot platform. Building on the literature review in Section 1, it outlines the datasets used for training and evaluation, as well as the preprocessing pipeline, which includes face detection, alignment, normalization, and data augmentation techniques.

Section 2.3 provides an overview of the deep learning architectures chosen for recognizing emotions, age, and gender. This includes both models that have been pre-trained and subsequently fine-tuned through transfer learning, and a custom-designed hybrid approach. Details regarding the training setup, such as the software frameworks employed, optimizers, loss functions, and hyperparameter configurations, are thoroughly described in Section 2.4.

Model optimization with NVIDIA TensorRT, focusing on FP16 quantization, is discussed in Section 2.5. Section 2.6 defines the performance metrics used to assess both predictive accuracy and computational efficiency.

Section 2.7 describes the design and implementation of a real-time experiment with human volunteers to evaluate the system's performance under realistic conditions, including variations in head pose and emotional expression. Finally, Section 2.8 outlines the benchmarking protocol used to assess the system both before and after model optimization, as well as the experiment, using standard datasets and the Jetson Xavier NX.

This methodological framework supports the results discussed in Section 3.

2.1. Datasets

This section describes the datasets used for training, validation, and generalization evaluation.

2.1.1. Training and validation datasets

Two primary datasets were used:

- **FER2013 (Emotions):** The emotion recognition models were trained and validated using the dataset [28], as explained in Section 1. The publicly available split was utilized, with 28,709 photos for training and 3,589 for validation. FER2013 includes grayscale facial images of size 48×48 pixels, categorized into seven emotion classes.
- **FairFace (Age/Gender):** This dataset [53] was used to train and validate both the age and gender recognition models, as justified in Section 1. The publicly available split was used, consisting of 86,744 images for training and 21,757 for validation. FairFace provides color facial images at a resolution of 224×224 pixels. Gender is framed as a binary classification task, while age is divided into nine categories.

2.1.2. Generalization datasets

Two main datasets were used to evaluate the generalization of the trained models from the training datasets. These datasets also played a key role in model selection for optimization and real-world deployment in the experiment.

- **CK+ (Emotions):** This dataset [30] was selected for several reasons. Firstly, it is a widely used benchmark in FER literature, allowing for comparison of results. Secondly, evaluating on CK+ assesses the model's ability to recognize clearly posed expressions under controlled laboratory conditions. This provides a valuable performance baseline, distinct from the challenges inherent in ’in-the-wild’ data like FER2013. Furthermore, testing a model trained on noisy ’in- the-wild’ data (FER2013) on this ’clean’ dataset is a strong indicator of generalization: high performance on CK+ suggests that the model has learned robust underlying features of facial expressions, rather than simply overfitting to the training set’s artifacts. The entire dataset, consisting of grayscale images at a resolution of 48×48 pixels, was used for evaluation, excluding the "contempt" category which is not present in the 7-class FER2013 setup. Consequently, 902 images were utilized for this evaluation. Figure 2.1 illustrates a significant class imbalance, with the majority class containing 593 images, while other classes have fewer than 90 images. Specifically, the "sad" and "fear" classes contain fewer than 30 images each. This imbalance must be taken into account when performing a proper evaluation of the models.

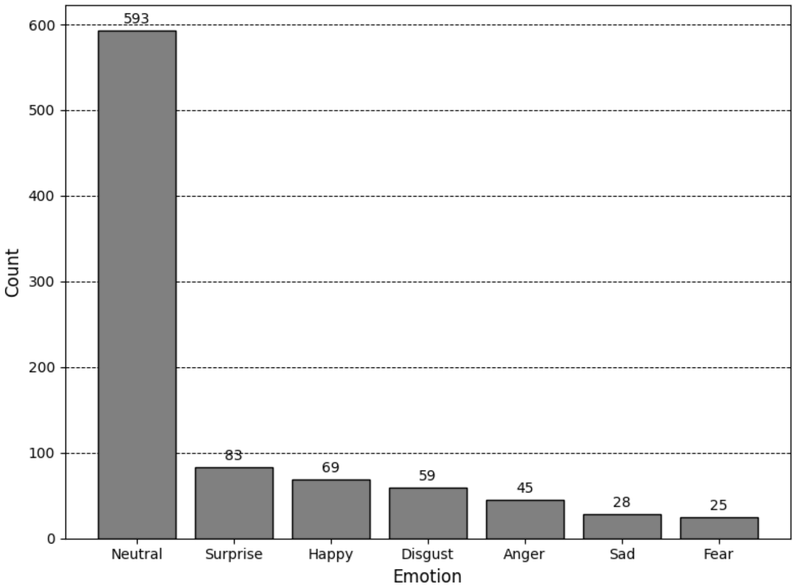


Fig. 2.1. CK emotion distribution.

– **UTK Face (Age/Gender):** This dataset [57] was chosen to evaluate generalization performance due to its ethnic diversity and substantial size, over 20,000 images captured ’in-the-wild’. The entire dataset was utilized for this evaluation. UTK Face provides facial images primarily at a resolution of 200x200 pixels, which were resized to 224x224 pixels for consistency during evaluation. Crucially, while UTK Face provides exact numerical age labels, these ages were mapped into the nine age categories used by the FairFace training dataset prior to evaluating age classification performance. The resulting age distribution, grouped into these categories (Figure 2.2), is somewhat imbalanced, particularly with peaks in the 20-29 and 30-39 age groups, although other groups are more evenly represented (around 1,000 images each). Therefore, caution is needed when interpreting the overall age classification metrics. In contrast, the gen- der distribution (Figure 2.3) is fairly well-balanced, suggesting gender metrics should provide reliable indications of model performance.

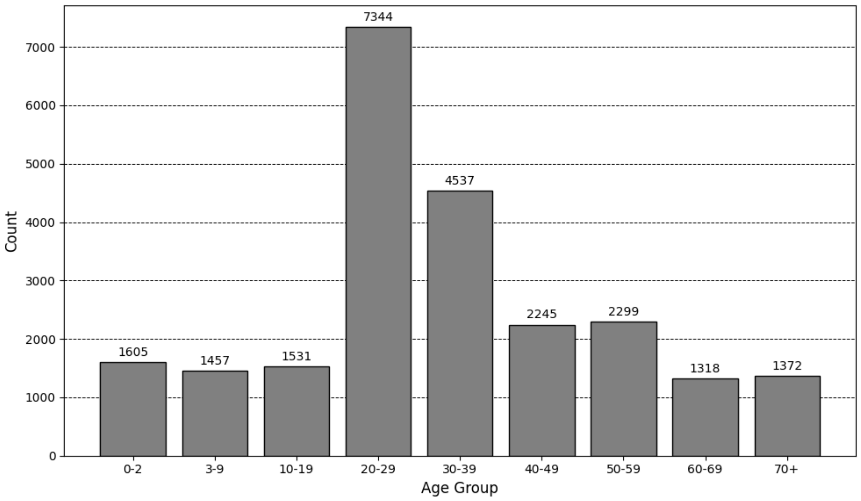


Fig. 2.2. UTK Face age distribution.

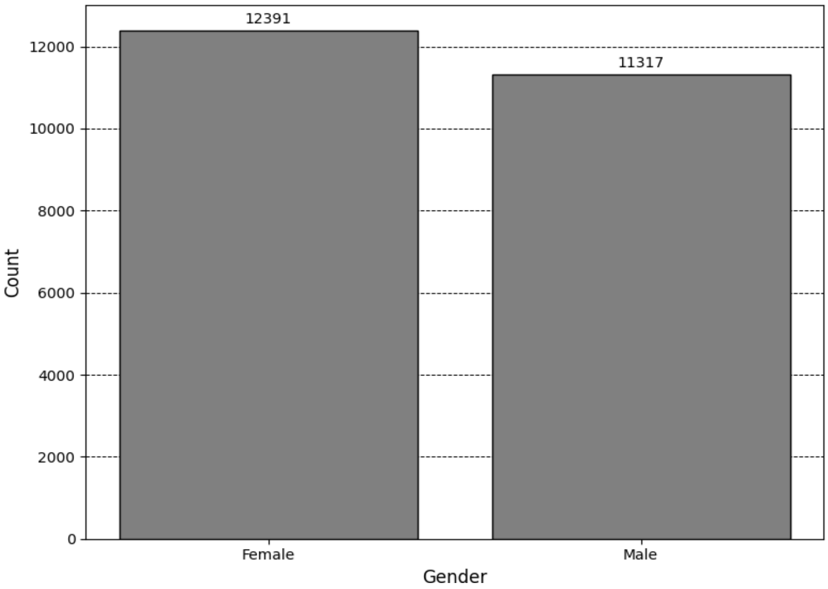


Fig. 2.3. UTK Face gender distribution.

2.2. Data processing

Prior to model training and evaluation, images from the selected datasets underwent preprocessing steps to ensure compatibility with the neural network architectures. Given that the primary training datasets (FER2013 and FairFace) provide pre-cropped facial images, no additional facial detection or alignment was applied to these standard datasets. The main preprocessing steps involved resizing and pixel normalization.

2.2.1. Image resizing

Input images were resized using bilinear interpolation to meet the specific input size requirements of each evaluated model:

- **For Emotion Recognition (FER2013/CK+ datasets):**

  –Models presented in [2] and [1] were fed the original 48x48 pixel grayscale images.

  –Models relying on pre-trained architectures, such as ResNet50 and InceptionV3, required inputs scaled to 224 x 224 pixels. Because these models require three input channels, the

single grayscale channel of the 48x48 photos was first scaled to 224x224 and then repeated across all three channels.
–The hybrid CNN-transformer were resized into 96x96 pixel grayscale images.

- **For Age/Gender Recognition (FairFace/UTKFace datasets):**
  –All evaluated architectures (MobileNetV2, ResNet50, Proposed Hybrid Model) received input images resized to 224x224 pixels, maintaining their original RGB color format (3 channels).

This resizing strategy ensured input dimensionality consistency for each model across training, validation, and generalization testing phases.

2.2.2. Pixel normalization

Following any resizing steps, pixel values for all images, regardless of the dataset or target model, were normalized to the standard floating-point range of [0, 1]. This was achieved simply by dividing each pixel intensity value by 255.0. Consistent normalization was applied during all training and evaluation stages.

2.2.3. Data augmentation (Training only)

Data augmentation was applied to improve generalization, as recommended by the literature and best practices. For the FER2013 dataset, augmentation was used not only to simulate various real-time environments, such as illumination changes, but also to address class imbalance. Each class was augmented to contain 8,000 samples, resulting in a total of 56,000 images. The augmentation process included:

- Random horizontal flipping (probability: 0.5) to simulate viewpoint asymmetry,
- Rotation (±15°) and scaling/translation (±10% scaling, ±2% translation) to mimic pose variation,
- Gamma correction ($\gamma \in [0.5, 1.5]$) to simulate illumination changes,
- Gaussian noise injection ($\sigma = 0.05$) to approximate sensor noise from real-world capture devices.

All transformations were implemented using the Albumentations library [67] in Python, ensuring both reproducibility and computational efficiency.

On the other hand, for the FairFace dataset, although there was also class imbalance in terms of age groups, the training was constrained by hardware limitations, specifically, the use of Google Colab. Due to the large size of the dataset (over 83,000 images), it was not feasible to fully balance the classes or simulate in-the-wild conditions. Therefore, only one augmented image was generated per sample using a randomly selected transformation from the set defined above for FER2013 (flipping, rotation, scaling/translation, gamma correction, or noise injection).

2.3. Model architectures

2.3.1. Model architectures for emotion recognition

Based on the literature review (Section 1) and the goal of comparing different approaches on the Jetson Xavier NX, several distinct neural network architectures were implemented and evaluated for the facial emotion recognition task using the FER2013 dataset. These included models generated from current frameworks, conventional pre-trained backbones updated through transfer learning, and new hybrid architecture. The specific structure of these models is described below.

Baseline Edge-Oriented Model: The first model evaluated was the CNN architecture proposed by Seringel et al. [2], chosen as a baseline due to its open-source availability and specific design intention for edge devices. Its layer structure is summarized in Table 2.1.

Table 2.1: CNN structure proposed in [2]

Layer	Type	Details
Conv	Convolutional	64 filters, 5x5 kernel
MaxPool	Pooling	2x2 pool
Conv	Convolutional	64 filters, 3x3 kernel
Conv	Convolutional	64 filters, 3x3 kernel
AveragePool	Pooling	3x3 pool, 2 strides
Conv	Convolutional	128 filters, 3x3 kernel
Conv	Convolutional	128 filters, 3x3 kernel
AveragePool	Pooling	3x3 pool, 2 strides
Fully Connected	Dense	1024 units
Fully Connected	Dense	1024 units
Fully Connected	Dense	7 units

Transfer Learning Models (InceptionV3 and ResNet50): To leverage knowledge learned from large-scale image datasets, standard pre-trained architectures were adapted. Specifically, InceptionV3 [47] and ResNet50 [46] backbones, pre-trained on ImageNet, were utilized. The original classification layer of each model was replaced with a custom layer suitable for the 7-class emotion recognition task. This usually entailed freezing the weights of the convolutional base and training new dense layers on top. Tables 2.2 and 2.3 provide details on the new classification head structure for InceptionV3 and ResNet50, respectively.

Table 2.2: CNN architecture with InceptionV3 as the backbone

Layer	Type	Details
InceptionV3	Pretrained model	Principal layer
Flatten	Flatten	Flatten the input
Dense	Dense	128 units, ReLU activation
Dropout	Dropout	Dropout rate of 0.5
Dense	**Dense**	**7 units with softmax activation**

Custom Lightweight CNN: The lightweight CNN architecture developed by Priyadarshini V et al. [1] was implemented to represent custom-designed models with efficiency in mind. Although originally assessed on CK+, its structure, shown in Table 2.4, was trained here on FER2013 to examine its performance in ’in-the-wild’ situations.

Table 2.3: CNN architecture with Resnet50 as the backbone

Layer	Type	Details
Resnet 50	Pretrained model	Principal layer
Flatten	Flatten	Flatten the input
Dense	Dense	128 units, ReLU activation
Dropout	Dropout	Dropout rate of 0.5
Dense	Dense	7 units with softmax activation

Custom Lightweight CNN: The lightweight CNN architecture developed by Priyadarshini V et al. [1] was implemented to represent custom-designed models with efficiency in mind. Although originally assessed on CK+, its structure, shown in Table 2.4, was trained here on FER2013 to examine its performance in ’in-the-wild’ situations.

Proposed Hybrid CNN-Transformer Architecture: Finally, a novel hybrid architecture was developed and deployed specifically for this thesis, with the purpose of integrating the local feature extraction power of CNNs alongside the global context understanding provided by attention mechanisms, drawing inspiration from Transformer models. The detailed configuration, which includes convolutional blocks and attention layers, is documented in Table 2.5.

This selection of architectures allows for a comparative analysis across different design philosophies, complexities, and performance characteristics relevant to deployment on the Jetson Xavier NX.

2.3.2. Model architectures for age recognition

For the age classification task using the FairFace dataset (predicting 9 age groups), three distinct architectures were implemented and evaluated. These models were adapted to process the [e.g., 224x224 RGB] input images from FairFace and output predictions across the nine defined age categories.

Table 2.4: CNN structure proposed in [1]

Layer	Type	Details
Input	Input	Shape: (48, 48, 1)
Conv2D	Convolution	Filters: 8, Kernel: 9x9
BatchNormalization	Normalization	-
Conv2D	Convolution	Filters: 8, Kernel: 9x9
BatchNormalization	Normalization	-
Activation	Activation	Elu
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.1
Conv2D	Convolution	Filters: 16, Kernel: 7x7
BatchNormalization	Normalization	-
Conv2D	Convolution	Filters: 16, Kernel: 7x7
BatchNormalization	Normalization	-
Activation	Activation	Elu
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.2
Conv2D	Convolution	Filters: 32, Kernel: 5x5
BatchNormalization	Normalization	-
Conv2D	Convolution	Filters: 32, Kernel: 5x5
BatchNormalization	Normalization	-
Activation	Activation	Elu
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.2
Conv2D	Convolution	Filters: 64, Kernel: 3x3
BatchNormalization	Normalization	-
Conv2D	Convolution	Filters: 64, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	Elu
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.1
Conv2D	Convolution	Filters: 128, Kernel: 3x3
BatchNormalization	Normalization	-
Conv2D	Convolution	Filters: 128, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	Elu
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.2
Flatten	Flatten	-
Dense	Dense	Units: 128, Activation: Elu

Dropout	Regularization	Rate: 0.5
Dense	Dense	7 units with softmax activation

MobileNetV2-based Model: The MobileNetV2 architecture [60], pre-trained on ImageNet, was used to estimate age. MobileNetV2 is notable for its computational efficiency, which is achieved by the use of inverted residual blocks and linear bottlenecks. The pre-trained convolutional base served as a feature extractor, followed by a bespoke classification head tailored to the 9-class age problem. Table 2.6 describes the specific structure of the adapted model.

ResNet50-based Model: Similarly, the ResNet50 architecture [46], pre-trained on ImageNet and utilizing residual connections, was adapted. The pre-trained backbone served as the feature extractor, with a custom classification head appended for the 9-class age prediction. The structure is presented in Table 2.7.

Proposed Hybrid CNN-Transformer Model for Age: The proposed hybrid CNN-Transformer architecture built for this thesis, whose complete structure is shown in Table 2.5, was also used for the age categorization problem. The adaptation involves two major changes:

- **Input Layer:** The input layer was configured to accept 224x224 RGB images (shape: (224, 224, 3)) consistent with the FairFace dataset preprocessing.
- **Output Layer:** The final dense classification layer was replaced with one containing 9 output units utilizing a softmax activation function.

Table 2.5: Sequential architecture of the proposed hybrid CNN-Transformer model for emotion recognition.

Layer	Type	Details
Input	Input	Shape: (96, 96, 1)
Conv2D	Convolution	Filters: 16, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
Conv2D	Convolution	Filters: 16, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.3
Conv2D	Convolution	Filters: 32, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
Conv2D	Convolution	Filters: 32, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.3
Conv2D	Convolution	Filters: 64, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
Conv2D	Convolution	Filters: 64, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.3
Conv2D	Convolution	Filters: 128, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
Conv2D	Convolution	Filters: 128, Kernel: 3x3
BatchNormalization	Normalization	-
Activation	Activation	ReLU
MaxPooling2D	Pooling	Pool Size: 2x2
Dropout	Regularization	Rate: 0.3
Conv2D	Convolution	Filters: 256, Kernel: 3x3
Dropout	Regularization	Rate: 0.3
Reshape	Reshape	36 × 256
Multi-Head Attention	Attention	4 heads, key dim=64

Add	Skip Connection	Element-wise addition
Flatten	Flatten	-
Dense	Dense	Units: 128
BatchNormalization	Normalization	-
Activation	Activation	ReLU
Dropout	Regularization	Rate: 0.5
Dense	Dense	Units: 7, Activation: Softmax

Table 2.6: CNN architecture with MobileNetV2 as the backbone

Layer	Type	Details
MobileNetV2	Pretrained model	Principal layer
Flatten	Flatten	Flatten the input
Dense	Dense	128 units, ReLU activation
Dropout	Dropout	Dropout rate of 0.5
Dense	Dense	9 units with softmax activation

Table 2.7: CNN architecture with Resnet50 as the backbone

Layer	Type	Details
Resnet 50	Pretrained model	Principal layer
Flatten	Flatten	Flatten the input
Dense	Dense	128 units, ReLU activation
Dropout	Dropout	Dropout rate of 0.5
Dense	Dense	9 units with softmax activation

To ensure consistency, all intermediate layers and their configurations were retained exactly as specified in Table 2.5. The resulting architectures occupy different positions along the accuracy–efficiency spectrum, and their performance on the FairFace age classification task is evaluated in Section 3.

2.3.3. Model architectures for gender recognition

The same three core architectures evaluated for age recognition, MobileNetV2 [60], ResNet50 [46], and the proposed hybrid CNN-Transformer model (Table 2.5), were also employed for the binary gender classification task using the FairFace dataset.

The setup for this task followed the same structure as the one used for age recognition. It used either the pre-trained backbones or the custom core architecture, processing 224x224 RGB input images. The only major difference was in the final classification layer: for all three architectures, it was replaced with a dense layer that had a single output unit and a sigmoid activation function. This output represented the probability of one class (e.g., female), with the probability of the other class (1 − output).

This minimal adaptation allowed for a direct comparison of the architectures performance on the gender task, reusing the feature extraction capabilities developed for the age task or inherited from pre-training. Performance results for gender recognition are presented in Section 3.

2.4. Training configuration

All models were created in Python (v3.10.0) using the TensorFlow (v2.15.0) framework and the Keras API. The training procedure was mostly carried out on Google Colab, using freely available NVIDIA T4 GPU resources.

The Adam optimizer was employed for gradient descent, chosen for its efficiency and adaptive learning rate capabilities, using the default framework parameters (β1 = 0.9, β2 = 0.999). An initial learning rate of 0.001 was determined. To dynamically change the learning rate during training, a scheduler ('ReduceLROnPlateau' Keras callback) monitored the validation loss; if no progress was seen for four consecutive epochs, the learning rate was half (by a factor of 0.5).

In terms of loss functions, Categorical Cross-Entropy was utilized to classify multi-class emotions (7 classes) and ages (9 classes). For the binary gender classification, Binary Cross-Entropy loss was utilized, corresponding to the model's single sigmoid output unit. A consistent batch size of 64 was used across all training procedures.

Training was set to run for a maximum of 100 epochs. However, to reduce overfitting and pick the best performing model iteration, early stopping ('EarlyStopping' Keras callback) was used. This callback monitored the validation loss and halted the training process if no improvement was noticed after 10 consecutive epochs. Specifically, the model weights from the epoch with the lowest validation loss were automatically recovered and preserved as the final weights for each training model.

2.5. Model optimization with TensorRT

To improve inference performance for real-time deployment, the chosen model architecture was optimized using NVIDIA TensorRT [66]. The optimization technique was executed directly on the target Jetson Xavier NX platform, which was running JetPack 4.5 for proper hardware optimizations.

The workflow involved two main stages. First, the TensorFlow-trained models were converted to ONNX format using the Python tool tf2onnx. This intermediate representation improves interoperability with the TensorRT optimization pipeline.

Subsequently, the TensorRT command-line tool trtexec, included with JetPack 4.5, was utilized to build the optimized inference engine from the ONNX model file. Crucially, the optimization targeted FP16 precision by invoking trtexec with the –fp16 flag. This procedure uses FP16 capabilities to reduce latency and memory utilization without the need for a calibration dataset, performs several graph optimizations, and creates a serialized engine file that is specifically tailored for the Jetson Xavier NX's GPU. The real-time experiment and all ensuing performance assessments were then conducted using this optimized file.

2.6. Evaluation metrics

To comprehensively evaluate the performance of the developed facial attribute recognition models, a combination of standard classification metrics and efficiency measures was employed. The choice of metrics aimed to provide insights into both overall predictive capability and specific model behaviors, particularly considering potential class imbalances in the datasets.

2.6.1. Classification performance metrics

The following metrics were computed, usually on a per-class basis when appropriate, and frequently aggregated, to assess the predicted performance on emotion, age group, and gender categorization tasks.

Accuracy: Defined as the overall percentage of correctly classified samples across all classes, presented as:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \tag{2.1}$$

While intuitive, accuracy can be misleading on imbalanced datasets, as a model might achieve high accuracy by simply predicting the majority class well while performing poorly on minority classes. Therefore, it was primarily used as a general indicator alongside other metrics.

Precision: Measures the proportion of correctly predicted positive instances among all instances predicted as positive for a specific class C. This calculation relies on counts of True Positives (TP_C, correctly predicted as C), False Positives (FP_C, incorrectly predicted as C), and False Negatives (FN_C, incorrectly predicted as not C when they are C). Precision is calculated as follows:

$$Precision_C = \frac{TP_C}{TP_C + FP_C} \tag{2.2}$$

High precision for a class indicates that when the model predicts that class, it is likely correct. It is important for minimizing false alarms (e.g., incorrectly identifying a neutral face as angry). Precision analysis helps identify specific classes where the model generates many incorrect positive predictions.

Recall: Measures the proportion of actual positive instances that were correctly identified by the model for a specific class C as follows:

$$Recall_C = \frac{TP_C}{TP_C + FN_C} \tag{2.3}$$

High recall for a class indicates that the model successfully identifies most instances belonging to that class. It is crucial for minimizing missed detections (e.g., failing to detect an important emotion like fear or sadness, or misclassify an older adult into a much younger category).

F1-Score: The harmonic meaning of Precision and Recall for a specific class C, providing a single score that balances both metrics and is defined as follows:

$$F1 - score_C = \frac{Precision_C \cdot Recall_C}{Precision_C + Recall_C} \tag{2.4}$$

The F1-score is particularly useful for evaluating performance on imbalanced datasets (as encountered in emotion and age tasks) because it penalizes models that achieve high precision at the cost of recall, or vice versa. Often, the macro-average F1-score (unweighted average across all classes) is reported to give equal importance to each class, regardless of its size.

Confusion Matrix: A figure that visualizes the performance of a classification model by summarizing the counts of actual versus predicted classes. The diagonal elements represent correct classifications, while off-diagonal elements represent misclassifications. Analyzing the confusion matrix is essential for understanding how the model errors. Specifically, which classes are most frequently confused with each other (e.g., fear mistaken for surprise, or adjacent age groups being confused). This provides deep insights into model behavior beyond single aggregated scores.

2.6.2. Classification performance metrics

Given the target deployment on the resource-constrained Jetson Xavier NX platform for real-time HRI, evaluating computational efficiency was critical. The following metrics were measured directly on the target hardware.

Inference Time (Latency): The average time, usually expressed in milliseconds (ms), needed for the model to process one input image and provide an output prediction (using a batch size of one for this measurement). Low latency is critical for responsive engagement. Measurements were averaged over many inferences runs after an initial warm-up phase to achieve consistent timings.

Frames Per Second (FPS): Calculated as the reciprocal of the total processing time per frame:

$$FPS = \frac{1}{Inference\ Time} \quad (2.5)$$

FPS represents the throughput of the system, indicating how many video frames can be processed each second. Higher FPS is essential for smooth real-time video processing.

Model Size: The storage space occupied by the models measured in Megabytes (MB). Smaller models are preferable for deployment of devices with limited storage capacity.

These complementary sets of metrics provide a holistic view of model performance, balancing predictive accuracy across potentially imbalanced classes with the practical computational constraints of the target edge deployment scenario.

2.7.Real-Time experiment design

To assess the practical performance and robustness of the final optimized model under more realistic, dynamic conditions, a real-time experiment involving human participants was designed and conducted. The primary focus was to evaluate the system’s accuracy, particularly for emotion recognition, under varying head poses generated naturally by participants while interacting with the system running live on the target hardware.

2.7.1. Participants

A total of 36 volunteers participated in the experiment. Prior to the session, each participant’s self-reported age group (mapped to the 9 FairFace categories) and gender were recorded to serve as ground truth for those tasks. The resulting age and gender distributions of the participant group are shown in Figure 2.4 and Figure 2.5, respectively.

It is important to note that this convenience sample exhibits a significant age imbalance, with 30 participants falling within the 20–29 age group, and a gender imbalance, with only 7 female participants. These imbalances limit the generalization of age and gender recognition results. However, the dataset still gives useful information about emotion recognition over a wide range of people in the sample. Furthermore, the presence of such an imbalance emphasizes the need of selecting proper evaluation metrics.

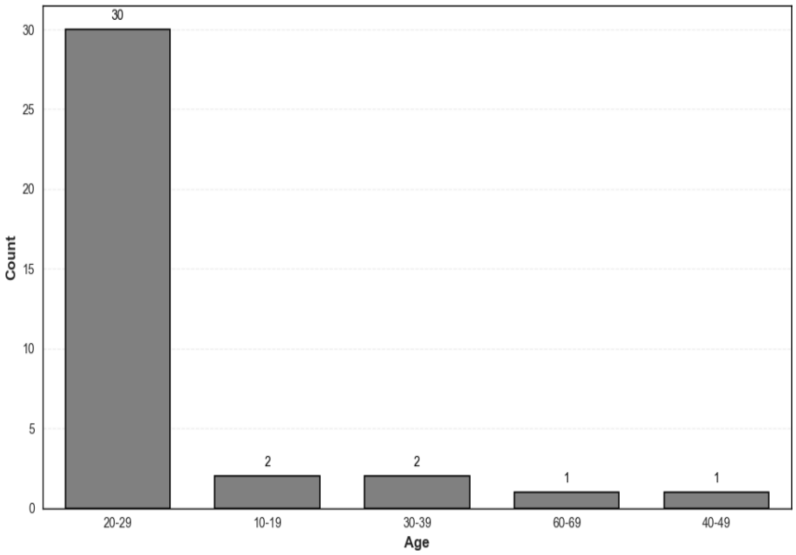


Fig. 2.4. Experiment age distribution.

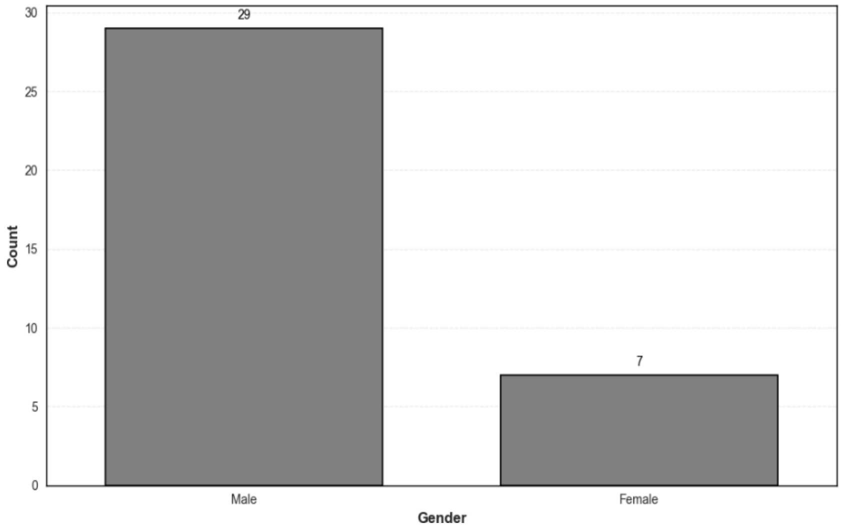


Fig. 2.5. Experiment gender distribution.

2.7.2. Experimental setup

The experiment took place in a typical indoor room environment with natural illumination conditions, meaning lighting varied depending on the time of day and weather, deliberately avoiding controlled studio lighting. Participants sat about a meter in front of the Logitech C270 HD webcam, which was mounted on the top of a computer screen. The system captured video frames at a resolution of 720, although the effective frame rate processed by the system during the experiment fluctuated between 12-15 FPS due to the computational load on the NVIDIA Jetson Xavier NX platform, which ran the entire perception pipeline in real-time. Camera calibration, according to the principles on [68] was performed beforehand primarily to ensure a consistent geometric setup. Finally, the full code in python for the experiment is shown in the Appendices Section.

2.7.3. Procedure

Each participant was guided by the experimenter through the following procedure for each of the 7 target emotions (Happiness, Sadness, Anger, Surprise, Fear, Disgust, Neutral):

1. **Emotion Prompt:** The experimenter verbally told the individual which emotion to fake next. This triggered feeling was used as the ground truth label for emotion recognition.
2. **Simulation and Head Movements:** The participant was asked to replicate the prompted facial expression while making continuous, natural head movements in various orientations:
   - Side-to-side rotation (yaw, approximately center-left-center-right).
   - Up-and-down tilting (pitch, approximately center-up-center-down).

Participants were encouraged to maintain the expression throughout the movements for each emotion trial.

3. **Duration:** Each emotion trial lasted approximately 7-15 seconds to capture various poses.

2.7.4. Real-Time system and data logging

The custom application on the Jetson Xavier NX executed the following real-time pipeline for each incoming frame (running at 12-15 FPS):

1. **Video Capture:** Frame acquisition from the camera.

2. **Face Detection:** The principal face was recognized by Dlib's HOG feature-based face detector. If no face was reliably detected, the frame was skipped.
3. **Preprocessing:** The detected facial region was cropped and preprocessed as described in Section 2.2.
4. **Inference:** The optimized models were fed the preprocessed face in order to estimate gender, age group, and emotion. To gather real-world performance data, the inference time for each frame processing was tracked during each session.
5. **Error Logging:** A crucial aspect of the system was logging instances where predictions potentially diverged from the ground truth. Specifically, when the predicted emotion, age, or gender did not match the prompted label, the system automatically saved the corresponding image, the incorrect prediction, and the timestamp for later analysis.

2.7.5. Post-Processing for pose analysis.

To analyze the impact of head orientation on prediction accuracy, a post-processing step was performed on the logged error frames:
1. **Facial Landmark Detection:** Dlib's facial landmark predictor [69] was used to detect key points on the saved face images.
2. **Head Pose Estimation**: Based on these landmarks, the approximate head pose (specifically, horizontal orientation: left, center, right; and vertical orientation: up, center, down) was estimated for each error frame with explicit geometric relationships with the landmarks [70].
3. **Image Quality Analysis:** For each logged image, additional visual quality metrics were saved, including illumination, contrast, and blur level. These features help provide deeper insights into how image conditions may affect prediction accuracy.
4. **Manual Verification:** The automatically estimated pose labels for the error frames were manually reviewed and corrected where necessary to ensure accuracy for the subsequent failure analysis.

This combined real-time logging and post-processing methodology provided the necessary data to evaluate not only the overall performance but also the specific conditions under which the optimized system failed during the experiment.

2.8. Evaluation protocol

An evaluation process was developed to systematically examine the performance of the implemented architectures across all three tasks (emotion, age, and gender detection) and select the best candidate for optimization and real-time deployment. The evaluation method consisted of the following main steps:
Step 1: Baseline Performance Evaluation (Before Optimization): First, all implemented architectures, after being trained on their respective primary datasets (FER2013 for emotion, FairFace for age/gender) as described in Section 2.4, were evaluated in their original, unoptimized state. This baseline evaluation included:

- **Validation Set Performance:** Performance was assessed using accuracy and loss curves on the validation split of the main training datasets (FER2013 PublicTest set and FairFace validation set). These curves help evaluate how well the model generalized during training. Up-and-down tilting (pitch, approximately center-up-center-down).
- **Generalization Set Performance:** Evaluating performance using the full suite of classification metrics defined in Section 2.6 on the independent generalization datasets (CK+

for emotion, UTKFace for age/gender). This step measured the models' ability to generalize data with different characteristics.

- **Computational Efficiency on Target Hardware:** Measuring the baseline computational performance of each unoptimized model directly on the NVIDIA Jetson Xavier NX platform.

Step 2: Model Selection for Optimization: Based on the results of Step 1, a thorough examination of both classification performance (especially on the generalization datasets) and baseline processing efficiency (latency/FPS on Jetson) was carried out. The architecture demonstrating the most promising overall balance between predictive accuracy and suitability for real-time edge deployment across the tasks was selected for further optimization using TensorRT.

Step 3: Post-Optimization Performance Evaluation: The selected models were optimized using the TensorRT FP16 quantization procedure described in Section 2.5. Following optimization, the evaluation process from Step 1 was repeated for the optimized model(s):

- **Re-evaluation of Classification Performance:** Performance on the generalization datasets was re-assessed using the same classification metrics to quantify any potential impact (positive or negative) of FP16 optimization on accuracy, precision, recall and f1-score.
- **Re-evaluation of Computational Efficiency:** Inference time, FPS, and model size were remeasured on the Jetson Xavier NX for the optimized TensorRT engine. These results were compared against the baseline measurements to quantify the speedup and size reduction achieved through optimization.

Step 4: Real-Time Experimental Evaluation: Finally, the selected and most effective optimized models were eventually deployed in the real-time experiment with human participants, as stated in Section 2.7. This final step was designed to test the system's practical performance and robustness in a dynamic, interactive environment that resembles real-world settings.

This structured protocol allowed for a systematic comparison of architectures, quantification of optimization benefits, and validation under increasingly realistic conditions, leading to the results and analysis presented in Section 3.

3. Results

This section presents the performance metrics on the validation datasets for all trained models, followed by their results on the generalization datasets along with computational metrics. It then details the models selected for optimization, presents the post-optimization performance and computational metrics, and concludes with the final experimental results.

3.1. Model performance on validation datasets (Before optimization)

3.1.1. Emotion models

For the emotion models, the one proposed by Seringel et al, [2], in the original paper does not provide information on the training loss or accuracy curves; it only reports an overall accuracy of 57.4% for the emotion recognition task.

In contrast, for the InceptionV3 architecture, detailed training and validation performance are available. The following figure illustrates the progression of accuracy and loss across epochs, as shown in Figure 3.1.

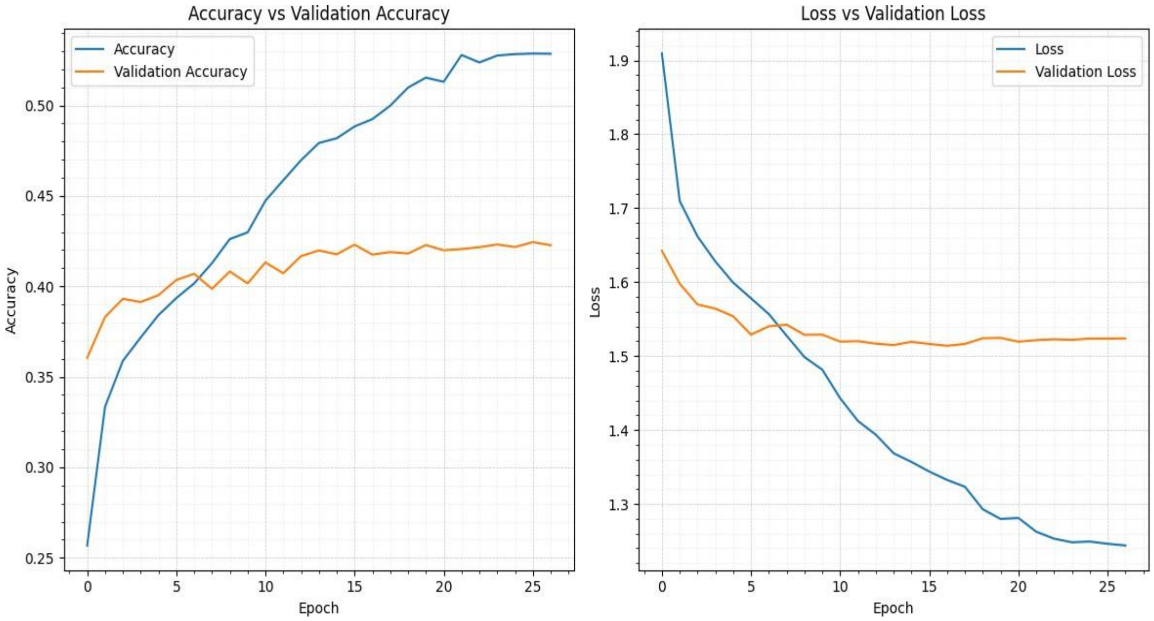


Fig. 3.1. Training and validation performance for the InceptionV3 model on the FER2013 dataset.

The training accuracy improved steadily across the epochs, reaching over 53%, but the validation accuracy plateaued at roughly 42% after the 10th epoch. This early stabilization of validation accuracy, in contrast to the ongoing improvement in training accuracy, indicated overfitting. Similarly, the training loss reduced gradually during the training phase, however the validation loss exhibited little progress after the first few epochs, oscillating slightly around 1.52. This behavior suggested that the model had reached a plateau, implying that learning parameters were no longer contributing to generalization.

Continuing with the ResNet architecture, Figure 3.2 shows the progression of accuracy and loss across epochs during the training process. The curves provide insight into the model's learning dynamics.

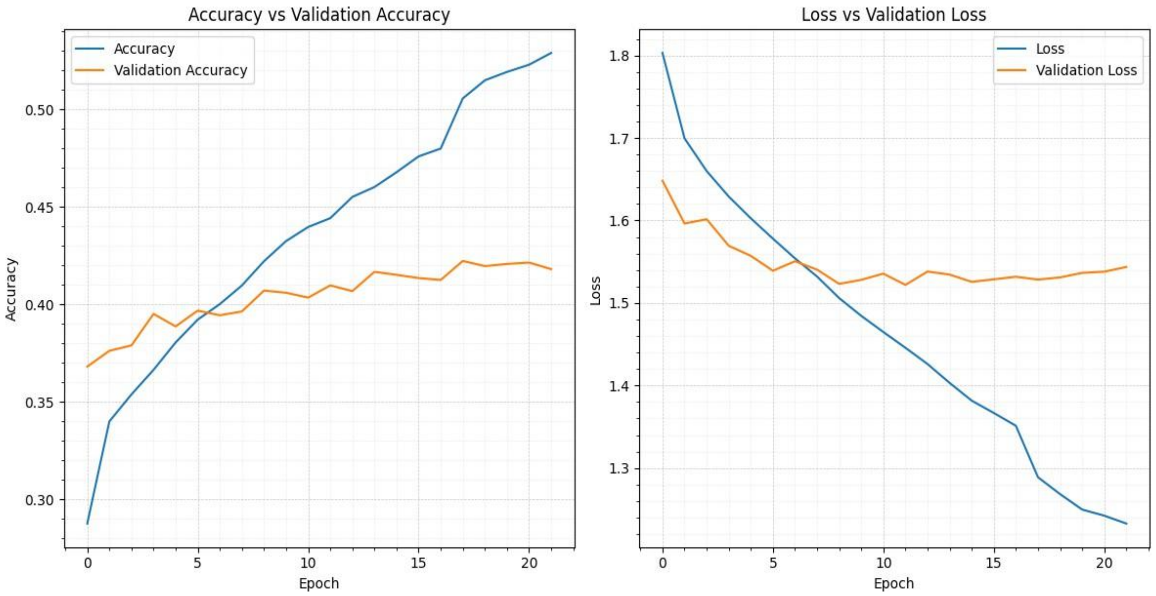


Fig. 3.2. Training and validation performance for the ResNet50 model on the FER2013 dataset.

The training accuracy showed a rapid improvement over the epochs, while the validation accuracy increased at a slower rate, reaching approximately 53% and 41%, respectively. In contrast, the training loss decreased steadily to around 1.35; however, the validation loss began to increase after epoch 15, reaching up to 1.65, which suggested clear signs of overfitting.

Based on the architecture suggested by Priyadarshini et al., Figure 3.3 depicts the growth of accuracy and loss over epochs during training. These curves provide crucial information about the model's learning processes.

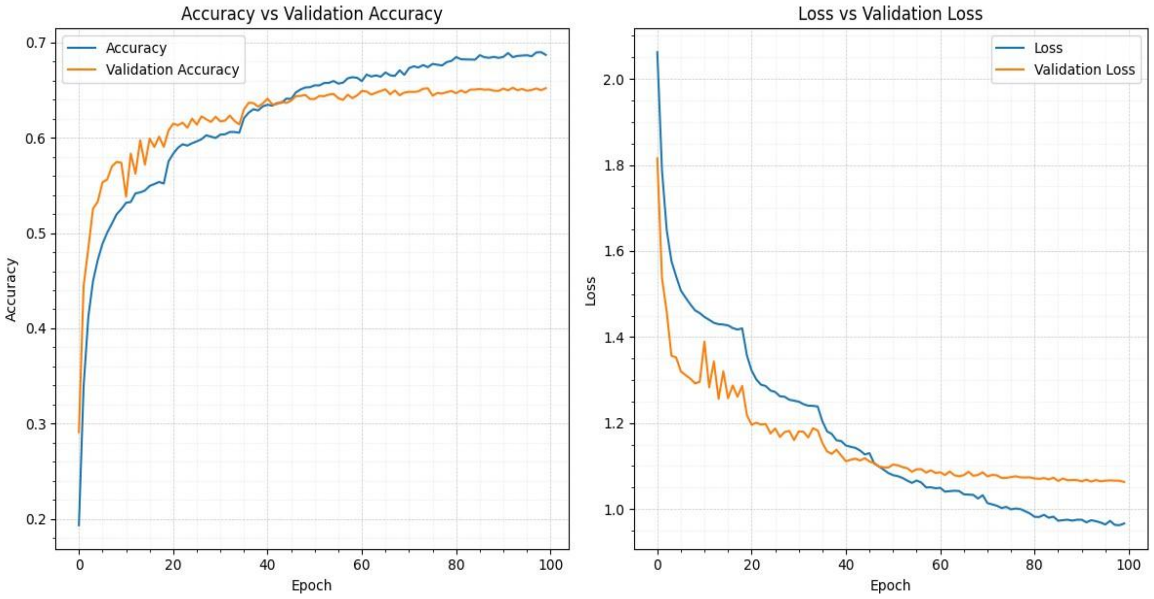


Fig. 3.3. Training and validation performance for the model in [1] on the FER2013 dataset.

In general, this model exhibits better learning dynamics compared to the other architectures. Training accuracy improved across epochs and stabilized at around 69%. Similarly, the validation accuracy improved significantly and stabilized early, at epoch 43, reaching roughly 65 percent. On the other hand, the training loss steadily decreased to around 0.9; however, the validation loss reached a plateau near epoch 43 at approximately 1.1, suggesting that the model had stopped learning features that contributed meaningfully to generalization.

Finally, the training dynamics of the model designed for this thesis are presented in Figure 3.4.

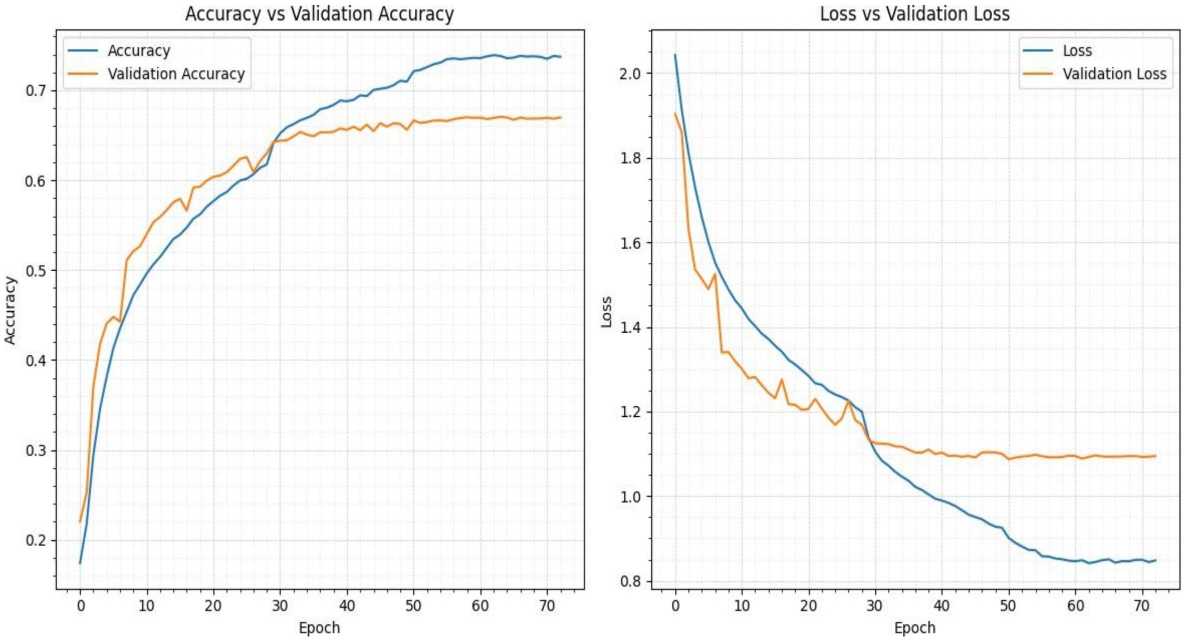


Fig. 3.4. Training and validation performance for the Hybrid CNN on the FER2013 dataset.

The training dynamics are like those shown in Figure 3.3, but here, the training accuracy stabilized around 74%, and the validation accuracy reached 67% around epoch 60. Meanwhile, the loss remained stable at 0.85, with the validation loss around 1.1, indicating that more training did not result in significant increases in generalization.

The Table 3.1 presents a summary of the best validation accuracy and the loss for the trained models is presented.

Table 3.1: Summary of best validation performance for emotion recognition models on the FER2013 validation set.

Model Architecture	Best Validation Accuracy (%)	Lowest Validation Loss	Epoch (Approx.)
Seringel et al. [2]	57.4	N/A	N/A
InceptionV3	42.1	1.52	16
ResNet50	41	1.52	11
Priyadarshini V et al. [1]	65	**1.1**	90
Proposed Hybrid CNN-Transformer	**67**	**1.1**	61

3.1.2. Age models

The Figure 3.5 shows the training dynamic for the age model on the MobileNetV2 architecture.

The training and validation accuracies exhibited a similar pattern, initially increasing and then attempting to stabilize. The validation accuracy reached approximately 42%, while the training accuracy stabilized around 41.5%. In terms of losses, 1.45 was the convergence point for both training and validation losses. The curves leveling out suggests that the model is no longer learning efficiently.

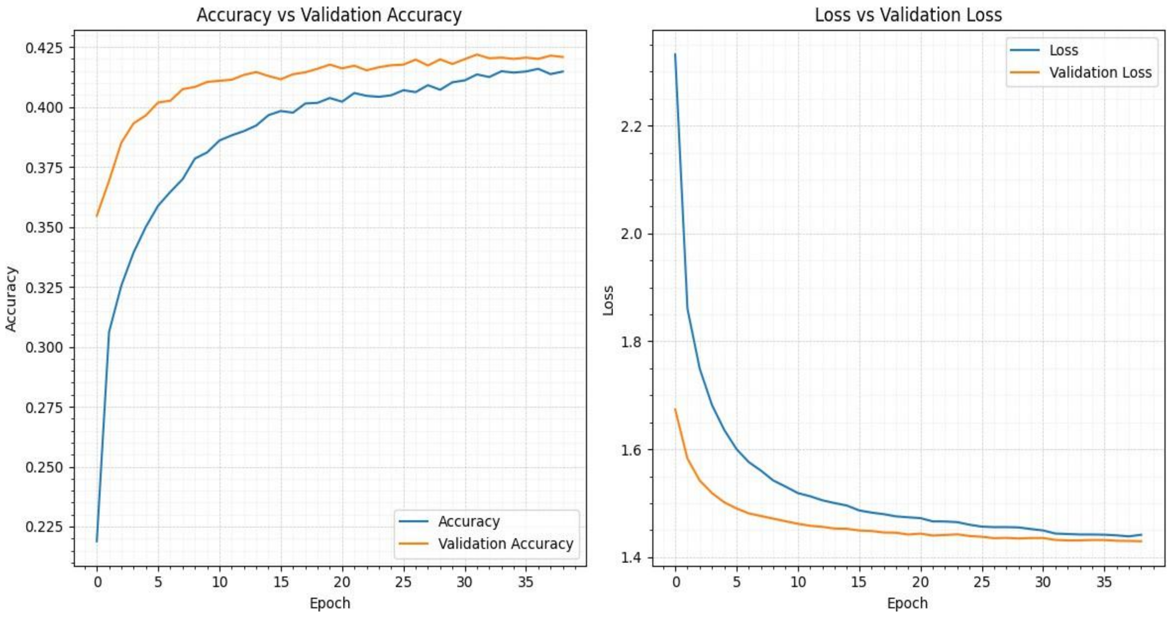


Fig. 3.5. Training and validation performance for MobileNetV2 on the age Fairface dataset.

Building on the ResNet model, the training curves are presented in Figure 3.6.

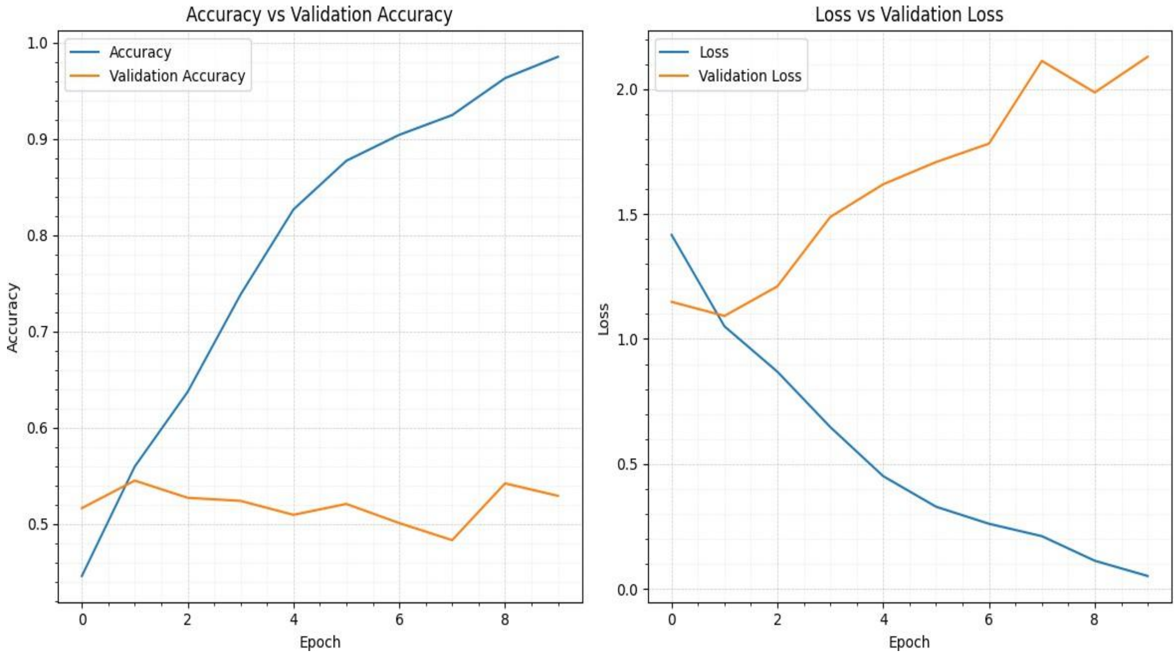


Fig. 3.6. Training and validation performance for the ResNet50 on the age Fairface dataset.

In this case, ResNet learns very quickly, reaching nearly 90% training accuracy by epoch 10. However, the validation accuracy stagnates at approximately 54% as early as epoch 1. Additionally, the training loss decreases rapidly, approaching zero by epoch 10, while the validation loss reaches its minimum of 1.1 at epoch 1 and begins to increase thereafter, eventually reaching 2.2 by epoch 10. This divergence between the training and validation curves indicates a clear and rapid overfitting pattern. While the best validation performance was achieved early on, the subsequent overfitting suggests that, for age recognition on this dataset, this architecture might benefit from adjustments to hyperparameters, such as a lower initial learning rate or stronger regularization.

For the Hybrid CNN-Transformer model, Figure 3.7 illustrates the training dynamics.

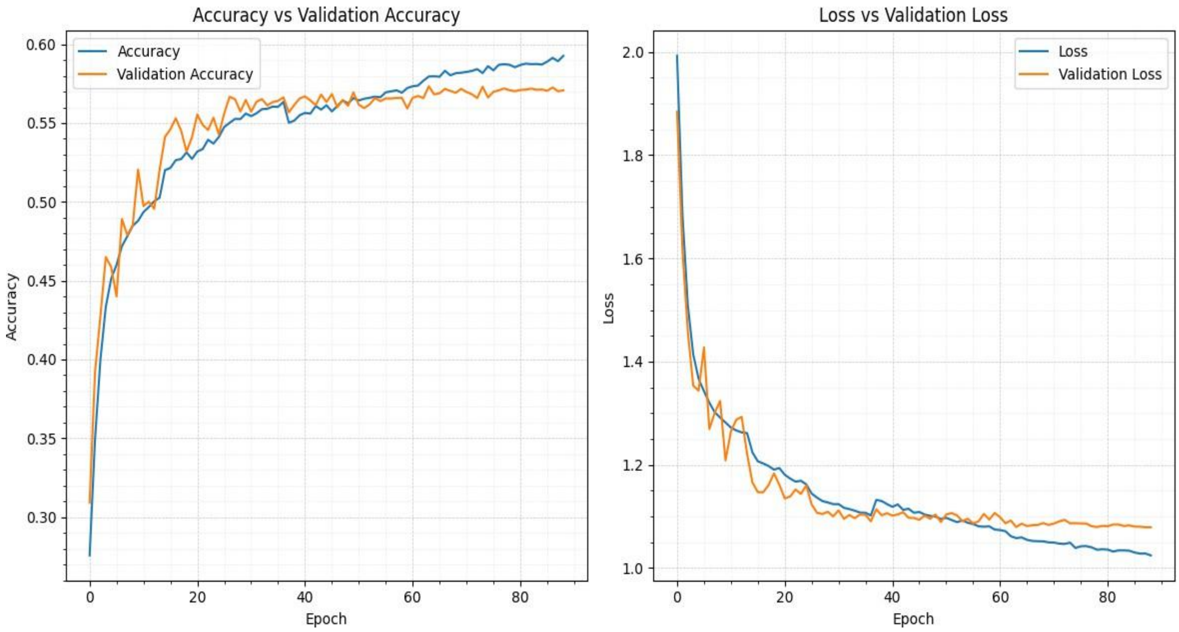


Fig. 3.7. Training and validation performance for the Hybrid CNN-Transformer on the age Fairface dataset.

For the accuracy curves, both the training and validation accuracy increase at the same rate until epoch 60, where the training accuracy continues to grow, reaching 59%, while the validation accuracy stabilizes at 57%. Similarly, both losses reduce steadily until epoch 60, when the validation loss plateaus at 1.08 and the training loss continues to fall, reaching around 1.02.

The Table 3.2 presents a summary of the best validation accuracy and the loss for the trained models is presented.

Table 3.2: Summary of best validation performance for age recognition models on the FairFace validation set.

Model Architecture	Best Validation Accuracy (%)	Lowest Validation Loss	Epoch (Approx.)
MobileNetV2	41.5	1.45	30
ResNet50	54	1.1	1
Proposed Hybrid CNN-Transformer	**57**	**1.09**	70

3.1.3. Gender models

Figure 3.8 illustrates the training dynamics for the gender classification model using MobileNetV2 architecture.

The training accuracy increases steadily, reaching a maximum of 78%. Meanwhile, the validation accuracy takes approximately 15 epochs to begin improving, after which it rises slightly and stabilizes around 77%. On the other hand, the training and validation loss curves behave similarly, falling regularly and stabilizing around epoch 25, with values of 0.46 for the training loss and 0.47 for the validation loss. This plateau indicates that the model has ceased learning features that help with further generalization.

Figure 3.9 displays the training and validation curves for both accuracy and loss of the ResNet50 based model.

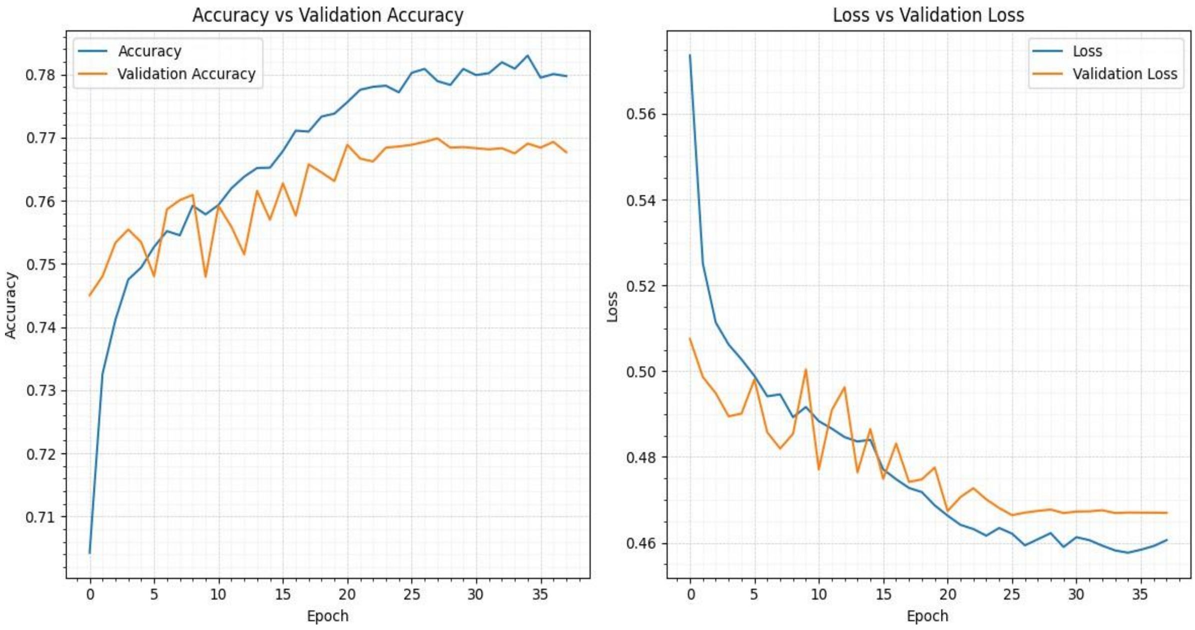


Fig. 3.8. Training and validation performance for the MobileNetV2 on the gender Fairface dataset.

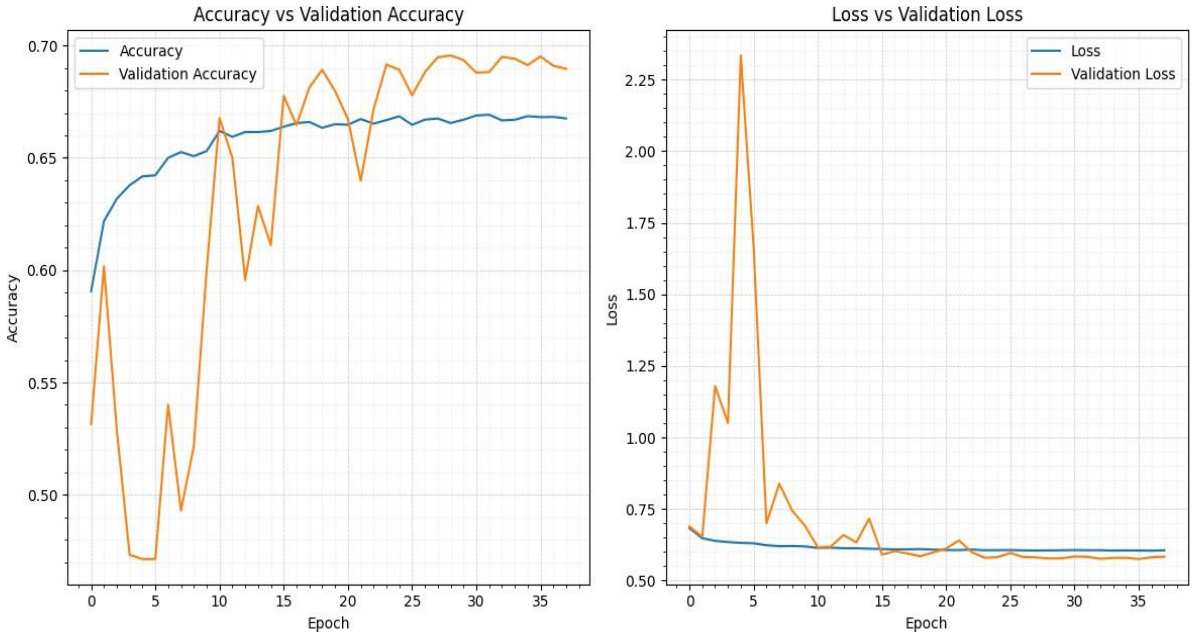


Fig. 3.9. Training and validation performance for the ResNet50 on the gender Fairface dataset.

For the ResNet model, the training accuracy increases progressively but stabilizes after epoch 25 at around 67%. In contrast, the validation accuracy demonstrates a more erratic trend, with low values up to epoch 10, before beginning to increase and stabilizing around 69% from epoch 25 forward. On the other hand, the loss curves converge to approximately 0.6. However, the validation loss initially shows high peaks, reaching up to 2.3, but then progressively decreases and stabilizes after epoch 15. This convergence suggests that the model has stopped learning features that enhance generalization. Figure 3.10 presents the training behavior of the Hybrid CNN-Transformer model.

Before stabilizing at about 93% and 92.5%, respectively, the training and validation accuracy curves show a similar upward trend until epoch 45, when the training accuracy increases a little more. Similarly, the loss curves reduce gradually until epoch 50, at which point the training loss continues to drop somewhat, reaching about 0.22, while the validation loss levels out at about 0.24.

The Table 3.3 provides a summary of the best validation accuracy and corresponding loss achieved by the trained models for gender recognition.

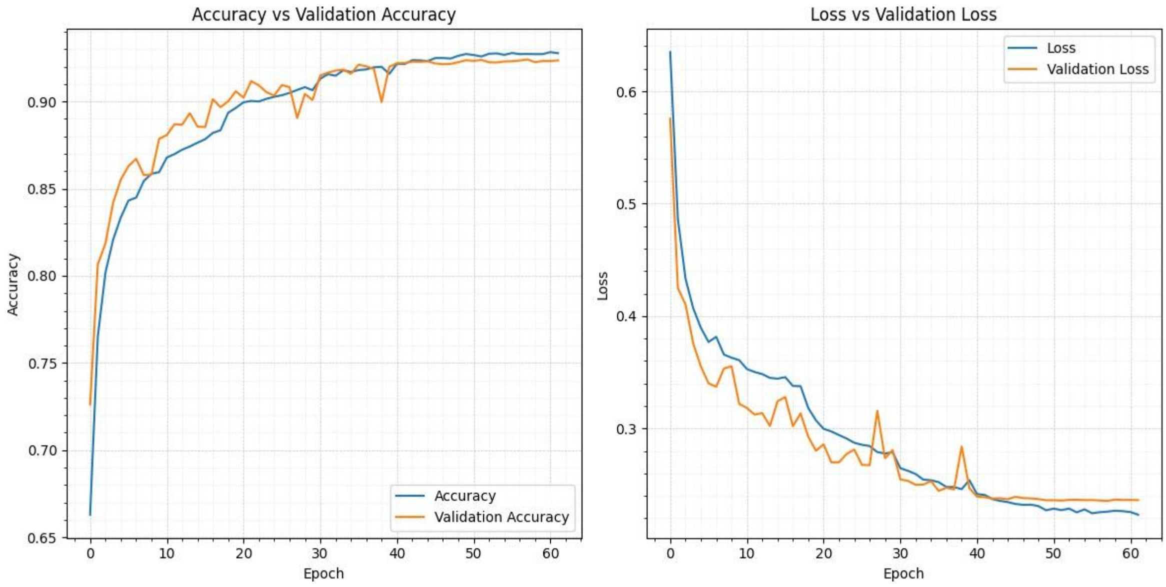


Fig. 3.10. Training and validation performance for the Hybrid CNN-Transformer on the gender Fairface dataset.

Table 3.3: Summary of best validation performance for gender recognition models on the FairFace validation set.

Model Architecture	Best Validation Accuracy (%)	Lowest Validation Loss	Epoch (Approx.)
MobileNetV2	77	0.47	27
ResNet50	69	0.6	26
Proposed Hybrid CNN-Transformer	**92.5**	**0.24**	50

3.2.Model performance on generalization datasets (Before optimization)

3.2.1. Emotions

Table 3.4 summarizes the overall evaluation metrics along with the F1 scores for each of the seven emotion classes.

Given the large class imbalance already seen in the CK+ dataset, the macro-average F1-score gives a more reliable assessment of overall model performance than accuracy alone. As shown in Table 3.4, the proposed Hybrid CNN-Transformer model demonstrates superior performance in this regard, achieving the highest macro-average F1 score 0.68. It also leads in macro-average precision 0.70 and recall 0.69, alongside attaining the best overall accuracy 0.87. These findings imply that the Hybrid model has the best generalization ability across all emotion classes, including those with less samples.

Table 3.4: Performance comparison of emotion recognition models.

Model Architecture	Overall				Class-wise F1 Scores						
	Acc	Prec	Rec	F1	Ang.	Disg.	Fear	Hap.	Sad	Surp.	Neut.
Seringel et al. [2]	0.70	0.56	0.52	0.48	0.14	0.26	0.17	0.86	0.18	0.89	0.84
InceptionV3	0.82	0.61	0.54	0.55	0.21	0.57	0.09	**0.98**	0.30	0.75	0.92
ResNet50	0.81	0.55	0.50	0.49	0.06	0.12	0.15	0.92	0.38	0.89	0.91

Priyadarshini et al. [1]	0.78	0.61	0.61	0.58	0.33	0.48	0.32	0.92	0.18	**0.96**	0.84
Proposed Hybrid	**0.87**	**0.70**	**0.69**	**0.68**	**0.39**	**0.62**	**0.41**	0.95	**0.49**	0.95	**0.95**

Note: Acc = Accuracy, Prec = Precision, Rec = Recall, F1 = F1-Score. Class abbreviations: Ang. = Anger, Disg. = Disgust, Hap. = Happy, Surp. = Surprise, Neut. = Neutral.

Notably, the proposed Hybrid model further demonstrates its advantage by achieving respectable F1 scores even in challenging minority classes like Anger 0.39, Disgust 0.62, Fear 0.41, and Sadness 0.49.This contrasts sharply with the baseline model from Seringel et al.[2] (0.48 macro F1), which struggles significantly across most non-majority classes (e.g., achieving only 0.14 F1 for Anger, 0.17 for Fear, and 0.18 for Sadness). While the lightweight model by Priyadarshini et al.[1] offers improved balance with 0.58 macro F1 compared to the baseline and performs well on Surprise 0.96 F1, it still lags behind the Hybrid model, particularly in recognizing Sadness where it scores only 0.18 compared to the Hybrid's 0.49.

In contrast to the more balanced performance of the Hybrid model, the transfer learning models based on deeper architectures, InceptionV3 and ResNet50, exhibit highly uneven performance across different emotion classes, despite their respectable overall accuracy scores 0.82 and 0.81, respectively. As highlighted in Table 3.4, InceptionV3 achieves near perfect recognition of Happiness 0.98 F1 but almost entirely fails to identify Fear 0.09 F1. Similarly, ResNet50 performs well on Happiness 0.92 F1 yet struggles significantly with Anger 0.06 F1 and Disgust 0.12 F1. These discrepancies underscore their limited ability to generalize effectively to the less frequent emotion categories within the CK+ dataset, favoring the majority classes instead.

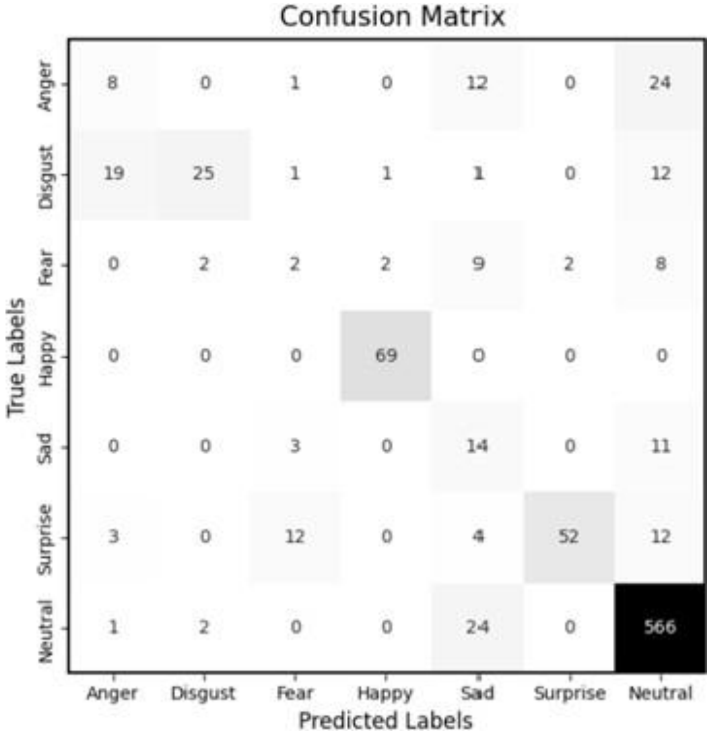
Overall, the proposed Hybrid CNN-Transformer model emerges as the most robust and consistent solution, leading in all evaluation metrics and providing a more balanced performance in the presence of class imbalance.

Although Table 3.4's aggregated metrics offer a useful summary, examining confusion matrices gives more detail on the particular mistake patterns and class confusions for each model. To illustrate these patterns without excessive repetition, the following analysis focuses on the confusion matrices for three representative architectures: InceptionV3 (as a strong transfer learning baseline), the lightweight model by Priyadarshini et al. [1] (representing custom efficient designs), and the top-performing proposed Hybrid model. The following section begins with an analysis of InceptionV3's confusion matrix.

The analysis of the InceptionV3 confusion matrix in Figure 3.11 supports the uneven performance seen in Table 3.4. The model performs very well in recognizing Happy faces, which matches its high F1 score of 0.98. However, it demonstrates significant shortcomings in distinguishing other emotions. Fear, for example, is recognized correctly only twice and is frequently confused with Sad or Neutral, resulting in an extremely low F1 score of 0.09. There is also a high level of ambiguity between Disgust and Anger, with 19 Disgust samples predicted as Anger. A similar, albeit minor, issue arises when Sad is misclassified as Neutral. These patterns suggest that, despite its complex architecture, InceptionV3 has trouble telling apart emotions that look similar or that are less represented in the CK+ dataset.

Turning to the confusion matrix for the lightweight model by Priyadarshini et al. [1] in Figure 3.12, we can see some of the same confusion patterns as in InceptionV3, but also clear improvements. This model shows better recognition for Fear, with an F1 score of 0.32, and Anger, with an F1 score of 0.33. It also maintains strong performance on Surprise, with an F1 of 0.96, which matches the results shown in Table 3.4. However, there are still challenges. Disgust is often misclassified as Anger, though this happens a bit less than with InceptionV3. Some Sad samples are also confused with

Neutral. In addition, because there are so many Neutral samples in the dataset, other emotions are sometimes wrongly predicted as Neutral. This shows how the class imbalance continues to affect the model's performance.



True \ Predicted	Anger	Disgust	Fear	Happy	Sad	Surprise	Neutral
Anger	8	0	1	0	12	0	24
Disgust	19	25	1	1	1	0	12
Fear	0	2	2	2	9	2	8
Happy	0	0	0	69	0	0	0
Sad	0	0	3	0	14	0	11
Surprise	3	0	12	0	4	52	12
Neutral	1	2	0	0	24	0	566

Fig. 3.11. Confusion Matrix for Inception V3 on CK+ dataset.

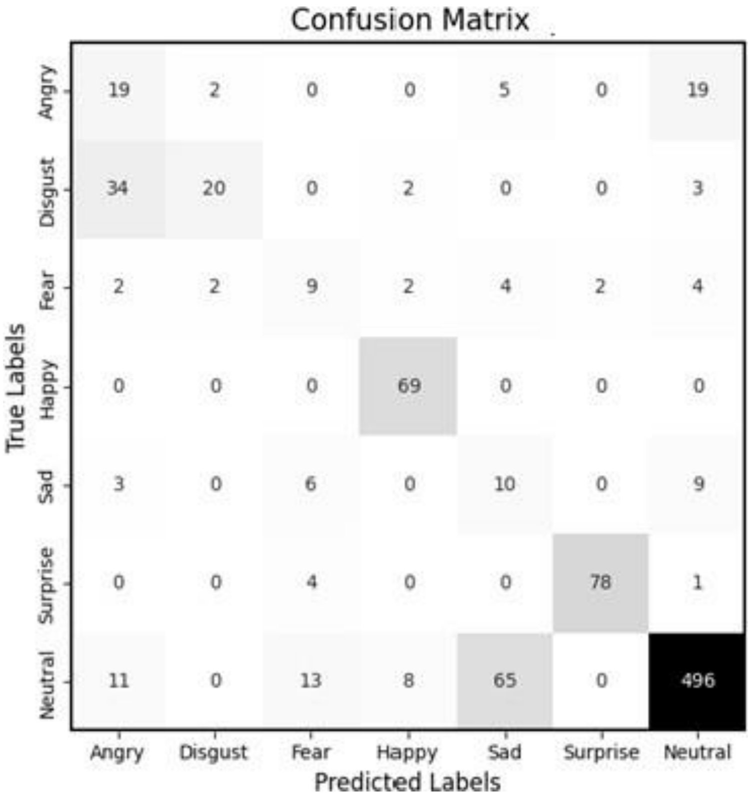
Finally, the confusion matrix for the proposed Hybrid CNN-Transformer model in Figure 3.13 visually confirms its superior overall performance. Particularly for difficult emotions like anger, disgust, fear, and sadness, the accurate classifications along the diagonal are typically more prominent than in the prior matrices (Figures 3.11 and 3.12), suggesting fewer misclassifications. While some residual confusion remains, particularly between Disgust and Anger, the overall pattern clearly shows a more balanced and accurate recognition across all emotion classes, reinforcing its stronger generalization capability on the CK+ dataset.

The evaluation on the CK+ generalization dataset, which achieved the greatest overall metrics and showed the most balanced recognition across the seven emotion classes, concluded that the suggested Hybrid CNN-Transformer was the best design. Persistent, task-inherent challenges were nevertheless exposed by examination of the confusion matrices. Common error patterns included significant confusion between Disgust and Anger, difficulties distinguishing Sad from Neutral, and generally poor recognition rates for the underrepresented Fear category across most architectures.

3.2.2. Age

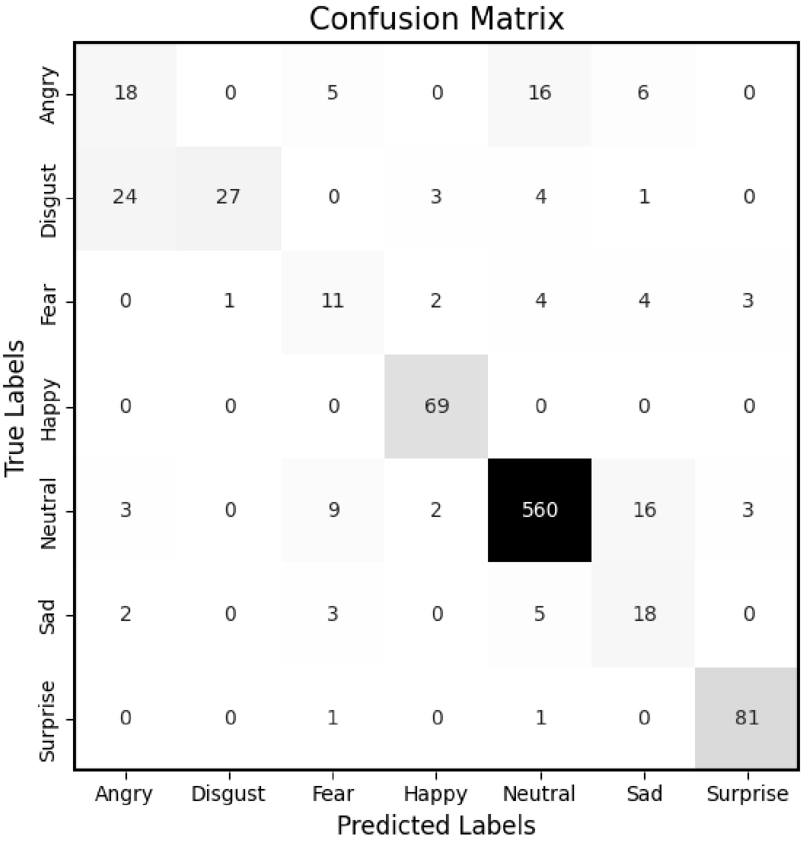
Table 3.5 summarizes the overall evaluation metrics along with the F1 scores for each of the nine age group classes.

Given the class imbalance in the UTK Face dataset, the macro-average F1-score is a better indicator of model performance for age recognition than overall accuracy. As shown in Table 3.5, the ResNet50 architecture achieved the best overall results, with an accuracy of 0.54 and a macro-average F1-score of 0.50. It performed especially well on the youngest age groups ('0–2': 0.86, '3–9': 0.66) and the most represented '20–29' category, where it achieved an F1 score of 0.71. However, its performance dropped in the remaining age groups, with F1 scores generally below 0.50. Despite this, it consistently outperformed the other models across almost all age brackets.



	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	19	2	0	0	5	0	19
Disgust	34	20	0	2	0	0	3
Fear	2	2	9	2	4	2	4
Happy	0	0	0	69	0	0	0
Sad	3	0	6	0	10	0	9
Surprise	0	0	4	0	0	78	1
Neutral	11	0	13	8	65	0	496

Fig. 3.12. Confusion Matrix for Model in [1] on CK+ dataset.



	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Angry	18	0	5	0	16	6	0
Disgust	24	27	0	3	4	1	0
Fear	0	1	11	2	4	4	3
Happy	0	0	0	69	0	0	0
Neutral	3	0	9	2	560	16	3
Sad	2	0	3	0	5	18	0
Surprise	0	0	1	0	1	0	81

Fig. 3.13. Confusion Matrix for Hybrid CNN-Transformer on CK+ dataset.

Table 3.5: Performance comparison of age recognition models.

Model Architecture	Overall				Class-wise F1 Scores								
	Acc	Prec	Rec	F1	0-2	3-9	10-19	20-29	30-39	40-49	50-59	60-69	70+
MobileNetV2	0.39	0.42	0.27	0.26	0.23	0.38	0.03	0.58	0.26	0.11	0.25	0.22	0.31
ResNet50	**0.54**	**0.56**	**0.49**	**0.50**	**0.86**	**0.66**	**0.41**	**0.71**	**0.34**	**0.32**	**0.39**	**0.35**	**0.48**
Proposed Hybrid	0.49	0.50	0.40	0.41	0.79	0.65	0.31	0.66	**0.34**	0.24	0.28	0.10	0.29

Note: Acc = Accuracy, Prec = Precision, Rec = Recall, F1 = F1-Score.

The proposed Hybrid model ranked second, with an accuracy of 0.49 and a macro-average F1-score of 0.41. It matched ResNet50 in the ’30–39’ category (F1 of 0.34) but showed lower F1 scores in most other age groups. MobileNetV2 yielded the lowest performance, with an accuracy of 0.39 and a macro-average F1-score of 0.26. It struggled in practically every age category, particularly ’10-19’, where it scored only 0.03 F1.

These results emphasize the difficulty of the 9-class age recognition task on the UTK Face dataset. Accurate predictions were mainly limited to the youngest age groups and the dominant ’20–29’ category. A deeper look into specific misclassification trends will be provided through the analysis of confusion matrices.

Analyzing the confusion matrix for the top-performing ResNet50 model (Figure 3.14) reveals complex error patterns beyond the overall metrics in Table 3.5. While performance is high in the ’0-2’ and ’20-29’ groups, major issues exist elsewhere. The predominant pattern seen is a systemic bias toward underestimating age for most groups older than ’3-9’. This is vividly visible for the ’10-19’, ’30-39’, and ’40-49’ brackets; huge numbers of these individuals (53%, 54%, and 21% correspondingly) are misclassified into the dominating, younger ’20-29’ category, possibly due to dataset imbalance. For individuals aged ’50-59’ and older, this under-estimation bias persists but typically manifests as misclassification into the immediately preceding (younger adjacent) age group (e.g., 48% of ’70+’ are predicted as ’60-69’). While errors into the adjacent older group also occur, they are generally less frequent than errors predicting a younger age, especially for those above 30. This suggests that after reaching young adulthood, the model has difficulty distinguishing between consecutive age groups and frequently tends to a younger classification.

The confusion matrix for the proposed Hybrid model in Figure 3.15) confirms its second place ranking in the Table 3.5 and, crucially, exhibits the same dominant pattern of age under-estimation bias described for ResNet50. It also struggles to discern older successive age groups and tends to misclassify people into younger brackets, such as the ’20-29’ category. Examining Recall, the Hybrid model scored slightly greater Recall particularly for the ’3-9’ age group (0.76) than ResNet50 (0.63), albeit this localized improvement had no meaningful effect on its overall F1-score relative to ResNet50 for that class.

Finally, the confusion matrix for MobileNetV2 in Figure 3.16, reflecting its lower overall performance, also displays the same pronounced under-estimation bias. Errors heavily favor predicting individuals as younger than their true age group, consistent with the patterns observed in the other, better-performing architectures. This reinforces that the under-estimation tendency is a pervasive challenge across models for this task on the UTK Face dataset.

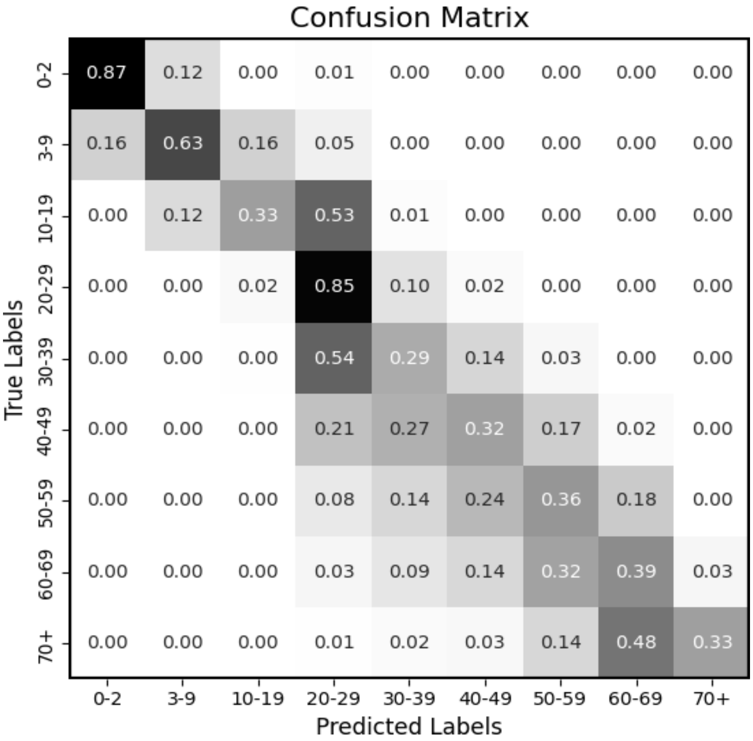


Fig. 3.14. Confusion Matrix for ResNet50 on UTK Face dataset.

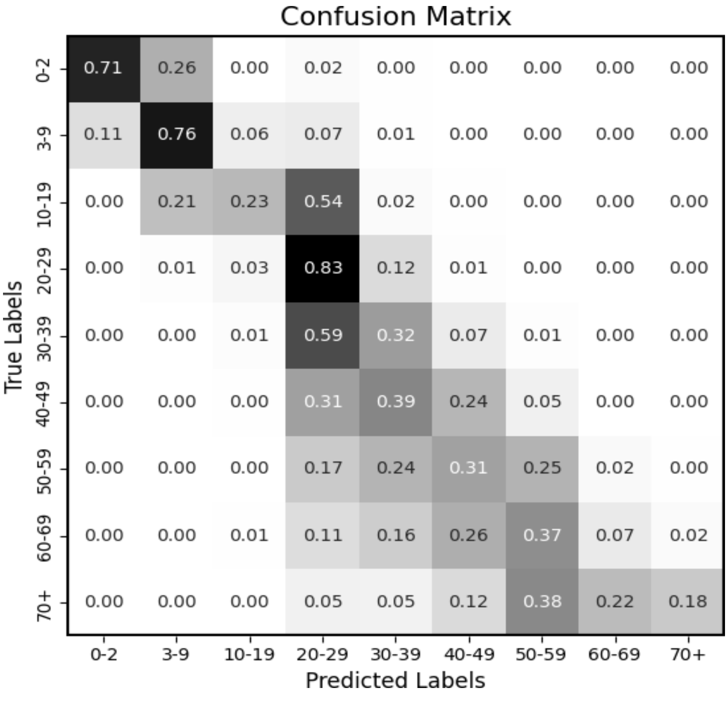
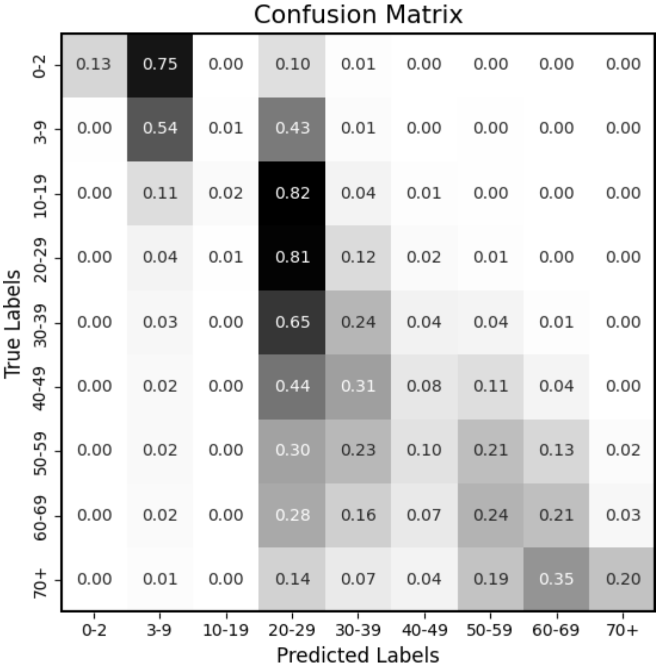


Fig. 3.15. Confusion Matrix for Hybrid CNN-Transformer on UTK Face dataset.

In summary, although ResNet50 delivered the strongest results, the confusion-matrix review for every model highlight just how challenging nine-class age estimation can be. All three architectures showed a clear tendency to underestimate age, especially for participants older than the youngest groups. Misclassifications were most typically placed in the next lower age bracket, and ages close to the dataset's modal 20-29 range were frequently assigned to that dominant category. This pattern underscores the inherent difficulty of discerning the subtle visual differences that separate adjacent, and particularly older, age groups.



	0-2	3-9	10-19	20-29	30-39	40-49	50-59	60-69	70+
0-2	0.13	0.75	0.00	0.10	0.01	0.00	0.00	0.00	0.00
3-9	0.00	0.54	0.01	0.43	0.01	0.00	0.00	0.00	0.00
10-19	0.00	0.11	0.02	0.82	0.04	0.01	0.00	0.00	0.00
20-29	0.00	0.04	0.01	0.81	0.12	0.02	0.01	0.00	0.00
30-39	0.00	0.03	0.00	0.65	0.24	0.04	0.04	0.01	0.00
40-49	0.00	0.02	0.00	0.44	0.31	0.08	0.11	0.04	0.00
50-59	0.00	0.02	0.00	0.30	0.23	0.10	0.21	0.13	0.02
60-69	0.00	0.02	0.00	0.28	0.16	0.07	0.24	0.21	0.03
70+	0.00	0.01	0.00	0.14	0.07	0.04	0.19	0.35	0.20

Fig. 3.16. Confusion Matrix for MobileNetv2 on UTK Face dataset.

3.2.3. Gender

Table 3.6 shows how each model fared on the UTK Face dataset's binary gender classification task. Because this dataset is fairly balanced between male and female samples, unlike the emotion and age tasks, overall accuracy becomes especially informative, complemented by precision, recall, and F1-score.

Table 3.6: Performance comparison of gender recognition models.

Model	Overall				Class-wise F1	
	Acc	Prec	Rec	F1	Male	Female
MobileNetV2	0.82	0.82	0.82	0.82	0.83	0.82
ResNet50	0.76	0.76	0.76	0.76	0.77	0.76
Proposed Hybrid	**0.91**	**0.91**	**0.90**	**0.90**	**0.91**	**0.90**

Note: Acc = Accuracy, Prec = Precision, Rec = Recall, F1 = F1-Score.

The proposed Hybrid CNN–Transformer model delivered the strongest overall performance, reaching an accuracy of 0.91 and precision, recall, and F1-scores of 0.90. It maintained an excellent balance between classes, with F1-scores of 0.91 for males and 0.90 for females. MobileNetV2 followed in second place, with all metrics hovering around 0.82. ResNet50 ranked third, with steady but lower scores of around 0.76 across the board.

The consistent alignment of accuracy, precision, recall, and F1-score across all used models emphasizes the balanced nature of the dataset for this particular task. While each architecture achieved commendable results, the proposed Hybrid model stood out with its notably superior and consistently reliable performance in gender classification. To visually support its high accuracy and balanced recognition, the confusion matrix for the Hybrid model is shown in Figure 3.17, revealing only a slight tendency to misclassify females as males more often than the reverse. Given the clear performance differences shown in Table 3.6 and considering the straightforward nature of this binary task compared to emotion and age recognition, confusion matrices for the other models are omitted for brevity.

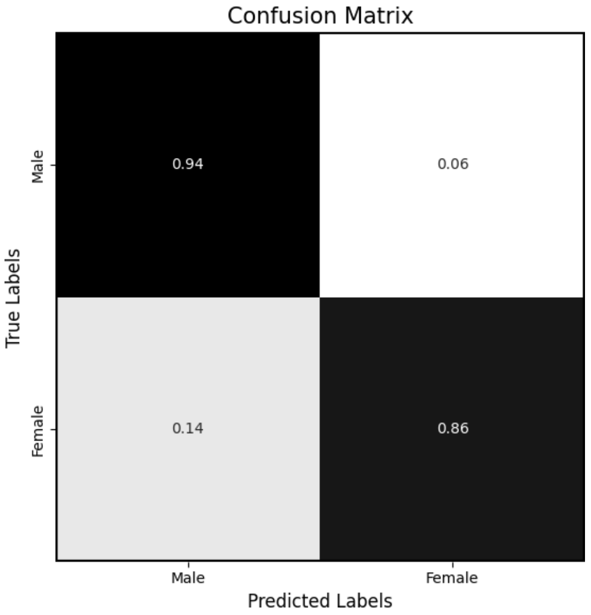


Fig. 3.17. Confusion Matrix for Hybrid CNN-Transformer on UTK Face dataset.

3.3.Computational performance on generalization datasets (Before optimization)

3.3.1. Emotions

Table 3.7 presents the inference times, FPS, and model sizes of various architectures in the emotion recognition task. As shown, the implementation by Seringel et al. [2] was the most computationally efficient, requiring only 34.70 ms per inference (equivalent to 28.81 FPS). While its model size is not reported here as the specific model file was not directly accessed, the low latency suggests a relatively compact design.

With a remarkably small footprint of only 4.3 MB and an inference time of 67 ms, Priyadarshini et al.'s [1] approach is the second most efficient model. The proposed Hybrid CNN-Transformer follows, with competitive performance at 129.3 ms (7.7 FPS) and a small size of 7.8 MB.

In contrast, standard transfer learning architectures show significantly higher computational demands: InceptionV3 requires 255.1 ms per inference, and ResNet50 slows to 776.3 ms (just 1.29 FPS). These models also exhibit substantially larger memory requirements, at 253.3 MB and 282.5 MB respectively.

Table 3.7: Comparison of inference performance and model size for emotion recognition models

Model Architecture	Inference time(ms)	FPS	Size (MB)
Seringel et al. [2]	**34.70**	**28.81**	N/A
InceptionV3	255.1	3.920	253.3
ResNet50	776.3	1.288	282.5
Priyadarshini et al. [1]	67.00	14.93	**4.300**
Proposed Hybrid	129.3	7.730	7.800

Overall, these baseline results clearly show that the models specifically designed or adapted for lightweight deployment (Seringel et al., Priyadarshini et al., Proposed Hybrid) indeed have sizes under 10 MB and achieve considerably faster inference times compared to the larger, standard pre-trained architectures like InceptionV3 and ResNet50 on the target hardware.

3.3.2. Age

The baseline computational performance for the architectures evaluated on the age recognition task is presented in Table 3.8. These measurements were carried out on the NVIDIA Jetson Xavier NX before optimization. As previously stated, MobileNetV2 achieved the maximum efficiency for this work, using only 63.8 ms per inference (15.67 FPS) and taking up only 9.1 MB of storage. The proposed Hybrid CNN-Transformer ranked second in terms of speed, with an inference time of 114.9 ms (8.70 FPS), although its model size was larger at 28.5 MB. ResNet50 exhibited the lowest computational performance, with the longest inference time at 470.5 ms (2.12 FPS) and the largest model size at 90.9 MB.

Table 3.8: Comparison of inference performance and model size for age recognition models

Model Architecture	Inference time(ms)	FPS	Size (MB)
MobileNetV2	**63.8**	**15.67**	**9.1**
ResNet50	470.5	2.12	90.9
Proposed Hybrid	114.9	8.7	28.5

3.3.3. Gender

Turning to the gender recognition task, the baseline computational performance of the evaluated architectures on the NVIDIA Jetson Xavier NX is detailed in Table 3.9. Leading the pack in efficiency was MobileNetV2, achieving the quickest inference time at 62.5 ms (equivalent to 15.97 FPS). It also maintained the smallest memory footprint at 9.1 MB. The proposed Hybrid architecture followed, requiring 110.9 ms per inference (9.02 FPS); while noticeably faster than ResNet50, it occupied 28.5 MB, similar to its configuration for the age task. In contrast, ResNet50 remained the least efficient option for gender recognition, with an inference time of 471.4 ms (2.12 FPS) and the highest storage requirement at 90.6 MB. The models' relative performance rankings are like those achieved for the age task, with only minor variations in individual timings.

Table 3.9: Comparison of inference performance and model size for gender recognition models

Model Architecture	Inference time(ms)	FPS	Size (MB)
MobileNetV2	**62.5**	**15.97**	**9.1**
ResNet50	471.4	2.12	90.6
Proposed Hybrid	110.9	9.02	28.5

3.4.Model selection for optimization

The Hybrid CNN-Transformer architecture was chosen for TensorRT optimization because it provided the optimal trade-off between all three tasks, combining strong predictive accuracy (Section 3.2) with solid baseline efficiency on the Jetson Xavier NX (Section 3.3). This architecture offered the most suitable trade-off, making it the primary candidate for developing an optimized, real-time perception system.

The emotion recognition task had a fairly straightforward selection procedure. The Hybrid model had the highest generalization accuracy and F1-scores (Table 3.4) while maintaining a moderate baseline computational cost compared to other high-performing alternatives (Table 3.7), making it the best choice for optimization.

In the case of age estimation, although the Hybrid model slightly underperformed ResNet50 in terms of F1-score (Table 3.5), it was significantly more efficient computationally at baseline (Table 3.8). Attempting to optimize the much slower ResNet50 to meet real-time inference constraints might necessitate aggressive optimization techniques (e.g., severe quantization, pruning) which could potentially degrade its accuracy substantially. Therefore, the Hybrid model represented a more practical starting point for achieving acceptable accuracy within feasible time limits post optimization.

Finally, for the gender recognition task, the choice was also clear. With emotion recognition, the Hybrid CNN-Transformer achieved the best evaluation metrics (Table 3.6) and maintained a favorable position regarding baseline inference time and model size compared to the alternatives (Table 3.9).

3.5. Optimized model

3.5.1. Emotion

Table 3.10 presents the performance metrics for the emotion recognition model before and after optimization with TensorRT FP16. The classification performance (accuracy, precision, recall, and F1-score) remained stable during optimization. However, there was a significant improvement in computational efficiency: the inference time was lowered from 129.3 ms to 6.18 ms, reflecting a 21x speedup. Consequently, the achievable frame rate increased to 161.81 FPS, and the model size nearly halved to 4.3 MB.

Table 3.10: Metrics for optimized architecture in the emotion task

Model Architecture	Accuracy	Precision	Recall	F1-Score	Time (ms)	FPS	Size (MB)
Proposed Hybrid	0.87	0.70	0.69	0.68	129.3	7.73	7.8
Proposed Hybrid Optimized	0.87	0.70	0.69	0.68	**6.18**	**161.81**	**4.3**

3.5.2. Age

Table 3.11 shows the metrics for the optimized age model. As with the emotion model, the performance metrics (accuracy, precision, recall, and F1-score) remain unchanged. However, there is a significant reduction in inference time, reaching just 4.47 ms, representing a speedup factor of around 25x, which allows the system to process up to 223.71 FPS. Additionally, the model size was reduced to only 14.8 MB.

Table 3.11: Metrics for optimized architecture in the age task

Model Architecture	Accuracy	Precision	Recall	F1-Score	Time (ms)	FPS	Size (MB)
Proposed Hybrid	0.49	0.50	0.40	0.41	114.9	8.7	28.5
Proposed Hybrid Optimized	0.49	0.50	0.40	0.41	**4.47**	**223.71**	**14.8**

3.5.3. Gender

The evaluation metrics for the optimized gender classification model are shown in Table 3.12. The inference time is significantly reduced to just 4.84 ms, indicating a speedup factor of about 23x, while maintaining the same performance metrics as the unoptimized version. This allows the system to handle up to 206.61 FPS, with a model size of only 14.8 MB.

Table 3.12: Metrics for optimized architecture in the gender task

Model Architecture	Accuracy	Precision	Recall	F1-Score	Time (ms)	FPS	Size (MB)
Proposed Hybrid	0.91	0.91	0.90	0.90	110.9	9.02	28.5
Proposed Hybrid Optimized	0.91	0.91	0.90	0.90	**4.84**	**206.61**	**14.8**

In general, the optimization process led to substantial improvements in both inference time and model size for all tasks. The total inference time was reduced to approximately 15.49 ms, allowing real-time processing at around 64 FPS.

3.6. Experiment results

3.6.1. Overall system performance in Real-Time

The experiment described in Section 2.7 was conducted using the NVIDIA Jetson Xavier NX. This section presents the most relevant performance metrics, focusing on inference times and the effectiveness of face detection.

Over the course of 252 experimental sessions, the system processed incoming images at an average frame rate of 11.57 FPS, with a standard deviation of 0.72 FPS. The performance of the face detection is summarized in Table 3.13. A total of 27,577 frames were analyzed, and a face was successfully detected in 22,490 of them, yielding a face detection rate of 81.5%. The remaining 5,087 frames, in which no faces were detected, were excluded from the subsequent analysis.

Table 3.13: Summary of Face Detection Performance During Experiment

Metric	Value
Total Frames with Detection Attempted	27,577
Frames with Face Detection Failure	5,087
Frames with Face Detection Success	22,490
Face Detection Success Rate (%)	81.5

As a result, the following classification performance analyses are based on the 22490 frames in which the models correctly spotted and processed a face.

3.6.2. Session-Level System Reliability

Beyond evaluating frame-by-frame accuracy, the reliability of the system during continuous, short interaction periods was assessed to understand its consistency. For this analysis, each of the 252 experimental sessions was evaluated on a binary basis. A session was defined as ’error-free’ (assigned a score of 100%) only if no misclassifications were logged by the system, for emotion, age, or gender, across all successfully processed frames within that session. Conversely, if one or more errors occurred in any of the three tasks during the session, it received a score of 0%.

The mean session-level reliability, obtained by averaging this binary score across all 252 sessions, was 10.5%, with a standard deviation of 18.44%. This result clearly indicates that, in 89.5% of the experimental trials, the system registered at least one perception error in one or more of the tasks being performed. Errors were most observed in emotion and age classification, while gender errors were less frequent.

3.6.3. Detailed emotion recognition performance (Frame level)

Table 3.14 shows the detailed classification metrics for each emotion class used in the experiment. In general, the distribution of emotions, indicated by the Support column, is relatively balanced.
However, the overall performance was low, with a global accuracy of only 0.232 and a macro average F1-score of just 0.166.
Disgust was completely unrecognized, with an F1-score of 0.0. Surprise performed poorly, with an F1-score of 0.027, as did Fear, which achieved only 0.093. Angry did slightly better but still had a low F1-score of 0.141. Sad performed somewhat better, with an F1-score of 0.213. Neutral and Happy were the best-performing classes in terms of F1 scores, with 0.302 and 0.383, respectively. Their significantly higher recall ratings, 0.498 for Neutral and 0.741 for Happy, imply that they were the most correctly identified classes among all the frames investigated representing those real emotions.

Table 3.14: Classification Report for Emotion Recognition (Experiment)

Emotion class	Precision	Recall	F1-Score	Support
Angry	0.277	0.095	0.141	3540
Disgust	0.000	0.000	0.000	3278
Fear	0.166	0.065	0.093	3154
Happy	0.258	0.741	0.383	3181
Neutral	0.216	0.498	0.302	3125
Sad	0.198	0.232	0.213	3075
Surprise	0.431	0.014	0.027	3137
Accuracy			**0.232**	22490
macro avg	0.221	0.235	0.166	22490

The analysis of the corresponding confusion matrix in Figure 3.18, which is row-normalized to show recall percentages, reveals specific error patterns. Overall, there is a strong tendency for many emotions to be misclassified into the Happy, Neutral, or Sad categories, as indicated by the high values in those columns for most rows.

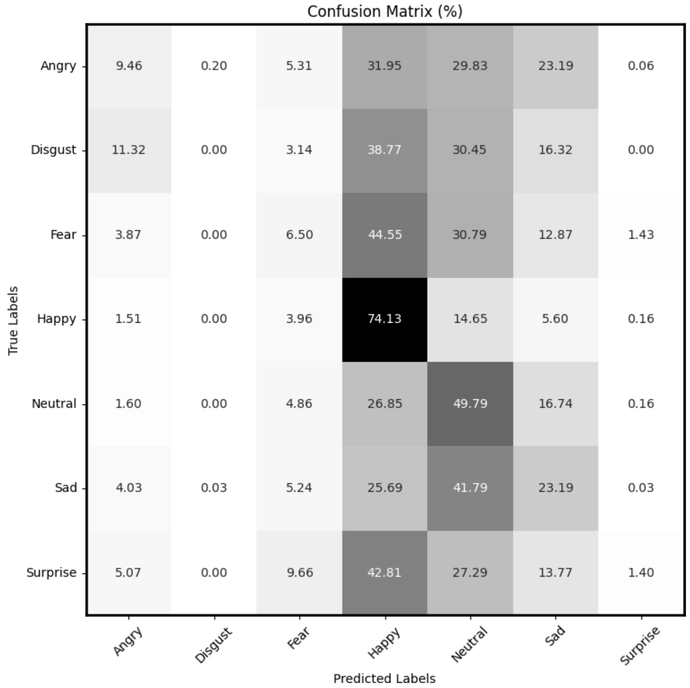


Fig. 3.18. Confusion Matrix for emotion in the experiment.

Although they are not the most common, some misunderstandings are significant. A common misunderstanding between these two negative emotions is demonstrated by the fact that 11.32% of disgust frames were incorrectly classified as angry. Similarly, 9.66% of Surprise instances were incorrectly predicted as Fear, indicating some shared features perceived by the model. These specific confusion pairs, along with the general tendency toward Happy, Neutral, and Sad predictions, contribute to the low overall performance metrics observed for emotion recognition in the experiment.

3.6.4. Detailed age recognition performance (Frame level)

Table 3.15 presents the classification report for age recognition in the experiment. The overall performance metrics show an accuracy of 0.48 and a low macro average F1-score of 0.248. A notable imbalance exists in the dataset, with 18,249 frames corresponding to the 20–29 age group. Despite this imbalance, the model has a relatively high F1-score of 0.668 in this class, indicating good performance. Other age groups, such as 30-39, 40-49, and 60-69, had F1 scores below 0.3. The poorest result was seen in the 10-19 age group, with an F1-score of only 0.031.

Table 3.15: Classification Report for Age Recognition (Experiment)

Age group	**Precision**	**Recall**	**F1-Score**	**Support**
10-19	0.138	0.017	0.031	1503
20-29	0.840	0.554	0.668	18249
30-39	0.073	0.219	0.109	1430
40-49	0.165	0.421	0.237	639
60-69	0.506	0.121	0.195	669
Accuracy			**0.48**	22490
macro avg	0.344	0.266	0.248	22490

The confusion matrix in Figure 3.19, which is row-normalized to reflect recall percentages, clearly illustrates the performance issues discussed earlier. As seen in similar experiments with the UTKFace dataset, there is a strong tendency for the model to confuse age groups, especially by predicting neighboring ages or defaulting to the most common group, 20–29. For example, the majority of instances in the 10–19 and 30–39 age groups were incorrectly classified as 20–29, with 93.07% and 56.42% of cases, respectively. Likewise, most of the 40–49 age group was misclassified as 30–39, which is the closest lower age range. The age group 20–29 demonstrated relatively strong performance, achieving a recall rate of 68.83%. In contrast, the model encountered difficulties in accurately predicting outcomes for other age ranges, particularly those with limited sample sizes.

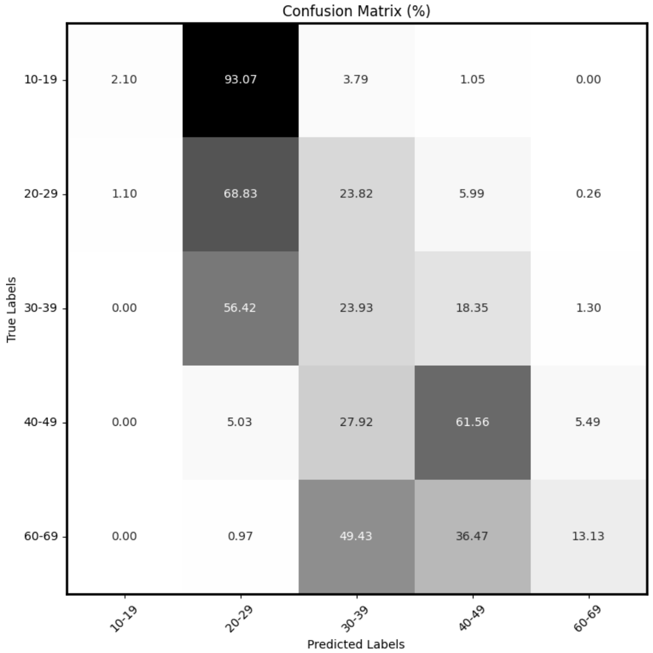


Fig. 3.19. Confusion Matrix for age in the experiment.

3.6.5. Detailed gender recognition performance (Frame level)

Table 3.16 presents the classification report for the gender recognition task. The data is skewed, with just 4,318 frames tagged female and 18,172 as male. Despite this imbalance, the model had a good overall accuracy of 0.878 and a strong macro-average F1-score of 0.834. The male class performed better in recognition, with an F1-score of 0.919, while the female class did well, with an F1-score of 0.749. These results indicate that the model performed well overall in the gender recognition task.

Table 3.16: Classification Report for Gender Recognition (Experiment)

Gender	Precision	Recall	F1-Score	Support
Female	0.617	0.953	0.749	4318
Male	0.987	0.860	0.919	18172
Accuracy			**0.878**	22490
macro avg	0.802	0.906	0.834	22490

Figure 3.20 presents the row-normalized confusion matrix for the gender recognition model during the experiment, reflecting the generally strong performance indicated by the classification metrics. The matrix visually confirms the model’s effectiveness, particularly for identifying females (Recall = 95.3%). While 4.72% of female instances were incorrectly projected as male, 14.0% of male instances were misclassified as female, demonstrating an imbalance in the mistakes. This implies that the model’s robustness in classifying male participants was marginally lower than that of female participants in the experiment.

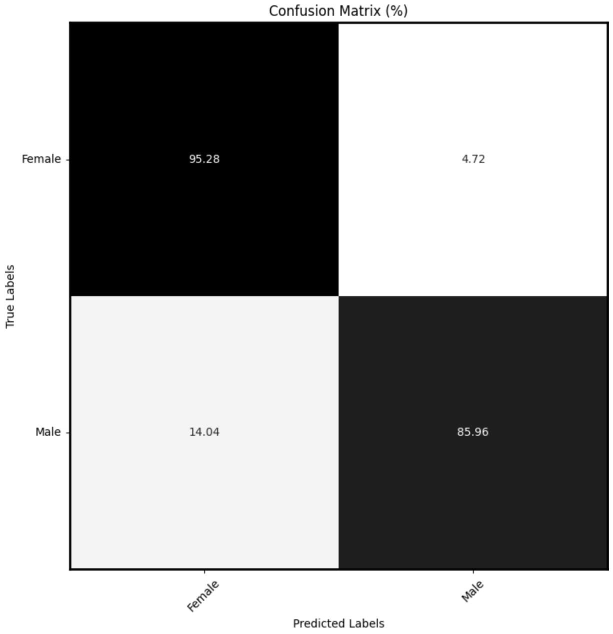


Fig. 3.20. Confusion Matrix for gender in the experiment.

3.6.6. Error analysis: Influence of head pose

To investigate the system's robustness to variations in head orientation during the experiment, the distribution of the 20,208 logged error instances was analyzed across different estimated head poses. Due to the data logging strategy focusing on errors, this analysis cannot determine absolute error rates per pose but examines where errors occurred most frequently and whether specific tasks showed different sensitivities. Combinations of vertical ('up', 'center', 'down') and horizontal ('left', 'center', 'right') orientation were used to classify the poses.

First, the overall distribution of head poses among all logged errors was examined to identify the most common orientations during system failures. The majority of errors occurred when participants held their heads in the central-horizontal position. The most common positions were center-right at 35.22%, followed by center-center at 25.10%, and center-left at 23.34%. These three orientations accounted for 84% of observed errors, indicating that individuals frequently chose these stances during error-prone trials. In contrast, poses involving vertical movement, such as gazing up or down, were significantly less common among the errors, accounting for only approximately 16%. This distribution reflects both participants' natural preferences to keep a level gaze and the system's performance in various head positions.

Next, as shown in Table 3.17, the distribution of errors for each task (Emotion, Age, and Gender) was examined across the different head poses.

Table 3.17: Granular Distribution of Errors by Exact Pose (%)

Pose (Vertical-Horizontal)	Emotion Errors (%)	Age Errors (%)	Gender Errors (%)
center-center	25.08	24.57	13.94
center-left	22.62	24.40	26.50
center-right	35.32	36.13	40.73
down-center	2.95	1.87	2.65
down-left	1.02	0.83	1.16
down-right	2.12	1.46	1.38
up-center	4.74	5.40	3.45

up-left	0.99	1.08	1.45
up-right	5.16	4.24	8.75

tAe task-specific error distributions in Table 3.17 with the overall pose distribution among all errors reveals different levels of sensitivity across tasks.

Regarding age estimation, the distribution of errors closely corresponded with the overall distribution of head poses. This indicates that the model maintained consistent performance across different angles of head orientation. Fundamentally, the errors primarily reflected how frequently each pose was represented in the dataset, rather than being greatly influenced by specific viewing angles.

A similar pattern, yet with slight variations, was found in the analysis of emotion recognition. For instance, compared to their overall frequency of 4.83%, upright poses showed a marginally higher percentage of emotional error of 5.16%. Pose alignments closer to the center-left, on the other hand, resulted in slightly fewer errors, measuring 22.6% as opposed to the overall 23.3%. Despite these small differences, there is no strong indication that particular head poses had a significant impact on emotion recognition performance.

In contrast, gender classification displayed a more distinct pattern. Errors in this task were noticeably less frequent for center-center poses, which made up only 13.9% of gender-related errors, even though this pose represented 25.1% of the overall error distribution. In the meantime, poses such as center-left and center-right contributed more considerably to gender recognition errors, 26.5% and 40.7% respectively, compared to their overall proportions of 23.3% and 35.2%. The upright pose was particularly notable, with gender error rates reaching 8.75%, nearly double its representation in the dataset. These patterns indicate that the gender recognition model tended to encounter more difficulty when analyzing images from non-frontal angles, especially when the subject's head was turned sideways or tilted slightly upward.

To sum up, the analysis of pose-specific error patterns shows that the robustness to head orientation varies among activities. Position changes appear to have the least impact on age estimation, but gender classification was noticeably more sensitive, particularly when faces were not facing the camera.

3.6.7. Error analysis: Influence of demographics

This section looks at how perception errors can vary across different demographic groups of the participants, considering the known sample imbalances noted in Section 2.7. The goal is to identify potential biases or differential performance related to these demographic factors within the experiment.

The percentage of participants from each age group and the percentage of logged errors coming from those same groups are compared in Table 3.18. The variation in age estimation errors is a significant finding. Only 69.6% of age errors were attributable to the dominant '20-29' group, indicating a lower error rate for this demographic. On the other hand, participants from underrepresented groups made disproportionately larger contributions: 12.6% of age errors were attributed to the '10-19' group, 9.6% to the '30-39' group, and 5.0% to the '60-69' group. This suggests that participants outside of the primary '20-29' age range in this study had a noticeably higher age estimation error rate. In general, the distribution across age groups more closely reflected the participant distribution for gender and emotion errors.

Table 3.18 Error Distribution vs. Participant Distribution by Age Group (%)

Real Age Group	% of Participants	% Emotion Error	% Age Error	% Gender Error
10-19	5.6	6.34	12.64	1.13
20-29	83.3	80.86	69.61	78.77
30-39	5.6	6.16	9.56	7.01
40-49	2.8	3.14	3.17	5.81
60-69	2.8	3.50	5.03	7.30

Similarly, Table 3.19 compares the participant gender distribution to the distribution of errors for each task. While the distribution of emotion and age errors closely matched the participant gender ratio, a significant asymmetry emerged for gender classification errors. Female participants accounted for only 7.4% of gender errors. In contrast, male participants were associated with a disproportionately high 92.6% of gender errors. This strongly suggests that, within this experimental context, the model exhibited a considerably higher error rate when classifying male participants compared to female participants.

Table 3.19: Error Distribution vs. Participant Distribution by Gender Group (%)

Real Gender	% of Participants	% Emotion Error	% Age Error	% Gender Error
Female	19.4	18.82	15.89	7.4
Male	80.6	81.2	84.1	92.6

To sum up, the analysis showed that different aspects of the task were affected by the participants' demographics. Those outside of the typical 20–29 age range had much worse age estimation accuracy. When it came to gender classification, there was a clear bias, meaning the system was less accurate for males than females, even though most participants were male. On the other hand, recognizing emotions seemed less impacted by the age or gender of participants. These results emphasize how the characteristics of the sample can influence the outcomes and suggest potential biases in the model.

3.6.8. Error analysis: Influence of image quality

To explore how image quality affected various error types, the distributions of luminosity, contrast, and blur metrics in the logged error instances were analyzed. Violin plots were then used to display these distributions for each category (Emotion, Age, Gender), providing a more nuanced comparison than simple averages.

Across all three categories of errors, the luminosity distribution in Figure 3.21 seemed to be remarkably consistent. Indicating that changes in lighting intensity, within the range observed during the experiment (primarily medium-to-high brightness, as indicated by the distribution centered around 168), did not differentially influence whether an emotion, age, or gender error occurred, the violins' shape, median, and inter quartile range (IQR) were nearly identical.

Similarly, the analysis of the blur level metric in Figure 3.22, likely representing Laplacian variance where higher values indicate sharper images, showed almost identical distributions for Emotion, Age, and Gender errors. The violins share the same basic shape, concentrating data in a range indicating moderate-to-good sharpness, with a long tail towards very sharp images. This suggests that the level of image focus was not a distinguishing factor between the different types of perception errors logged.

The contrast distributions in Figure 3.23 showed a slight possible difference. The median and IQR for gender errors seemed to be somewhat lower than those for emotion and age errors, despite the fact that the general shapes were similar. Because the distributions still displayed significant overlap, this may indicate a weak tendency for gender misclassifications to be linked to slightly lower contrast images within this dataset.

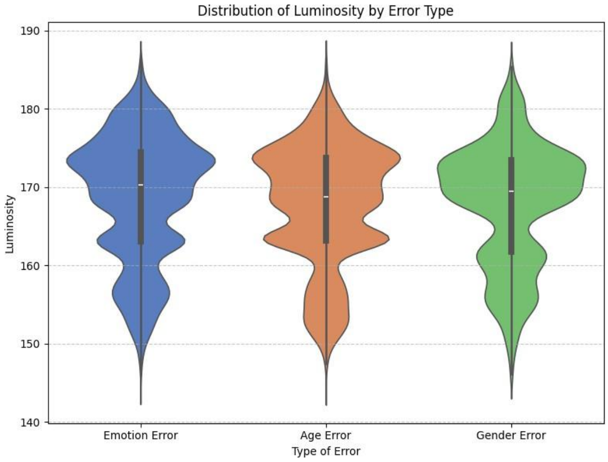


Fig. 3.21. Luminosity violin plot by error type.

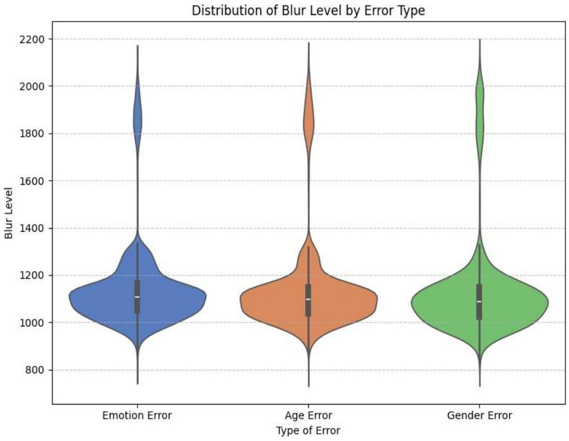


Fig. 3.22. Blur level violin plot by error type.

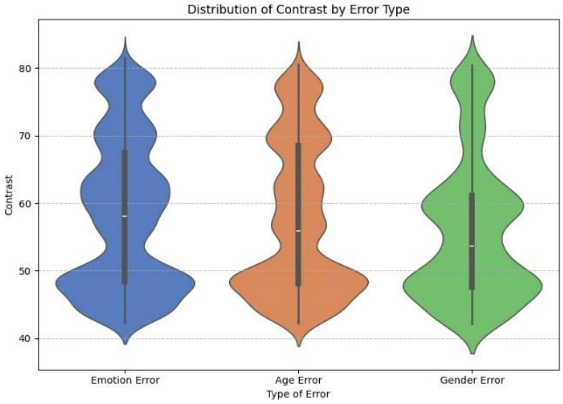


Fig. 3.23. Contrast violin plot by error type.

In summary, the distributional analysis using violin plots largely indicates that the measured image quality metrics (luminosity, contrast, blur level) did not strongly differentiate between the types of perception errors encountered during the experiment. Despite a slight correlation between lower contrast and gender errors, the largely similar distributions indicate that image quality variations were unlikely to be the main factor in determining which task failed. Instead, aspects like head pose or the inherent difficulty of each task likely played a more significant role.

4. Discussion

4.1. Summary and interpretation of key findings from training and validation

Analyzing the performance on the validation sets during model training provides initial insights into the capabilities and limitations of the evaluated architectures for each task.

One of the main observations regarding emotion recognition on the FER2013 validation set is that the task is inherently challenging because it relies just on facial appearance. Out of all the models tested, the suggested Hybrid CNN-Transformer had the highest validation accuracy, but it only reached 67%. This raises the possibility of data set limitations or the difficulty of extracting features that are sufficiently discriminative. Additionally, training observations revealed overfitting or stagnation tendencies across different architectures, suggesting that it is difficult to capture strong, broadly applicable features for all seven emotions. Although optimizing hyperparameters such as kernel sizes, learning rate, or the amount of regularization, or extending the training time may offer improvements, the small amount of data in classes like Disgust, Fear, and Angry will likely impose a limit on validation performance and on the model's ability to learn these categories.

Similarly, the FairFace validation set yielded moderate age estimates. The Hybrid model outperformed others, including MobileNetV2, with a validation accuracy of 57%. However, there is still potential for improvement. This limited validation accuracy indicates potential flaws in the feature extraction technique for capturing the nuanced and variable properties associated with age groups, especially considering the FairFace dataset's acknowledged imbalance between age categories. Ad- dressing this data imbalance, as well as potentially investigating more specialized architectures or targeted hyperparameter tweaking, likely to be critical for significant progress in this endeavor, even before addressing generalization.

In contrast, gender recognition on the FairFace validation set yielded much stronger results. The Hybrid CNN-Tranformer model achieved 92.5% validation accuracy, indicating good competence for this binary classification problem. The considerable gap between the best and worst performers, down to 69%, demonstrates architectural decisions matter. However, the models were better suited to extract discriminative gender traits from the FairFace data compared to the more difficult emotion or age tasks. Even so, the fact that the best model's training seemed to stall suggests that there might still be room for improvement, either by fine-tuning it more or training it for longer.

These initial findings from the validation phase set the stage for evaluating how well these models generalize to unseen datasets and perform under real-time experimental conditions

4.2. Summary and interpretation of key findings from generalization datasets

When evaluating the trained models on unseen generalization datasets (CK+ for emotion, UTKFace for age/gender), several trends were confirmed, and key challenges became more apparent. As anticipated, architectures that performed well during validation, such as the Hybrid CNN-Transformer, generally maintained their leading performance on these datasets, suggesting that the learned features were effectively transferred despite existing limitations.

In the case of emotion recognition on the CK+ dataset, the Hybrid model performed the best overall, but significant issues persisted, notably with emotions that were either underrepresented in the training set or visually ambiguous. As shown in Table 3.4, recognition rates for emotions such as Fear (F1 = 0.41), Angry (F1 = 0.39), and Sad (F1 = 0.49) were relatively low. The frequent confusion pairs, such as disgust being mistakenly classified as angry and sadness being mistaken for neutral, show how difficult it is to distinguish some emotions based only on static facial features (Figure 3.13).

Unbalances in the dataset most likely made these misclassifications worse. Yet, the model's impressive performance on emotions like happiness and neutral indicates that it was able to comprehend some significant expressive traits, such as smiling.

A similar pattern of difficulty emerged for age estimation on UTKFace. All models suffered, presumably due to skewed FairFace training data. In Table 3.5, the results were clearly dominated by the '20-29' age group, which not only had the highest F1 score (0.71 for ResNet50, 0.66 for Hybrid), but also was the main target for misclassification for nearby groups such as '10-19' and '30-39'. This reinforces the previously observed underestimation bias and highlights the critical need for architectures or techniques capable of capturing the subtle facial differences distinguishing adjacent age brackets more effectively.

With the Hybrid model achieving an F1 score of 0.90, the gender recognition task on the UTKFace dataset was completed with high accuracy. This implies that separating gender-related characteristics was reasonably simple, and it supports the decision to use the Hybrid model for optimization due to its superior predictive performance over the faster but less accurate MobileNetV2.

Finally, the computational performance on the Jetson Xavier NX prior to optimization starkly contrasted lightweight versus complex models in Tables 3.7,3.8,3.9. Architectures like ResNet50, while sometimes accurate, were prohibitively slow, whereas custom lightweight models offered speed but often compromised on generalization accuracy. This confirmed the expected trade-off and motivated the selection of the balanced Hybrid model for subsequent optimization.

4.3. Summary and interpretation of key findings from optimization

The application of TensorRT FP16 optimization to the chosen Hybrid CNN-Transformer architecture resulted in significant increases in computational efficiency while keeping the same level of performance, as described in Section 3.5. Inference times for the emotion, age, and gender tasks were drastically reduced by factors ranging from approximately 21× to 26× compared to the baseline, while model sizes were also significantly decreased, as shown in Tables 3.10, 3.11, and 3.12.

These outcomes highlight how essential post-training optimization is for running deep learning models on resource-limited edge platforms like the Jetson Xavier NX, and they showcase the performance gains unlocked by TensorRT's FP16 mode. It should be noted, however, that only FP16 optimization was evaluated here; additional improvements might come from techniques such as INT8 quantization or model pruning, which lie beyond the scope of this analysis.

An interesting and somewhat counter-intuitive finding emerged when comparing the optimized inference times across tasks. Although the Hybrid model adapted for age and gender recognition processes larger input images (224x224) compared to the emotion recognition variant (96x96), the optimized versions for age of 4.47 ms and gender of 4.84 ms achieved faster inference speeds than the optimized emotion model of 6.18 ms. This occurred despite the final optimized model size for emotion of 4.3 MB being smaller than for age/gender of 14.8 MB.

These results show that, in addition to input dimensions and overall model size, other factors have a significant impact on post-optimization performance on target hardware. Variations in internal layer configurations, the specific convolutional operations employed, and the degree to which those operations benefit from TensorRT's FP16 optimizations likely explain why architectures built for larger inputs ultimately delivered faster inference engines in this scenario.

4.4. Summary and interpretation of key findings from optimization

The real-time experiment provided relevant insights into the system's performance under changing conditions. Starting with aspects of its procedure, the face detection phase achieved a success rate of 81.5% as shown in Table 3.13. Although functional, due to the 19% failure rate, it can be improved with more robust detectors like Multi-task Cascaded Convolutional Neural Networks (MTCNN), as it does not lose data from difficult poses and is also a highly reliable detector in various environments. Furthermore, the session-level reliability analysis revealed that errors occurred in 89.5% of the short interaction sessions in Section 3.6.2. This highlights the considerable challenge of maintaining continuous, error-free multi-task perception, primarily driven by frequent errors in emotion and age classification during the trials.

Examining the frame-level performance for emotion recognition, the results were significantly lower than those observed on the controlled CK+ dataset, yielding an overall accuracy of only 0.23 and a macro F1-score of 0.17 in Table 3.14. This decrease in performance highlights the disconnect between dynamic, interactive scenarios and controlled datasets. Despite sufficient representation in the experiment itself, the model had considerable difficulty identifying some negative or ambiguous emotions, such as anger and sadness, which is consistent with offline findings and probably reflects difficulties from the FER2013 training data. Disgust, Fear, and Surprise all performed terribly, with Disgust most likely being severely underrepresented in the training set.

Figure 3.18, the confusion matrix, makes it clear that the emotion in the model finds hardest to recognize tend to get lumped in as Happy, Neutral, or Sad. Even Happy, which scored the highest recall at 0.74, only managed a precision of 0.26, hinting that the model might be seeing smiles where there are not any (for example, mistaking visible teeth for a genuine grin). Taken together, these findings underscore the need for richer feature-extraction methods that can pick up on subtler facial cues, as well as more balanced training data to shore up those weaker emotion classes.

Regarding the model's sensitivity to pose, image quality, and demographic variations, no direct correlations were found that would indicate these factors as clear causes of prediction errors, except for a slight trend associated with the up-right facial position. This implies that training the model on a real-world dataset does enhance its robustness, making it less vulnerable to these types of fluctuations.

With a similar overall accuracy of 0.48 and a macro F1-score of 0.25, the model's performance in the age estimation task during the experiment closely matched the pattern seen in the offline evaluations using the UTKFace dataset, as shown in Table 3.15. The model significantly struggled with the other age ranges, but as anticipated, it did best on the '20–29' age group, which was the most represented category in the data. The underestimation bias seen earlier in Figure 3.19 is probably reflected here as well. Although the model appears to have been able to generalize its limited learning, as evidenced by the consistency between offline and real-time results, the low overall metrics underscore the continued need for improved model architectures and more balanced training datasets, particularly when addressing the intricacy of fine-grained age prediction.

Similar to the emotion recognition model, the age estimation errors did not show any strong correlation with head pose, image quality, or demographic attributes. This suggests that training the model on an in-the-wild dataset contributed positively to its robustness against these types of variations.

The model performed effectively in the gender recognition task, obtaining an F1-score of 0.83 and an accuracy of 0.88. However, a deeper analysis, based on the classification report (Table 3.16) and the demographic error breakdown in Section 3.6.7, revealed a more nuanced picture in terms of bias.

Although the model's precision for the Female class was relatively low (0.617), suggesting that male faces were often misclassified as female, the broader error analysis showed that most gender misclassifications actually involved male participants: 92.6% of all errors came from this group, which represented 80.6% of the participants. This shows that while the model tends to systematically predict Female when uncertain, it ends up making more incorrect predictions when the true gender is Male. In the context of the experiment, this bias against correctly identifying male faces stands out and contrasts with some patterns seen in offline evaluations, suggesting that more research is necessary.

Unlike the models used for emotion and age recognition, the gender classification model showed a noticeable pattern in its errors, which seemed to be linked to certain head orientations. Specifically, lateral poses (facing left or right) and top-right angles were more likely to result in misclassifications. This implies that the model is potentially sensitive to changes in viewpoint, and that adding more training data with these particular angles could help increase the model's accuracy and resilience in practical situations.

Furthermore, there is evidence that contrast levels may influence the model's decisions. In certain circumstances, lesser contrast appears to result in misclassification of female faces as male. This limitation can be mitigated by applying data augmentation techniques that simulate varying contrast levels or by enhancing the model's generalization capabilities to perform reliably under such conditions.

Overall, head pose, and image quality did not emerge as major factors contributing to model errors. However, it is important to note that there were considerably fewer frames with extreme vertical head orientations, suggesting that spending more time in these positions during data collection could help capture potential correlations. In the same way, the dataset's range of contrast, brightness, and blur level was constrained, even though no significant correlations were found. Therefore, it cannot be said with certainty that these factors have no effect on model performance; rather, it can be said that no discernible effect was found within the observed ranges.

Finally, the post-training optimization proved effective. In all experimental sessions, inference times were consistent with those measured on the benchmark datasets, confirming that real-time inference was successfully achieved.

4.5. Comparing with existing literature

There are both connections and gaps when the results are positioned within the body of existing literature. There are few direct comparisons of multi-task, real-world facial analysis deployments on edge hardware such as the Jetson Xavier NX, especially for gender and age recognition. However, context is provided by pertinent optimization studies and benchmarks.

For emotion recognition, Pascual et al [44] offers a useful comparison, as they optimized an Xception model for the Jetson Nano. Their accuracy on FER2013 of 69.9% is fairly close to the validation accuracy of the Hybrid CNN-Transformer model in this study of 67%. However, the computational performance is where the real difference shows, mainly due to the hardware differences. While their model ran at 5.5 FPS on the resource-limited Jetson Nano, the optimized Hybrid model in this study hit 161.8 FPS on the much more powerful Jetson Xavier NX. Even though it is tricky to make a direct comparison because of the hardware differences, this result really highlights the impressive performance potential of the proposed optimized model on the Xavier NX platform.

In age and gender recognition, comparisons can be made to the original FairFace benchmarks [53] obtained using ResNet-34. That work reported slightly higher accuracies (e.g., 0.57 vs. 0.49 for Age on UTKFace; 0.94 vs. 0.91 for Gender on UTKFace, compared to the Hybrid model's baseline generalization results). However, architectures like ResNet-34 typically incur substantial computational costs, likely comparable to the slow baseline ResNet-50 evaluated in this work. On the other hand, the optimized Hybrid model provided inference in 4-5 ms, albeit at the expense of a slight offline accuracy margin (Tables 3.11-3.12). The optimized Hybrid model is a much more practical option for real-time HRI applications because of this enormous speed difference, which highlights the crucial trade-off between deployment viability on edge devices and peak accuracy.

This study brings into sharp relief the necessity of evaluating models on the hardware for which they are intended. Performance figures derived only from offline datasets or powerful server clusters often paint an unrealistically optimistic portrait that rarely translates well to low-power, resource-limited devices. By incorporating hardware specific benchmarks into the evaluation process, researchers obtain a realistic gauge of model efficiency and reliability, an essential step for ensuring that edge AI solutions will perform as expected in real-world environments.

4.6. Contributions of this thesis

This thesis provides several contributions to the field of facial analysis for Human-Robot Interaction on edge devices:

1. Comprehensive Comparative Analysis: A systematic comparison of diverse deep learning architectures, including established baselines (Seringel et al. [2], Priyadarshini et al. [1]), standard transfer learning models (InceptionV3, ResNet50, MobileNetV2), as well as a proposed Hybrid CNN-Transformer, were used to recognize emotion, age, and gender. Performance was evaluated not only on standard validation and generalization datasets (FER2013, FairFace, CK+, UTKFace) but also considering baseline computational efficiency (latency, FPS, size) on the target NVIDIA Jetson Xavier NX platform. This multi-faceted comparison provides valuable insights into the accuracy-efficiency trade-offs inherent in selecting models for edge deployment.
2. Hybrid Architecture Performance: The proposed Hybrid CNN-Transformer architecture results as a particularly effective solution, demonstrating a strong balance between competitive predictive accuracy across all three tasks, notably leading in emotion and gender generalization, and much superior baseline computational efficiency than deeper models such as ResNet50. Its choice demonstrated the possibility of integrating attention-based and convolutional methods for multi-task facial analysis on hardware with limited resources.
3. Quantified Optimization Impact on Jetson Xavier NX: The work details the successful application and impact of NVIDIA TensorRT optimization using FP16 precision for the selected Hybrid model on the Jetson Xavier NX. Specific, quantified results demonstrate substantial improvements: inference times were reduced by factors of 21x (Emotion), 26x (Age), and 23x (Gender), achieving individual task speeds of 4-6 ms, while critically preserving the original predictive accuracy (Accuracy, F1-Score) for all three tasks. Model sizes were also significantly reduced. This provides concrete evidence of achieving high-performance, real-time inference capabilities (64 FPS combined throughput) for complex facial analysis on this specific edge platform.
4. Experimental Insights into Real-Time Robustness: The 36-person experiment with different head positions gave important new perspectives on the real-world behavior of the optimal system. Although the analysis revealed differential robustness across tasks regarding pose variations (gender being most sensitive) and underlined the impact of participant demographics (age estimation and gender classification biases), it confirmed the challenges in maintaining perfect continuous accuracy (low session-level reliability). This thorough error analysis helps one to grasp the pragmatic difficulties beyond the evaluation of a stationary dataset.

4.7. Study limitations

This work led to the identification of several limitations. First, there was a lack of access to training data, including larger datasets like AffectNet, which are frequently used because of their resilience in emotion recognition. As a result, class imbalances were evident in the datasets (FER2013, FairFace), especially in younger age groups and emotions like fear and disgust. This created limitations on how well the models could generalize.

Secondly, hardware and software constraints influenced the methodology. The use of Google Colab's free tier for training reduced the practicality of computationally costly data augmentation strategies, especially for the large FairFace dataset. Even though the Jetson Xavier NX is a powerful target platform, using it required optimization using NVIDIA's TensorRT. This made it impractical to investigate other optimization frameworks that might provide different trade-offs in terms of performance.

Thirdly, the experimental design clearly shows limits that influence the extent of application for the real-time evaluation results. With a notable demographic imbalance and a clear bias toward male participants (80.6%) and those in the "20-29" age group (83.3%). This greatly lessens the validity of performance conclusions derived for underrepresented groups. Moreover, the experiment depended on acted emotional expressions instead of naturalistic ones, which reduced the ecological validity of the interactions under observation even in naturalistic surroundings. A definitive analysis of the effects of illumination was also impossible due to the experiment's timing, which resulted in little variation in natural lighting conditions.

Finally, regarding experimental conditions, the range of head poses naturally adopted by participants during the experiment, while varied, lacked sufficient instances of extreme vertical tilt (pitch). This limits the evaluation of the system's resilience to the whole spectrum of possible head orientations encountered in dynamic HRI situations. Analogous to this, the observed range for measures of image quality such as contrast and blur was rather small, hence possible sensitivities outside these ranges could not be assessed.

These limitations highlight areas where the current findings should be interpreted with caution and suggest directions for future research to build upon this work.

4.8. Future work

Building on the study's findings and limitations, there are several intriguing areas for further research. One of the most exciting next steps is to run experiments that truly mirror real-world conditions.

Future research should use emotionally charged video clips as stimuli to record more organic, unplanned emotional expressions. To better understand how well the system generalizes across different persons, a more gender and age-balanced set of participants is required.

One suggestion would be to control the lighting to test the model's sensitivity in various scenarios and incorporate dynamic camera movements throughout the session to mimic various head poses. Another important point is to go deeper into qualitative analysis of the errors, looking at the specific segments where the model failed could help spot subtle visual cues like ambiguous expressions, micro-expressions, or shadow patterns that quantitative metrics might miss.

Future assessments should also specifically test the model's handling of partial occlusions, which are typical in real-world scenarios and include things like hands close to the face, spectacles that can make it difficult to identify emotions like anger, and scarves that conceal parts of the face.

With respect to modeling, additional work is needed on architectures that can learn these fine facial movements. The potential balance of the Hybrid CNN-Transformer suggests further investigation of these models. Additionally, Incorporating Explainable AI (XAI) techniques into the existing Hybrid architecture, such as Grad-CAM (Gradient-weighted Class Activation Mapping) or SHAP (SHapley Additive exPlanations), could provide valuable insight into which facial traits are leading to its predictions, particularly for the challenging emotion and age tasks. Furthermore, gaining insight into how the model forms its internal representations may inform targeted architectural adjustments or data augmentation strategies to address specific limitations, such as the tendency to underestimate age or the reduced accuracy in recognizing certain emotions.

For unbalanced datasets, the training corpus can be enhanced by adding synthetic examples produced by Generative Adversarial Networks (GANs) or by synthesizing and rendering 3D faces, especially for age groups and emotion categories that are underrepresented. To further enhance sample diversity, different lighting and stance can be used. Finally, it is essential to conduct a systematic evaluation to verify the effectiveness of these models, trained on synthetic data, when applied to real-world data.

Examining multimodal fusion is a very pertinent avenue from the standpoint of HRI systems. When paired with data from other modalities, like speech prosody (voice tone) or body language (posture, gestures), the facial analysis outputs (emotion, age, gender) may provide a far more comprehensive and accurate picture of the user's state than when facial cues are used alone. An important area of research is creating and assessing architectures that can successfully fuse these disparate data streams.

On the side of model optimization, exploring techniques beyond FP16, such as network pruning and INT8 or INT4 quantization (with careful calibration), could potentially achieve further efficiency gains on edge devices like the Jetson Xavier NX, while closely monitoring any impact on predictive accuracy.

Lastly, to confirm the work's usefulness in practice, it must be integrated into a broader HRI system. assessing how these metrics are used to gauge participants experiences (e.g., technical metrics with the perception of human-robot interactions in real scenarios in terms of task performance, user engagement, perceived social intelligence, etc.).

Conclusions

The conclusions of this thesis are the following:

1. Successfully demonstrated the feasibility of developing and deploying a system capable of performing real-time emotion, age, and gender recognition concurrently on an embedded platform suitable for HRI (NVIDIA Jetson Xavier NX) achieving 64 FPS on inference time.
2. Post-training optimization with TensorRT FP16 was extremely effective, resulting in significant inference speedups (approx. 21x-26x) and model size reductions for the Hybrid model across all tasks while maintaining the original classification accuracy. This emphasizes the need for optimization for edge deployment, but it also emphasizes how crucial it is to choose a baseline model that strikes a balance, as optimizing extremely complicated models may still fail to satisfy real-time requirements without suffering a considerable loss in accuracy.
3. The evaluation showed that performance on offline datasets does not quite match up with how models perform in real-world situations, especially when it comes to accurately recognizing emotions. This demonstrates how difficult it can be to correctly identify nuanced or ambiguous facial expressions and other characteristics such as age groups in less controlled circumstances. These challenges are even greater than those faced when working with static datasets like CK+ or UTKFace.
4. Dataset limitations, including class imbalance (especially for specific emotions and age groups) in training data, clearly impacted model generalization and performance in the experiment. Furthermore, the experiment highlighted how participant demographic imbalances can lead to performance biases, as seen in the lower robustness of gender classification for male participants within the study’s sample. This underscores the critical need for diverse, balanced datasets and bias-aware evaluation.
5. Although the system demonstrated a respectable level of robustness, which may have been derived from "in-the-wild" training data, the experimental error analysis revealed task-specific differences in sensitivity. Gender classification was notably more sensitive to head pose variations than emotion or age estimation. Pose or intrinsic task difficulty seems to have a greater differential influence than measured changes in image quality (luminosity, contrast, and blur) during the trial.
6. Demonstrate substantial progress in efficient edge-based face perception; nonetheless, establishing highly accurate, robust, and unbiased recognition of complex human states (especially emotion and age) appropriate for seamless and dependable HRI remains an unresolved problem. Success necessitates overcoming model constraints, data inadequacies, and perhaps incorporating multimodal information, as well as expanding present capabilities to enable fully sophisticated social robot interactions.

List of references

1. Priyadarshini V, Srinivasulu Reddy U, Venkata Rami Reddy Chirra, Mrudula M, and Suneetha M. Facial expression recognition using multi-block deep cnn. In 2024 5th International Conference on Circuits, Control, Communication and Computing (I4C), pages 427–431, 2024. doi: 10.1109/ I4C62240.2024.10748516.
2. Sefik Ilkin Serengil and Alper Ozpinar. Hyperextended lightface: A facial attribute analysis framework. In 2021 International Conference on Engineering and Emerging Technologies (ICEET), pages 1–4. IEEE, 2021. doi: 10.1109/ICEET53442.2021.9659697. URL https: //ieeexplore.ieee.org/document/9659697.
3. Anna Henschel, Guy Laban, and Emily Cross. What makes a robot social? a review of social robots from science fiction to a home or hospital near you. Current Robotics Reports, 2, 03 2021. doi: 10.1007/s43154-020-00035-0.
4. Matteo Spezialetti, Giuseppe Placidi, and Silvia Rossi. Emotion recognition for human-robot interaction: Recent advances and future perspectives. Frontiers in Robotics and AI, 7:145, 12 2020. doi: 10.3389/frobt.2020.532279.
5. Joost Broekens, Marcel Heerink, and Henk Rosendal. Assistive social robots in elderly care: A review. Gerontechnology, 8:94–103, 04 2009. doi: 10.4017/gt.2009.08.02.002.00.
6. Kennedy, J and Baxter, P and Belpaeme, Tony. Comparing robot embodiments in a guided discovery learning interaction with children. INTERNATIONAL JOURNAL OF SO- CIAL ROBOTICS, 7(2):293–308, 2015. ISSN 1875-4791. URL http://doi.org/10.1007/ s12369-014-0277-4.
7. Hannes Ritschel, Tobias Baur, and Elisabeth Andre. Adapting a robot’s linguistic style based on socially-aware reinforcement learning. 10 2017. doi: 10.1109/ROMAN.2017.8172330.
8. Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In CVPR 2011, pages 1521–1528, 2011. doi: 10.1109/CVPR.2011.5995347.
9. Yuchuang Tong, Haotian Liu, and Zhengtao Zhang. Advancements in humanoid robots: A comprehensive review and future prospects. IEEE/CAA Journal of Automatica Sinica, 11(2):301–328, 2024. doi: 10.1109/JAS.2023.124140.
10. Nicole Robinson, Brendan Tidd, Dylan Campbell, Dana Kulic´, and Peter Corke. Robotic vision for human-robot interaction and collaboration: A survey and systematic review. J. Hum.-Robot Interact., 12(1), February 2023. doi: 10.1145/3570731. URL https://doi.org/10.1145/ 3570731.
11. Kit Yan Chan, Bilal Abu-Salih, Raneem Qaddoura, Ala’ M. Al-Zoubi, Vasile Palade, Duc- Son Pham, Javier Del Ser, and Khan Muhammad. Deep neural networks in the cloud: Re- view, applications, challenges and research directions. Neurocomputing, 545:126327, 2023. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.126327. URL https://www. sciencedirect.com/science/article/pii/S0925231223004502.
12. Ben Kehoe, Sachin Patil, Pieter Abbeel, and Ken Goldberg. Cloud robotics: Architecture, challenges and applications. IEEE Network, 29:64–69, 2015.
13. Gaël Langevin. InMoov – open-source 3d-printed humanoid robot, 2012. URL https: //inmoov.fr. Accessed: March 1, 2025.
14. Christoph Bartneck, Tony Belpaeme, Friederike Eyssel, Takayuki Kanda, Merel Keijsers, and Selma Šabanovic´. Human-Robot Interaction: An Introduction. Cambridge University Press, 2 edition, 2024.

15. Mauro Sarrica, Sonia Brondi, and Leopoldina Fortunati. How many facets does a “social robot” have? a review of scientific and popular definitions online. Information Technology & People, 33, 04 2019. doi: 10.1108/ITP-04-2018-0203.
16. Ruth Stock-Homburg. Survey of emotions in human–robot interactions: Perspectives from robotic psychology on 20 years of research. International Journal of Social Robotics, 14, 06 2021. doi: 10.1007/s12369-021-00778-6.
17. Terrence Fong, Illah Nourbakhsh, and Kerstin Dautenhahn. A survey of socially interactive robots. Robotics and Autonomous Systems, 42(3):143–166, 2003. ISSN 0921-8890. doi: https://doi.org/10.1016/S0921-8890(02)00372-X. URL https://www.sciencedirect.com/science/article/pii/S092188900200372X. Socially Interactive Robots.
18. M. M. A. de Graaf, S. Ben Allouch, and J. A. G. M. van Dijk. What makes robots social?: A user’s perspective on characteristics for social human-robot interaction. In Adriana Tapus, Elisabeth André, Jean-Claude Martin, François Ferland, and Mehdi Ammi, editors, Social Robotics, pages 184–193, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25554-5.
19. Zihan Lin, Francisco Cruz, and Eduardo Sandoval. Self context-aware emotion perception on human-robot interaction. 12 2023.
20. Jagendra Singh, Akansha Singh, Krishna Kant Singh, Bechoo Lal, Harsh Verma, Niranjan Samudre, and Harsh Raperia. Real-time convolutional neural networks for emotion and gen- der classification. Procedia Computer Science, 235:1429–1435, 2024. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2024.04.134. URL https://www.sciencedirect.com/science/article/pii/S187705092400810X.
21. Taciana Saad Rached and Angelo Perkusich. Emotion recognition based on brain-computer interface systems. In Reza Fazel-Rezai, editor, Brain-Computer Interface Systems, chapter 13. IntechOpen, Rijeka, 2013. doi: 10.5772/56227. URL https://doi.org/10.5772/56227.
22. Smith K. Khare, Victoria Blanes-Vidal, Esmaeil S. Nadimi, and U. Rajendra Acharya. Emotion recognition and artificial intelligence: A systematic review (2014–2023) and research recommendations. Information Fusion, 102:102019, 2024. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2023.102019. URL https://www.sciencedirect.com/science/article/pii/S1566253523003354.
23. Paul Ekman. An argument for basic emotions. Cognition & Emotion, 6:169–200, 1992.
24. Albert Mehrabian. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament. Current Psychology, 14(4):261–292, Dec 1996. ISSN 1936-4733. doi: 10.1007/BF02686918. URL https://doi.org/10.1007/BF02686918.
25. Lin Shu, Jinyan Xie, Mingyue Yang, Ziyi Li, Zhenqi Li, Dan Liao, Xiangmin Xu, and Xinyi Yang. A review of emotion recognition using physiological signals. Sensors, 18(7), 2018. ISSN 1424-8220. doi: 10.3390/s18072074. URL https://www.mdpi.com/1424-8220/18/7/2074.
26. Keiron O’Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. URL https://arxiv.org/abs/1511.08458.
27. Yan Wang, Shaoqi Yan, Yang Liu, Wei Song, Jing Liu, Yang Chang, Xinji Mai, Xiping Hu, Wenqiang Zhang, and Zhongxue Gan. A survey on facial expression recognition of static and dynamic emotions, 2024. URL https://arxiv.org/abs/2408.15777.
28. Ian J. Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ra-

maiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Zhang Chuang, and Yoshua Bengio. Challenges in representation learning: A report on three machine learning contests. Neural Networks, 64:59–63, 2015. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2014.09.005. URL https://www.sciencedirect.com/science/article/pii/S0893608014002159. Special Issue on "Deep Learning of Representations".

29. Miyuki Kamachi, Michael Lyons, and Jiro Gyoba. The japanese female facial expression (jaffe) database. Availble: http://www. kasrl. org/jaffe. html, 01 1997.

30. Patrick Lucey, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, pages 94–101, 2010. doi: 10.1109/CVPRW.2010.5543262.

31. Ali Mollahosseini, Behzad Hasani, and Mohammad H. Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. IEEE Transactions on Affective Computing, 10(1):18–31, 2019. doi: 10.1109/TAFFC.2017.2740923.

32. Shan Li, Weihong Deng, and JunPing Du. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2584–2593, 2017. doi: 10.1109/CVPR.2017.277.

33. Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. In 2008 8th IEEE International Conference on Automatic Face & Gesture Recognition, pages 1–8, 2008. doi: 10.1109/AFGR.2008.4813399.

34. Jiyoung Lee, Seungryong Kim, Sunok Kim, Jungin Park, and Kwanghoon Sohn. Context-aware emotion recognition networks. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 10142–10151, 2019. doi: 10.1109/ICCV.2019.01024.

35. Xingxun Jiang, Yuan Zong, Wenming Zheng, Chuangao Tang, Wanchuang Xia, Cheng Lu, and Jiateng Liu. Dfew: A large-scale database for recognizing dynamic facial expressions in the wild. In Proceedings of the 28th ACM International Conference on Multimedia, MM '20, page 2881–2889, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379885. doi: 10.1145/3394171.3413620. URL https://doi.org/10.1145/3394171.3413620.

36. Yan Wang, Yiwen Huang, Zhongying Liu, Shuyong Gao, Wei Zhang, Weifeng Ge, and Wenqiang Zhang. Ferv39k: A large-scale multi-scene dataset for facial expression recognition in videos. pages 20890–20899, 06 2022. doi: 10.1109/CVPR52688.2022.02025.

37. Yuanyuan Liu, Wei Dai, Chuanxu Feng, Wenbin Wang, Guanghao Yin, Jiabei Zeng, and Shiguang Shan. Mafw: A large-scale, multi-modal, compound affective database for dynamic facial expression recognition in the wild. In Proceedings of the 30th ACM International Conference on Multimedia, MM '22, page 24–32, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392037. doi: 10.1145/3503161.3548190. URL https://doi.org/10.1145/3503161.3548190.

38. Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial expression recognition to interpersonal relation prediction. International Journal of Computer Vision, 126 (5):550–569, May 2018. ISSN 1573-1405. doi: 10.1007/s11263-017-1055-1. URL https://doi.org/10.1007/s11263-017-1055-1.

39. Guoying Zhao, Xiaohua Huang, Matti Taini, Stan Z. Li, and Matti Pietikäinen. Facial expression recognition from near-infrared videos. Image and Vision Computing, 29(9):607–619, 2011. ISSN 0262-8856. doi: https://doi.org/10.1016/j.imavis.2011.07.002. URL https://www.sciencedirect.com/science/article/pii/S0262885611000515.

40. Arman Savran, Nese Alyuz, Hamdi Dibeklioglu, Oya Çeliktutan, Berk Gokberk, Bulent Sankur, and Lale Akarun. Bosphorus database for 3d face analysis. pages 47–56, 01 2008. ISBN 978-3-540-89990-7. doi: 10.1007/978-3-540-89991-4_6.

41. Wafa Mellouk and Wahida Handouzi. Facial emotion recognition using deep learning: review and insights. Procedia Computer Science, 175:689–694, 2020. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2020.07.101. URL https://www.sciencedirect.com/science/article/pii/S1877050920318019. The 17th International Conference on Mobile Systems and Pervasive Computing (MobiSPC), The 15th International Conference on Future Networks and Communications (FNC), The 10th International Conference on Sustainable Energy Informa- tion Technology.

42. Hongli Zhang, Alireza Jolfaei, and Mamoun Alazab. A face emotion recognition method using convolutional neural network and image edge computing. IEEE Access, PP:1–1, 10 2019. doi: 10.1109/ACCESS.2019.2949741.

43. Kalpana Chowdary, Tu Nguyen, and Jude D. Deep learning-based facial emotion recognition for human–computer interaction applications. Neural Computing and Applications, 35:1–18, 04 2021. doi: 10.1007/s00521-021-06012-8.

44. Alexander M. Pascual, Erick C. Valverde, Jeong-in Kim, Jin-Woo Jeong, Yuchul Jung, Sang- Ho Kim, and Wansu Lim. Light-fer: A lightweight facial emotion recognition system on edge devices. Sensors, 22(23), 2022. ISSN 1424-8220. doi: 10.3390/s22239524. URL https://www.mdpi.com/1424-8220/22/23/9524.

45. Raghubir Singh and Sukhpal Singh Gill. Edge ai: A survey. Internet of Things and Cyber-Physical Systems, 3:71–92, 2023. ISSN 2667-3452. doi: https://doi.org/10.1016/j.iotcps.2023.02.004. URL https://www.sciencedirect.com/science/article/pii/S2667345223000196.

46. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

47. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1–9, 2015. doi: 10.1109/CVPR.2015.7298594

48. Francois Chollet et al. Keras applications. https://keras.io/api/applications/, 2024. Accessed: 2024-12-27.

49. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

50. Sandeep Kumar Gupta and Neeta Nain. Review: Single attribute and multi attribute facial gender and age estimation. Multimedia Tools and Applications, 82(1):1289–1311, Jan 2023. ISSN 1573-7721. doi: 10.1007/s11042-022-12678-6. URL https://doi.org/10.1007/ s11042-022-12678-6.

51. Hussain Younis, Ameer A. Badr, Alia Karim, Ibrahim Alfadli, Weam Binjumah, Eman Altuwaijri, Maged Nasser, and Nur Intan Raihana Ruhaiyem. Multimodal age and gender estimation for adaptive human-robot interaction: A systematic literature review. Processes, 11, 05 2023. doi: 10.3390/pr11051488.
52. Khaled ELKarazle, Valliappan Raman, and Patrick Then. Facial age estimation using machine learning techniques: An overview. Big Data and Cognitive Computing, 6(4), 2022. ISSN 2504-2289. doi: 10.3390/bdcc6040128. URL https://www.mdpi.com/2504-2289/6/4/128.
53. Kimmo Kärkkäinen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 1547–1557, 2021. doi: 10.1109/WACV48630.2021.00159.
54. Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), pages 252–257, 2015. doi: 10.1109/ICCVW.2015.41.
55. K. Ricanek and T. Tesafaye. Morph: a longitudinal image database of normal adult age-progression. In 7th International Conference on Automatic Face and Gesture Recognition (FGR06), pages 341–345, 2006. doi: 10.1109/FGR.2006.78.
56. Eran Eidinger, Roee Enbar, and Tal Hassner. Age and gender estimation of unfiltered faces. IEEE Transactions on Information Forensics and Security, 9(12):2170–2179, 2014. doi: 10.1109/TIFS. 2014.2359646.
57. Song Yang Zhang, Zhifei and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2017.
58. Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou. Agedb: The first manually collected, in-the-wild age database. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1997–2005, 2017. doi: 10.1109/CVPRW.2017.250.
59. Hu Han, Anil K. Jain, Fang Wang, Shiguang Shan, and Xilin Chen. Heterogeneous face attribute estimation: A deep multi-task learning approach. IEEE Trans. Pattern Anal. Mach. Intell., 40 (11):2597–2609, November 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2738004. URL https://doi.org/10.1109/TPAMI.2017.2738004
60. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4510–4520, 2018. doi: 10.1109/CVPR.2018.00474.
61. NVIDIA Corporation. Nvidia jetson xavier nx series systemon-module, 2024. URL https: //developer.download.nvidia.com/assets/embedded. Accessed: 2024-12-28.
62. NVIDIA Corporation. Nvidia tensorrt, 2024. URL https://developer.nvidia.com/ tensorrt. Accessed: 2024-12-28.
63. Yuxiao Zhou and Kecheng Yang. Exploring tensorrt to improve real-time inference for deep learning. In 2022 IEEE 24th Int Conf on High Performance Computing & Communications, pages 2011–2018, 2022. doi: 10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00299.
64. Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and quantization for deep neural network acceleration: A survey. Neurocomput., 461(C):370–403, October 2021. ISSN 0925-2312. doi: 10.1016/j.neucom.2021.07.045. URL https://doi.org/10.

1016/j.neucom.2021.07.045.

65. Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference, 2021. URL https://arxiv.org/abs/2103.13630.
66. VIDIA Corporation. TensorRT Documentation. https://docs.nvidia.com/deeplearning/tensorrt/latest/index.html, 2025. Accessed: March 2025.
67. Alexander Buslaev, Vladimir Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr Kalinin. Albumentations: Fast and flexible image augmentations. Information, 11: 125, 02 2020. doi: 10.3390/info11020125.
68. Zhengyou Zhang. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330–1334, 2000. doi: 10.1109/34.888718.
69. Davis E. King. Dlib-ml: A machine learning toolkit. J. Mach. Learn. Res., 10:1755–1758, December 2009. ISSN 1532-4435.
70. Erik Murphy-Chutorian and Mohan Manubhai Trivedi. Head pose estimation in computer vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(4):607–626, 2009. doi: 10.1109/TPAMI.2008.106.

Appendices

Appendix 1. Experiment code

```python
import cv2
import dlib
import numpy as np
import tensorrt as trt
import pycuda.driver as cuda
import pycuda.autoinit
import time
import os
import json
from datetime import datetime
from collections import deque
from PyQt5.QtWidgets import (QApplication, QMainWindow, QPushButton, QVBoxLayout, QWidget,
                             QLabel, QHBoxLayout, QSpinBox, QRadioButton, QButtonGroup, QMessageBox)
from PyQt5.QtCore import QTimer, Qt
from PyQt5.QtGui import QImage, QPixmap

# Create directory for saving images if it doesn't exist
os.makedirs("captured_images", exist_ok=True)

# Load face detector
detector = dlib.get_frontal_face_detector()

# Load TensorRT models
TRT_LOGGER = trt.Logger()
emotion_path = "models/model_fp16.trt"
gender_path = "models/gender_fp16.trt"
age_path = "models/age_fp16.trt"

def load_engine(engine_path):
    with open(engine_path, "rb") as f, trt.Runtime(TRT_LOGGER) as runtime:
        return runtime.deserialize_cuda_engine(f.read())

# Load the engines for all three models
emotion_engine = load_engine(emotion_path)
age_engine = load_engine(age_path)
gender_engine = load_engine(gender_path)

```

```python
emotion_context = emotion_engine.create_execution_context()
age_context = age_engine.create_execution_context()
gender_context = gender_engine.create_execution_context()

# Define shapes for input and output
emotion_input_shape = (1, 96, 96, 1)
emotion_output_shape = (1, 7)
age_input_shape = (1, 224, 224, 3)
age_output_shape = (1, 9)
gender_input_shape = (1, 224, 224, 3)
gender_output_shape = (1, 1)

# Memory allocation for GPU
emotion_d_input = cuda.mem_alloc(int(np.prod(emotion_input_shape) * np.dtype(np.float32).itemsize))
emotion_d_output = cuda.mem_alloc(int(np.prod(emotion_output_shape) * np.dtype(np.float32).itemsize))

age_d_input = cuda.mem_alloc(int(np.prod(age_input_shape) * np.dtype(np.float32).itemsize))
age_d_output = cuda.mem_alloc(int(np.prod(age_output_shape) * np.dtype(np.float32).itemsize))

gender_d_input = cuda.mem_alloc(int(np.prod(gender_input_shape) * np.dtype(np.float32).itemsize))
gender_d_output = cuda.mem_alloc(int(np.prod(gender_output_shape) * np.dtype(np.float32).itemsize))

# Labels for predictions
emotion_labels = ['Angry', 'Disgust', 'Fear', 'Happy', 'Neutral', 'Sad', 'Surprise']
gender_labels = ['Male', 'Female']
age_labels = ['0-2', '3-9', '10-19', '20-29', '30-39', '40-49', '50-59', '60-69', '70+']

class FaceAnalysisApp(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Face Analysis System")
        self.setGeometry(100, 100, 1200, 650)  # Made taller to accommodate new metrics

        # Main widget and layout
        central_widget = QWidget()
        self.setCentralWidget(central_widget)
        main_layout = QHBoxLayout(central_widget)

        # Video display area
        self.video_label = QLabel()
        self.video_label.setFixedSize(640, 480)
        self.video_label.setAlignment(Qt.AlignCenter)

        # Face display area
        self.face_label = QLabel()
        self.face_label.setFixedSize(200, 200)
        self.face_label.setAlignment(Qt.AlignCenter)

        # Control panel
        control_panel = QWidget()
        control_layout = QVBoxLayout(control_panel)

        # Baseline inputs using radio buttons
        baseline_layout = QVBoxLayout()

        # Baseline Emotion Buttons
        emotion_buttons_layout = QHBoxLayout()
        emotion_buttons_layout.addWidget(QLabel("Baseline Emotion:"))
        self.emotion_group = QButtonGroup(self)
        for emo in emotion_labels:
            radio = QRadioButton(emo)
            self.emotion_group.addButton(radio)
            emotion_buttons_layout.addWidget(radio)
            if emo == "Happy":  # default selection
                radio.setChecked(True)
        baseline_layout.addLayout(emotion_buttons_layout)

        # Baseline Age Buttons
        age_buttons_layout = QHBoxLayout()
        age_buttons_layout.addWidget(QLabel("Baseline Age:"))
        self.age_group = QButtonGroup(self)
        for age in age_labels:
            radio = QRadioButton(age)
            self.age_group.addButton(radio)
            age_buttons_layout.addWidget(radio)
            if age == "20-29":  # default selection
                radio.setChecked(True)
        baseline_layout.addLayout(age_buttons_layout)

        # Baseline Gender Buttons
        gender_buttons_layout = QHBoxLayout()
        gender_buttons_layout.addWidget(QLabel("Baseline Gender:"))
        self.gender_group = QButtonGroup(self)
```

```python
        for gender in gender_labels:
            radio = QRadioButton(gender)
            self.gender_group.addButton(radio)
            gender_buttons_layout.addWidget(radio)
            if gender == "Male":  # default selection
                radio.setChecked(True)
        baseline_layout.addLayout(gender_buttons_layout)

        control_layout.addLayout(baseline_layout)

        # Session duration control
        duration_layout = QHBoxLayout()
        duration_layout.addWidget(QLabel("Session Duration (seconds):"))
        self.duration_spinbox = QSpinBox()
        self.duration_spinbox.setRange(1, 60)
        self.duration_spinbox.setValue(5)
        duration_layout.addWidget(self.duration_spinbox)
        control_layout.addLayout(duration_layout)

        # Start/Stop button
        self.start_button = QPushButton("Start Session")
        self.start_button.clicked.connect(self.toggle_session)
        control_layout.addWidget(self.start_button)

        # Status label
        self.status_label = QLabel("Status: Ready")
        control_layout.addWidget(self.status_label)

        # Current prediction label
        self.prediction_label = QLabel("No predictions yet")
        control_layout.addWidget(self.prediction_label)

        # Saved images count
        self.saved_count_label = QLabel("Images saved: 0")
        control_layout.addWidget(self.saved_count_label)

        # Inference count label
        self.inference_count_label = QLabel("Inferences: 0")
        control_layout.addWidget(self.inference_count_label)

        # FPS counter label
        self.fps_label = QLabel("Camera FPS: 0.0")
        control_layout.addWidget(self.fps_label)

        # Inference timing label
        self.inference_time_label = QLabel("Inference time: 0.0 ms")
        control_layout.addWidget(self.inference_time_label)

        # Model breakdown timing labels
        self.emotion_time_label = QLabel("Emotion model: 0.0 ms")
        self.age_time_label = QLabel("Age model: 0.0 ms")
        self.gender_time_label = QLabel("Gender model: 0.0 ms")

        # Add model timing labels
        control_layout.addWidget(self.emotion_time_label)
        control_layout.addWidget(self.age_time_label)
        control_layout.addWidget(self.gender_time_label)

        # Add displays and control panel to main layout
        main_layout.addWidget(self.video_label)
        display_panel = QWidget()
        display_layout = QVBoxLayout(display_panel)
        display_layout.addWidget(self.face_label)
        display_layout.addWidget(QLabel("Detected Face"))
        main_layout.addWidget(display_panel)
        main_layout.addWidget(control_panel)

        # Video capture initialization
        self.video = cv2.VideoCapture(0)
        if not self.video.isOpened():
            self.status_label.setText("Error: Could not open camera")

        # Timer for updating video frames
        self.timer = QTimer()
        self.timer.timeout.connect(self.update_frame)
        self.timer.start(30)  # roughly 30 FPS

        # Timer for session duration
        self.session_timer = QTimer()
        self.session_timer.timeout.connect(self.end_session)

        # Session state variables
        self.session_active = False
        self.saved_count = 0
        self.inference_count = 0
        self.session_data = []
        self.current_session_dir = None

        # Baseline reference values from user inputs (set on session start)
        self.input_emotion = None
```

```python
        self.input_age = None
        self.input_gender = None

        # FPS calculation variables
        self.prev_frame_time = time.time()
        self.frame_times = deque(maxlen=30) # Store the last 30 frame times for smoothing

        # Inference timing variables
        self.total_inference_time = 0
        self.emotion_inference_time = 0
        self.age_inference_time = 0
        self.gender_inference_time = 0

        # Frame counters for FPS and no face detection
        self.total_frames = 0
        self.frames_no_face = 0
        self.session_start_time = None

    def toggle_session(self):
        if not self.session_active:
            self.start_session()
        else:
            self.end_session()

    def start_session(self):
        # Read baseline values from the radio button groups
        self.input_emotion = self.emotion_group.checkedButton().text() if self.emotion_group.checkedButton() else "Happy"
        self.input_age = self.age_group.checkedButton().text() if self.age_group.checkedButton() else "20-29"
        self.input_gender = self.gender_group.checkedButton().text() if self.gender_group.checkedButton() else "Male"

        # Create session directory using the baseline values and a timestamp
        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
        folder_name = f"{self.input_emotion}_{self.input_age}_{self.input_gender}_{timestamp}"
        self.current_session_dir = os.path.join("captured_images", folder_name)
        os.makedirs(self.current_session_dir, exist_ok=True)

        self.session_active = True
        self.start_button.setText("Stop Session")
        self.status_label.setText("Status: Session active (Monitoring changes)")

        # Reset session data and counters
        self.session_data = []
        self.saved_count = 0
        self.inference_count = 0
        self.saved_count_label.setText("Images saved: 0")
        self.inference_count_label.setText("Inferences: 0")

        # Reset frame counters and record session start time
        self.total_frames = 0
        self.frames_no_face = 0
        self.session_start_time = time.time()

        # Start the session timer (duration in milliseconds)
        duration = self.duration_spinbox.value() * 1000
        self.session_timer.start(duration)

    def end_session(self):
        if self.session_active:
            # Calculate actual session duration
            session_duration_actual = time.time() - self.session_start_time
            avg_camera_fps = self.total_frames / session_duration_actual if session_duration_actual > 0 else 0

            if self.current_session_dir and self.session_data:
                metadata_file = os.path.join(self.current_session_dir, "session_data.json")
                with open(metadata_file, ’w’) as f:
                    avg_inference_time = self.total_inference_time / self.inference_count if self.inference_count > 0 else 0
                    avg_emotion_time = self.emotion_inference_time / self.inference_count if self.inference_count > 0 else 0
                    avg_age_time = self.age_inference_time / self.inference_count if self.inference_count > 0 else 0
                    avg_gender_time = self.gender_inference_time / self.inference_count if self.inference_count > 0 else 0

                    json.dump({
                        'baseline': {
                            'emotion': self.input_emotion,
                            'age': self.input_age,
                            'gender': self.input_gender
                        },
                        'timestamp': datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
                        'duration': self.duration_spinbox.value(),
                        'total_inferences': self.inference_count,
                        'total_saved_images': self.saved_count,
                        'performance_metrics': {
                            'avg_inference_time_ms': round(avg_inference_time * 1000, 2),
                            'avg_emotion_model_time_ms': round(avg_emotion_time * 1000, 2),
                            'avg_age_model_time_ms': round(avg_age_time * 1000, 2),
                            'avg_gender_model_time_ms': round(avg_gender_time * 1000, 2),
                            'inferences_per_second': round(self.inference_count / self.duration_spinbox.value(), 2) if self.duration_spinbox.value() > 0 else 0,
                            'avg_camera_fps': round(avg_camera_fps, 2),
                            'total_frames': self.total_frames,
                            'frames_no_face': self.frames_no_face
                        },
                        'data': self.session_data
                    }, f, indent=4)
                print(f"Session data saved to {metadata_file}")

        self.session_active = False
        self.start_button.setText("Start Session")
        self.status_label.setText("Status: Ready")
        self.session_timer.stop()

        # Optionally, show a message box indicating the session has ended
        if self.inference_count > 0:
            avg_time = (self.total_inference_time / self.inference_count) * 1000  # convert to ms
            QMessageBox.information(self, "Session Ended",
                                    f"The session has ended.\nTotal inferences: {self.inference_count}\n"
                                    f"Images saved: {self.saved_count}\nAvg. inference time: {avg_time:.2f} ms")
        else:
            QMessageBox.information(self, "Session Ended",
                                    f"The session has ended.\nTotal inferences: {self.inference_count}\n"
                                    f"Images saved: {self.saved_count}")

        # Reset inference timing counters for the next session
        self.total_inference_time = 0
        self.emotion_inference_time = 0
        self.age_inference_time = 0
        self.gender_inference_time = 0
        self.inference_count = 0

    def save_image_with_data(self, frame, prefix, emotion, age, gender):
        if not self.current_session_dir:
            return None

        timestamp = datetime.now().strftime("%Y%m%d_%H%M%S_%f")
        # Construct file name with baseline inputs and current predictions
        filename = os.path.join(
            self.current_session_dir,
            f"{self.input_emotion}_{self.input_age}_{self.input_gender}_{prefix}_E{emotion}_A{age}_G{gender}_{timestamp}.jpg"
        )
        cv2.imwrite(filename, frame)

        data_entry = {
            'filename': os.path.basename(filename),
            'timestamp': timestamp,
            'type': prefix,
            'inference_number': self.inference_count,
            'predictions': {
                'emotion': emotion,
                'age': age,
                'gender': gender
            },
            'timing': {
                'total_inference_ms': round(self.total_inference_time * 1000, 2),
```

```python
                'emotion_inference_ms': round(self.emotion_inference_time * 1000, 2),
                'age_inference_ms': round(self.age_inference_time * 1000, 2),
                'gender_inference_ms': round(self.gender_inference_time * 1000, 2)
            }
        }
        self.session_data.append(data_entry)
        self.saved_count += 1
        self.saved_count_label.setText(f"Images saved: {self.saved_count}")
        return filename

    def update_frame(self):
        # Calculate FPS using a deque of frame times
        current_time = time.time()
        self.frame_times.append(current_time - self.prev_frame_time)
        self.prev_frame_time = current_time

        if len(self.frame_times) > 0:
            avg_frame_time = sum(self.frame_times) / len(self.frame_times)
            fps = 1.0 / avg_frame_time if avg_frame_time > 0 else 0
            self.fps_label.setText(f"Camera FPS: {fps:.1f}")

        ret, frame = self.video.read()
        if not ret:
            self.status_label.setText("Error: Failed to capture frame")
            return

        self.total_frames += 1  # Count every captured frame

        display_frame = frame.copy()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = detector(gray)

        face_found = False

        for i in faces:
            face_found = True
            x, y, w, h = (i.left(), i.top(), i.width(), i.height())
            padding = 30
            x1 = max(0, x - padding)
            y1 = max(0, y - padding)
            x2 = min(frame.shape[1], x + w + padding)
```

```
                y2 = min(frame.shape[0], y + h + padding)

                cv2.rectangle(display_frame, (x1, y1), (x2, y2), (43, 91, 132), 2)
                face = gray[y1:y2, x1:x2]
                if face.size == 0:
                    continue

                # Preprocess face for emotion inference
                face_emotion = cv2.resize(face, (96, 96))
                face_emotion = face_emotion.astype(np.float32) / 255.0
                face_emotion = np.expand_dims(face_emotion, axis=[0, -1])

                # Preprocess face for age and gender inference
                face_age_gender = cv2.resize(face, (224, 224))
                face_age_gender = cv2.cvtColor(face_age_gender, cv2.COLOR_GRAY2BGR)
                face_age_gender = face_age_gender.astype(np.float32) / 255.0
                face_age_gender = np.expand_dims(face_age_gender, axis=0)

                if self.session_active:
                    # Increment inference counter when session is active and processing a face
                    self.inference_count += 1
                    self.inference_count_label.setText(f"Inferences: {self.inference_count}")

                    # Time the entire inference process
                    inference_start = time.time()

                    # Emotion inference with timing
                    emotion_start = time.time()
                    cuda.memcpy_htod(emotion_d_input, face_emotion.ravel())
                    emotion_context.execute_v2([int(emotion_d_input), int(emotion_d_output)])
                    output_emotion = np.empty(emotion_output_shape, dtype=np.float32)
                    cuda.memcpy_dtoh(output_emotion, emotion_d_output)
                    emotion_time = time.time() - emotion_start
                    self.emotion_inference_time += emotion_time
                    self.emotion_time_label.setText(f"Emotion model: {emotion_time*1000:.2f} ms")

                    # Age inference with timing
                    age_start = time.time()
                    cuda.memcpy_htod(age_d_input, face_age_gender.ravel())
                    age_context.execute_v2([int(age_d_input), int(age_d_output)])
                    output_age = np.empty(age_output_shape, dtype=np.float32)
                    cuda.memcpy_dtoh(output_age, age_d_output)
                    age_time = time.time() - age_start
                    self.age_inference_time += age_time
                    self.age_time_label.setText(f"Age model: {age_time*1000:.2f} ms")

                    # Gender inference with timing
                    gender_start = time.time()
                    cuda.memcpy_htod(gender_d_input, face_age_gender.ravel())
                    gender_context.execute_v2([int(gender_d_input), int(gender_d_output)])
                    output_gender = np.empty(gender_output_shape, dtype=np.float32)
                    cuda.memcpy_dtoh(output_gender, gender_d_output)
                    gender_time = time.time() - gender_start
                    self.gender_inference_time += gender_time
                    self.gender_time_label.setText(f"Gender model: {gender_time*1000:.2f} ms")

                    # Total inference time for this frame
                    total_time = time.time() - inference_start
                    self.total_inference_time += total_time
                    self.inference_time_label.setText(f"Inference time: {total_time*1000:.2f} ms")
                else:
                    # If not in active session, still run inference but without timing
                    cuda.memcpy_htod(emotion_d_input, face_emotion.ravel())
                    emotion_context.execute_v2([int(emotion_d_input), int(emotion_d_output)])
                    output_emotion = np.empty(emotion_output_shape, dtype=np.float32)
                    cuda.memcpy_dtoh(output_emotion, emotion_d_output)

                    cuda.memcpy_htod(age_d_input, face_age_gender.ravel())
                    age_context.execute_v2([int(age_d_input), int(age_d_output)])
                    output_age = np.empty(age_output_shape, dtype=np.float32)
                    cuda.memcpy_dtoh(output_age, age_d_output)
```

```
            cuda.memcpy_htod(gender_d_input, face_age_gender.ravel())
            gender_context.execute_v2([int(gender_d_input), int(gender_d_output)])
            output_gender = np.empty(gender_output_shape, dtype=np.float32)
            cuda.memcpy_dtoh(output_gender, gender_d_output)

        emotion_idx = np.argmax(output_emotion)
        emotion = emotion_labels[emotion_idx]

        age_idx = np.argmax(output_age)
        age = age_labels[age_idx]

        gender_prob = output_gender[0][0]
        gender = 'Female' if gender_prob > 0.5 else 'Male'

        # Update current prediction label
        self.prediction_label.setText(f"Current: {emotion}, {age}, {gender}")

        # If session is active, compare current predictions with the baseline inputs
        if self.session_active:
            if (emotion != self.input_emotion or age != self.input_age or gender != self.input_gender):
                change_filename = self.save_image_with_data(frame, "change", emotion, age, gender)
                print(f"Change detected! New - Emotion: {emotion}, Age: {age}, Gender: {gender}")
                print(f"Saved image: {change_filename}")

        # Draw text with outline on the display frame
        font_size = 0.7
        font = cv2.FONT_HERSHEY_SIMPLEX
        font_color = (248, 249, 250)
        font_thickness = 2
        line_spacing = 25

        def put_text_with_outline(text, y_pos):
            cv2.putText(display_frame, text, (x1, y_pos), font, font_size, (0, 0, 0), font_thickness + 1)
            cv2.putText(display_frame, text, (x1, y_pos), font, font_size, font_color, font_thickness)

```

```python
            put_text_with_outline(f"Emotion: {emotion}", y1 - line_spacing * 3)
            put_text_with_outline(f"Age: {age}", y1 - line_spacing * 2)
            put_text_with_outline(f"Gender: {gender}", y1 - line_spacing)

            # Update the face display area
            face_display = cv2.resize(face, (200, 200))
            h_face, w_face = face_display.shape
            q_face_image = QImage(face_display.data, w_face, h_face, w_face, QImage.Format_Grayscale8)
            self.face_label.setPixmap(QPixmap.fromImage(q_face_image))

            break  # Process only the first detected face

        if not face_found:
            self.frames_no_face += 1  # Count frames with no face detected
            self.face_label.clear()
            self.prediction_label.setText("No face detected")

        # Convert the display frame from BGR to RGB and update the video label
        rgb_frame = cv2.cvtColor(display_frame, cv2.COLOR_BGR2RGB)
        h_frame, w_frame, ch = rgb_frame.shape
        q_image = QImage(rgb_frame.data, w_frame, h_frame, w_frame * ch, QImage.Format_RGB888)
        self.video_label.setPixmap(QPixmap.fromImage(q_image))

    def closeEvent(self, event):
        self.timer.stop()
        self.session_timer.stop()
        self.video.release()
        event.accept()

if __name__ == "__main__":
    app = QApplication([])
    window = FaceAnalysisApp()
    window.show()
    app.exec_()
```