Gender, Age, and Ethnicity Classification Using Multi-Task Learning

Utpalraj Kemprai

December 29, 2024

Abstract

This project presents a multitask deep learning framework designed to simultaneously predict gender, age group, and ethnicity from facial images. Leveraging the FairFace dataset, we trained a MobileNetV2-based Convolutional Neural Network (CNN) to achieve a balance between computational efficiency and high accuracy. The model achieved a gender classification accuracy of 91.77%, with ethnicity and age group predictions at 66.81% and 55.59%, respectively. The pipeline begins with face detection and cropping using Haar cascades, followed by feature extraction through the MobileNetV2 architecture, optimized for multitask learning. Model performance was assessed across demographic groups, ensuring fairness and inclusivity in predictions. Compared to single-task models, the multitask model demonstrated improved accuracy across all tasks. The trained model is deployed via Streamlit, showcasing its versatility for real-world applications such as security, marketing analytics, and social research.

Contents

List of Figures

List of Tables

1. Introduction

The surge in demand for automated human profiling systems has propelled significant advancements in computer vision and deep learning technologies. Accurately predicting demographic attributes—namely gender, age group, and ethnicity—from facial images holds substantial value across various sectors, including marketing analytics, healthcare diagnostics, and security surveillance. This project aims to develop a robust multitask learning framework utilizing Convolutional Neural Networks (CNNs) to deliver precise predictions of these demographic factors.

Multitask learning presents several advantages over traditional single-task models. By sharing feature representations across related tasks, multitask frameworks not only enhance prediction accuracy but also reduce computational complexity and improve model generalizability. However, challenges such as data imbalance and inter-class similarities in ethnicity and age groups necessitate thoughtful model design and evaluation.

The project encompasses three primary areas:

1. **Face Detection and Preprocessing**: Utilizing Haar cascades for accurate face detection and image cropping.

2. **Demographic Prediction**: Implementing a lightweight multitask CNN architecture to predict gender, age group, and ethnicity concurrently.

3. **Model Deployment**: Developing a user-friendly application using Streamlit to demonstrate the model's real-world applicability.

While the focus is on demographic attribute classification, broader applications such as emotion recognition and face identification are beyond the scope of this study.

2. Literature Review

2.1. Metrics for Evaluating Data Imbalance

The challenge of data imbalance has been extensively studied in the field of machine learning, as it can significantly impact model performance, particularly for classification tasks. To address this issue, various metrics have been proposed to quantitatively assess the degree of imbalance in datasets. This section reviews key metrics commonly used in literature:

2.1.1. Max Ratio and Min Ratio

These metrics quantify the imbalance by comparing the maximum and minimum proportions of class frequencies. The **max ratio** is the proportion of the largest class relative to the total dataset size, while the **min ratio** measures the smallest class's proportion. Together, they highlight the spread of class distributions and are simple yet effective for identifying extreme imbalance scenarios.

Let the dataset have C classes, and let N_c be the number of instances in class c. The total number of instances is denoted as $N = \sum_{c=1}^{C} N_c$. Then the **max ratio** and **min ratio** can be defined as:

$$\text{Max Ratio} = \frac{\max(N_c)}{N}$$

$$\text{Min Ratio} = \frac{\min(N_c)}{N}$$

2.1.2. Ratio Range

The ratio range, computed as the difference between the max and min ratios, provides a straightforward measure of the variability in class proportions. A higher ratio range indicates greater disparity among classes, which could lead to biased model predictions.

$$\text{Ratio Range} = \text{Max Ratio} - \text{Min Ratio}$$

2.1.3. Coefficient of Variation (CV)

The **coefficient of variation** (CV) measures the relative variability of class frequencies, calculated as the standard deviation normalized by the mean class frequency. It captures the extent of imbalance relative to the average class size, with higher values indicating more pronounced imbalance.

Let the frequencies of the C classes be $N_1, N_2, \ldots, N_C$. The mean class frequency μ and the standard deviation σ are defined as:

$$\mu = \frac{1}{C}\sum_{c=1}^{C} N_c$$

$$\sigma = \sqrt{\frac{1}{C}\sum_{c=1}^{C}(N_c - \mu)^2}$$

The **coefficient of variation** is then:

$$\text{CV} = \frac{\sigma}{\mu}$$

2.1.4. Entropy

Entropy quantifies the uncertainty or diversity in class distributions. A perfectly balanced dataset has maximum entropy, while a highly imbalanced dataset has lower entropy. Entropy is particularly useful for multi-class problems, where it provides an aggregated view of imbalance.

Entropy H for a multi-class dataset is defined as:

$$H = -\sum_{c=1}^{C} p_c \log(p_c)$$

where $p_c = \frac{N_c}{N}$ is the proportion of class c in the dataset.

2.1.5. Gini Index

The **Gini index**, commonly used in economics to measure inequality, has been adapted to evaluate imbalance in machine learning. It ranges from 0 (perfect balance) to 1 (complete imbalance), capturing the inequality in class frequencies. The Gini index G is defined as:

$$G = 1 - \sum_{c=1}^{C} p_c^2$$

where $p_c = \frac{N_c}{N}$ is the proportion of class c in the dataset.

2.2. Face Analysis Techniques

Face analysis has been a significant research area in computer vision, evolving from traditional methods to state-of-the-art deep learning-based approaches. Early techniques relied heavily on handcrafted features, such as Haar cascades [9] and Local Binary Patterns (LBP) [10] , which were effective for simple tasks like face detection and basic feature extraction. Haar cascades, introduced by Viola and Jones (2001), revolutionized real-time face detection by using an integral image representation and a cascaded classifier structure. LBP further contributed to texture-based face representation by encoding local image patterns.

With the advent of deep learning, Convolutional Neural Networks (CNNs) have emerged as the standard for face analysis tasks due to their ability to automatically learn hierarchical feature representations from raw pixel data. CNN-based architectures like AlexNet, VGG, and ResNet have demonstrated superior performance in tasks such as face detection, recognition, and attribute classification. Advanced face analysis frameworks now integrate facial landmarks for alignment and preprocessing to improve accuracy. Techniques like the Single Shot Multibox Detector (SSD) and Multi-Task Cascaded Convolutional Networks (MTCNN) have also been adopted for robust face detection under varying conditions of lighting, pose, and occlusion.

Recent trends emphasize fairness and explainability in face analysis, as traditional models have often shown bias against underrepresented demographic groups. The shift toward datasets and methods that account for demographic diversity has become crucial in reducing these disparities.

2.3. Multitask Learning

Multitask learning (MTL) is a paradigm in which a model is trained to optimize multiple objectives simultaneously. By sharing a common feature representation across tasks, MTL improves generalization and reduces computational overhead. This approach has shown remarkable success in domains like facial attribute analysis, emotion detection, and biometric verification.

Caruana (1997) [13] introduced MTL as a method to improve generalization by leveraging the domain-specific information contained in the training signals of related tasks. In the context of face analysis, MTL is particularly advantageous because facial features such as eyes, nose, and mouth provide shared information relevant to multiple tasks like gender classification, age estimation, and ethnicity prediction.

Research has demonstrated that MTL can mitigate overfitting, especially when labeled data is limited. Studies by Ranjan et al. (2017) highlight that MTL frameworks achieve state-of-the-art performance by balancing task-specific losses with a shared feature extractor. Moreover, task interdependence plays a critical role; for instance, learning age and gender together can improve the prediction accuracy of both tasks, as they share biological and social correlations.

Advanced implementations of MTL employ attention mechanisms and task-specific adapters to manage conflicts between tasks with varying levels of complexity or data availability. These methods further refine the shared representation to maximize task performance without sacrificing generalizability.

2.4. FairFace Dataset

The FairFace dataset [1] represents a significant advancement in addressing bias in face analysis datasets. Traditional datasets like LFW (Labeled Faces in the Wild) and CelebA, while widely used, often exhibit skewed distributions, overrepresenting certain demographic groups (e.g., white males) while underrepresenting others. Such biases lead to models that perform poorly on underrepresented groups, raising concerns about fairness and equity in automated systems.

FairFace, introduced by Karkkainen and Joo (2021) [1], addresses these challenges by providing a balanced dataset annotated for gender, age, and ethnicity across seven racial groups. This diversity ensures that models trained on FairFace are better equipped to generalize across different demographic groups, reducing performance disparities.

The dataset contains around 100,000 high-quality face images with labels for attributes such as race (White, Black, Indian, East Asian, Southeast Asian, Middle Eastern, and Latino), gender (Male and Female), and age (seven age groups). It has been widely adopted in research and commercial applications for training fair and unbiased models. Studies using FairFace have shown improved fairness metrics, such as equalized odds and demographic parity, in tasks ranging from facial recognition to demographic attribute classification.

By addressing the limitations of earlier datasets, FairFace not only promotes ethical AI practices but also enables researchers to build robust systems applicable to diverse populations. Its balanced representation has made it a benchmark for fairness in face analysis tasks.

3. Methodology

3.1. Dataset

The FairFace dataset [1], introduced by Kärkkäinen and Joo (2021), is a large-scale facial dataset designed to address biases that exist in many facial recognition and demographic classification datasets. Traditional datasets, such as CelebA and LFW, often suffer from biases in representation, which can lead to suboptimal performance on underrepresented demographic groups. The FairFace dataset was created to mitigate this issue by ensuring balanced representation across multiple demographic groups, including racial, gender, and age categories.

3.1.1. Dataset Overview

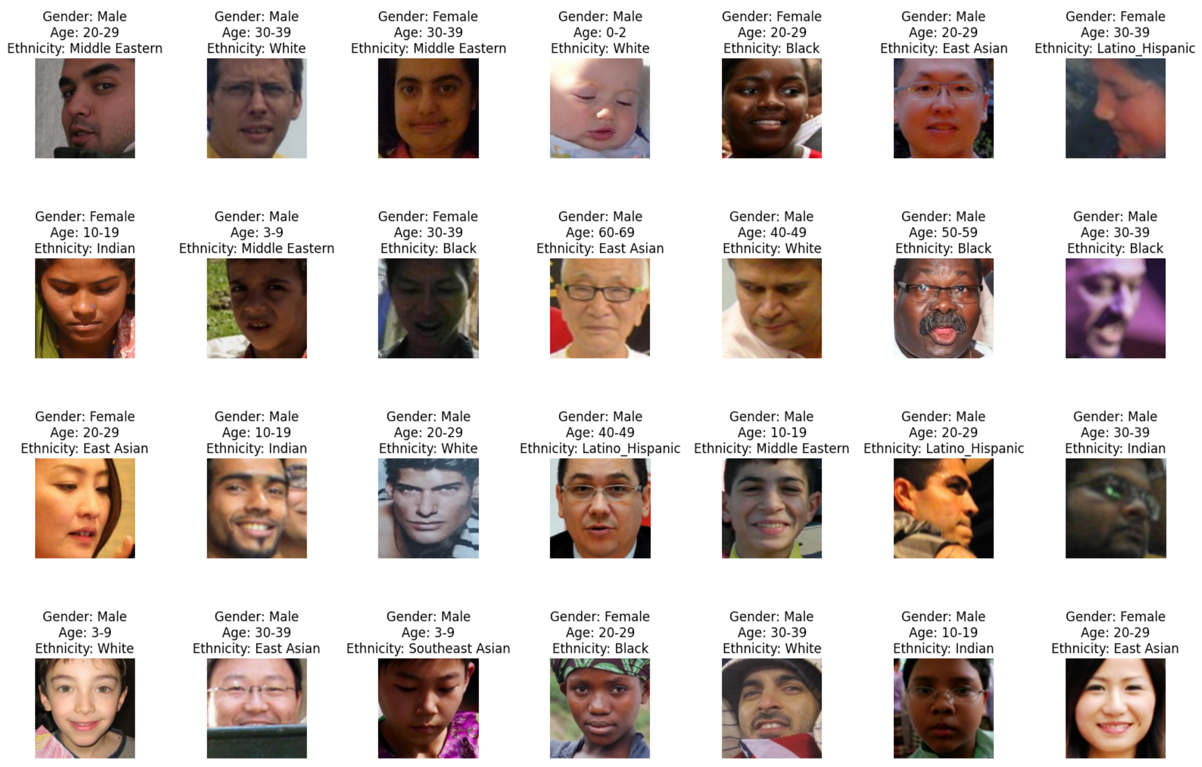

Gender: Male Age: 20-29 Ethnicity: Middle Eastern
Gender: Male Age: 30-39 Ethnicity: White
Gender: Female Age: 30-39 Ethnicity: Middle Eastern
Gender: Male Age: 0-2 Ethnicity: White
Gender: Female Age: 20-29 Ethnicity: Black
Gender: Male Age: 20-29 Ethnicity: East Asian
Gender: Female Age: 30-39 Ethnicity: Latino_Hispanic
Gender: Female Age: 10-19 Ethnicity: Indian
Gender: Male Age: 3-9 Ethnicity: Middle Eastern
Gender: Female Age: 30-39 Ethnicity: Black
Gender: Male Age: 60-69 Ethnicity: East Asian
Gender: Male Age: 40-49 Ethnicity: White
Gender: Male Age: 50-59 Ethnicity: Black
Gender: Male Age: 30-39 Ethnicity: Black
Gender: Female Age: 20-29 Ethnicity: East Asian
Gender: Male Age: 10-19 Ethnicity: Indian
Gender: Male Age: 20-29 Ethnicity: White
Gender: Male Age: 40-49 Ethnicity: Latino_Hispanic
Gender: Male Age: 10-19 Ethnicity: Middle Eastern
Gender: Male Age: 20-29 Ethnicity: Latino_Hispanic
Gender: Male Age: 30-39 Ethnicity: Indian
Gender: Male Age: 3-9 Ethnicity: White
Gender: Male Age: 30-39 Ethnicity: East Asian
Gender: Male Age: 3-9 Ethnicity: Southeast Asian
Gender: Female Age: 20-29 Ethnicity: Black
Gender: Male Age: 30-39 Ethnicity: White
Gender: Male Age: 10-19 Ethnicity: Indian
Gender: Female Age: 20-29 Ethnicity: East Asian

Figure 3.1: Sample Images from FairFace Dataset

The FairFace dataset contains around 100,000 images of faces, with annotations for gender, age group, and ethnicity. The images are sourced from the YFCC-100M dataset, ensuring diversity in the facial characteristics represented. This dataset provides a critical resource for building AI systems that are not only accurate but also fair and ethical.

- Race/Ethnicity: Seven categories – White, Black, Indian, East Asian, Southeast Asian, Middle Eastern, and Latino.
- Gender: Male and Female.
- Age: Nine age groups – `"0-2"`, `"3-9"`, `"10-19"`, `"20-29"`, `"30-39"`, `"40-49"`, `"50-59"`, `"60-69"` and `"70+"`.

3.1.2. Key Features

- **Balanced Representation**: Unlike many other face datasets that have a skewed representation of certain groups, FairFace ensures an equal distribution of images across the seven racial/ethnic groups, promoting fairness in facial recognition and demographic prediction tasks.
- **High-Quality Annotations**: Each image is labeled with gender, age group, and ethnicity by multiple annotators to ensure accuracy and consistency in the data.
- **Focus on Fairness**: FairFace is specifically designed to help reduce the biases that often occur in machine learning models by providing a balanced and diverse dataset, making it an important tool for training more inclusive and fair AI systems.

3.1.3. Comparison with other datasets

Dataset	# Faces	In-the-wild?	Age	Gender	Ethnicity	Balanced?	White	Asian	Black
FairFace	108K	Yes	Yes	Yes	7 Categories	Yes	Yes	E, SE	Yes
UTKFace	20K	Yes	Yes	Yes	Merged	Yes	Yes	Merged	Yes
LFW+	15K	Yes	Yes	Yes	Merged	No	Yes	Merged	Yes
IMDB-WIKI	500K	Yes	Yes	Yes	No	No	Yes	No	No
MORPH	55K	Yes	Yes	Yes	Merged	No	Yes	No	Yes
FotW	25K	Yes	Yes	Yes	No	Yes	Yes	No	No
CACD	160K	Yes	Yes	No	No	No	Yes	No	No
CelebA	200K	Yes	No	Yes	Merged	No	Yes	No	No

Table 3.1: Comparison of FairFace Dataset with Other Public Face Datasets. Sources: FairFace [1], UTKFace [2], LFW+ [3], IMDB-WIKI [4], MORPH [5], FotW [6], CACD [7], CelebA [8].

3.2. Preprocessing

Proper preprocessing is crucial for ensuring the dataset is suitable for model training.

3.2.1. Data Loading and Preparation

The facial images were loaded using their file paths specified in the dataset's annotation file. The images were opened using the Python Imaging Library (PIL) and converted to the RGB color format to maintain consistency in input channels, essential for CNN processing.

3.2.2. Data Transformations

To prepare the images for training, a transformation pipeline was applied using PyTorch's `transforms` module. This pipeline consisted of the following:

- **Tensor Conversion:** The images were converted from their original PIL format into PyTorch tensors. This process scaled pixel values from their original range of $[0, 255]$ to $[0, 1]$, ensuring better numerical stability during training.
- **Normalization:** The pixel values of the images were standardized using channel-wise mean and standard deviation values derived from the ImageNet [11] dataset:
  - **Mean:** $[0.485, 0.456, 0.406]$
  - **Standard Deviation:** $[0.229, 0.224, 0.225]$

Label Mapping

The dataset contained categorical labels for gender, age, and ethnicity, which were encoded into numerical values for compatibility with the model. The mappings were as follows:

- **Gender Mapping:**
  - `Male` $\to 0$, `Female` $\to 1$
- **Age Group Mapping:** The age groups (e.g., `"0-2"`, `"3-9"`, etc.) were mapped to integer values ranging from 0 to 8, effectively representing ordinal age groups.
- **Ethnicity Mapping:** Seven ethnicity categories (e.g., `"White"`, `"Black"`, etc.) were mapped to integers ranging from 0 to 6, ensuring compatibility with the model's output layer for ethnicity classification.

Output Format

Each preprocessed image, along with its corresponding labels for gender, age, and ethnicity, was returned as a tuple in the form:

$$(\text{image}, (\text{gender label}, \text{age group label}, \text{ethnicity label}))$$

This structure enabled efficient multitask learning, where the model shared input features but had separate outputs for each task.

Validation Data Preprocessing

For the validation dataset, the same transformation pipeline was applied. The images underwent tensor conversion and normalization to ensure consistency between training and validation data.

3.3. Model Architecture

The proposed model leverages MobileNetV2, an efficient and lightweight neural network architecture, for multitask learning. It is designed to simultaneously predict gender, age group, and ethnicity from facial images. MobileNetV2's architecture is optimized for resource-constrained environments, making it suitable for deployment on edge devices. Below is a detailed description of the model's components:

3.3.1. MobileNetV2 Architecture

MobileNetV2 is built upon **inverted residual blocks** and **linear bottlenecks**, which enhance feature representation while maintaining computational efficiency. These key components are critical for multitask learning:

- **Inverted Residual Blocks:** Use lightweight depthwise convolutions and expand-reduce operations to preserve spatial and channel-specific features with minimal computational cost.
- **Linear Bottlenecks:** Avoid non-linearities in low-dimensional spaces to prevent loss of information, ensuring high-quality feature extraction.

3.3.2. Feature Extraction Layers

The input RGB image of size $\mathbf{224 \times 224 \times 3}$ is processed through the MobileNetV2 backbone:

- **Initial Layers:** Capture low-level features such as edges and textures.
- **Intermediate Blocks:** Extract mid-level and high-level features, progressively increasing feature depth while reducing spatial dimensions.
- **Final Layers:** Provide a compact representation of the global features using a global average pooling layer.

3.3.3. Task-Specific Fully Connected Layers

After feature extraction, the global feature representation is passed through:

- **Shared Fully Connected Layer:** A layer with 256 neurons that creates a common feature space for all tasks, using ReLU activation for non-linearity.
- **Task-Specific Classifiers:**
  - **Gender Classifier:** Outputs probabilities for two classes (male and female).
  - **Age Group Classifier:** Predicts probabilities for nine age groups.
  - **Ethnicity Classifier:** Predicts probabilities for seven ethnic groups.

3.3.4. Multitask Learning Framework

MobileNetV2's shared backbone enables efficient feature sharing across tasks, while the task-specific heads specialize in predicting gender, age group, and ethnicity. This multitask approach optimizes both accuracy and computational efficiency.

3.3.5. Pipeline Summary

1. Input image passes through the MobileNetV2 backbone for hierarchical feature extraction.
2. A global average pooling layer reduces the spatial dimensions to produce a compact feature representation.
3. Features are passed through a shared fully connected layer.
4. Task-specific classifiers output predictions for gender, age group, and ethnicity.

3.3.6. Advantages of the MobileNetV2-Based Architecture

- **Computational Efficiency:** MobileNetV2's depthwise separable convolutions and bottleneck structures minimize parameters and computational complexity.
- **Compact Design:** Lightweight architecture ensures suitability for edge devices and real-time applications.
- **Multitask Learning:** Effectively balances feature sharing and task-specific specialization, improving performance across all tasks.

3.4. Model Training

For our purpose of training the model for two gender classes, nine age groups and seven ethnicities the model comprised of 3527930 trainable parameters.

The training procedure for the model involves training the model for a set number of epochs, during which the model learns three tasks: predicting gender, age group, and ethnicity from facial images. The procedure is as follows:

3.4.1. Training and Validation Data

- Training set: $\sim$ 85,000 images
- Validation set: $\sim$ 11,000 images

Training Data

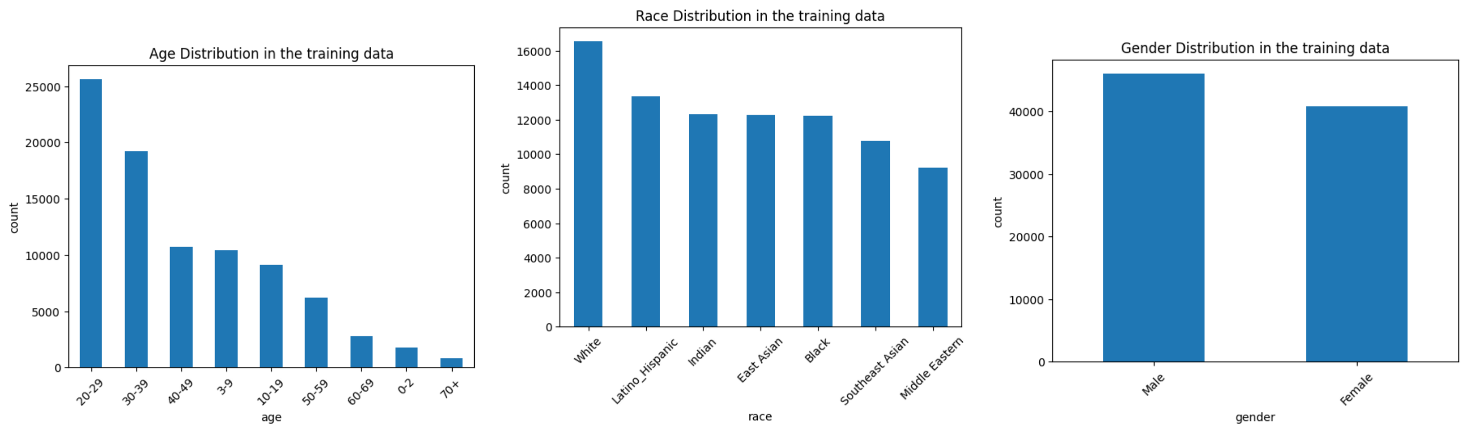


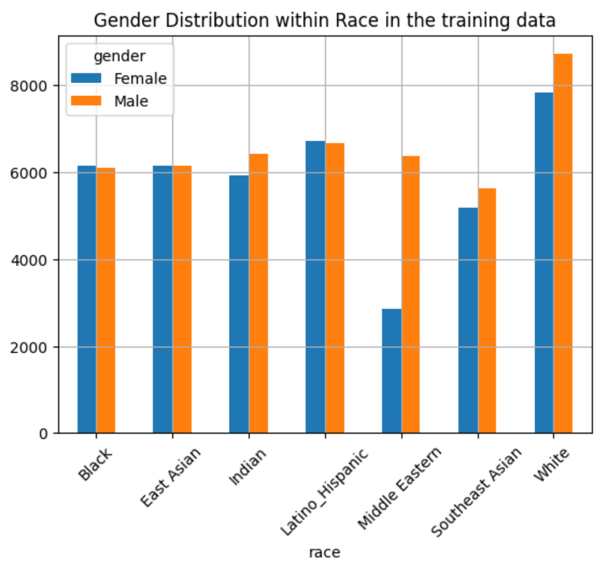
Figure 3.2: Distributions in the Training Data: Age, Ethnicity, and Gender.

The training data has significantly more people in the age group of `"20-29"` and `"30-39"`, the numbers are about the same for `"40-49"`, `"3-9"` and `"10-19"` (around 10000). For the age groups `"60-69"`, `"0-2"` and `"70+` the number of individuals is significantly less than 5000.

The distribution is fairly uniform across gender and Ethnicity. With both genders being almost equal and there is much significant variation in the number of people in different ethnicities.

	Max Ratio	**Min Ratio**	**Ratio Range**	**Coefficient of Variation**	**Entropy**	**Gini Index**
Age	0.295098	0.009707	0.285391	0.808729	2.695531	0.816217
Gender	0.530135	0.469865	0.060269	0.060269	0.997378	0.498184
Race	0.190526	0.106244	0.084282	0.169074	2.787104	0.853059

Table 3.2: Data Imbalance for Age, Gender, and Ethnicity in the training data



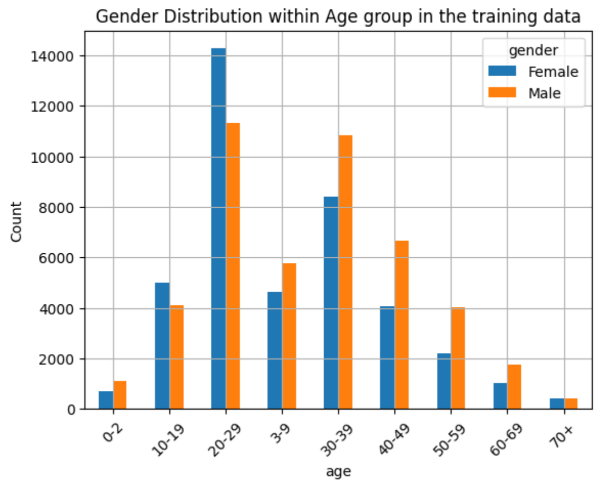


Figure 3.3: Distribution of Gender across age and ethnicity in training data

The distribution of gender is also fairly uniform across most age groups except 40-69 with significantly more males. It is also fairly even across all the ethnicity groups except Middle Eastern, where there are significantly more males than females.

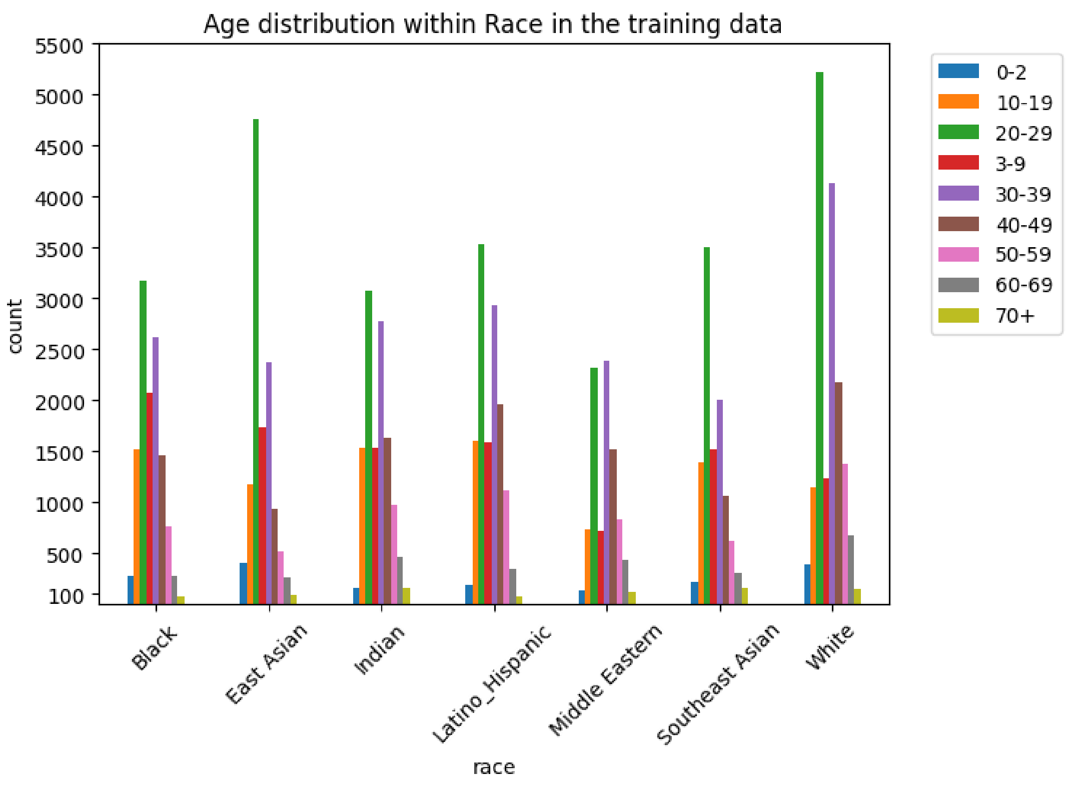


Figure 3.4: Distribution of age across ethnicity in the training data

The distribution of age groups between ethnicities is similar for all the ethnicity groups. Similar pattern is also observable in the distribution of ethnicities between age groups.

Validation Data

The distribution of gender, age groups and ethnicities in the validation data is almost identical to the training data. This is also easily observable for the distribution of gender across age groups and ethnicities as well as distribution of ethnicities across age groups.

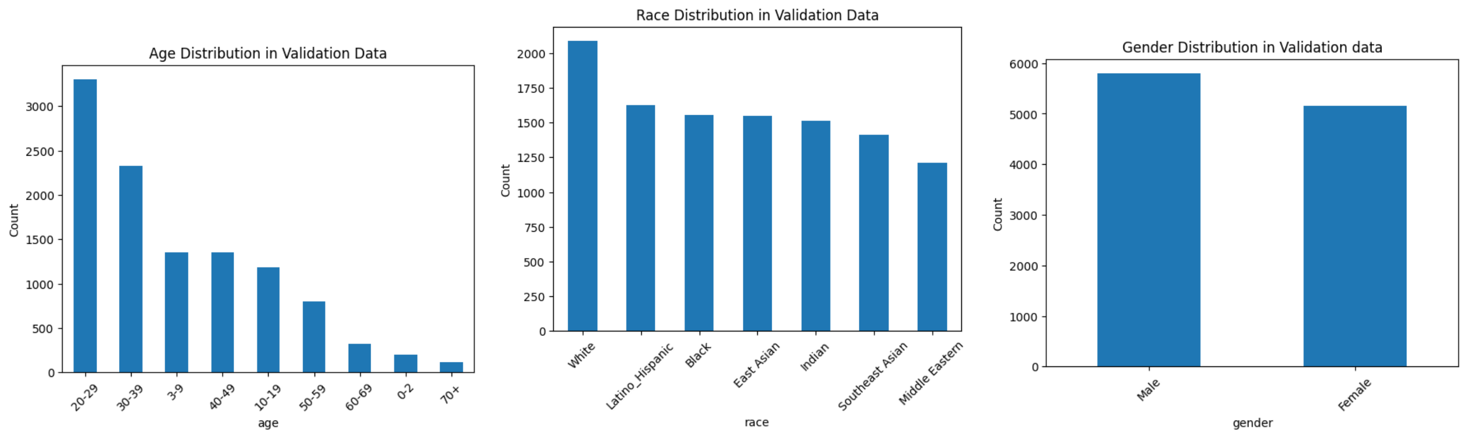
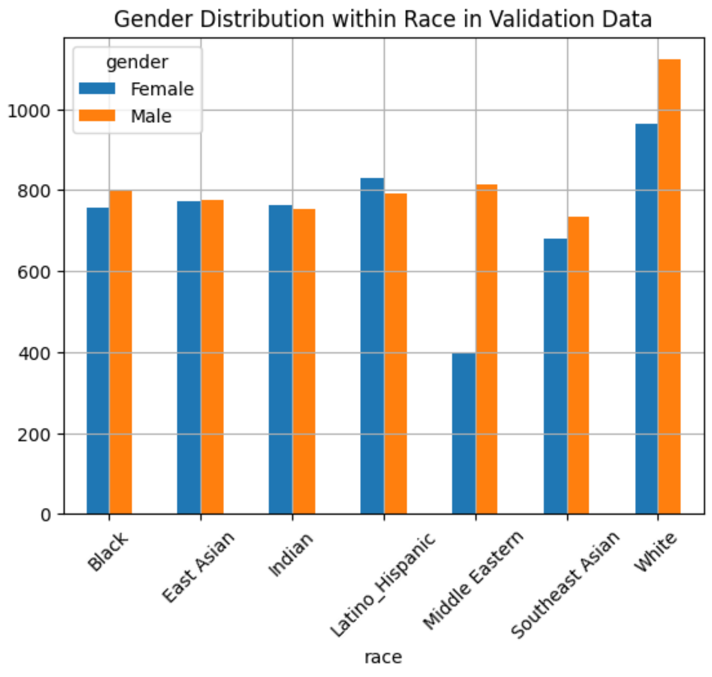


Figure 3.5: Distributions in the Validation Data: Age, Ethnicity, and Gender.

	Max Ratio	**Min Ratio**	**Ratio Range**	**Coefficient of Variation**	**Entropy**	**Gini Index**
Age	0.295098	0.009707	0.285391	0.808729	2.695531	0.816217
Gender	0.530135	0.469865	0.060269	0.060269	0.997378	0.498184
Race	0.190526	0.106244	0.084282	0.169074	2.787104	0.853059

Table 3.3: Data Imbalance for Age, Gender, and Ethnicity in the validation data



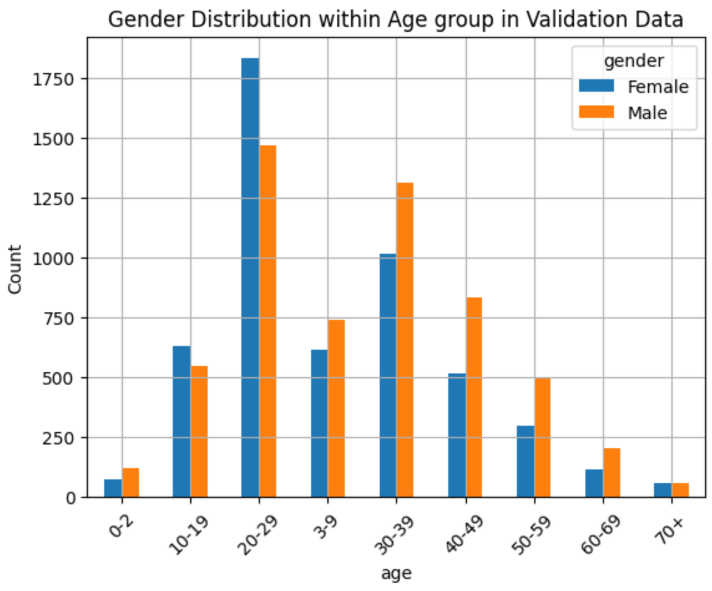


Figure 3.6: Distribution of Gender across age and ethnicity in validation data

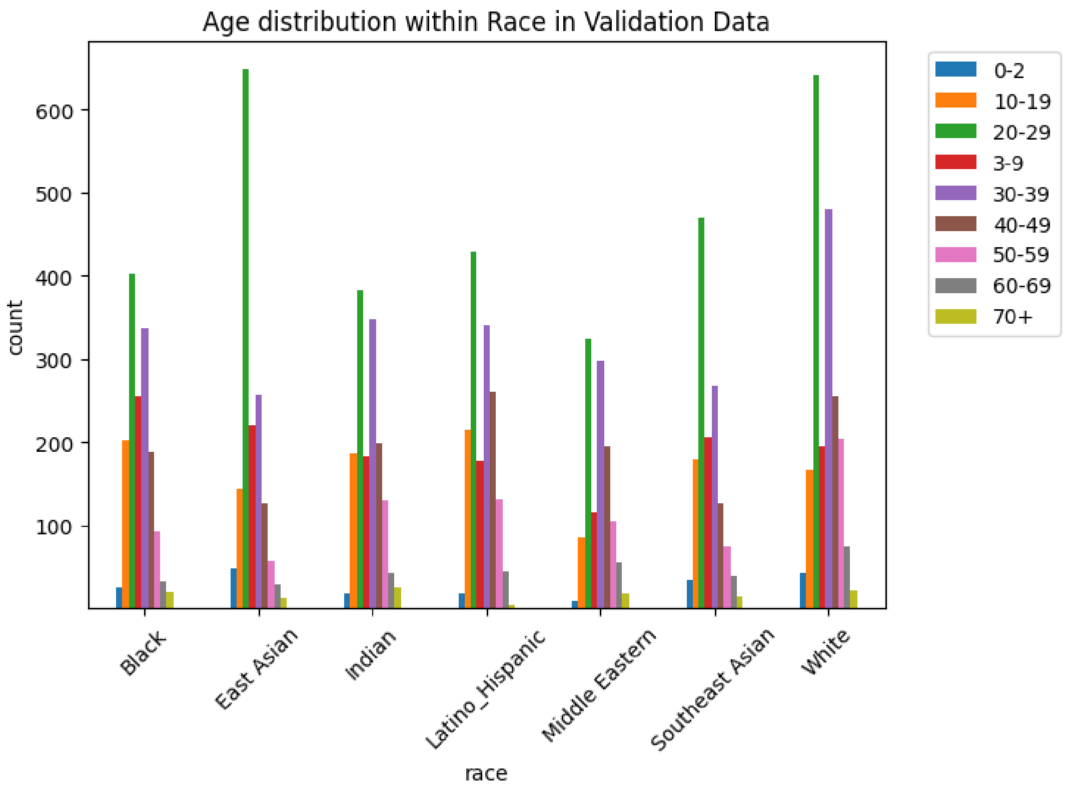


Figure 3.7: Distribution of age acorss ethnicity in validation data

3.4.2. Loss Functions

For each of the three tasks (gender, age, and ethnicity), we utilize the `CrossEntropyLoss` function. This loss function is appropriate for multi-class classification tasks, where the goal is to minimize the difference between the predicted and actual class labels. The loss functions for the three tasks are defined as follows:

- **Gender Loss:** A `CrossEntropyLoss` is used to minimize the error between the predicted and actual gender labels.
- **Age Loss:** A `CrossEntropyLoss` is employed to minimize the error between the predicted and actual age group labels.
- **Ethnicity Loss:** A `CrossEntropyLoss` is used to minimize the error between the predicted and actual ethnicity labels.

3.4.3. Optimizer

We use the Adam optimizer with a learning rate of 0.0001 to update the model's weights during training. Adam is a popular choice due to its adaptive learning rate and efficient performance in training deep learning models.

3.4.4. Training Loop

The model was trained for up to 10 epochs with early stopping based on total training loss improvement:

Epoch Steps

1. Set model to training mode.
2. Iterate over mini-batches:
   - Load images and labels to GPU.
   - Zero optimizer gradients.
   - Forward pass through the model.
   - Compute individual and total loss.
   - Backpropagate through total loss and update weights.
   - Monitor and print loss every 200 batches.
3. Evaluate on the validation set at each epoch end.
4. Implement early stopping if loss improvement is below 0.01 or there is no improvement on the test set.

3.4.5. Early Stopping

To prevent overfitting, training was halted if the decrease in training loss between consecutive epochs was less than 0.01 or there is no improvement on the test set.

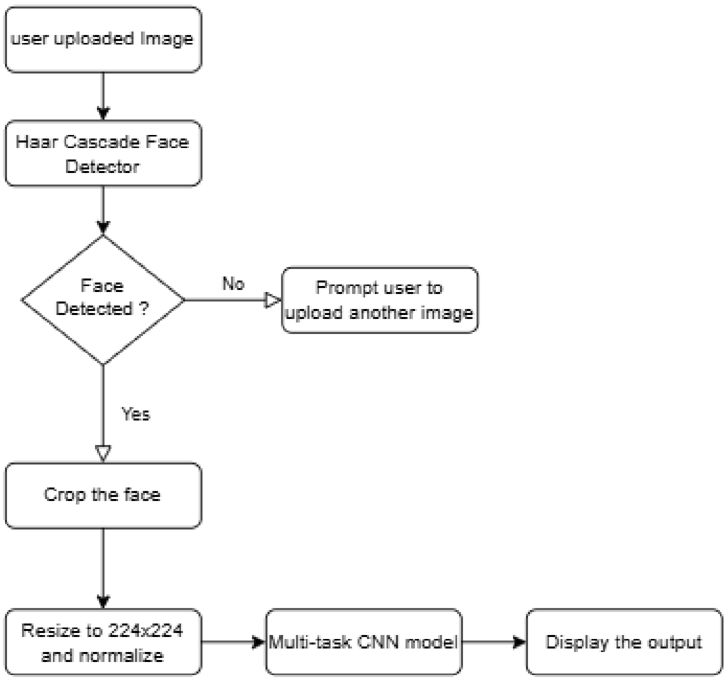
3.4.6. Monitoring and Evaluation

During training, losses for each task were tracked and averaged per epoch for both training and validation sets. Metrics reported included:

- Gender Loss
- Age Loss
- Ethnicity Loss
- Total Loss (sum of the above three losses)

3.5. Model Deployment

The trained model was deployed using Streamlit, enabling interactive and user-friendly access to demographic predictions.



user uploaded Image
Haar Cascade Face Detector
Face Detected ?
No
Prompt user to upload another image
Yes
Crop the face
Resize to 224x224 and normalize
Multi-task CNN model
Display the output

Figure 3.8: Model Pipeline

3.5.1. Model Pipeline

1. **Model Loading**:

   Load the pre-trained model from a .pth file and set it to evaluation mode.

2. **Face Detection**:

   Use a pre-trained Haar Cascade classifier to detect and crop faces in uploaded images. If no face is detected, the user is prompted to upload another image.

3. **Image Preprocessing**:

   Resize images to 224×224 pixels, convert to tensors, and normalize using ImageNet statistics.

4. **Prediction**:

   Pass the processed image through the model to obtain gender, age group, and ethnicity probabilities.

5. **Display Results**:

   Present the original image alongside predicted labels and probability distributions via bar charts.

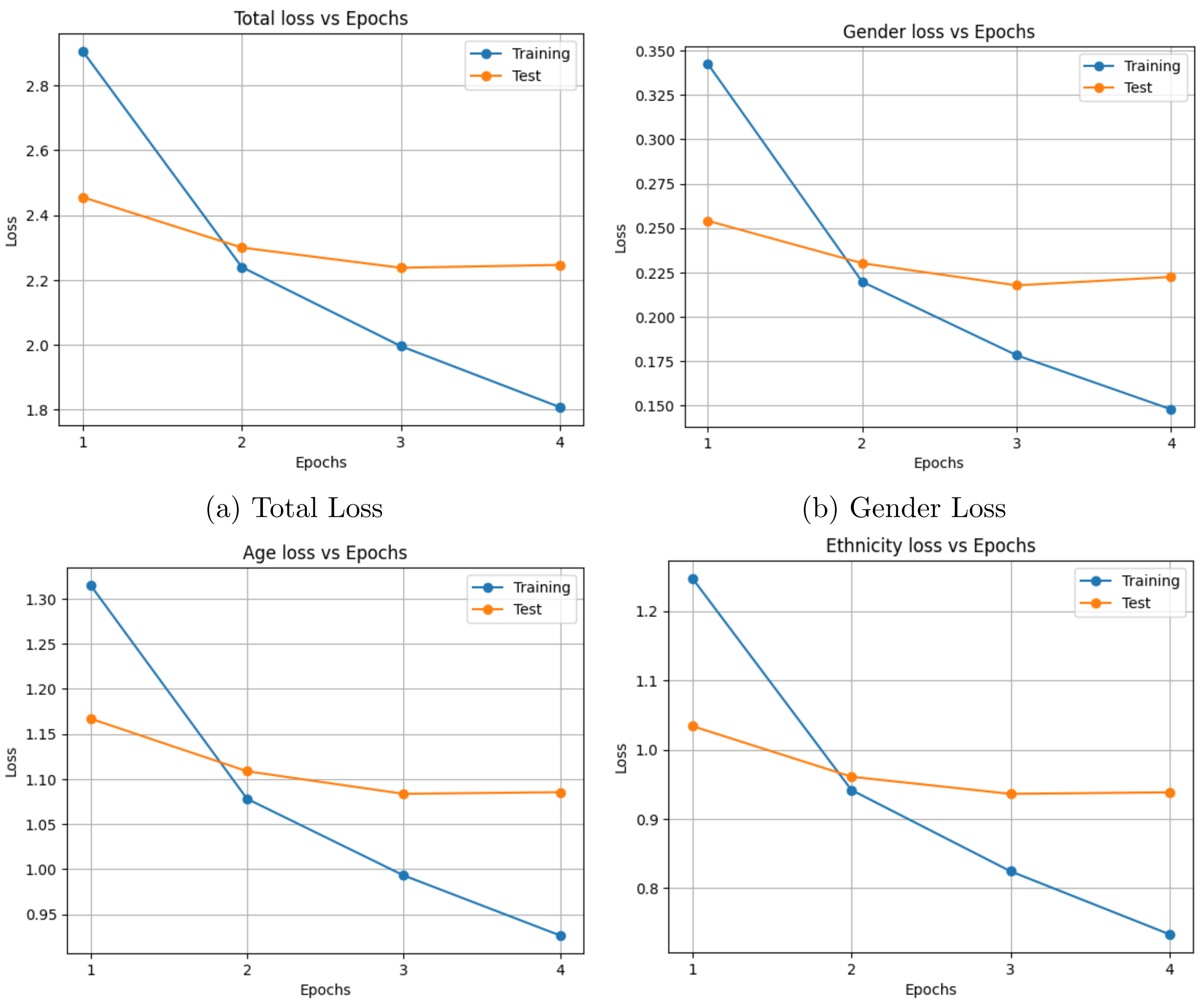
3.5.2. Technology Stack

The following technologies were used for the model deployment:

- **Streamlit**: For building the web application interface.
- **PyTorch**: For implementing and deploying the CNN model.
- **OpenCV**: For face detection using Haar Cascades.
- **Matplotlib**: For visualizing prediction probabilities.
- **NumPy and PIL**: For image manipulation and processing.

4. Results and Discussion

4.1. Training Performance



(a) Total Loss

(b) Gender Loss

(c) Age Loss

(d) Race Loss

Figure 4.1: Training losses for different aspects of our model.

The training process concluded after four epochs due to the early stopping criterion. Both training and validation losses stabilized, indicating that the model achieved a balance between learning and generalization.

4.2. Classification Accuracy

The model's accuracy metric across tasks are summarised below:

Task	Accuracy
Gender prediction	91.77%
Ethnicity prediction	66.81%
Age group prediction	55.59%

Table 4.1: Model accuracy accross tasks

4.3. Performance for each task across demographics

Confusion matrices provide deeper insights into the model's performance:

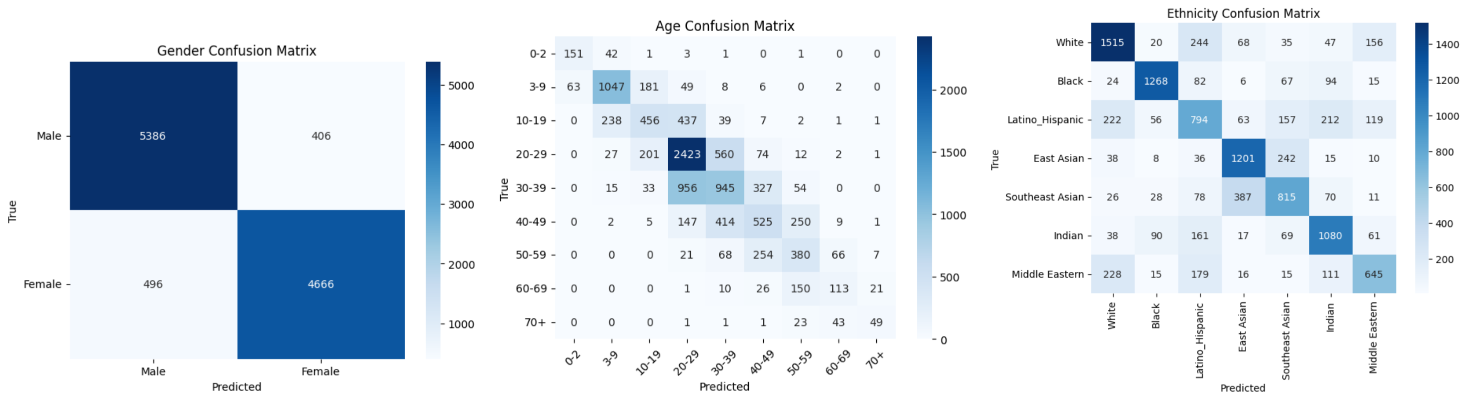


Figure 4.2: Confusion matrices for Gender, Age and Ethnicity

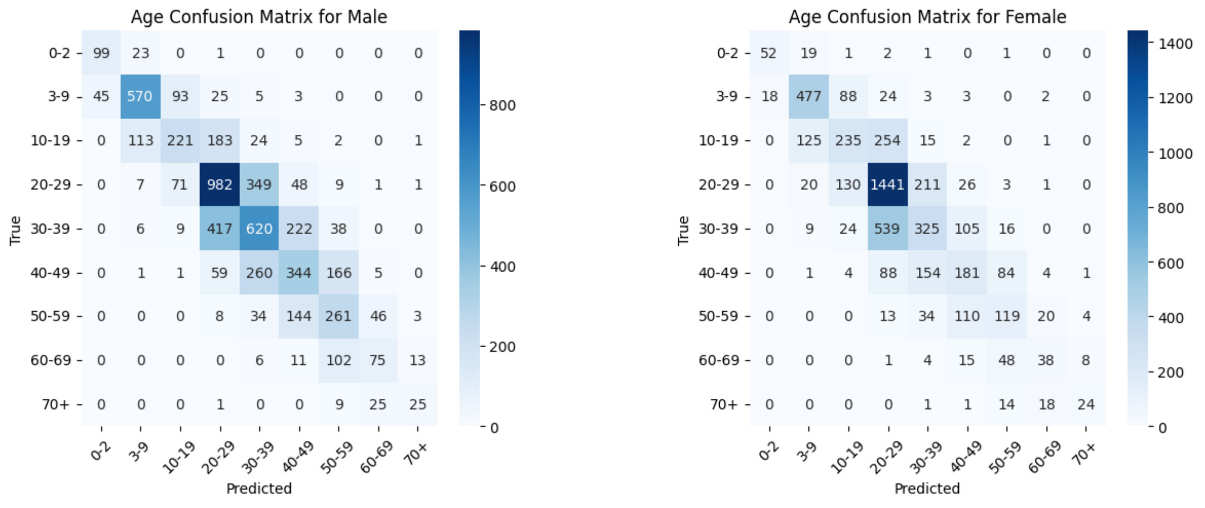


Figure 4.3: Age Confusion Matrices across gender

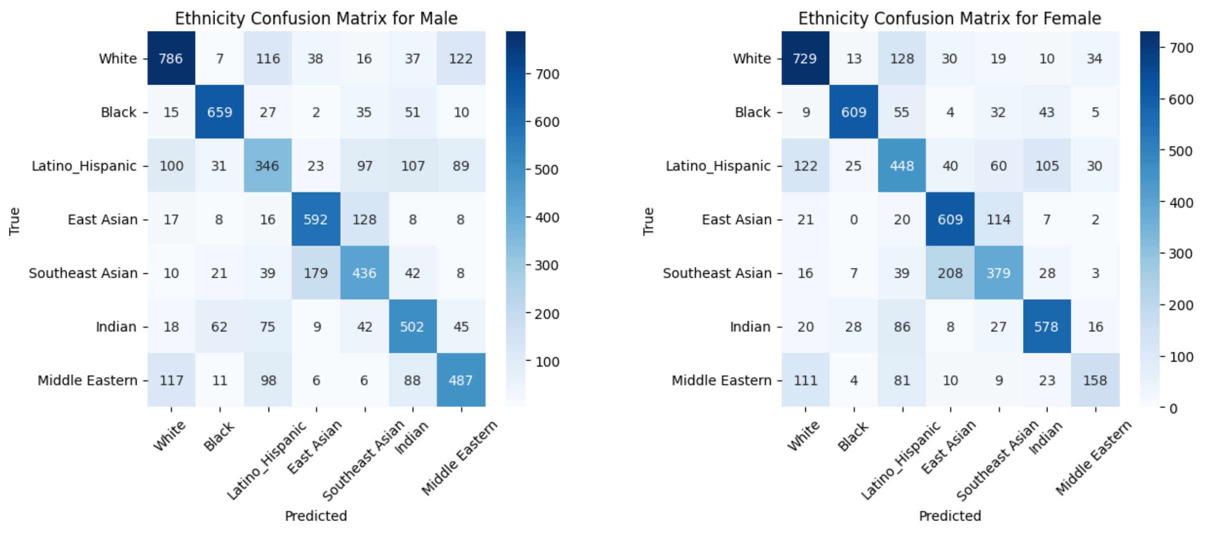


Figure 4.4: Race Confusion Matrices across gender

Confusion matrices revealed:

- **Age Group Challenges**: Misclassifications often occurred between adjacent age groups.
- **Ethnicity Misclassifications**: Overlaps between similar ethnicities, for example `"East Asian"` and `"Southeast Asian"`, `"White"` and `Latino Hispanic`.

Age Group	**White**	**Black**	**Latino/Hispanic**	**East Asian**	**Southeast Asian**	**Indian**	**Middle Eastern**
0-2	0.61	0.84	0.89	0.78	0.71	0.67	0.90
3-9	0.85	0.74	0.83	0.86	0.90	0.80	0.83
10-19	0.83	0.77	0.91	0.87	0.91	0.85	0.94
20-29	0.94	0.90	0.95	0.95	0.93	0.97	0.96
30-39	0.96	0.92	0.96	0.93	0.94	0.97	0.99
40-49	0.95	0.90	0.97	0.94	0.96	0.97	0.97
50-59	0.95	0.88	0.97	0.91	0.92	0.95	0.97
60-69	0.93	0.75	0.93	0.87	0.90	0.89	0.96
70+	0.91	0.80	1.00	0.71	0.80	0.92	0.94

Table 4.2: Gender Prediction Accuracy Across Age and Ethnicity

Gender	**White**	**Black**	**Latino/Hispanic**	**East Asian**	**Southeast Asian**	**Indian**	**Middle Eastern**
Male	0.54	0.55	0.57	0.56	0.54	0.56	0.56
Female	0.56	0.50	0.54	0.62	0.56	0.56	0.60

Table 4.3: Age Prediction Accuracy Across Gender and Ethnicity

Gender	0-2	3-9	10-19	20-29	30-39	40-49	50-59	60-69	70+
Male	0.65	0.68	0.66	0.67	0.66	0.65	0.64	0.59	0.63
Female	0.66	0.67	0.71	0.67	0.68	0.68	0.70	0.75	0.66

Table 4.4: Ethnicity Prediction Accuracy Across Gender and Age

Number of labels correctly predicted	Proportion
3	0.34
2	0.47
1	0.18
0	0.01

Table 4.5: Proportion of the number of labels correctly predicted for images

4.4. Comparison with single-task CNN models

Task	Multi-task CNN	Single-task CNN
Gender prediction	91.77%	90.72%
Ethnicity prediction	66.81%	64.83%
Age group prediction	55.59%	54.22%

Table 4.6: Comparison with single-task CNN models based on prediction accuracy

5. Conclusion

This project successfully developed and deployed a multitask CNN capable of predicting gender, age group, and ethnicity from facial images. Utilizing the FairFace dataset, the model demonstrated high accuracy in gender classification while highlighting areas for improvement in age and ethnicity predictions. High similarity between certain ethnic groups led to increased misclassification rates in Ethnicity prediction. Most of the times age groups were misclassified, they were classified as the class just below or above the actual age group. The multitask model also performed better than its single task counterpart in correctly predicting all the demographic labels, age group, ethnicity and gender. The deployment via Streamlit showcased the model's practical applicability, bridging the gap between development and real-world use cases.

While the current model excels in gender classification, significant improvements are needed for age group and ethnicity predictions. Future work will focus on advanced data handling, model optimization, fairness enhancements, and expanded deployment strategies to create a more robust and equitable demographic prediction system.

6. Appendix

6.1. Model Implementation Code

```python
import torch
import torch.nn as nn
import torch.nn.functional as F
from torchvision import models

class MultitaskMobileNet(nn.Module):
    def __init__(self, num_gender_classes, num_age_classes, num_ethnicity_classes):
        super(MultitaskMobileNet, self).__init__()
        # Load pretrained MobileNet
        self.mobilenet = models.mobilenet_v2(weights=models.MobileNet_V2_Weights.
            DEFAULT)

        # Remove the last classification layer
        self.features = self.mobilenet.features
        self.global_pool = nn.AdaptiveAvgPool2d(1)
        self.last_channel = self.mobilenet.last_channel

        # Task-specific heads
        self.gender_head = nn.Linear(self.last_channel, num_gender_classes)
        self.age_head = nn.Linear(self.last_channel, num_age_classes)
        self.ethnicity_head = nn.Linear(self.last_channel, num_ethnicity_classes)

    def forward(self, x):
        # Shared feature extraction
        x = self.features(x)
        x = self.global_pool(x)
        x = torch.flatten(x, 1)

        # Task-specific outputs
        gender_out = self.gender_head(x)
        age_out = self.age_head(x)
        ethnicity_out = self.ethnicity_head(x)

        return gender_out, age_out, ethnicity_out
```

6.2. Model Training Code

```python
# Define loss functions for each task
criterion_gender = nn.CrossEntropyLoss()
criterion_age = nn.CrossEntropyLoss()
criterion_ethnicity = nn.CrossEntropyLoss()

# Define optimizer
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)

num_epochs = 10
weights = {}
losses = []
test_losses = []
losses_gender = []
losses_age = []
losses_ethnicity = []
test_losses_gender = []
test_losses_age = []
test_losses_ethnicity = []

for epoch in range(num_epochs):
    model.train()
    running_loss = 0.0
    running_loss_gender = 0.0
    running_loss_age = 0.0
    running_loss_ethnicity = 0.0

    batch_idx = 1
    for images, (gender_labels, age_labels, ethnicity_labels) in train_loader:
        images = images.to(device)
        gender_labels = gender_labels.to(device)
        age_labels = age_labels.to(device)
        ethnicity_labels = ethnicity_labels.to(device)

        optimizer.zero_grad()

        # Forward pass
        gender_output, age_output, ethnicity_output = model(images)

        # Compute individual losses and combine them
        gender_loss = criterion_gender(gender_output, gender_labels)
        age_loss = criterion_age(age_output, age_labels)
        ethnicity_loss = criterion_ethnicity(ethnicity_output, ethnicity_labels)

        total_loss = gender_loss + age_loss + ethnicity_loss
        total_loss.backward()
        optimizer.step()

        if batch_idx % 200 == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f} \tGender Loss: {:.6f
                } \tAge Loss: {:.6f}\tEthnicity Loss: {:.6f}'.format(
                epoch+1, batch_idx * len(gender_labels), len(train_dataset),
                100. * batch_idx / len(train_loader), total_loss.item(),
                gender_loss.item(), age_loss.item(), ethnicity_loss.item()))
        batch_idx += 1

        running_loss += total_loss.item()
        running_loss_gender += gender_loss.item()
        running_loss_age += age_loss.item()
        running_loss_ethnicity += ethnicity_loss.item()

    losses.append(running_loss/len(train_loader))
    losses_gender.append(running_loss_gender/len(train_loader))
    losses_age.append(running_loss_age/len(train_loader))
    losses_ethnicity.append(running_loss_ethnicity/len(train_loader))

    print(f"Epoch [{epoch+1}/{num_epochs}], Training Loss: {running_loss/len(train_loader):.4f}, Gender Loss: {running_loss_gender/len(train_loader):.4f}, Age Loss: {running_loss_age/len(train_loader):.4f}, Ethnicity Loss: {running_loss_ethnicity/len(train_loader):.4f}")

    running_loss = 0.0
    running_loss_gender = 0.0
    running_loss_age = 0.0
    running_loss_ethnicity = 0.0

    for images, (gender_labels, age_labels, ethnicity_labels) in val_loader:

        images = images.to(device)
        gender_labels = gender_labels.to(device)
        age_labels = age_labels.to(device)
        ethnicity_labels = ethnicity_labels.to(device)

        gender_output, age_output, ethnicity_output = model(images)

        # Compute individual losses and combine them
        gender_loss = criterion_gender(gender_output, gender_labels)
        age_loss = criterion_age(age_output, age_labels)
        ethnicity_loss = criterion_ethnicity(ethnicity_output, ethnicity_labels)

        total_loss = gender_loss + age_loss + ethnicity_loss

        running_loss += total_loss.item()
        running_loss_gender += gender_loss.item()
        running_loss_age += age_loss.item()
        running_loss_ethnicity += ethnicity_loss.item()

    test_losses.append(running_loss/len(val_loader))
    test_losses_gender.append(running_loss_gender/len(val_loader))
    test_losses_age.append(running_loss_age/len(val_loader))
    test_losses_ethnicity.append(running_loss_ethnicity/len(val_loader))
    print(f"Epoch [{epoch+1}/{num_epochs}], Test Loss: {running_loss/len(val_loader):.4f}, Gender Loss: {running_loss_gender/len(val_loader):.4f}, Age Loss: {running_loss_age/len(val_loader):.4f}, Ethnicity Loss: {running_loss_ethnicity/len(val_loader):.4f}")

    evaluate(model, val_loader)
    weights[epoch] = model.state_dict()

    # stopping criteria
    if len(losses) > 1 and ((losses[-2] - losses[-1] < 0.01) or (test_losses[-2] < test_losses[-1])):
        print("Early stopping triggered")
        break
```

Bibliography

[1] Kimmo Karkkainen and Jungseock Joo, ”FairFace: Face Attribute Dataset for Balanced Race, Gender, and Age for Bias Measurement and Mitigation”, *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[2] UTKFace Dataset: Z. Zhang et al. *Age, Gender and Ethnicity Recognition with Deep Learning*.

[3] LFW+ Dataset: G. Huang et al. *Labeled Faces in the Wild*.

[4] IMDB-WIKI Dataset: R. Rothe et al. *Deep Expectation of Real and Apparent Age from a Single Image*.

[5] MORPH Dataset: K. Ricanek et al. *Morph: Longitudinal Data*.

[6] FotW Dataset: H. Kim et al. *Faces of the World: Diversity in Face Recognition*.

[7] CACD Dataset: B. Chen et al. *Cross-Age Reference Coding for Age-Invariant Face Recognition*.

[8] CelebA Dataset: Z. Liu et al. *Deep Learning Face Attributes in the Wild*.

[9] Paul Viola and Michael Jones, ”Rapid Object Detection using a Boosted Cascade of Simple Features”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[10] Timo Ojala, Matti Pietikainen, and Topi Maenpaa, ”Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, ”ImageNet Classification with Deep Convolutional Neural Networks”, *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[12] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. & Adam, H. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,2017*

[13] Rich Caruana, ”Multitask Learning”, *Machine Learning*, 1997.