**The National University of Computer and Emerging Sciences**

**Islamabad Campus**



# **DriverPAL**

# **Digital Image Processing**

**Hamza Asad 22i-1908 / Taha Rasheed 22i-2009/ Shafay Siddiq 22i-2007**

**DIP B**

**Contents**

**1. Age model**

**1.1. Dataset Prep 2**

**1.2. Model Architecture 3**

**1.3. Validation Metrics 6**

**2. Gender Model**

**2.1. Dataset Prep 8**

**2.2. Architecture 9**

**2.3. Final accuracy 11**

**3. Emotion model**

**3.1. Dataset Prep 12**

**3.2. Model Architecture 12**

**3.3. Validation Metrics 15**

**4. Application 17**

**1. Age Model**

**1. Preparing dataset for model**

**Data Set :** <https://www.kaggle.com/datasets/chiragsaipanuganti/morph?select=Dataset>

**1.1 Face Crop**

* **What**:  
  Each input is assumed to be a tight frontal‐face crop (224×224) showing the full head.
* **Why**:  
  MORPH-II images are tightly aligned. Feeding full-scene photos introduces unseen background noise. Cropping (via MTCNN or Haar Cascade) ensures only face-relevant features are learned.

**1.2 Resizing & RGB Normalization**

* **What**:
  + Resize face crop to **224×224 pixels**.
  + Convert to PyTorch tensor with values in **[0,1]**.
  + Normalize using **ImageNet mean/std**:
    - Mean: [0.485, 0.456, 0.406]
    - Std: [0.229, 0.224, 0.225]
* **Why**:
  + 224×224 matches **ResNet-50’s input requirement**.
  + Normalization ensures the model sees familiar data scales, accelerating training and preventing "dead filters."

**1.3 Classical Feature Augmentation (CLAHE, LBP, HOG)**

* **Overview**:  
  Enhance RGB images with handcrafted grayscale-derived texture maps.

**1.3.1 CLAHE (Contrast-Limited Adaptive Histogram Equalization)**

* **What**:  
  Local histogram equalization boosting fine details like wrinkles.
* **Why**:  
  Highlights aging-related microstructures.

**1.3.2 LBP (Local Binary Patterns)**

* **What**:  
  Each pixel thresholds its 8 neighbors to create a texture code (0–255).
* **Why**:  
  Captures micro-textures like pores and fine lines, critical for aging patterns.

**1.3.3 HOG (Histogram of Oriented Gradients)**

* **What**:  
  Captures gradient orientations across local 16×16 cells.
* **Why**:  
  Encodes global structure — face contours, cheekbones, jawlines.

**2. Six-Channel Fusion**

* After feature extraction, we combine:
  1. 3 RGB channels
  2. 1 CLAHE map
  3. 1 LBP map
  4. 1 HOG map
* Result: A **6×224×224 tensor** input.
* **Why**:  
  Enables end-to-end learning by merging color, texture, and structure cues into a unified representation.

**3. Model Architecture: ResNet-50 Adapted for Regression**

**3.1 Choice of ResNet-50**

* **Depth vs. Compute**:  
  Deep enough (≈25M parameters) to learn complex face features, yet feasible to train on a single GPU.
* **Pretraining**:  
  Initialized with ImageNet weights to transfer learned edges, textures, and patterns.

**3.2 Adapting for 6-Channel Input**

* **Original**:  
  ResNet-50’s stem expected 3 input channels.
* **Modification**:  
  Replace initial convolutional layer with a new **7×7 stride-2 convolution** accepting **6 channels**.  
  Rest of ResNet-50 remains intact to benefit from pretrained downstream layers.

**3.3 Regression Head**

* **Original**:  
  Fully-connected layer for 1,000 ImageNet classes.
* **New**:  
  Simple **linear layer** mapping ResNet’s **2048-dim feature vector** to a single real-valued age prediction.
* **Note**:  
  No activation function; ages are unbounded positive reals.

**4. Training Strategy**

**4.1 Loss: Smooth L1 (Huber Loss)**

* **Definition**:  
  Quadratic for small residuals, linear for large residuals.
* **Why**:  
  Combines benefits of MAE (outlier-robustness) and MSE (strong penalty on large errors) for faster and more stable convergence.

**4.2 Optimizer: Adam**

* **Why**:  
  Adaptive learning rates per-parameter make it highly effective for heterogeneous visual features.

**4.3 Scheduler: Cosine Annealing**

* **What**:  
  Smooth decay of learning rate using a cosine function.
* **Why**:  
  Warm restarts and smooth decay enhance convergence to better minima, improving final MAE.

**4.4 Oversampling by Age**

* **Problem**:  
  MORPH-II is age-imbalanced (biased toward younger faces).
* **Solution**:  
  Use a **WeightedRandomSampler** inversely proportional to age-bin frequencies for balanced minibatches.

**5. Data Augmentation (Training Only)**

* **RandomHorizontalFlip**:  
  Handles left-right symmetry.
* **RandomRotation (±10°)**:  
  Makes model robust to slight pose changes.
* **ColorJitter**:  
  Simulates lighting variations.
* **Why**:  
  Prevents overfitting to controlled studio-like MORPH-II conditions.

**6. Validation Metrics**

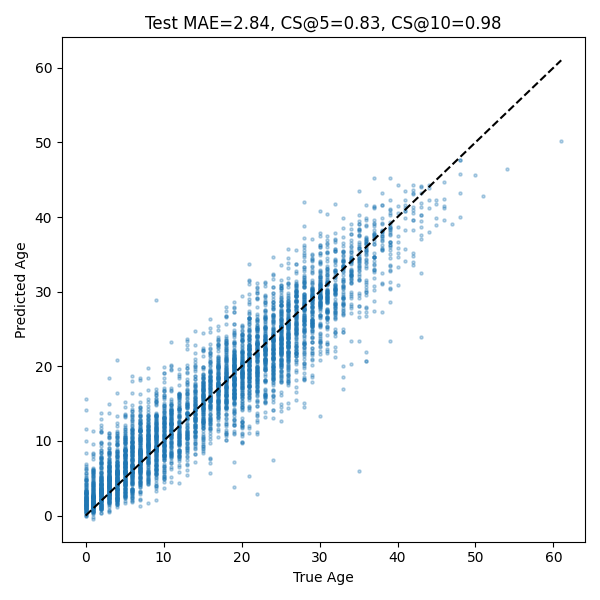
**6.1 MAE (Mean Absolute Error)**

* **Definition**:  
  Average absolute prediction error.
* **Result**:  
  **2.36 years** average error on validation set.

**6.2 Cumulative Score**

| **Metric** | **Result** |
| --- | --- |
| **CS@5** | 83% within ±5 years |
| **CS@10** | 98% within ±10 years |

* **Interpretation**:  
  High tolerance-based accuracy reflects reliable aging predictions.



**7. Inference Pipeline**

* **Steps**:
  1. Detect and crop face using MTCNN or Haar.
  2. Resize, normalize, generate CLAHE, LBP, HOG.
  3. Concatenate into 6-channel tensor.
  4. Load trained AgeResNet50 model.
  5. Predict age scalar.
  6. (Optional) Display prediction with a ±5-year tolerance for user clarity.

**8. Why This Design Works**

* **Handcrafted + Learned Features**:  
  Combining clinical-like maps (CLAHE, LBP, HOG) with CNN-learned embeddings strengthens feature richness.
* **ResNet-50**:  
  Deep yet efficient, pretrained for strong generalization.
* **Robust Loss & Balanced Sampling**:  
  Smooth L1 loss and oversampling ensure stability across skewed datasets.
* **Augmentation + Cosine Annealing**:  
  Improved generalization and smooth convergence.

**2. Gender Model**

**1. Introduction**

This project involves building a gender classification system using deep learning techniques, particularly a Convolutional Neural Network (CNN). The model is trained to classify images of individuals as either male or female based on facial features. The implementation includes dataset handling, preprocessing, augmentation, model design, and evaluation steps.

**2. Dataset Details**

**Source and Structure**

* The dataset is was provide on <https://www.kaggle.com/datasets/ashishjangra27/gender-recognition-200k-images-celeba>
* After extraction, images are divided into two main categories:
  + /male
  + /female
* The data is manually split into:
  + Training (80% of the data)
  + Validation (20% of the data)

**Sample Size and Organization**

* Exact counts depend on the number of valid and clean images post-processing.
* A dictionary-based structure is used to manage image paths by category.
* The dataset is balanced post-cleaning to ensure equal representation of both classes during training.

**Access**

* The dataset is locally extracted; no external online link is referenced in the notebook.

**3. Data Validation and Cleaning**

Before training, the images undergo a cleaning process to ensure only high-quality samples are used:

* Corrupt or unreadable images are removed using OpenCV.
* Images smaller than 50x50 pixels are discarded.
* Duplicate image paths are filtered out to prevent overfitting or redundancy.

**4. Dataset Balancing**

To prevent model bias toward a dominant class, both the male and female datasets are trimmed to the size of the smaller category. This ensures a balanced training dataset, which is critical for effective binary classification.

**5. Image Preprocessing and Augmentation**

**Preprocessing**

* All images are resized to 100x100 pixels for uniform input dimensions.
* Images are normalized by rescaling pixel values to the range [0, 1].

**Augmentation**

* The model uses Keras’s ImageDataGenerator to enhance the training data with:
  + Rotation
  + Width and height shifts
  + Horizontal flipping
  + Zoom
  + Brightness adjustment

These augmentations increase data variability, improving the model’s generalization on unseen images.

**6. CNN Model Architecture**

The classification model is built using the Keras Sequential API, designed for binary output (male/female):

**Model Layers**

* Two convolutional layers with ReLU activation and max pooling to extract features and reduce spatial dimensions.
* A flattening layer followed by a fully connected dense layer for classification learning.
* Dropout is applied to prevent overfitting.
* A final sigmoid-activated dense layer outputs a probability representing the gender class.

**Training Configuration**

* Optimizer: **Adam**
* Loss Function: **Binary Crossentropy**
* Metrics: **Accuracy**
* Number of Epochs: **10**
* Batch Size: **32**

**7. Model Summary Display**

After training, the model structure is displayed using model.summary(). This shows the number of layers, parameters, output shapes, and the total number of trainable parameters.

**8. Image Loading and Labeling**

For internal visualization and understanding:

* Images are loaded from both male and female training folders.
* Each image is assigned a binary label: 1 for male, 0 for female.
* Images and labels are stored in lists for easy sampling and display.

**9. Displaying Random Training Images**

A set of four random training images are displayed using Matplotlib:

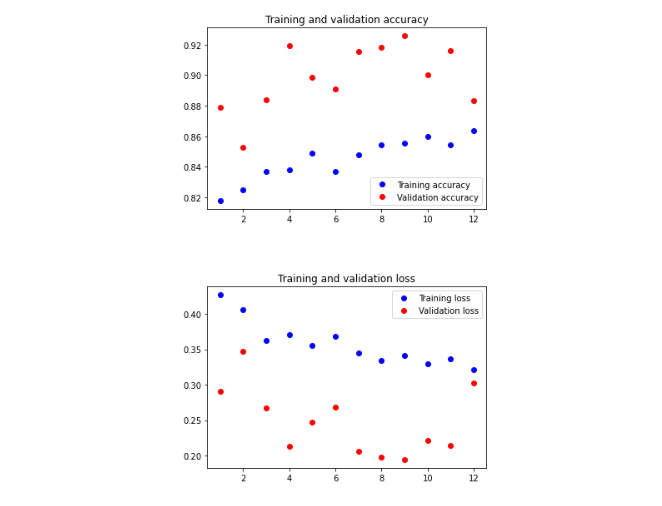
* Images are converted from BGR (OpenCV format) to RGB for accurate visualization.
* Each image is shown with a title indicating its gender label (Male/Female).

**10. Limitations and Future Work**

**Future Improvements**

* Implement evaluation plots and metrics for deeper insight.
* Use callbacks like early stopping and learning rate scheduling.
* Explore transfer learning using pre-trained models such as MobileNet, VGG16, or ResNet.
* Build an interactive web interface or API for real-time gender classification.
* Automate the dataset preprocessing and augmentation pipeline for scalability.

**Final Test Accuracy : 92.8**



**3. Emotion Model**

**1. Introduction**

This report compares two deep learning models trained for facial emotion recognition on the **FER-2013** dataset <https://www.kaggle.com/datasets/msambare/fer2013/data>. The goal is to identify the reasons for the observed performance differences, focusing on the role of preprocessing, network architecture, and training strategies.

The two models analyzed are:

* **Base + CLAHE Model** (base\_augmented\_model.ipynb)
* **Optimal Detector Model** (emotion-detector-optimal.ipynb)

**2. Dataset: FER-2013**

FER-2013 is a widely used benchmark dataset for facial expression recognition. It consists of grayscale 48×48 pixel images labeled across 7 emotions (angry, disgust, fear, happy, sad, surprise, neutral).

**Important Note:**

* The FER-2013 images are **already preprocessed** (cropped, aligned, resized, and normalized).
* Thus, additional heavy preprocessing, such as applying **CLAHE** (Contrast-Limited Adaptive Histogram Equalization), can lead to **over-processing**, resulting in degraded image quality and making learning more difficult for CNN models.

**3. Data Preprocessing Methods**

| **Feature** | **Base + CLAHE Model** | **Optimal Detector Model** |
| --- | --- | --- |
| **Histogram Equalization** | Applied CLAHE to all images manually | No additional equalization |
| **Rescaling** | rescale=1./255 in generator | rescale=1./255 in generator |
| **Data Augmentation** | Rotations, shifts, flips | Rotations, shifts, zooms, flips |
| **Train/Validation Split** | 80/20 split during data generation | 80/20 split during data generation |

**Before After**

|  |  |
| --- | --- |
|  |  |

**3.1 Effect of Over-Processing in Base + CLAHE**

Applying CLAHE to FER-2013 images — which were already normalized and enhanced — introduced **excessive contrast and noise**.

* It altered important facial textures that the model needs to learn.
* This made the learning process harder, as the model was forced to compensate for unnatural contrast distributions.

In contrast, the **Optimal Detector** trained directly on FER-2013's original images benefited from the dataset’s built-in preprocessing, resulting in **more stable learning**.



**4. Model Architectures**

**4.1 Base + CLAHE CNN**

* **Depth**: 2 convolutional blocks
* **Features**:
  + Convolution → BatchNorm → Activation → Pooling → Dropout
  + Global Average Pooling → Dense Layer (256 units) → Output
* **Max Filters**: 128
* **Regularization**: Dropout (30–50 %)

**4.2 Optimal Detector CNN**

* **Depth**: 3 convolutional blocks
* **Features**:
  + Convolution → Activation → BatchNorm → Pooling → Dropout
  + Deeper and wider convolutional layers (up to 512 filters)
  + Two Dense layers (256 → 512 units) with BatchNorm and Dropout
* **Regularization**: Dropout (25 %) + L₂ weight decay (0.01)

**5. Training Strategies**

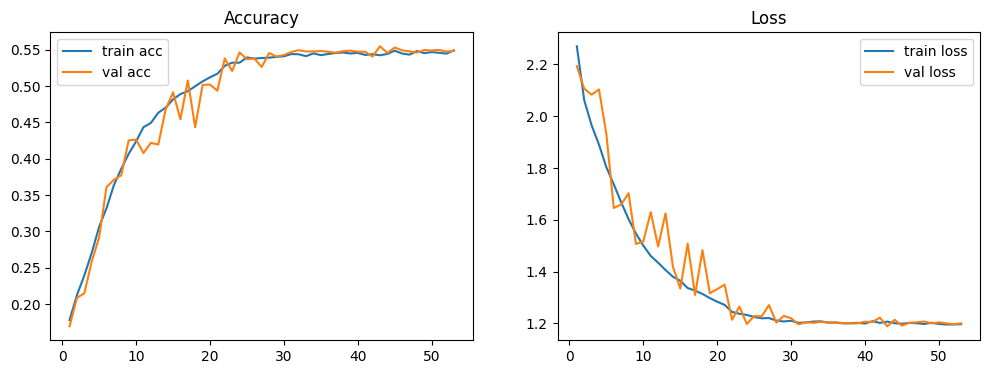
| **Feature** | **Base + CLAHE Model** | **Optimal Detector Model** |
| --- | --- | --- |
| **Optimizer** | Adam (default LR = 0.001) | Adam (lower LR = 0.0001) |
| **Learning Rate Scheduling** | ReduceLROnPlateau callback | ReduceLROnPlateau callback |
| **Batch Normalization** | Only in early layers | Used after every block and Dense layers |
| **Regularization Techniques** | Only Dropout | Dropout + L₂ regularization |
| **Epochs** | 60 | 50–60 |

**6. Results**

| **Metric** | **Base + CLAHE Model** | **Optimal Detector Model** |
| --- | --- | --- |
| **Training Accuracy** | ~55 % | ~72 % |
| **Validation Accuracy** | ~54 % | ~64 % |

**Observations:**

* The Optimal Detector shows **better generalization**, owing to stronger architecture and more appropriate preprocessing.



**7. Analysis and Discussion**

**7.1 Impact of Preprocessing**

* The FER-2013 dataset is already well-preprocessed; adding CLAHE resulted in **over-processing**, distorting facial features.
* The Optimal Detector model, by relying on minimal preprocessing and internal normalization (BatchNorm layers), **learned more natural representations**.

**7.2 Architectural Superiority**

* **Depth and Width**: Optimal Detector uses more filters (up to 512) and deeper convolutional stacks.
* **Normalization**: BatchNorm after every convolution layer helped stabilize gradients and reduce internal covariate shift.
* **Regularization**: The combination of Dropout and L₂ weight decay allowed the larger model to avoid overfitting.

**7.3 Training Stability**

* Using a lower learning rate (1e-4) in the Optimal Detector enabled smoother convergence.
* ReduceLROnPlateau further helped fine-tune the model near minima.

**8. Conclusion**

The **Optimal Detector** model achieved significantly higher validation accuracy on FER-2013 compared to the **Base + CLAHE** model.  
This improvement can be attributed to:

* **Better architectural design**: deeper network, higher capacity, consistent use of normalization.
* **Appropriate preprocessing**: avoiding over-processing already-cleaned images.
* **Improved training strategies**: lower learning rate, better regularization.

**Key Lesson:**

*When working with datasets like FER-2013 that are already preprocessed, additional aggressive preprocessing (such as CLAHE) can be counterproductive. Instead, emphasis should be placed on building stronger models capable of learning robust features directly.*

**4. Final Application**

**Driver PAL** is a real-time, AI-powered driver assistance system designed to enhance road safety and personalize the in-vehicle experience using computer vision and deep learning models

Driver PAL uses webcam-based face detection to:

* 🔹 **Classify Gender** (Male / Female)
* 🔹 **Detect Age Group** (Child / Adult / Senior)
* 🔹 **Recognize Emotions** (Happy, Sad, Angry, etc.)

These insights drive intelligent UI adjustments and emotional support actions while driving.

**Key Features**

* **Real-Time Face Analysis**: Continuous detection and processing of facial cues using OpenCV.
* **Deep Learning Models**:
  + TensorFlow-based **Gender** and **Emotion** classifiers
  + PyTorch-based **Age Group** model
* **Emotion-Based Music Recommendations**: Spotify integration to boost mood during stress or sadness.
* **Dynamic UI Personalization**: Interface adapts based on detected age and gender.
* **Conversational Support**: Voice interaction is triggered during emotional distress (e.g., anger, sadness) for safety reassurance.

**Purpose**

To promote **emotional well-being**, **driver focus**, and **personalized engagement** using AI. The application proactively responds to negative emotional states and enhances comfort through customized interactions.

**Impact & Future Scope**

Driver PAL showcases a powerful fusion of machine learning and user empathy, setting a foundation for emotionally aware automotive systems. Future enhancements may include:

* Fatigue and drowsiness detection
* Expanded speech-based interaction
* Long-term emotional pattern tracking

