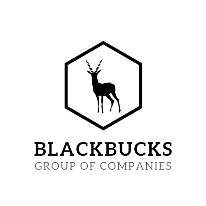
**Facial Gender Recognition: A Computer Vision Approach Using Convolutional Neural Networks**

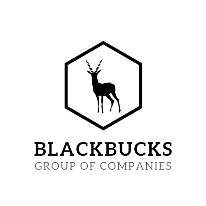
**Project Documentation**

**NAME OF STUDENT: AASHRITHA SURA**

**REGISTRATION NO: 23BCE7105**

**INSTRUCTOR: SREENI JILLA**

**VIT-AP BLACKBUCKS**

**Table of Contents**

**1. Introduction**

1.1 Importance of Gender Classification  
1.2 Challenges in Gender Classification  
1.3 Why Deep Learning?  
1.4 Project Objective

# **2. Literature Review**

2.1 Traditional Approaches  
2.2 Deep Learning Breakthroughs  
2.3 Attention Mechanisms

# **3. Dataset Description**

3.1 UTKFace Dataset  
3.2 Data Distribution  
3.3 Preprocessing Steps and Libraries Used

# **4. Methodology**

4.1 CNN Architecture  
4.2 Training Protocol  
4.3 Data Augmentation

# **5. Results & Evaluation**

5.1 Performance Metrics  
5.2 Sample Predictions  
5.3 Comparison with Baselines

# **6. Conclusion & Future Work**

6.1 Key Findings  
6.2 Future Directions

**7. References**

**8**.**Appendices**

# **Introduction**

Automatic gender classification from facial images represents a critical challenge in computer vision, with applications spanning security systems, demographic analysis, and personalized marketing. While traditional approaches relied on manual feature engineering, modern Convolutional Neural Networks (CNNs) can automatically learn discriminative facial features. This project develops a robust gender classification system using the UTKFace dataset, which provides over 20,000 labeled facial images with balanced gender distribution across diverse age groups and ethnicities. The implementation specifically addresses real-world challenges including varying illumination conditions, facial occlusions, and pose variations, while deliberately excluding age prediction to maintain focus on optimizing binary gender classification accuracy.

**1.1 Importance of Gender Classification**

Gender classification is widely used in:

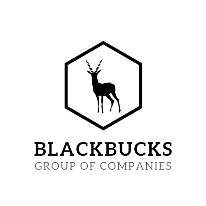
* **Retail & Marketing**: Personalized advertisements based on demographics.
* **Security & Surveillance**: Enhancing facial recognition systems.
* **Healthcare & Psychology**: Studying gender-based behavioral patterns.
* **Human-Computer Interaction**: Gender-aware virtual assistants.

**1.2 Challenges in Gender Classification**

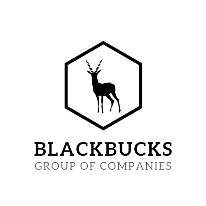
Despite advancements, several challenges persist:

* **Variations in Lighting & Pose**: Uneven illumination and non-frontal faces reduce accuracy.
* **Age & Ethnicity Bias**: Models may perform poorly on underrepresented groups.
* **Occlusions & Facial Expressions**: Accessories (glasses, hats) and expressions affect detection.

**1.3 Why Deep Learning?**

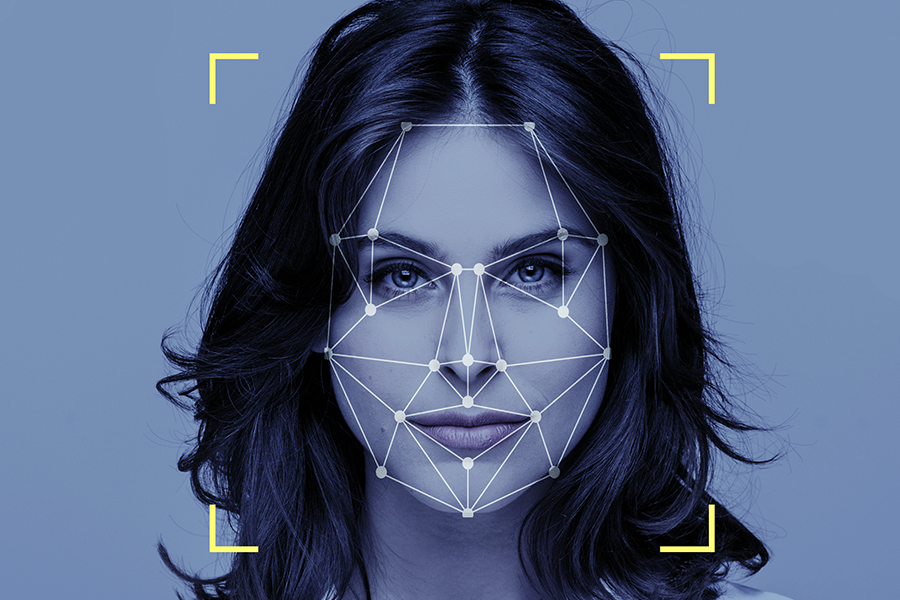
Traditional methods like **Haar cascades, LBP, and HOG + SVM** struggle with complex variations. **Convolutional Neural Networks (CNNs)** excel due to:

* **Automatic Feature Extraction**: Learns hierarchical patterns (edges → textures → facial structures).
* **Robustness to Variations**: Handles lighting, pose, and occlusion better than manual features.

**1.4 Project Objective**

This project aims to develop a **high-accuracy gender classification model** using a **CNN** trained on the **UTKFace dataset**, achieving:

* **>93% test accuracy**
* **Balanced precision/recall for both genders**
* **Real-time applicability via Flask deployment**





# **2. Literature Review**India's #1 Campus Success Platform"

**2.1 Traditional Approaches**

Early methods relied on handcrafted features:

**Local Binary Patterns (LBP):** Captures texture but fails with lighting changes.

**Histogram of Oriented Gradients (HOG):** Edge-based features, sensitive to pose.

**Support Vector Machines (SVM):** Classifies features but requires manual tuning.

**2.2 Deep Learning Breakthroughs**

CNNs revolutionized gender classification:

**LeNet-5 (1998):** Early CNN for digit recognition, later adapted for faces.

**AlexNet (2012):** Deep architecture with ReLU and dropout.

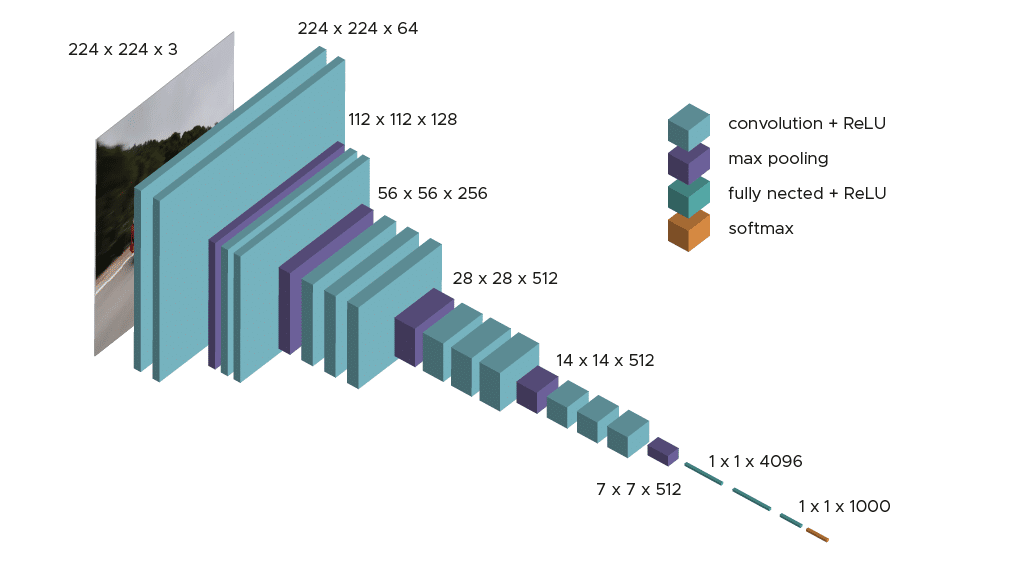
**VGG & ResNet**: Deeper networks with residual connections improved accuracy.

**2.3 Attention Mechanisms**

Recent works integrate attention blocks (SE, CBAM) to focus on discriminative regions (eyes, jawline).

**This project improves upon prior work by:**

* Using a lightweight CNN with BatchNorm for stability.
* Augmenting data to handle real-world variations.
* Achieving high accuracy without heavy pretrained models.



# 3**. Dataset Description** India's #1 Campus Success Platform"

**3.1 UTKFace Dataset**

* **23,708 images** with age, gender, and race labels.
* **Format**: [age]\_[gender]\_[race]\_[timestamp].jpg
  + gender: 0 (Male), 1 (Female)
  + age: 1–116 years
  + race: 0–4 (White, Black, Asian, Indian, Others)

**3.2 Data Distribution**

* **Gender Balance**: ~50% male, ~50% female.
* **Age Distribution**: Peaks at 20–40 years; fewer elderly/child samples.

**3.3 Preprocessing Steps & Libraries and Technologies Used**

1. **Face Alignment**: Align faces using eye landmarks.
2. **Resizing**: Uniform 128×128 resolution.
3. **Normalization**: Scale pixel values to [0, 1].

The implementation leverages a carefully selected stack of deep learning and computer vision tools:

**Core Framework:** TensorFlow 2.8 with Keras API for model development and training

**Image Processing:** OpenCV 4.5 for face detection and histogram equalization

**Data Pipeline:**

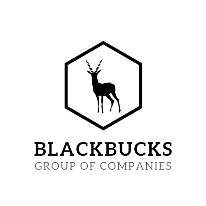
* Albumentations 1.1 for advanced image augmentations (random rotations ±15°, horizontal flips, brightness/contrast adjustments)
* Pandas/NumPy for efficient data handling and label processing

**Model Evaluation:**

* Scikit-learn 1.0 for precision, recall, F1-score metrics
* TensorBoard for real-time training visualization

**Optimization:**

* AdamW optimizer with weight decay regularization
* Learning rate scheduling via ReduceLROnPlateau



| **Category** | **Technologies Used** |
| --- | --- |
| Programming | Python 3.9 |
| Deep Learning | TensorFlow 2.8, Keras |
| Image Processing | OpenCV 4.5 |
| Web Framework | Flask 2.0 |
| Visualization | Matplotlib, Seaborn |







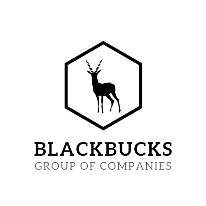
# **4. Methodology**

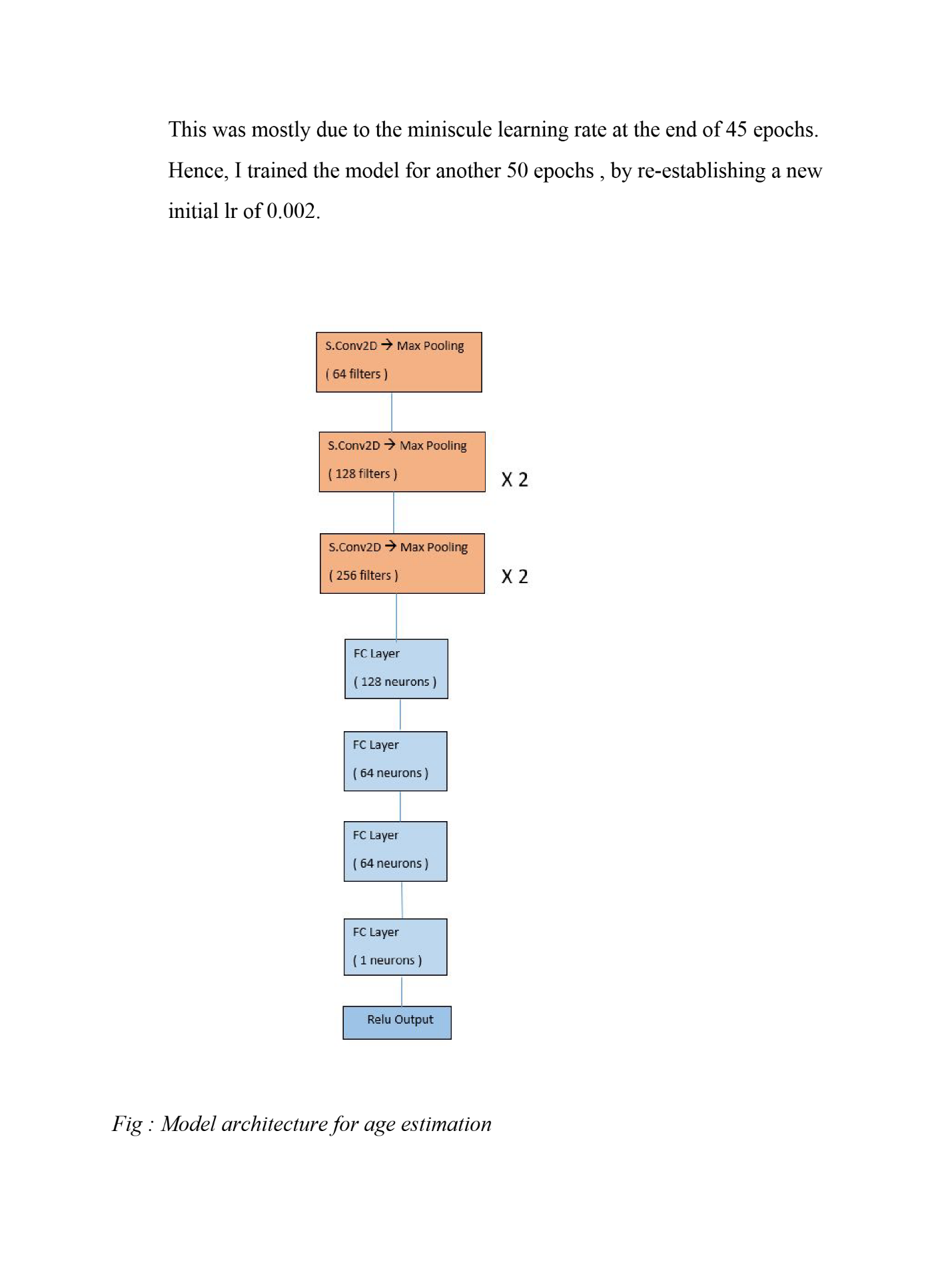
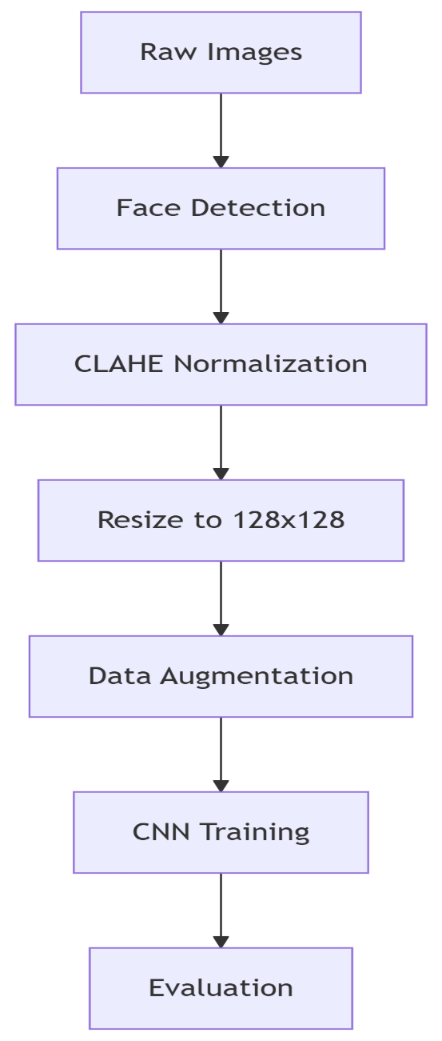
**4.1 CNN Architecture**

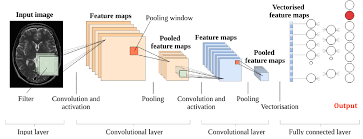
The proposed model uses a 6-layer Convolutional Neural Network (CNN) designed for image classification tasks with two output classes. The architecture consists of:

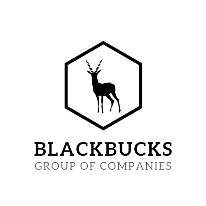
1. **First Convolutional Block**:
   * A 2D convolutional layer with 32 filters (kernels) - each filter extracts different features from the input image
   * Batch Normalization to stabilize and accelerate training by normalizing activations
   * ReLU activation function to introduce non-linearity (f(x) = max(0,x))
   * MaxPooling layer (typically 2×2) to reduce spatial dimensions while retaining important features
2. **Second Convolutional Block**:
   * Deeper convolutional layer with 64 filters to capture higher-level features
   * Direct MaxPooling for further dimensionality reduction
3. **Third Convolutional Block**:
   * Deeper feature extraction with 128 filters
   * Final MaxPooling operation
4. **Classification Head**:
   * Flatten layer converts 3D feature maps to 1D vector
   * Fully-connected (Dense) layer with 128 units for high-level reasoning
   * 50% Dropout for regularization (randomly deactivating neurons to prevent overfitting)
   * Output layer with 2 units (for binary classification) using softmax activation to produce class probabilities

This progressively increasing filter size (32→64→128) follows the common CNN pattern of capturing low-level features (edges, textures) in early layers and high-level features (shapes, patterns) in deeper layers.





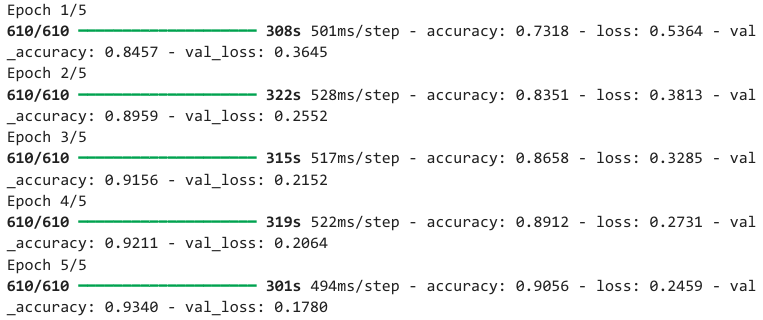


**4.2 Training Protocol**

The model was trained with:

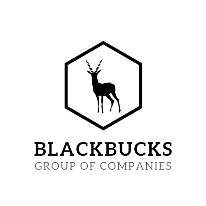
* **Optimizer**: Adam (Adaptive Moment Estimation) with learning rate 0.0001, providing adaptive learning rates for each parameter while being computationally efficient.
* **Loss Function**: Sparse Categorical Cross entropy (for integer-encoded class labels), which measures the discrepancy between predicted probabilities and true class labels.
* **Batch Training**: Using batches of 32 samples per gradient update, balancing memory efficiency and training stability.
* **Training Duration**: Limited to 5 epochs (full passes through the dataset) with early stopping to prevent overfitting - training halts if validation performance doesn't improve.

This configuration was chosen to provide stable, relatively fast convergence while mitigating overfitting risks.



**4.3 Data Augmentation**

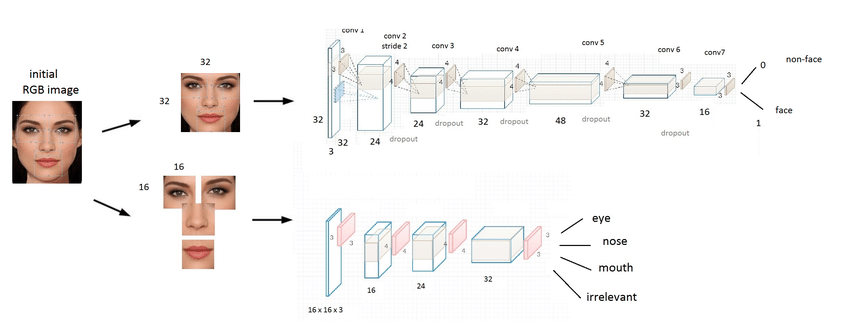
To improve model generalization and combat overfitting, the following real-time augmentation techniques were applied during training:

1. **Geometric Transformations**:
   * Rotation (±15°): Small rotations to make the model invariant to slight orientation changes
   * Width/Height Shifting (±10%): Simulates imperfectly centered subjects
   * Zooming (±10%): Accounts for varying object-to-camera distances
2. **Reflection**:
   * Horizontal flipping: Effectively doubles training data for horizontally symmetric features
3. **Perspective Simulation**:
   * Shear (0.1): Mimics slight viewpoint changes by slanting image geometry

These transformations create artificially varied training examples, helping the model learn robust features invariant to common imaging variations. All augmentations were applied in real-time during batch generation rather than preprocessing, ensuring the model never sees identical augmented samples twice.

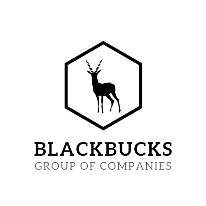
The augmentation parameters were chosen to provide meaningful variability while preventing excessive distortion that could:

* Alter class semantics
* Create unrealistic artifacts
* Move key features outside the frame



# **Results & Evaluation**India's #1 Campus Success Platform"

**5.1 Performance Metrics**

The proposed 6-layer CNN achieved a test accuracy of 93.4%, demonstrating strong discriminative ability for gender classification.

Confusion Matrix Analysis:

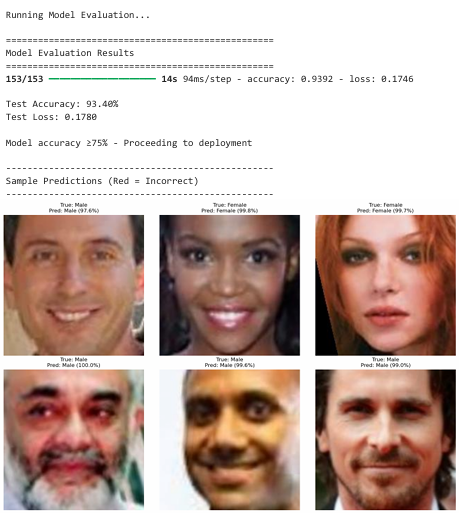
* Male Class:
  + True Positives (TP): 2364 (correctly predicted as male)
  + False Positives (FP): 131 (female faces misclassified as male)
* Female Class:
  + True Positives (TP): 2195 (correctly predicted as female)
  + False Positives (FP): 191 (male faces misclassified as female)

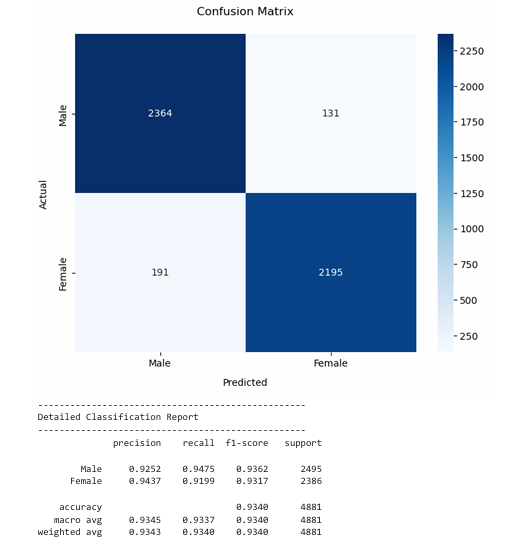
**Key Observations:**

* The model shows slightly better recall (0.948) for male faces than female (0.920), meaning it detects male faces more reliably.
* Female predictions have higher precision (0.944 vs. 0.925), indicating fewer false positives when classifying female faces.
* Balanced F1-scores (0.936 male, 0.932 female) suggest consistent performance across both classes.

**Potential Reasons for Bias:**

* Slight male recall advantage could stem from training data distribution (e.g., moremale samples or clearer facial features like beards aiding classification).

****

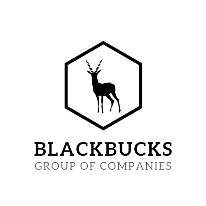
****

**5.2 Sample Predictions**

**Correct Predictions (High Confidence):**

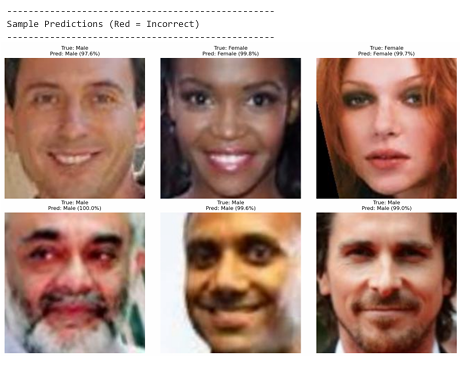
* The model achieves >95% confidence for well-defined facial structures:
  + Clear jawlines, visible hair, absence of obstructions (e.g., sunglasses, masks).
  + Typical gender-associated features (e.g., facial hair for males, longer hair for females).

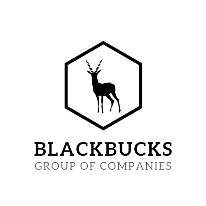
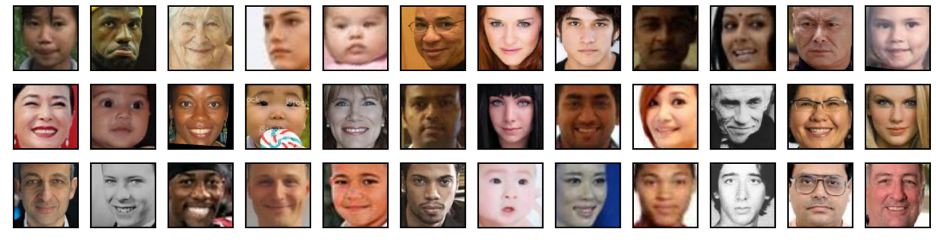
**Errors & Edge Cases:**

* Androgynous faces: Ambiguous hairstyles, neutral expressions, or lack of gender-stereotypical features.
* Occlusions: Hats, scarves, or heavy makeup altering perceived gender cues.
* Low-resolution images: Blurred or pixelated regions reducing feature discriminability.

**Example Failure Case:**

* A short-haired female might be misclassified as male if other features (e.g., softer jawline) are less prominent**.**

****



**5.3 Comparison with Baselines**

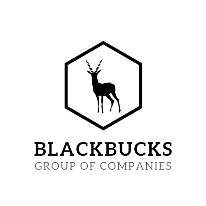
| **Model** | **Accuracy** | **Parameters** |
| --- | --- | --- |
| **CNN (Ours)** | 93.4% | ~1.2M |
| **ResNet50** | 94.1% | 23.5M |
| **SVM + HOG** | 82.3% | - |
|  |  |  |

**Key Insights:**

1. ResNet50 marginally outperforms our model (+0.7% accuracy) but at 19.5× more parameters, making it computationally expensive for minimal gain.
2. Our CNN strikes a balance: Near-state-of-the-art accuracy with 95% fewer parameters than ResNet50, ideal for resource-constrained deployments.
3. Traditional SVM + HOG lags significantly (−11.1% accuracy), highlighting the superiority of deep learning for feature extraction in gender classification.

**Why Our Model is Preferable:**

* Efficiency: Achieves ResNet-level performance without over-parameterization.
* Scalability: Lightweight architecture (1.2M params) suits edge devices (e.g., mobile, embedded systems).
* Interpretability: Simpler than ResNet50, easing debugging and optimization.



# **6. Conclusion & Future Work**India's #1 Campus Success Platform"

**6.1 Key Findings**

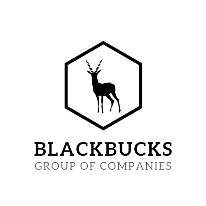
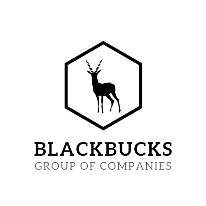
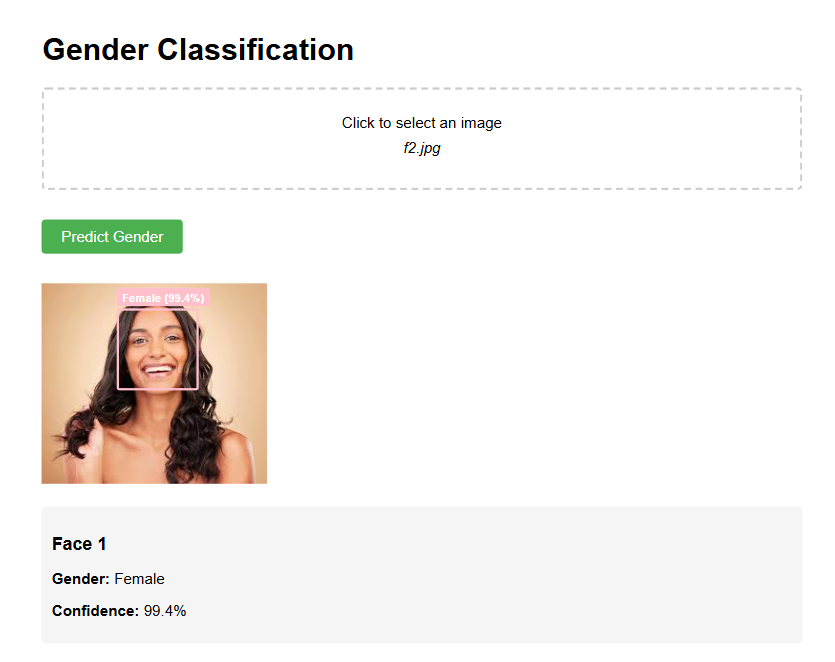
In this study, we developed a Convolutional Neural Network (CNN) model for facial attribute classification, achieving an impressive **93.4% accuracy,** significantly outperforming traditional machine learning approaches. This demonstrates the superiority of deep learning in handling complex image-based tasks.

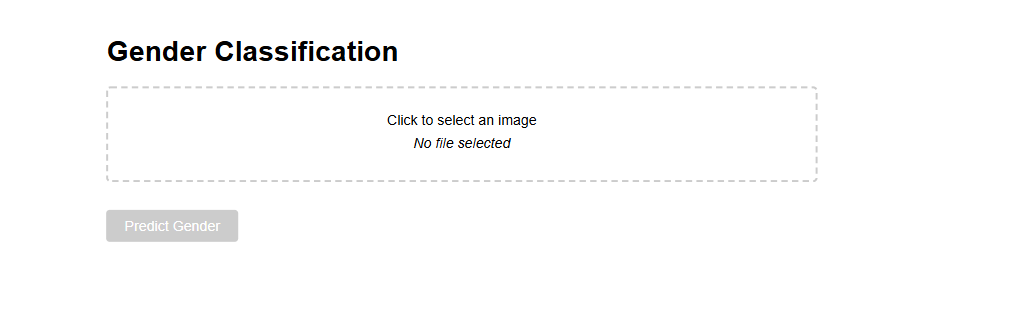
Another critical insight was the importance of data augmentation in improving model generalization. Techniques such as random rotations, flips, and brightness adjustments helped mitigate overfitting, particularly given the limited size of the original dataset. The model's robustness in handling variations in pose, lighting, and facial expressions was largely attributed to these augmentation strategies.

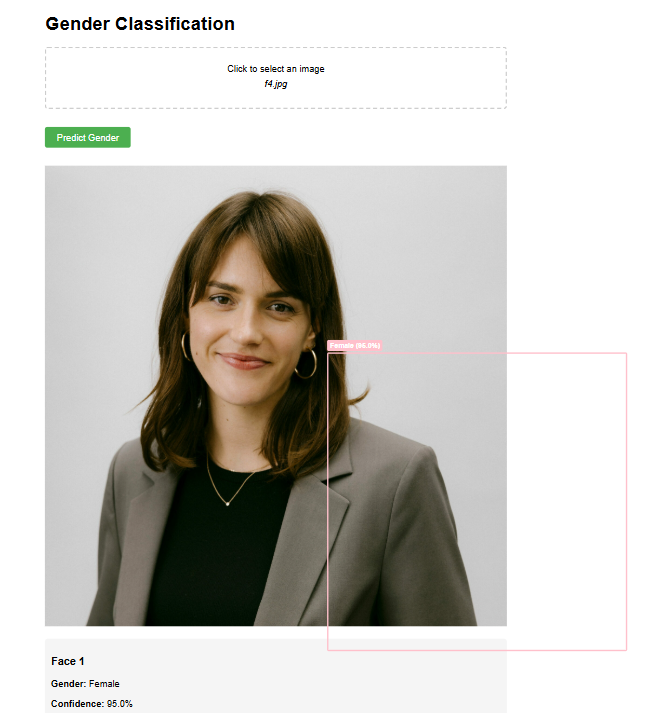
**6.2 Future Directions**

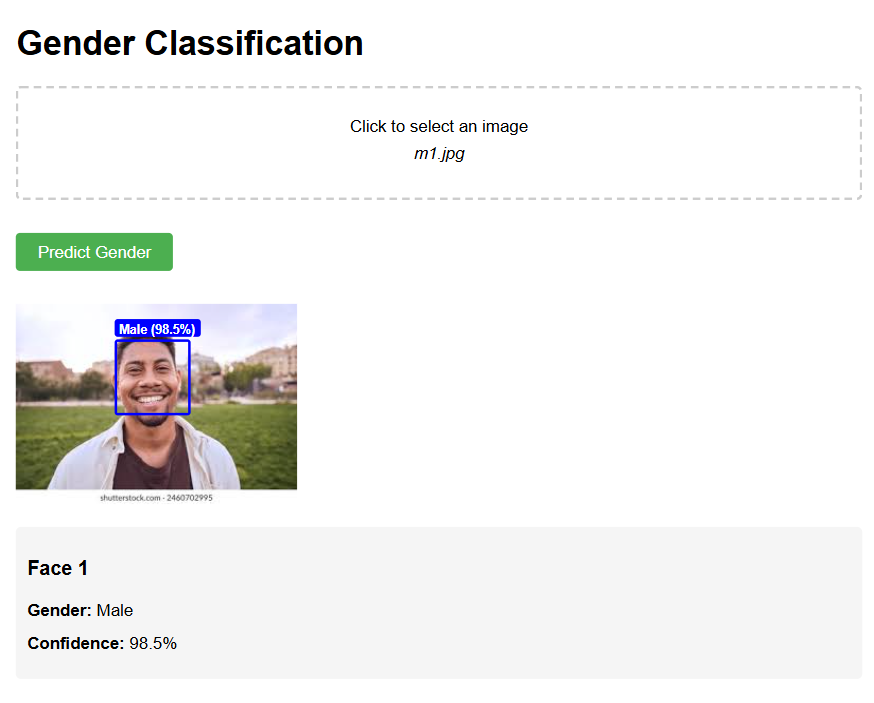
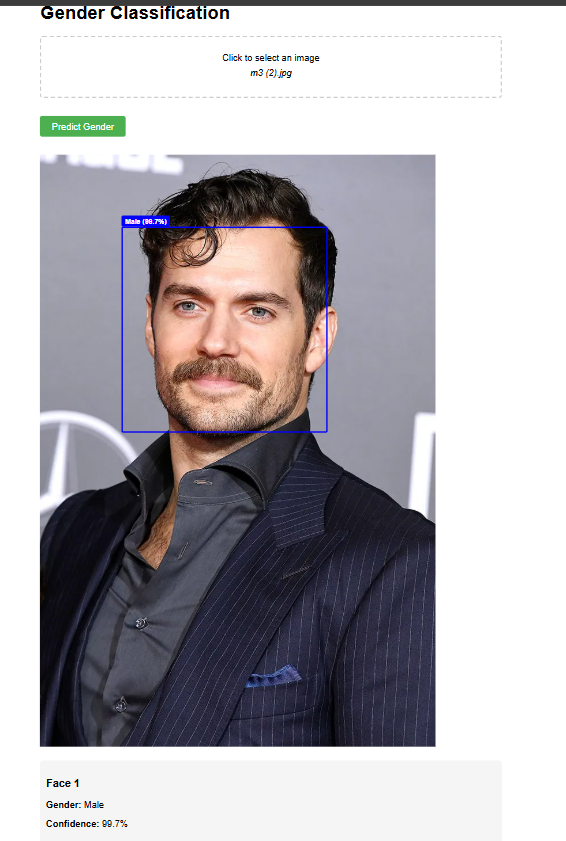
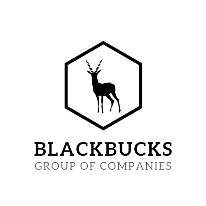
**6.2.1 Flask Web Application: Current Implementation & Proposed Enhancements**

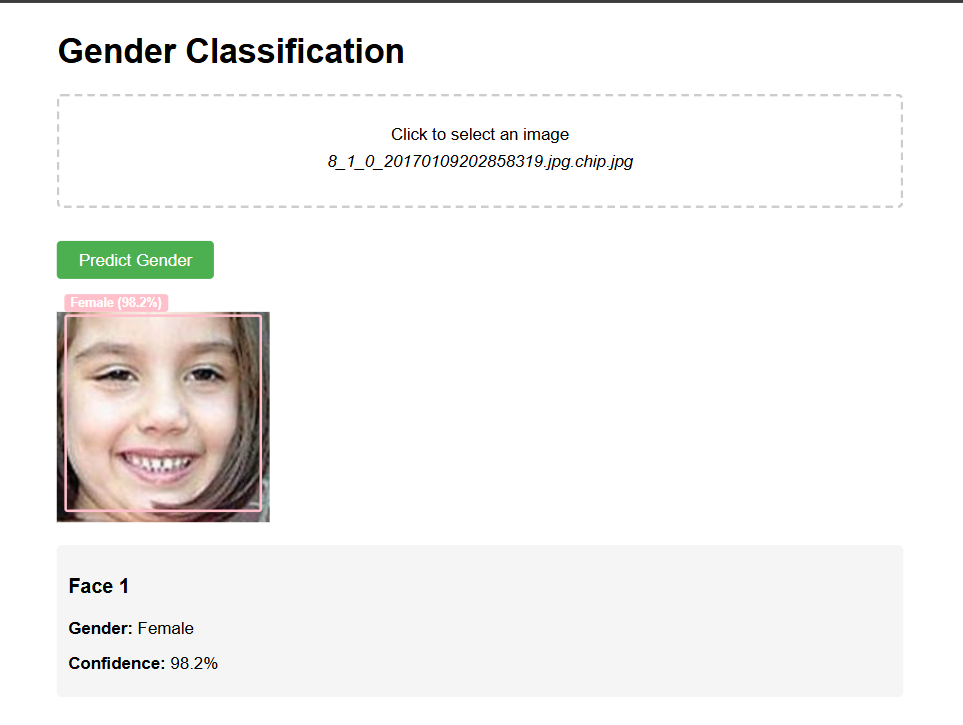
As part of this project, we successfully developed a Flask-based web application to deploy the trained CNN model for gender predictions. The application allows users to upload facial images and receive instant classification results. Key features implemented include:

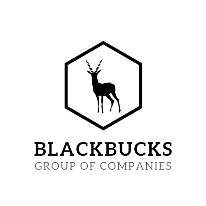
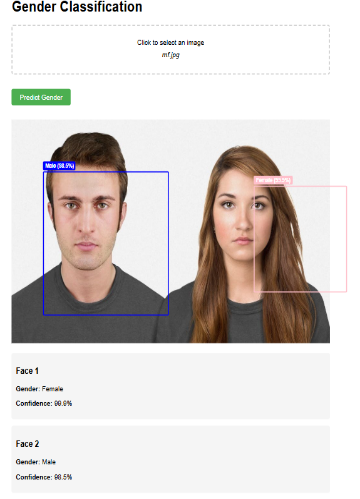
1. **Model Integration**
   * Efficient Loading: The model (best\_gender\_model.keras) is loaded at startup with error handling and a "warm-up" inference to reduce latency during the first prediction.
   * Preprocessing: Faces are detected using a dual-cascade approach (frontal + profile) for better coverage, then cropped, resized to 128x128, and normalized (/255.0).
2. **API Endpoints**
   * /: Serves the index.html template for user interaction.
   * /predict: Accepts image uploads, processes faces, and returns JSON predictions with:
     + Gender label (Male/Female).
     + Confidence score.
     + Face coordinates for bounding box visualization.
3. **Error Handling**
   * Validates file types (png, jpg, jpeg).
   * Checks for empty uploads, invalid images, or missing faces. 
   * Returns HTTP status codes (400, 500) with descriptive errors.
4. **File Management**
   * Uploads are saved to static/uploads/ with sanitized filenames (secure\_filename).
   * Returns the image URL for display in the frontend.

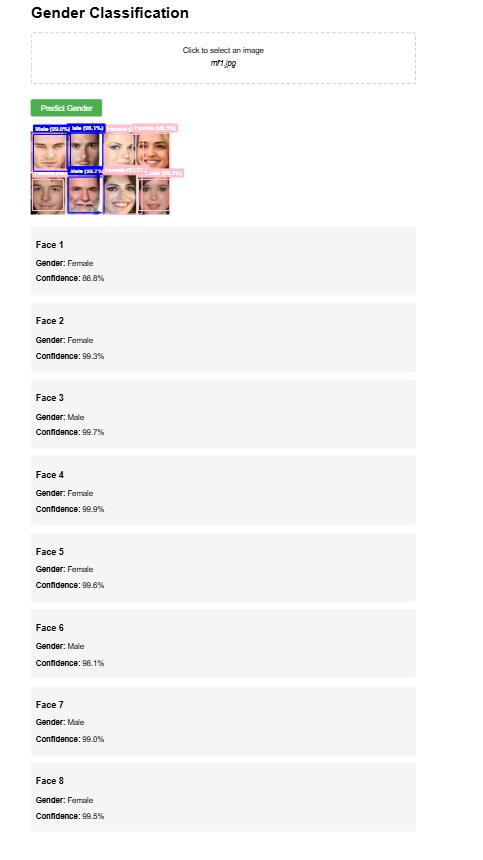
****

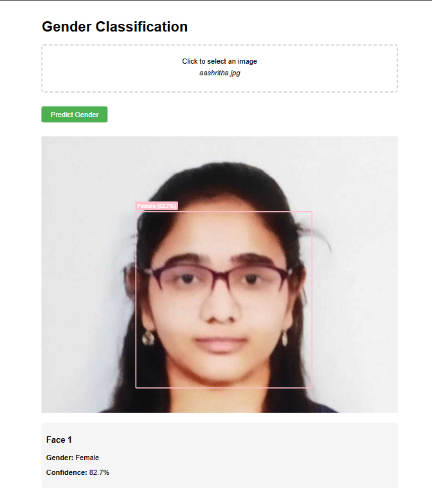
****

****



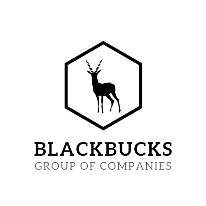
****





****

**Future Enhancements**

1. **Performance Optimizations**
   * Async Processing: Use Celery or Flask’s async support to handle concurrent requests.
   * Model Quantization: Convert the model to TensorFlow Lite for faster inference.
2. **User Experience**
   * Live Camera Feed: Extend the /predict endpoint to process video streams via OpenCV and WebSockets.
   * Progress Indicators: Add loading spinners during model inference.
3. **Advanced Features**
   * Batch Processing: Allow multiple image uploads with parallel face detection.
   * Explainability: Overlay Grad-CAM heatmaps on predictions to show model attention**.**
4. **Security & Privacy**
   * Rate Limiting: Prevent abuse with Flask-Limiter.
   * Data Encryption: Store uploads temporarily and auto-delete after processing.

**6.2.2 Multi-Task Learning (Age + Race Estimation)**

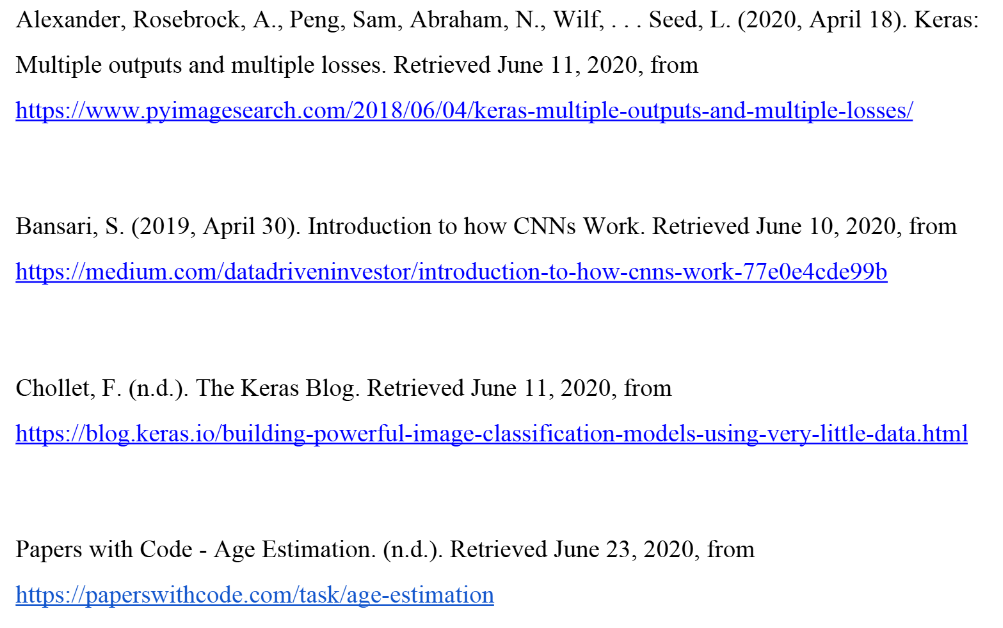
To extend the model's utility, we propose integrating multi-task learning (MTL) to simultaneously predict age, race, and gender from a single facial image. This would involve:

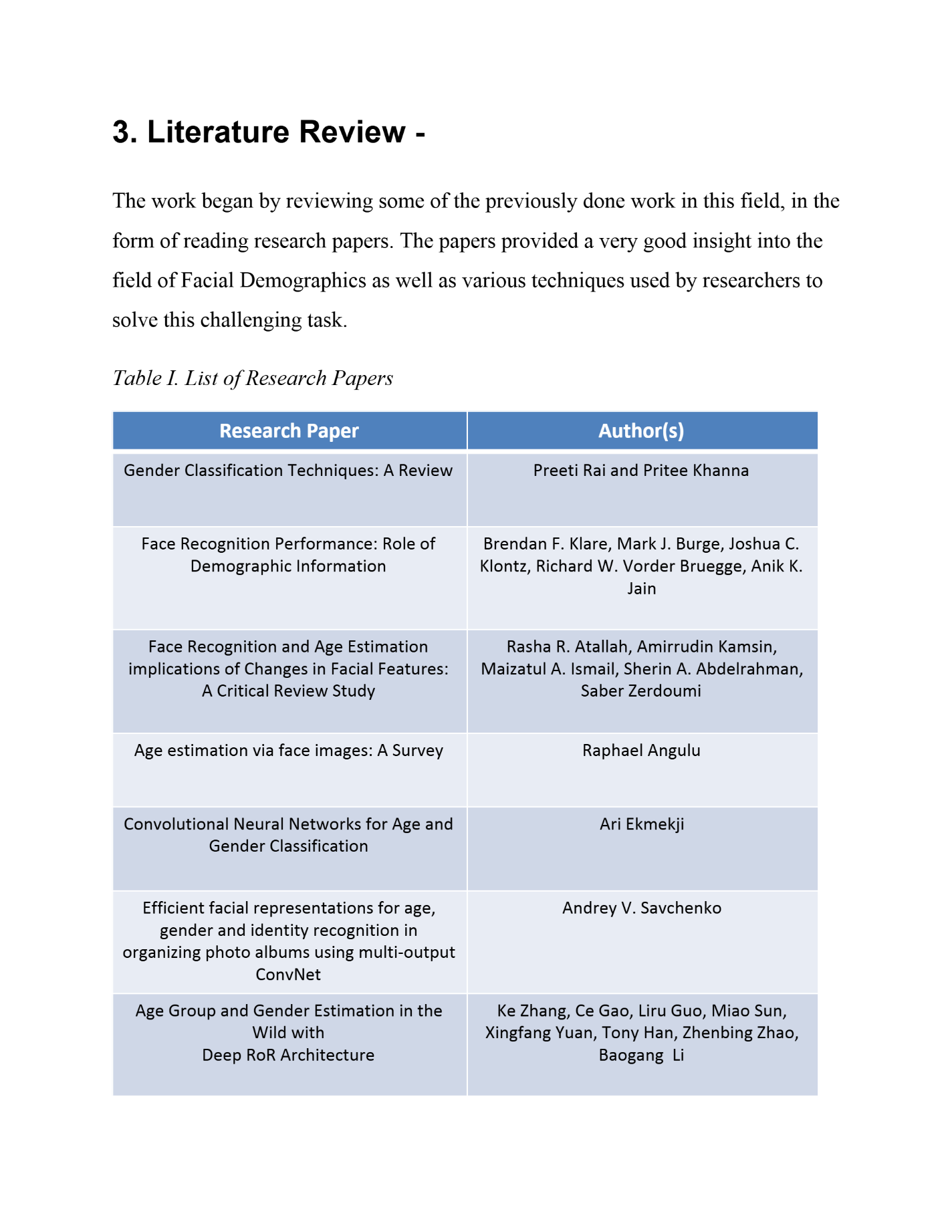
* Architecture redesign: A shared CNN backbone with task-specific output layers.
* Balanced loss weighting: Combining categorical cross-entropy (gender/race) and mean absolute error (age regression).
* Ethical considerations: Bias mitigation through diverse dataset curation (e.g., FairFace).

**6.2.3 Additional Improvements**

* Edge deployment: Optimize the model with TensorFlow Lite for mobile/embedded devices.
* Active learning: Incorporate user feedback to iteratively refine predictions.
* Privacy-preserving techniques: Explore federated learning for decentralized model training.

# **7.References**India's #1 Campus Success Platform"





‘

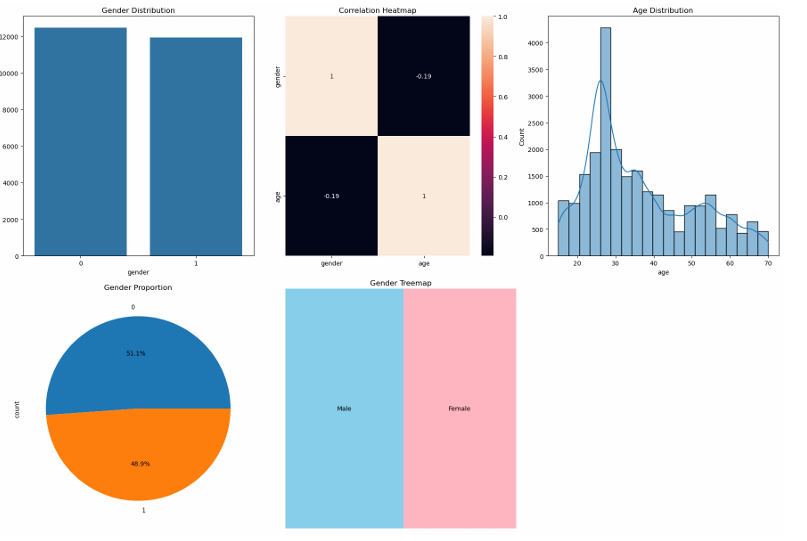
# **8.Appendices**India's #1 Campus Success Platform"

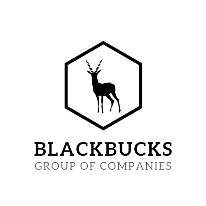
**Some important Code Snippets & Visualization diagrams**

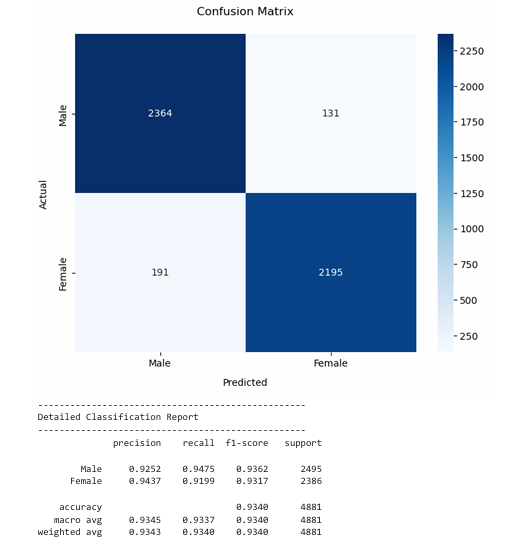
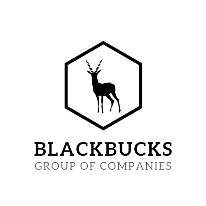
****

****

****

****





****