International Journal of Innovative Research in Engineering
Volume 6, Issue 3 (May-June 2025), PP: 45-47.
www.theijire.com ***ISSN No: 2582-8746***

Smart Face Analysis for Age and Gender Recognition

Anshika Chaurasia[1], Amna Tooba[2], Anamika Pandey[3]
[1,2,3] *Department of Computer Science, Institute of Technology and Management, GIDA, Gorakhpur, Uttar Pradesh, India.*

How to cite this paper:
Anshika Chaurasia[1], Amna Tooba[2], Anamika Pandey[3], "Smart Face Analysis for Age and Gender Recognition", IJIRE-V6I3-45-47.

Abstract: *Understanding a person's age and gender just by looking at their face has become an important area of research, with many real-world uses. In this project, we used the Caffe deep learning framework to improve how accurately we can analyse age and gender from face images. By training deep convolutional neural networks (CNNs), we were able to get much better results than existing methods. We also designed a simpler version of the network that still works well, even when there's little or no training data available. Overall, our method offers a more effective and efficient way to perform age and gender analysis and sets a strong standard for future work in this field.*

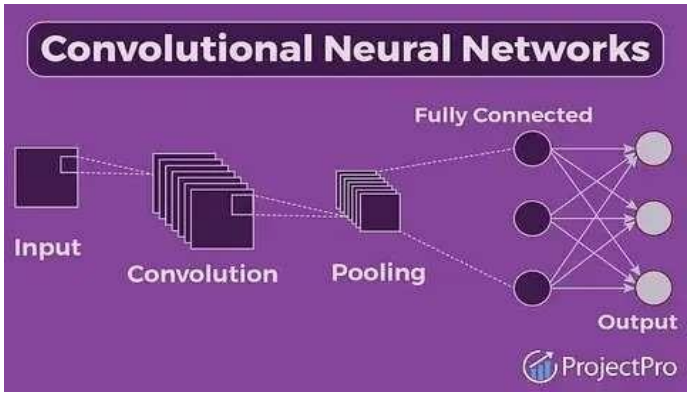
Key Words: *Caffe model, Deep-Convolutional neural networks, Deep Learning framework, Tensor Flow Abbreviations and Acronyms: CNN- Convolutional Neural Networks DNN-Deep Neural Network.*

I.INTRODUCTION

Knowing a person's age, gender, and whether they are smiling can be useful in many real-life situations—like security, customer service, online ads, product suggestions, and studying how people behave in crowds. But figuring out this information from a photo can be hard because people's faces look different depending on things like lighting, camera angles, and facial expressions. Older methods often looked at the size and shape of facial features to guess age or gender, but they didn't work well with everyday, casual photos. They also didn't use the large number of face images available online to get better. In this project, we used deep learning to smartly analyze a person's age, gender, and smile from just one photo of their face.In our system, gender is identified as either Male or Female. For age, we don't try to predict the exact number. Instead, we use one of eight age groups: (0–5), (6–10), (11–15), (16–20), (21–35), (36–40), (41 45), (46–50), (51-55), (56-60), (61-75),(76 80),(81-85), (86-90), (91-95),(96-100). These groups are used in the final layer of our model to make a prediction. Figuring out someone's exact age from just one photo is difficult because of things like makeup, lighting, face coverings, or different facial expressions. That's why we treat age detection as a classification problem, not a regression one.We used the Caffe deep learning framework to train our models, and in our internal tests, Caffe performed 1 to 5 times faster than TensorFlow. After training, we used the .caffemodel file to make predictions on new, unseen images. A Python script using OpenCV was created to run the system. This script automatically finds a face in an image and predicts the person's age, gender, and whether they are smiling. Age, gender, and smile detection are key facial features that can help in many smart applications. These include access control, human-computer interaction, digital marketing, video surveillance, counting children and adults in hospitals, and automatic ticket machines that issue tickets based on age.

II.METHODOLOGY

We used techniques like face detection and face alignment with OpenCV to process images. Then, we trained our models using a CNN (Convolutional Neural Network). After that, we accurately predicted a person's age group, gender, and whether they are smiling from a single photo.



The Architecture of CNN Model

Our CNN model is shown in this diagram. It has three convolutional layers, each followed by a ReLU activation to help the model learn better. After that, there's a pooling layer to reduce the size of the data. The first two layers also use a method called local response normalization to keep the data balanced during training.

The first convolutional layer in our CNN has 96 filters, with a resolution of 77%. The second layer has 256 filters and a resolution of 55%. The third and final convolutional layer uses 384 filters with a resolution of 33%. After that, we add two fully connected layers, each with 512 neurons. In this approach, we use the Video Capture() function to capture faces from the webcam. Once we get the video frames, we detect the faces in them. We then process and normalize the faces using a function called detect Face(), which applies another function called blob From Image(). This function helps by subtracting the mean, scaling the image, and changing the color format from BGR to RGB.

After the images are processed, we use the read Net() function to load the deep learning model, which can be in different supported formats. We then send the processed faces to the pre-trained Caffe models to get the predictions. The results are displayed on the screen with green bounding boxes around the predicted faces.

For this project, we used the Adience dataset, available on Kaggle. This dataset contains around 26,580 photos of 2,284 people across eight different age groups. It includes various challenges like noise, lighting, angles, and facial expressions, making it a great real-world dataset for this task. The pre-trained models are tested on this dataset to make prediction.

Fasting capillary blood glucose [CBG] was determined by using One Touch Ultra glucose meter (Johnson & Johnson, Milpitas, California) after eight hours of overnight fasting. A fasting venous sample was collected and lipids were measured. (10) All biochemical assays was carried out by the same team of laboratory technicians using the same method, throughout the study period. The samples were assayed for total cholesterol, triglycerides and HDL cholesterol.

Serum cholesterol (cholesterol esterase oxidase-peroxidase-amidopyrine method), serum triglycerides (glycerol phosphate oxidase-peroxidase-amidopyrine method), and high-density lipoprotein cholesterol (direct method poly-ethylene-glycol-pretreated enzymes) was measured using the Beckman Coulter AU 2700/480 Autoanalyser (Beckm an AU [Olympus], Ireland). The intra– and inter-assay coefficients of variants (CV) for the biochemical assays ranged from 3.1% to 7.6%. (10)

In every subject, a semi-quantitative food frequency questionnaire was administered to collect detailed information on dietary intake over the past year. Dietary fat and oil intake was assessed as the amount of fat/oil used during cooking and/or added at the table.

III.IMPLEMENTATION

Module -1: Data Preparation:

At this stage, we cleaned the photos and saved them in a format that works with Caffe. We also created a Python script to handle the image processing and saving them properly.

Module -2: Model definition:

In this step, we choose a CNN architecture and define its settings in a.prototxt configuration file.

Module-3: Defining the Solver:

The solver is responsible for improving the model during training. We set the solver's settings in another .prototxt file.

Module-4: Training the Model:

To train the model, we use a single command in the Caffe terminal. After training, we get the model in a file with the .caffemodel extension. For face recognition, we also use a .pb file, which is a protocol buffer that contains the graph and weights used for training the model. This is the file we use to run the trained model. The .pb file stores the model in binary format (using 1s and 0s), while the .pbtxt file stores it in a readable text format. The model's structure for age and gender is described in .prototxt files, and the .caffemodel file stores the trained values for the layers. We also include TensorFlow files along with the .caffemodel when loading the networks. To read the image path from the command line, we used Python's argparse library. This lets us pass the image file as input and then predict age and gender. A protocol buffer is created to handle face, age, and gender data. We also define the average values (mean) for images and set the list of age groups and gender labels. The readNet() function is used to load the models. One input stores the model's weights, and the other stores the configuration settings. We use the webcam to capture a live video stream and set padding to 20. We keep reading frames from the video until a key is pressed. If it's not a video file, the program waits using cv2.waitKey(), then breaks the loop. Next, using the faceNet model and the current video frame, we run the detectFace() function. The result is stored in frameOpencvDNN and faceBoxes. If there are zero faceBoxes, it means no faces were found. The faceNet model we used is a Deep Neural Network (DNN) Face Detector, and it only takes about 2.7MB of storage.

To detect faces using a deep learning model (DNN), we follow these steps:

1. We first make a copy of the video frame to get its height and width.
2. From this copy, we create a special input format called a blob that the model can understand.
3. We set this blob as the input and run it through the network to detect faces.
4. We start with an empty list called faceBoxes to store the face positions. For each possible face, we check how confident the model is. If the confidence is higher than 0.7, we get the position (x1, y1, x2, y2) of the face and add it to the list.
5. For every face found, we draw a rectangle on the image and return both the edited image and the list of face positions.

If any faces are detected, we then process

- We create another blob for the face to prepare it for age and gender prediction.
- We scale and resize the image, apply average (mean) values, and send it into the network.
- The network gives us confidence scores for two genders. The one with the higher score is chosen as the predicted gender.
- The same process is repeated to predict the age group. Finally, the age and gender labels are added to the image, and we use imshow to display the result.

IV. RESULTS

As mentioned before, when the system is running with live video, it automatically detects the user's face and draws a green rectangle around it. Inside this box, the system displays two main of information: gender and age group. approximate age is enough. It's especially helpful approximate age is enough. It's especially helpful inplaces like smart surveillance, automated ticket counters, targeted advertising, or even crowd analysis. The goal is not to give exact personal data,but to provide a quick and smart estimate of someone's age group and gender using just their face from a camera feed As mentioned, the Caffe framework has a performance of 1to5 times more than TensorFlow in the internal benchmarking and better performance compared to other deep learning frameworks.

V. CONCLUSION

The results of this project, which focused on identifying gender and estimating age from facial images, can be effectively applied to real time applications. While earlier research also looked into gender and age classification, most of those studies used controlled lab photos. These don't fully reflect the wide variety of faces, expressions, lighting, and conditions we see in everyday pictures—especially on social media or the internet. In contrast, real-world images found online are not only more challenging but also much larger in number. In this project, we explored how well a Deep Convolutional Neural Network (Deep-CNN) can handle gender and age prediction using images from the internet, which are more natural and diverse. Because labeled data (i.e., data that already has age and gender tags) is limited, we used a simple and efficient deep learning model. Our model avoids overfitting by keeping the architecture shallow—meaning it uses fewer layers and parameters than more complex models. This helps the system perform better on new, unseen images. Overall, our approach shows that with the right deep learning techniques, it's possible to build an effective and lightweight system for gender classification and age group estimation that works well even in challenging, real-world scenarios.

Future Enhancements:

We will look into a more complex CNN architecture and a more reliable image processing approach for estimating exact ages for future work. We can use this project for electronic customers, crowd behaviour analysis.

References

1. *G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2016.*
2. *S. Levi and T. Hassner, "Age and Gender Classification Using Convolutional Neural Networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Boston, MA, USA, Jun. 2015, pp. 34–42.*
3. *OpenCV.org, "OpenCV: Open Source Computer Vision Library," [Online]. Available: https://opencv.org/. [Accessed: 10-May-2025].*
4. *BVLC, "Caffe Model Zoo," GitHub Repository, 2016. [Online]. Available: https://github.com/BVLC/caffe. [Accessed: 10-May 2025].*
5. *Python Software Foundation, "Python 3 Documentation," [Online]. Available: https://docs.python.org/3/. [Accessed: 10-May-2025].*
6. *TkDocs, "Tkinter GUI Programming," [Online]. Available: https://tkdocs.com/. [Accessed: 10-May 2025].*
7. *A. Mollahosseini, D. Chan, and M. H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," in Proc. IEEE Winter Conf.Appl. Comput. Vis. (WACV), Lake Placid, NY, USA, Mar. 2016, pp. 1–10.*