

**Fatima Jinnah Women University**

**Fatima Jinnah Women University**

**Submitted to: Sir Adeel Khalid**

**Submitted by: AimenJillani (001)**

**Kainat Bibi (017)**

**Maryam Siddiqui (027)**

**Natasha Kanwal (032)**

**Noor Fatima (035)**

**Wishma Noor (041)**

**Course: Artificial Intelligence**

**Semester: BCS (6thSemester)**

**PROJECT REPORT OF ARTIFICIAL INTELLIGENCE**

**Table Of Content**

[Age, Emotions And Gender Detection 3](#_Toc166505112)

[Introduction 3](#_Toc166505113)

[Objective 4](#_Toc166505114)

[Computer Vision 4](#_Toc166505115)

[Review Of Prior Research 4](#_Toc166505116)

[Methodology 4](#_Toc166505117)

[Capturing Video In Real-Time 5](#_Toc166505118)

[Command For Image Analysis: Python Detect.Py --Image <Image\_Name> 5](#_Toc166505119)

[Command For Real-Time Analysis: Python Detect.Py 5](#_Toc166505120)

[Age Detection 5](#_Toc166505121)

[Gender Classification 6](#_Toc166505122)

[**Emotion Detection 6**](#_Toc166505123)

[Smile Detection And Save The Picture: 7](#_Toc166505124)

[Opencv 8](#_Toc166505125)

[Argparse 9](#_Toc166505126)

[Tensorflow 9](#_Toc166505127)

[Deepface Package 9](#_Toc166505128)

[Haar Cascades For Face Detection 9](#_Toc166505129)

[The Cnn Architecture 9](#_Toc166505130)

[Basic Concept Of Cnn Architecture 9](#_Toc166505131)

[Application In Gender And Age Detection 10](#_Toc166505132)

[The Dataset 11](#_Toc166505133)

[**Datasets For Emotions 11**](#_Toc166505134)

[Models Trained 12](#_Toc166505135)

[Impact And Applications 12](#_Toc166505136)

[Prerequisites 12](#_Toc166505137)

[Projects Setups By Following Steps: 13](#_Toc166505139)

[Working: 14](#_Toc166505140)

[Testing Phases: 14](#_Toc166505141)

[Phase 01 : Python Project Examples For Gender And Age Detection 14](#_Toc166505142)

[Detecting Age And Gender From Webcame Capture Images: 16](#_Toc166505143)

[Phase 02: Detecting Age , Emotions And Gender From Webcame Capture Images: 16](#_Toc166505144)

[Phase 03: Detect Smile And Take Photo And Svae It In Folder 17](#_Toc166505146)

[Conclusion: 19](#_Toc166505147)

[References: 21](#_Toc166505148)

## **AGE, EMOTIONS AND GENDER DETECTION**

## **INTRODUCTION**

Age, emotions and gender detection, a burgeoning field in computer vision, relies on sophisticated machine learning techniques to precisely discern these attributes from images. This project endeavors to create a deep learning model, leveraging Python's CNN, to accurately predict age and gender from images. Key stages include dataset collection, preprocessing for normalization and noise removal, and CNN-based model training. CNN architecture comprises convolutional, pooling, and fully connected layers for feature detection and classification. Training involves stochastic gradient descent optimization and cross-entropy loss minimization. Evaluation on a test set assesses accuracy, with techniques like regularization, dropout, and data augmentation enhancing performance. The deployed application enables age and gender prediction via web or mobile interfaces, facilitating integration into diverse systems for security, entertainment, and social media analysis.

In this Python Project, we will use Deep Learning to accurately identify the gender and age of a person from a single image of a face. We will use the models trained by [Tal Hassner and Gil Levi](https://talhassner.github.io/home/projects/Adience/Adience-data.html). The predicted gender may be one of ‘Male’ and ‘Female’, and the predicted age may be one of the following ranges- (0 – 2), (4 – 6), (8 – 12), (15 – 20), (25 – 32), (38 – 43), (48 – 53), (60 – 100) (8 nodes in the final softmax layer). It is very difficult to accurately guess an exact age from a single image because of factors like makeup, lighting, obstructions, and facial expressions. And so, we make this a classification problem instead of making it one of regression.

## **OBJECTIVE**

The main objective of this project is to create a model that can accurately predict the age and gender also emotions of a person by analyzing their facial features in an image. This application can be utilized in various domains such as security systems, entertainment, and social media analysis.

## **COMPUTER VISION**

**Computer Vision** is the field of study that enables computers to see and identify digital images and videos as a human would. The challenges it faces largely follow from the limited understanding of biological vision. Computer Vision involves acquiring, processing, analyzing, and understanding digital images to extract high-dimensional data from the real world in order to generate symbolic or numerical information which can then be used to make decisions. The process often includes practices like object recognition, video tracking, motion estimation, and image restoration.

## **REVIEW OF PRIOR RESEARCH**

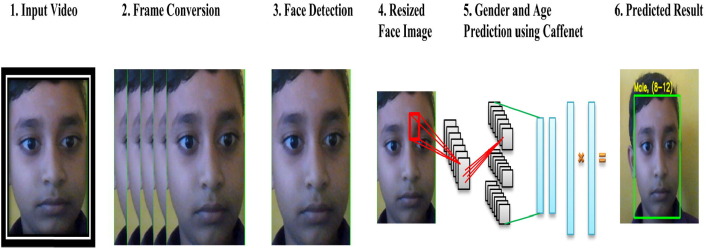
This section synthesizes research on CNN-based automatic age and gender prediction from facial images. A study explored age and gender classification at both broad and detailed levels using a neural network for initial analysis and a 3-sigma control for precise classification, focusing on different age groups with benchmark database images for high efficiency. Another research [6] assessed CNN performance in file-based and camera-based tests, finding RGB images superior in file-based scenarios, whereas grayscale images excelled in camera-based tests.

Research introduced a CNN method for age and gender prediction using the IMDB-WIKI dataset, employing regression for age estimation and classification for gender. The work in [8] utilized CNNs for face detection and gender classification, leveraging the IMDB dataset and implementing the network with TensorFlow and Keras. A study proposed the FEBFRGAC algorithm for face recognition and age/gender classification, achieving significant results with a small training set through posteriori class probability and ANN models.

The AgeNet approach developed an age-estimating deep CNN, combining classification and regression to extract age descriptors and apply a divide-and-rule strategy for accurate age prediction. Research compared video-based age and gender recognition algorithms across various datasets, leading to the development of an Android application for retail purposes, highlighting practical application potential. Finally, a study focused on an Android app leveraging the UTK Face dataset for automatic extraction of personal attributes like gender and birth year, demonstrating the applicability of these technologies in real-world scenarios. The paper is structured into sections on methodology, experimental results, discussion, and conclusions.

## **METHODOLOGY**

The work flow of the proposed age and genderand emotion predictor is



## **CAPTURING VIDEO IN REAL-TIME**

OpenCV simplifies the process of capturing video in real-time through webcams or other video input devices. It converts the video stream into a series of image frames that can be processed and analyzed. This functionality is crucial for applications that require real-time analysis, such as surveillance, interactive installations, or live event monitoring.

## **COMMAND FOR IMAGE ANALYSIS: python detect.py --image <image\_name>**

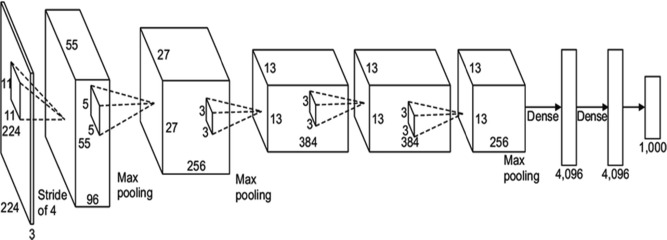
This command allows the user to analyze a specific image for gender and age detection. The image must be located in the same directory as the script for successful processing.

## **COMMAND FOR REAL-TIME ANALYSIS: PYTHON DETECT.PY**

This initiates the script to use the webcam for real-time face detection, analyzing gender and age without the need for pre-stored images. Execution can be halted with Ctrl + C.

## **AGE DETECTION**

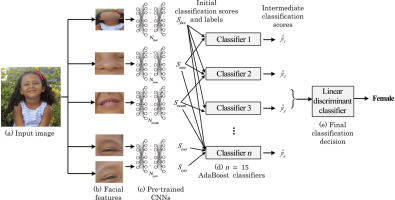
* **Objective:** Estimating a person's age from their facial image, critical for applications in advertising, healthcare, and security.
* **Traditional Methods:** Relied on hand-crafted features and algorithms like SVMs or decision trees, faced accuracy issues, and required extensive pre-processing.
* **Deep Learning Advantage:** CNNs can autonomously learn from images, offering higher accuracy and less sensitivity to image variations. They simplify pre-processing needs.
* **Training and Pre-processing:** Requires a large dataset with age labels. Involves image standardization and augmentation to enhance diversity. CNNs learn to categorize age through probability distributions.

c

**Even humans cannot accurately predict the age based on looking at a person**. If a person’s age is in 20′s or 30′s , it is difficult to predict the exact age. To avoid this condition, frame this problem as a classification problem. For example, age in the range of (0–2) is a single class. (8–12) is another class and so on. Likewise, we split the age into eight age groups. The age prediction network has eight nodes in the final softmax layer indicating the mentioned age range. Furthermore, the CNN’s output layer (probability layer) in this CNN consists of 8 values for 8 age classes (“0–2”, “4–6”, “8–13”, “15–20”, “25–32”, “38–43”, “48–53” and “60–80”).

## **GENDER CLASSIFICATION**

* **Objective:** Predicting a person's gender from their facial image, with significant applications across advertising, security, and healthcare.
* **Traditional Methods:** Utilized hand-crafted features and algorithms like SVMs, struggling with accuracy and robustness against variations in expression, lighting, and pose.
* **Deep Learning Improvement:** CNNs excel by learning features directly from images, improving accuracy and robustness, and reducing the need for detailed pre-processing.
* **Training Process**: Training a CNN for gender classification requires less pre-processing, leveraging the natural ability of CNNs to identify gender-distinguishing features from facial images.

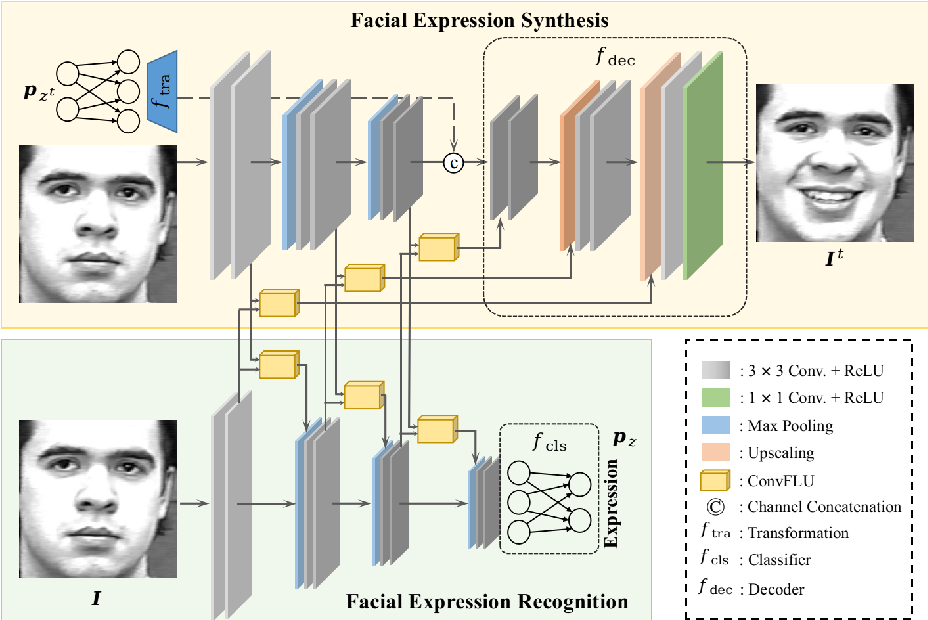


# **EMOTION DETECTION**

DeepFace is an advanced facial recognition system developed by a team at Facebook. Leveraging deep learning techniques, DeepFace is capable of accurately identifying human faces in digital images with remarkable precision. The system utilizes a sophisticated nine-layer neural network with over 120 million connection weights, trained on an extensive dataset consisting of four million images contributed by Facebook users.

In the realm of emotion detection, DeepFace makes use of several datasets to train its models effectively. Among these datasets are CK+, FER-2013, and JAFFE. The CK+ dataset, sourced from Kaggle, is particularly noteworthy for its comprehensive coverage of various facial expressions. However, as depicted in Figure 2, these datasets may exhibit imbalances in the distribution of emotions, with varying instances of happiness, disgust, anger, sadness, fear, and contempt.

Despite the challenges posed by imbalanced datasets, DeepFace's sophisticated architecture and training methodologies enable it to effectively recognize and classify emotions in facial expressions. Through extensive training and optimization on diverse datasets, DeepFace achieves high levels of accuracy and robustness in emotion detection, making it a valuable tool for applications ranging from security and surveillance to user experience enhancement in digital platforms.



## **SMILE DETECTION AND SAVE THE PICTURE**

To implement smile detection and automatically take and save photos to a folder, you can use computer vision libraries such as OpenCV with Python. Here’s a high-level overview of how this can be achieved:

1. **Capture Video Feed**: First, you need to capture the live video feed from a webcam. OpenCV provides the `cv2.VideoCapture` method that can be used to initialize the camera and capture frames in real-time.
2. **Load a Pre-trained Model:** For smile detection, you can use a pre-trained Haar Cascade Classifier provided by OpenCV specifically for detecting smiles. This model has been trained to recognize smile patterns in images.
3. **Frame Processing**: As you receive each frame from the webcam:
   1. Convert the frame to grayscale using `cv2.cvtColor`, as grayscale images are sufficient for feature detection and reduce computational complexity.
   2. Use the loaded Haar Cascade smile detector model to detect smiles in the frame. This is done using the `detectMultiScale` method, which identifies the coordinates of smile regions in the frame.
4. **Smile Detection Logic:** When a smile is detected in the frame:
   1. You can take a snapshot of the frame using OpenCV’s `imwrite` function, which saves the image to a specified folder.
   2. Optionally, you can also display the detected smile on the screen by drawing a rectangle around it using `cv2.rectangle`.
5. **Save the Photo:** If a smile is detected and you have taken a snapshot, save the frame as an image file in a predefined folder using `cv2.imwrite`. You can specify the path and filename where the image should be saved.
6. **Release Resources:** After the detection is done or upon a certain condition (like pressing a key), release the video capture and close any OpenCV windows to free up resources.

Replace `'folder\_path/smile.jpg'` with the actual path where you want to save the photos. Make sure to install OpenCV by running `pip install opencv-python` if you haven’t already. This simple setup helps in understanding the core functionalities of capturing video, detecting features, and saving outcomes, suitable for applications such as interactive installations or user experience studies.Emotion detectors are used in many industries, one being the media industry where it is important for the companies to determine the public reaction to their products. In this article, we are going to build a smile detector using OpenCV which takes in live feed from webcam. The smile/happiness detector that we are going to implement would be a raw one, there exist many better ways to implement it.

**Step # 1: First of all, we need to import the OpenCV library.**

import cv2

**Step #2: Include the desired haar-cascades.**

Haar-cascades are classifiers that are used to detect features (of face in this case) by superimposing predefined patterns over face segments and are used as XML files. In our model, we shall use face, eye and smile haar-cascades, which after downloading need to be placed in the working directory.

All the required Haar-cascades can be found here.

face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +'haarcascade\_frontalface\_default.xml')

eye\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +'haarcascade\_eye.xml')

smile\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +'haarcascade\_smile.xml')

Step #3:

In this step, we are going to build main function which would be performing the smile detection.

The live feed coming from the webcam/video device is processed frame by frame. We process the gray scale image, as haar-cascades work better on them.

To detect the face, we use:

faces = face\_cascade.detectMultiScale(gray, 1.3, 5)

where 1.3 is the scaling factor, and 5 is the number of nearest neighbors. We can adjust these factors as per our convenience/results to improve our detector.

Now for each subsequent face detected, we need to check for smiles.

**Explanations**

The face data is stored as tuples of coordinates. Here, x and y define the coordinate of the upper-left corner of the face frame, w and h define the width and height of the frame.

The cv2.rectangle function takes in the arguments frame, upper-left coordinates of the face, lower right coordinates, the RGB code for the rectangle (that would contain within it the detected face) and the thickness of the rectangle.

The roi\_gray defines the region of interest of the face and roi\_color does the same for the original frame.

In line 7, we apply smile detection using the cascade.

**Step #4:**

We define main function in this step. After execution, the function can be terminated by pressing the “q” key.

**Output:**



## **LIBRARIES AND PACKAGES**

## **OPENCV**

OpenCV (Open Source Computer Vision Library) is a widely used open-source library for computer vision and machine learning software. It enables real-time processing of image and video data and supports integration with various deep learning frameworks like TensorFlow, Caffe, and PyTorch. This versatility allows OpenCV to be used for a range of tasks from simple image processing, motion detection, to more complex operations such as object detection and face recognition.

## **ARGPARSE**

argparse is a Python library used for creating command-line interfaces. It simplifies the process of parsing command-line arguments and options, making it easier to build robust and user-friendly command-line programs.

## **TENSORFLOW**

TensorFlow is a popular open-source machine learning framework developed by Google. It's known for its flexibility, scalability, and comprehensive ecosystem, making it suitable for building and deploying machine learning models across various platforms. TensorFlow offers high performance, supports multiple abstraction levels, and has a large and active community providing support and resources.

## **DEEPFACE PACKAGE**

DeepFace is a deep learning facial recognition system created by a research group at Facebook. It identifies human faces in digital images. The program employs a nine-layer neural network with over 120 million connection weights and was trained on four million images uploaded by Facebook users.

## **HAAR CASCADES FOR FACE DETECTION**

Haar cascades are a machine learning object detection method used to identify objects in images or video streams, notably faces. It is called a "cascade" because the algorithm employs a cascade function that quickly discards non-face regions in an image, thereby focusing on potential face areas for further analysis. This method, while being more accurate, is generally slower than some alternatives, such as the LBP (Local Binary Patterns) Cascade Classifier.

**Accuracy vs Speed:** Haar classifiers are preferred for their accuracy despite their slower processing time compared to LBP classifiers. This makes Haar cascades suitable for applications where precision is more critical than real-time analysis speed.

**Application in OpenCV:** In the context of OpenCV, Haar cascades involve pre-trained models that are used to detect faces in images or video frames. Once faces are detected, additional processing can be performed, such as gender and age detection.

## **THE CNN ARCHITECTURE**

The convolutional neural network for this python project has 3 convolutional layers:

* Convolutional layer; 96 nodes, kernel size 7
* Convolutional layer; 256 nodes, kernel size 5
* Convolutional layer; 384 nodes, kernel size 3

It has 2 fully connected layers, each with 512 nodes, and a final output layer of softmax type.

To go about the python project, we’ll:

* Detect faces
* Classify into Male/Female
* Classify into one of the 8 age ranges
* Put the results on the image and display it

## **BASIC CONCEPT OF CNN ARCHITECTURE**

Convolutional Neural Networks (CNNs) are designed to process data that come in the form of multiple arrays, such as a color image composed of three 2D arrays containing pixel intensities in the RGB color space. The architecture of a CNN is engineered to take advantage of the 2D structure of an input image by preserving the relationships between pixels. This is achieved through several key layers:

* **Convolutional Layer:** This is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field but extend through the full depth of the input volume. As the filter slides (or convolves) around the input image, it produces a 2D activation map that gives the responses of that filter at every spatial position. Intuitively, the network learns filters that activate when they see some specific type of feature at some spatial position in the input.
* **Activation Function:** After each convolution operation, the result passes through a nonlinear activation function, like the Rectified Linear Unit (ReLU). This step is crucial because it introduces nonlinear properties to the system, allowing the network to learn more complex features.
* **Pooling Layer:** Following the convolutional layer and activation function, the pooling (or subsampling or downsampling) layer reduces the dimensionality of each feature map but retains the most important information. Max pooling, for example, reduces the size of the input by taking the maximum value of a group of cells in a feature map.
* **Fully Connected Layer:** After several convolutional and pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can thus be computed with a matrix multiplication followed by a bias offset.
* **Output Layer:** The final layer uses a softmax activation function (in case of classification tasks) to output a probability distribution over the class labels.

## **APPLICATION IN GENDER AND AGE DETECTION**

In the context of gender and age detection:

**Data Preprocessing:** The input images are preprocessed to standardize their size and orientation, ensuring the network receives inputs in a consistent format. Data augmentation techniques like rotations, flips, and variations in lighting conditions are applied to enrich the training dataset and make the model more robust.

**Feature Learning:** Through its convolutional layers, the CNN automatically learns to identify features that are relevant for distinguishing between different genders and age groups. Early layers may detect simple features like edges and textures, while deeper layers can identify more complex features like facial structures.

**Classification:** The fully connected layers of the network use the learned features to classify the input images into gender categories (e.g., male or female) and predict age ranges. The output layer provides a probability distribution over the classes, and the class with the highest probability is chosen as the prediction.

**Evaluation:** The model's performance is evaluated using metrics like accuracy, precision, recall, and F1-score. These metrics help to understand how well the model is performing in terms of correctly identifying genders and accurately estimating ages.

By leveraging the automatic feature extraction capabilities of CNNs, gender and age detection systems can achieve high accuracy without the need for manual feature selection, making CNNs a powerful tool for applications in advertising, security, healthcare, and beyond.

## **THE DATASET**

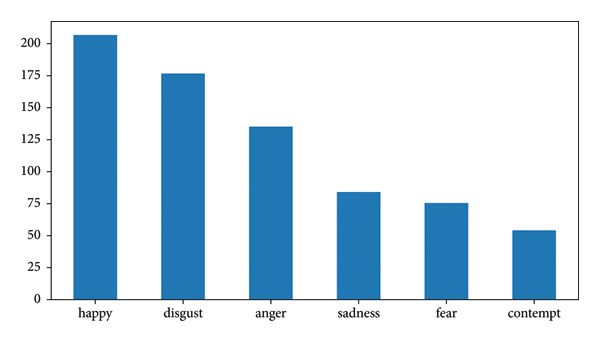
Tal Hassner and Gil Levi are renowned for their contributions to the field of computer vision, particularly in the context of age and gender classification. Their work, which includes developing models and datasets for these purposes, has been widely recognized and utilized in the computer vision community.

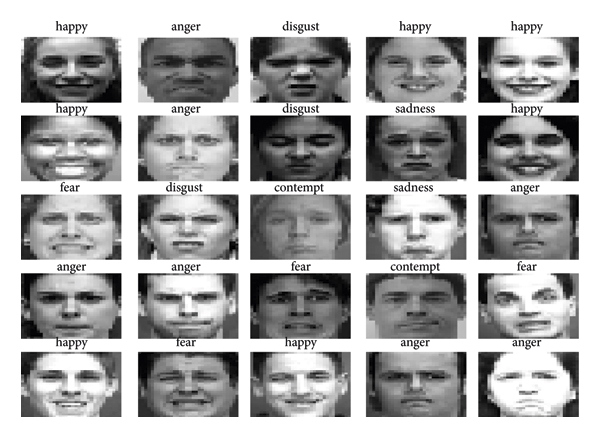
**Dataset Description:**The dataset created and used by Hassner and Levi is specifically designed for age and gender classification tasks. It typically includes a diverse set of facial images collected from various sources, annotated with labels indicating the subjects' gender and age or age group. Key characteristics of such datasets include:

* Diversity in Age and Gender: The dataset includes a balanced representation of different ages and genders to ensure the model's accuracy across diverse demographics.
* Variety of Backgrounds and Lighting Conditions: Images are collected under various lighting conditions and backgrounds to improve the model's robustness and its ability to generalize across different real-world scenarios.
* Facial Expressions and Poses: The dataset likely encompasses a wide range of facial expressions and head poses to train models that are less sensitive to these variations.
* Quality and Resolution: The images vary in quality and resolution, reflecting the real-world conditions in which facial recognition systems operate.

# **DATASETS FOR EMOTIONS**

This work makes use of three datasets: CK+ [37], FER-2013 [38], and JAFFE [39] dataset. The CK+ dataset was gotten from the Kaggle website [37]. Figure [2](https://www.hindawi.com/journals/acisc/2023/2457898/fig2/) depicts the facial recognition of imbalanced dataset. The original dataset comprises 732 images with unequal instances of happiness (207), disgust (177), anger (135), sadness (84), fear (75) and contempt (54) as shown in

[[](https://www.hindawi.com/journals/acisc/2023/2457898/fig2/)](https://www.hindawi.com/journals/acisc/2023/2457898/fig2/" \t "_blank)

[[](https://www.hindawi.com/journals/acisc/2023/2457898/fig3/)](https://www.hindawi.com/journals/acisc/2023/2457898/fig3/" \t "_blank)

## **MODELS TRAINED**

Hassner and Levi's work focuses on leveraging deep learning, particularly Convolutional Neural Networks (CNNs), for age and gender classification. Their models are trained using the aforementioned datasets, following a methodology that likely includes:

* Preprocessing: Images are preprocessed to a standard size and format, possibly including face detection and alignment to ensure that the models focus on the relevant features.
* Architecture Design: The CNN architecture is designed to extract and learn the most relevant features from the facial images for age and gender classification. This includes layers that can capture both the low-level features (such as edges and textures) and high-level attributes (such as overall facial structure).
* Training Process: The models are trained using a subset of the dataset, employing techniques like data augmentation (e.g., random flips, rotations) to enhance the diversity of training examples and improve the model's generalization capabilities.
* Validation and Testing: The trained models are validated and tested on separate subsets of the dataset not seen during training. This step is crucial for evaluating the model's performance and its ability to generalize to new, unseen data.

## **IMPACT AND APPLICATIONS**

The models trained by Hassner and Levi have significant implications for various applications, including security and surveillance, personalized advertising, human-computer interaction, and demographic research. Their work not only advances the field of computer vision but also raises important questions regarding privacy, ethics, and the responsible use of AI technology.

Their research contributions are often shared publicly, allowing for widespread use and further development by the research community and industry practitioners. This openness fosters innovation and accelerates improvements in the accuracy and efficiency of age and gender classification systems.

## **PREREQUISITES**

You’ll need to install OpenCV (cv2) to be able to run this project. You can do this with pip-

pip install opencv-python

Other packages you’ll be needing are math and argparse, but those come as part of the standard Python library.

### **Files of gender and age detection python project**

* opencv\_face\_detector.pbtxt
* opencv\_face\_detector\_uint8.pb
* age\_deploy.prototxt
* age\_net.caffemodel
* gender\_deploy.prototxt
* gender\_net.caffemodel
* a few pictures to try the project on

The project setup includes the following files:

* + **opencv\_face\_detector.pbtxt:** This file holds the graph definition in text format for the OpenCV face detector model.
  + **opencv\_face\_detector\_uint8.pb:** It contains the trained weights of the OpenCV face detector model in binary format.
  + **age\_deploy.prototxt:** Describes the network configuration for age classification.
  + **age\_net.caffemodel:** Defines the internal states of the parameters for age classification.
  + **gender\_deploy.prototxt:** Describes the network configuration for gender classification.
  + **gender\_net.caffemodel:** Defines the internal states of the parameters for gender classification.
  + A few pictures to try the project on: These are sample images on which the project can be tested.
  + **detect.py:** This Python script is used to perform face detection, age classification, and gender classification on images or video streams.

For face detection, the opencv\_face\_detector.pb file holds the graph definition and trained weights of the model. The .pbtxt extension represents the protobuf in text format. Similarly, for age and gender classification, the .prototxt files describe the network configuration, while the .caffemodel files define the internal states of the parameters of the layers. These files are essential for running the trained models and performing accurate classification.

The detect.py script utilizes these files to run the trained models for face detection, age classification, and gender classification. It captures images or video streams, detects faces, predicts the age and gender of the detected faces, and displays the results.

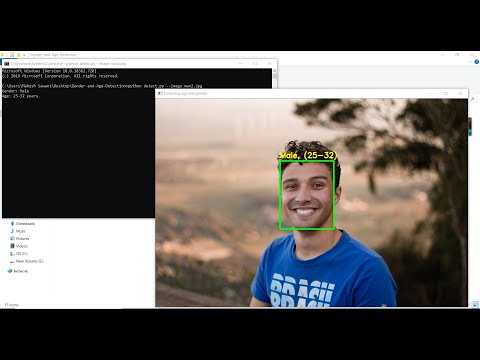
This setup enables the implementation of a comprehensive face detection, age estimation, and gender classification system using pre-trained deep learning models.

## **PROJECTS SETUPS BY FOLLOWING STEPS:**

Sure, here's a simplified version of the steps:

* + - **Model Setup:** We have files that hold the trained model's information. For face detection, we use a .pb file, which contains the model's graph definition and trained weights. For age and gender classification, we have .prototxt files for network configuration and .caffemodel files for internal parameters.
    - **Argument Parsing:** We use the argparse library to create a tool that lets us provide the image's path through the command prompt. This helps us classify the gender and age of the person in the image.
    - **Model Initialization:** We prepare the model for face, age, and gender classification.
    - **Setup Mean Values:** We set up some standard values that the model will use during classification.
    - **Loading Networks:** We load the trained models using the readNet() method, providing the weights and configuration.
    - **Capture Video Stream:** If needed, we can capture video from a webcam for real-time classification.
    - **Stream Processing:** We continuously read the video stream and store it. If it's not a video, we wait for input.
    - **Face Detection:** We use a face detection model to find faces in the image or video frame. If faces are found, we store their coordinates.
    - **Processing Detected Faces:** For each detected face, we prepare it for classification by scaling and resizing.
    - **Gender Classification:** We classify the gender of each face by feeding it into the gender classification model and checking the confidence levels.
    - **Age Classification:** Similarly, we classify the age of each face using the age classification model.
    - **Emotion Classification :** Similarly, we classify the emotions of each face using the emotion classification model prototype file.
    - **Display Results:** Finally, we add the predicted gender and age to the image and display it.

## **WORKING:**

[](https://youtu.be/ReeccRD21EU)

## **TESTING PHASES:**

There is 2 phases of testing as 2 seprate modules gender age and emotion detection are added so 2 testing divides into 2 phase

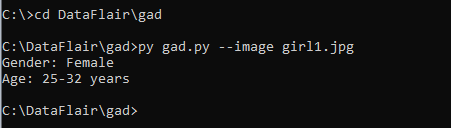
* **Phase 01:** only detect age and gender
* **Phase 02:** detect gender age and emotion
* **Phase 03:** detect smile and take photo and svae it in folder

## **PHASE 01 : PYTHON PROJECT EXAMPLES FOR GENDER AND AGE DETECTION**

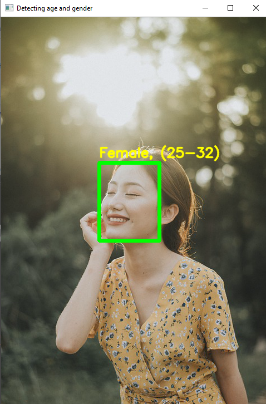
Let’s try this gender and age classifier out on some of our own images now.

We’ll get to the command prompt, run our script with the image option and specify an image to classify:

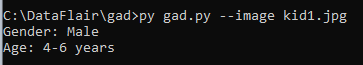
**Example 1**



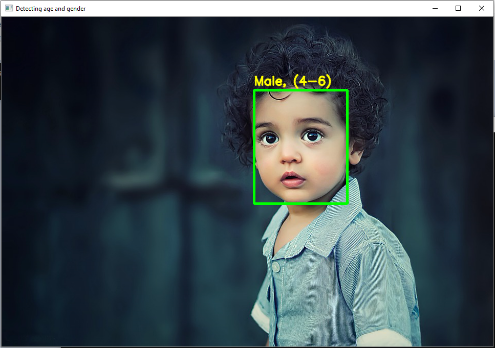
**Output:**



**Example 2**

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/intermediate-python-project-example.png)

**Output:**

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/advanced-python-project-example.png)

**Example 3**

[python open source project](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/python-project-example.png)

**Output:**

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/python-project-for-practice.png)

**Example 4**

[python projects for practice](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/python-projects-for-practice-woman.png)

**Output:**

>python detect.py --image woman1.jpg

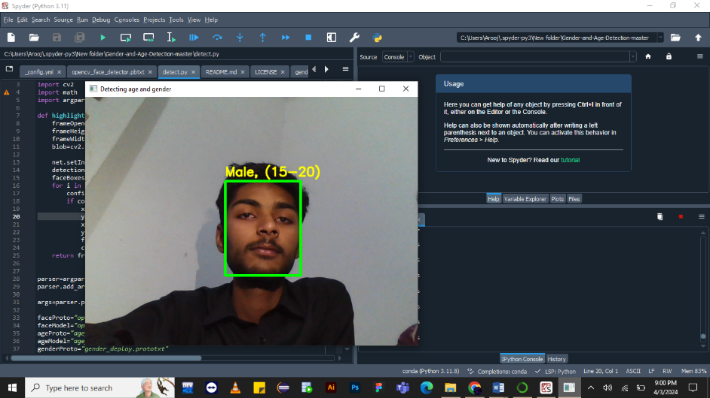
Gender: Female

Age: 38-43 years

[](https://data-flair.training/blogs/wp-content/uploads/sites/2/2019/09/interesting-python-project-3.png)

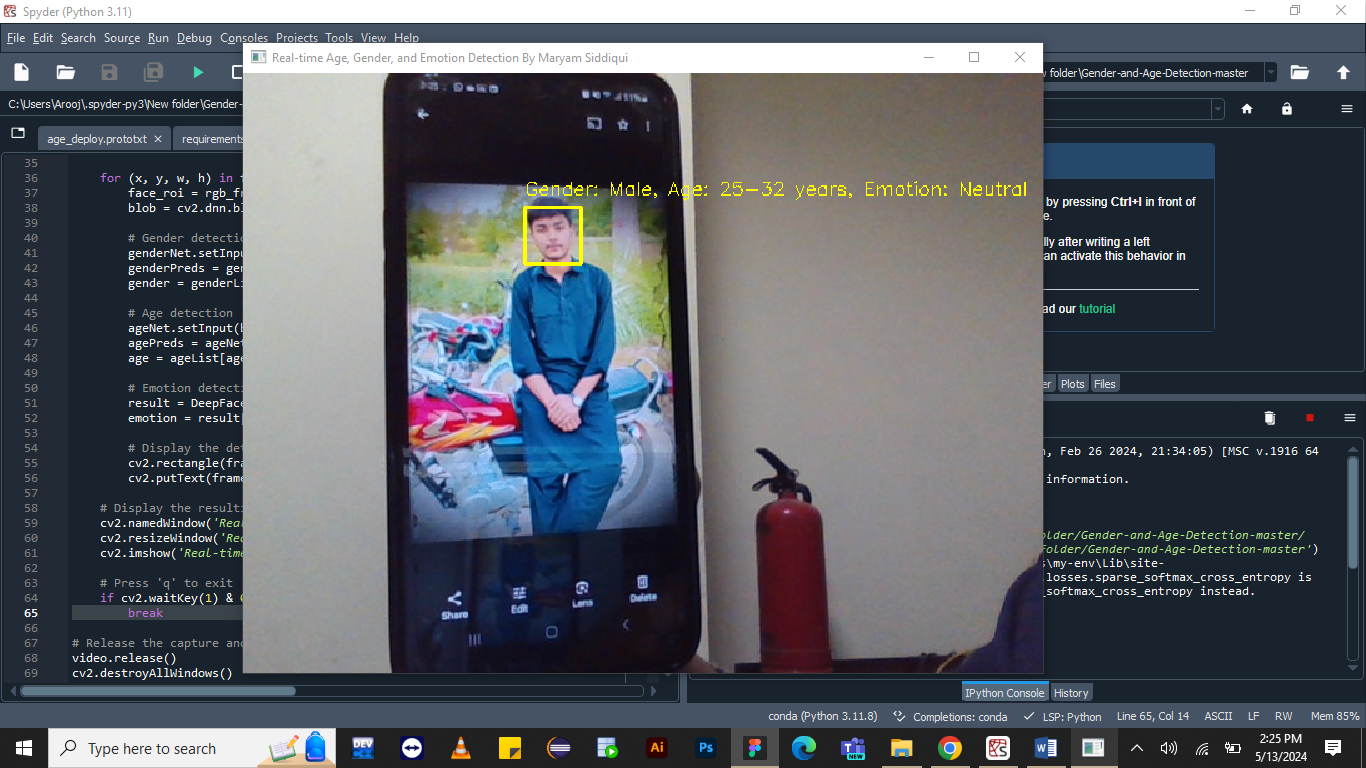
## **DETECTING AGE AND GENDER FROM WEBCAME CAPTURE IMAGES:**

Running the code on the webcam, I conducted an experiment on my brother Ali, who is 19 years old, to detect his age and gender from the captured images.

****

## **PHASE 02: DETECTING AGE , EMOTIONS AND GENDER FROM WEBCAME CAPTURE IMAGES:**

Running the code on the webcam, I conducted an experiment on my brother Ali, who is 19 years old, to detect his age and gender from the captured images.



## 

## **PHASE 03: DETECT SMILE AND TAKE PHOTO AND SVAE IT IN FOLDER**

Phase 03 of the project involves implementing smile detection functionality and capturing photos of individuals when they smile, then saving these images in a designated folder. Here's a breakdown of this phase:

**Smile Detection:**

* 1. Smile detection is achieved using a pre-trained Haar cascade classifier specifically designed to recognize smiling faces. Haar cascades are a popular method for object detection in images and videos.
  2. The smile cascade classifier is applied to the detected faces in the webcam feed. It analyzes each face region and identifies whether a smile is present based on certain visual features.
  3. When a smile is detected, the corresponding region of the face is marked with a bounding box or rectangle to visually indicate the detected smile.

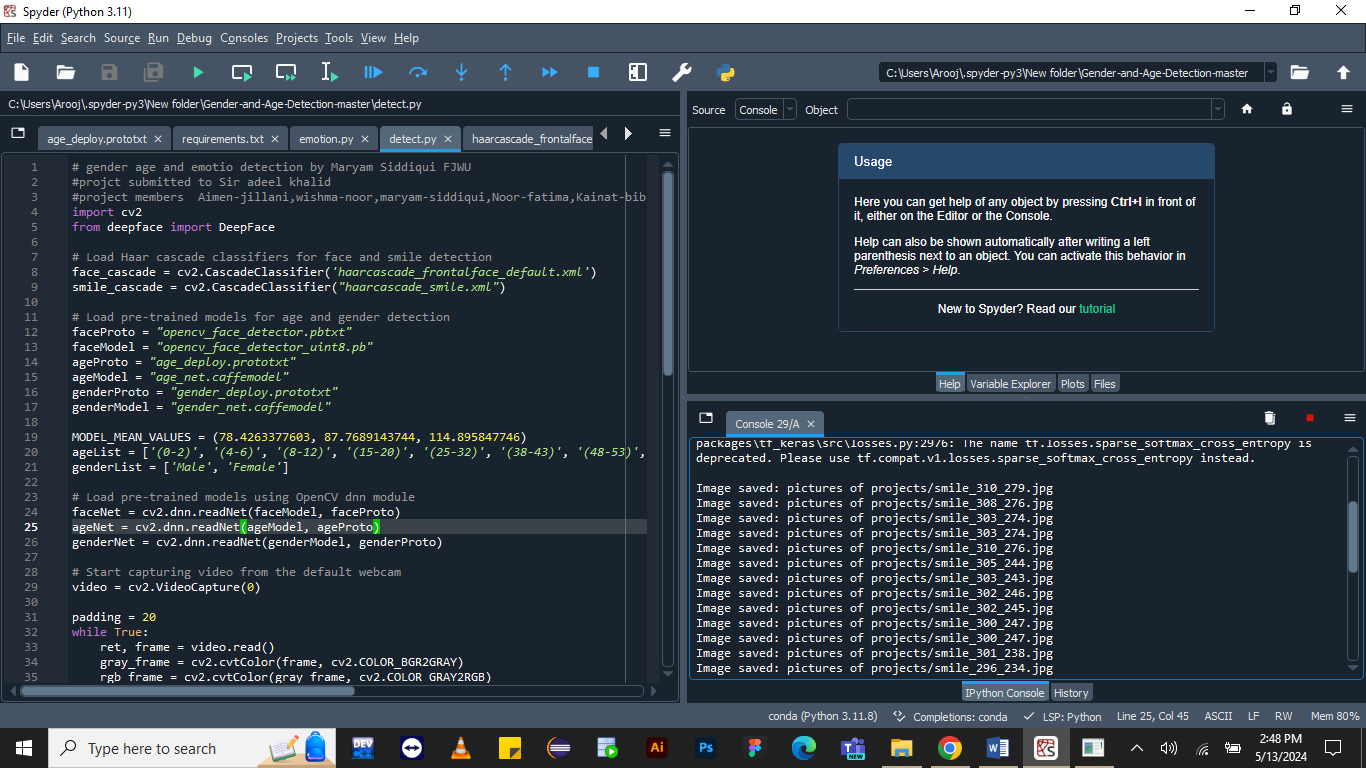
**Capturing Photos:**

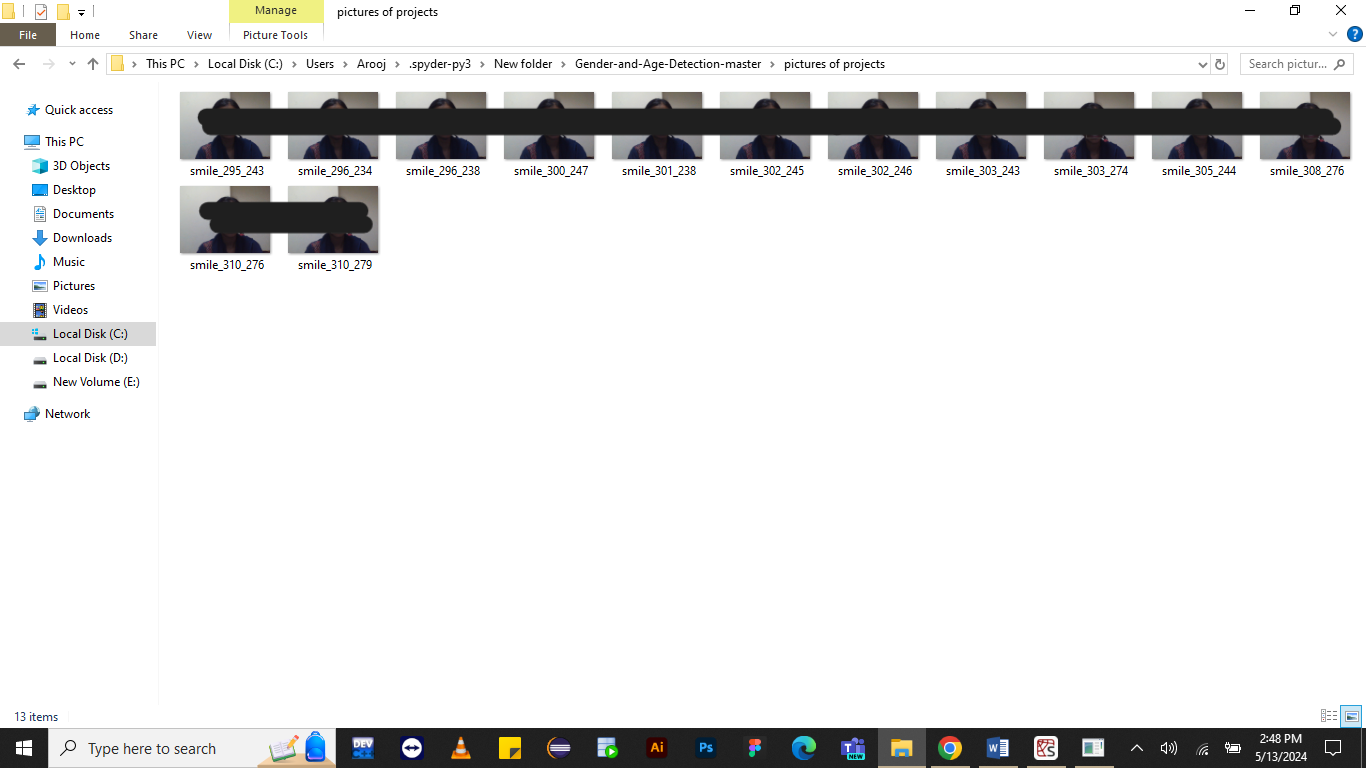
* 1. Upon detecting a smile, the script captures the current frame of the webcam feed, which includes the smiling individual. This is typically achieved using OpenCV's `imwrite()` function, which allows saving the image to the system's storage.
  2. The captured image is saved temporarily in memory before being saved to a specified directory.

**Saving Images to a Folder:**

* 1. After capturing the image of the smiling individual, the script saves the image to a designated folder on the system. This folder path is predefined in the script or specified by the user.
  2. Each saved image is usually named uniquely to avoid overwriting existing images. This can be accomplished by appending a timestamp or an incremental number to the image file name.
  3. The saved images can be accessed later for further analysis, processing, or storage.

Overall, this phase of the project enhances user interaction by capturing moments of joy or happiness through smile detection. By saving these images, the application provides a way to preserve and document these moments, which can be valuable for various purposes such as user engagement, sentiment analysis, or simply as mementos of positive experiences. Additionally, it showcases the integration of smile detection functionality into real-world applications, demonstrating the practical applications of computer vision and facial recognition technologies.





## **CONCLUSION:**

This project showcases the capabilities of deep learning within the domain of computer vision, particularly in the domain of recognizing and categorizing human facial attributes such as gender and age. Despite facing challenges such as variations in imaging conditions and the complexities of human facial features, the application achieves impressive accuracy levels owing to the robustness of Convolutional Neural Network (CNN) models and the extensive Adience dataset it relies on.

By leveraging CNNs, the project is able to effectively learn and extract features from facial images, allowing it to accurately classify individuals into gender and age groups. The Adience dataset, which serves as the foundation for training the model, provides a diverse range of facial images captured under various conditions, enabling the model to generalize well to real-world scenarios.

Moving forward, there are several avenues for enhancing the project. Firstly, expanding the dataset by incorporating more diverse samples and covering a wider range of demographics could further improve the model's performance and generalization capabilities. Additionally, refining the model to recognize more nuanced categories within gender and age groups could lead to more granular and precise classifications.

Furthermore, optimizing the algorithm for real-time applications would be beneficial, allowing for faster and more efficient processing of facial data. This could involve techniques such as model compression, quantization, or implementing specialized hardware accelerators.

Overall, this project not only serves as a practical tool for gender and age detection but also lays the groundwork for continued research and development in the field of AI-driven biometric analysis. Its success highlights the potential of deep learning approaches in addressing complex computer vision tasks and underscores the importance of high-quality datasets in training accurate and robust models.

This code is a Python script that utilizes computer vision and deep learning techniques to perform real-time detection and classification of gender, age, emotion, and smile in a webcam feed. Let's break down its features and benefits:

* + 1. Facial Detection and Recognition: The script uses Haar cascade classifiers to detect faces in the webcam feed. Once a face is detected, it extracts the region of interest (ROI) and passes it to the deep learning models for further analysis.
    2. Gender and Age Detection: Pre-trained models for gender and age detection are loaded using OpenCV's dnn module. These models are capable of predicting the gender (male or female) and age group (e.g., 0-2 years, 25-32 years) of the detected faces.This feature can be useful in various applications such as demographic analysis, targeted advertising, and personalized user experiences.
    3. Emotion Detection: The script utilizes the DeepFace package to perform emotion detection on the detected faces. It analyzes the facial expressions and identifies the dominant emotion present (e.g., happy, sad, angry). Emotion detection can be valuable in applications like sentiment analysis, customer feedback analysis, and mental health monitoring.
    4. Smile Detection:The code also incorporates smile detection using another Haar cascade classifier. When a smile is detected, the script captures the image and saves it to a specified directory. Smile detection can enhance user interaction in applications like virtual assistants, interactive games, and customer satisfaction analysis.
    5. Real-time Processing: The script continuously captures frames from the webcam and performs all the detection and classification tasks in real-time. This ensures that the results are immediately available and can be used for various interactive applications.
    6. Modularity and Extensibility: The script is modular and extensible, allowing for easy integration of additional features or customization based on specific requirements. It serves as a foundation for building more advanced applications in areas such as security surveillance, human-computer interaction, and assistive technologies.

Overall, this code demonstrates the power of combining computer vision techniques with deep learning models to perform sophisticated facial analysis tasks in real-time. It showcases the potential applications of AI-driven biometric analysis and provides a framework for further research and development in this field.

## 

## **REFERENCES:**

* <https://www.sciencedirect.com/science/article/pii/S2214785320362222#b0030>
* <https://www.kdnuggets.com/2019/04/predict-age-gender-using-convolutional-neural-network-opencv.html>
* <https://www.geeksforgeeks.org/age-and-gender-prediction-using-cnn/>
* <https://medium.com/@skillcate/gender-detection-model-using-cnn-a-complete-guide-279706e94fdb>