UNIVERSITY OF MADRAS

GUINDY CAMPUS, CHENNAI- 600 085.

DEPARTMENT OF COMPUTER SCIENCE

ADVANCED JAVA PROGRAMMING

TEAM PROJECT

TOPIC

FACE DETECTION AND AGE PREDICTION

Submitted by

(I M.Sc. Computer Science)

AKSHAYA PRIYA P J	**36823101**
KAVITHA G	**36823103**
SANGEETHA K	**36823106**
SWATHY B	**36823107**

Under the Guidance of

Ms. S. LAKSHMI BALA

UNIVERSITY OF MADRAS

DEPARTMENT OF COMPUTER SCIENCE

GUINDY CAMPUS, CHENNAI- 600 085.

This is to certify that the Document work entitled as **"FACE DETECTION AND AGE PREDICTION"** submitted to the University Of Madras,Chennai in partial fulfillment of the requirement for the award of the degree MASTER OF SCIENCE IN COMPUTER SCIENCE is a Bonafied record of the Mini-Project carried out the work has been successfully completed by our team during the period of NOVEMBER 2023- DECEMBER 2023.

BONAFIDE CERTIFICATE

Staff - in – charge
(Ms. S. LAKSHMI BALA)

Head of the Department
(Prof.Dr.S.GOPINATHAN)

ACKNOWLEDGEMENT

We take this opportunity to acknowledge with great pleasure, deep satisfaction and gratitude, to the contribution of many individuals int the successful completion of the project.

We express our sincere and profound gratitude to our Head of the Department **Dr. S.GOPINATHAN** for his encouragement and support to do our project.

We owe deep gratitude to our project guide **Ms. S.LAKSHMI BALA,** who took interest on our project work and guided us,till the completion of our project work by providing all the necessary information.

We take this opportunity to thank all the staff members and the Lab Administrators of the Computer Science Department who rendered their help directly to finish our project in time. We would like to express our hearty thanks to our parents, without whom we would not have come to this level in our life. Our hearty thanks to our friends and well-wishers who supported and encouraged us to complete this project successfully.

TABLE OF CONTENTS

I.PROJECT OBJECTIVE

To create a **FACE DETECTION AND AGE PREDICTION** using java that detects the face, predicts the age and their gender.

II.ABSTRACT

This project aims to develop a Java-based application for real-time face detection and subsequent age & gender prediction from detected faces. The system utilizes computer vision techniques and machine learning models to achieve these functionalities. This project focuses on developing a Java-based system capable of detecting faces within images or video streams and subsequently predicting the approximate age and gender of the detected individuals.

III.INTRODUCTION

Automatic age and gender classification has become relevant to an increasing amount of applications, particularly since the rise of social platforms and social media. Nevertheless, performance of existing methods on real-world images is still significantly lacking, especially when compared to the tremendous leaps in performance recently reported for the related task of face recognition.

SYSTEM OVERVIEW

In an era of burgeoning digital technology, the ability to discern and interpret human characteristics from visual data stands as a pivotal innovation. Face detection, coupled with age and gender prediction, constitutes a fundamental facet of computer vision systems, playing a pivotal role in diverse domains such as security, marketing analytics, and human-computer interaction. This project delves into the development of a Java-based system adept at analyzing visual data, specifically targeting the identification of human faces within images or video streams. The primary objective revolves around accurately detecting faces and subsequently predicting the approximate age and gender of the individuals depicted.

TERMINOLOGIES

- ### Face Detection

  Facilitated by the OpenCV (Open Source Computer Vision) library, the project aims to leverage robust algorithms for face detection. Through the integration of a pre-trained Haar Cascade classifier, the system endeavors to locate facial features, delineating bounding boxes around detected faces in provided visual data.

- ### Age Prediction

  Age estimation serves as a pivotal aspect of this system, striving to predict the age range or exact age of the individuals captured within the detected facial regions. This entails employing machine learning models trained on extensive facial age datasets. Upon

face detection, the identified facial features undergo preprocessing and feature extraction, facilitating the utilization of these features by the trained age prediction model.

- **Gender Prediction**

Similarly, gender prediction constitutes another crucial dimension of this project. By utilizing machine learning models optimized for gender classification, the system endeavors to assign gender labels (e.g., male, female) to the detected faces. The isolated facial regions undergo transformations to align with the requirements of the gender prediction model, enabling accurate gender attribution.

- **Java Framework**

The Java programming language serves as the cornerstone of this project's implementation, providing a robust and versatile platform for integrating the various functionalities. The amalgamation of OpenCV for face detection and machine learning models for age and gender prediction encapsulates the core architecture of this system.

IV.SYSTEM STUDY

EXISTING APPROACH

Traditional approaches for face detection involve algorithms like HOG (Histogram of Oriented Gradients). Age estimation might use methods like feature extraction (wrinkles, texture) or regression on facial landmarks. Gender prediction can rely on features like hair length or facial structure, analyzed via classifiers such as SVMs (Support Vector Machines) or decision trees. These methods often use handcrafted features and statistical models for predictions.

PROPOSED APPROACH

The proposed solution involves employing a Haar Cascades and Convolutional Neural Network (CNN) architecture for comprehensive face detection and accurate age-gender prediction. Utilizing CNN's deep learning capabilities, the model will learn intricate facial features, allowing simultaneous detection of faces and extraction of fine-grained details for precise age estimation and gender classification. This CNN-based approach aims for superior accuracy and robustness in facial analysis tasks.

DATASET

Our method is implemented using the Caffe open-source framework [26]. Training was performed on an Amazon GPU machine with 1,536 CUDA cores and 4GB of video memory. Training each network required about four hours, predicting age or gender on a single image using our network requires about 200ms. Prediction running times can conceivably be substantially improved by running the network on image batches.

GOALS OF THE PROJECT

The primary goals are to develop a robust CNN-based system for face detection and precise age-gender prediction. Emphasizing accuracy, the system aims to autonomously identify faces within images, estimate age with fine granularity, and accurately classify gender, leveraging deep learning to enhance facial analysis in diverse contexts.

V.SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

• Processor: Intel CORE i5 series

• Processor Speed: 1.80 GHz

• RAM: 4 GB RAM

• HDD: 500 GB and above

• Resolution: 1920*1080

SOFTWARE REQUIREMENTS

• Operating system: Windows 10

• Language: JAVA.

• Software: Netbeans.

VI.WORKING

STEP 1: Start the program

STEP 2: The system begins by obtaining input data in real-time. These data sources serve as the foundation for face detection and subsequent age-gender prediction.

STEP3: It scans through the input data, identifies facial features, and marks regions of interest by drawing bounding boxes around detected faces.

STEP4: The identified facial regions within the bounding boxes are extracted from the input data. These regions serve as the basis for age and gender prediction.

STEP 5: The extracted facial regions undergo preprocessing to isolate pertinent features which provides an estimate of the individual's age or an age range.

STEP 6: The model assigns gender labels (e.g., male, female) to the detected faces.

STEP 7: The system presents the processed results.

VII.MODULES

The Java programming language serves as the primary framework for integrating these functionalities. The OpenCV library assists in face detection, providing algorithms for face localization within images or real-time video feeds. The age and gender prediction models, trained using appropriate datasets, are incorporated into the system to deliver accurate predictions.

FACE PREDICTION MODULE

Utilizes OpenCV or a similar library to detect faces within the provided input data.Implements algorithms (e.g., Haar Cascade classifier, deep learning-based models) for accurate face localization and bounding box identification.

AGE PREDICTION MODULE

Performs feature extraction and prediction on the isolated facial regions to estimate age or age ranges.

GENDER PREDICTION MODULE

Processes facial regions to predict the gender (e.g., male, female) of the individuals depicted in the images or video frames.

JAVA OPENCV MODULE

OpenCV's versatility in face detection, facial landmark detection, and real-time processing makes it a valuable tool for developing facial analysis applications, particularly when integrated with machine learning models for age and gender prediction.

CAFFE MODEL AGE MODULE

AgeNet utilizes a deep convolutional neural network to estimate the age of individuals in facial images.Caffe's AgeNet model operates using a deep neural network architecture, leveraging convolutional layers to extract facial features and predict age ranges or precise ages from facial images.

CAFFE MODEL GENDER MODULE

Utilizing Caffe for gender detection demands understanding model architecture, dataset preparation, training, and model integration within Java-based systems, enabling accurate predictions in facial analysis tasks.

VIII.ALGORITHM SPECIFICATION

CONVOLUTION NEURAL NETWORK

A Convolutional Neural Network (CNN) is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers.The convolutional layers are the key component of a CNN, where filters are applied to the input image to extract features such as edges, textures, and shapes. The output of the convolutional layers is then passed through pooling layers, which are used to down-sample the feature maps, reducing the spatial dimensions while retaining the most important information.

WHY CONVOLUTION NEURAL NETWORK?

CNNs (Convolutional Neural Networks) are particularly well-suited for tasks like face detection, age estimation, and gender prediction due to several inherent advantages:

1. **FEATURE LEARNING:** CNNs excel in learning hierarchical representations of data. In the context of facial analysis, they automatically extract features like edges, textures, and patterns from images, crucial for recognizing facial structures and attributes.

2. **SPATIAL HIERARCHIES**: They preserve spatial relationships within images by using convolutional and pooling layers. This is essential in understanding facial features where the spatial arrangement of pixels plays a significant role (like the arrangement of eyes, nose, and mouth).

3. **PARAMETER SHARING:** CNNs use parameter sharing in convolutional layers, reducing the number of learnable parameters compared to fully connected networks. This property allows them to generalize better with fewer training samples, critical when working with limited datasets.

4. **TRANSLATION INVARIANCE:** Due to the use of convolutional layers, CNNs can detect features irrespective of their location in the image. This is beneficial for facial analysis, where the position of facial features may vary across images.

5. **ARCHITECTURE FLEXIBILITY:** CNNs offer a flexible architecture allowing for easy adaptation to various image sizes and complexities. This adaptability is crucial in handling different resolutions and quality of facial images encountered in real-world applications.

6. **TRANSFER LEARNING:** Pre-trained CNN models on large image datasets (like ImageNet) can be leveraged as a starting point for facial analysis tasks. Transfer learning enables using the knowledge gained from general image features and fine-tuning the model on specific facial analysis tasks, saving time and computational resources.

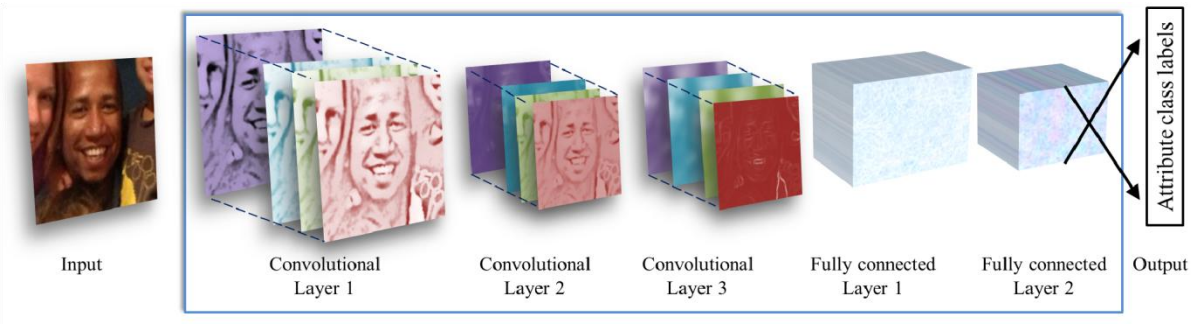
7. **STATE-OF-THE-ART PERFORMANCE:** CNNs have demonstrated state-of-the-art performance in various computer vision tasks, including face detection, age estimation, and gender prediction, making them a reliable choice for these applications.

8. **INTERPRETABILITY:** CNNs can offer insights into what features or patterns contribute to a prediction, aiding in understanding the reasoning behind the model's decisions, which is valuable in sensitive applications like facial analysis.

Overall, the ability of CNNs to automatically learn hierarchical representations from images, generalize well, and handle spatial relationships makes them an excellent choice for tasks involving facial analysis, providing accurate and reliable predictions of age and gender from facial images.

WORKING OF CONVOLUTION NEURAL NETWORK

The network contains three convolutional layers, each followed by a rectified linear operation and pooling layer. The first two layers also follow normalization using local response normalization [28]. The first Convolutional Layer contains 96 filters of 7×7 pixels, the second Convolutional Layer contains 256 filters of 5×5 pixels, The third and final Convolutional Layer contains 384 filters of 3 × 3 pixels. Finally, two fully-connected layers are added, each containing 512 neurons. See Figure 3 for a detailed schematic view and the text for more information


Attribute class labels
Input
Convolutional Layer 1
Convolutional Layer 2
Convolutional Layer 3
Fully connected Layer 1
Fully connected Layer 2
Output

VII.SOURCE CODE AND OUTPUT

JAVA CV

```
package org.imesha.examples.javacv;
import org.bytedeco.javacv.Frame;
import org.bytedeco.javacv.FrameGrabber;
import org.bytedeco.javacv.OpenCVFrameConverter;
import org.bytedeco.javacv.OpenCVFrameGrabber;
import org.bytedeco.opencv.opencv_core.Mat;
import org.bytedeco.opencv.opencv_core.Point;
import org.bytedeco.opencv.opencv_core.Rect;
import org.bytedeco.opencv.opencv_core.Scalar;
import org.imesha.examples.javacv.util.ImageUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import javax.swing.*;
import java.awt.*;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.image.BufferedImage;
import java.util.Map;

import static org.bytedeco.opencv.global.opencv_imgproc.*;


public class JavaCVExample {

                    private     static     final     Logger     logger     =
LoggerFactory.getLogger(JavaCVExample.class);

    private FrameGrabber frameGrabber;
              private    OpenCVFrameConverter.ToMat    toMatConverter    =    new
OpenCVFrameConverter.ToMat();
    private volatile boolean running = false;

    private HaarFaceDetector faceDetector = new HaarFaceDetector();
    private CNNAgeDetector ageDetector = new CNNAgeDetector();
    private CNNGenderDetector genderDetector = new CNNGenderDetector();

    private JFrame window;
    private JPanel videoPanel;

    public JavaCVExample() {
        window = new JFrame();
        videoPanel = new JPanel();
```

```
        window.setLayout(new BorderLayout());
        window.setSize(new Dimension(1280, 720));
        window.add(videoPanel, BorderLayout.CENTER);
        window.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                stop();
            }
        });
    }


    public void start() {

        frameGrabber = new OpenCVFrameGrabber(0);

        //frameGrabber.setFormat("mp4");
        frameGrabber.setImageWidth(1280);
        frameGrabber.setImageHeight(720);

        logger.debug("Starting frame grabber");
        try {
            frameGrabber.start();
             logger.debug("Started frame grabber with image width-height : {}-
{}", frameGrabber.getImageWidth(), frameGrabber.getImageHeight());
        } catch (FrameGrabber.Exception e) {
            logger.error("Error when initializing the frame grabber", e);
            throw new RuntimeException("Unable to start the FrameGrabber", e);
        }

        SwingUtilities.invokeLater(() -> {
            window.setVisible(true);
        });

        process();

        logger.debug("Stopped frame grabbing.");
    }


    private void process() {
        running = true;
        while (running) {
            try {

                final Frame frame = frameGrabber.grab();
```

```
                Map<Rect, Mat> detectedFaces = faceDetector.detect(frame);
                Mat mat = toMatConverter.convert(frame);

                detectedFaces.entrySet().forEach(rectMatEntry -> {
                                                                String   age   =
ageDetector.predictAge(rectMatEntry.getValue(), frame);
                                                CNNGenderDetector.Gender  gender  =
genderDetector.predictGender(rectMatEntry.getValue(), frame);
                    String caption = String.format("%s:[%s]", gender, age);
                    logger.debug("Face's caption : {}", caption);

                            rectangle(mat, new Point(rectMatEntry.getKey().x(),
rectMatEntry.getKey().y()),
                                    new Point(rectMatEntry.getKey().width() +
rectMatEntry.getKey().x(),          rectMatEntry.getKey().height()          +
rectMatEntry.getKey().y()),
                             Scalar.RED, 2, CV_AA, 0);

                    int posX = Math.max(rectMatEntry.getKey().x() - 10, 0);
                    int posY = Math.max(rectMatEntry.getKey().y() - 10, 0);
                                putText(mat, caption, new Point(posX, posY),
CV_FONT_HERSHEY_PLAIN, 1.0,
                             new Scalar(255, 255, 255, 2.0));
                });


                Frame processedFrame = toMatConverter.convert(mat);

                Graphics graphics = videoPanel.getGraphics();
                                                BufferedImage   resizedImage   =
ImageUtils.getResizedBufferedImage(processedFrame, videoPanel);
                SwingUtilities.invokeLater(() -> {
                    graphics.drawImage(resizedImage, 0, 0, videoPanel);
                });
            } catch (FrameGrabber.Exception e) {
                logger.error("Error when grabbing the frame", e);
                logger.error("ExceptionOccured")
            } catch (Exception e) {
                   logger.error("Unexpected error occurred while grabbing and
processing a frame", e);
            }
        }
    }


    public void stop() {
        running = false;
        try {
```

```
            logger.debug("Releasing and stopping FrameGrabber");
            frameGrabber.release();
            frameGrabber.stop();
        } catch (FrameGrabber.Exception e) {
            logger.error("Error occurred when stopping the FrameGrabber", e);
        }

        window.dispose();
    }

    public static void main(String[] args) {
        JavaCVExample javaCVExample = new JavaCVExample();

        logger.info("Starting javacv example");
        new Thread(javaCVExample::start).start();

        Runtime.getRuntime().addShutdownHook(new Thread(() -> {
            logger.info("Stopping javacv example");
            javaCVExample.stop();
        }));

        try {
            Thread.currentThread().join();
        } catch (InterruptedException ignored) { }
    }
}
```

FACE DETECTION

```
package org.imesha.examples.javacv;
import org.bytedeco.javacv.Frame;
import org.bytedeco.javacv.OpenCVFrameConverter;
import org.bytedeco.opencv.opencv_core.CvMemStorage;
import org.bytedeco.opencv.opencv_core.Mat;
import org.bytedeco.opencv.opencv_core.Rect;
import org.bytedeco.opencv.opencv_core.RectVector;
import org.bytedeco.opencv.opencv_objdetect.CascadeClassifier;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.io.File;
import java.util.HashMap;
import java.util.Map;

import static org.bytedeco.opencv.global.opencv_core.cvReleaseMemStorage;
```

```
public class HaarFaceDetector {

    private static final Logger logger =
LoggerFactory.getLogger(HaarFaceDetector.class);


    CascadeClassifier faceCascade;
    private CvMemStorage storage;
    private OpenCVFrameConverter.ToIplImage iplImageConverter;
    private OpenCVFrameConverter.ToMat toMatConverter;

    public HaarFaceDetector() {
        iplImageConverter = new OpenCVFrameConverter.ToIplImage();
        toMatConverter = new OpenCVFrameConverter.ToMat();

        try {
            File haarCascade = new
File(this.getClass().getResource("/detection/haarcascade_frontalface_alt.xml").
toURI());
            logger.debug("Using Haar Cascade file located at : {}",
haarCascade.getAbsolutePath());
            //haarClassifierCascade = new
CvHaarClassifierCascade(cvload(haarCascade.getAbsolutePath()));
            faceCascade = new
CascadeClassifier(haarCascade.getCanonicalPath());

        } catch (Exception e) {
            logger.error("Error when trying to get the haar cascade", e);
            throw new IllegalStateException("Error when trying to get the haar
cascade", e);
        }
        storage = CvMemStorage.create();
    }


    public Map<Rect, Mat> detect(Frame frame) {
        Map<Rect, Mat> detectedFaces = new HashMap<>();

        RectVector detectObjects = new RectVector();

        Mat matImage = toMatConverter.convert(frame);
        faceCascade.detectMultiScale(matImage, detectObjects);

        long numberOfPeople = detectObjects.size();
        for (int i = 0; i < numberOfPeople; i++) {
            Rect rect = detectObjects.get(i);
```

```
            Mat croppedMat = matImage.apply(new Rect(rect.x(), rect.y(),
rect.width(), rect.height()));
            detectedFaces.put(rect, croppedMat);
        }

        return detectedFaces;
    }

    @Override
    public void finalize() {
        cvReleaseMemStorage(storage);
    }
}
```

AGE DETECTION

```
package org.imesha.examples.javacv;

import org.bytedeco.javacpp.DoublePointer;
import org.bytedeco.javacv.Frame;
import org.bytedeco.opencv.opencv_core.Mat;
import org.bytedeco.opencv.opencv_core.Point;
import org.bytedeco.opencv.opencv_core.Size;
import org.bytedeco.opencv.opencv_dnn.Net;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.io.File;
import java.net.URISyntaxException;

import static org.bytedeco.opencv.global.opencv_core.*;
import static org.bytedeco.opencv.global.opencv_dnn.blobFromImage;
import static org.bytedeco.opencv.global.opencv_dnn.readNetFromCaffe;
import static org.bytedeco.opencv.global.opencv_imgproc.resize;

public class CNNAgeDetector {

    private static final Logger logger =
LoggerFactory.getLogger(CNNAgeDetector.class);

    private static final String[] AGES = new String[]{"0-2", "4-6", "8-13",
"15-20", "25-32", "38-43", "48-53", "60-"};

    private Net ageNet;

    public CNNAgeDetector() {
        try {
            ageNet = new Net();
```

```
            File protobuf = new
File(getClass().getResource("/caffe/deploy_agenet.prototxt").toURI());
            File caffeModel = new
File(getClass().getResource("/caffe/age_net.caffemodel").toURI());

            ageNet = readNetFromCaffe(protobuf.getAbsolutePath(),
caffeModel.getAbsolutePath());
        } catch (URISyntaxException e) {
            logger.error("Unable to load the caffe model", e);
            throw new IllegalStateException("Unable to load the caffe model",
e);
        }
    }

    public String predictAge(Mat face, Frame frame) {
        try {
            Mat resizedMat = new Mat();
            resize(face, resizedMat, new Size(256, 256));
            normalize(resizedMat, resizedMat, 0, Math.pow(2, frame.imageDepth),
NORM_MINMAX, -1, null);

            Mat inputBlob = blobFromImage(resizedMat);
            ageNet.setInput(inputBlob, "data", 1.0, null);      //set the
network input

            Mat prob = ageNet.forward("prob");

            DoublePointer pointer = new DoublePointer(new double[1]);
            Point max = new Point();
            minMaxLoc(prob, null, pointer, null, max, null);
            return AGES[max.x()];
        } catch (Exception e) {
            logger.error("Error when processing gender", e);
        }
        return null;
    }
}
```

GENDER DETECTOR

```
package org.imesha.examples.javacv;
import org.bytedeco.javacpp.indexer.Indexer;
import org.bytedeco.javacv.Frame;
import org.bytedeco.opencv.opencv_core.Mat;
import org.bytedeco.opencv.opencv_core.Size;
import org.bytedeco.opencv.opencv_dnn.Net;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

```
import java.io.File;

import static org.bytedeco.opencv.global.opencv_core.NORM_MINMAX;
import static org.bytedeco.opencv.global.opencv_core.normalize;
import static org.bytedeco.opencv.global.opencv_dnn.blobFromImage;
import static org.bytedeco.opencv.global.opencv_dnn.readNetFromCaffe;
import static org.bytedeco.opencv.global.opencv_imgproc.resize;

public class CNNGenderDetector {

    private static final Logger logger =
LoggerFactory.getLogger(CNNGenderDetector.class);

    private Net genderNet;

    public CNNGenderDetector() {
        try {
            genderNet = new Net();
            File protobuf = new
File(getClass().getResource("/caffe/deploy_gendernet.prototxt").toURI());
            File caffeModel = new
File(getClass().getResource("/caffe/gender_net.caffemodel").toURI());
            genderNet = readNetFromCaffe(protobuf.getAbsolutePath(),
caffeModel.getAbsolutePath());
        } catch (Exception e) {
            logger.error("Error reading prototxt", e);
            throw new IllegalStateException("Unable to start
CNNGenderDetector", e);
        }
    }


    public Gender predictGender(Mat face, Frame frame) {
        try {
            Mat croppedMat = new Mat();
            resize(face, croppedMat, new Size(256, 256));
            normalize(croppedMat, croppedMat, 0, Math.pow(2, frame.imageDepth),
NORM_MINMAX, -1, null);

            Mat inputBlob = blobFromImage(croppedMat);
            genderNet.setInput(inputBlob, "data", 1.0, null);

            Mat prob = genderNet.forward("prob");

            Indexer indexer = prob.createIndexer();
            logger.debug("CNN results {},{}", indexer.getDouble(0, 0),
indexer.getDouble(0, 1));
```

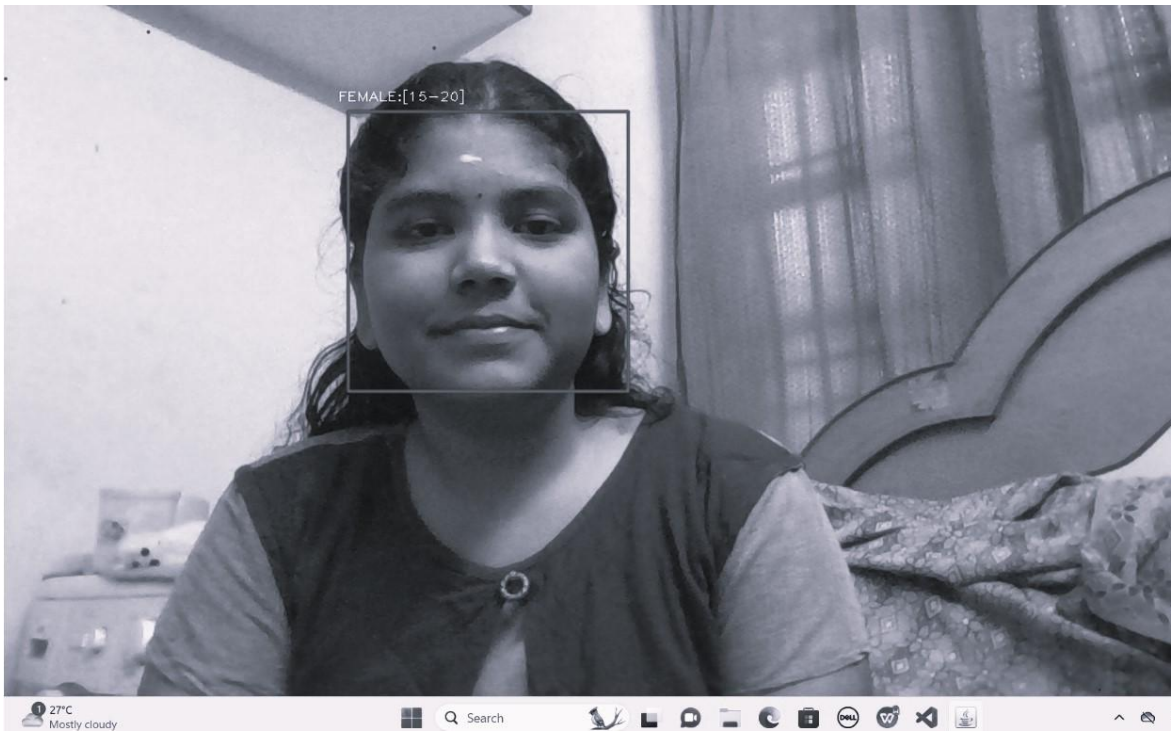
```
            if (indexer.getDouble(0, 0) > indexer.getDouble(0, 1)) {
                logger.debug("Female detected");
                return Gender.FEMALE;
            } else {
                logger.debug("Female detected");
                return Gender.FEMALE;
            }
        } catch (Exception e) {
            logger.error("Error when processing gender", e);
        }
        return Gender.NOT_RECOGNIZED;
    }

    public enum Gender {
        MALE,
        FEMALE,
        NOT_RECOGNIZED
    }
}
}
```

IMAGE UTILS

```
package org.imesha.examples.javacv.util;
import net.coobird.thumbnailator.Thumbnails;
import org.bytedeco.javacv.Frame;
import org.bytedeco.javacv.Java2DFrameConverter;
import org.bytedeco.javacv.OpenCVFrameConverter;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import javax.swing.*;
import java.awt.image.BufferedImage;
import java.io.IOException;

public class ImageUtils {

    private static final Logger logger =
LoggerFactory.getLogger(ImageUtils.class);
    private static Java2DFrameConverter frameConverter = new
Java2DFrameConverter();
    private static OpenCVFrameConverter.ToMat matConverter = new
OpenCVFrameConverter.ToMat();



    public static BufferedImage getResizedBufferedImage(Frame frame, JPanel
videoPanel) {
```

```
        BufferedImage resizedImage = null;

        try {
            resizedImage =
Thumbnails.of(frameConverter.getBufferedImage(frame))
                    .size(videoPanel.getWidth(), videoPanel.getHeight())
                    .asBufferedImage();
        } catch (IOException e) {
            logger.error("Unable to convert the image to a buffered image", e);
        }

        return resizedImage;
    }
    }
```

OUTPUT



VIII.CONCLUSION

In conclusion, the utilization of Convolutional Neural Networks (CNNs) presents a powerful paradigm for face detection, age estimation, and gender prediction. The proposed CNN-based approach signifies a promising avenue, offering superior accuracy and comprehensive insights into facial attributes. This technology holds significant potential for advancing various applications reliant on precise facial analysis.

IX.REFERENCE

- http://www.openu.ac.il/home/hassner/projects/cnn agegender
- https://github.com
- https://geeksforgeeks.com
- https://javatpoint.com