A Project Report

on

AGE AND GENDER CLASSIFICATION

Submitted in partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

by

K. Shreya (16WH1A1235)

Sk. Sumera Sultana (16WH1A1255)

T. Alekhya (16WH1A1257)

Under the esteemed guidance of

Ms. S. Rama Devi

Associate Professor

Department of Information Technology

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Accredited by NBA & NAAC '**A**' Grade, Approved by AICTE, and

Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

April 2020

DECLARATION

We hereby declare that the work presented in this project entitled **"AGE AND GENDER CLASSIFICATION"** submitted towards completion of the major project in IV year of B.Tech IT at "BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN**"**, Hyderabad is an authentic record of our original work carried out under the esteem guidance of Ms. S. Rama Devi, Associate Professor, IT department.

K. Shreya
(16WH1A1235)

Sk. Sumera Sultana
(16WH1A1255)

T. Alekhya
(16WH1A1257)

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(Accredited by NBA & NAAC 'A' Grade, Approved by AICTE, and Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

Department of Information Technology

Certificate

This is to certify that the Project report on "**Age and Gender Classification**" is a bonafide work carried out by **K. Shreya (16WH1A1235), Sk. Sumera Sultana (16WH1A1255), T. Alekhya (16WH1A1257)** in the partial fulfillment for the award of B.Tech degree in **Information Technology, BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Internal Guide
Ms. S. Rama Devi
Associate Professor
Department of IT

Head of the Department
Dr. Aruna Rao S L
Professor and HOD
Department of IT

External Examiner

Acknowledgements

We would like to express our sincere thanks to **Dr. K. V. N. Sunitha, Principal**, BVRIT Hyderabad, for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr**. **Aruna Rao S L, Head**, Dept. of IT, BVRIT Hyderabad for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Ms. S. Rama Devi, Associate Professor, Department of IT,** BVRIT Hyderabad for her constant guidance, encouragement and moral support throughout the project.

Finally, We would also like to thank our Project Co-ordinator, all the faculty and staff of IT Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

K. Shreya
(16WH1A1235)

Sk. Sumera Sultana
(16WH1A1255)

T. Alekhya
(16WH1A1257)

ABSTRACT

Age and gender play fundamental roles in social interactions. Making age and gender estimation from a single face image is an important task in intelligent applications, such as access control, human-computer interaction, law enforcement, marketing intelligence and visual surveillance, etc. Quividi which is an AI software application used to detect age and gender of users who passes by based on online face analyses and automatically starts playing advertisements based on the targeted audience and AgeBot which is an Android application that determines age from your photos using facial recognition. It can guess your age and gender and can also find multiple faces in a picture and estimate the age for each face. Inspired by the above use cases we are going to build a simple Age and Gender classification model. In this project, we will use Convolutional Neural Networks (CNN) and Open CV to accurately identify the age and gender of a person from the image of a face. The predicted gender may be one of 'Male' or 'Female', and the predicted age may be one of the following given ranges - (0 – 5), (6 – 10), (11 – 15), (16 – 20), (21 – 25), (26 – 30), (31 – 35), (36 – 40), (41 – 45), (46 – 50), (51 – 55), (56 - 60), (61 – 100). 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 as a classification problem instead of making it one of regression. The steps we follow are detect the faces, classify into Male/Female, classify into one of the 13 age ranges given, put the results on the image and display it to the user.

LIST OF FIGURES

LIST OF ABBREVATIONS

Term/Abbreviation	Definition
CNN	Convolutional Neural Network
SVM	Support Vector Machine
PCA	Principal Component Analysis
LDA	Linear Discriminant Analysis
DWT	Discrete Wavelet Transform
DCT	Discrete Cosine Transform
RELU	Rectified Linear Unit
RGB	Red Green Blue
FCN	Fully Connected Layer

TABLE OF CONTENTS

1. INTRODUCTION

Age and gender, two of the key facial attributes, play a very foundational role in social interactions. Languages reserve different salutations and grammar rules for men or women, and very often different vocabularies are used when addressing elders compared to young people. Despite the basic roles these attributes play in our day-to-day lives, the ability to automatically estimate them accurately and reliably from face images is still far from meeting the needs of commercial applications. We test our network on the IMDB-WIKI dataset for age and gender classification. Convolutional neural networks (CNN) is a special architecture of artificial neural networks, proposed by Yann LeCun in 1988. CNN is a class of deep, feed-forward (not recurrent) artificial neural networks that are applied to analyzing visual imagery. One of the most popular uses of this architecture is image classification. For example Facebook uses CNN for automatic tagging algorithms, Amazon for generating product recommendations and Google for search through among users photos. The Convolutional Neural Network has shown excellent performance in many computer vision and machine learning problems. CNN is useful in a lot of applications, especially in image related tasks. Applications of CNN include image classification, image semantic segmentation, object detection in images, etc.

1.1 OBJECTIVE

Our main objective is to predict the age and gender for a particular person by considering the images in the dataset. Based on the images we predict whether the person in the image is male or female and the age range of that particular person. Thus, two separate problems will be there, they are face recognition and determining the corresponding age and gender of the person in the image.

1.2 DATASET DESCRIPTION

The dataset used for training and testing for this project is the IMDB-WIKI dataset which is a collection of unfiltered face images. It contains 62328 images to predict the age and gender of the person. There are two possible gender labels that is Male and Female and 13 possible age ranges they are (0 – 5), (6 – 10), (11 – 15), (16 – 20), (21 – 25), (26 – 30), (31 – 35), (36 – 40), (41 – 45), (46 – 50), (51 – 55), (56 - 60), (61 – 100). From the original dataset we used mostly the frontal face images reducing the dataset size to 35538 images.

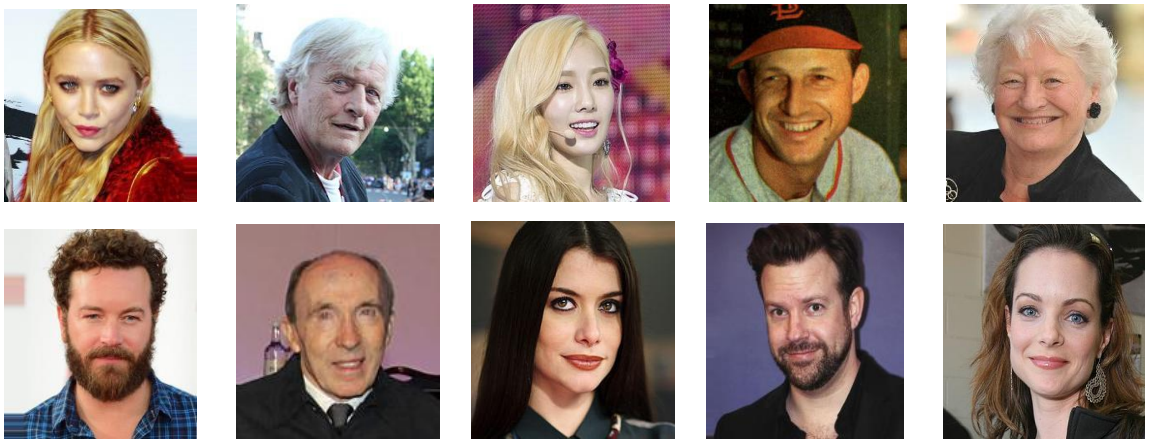


Figure 1.1 : Dataset Description

1.3 PROBLEM IN EXISTING SYSTEM

In the existing system we are using Support Vector Machine(SVM) algorithm which can be used for both classification and regression challenges. Some disadvantages that are associated with SVM are, this algorithm is not suitable for large datasets, SVM does not perform very well when the dataset has more noise that is the target classes are overlapping in cases where number of features for each data point exceeds the number of training data sample, the SVM will under perform, It is only suitable for small datasets.

1.4 ADVANTAGES OF CNN OVER EXISTING SYSTEM

The major advantage of CNN is that it is primarily used for image classification and recognition because of its high accuracy. Also, it is difficult to parallelize SVM, but CNN architecture inherently support parallelization. CNN has higher accuracy, precision and recall rather than other methods like SVM and Random Forest. The main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision and it is also computationally efficient. Here we are using IMDB-WIKI dataset consists of 523,051 images.

2. LITERATURE SURVEY

Face processing has long been recognized as an important module for many computer vision applications. Face recognition, and the classification of the age and gender of face objects are two interesting field of research in this area. With such a face analysis component, it becomes possible to identify a person in order to allow access to private facilities or to display targeted information in advertising based on demographic category of individuals in public places.

2.1 FACE RECOGNITION

Face recognition has received significant attention in recent years. In spite of the progresses in this field, there are still several challenges associated with applying face recognition to real world situations. Among the most important applications in which face recognition plays a key role, we can mention access control, law enforcement and video surveillance. When given an image to process, the face recognition system detects a human subject through face detection techniques and the face region is segmented from the image. The next stage is then to identify the facial features of the face in order to align it in a canonical way.

• **Face Identification:** The system has to identify the unknown face from a pool of candidate faces.

• **Face Verification:** The system must either accept or reject the claimed face as belonging to a specific person. This type of system is used in different applications, most often those related to access control or log in.

There are multiple issues present when working with real-world images. Varying image qualities, background clutter, different face poses and expressions and changes in illumination are among the most common problems encountered in these types of applications. The first attempts made in the history of face recognition were focused on extracting features from an edge image and finding the best match by evaluating distances between captured and trained faces. In recent years, however, a large number of approaches have been proposed. These approaches can be generally divided into two categories: holistic approaches and feature-based approaches.

2.1.1 HOLISTIC APPROACHES

Holistic approaches developed for face recognition are based on processing the whole face and then representing it in a generic form. These methods extract features from a face without considering from which face areas they hail. As the dimension of each image is quite large,

processing the whole face is difficult in practice. Because of this, dimension reduction approaches are applied in conjunction with holistic techniques. Among the large amount of proposed algorithms in holistic approaches we can mention: Principal Component Analysis (PCA) , Linear Discriminant Analysis (LDA) , Discrete Wavelet Transform (DWT), Discrete Cosine Transform (DCT). PCA, as proposed by K. Pearson and H. Hotelling, extracts a set of orthogonal basis vectors from the training data and then represents each data vector by a weighted sum of the extracted basis. This method has also been applied to face images by M. Turk in where PCA was applied as a dimension reduction technique to preserve the most significant basis (called eigenfaces). Following this, each face was represented in the subspace spanned by these bases. They proposed a near-real-time application based on this method and experimented on a set of 2500 face images collected under controlled conditions. LDA, invented by R.A. Fisher, is a method for dimension reduction and data classification which relies on mapping data to a new space within which each class, data have minimum variance and the maximum distance between classes.

This method was applied to face recognition by K. Etemad and R. Chellappa on a set of features extracted by Wavelet Transform and experimented on a database provided by The Olivetti Research Laboratory (ORL) and containing 40 different subjects featuring a variety of changes in illumination, facial expression and facial details (glasses or no glasses) as well as some side movements. DWT is another holistic techniques and is based on the discrete sampling wavelet transform. In its basic format, DWT is used by a 2-D wavelet decomposition, which applies a 1-D transformation in X-direction and another Ydirection. This transformation generates four sub-images, which correspond to the different frequencies in each direction (High-High, High-Low, Low-High, Low-Low). Following this, the feature vector is constructed by using information of the Low-Low sub-image which is then employed for finding the match within trained faces. Discrete Cosine transform is a popular method in image compression and is another method in the category of holistic transform. This method applies a 2-D DCT on image and then codes the entropy of the quantized samples to get the result. DCT is a strong technique for ”energy compaction” and was applied to the process of face recognition, who carried out their experiments on three databases : 1) ORL, 2) The FERET database and 3) The Yale database.

2.1.2 FEATURE BASED APPROACHES

These approaches rely on the fact that all human subjects share the same fundamental structures and the only true difference is the geometric relations between said structures.

Therefore, feature-based methods will, at first, extract facial features like eyes, nose, mouth, eyebrows, etc. and then extract the existing relationship between these facial features. Of the common algorithms employed for this purpose, we can mention deformable templates, the Hough Transform method and Elastic Bunch Graph Matching. The deformable templates method is based on the idea that a family of objects can be represented by deformations of a basic ideal template. This notion is what makes this technique more robust against the different scales and positions that a face object can adopt as well as changes in illumination. This method finds sets of features by using spatial relations and prior knowledge of the general layout of a face to build a vector of features, which is then used to represent and recognize faces. The basic Hough Transform method has been applied to detect straight lines and was later extended by R. Duda and P. Hart so as to be able to compute shape analysis and identify arbitrary shapes.

This approach was applied for the purposes of face recognition in and proved its robustness against various noises. It also has high level of efficiency in terms of its memory usage. One of the other feature-based methods frequently used is Elastic Bunch Graph Matching. This approach builds a graph based on the face structure, that connects the detected facial features, such as eyes, nose, etc. The edges represent the similarity distance between the nodes. Wiskott et al. used Elastic Bunch Graph Matching in face recognition by constructing each node with Gabor filter results and utilizing the edges to describe the geometrical properties of each face. In this section, we presented a short overview of the two main categories in face recognition. Holistic approaches process the entire face and extract a global representation, which is then be fed to the recognition stage system.

Feature-based methods directly identify facial features and employ the spatial structures between them to identify the face. Although processing each facial structure separately creates a system that is robust against position variation, lighting condition changes and noise, in the last few decades, holistic approaches have attracted more attention from researchers specializing in the field of face processing.

2.2 AGE AND GENDER RECOGNITION

Age and gender recognition is an important modules for many computer vision applications such as human-robot interaction, visual surveillance and passive demographic data collections. More recently, the advertising industry’s growing interest in the launching demographic-specific marketing and targeted advertisements in public places has attracted the attention of more and more researchers specialized in the field of computer vision.

Among the early algorithms in the field of age and gender recognition, Cottrel and Metcalfe extracted whole-face features called Holons, which were fed into a back propagation network model to classify males and females. Golomb et al. proposed ”SEXNET” Neural Network model for gender recognition. This network compresses faces using faces’ raw pixels and then estimates their sex in subsequent layers of their proposed network. In 1995, Brunelli and Poggio achieved a 79% recognition rate for gender by using the HyperBF network on a set of geometrical features extracted from faces and, shortly after, Abdi et al. employed the RBF network and achieved a 90% recognition rate on data preprocessed by PCA.

In 1997, wavelet components (jets) were exploited by Wiskott et al. in order to describe face features and build Elastic Bunch Graph models. In 1999, Kwon and Lobo proposed a method for age classification that first extracted specific features of the face elements such as eyes, noses, mouths and chins. It then compute the ratios estimated between the top of sides of the head before, finally, processing skin wrinkle information in order to classify people in three classes: babies, young adults and seniors. Wen Bing Horng et al employed Sobel edge detector with a back-propagation Neural network to classify human face subjects into four classes: babies, young adults, middle adults and old adults. Lyons et al. applied Gabor wavelets and LDA to create a neuro-fuzzy system for gender classification, which gave them a more accurate system when compared with previously existing methods.

In 2002 Sun et al. proposed a feature selection method by using genetic algorithms to select features extracted by PCA. They compared different classifiers such as Bayesian, NN, LDS and SVM and demonstrated that using a SVM classifier is a better approach for classifying gender. Moghaddam and Yang claimed that the support vector machine (SVM), when using a RBF kernel, generates a stronger classifier than those previously used in gender recognition. They experimented on both small images (21×12) and good quality images (64×72) of the FERET database and they achieved a 96.6% accuracy on the second image data. In addition to this, they claimed that the difference between the two different qualities is just 1%. Following this, Lanitis et al. claimed that combination of shape and texture features with different techniques such as an active appearance model, PCA , Mahalanobis distance, a Multi-Layer perceptron and a Neural network can be used as a good age estimator.

In 2004, Jain and Huang proposed an approach using an independent component analysis (ICA) as one of the feature-based methods for extracting features and employed LDA as classifier. Costen et al. proposed a sparse SVM to classify genders and claimed an accuracy rate of 94.42% on Japanese face images. In 2005, Ye Sun et al. proposed an approach that used the relationship between key feature points in human faces to train an Embedded Hidden

Markov Model(EHMM) to estimate age. PCA and Locality Sensitive Discriminant Analysis (LSDA) was later applied to learn different manifolds of aging patterns based on the fact that each age category was distributed on a specific manifold in a large dimensional subspace. Local Binary Pattern (LBP) was used as a method for extracting texture features by Sun et al. in 2006. LBP, which builds up the spatial structure of each pixel by comparing its intensity with that of its neighborhood, was used with both Adaboost classifier and SVM classifier to facilitate gender classification. Although Saatci and Town investigated the interdependency of gender recognition upon expression and they showed that the gender classification rate decreased even with using separate gender data for different expression 12 classes. In 2007, a color descriptor relying on the construction of histograms with 4 bins per color channel in the RGB color space was proposed for gender classification. The fusion approach, 2D PCA and the centralized Gabor gradient histogram (CGGH) were other methods that were employed in feature extraction. Xiaofeng Fu combined CBP and Gabor gradient magnitudes to extract more discriminative features at multiple scales and orientation. By feeding these features into a nearest center-based neighbor classifier, they achieved a 96.56% accuracy rate on FERET and a 95.25% accuracy rate on CAS-PEAL.

Meanwhile, in 2010, Guo-Shiang Lin and Yi-Jie Zhao proposed a color-based approach on SVM for gender classification. In 2011, compressive sensing framework was applied by Duan-Yu Chen and Po-Chiang Hsieh to represent face images in a sparse frequency domain. They extracted two feature sets for each gender; the first set presenting features that are common between all the images in corresponding gender classes and the other one representing each individual face in that class. Ihsan Ullah et al. used spatial WLD as a texture descriptor for gender recognition. They divided the image into a number of blocks, calculated the WLD descriptor for each block and concatenated them.

In 2013, Chen, Y. et al. introduced a new method based on subspace learning that operates as a set of constrained optimization problems to characterize age-related features. By employing semi-supervised learning techniques they applied the Support Vector Regression(SVR) methods onto the features to create an age estimators model. Juan E. Tapia and Claudio A. Perez proposed a method, which uses feature selection based on mutual information and the fusion of intensity, shape and texture features to classify gender classes. They then calculated LBP features with different radial and spatial scales, and then selected features using mutual information. They tested three different techniques to measure mutual information: minimum redundancy and maximum relevance, normalized mutual information, and conditional mutual information-based. Therefore, we can summarize that the most of the efforts made to

optimize in gender and age recognition from a face object attempt to best represent the face object. While some methods choose to use raw pixels without any modification, the majority of the existing methods use local visual descriptors to produce stronger and, often, more compact representations of face images. Examples of visual information commonly used for age and/or gender recognition are texture information. In these methods, local descriptors are extracted from a dense regular grid placed over the entire image and the face representation is built by concatenating these extracted descriptors into a single vector. A key issue in this framework is to determine the optimal grid parameters (e.g., spacing, size, number of grids in multi-resolution/pyramid approaches, etc.).

Dago-Casas et al. proposed to use raw pixels, Gabor jets and LBPs on Gallaghers database for gender recognition. They reduced the size of extracted features by using Principle Component Analysis and showed that, by using Gabor jets followed by SVM high gender classification, a greater accuracy is obtained. While the aforementioned methods for age and gender recognition used fixed settings and performed trial-and-error to determine the right grid parameters, a better approach consists in using feature selection to allow only the most informative image regions (or grid cells) to contribute to the face representation, i.e., those that can best separate face images that belong to different demographic classes. This approach further facilitates the integration of different types of descriptors (e.g., color based, shape based, texture based, etc.) and allow for more compact representations by preventing redundant features from contributing to the face representation.

This has described different attempts that have been made in the fields of face, gender and age recognition. The main issue addressed by these methods is how to best represent a face in such a way that it will be most efficiently identified.

3. REQUIREMENT SPECIFICATIONS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements, minimum and recommended.

3.1.1 SOFTWARE REQUIREMENT

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software project.

- Operating system : Windows XP/10
- Coding Language : Python 3
- IDE : Jupyter Notebook
- Cloud Service : Google Colaboratory

3.1.2 HARDWARE REQUIREMENT

The hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

- Operating System - Windows 8 or higher version is required
- Hard Driver - Minimum 32Gb , Recommended 64Gb or more
- Processor - intel core i5 CPU and 2.50 GHz processor is needed
- Physical Memory - Minimum 8Gb RAM is required

3.2 LIBRARIES

- **Keras**

  Keras is an open-source neural-network library written in Python. It allows for easy and fast prototyping. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It runs seamlessly on Central Processing Unit (CPU) and Graphics Processing Unit (GPU).

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. is an open-source neural-network library written in Python. It allows for easy and fast prototyping. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It runs seamlessly on Central Processing Unit (CPU) and Graphics Processing Unit (GPU). Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

- **Keras.applications**

  Keras Applications are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning. Weights are downloaded automatically when instantiating a model.

- **Keras.models**

  There are two main types of models available in Keras: the Sequential model, and the Model class used with the functional Application programming interface (API). These models have a number of methods and attributes in common:

  - model.layers() is a flattened list of the layers comprising the model.
  - model.inputs() is the list of input tensors of the model.
  - model.outputs() is the list of output tensors of the model.
  - model.summary() prints a summary representation of your model.

- **Keras.optimizers**

  An optimizer is one of the two arguments required for compiling a Keras model. You can either instantiate an optimizer before passing it to model.compile() or you can call it by its name. In the latter case, the default parameters for the optimizer will be used.

- **TensorFlow**

  TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. Its flexible architecture

allows for the easy deployment of computation across a variety of platforms and from desktops to clusters of servers to mobile and edge devices.

- **NumPy**

  NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

- **Pandas**

  Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

- **Matplotlib**

  Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose Graphical User Interface (GUI) toolkits.

- **Scikit-learn**

  Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

- **Python Imaging Library (PIL)**

  It is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.

4. DESIGN OF THE SYSTEM

Using a fully connected neural network to make an image classification requires a large number of layers and neurons in the network, which increases the number of parameters leading the network to over-fitting (memorizing the training data only). The input image may also lose its pixels correlation properties since all neurons (carrying pixels values) are connected to each other. Convolutional neural networks have emerged to solve these problems through their kernel filters to extract main features of the input image and then inject them into a fully connected network to define the class.

The chosen architecture in our application is convolutional neural network. It contains 6 layers of convolution and simplification functions made by 3x3 kernel filters, Batch Normalization and a max pooling filter of 3x3 to reduce at last the input image of 32x32. The feature images carry most important features to define a specified age and gender classes by processing them into fully connected network.

In deep learning, a CNN is a class of deep neural network most commonly applied to analyzing visual imagery. The name “convolutional neural network” indicates that the network employs a mathematical operation called convolution. Convolution is a specialized kind of linear operation. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers that convolve with a multiplication or other dot product.

The activation function is commonly a RELU layer, and is subsequently followed by additional convolutions such as pooling layers, fully connected layers and normalization layers, referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

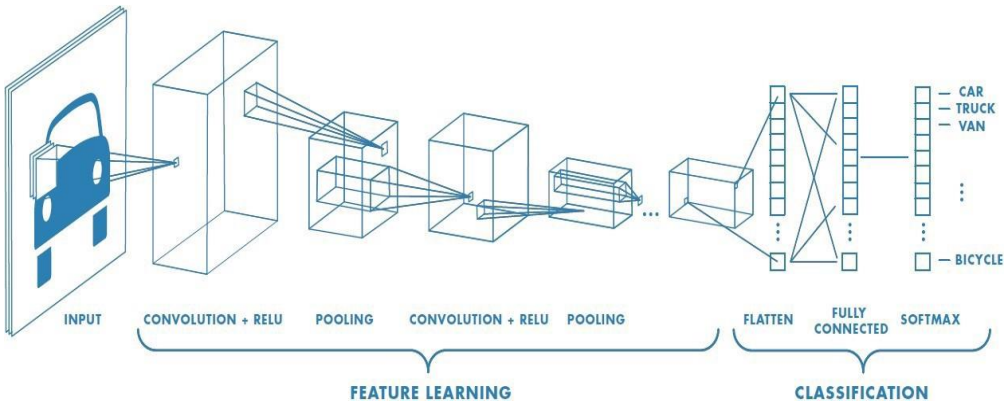


Figure 4.1 : CNN Architecture

A CNN typically has three layers: a convolutional layer, pooling layer, and fully connected layer.

CONVOLUTIONAL LAYER

The convolution layer is the core building block of CNN. It carries the main portion of the network's computational load. The main objective of convolution is to extract features such as edges, colors, corners from the input. As we go deeper inside the network, the network starts identifying more complex features such as shapes, digits, face parts as well. This layer performs a dot product between two matrices, where one matrix(known as filter/kernel)is the set of learnable parameters, and the other matrix is the restricted portion of the image. If the image is RGB then the filter will have smaller height and width compared to the image but it will have the same depth(height x width x 3) as of the image. At the end of the convolution process, we have a featured matrix which has lesser parameters(dimensions) than the actual image as well as more clear features than the actual one. The learnable parameters of each layer consist of filters, extended through the full depth of the input volume but these have a small receptive field. When an input is subjected to forward pass, each kernel is convolved across the height and width of the input volume, creating a 2-D activation map of that filter. If 'N' filters are used, then stacking those 'N' activation maps along the depth forms the full output of the convolutional layer. The activation layer is very useful as it helps to approximate almost any nonlinear function. The feature map from the convolutional layer is taken as input to the activation layer.

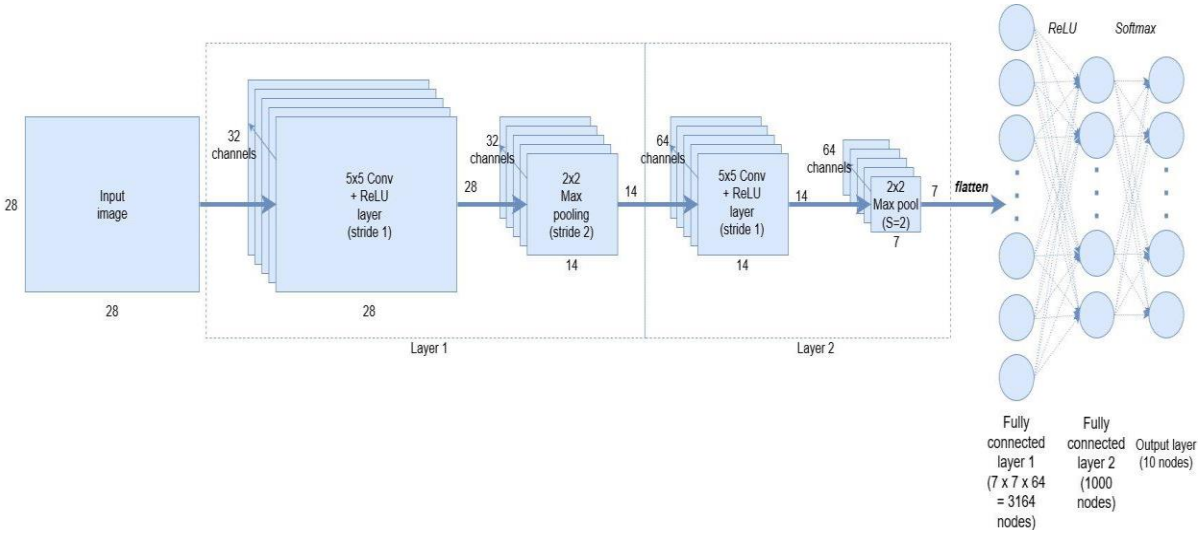


Figure 4.2 : Filter Operation over Convolutional Layers

POOLING LAYER

Pooling layers are used to reduce the spatial size of representation generated by previous kernels after convolution. This layer is solely to decrease the computational power required to process the data. It is done by decreasing the dimensions of the featured matrix even more. In this layer, we try to extract the dominant features from a restricted amount of neighborhood. There are two types of pooling techniques: AVERAGE-pooling and MAX-pooling. The difference between these two is, in AVERAGE-pooling layer, we take the average of all the values of pooling region and in MAX-pooling layer, it separates input into squares of a given size, and outputs the maximum value of each square lying inside the pooling region. So, after pooling layer, we have a matrix containing main features of the image and this matrix has even lesser dimensions. These layers are used to extract dominant features that are positional and rotational invariant. It is common practice to include a pooling layer in between two convolutional layers. The most common pooling layer is the max pooling layer but both the methods reduce the dimensionality and computation efforts.

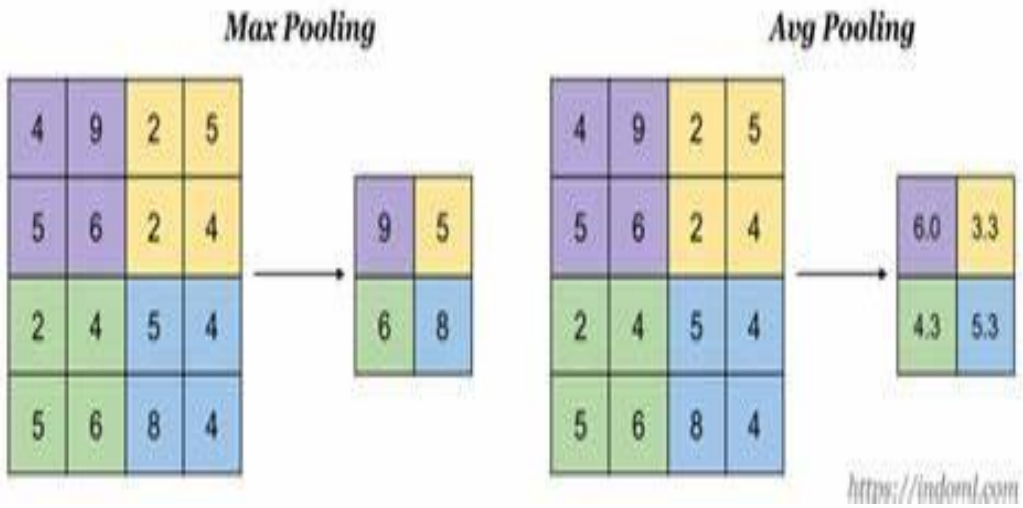


Figure 4.3 : Max Pooling and Average Pooling

FULLY CONNECTED LAYER

After the pooling layer, we find the fully connected layers (FCN) that are used to make final predictions. A FCN layer obtains resources in a vector form from a previous resource extraction layer, multiplies a weight matrix, and generates a new resource vector whose computation pattern is a dense matrix vector multiplication. Some FCNs are used in a cascade mode that ultimately produce the CNN classification result that generates a probability (a number ranging from 0 to 1) for each of the classification labels that the model is trying to predict. Now, the current output is

flattened, converted to one long vector, which will be crucial for the classification algorithms. Flatting is applied for converting the data to a more manageable version within the classification algorithm, artificial neural network, as it intakes the flattened data as input. After this point, the network obtained the feature map of the input value, which will proceed with a regular feed forward back propagation neural network.

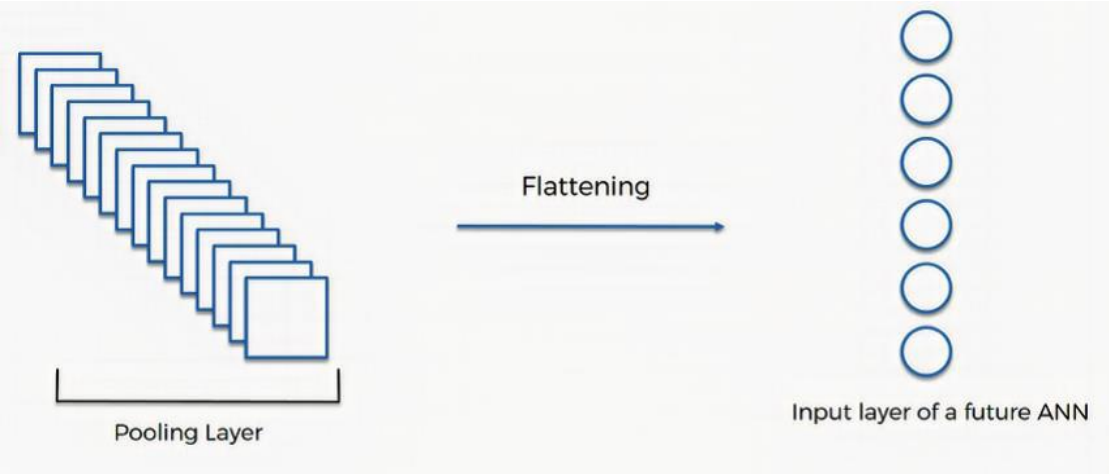


Figure 4.4 : Flattening after the Convolutional Steps

Now, the current output is flattened, converted to one long vector, which will be crucial for the classification algorithms. Flatting is applied for converting the data to a more manageable version within the classification algorithm, artificial neural network, as it intakes the flattened data as input. After this point, the network obtained the feature map of the input value, which will proceed with a regular feed forward back propagation neural network.

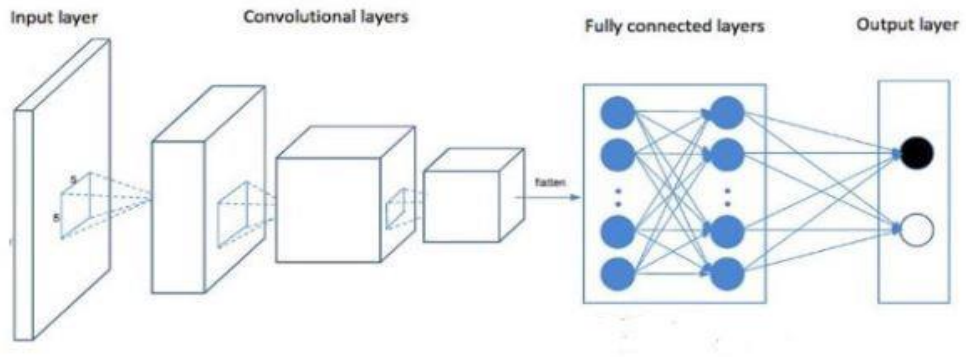


Figure 4.5 : Fully connected layer inside CNN

5. MODULES

5.1 FACE DETECTION USING OpenCV

Face detection is a problem in computer vision of locating and localizing one or more faces in a photograph. Feature-based face detection algorithms are fast and effective. OpenCV provides the CascadeClassifier class that can be used to create a cascade classifier for face detection. This classifier is loaded as follows

```
classifier=CascadeClassifier('haarcascade_frontalface_default.xml')
```

Once loaded, the model can be used to perform face detection on an image by calling the detectMultiScale() function. This function will return a list of bounding boxes for all faces detected in the photograph. The image can be loaded by using imread() function which is present in OpenCv. The path of the image is given as input to the imread() function. This is given as

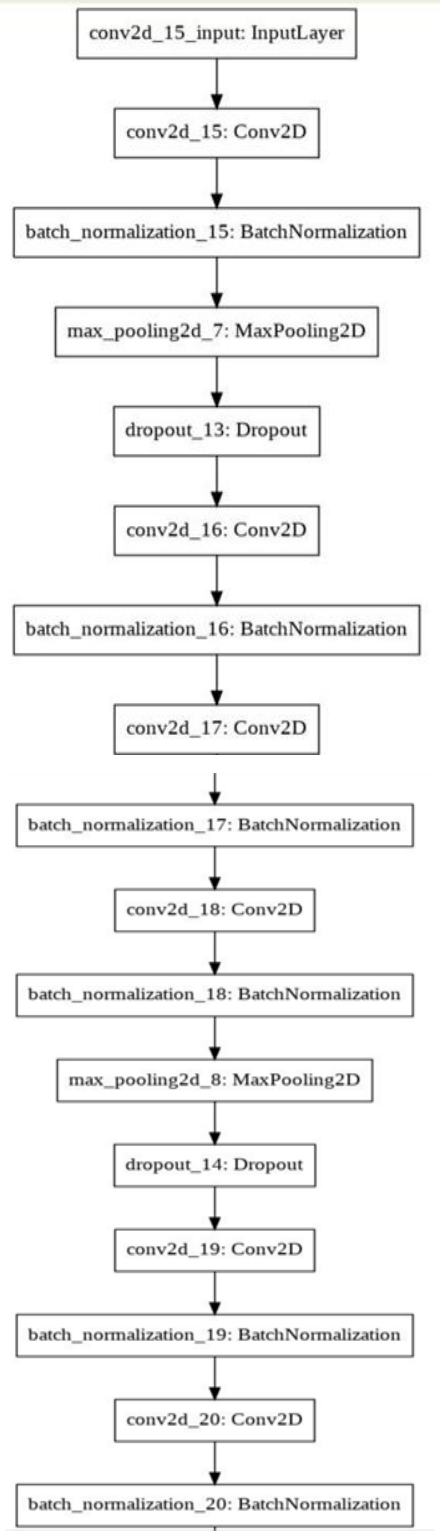
```
path="/content/wiki_crop/05/10024005_1955-12-24_2012.jpg"
image = cv2.imread(path)
```

This first loads the photograph, then loads and configures the cascade classifier; faces are detected and each bounding box is printed. Each box lists the *x* and *y* coordinates for the bottom-left-hand-corner of the bounding box.

5.2 AGE AND GENDER MODEL

Gathering a large, labelled image training set for age and gender estimation from social image repositories requires either access to personal information on the subjects appearing in the images (their birth date and gender), which is often private, or is tedious and time-consuming to manually label. Data-sets for age and gender estimation from real-world social images are therefore relatively limited in size and presently no match in size with the much larger image classification data-sets. Overfitting is common problem when machine learning based methods are used on such small image collections. This problem is exacerbated when considering deep convolutional neural networks due to their huge numbers of model parameters. Care must therefore be taken in order to avoid overfitting under such circumstances. We trained our model by taking all images of persons in all age groups from

children to the old aged. The following diagram will show the flow of our project. At last the final result that is the age and gender will be shown on image.



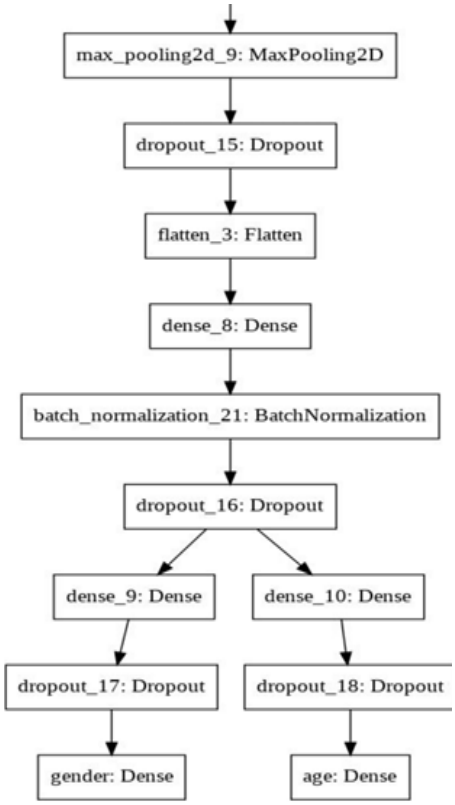


Figure 5.1 : Design for Age and Gender Model

First the input Layer takes the input as an image with a particular size given. Computer sees the image as an array of pixels. For example, if image size is 300 x 300. In this case, the size of the array will be 300x300x3. Where 300 is width, next 300 is height and 3 is RGB channel values. The computer is assigned a value from 0 to 255 to each of these numbers. This value describes the intensity of the pixel at each point.

To solve this problem the computer looks for the characteristics of the base level. For the computer, these characteristics are boundaries or curvatures. And then through the groups of convolutional layers the computer constructs more abstract concepts that is the image is passed through a series of convolutional, nonlinear, pooling layers and fully connected layers, and then generates the required output.

The Convolution layer is always the first layer. The image (matrix with pixel values) is entered into it. Imagine that the reading of the input matrix begins at the top left of image. Next the software selects a smaller matrix there, which is called a filter (or neuron, or core). Then the filter produces convolution, i.e. moves along the input image. The filter's task is to multiply its values by the original pixel values. All these multiplications are summed up. One number is obtained in the end. Since the filter has read the image only in the upper left corner,

it moves further and further right by 1 unit performing a similar operation. After passing the filter across all positions, a matrix is obtained, but smaller then a input matrix.

This operation, from a human perspective, is analogous to identifying boundaries and simple colors on the image. The network will consist of several convolutional networks mixed with nonlinear and pooling layers. When the image passes through one convolution layer, the output of the first layer becomes the input for the second layer. And this happens with every further convolutional layer.

The nonlinear layer is added after each convolution operation. It has an activation function, which brings nonlinear property. Without this property a network would not be sufficiently intense and will not be able to model the response variable (as a class label). This non linear layer squashes all the negative values and gives only the positive values.

The pooling layer follows the nonlinear layer. It works with width and height of the image and performs a down sampling operation on them. As a result the image volume is reduced. This means that if some features (as for example boundaries) have already been identified in the previous convolution operation, than a detailed image is no longer needed for further processing, and it is compressed to less detailed pictures.

After completion of series of convolutional, nonlinear and pooling layers, it is necessary to attach a fully connected layer. This layer takes the output information from convolutional networks. Attaching a fully connected layer to the end of the network results in an N dimensional vector, where N is the amount of classes from which the model selects the desired class. At last the output will be shown by drawing the bounding box and printing the age and gender of the person in the image.

6. IMPLEMENTATION

Source Code

```
#Downloading dataset into Google Collab

!wget https://data.vision.ee.ethz.ch/cvl/rrothe/imdb-wiki/static/wiki_crop.tar

#Extracting the dataset

!tar -xf wiki_crop.tar

import os

path = "/content/wiki_crop/00/"

files = os.listdir(path)

size = len(files)

print("Total samples:",size)

#CNN model construction

import keras

from keras.models import Sequential,Model,Input

from keras.layers import Conv2D, MaxPooling2D, BatchNormalization, Dropout, Dense,

Flatten

inputShape=(64,64,3)

model= Sequential()

model.add(Conv2D(32,(3,3),padding="same",input_shape=inputShape,activation='relu'))

model.add(BatchNormalization(axis=-1))
```

```
model.add(MaxPooling2D(pool_size=(3,3)))

model.add(Dropout(0.25))

model.add(Conv2D(32,(1,1)))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(64,(3,3),padding="same",activation='relu'))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(64,(1,1)))

model.add(BatchNormalization(axis=-1))

model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Dropout(0.25))

model.add(Conv2D(128,(3,3),padding="same",activation='relu'))

model.add(BatchNormalization(axis=-1))

model.add(Conv2D(128,(1,1)))

model.add(BatchNormalization(axis=-1))

model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(1024,activation='relu'))

model.add(BatchNormalization(axis=-1))

model.add(Dropout(0.25))
```

```
x = model.get_output_at(-1)

y1 = Dense(128,activation='relu')(x)

y1 = Dropout(0.25)(y1)

y2 = Dense(128,activation='relu')(x)

y2 = Dropout(0.25)(y2)

y1 = Dense(2,activation='softmax',name='gender')(y1)

y2 = Dense(13,activation='softmax',name='age')(y2)

mod = Model(inputs = model.get_input_at(0), outputs = [y1,y2] )

mod.save("2-class-base-dropouts.h5")

#Preprocessing the dataset

import cv2

import numpy as np

import os

import glob

path='/content/wiki_crop' #path where dataset

def doit(image_path):

    image = cv2.imread(image_path)

    face_cascade =

    cv2.CascadeClassifier(cv2.data.haarcascades+'haarcascade_frontalface_default.xml')

    try:
```

```
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        gray=np.array(gray,dtype='uint8')
        faces = face_cascade.detectMultiScale(
            gray,
            scaleFactor= 1.1,
            minNeighbors= 5,
            minSize=(10, 10)
        )
        for (x, y, w, h) in faces:
            face_crop = np.copy(image[y:y+h,x:x+w])
            face_crop = cv2.resize(face_crop,(64,64))
            if face_crop.any():
                cv2.imwrite(image_path,face_crop)
                return 1
    except Exception as e:
        print(e)
        return -1
    return 0
written=0
unable_to_read=0
```

```
no_faces_found=0
for i in glob.glob(path+'/*/*'):
    k=doit(i)
    if k==1:
      if written<10:
        print('Image is written '+str(i))
      written+=1
    elif k==-1:
      os.remove(i)
      if unable_to_read<10:
        print('Unable to read '+str(i))
      unable_to_read+=1
    elif k==0:
      os.remove(i)
      if no_faces_found<10:
        print('No face found '+str(i))
      no_faces_found+=1
    if ((written+unable_to_read+no_faces_found) % 500 ==0) and
    ((written+unable_to_read+no_faces_found)!=0):
        print(str(written+unable_to_read+no_faces_found)+' Faces Read')
```

```
    if (written % 500 ==0) and (written!=0):
        print(str(written)+' Faces done')
    if (unable_to_read % 500 ==0) and (unable_to_read!=0):
        print(str(unable_to_read)+' Unable to read')
    if (no_faces_found % 500 ==0) and (no_faces_found!=0):
        print(str(no_faces_found)+' no_faces_found')
print(str(written+unable_to_read+no_faces_found)+" Total pics")
print(str(written)+' Images cropped')
print(str(unable_to_read)+' unable to read')
print(str(no_faces_found)+' Faces not found')
#Training the data
import numpy as np
import cv2
import pandas as pd
import zipfile
path_to_zip_file='/content/Manual DataSet.zip'
path='/content/Manual Dataset'
with zipfile.ZipFile(path_to_zip_file, 'r') as zip_ref:
   zip_ref.extractall(path)
import glob
```

```
a = glob.glob(path+'/*/*/*')

print(len(a))

int_to_gen = {0: 'female',1: 'male'}

gen_to_int = {'Female':0,'Male':1}

int_to_age = {    0: '(0-5)',

  1: '(6-10)',

  2: '(11-15)',

  3:'(16-20)',

  4:'(21-25)',

  5:'(26-30)',

  6:'(31-35)',

  7:'(36-40)',

  8:'(41-45)',

  9:'(46-50)',

  10:'(51-55)',

  11:'(56-60)',

  12:'(61-100)'}

age_to_int = {    '1-5': 0,

   '6-10': 1,

   '11-15': 2,
```

```
    '16-20': 3,
    '21-25': 4,
    '26-30': 5,
    '31-35': 6,
    '36-40': 7,
    '41-45': 8,
    '46-50': 9,
    '51-55': 10,
    '56-60': 11,
    '61-100': 12}
gen=[]
age=[]
for da in a:
  gen.append(gen_to_int[da.split('/')[3]])
  age.append(age_to_int[da.split('/')[4].split(' ')[0]])
from keras.utils import to_categorical
age=to_categorical(age,dtype='int')
gen=to_categorical(gen,dtype='int')
df = pd.DataFrame({'path':a,'gender':gen.tolist(),'age':age.tolist()})
print(df.head(652))
```

```
from keras.preprocessing.image import ImageDataGenerator

datagen=ImageDataGenerator(rescale=1./255.,validation_split=0.30)

train_generator=datagen.flow_from_dataframe(

dataframe=df,

directory=None,

x_col='path',

y_col=['gender','age'],

subset="training",

batch_size=32,

seed=42,

shuffle=True,

class_mode="multi_output",

target_size=(64,64))

validation_generator=datagen.flow_from_dataframe(

dataframe=df,

directory=None,

x_col='path',

y_col=['gender','age'],

subset="validation",

batch_size=32,
```

```
seed=42,

shuffle=True,

class_mode="multi_output",

target_size=(64,64))

from keras.models import load_model

model = load_model('/content/2-class-base-dropouts.h5')

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

history=model.fit_generator(generator=validation_generator,steps_per_epoch=validation_ge
nerator.samples//validation_generator.batch_size,epochs=30,validation_data=train_generator
,validation_steps=train_generator.samples//train_generator.batch_size)

model.save('trainedmodel.h5')

import matplotlib.pyplot as plt

print(history.history.keys())

# summarize history for Gender accuracy

plt.plot(history.history['gender_acc'])

plt.title('Gender accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.show()

# summarize history for Age accuracy

plt.plot(history.history['age_acc'])
```

```
plt.title('Age accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.show()

#Testing on the IMDB – WIKI dataset

import keras

path = "/content/wiki_crop/05/10024005_1955-12-24_2012.jpg" #path to image

import cv2,numpy as np

from PIL import Image

from keras.preprocessing.image import img_to_array

from google.colab.patches import cv2_imshow

from keras.models import load_model

a = load_model("/content/trainedmodel.h5")

int_to_gen = {0: 'female',1: 'male'}

int_to_age = {    0: '(0-5)',

    1: '(6-10)',

    2: '(11-15)',

    3:'(16-20)',

    4:'(21-25)',

    5:'(26-30)',
```

```
    6:'(31-35)',
    7:'(36-40)',
    8:'(41-45)',
    9:'(46-50)',
    10:'(51-55)',
    11:'(56-60)',
    12:'(61-100)'}
image = cv2.imread(path)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
facer = cv2.CascadeClassifier(cv2.data.haarcascades+'haarcascade_frontalface_default.xml')
faces = facer.detectMultiScale(
  gray,
  scaleFactor= 1.1,
  minNeighbors= 5,
  minSize=(10, 10)
)
for (x, y, w, h) in faces:
    cv2.rectangle(image,(x,y),(x+w,y+h),(0,255,0),2)
    face_crop = np.copy(image[y:y+h,x:x+w])
    face_crop = cv2.resize(face_crop,(64,64))
```

```
    if face_crop.any():
        i=cv2.cvtColor(face_crop, cv2.COLOR_BGR2RGB)
        i = Image.fromarray(i)
        i = i.resize((64,64))
        i = img_to_array(i)
        i=i/255.
        ans=a.predict(np.expand_dims(i,0))
        gender=int_to_gen[ans[0].argmax()]
        age=int_to_age[ans[1].argmax()]
        cv2.putText(image, gender+' '+age, (x,y-5), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                    (0,255,0), 2)
try:
    if not faces:
            i=cv2.cvtColor(face_crop, cv2.COLOR_BGR2RGB)
            i = Image.fromarray(i)
            i = i.resize((64,64))
            i = img_to_array(i)
            i=i/255.
            ans=a.predict(np.expand_dims(i,0))
            gender=int_to_gen[ans[0].argmax()]
```

```
        age=int_to_age[ans[1].argmax()]
        cv2.putText(image, gender+' '+age, (x,y-5), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
                    (0,255,0), 2)
except:
   pass
cv2_imshow(image)
cv2.imwrite('result.png',image)
print(gender)
print(age)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

7. TESTING

To evaluate and validate the effectiveness of the proposed approach, we conducted the experiments respectively. Different results were obtained, but this project reports only the most valid.


male (46-50)


female (61-100)

8. CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

This system provides approach for classifying the images into respective age and gender classes with quality output. It includes efficient feature extraction methods which results in appropriate outcomes. The CNN model is the finest techniques of Deep Learning which ensures accuracy in the achieved output. We have presented our method for age and gender classification based on deep Convolutional Neural Networks. We measured our performance on the IMDB-WIKI dataset. The work includes processing of RGB color images, in which RGB images gives more accuracy i.e. gender accuracy is 100% and age accuracy is 91.75%.

We provide results with a CNN architecture designed to avoid overfitting. The overfitting of the architecture is removed by dropout layers. We further inflate the size of the training data by artificially adding cropped versions of the images in our training set. Two important conclusions can be made from our results. First, CNN model is efficient for image classification. Second, increase in the size of dataset will be capable of substantially improving results. Correct age and gender detection is essential for accurate classification because if it is not detected properly it cannot produce proper result.

8.2 FUTURE SCOPE

In future, we will try to detect emotions of the persons along with the age and gender. We will try to include different emotions into account. The overall performance could also be improved and customized with the help of increase in the size of the datasets and also from different datasets.

9. REFERENCES

[1]A Review on Age and Gender Classification
https://ieeexplore.ieee.org/document/7301352

[2]Convolution Neural Networks
https://medium.com/@ksusorokina/image-classification-with-convolutional-neural-networks-496815db12a8

[3]Detection of faces from various images
https://www.digitalocean.com/community/tutorials/how-to-detect-and-extract-faces-from-an-image-with-opencv-and-python

[4]Face Detection
https://medium.com/analytics-vidhya/how-to-build-a-face-detection-model-in-python-8dc9cecadfe9

[5]https://docs.opencv.org/2.4/modules/contrib/doc/facerec/tutorial/facerec_gender_classification.html

[6]https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

[7]https://www.pyimagesearch.com/2018/02/26/face-detection-with-opencv-and-deep-learning/