AGE AND GENDER DETECTOR USING DEEP LEARNING

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering

by

Vanshika Dravid(38110616)
Shivangi Ashim Sen(38110685)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH - 2022

DECLARATION

We Vanshika Dravid & Shivangi Ashim Sen hereby declare that the Project Report entitled **Age and Gender Detector using Deep Learning** done by me under the guidance of Dr. L. Lakshamanan(M.E, Ph.d) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Sathyabama Institute of Science and Technology.

DATE: 1st March

Signature:

PLACE: SIST, CHENNAI

Vanshika Dravid

Shivangi Ashim Sen

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Vanshika Dravid (38110616) and Shivangi Ashim Sen (38110685) who carried out the project entitled " AGE AND GENDER DETECTOR USING DEEP LEARNING" under my supervision from October 2021 to May 2022.

INTERNAL GUIDE
DR. L. LAKSHMANAN(M.E, Ph.d)

HEAD OF DEPARTMENT
DR. L. LAKSHMANAN(M.E.,Ph.d)

Submitted for Viva voce Examination held on________________________________

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D**, **Dean**, School of Computing **Dr. L. Lakshmanan M.E., Ph.D. ,** and **Dr.S.Vigneshwari M.E., Ph.D. Heads** of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Dr. L. Lakshmanan for his valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project

Abstract

Since the advent of social media, there has been an increased interest in automatic age and gender classification through facial images. So, the process of age and gender classification is a crucial stage for many applications such as face verification, aging analysis, ad targeting and targeting of interest groups. Yet most age and gender classification systems still have some problems in real-world applications. This work involves an approach to age and gender classification using multiple convolutional neural networks (CNN). The proposed method has 5 phases as follows: face detection, remove background, face alignment, multiple CNN and voting systems. The multiple CNN model consists of three different CNN in structure and depth; the goal of this difference It is to extract various features for each network. Each network is trained separately on the AGFW dataset, and then we use the Voting system to combine predictions to get the result.

CHAPTER 1

INTRODUCTION

Biometrics, is the science of analyzing the physical or behavioral characteristics of each individual that enable the authentication of their identity in a reliable manner, it offers significant advantages conventional identification methods, such as passwords and cards, are not transferable, exclusive to each person and are not lost or stolen, particularly because of biometric features. The range of biometric solutions
relies on user approval, security, cost and time for implementation...etc. Recently, face recognition has been one of the most interesting tasks in pattern recognition, many applications use this technique because the human face is considered a very rich source of information. In particular, gender and age are facial features that can be very useful for a multitude of applications, for example an automatic gender and age prediction system is used to profile customers who are interested for a product or for target advertising. The areas of age and gender classification have been studied for decades. Until detailing the methods used in this article, we will first provide a summary of the facial recognition experiments carried out by scholars, which can be grouped into tree classes of interest. Over the last decade, the rate of image uploads to the Internet has grown at a nearly exponential rate. This newfound wealth of data has empowered computer scientists to tackle problems in computer vision that were previously either irrelevant or intractable. Consequently, we have witnessed the dawn of highly accurate and effificient facial detection frameworks that leverage convolutional neural networks under the hood. One of the most critical barriers that face any system to ageestimation or age-classification is the absence of a consistent pattern of facial aging. This is due to the nature of human faces, and the stages of aging may differ from one human to another.

CHAPTER 2
LITERATURE SURVEY

2.1 Age and Gender Classification using Multiple Convolutional Neural Network

Abstract

Since the advent of social media, there has been an increased interest in automatic age and gender classification through facial images. So, the process of age and gender classification is a crucial stage for many applications such as face verification, aging analysis, ad targeting and targeting of interest groups. Yet most age and gender classification systems still have some problems in real-world applications. This work involves an approach to age and gender classification using multiple convolutional neural networks (CNN). The proposed method has 5 phases as follows: face detection, remove background, face alignment, multiple CNN and voting systems. The multiple CNN model consists of three different CNN in structure and depth; the goal of this difference It is to extract various features for each network. Each network is trained separately on the AGFW dataset, and then we use the Voting system to combine predictions to get the result.

Introduction

Age and gender play fundamental roles 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. This is particularly perplexing when considering recent claims to super-human capabilities in the related task of face recogni
tion (e.g., [48]).

Past approaches to estimating or classifying these attributes from face images have relied on differences in facial feature dimensions [29] or “tailored” face descriptors
(e.g., [10, 15, 32]). Most have employed classifification schemes designed particularly for age or gender estimation tasks, including [4] and others. Few of these past methods were designed to handle the many challenges of unconstrained imaging conditions [10]. Moreover, the machine learning methods employed by these systems did not fully Figure 1. Faces from the Adience benchmark for age and gender classifification [10].

These images represent some of the challenges of age and gender estimation from real-world, unconstrained images. Most notably, extreme blur (low-resolution), occlusions, out-of-plane pose variations, expressions and more exploit the massive

numbers of image examples and data available through the Internet in order to improve classification capabilities.
In this paper we attempt to close the gap between automatic face recognition capabilities and those of age and gender estimation methods. To this end, we follow the successful example laid down by recent face recognition systems: Face recognition techniques described in the last few years have shown that tremendous progress can be made by the use of deep convolutional neural networks (CNN) [31].

We demonstrate similar gains with a simple network architecture, designed by considering the rather limited availability of accurate age and gender labels in existing face data sets.
We test our network on the newly released Adience benchmark for age and gender classification of unfifiltered face images [10]. We show that despite the very challenging nature of the images in the Adience set and the simplicity of our network design, our method outperforms existing state of the art by substantial margins. Although these results provide a remarkable baseline for deep-learning-based approaches, they leave room for improvements by more elaborate system designs, suggesting that the problem of accurately estimating age and gender in the unconstrained settings, as reflflected by the Adience images, remains unsolved.
In order to provide a foothold for the development of more effective future methods, we make our trained models and classification system publicly available

A CNN for age and gender estimation
Gathering a large, *labeled* 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 classifification data-sets (e.g. the Imagenet dataset [45]). Overfifitting 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 overfifitting under such circumstances.

3.1. **Network architecture**
Our proposed network architecture is used throughout our experiments for both age and gender classifification. It is illustrated in Figure 2. A more detailed, schematic diagram of the entire network design is additionally provided in Figure 3. The network comprises of only three convolutional layers and two fully-connected layers with a small number of neurons. This, by comparison to the much larger architectures applied, for example, in [28] and [5]. Our choice of a smaller network design is motivated both from our desire to reduce the risk of overfitting as well as the nature
36Figure 3. Full schematic diagram of our network architecture. Please see text for more details. of the problems we are attempting to solve: age classification on the Audience set

requires distinguishing between eight classes; gender only two. This, compared to, e.g., the ten thousand identity classes used to train the network used
for face recognition in [48].

All three color channels are processed directly by the network. Images are fifirst rescaled to 256 × 256 and a crop of 227 × 227 is fed to the network. The three subsequent convolutional layers are then defined as follows.

1. 96 filters of size 3×7×7 pixels are applied to the input in the first convolutional layer, followed by a rectified linear operator (ReLU), a max pooling layer taking the maximal value of 3 × 3 regions with two-pixel strides and a local response normalization layer [28].

2. The 96 × 28 × 28 output of the previous layer is then processed by the second convolutional layer, containing 256 filters of size 96 × 5 × 5 pixels. Again, this is followed by ReLU, a max pooling layer and a local response normalization layer with the same hyper parameters as before.

3. Finally, the third and last convolutional layer operates on the 256 × 14 × 14 blob by applying a set of 384 filters of size 256 × 3 × 3 pixels, followed by ReLU and a max pooling layer. The following fully connected layers are then defined by:

4. A first fully connected layer that receives the output of the third convolutional layer and contains 512 neurons, followed by a ReLU and a dropout layer.

5. A second fully connected layer that receives the 512- dimensional output of the first fully connected layer and again contains 512 neurons, followed by a ReLU
and a dropout layer.

6. A third, fully connected layer which maps to the final classes for age or gender.

Finally, the output of the last fully connected layer is fed to a soft-max layer that assigns a probability for each class. The prediction itself is made by taking the class with the maximal probability for the given test image.

3.2. **Testing and training**

Initialization. The weights in all layers are initialized with random values from a zero mean Gaussian with standard deviation of 0.01. To stress this, we do not use pretrained models for initializing the network; the network is trained, from scratch, without using any data outside of the images and the labels available by the benchmark. This, again, should be compared with CNN implementations used for
face recognition, where hundreds of thousands of images are used for training [48].

Target values for training are represented as sparse, binary vectors corresponding to the ground truth class of the number of classes (two for gender, eight for the eight
age classes of the age classifification task), containing 1 in the index of the ground truth and 0 elsewhere Network training. Aside from our use of a lean network
architecture, we apply two additional methods to further limit the risk of overfifitting. First we apply dropout learning [24] (i.e. randomly setting the output value of net
work neurons to zero). The network includes two dropout layers with a dropout ratio of 0.5 (50% chance of setting a neuron's output value to zero). Second, we use data augmentation by taking a random crop of 227 × 227 pixels from the 256 × 256 input image and randomly mirror it in each forward-backward training pass. This, similarly to the multiple crop and mirror variations used by [48]. Training itself is performed using stochastic gradient decent with image batch size of fififty images. The initial learning rate is e−3, reduced to e−4 after 10K iterations. Prediction. We experimented with two methods of using the network in order to produce age and gender predictions
for novel faces:

• Center Crop: Feeding the network with the face image, cropped to 227 × 227 around the face center.
• Over-sampling: We extract fifive 227 × 227 pixel crop regions, four from the corners of the 256 × 256 face image, and an additional crop region from the center
of the face. The network is presented with all five images, along with their horizontal reflections. Its final prediction is taken to be the average prediction value
across all these variations. We have found that small misalignments in the Audience images, caused by the many challenges of these images (occlusions, motion blur, etc.) can have a noticeable impact on the quality of our results. This second, over-sampling method, is designed to compensate for these small misalignments, bypassing the need for improving alignment quality, but rather directly feeding the network with multiple translated versions of the same face.

4. Experiments

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.

4.1. **The Audience benchmark**

We test the accuracy of our CNN design using the recently released Adience benchmark [10], designed for age and gender classifification. The Adience set consists of images automatically uploaded to Flickr from smart-phone devices. Because these images were uploaded without prior manual fifiltering, as is typically the case on media webpages (e.g., images from the LFW collection [25]) or social
websites (the Group Photos set [14]), viewing conditions in these images are highly unconstrained, reflflecting many of the real-world challenges of faces appearing in Internet images. Adience images therefore capture extreme variations in head pose,

lightning conditions quality, and more. The entire Adience collection includes roughly 26K images of 2,284 subjects. Table 1 lists the breakdown of the collection into the different age categories. Testing for both age or gender classifification is performed using a standard five-fold, subject-exclusive cross-validation protocol, defined in [10]. We use the in-plane aligned version of the faces, originally used in [10]. These images are used rather than newer alignment techniques in order to highlight the performance gain attributed to the network architecture, rather than better preprocessing. We emphasize that the same network architecture is used for all test folds of the benchmark and in fact, for both gender and age classifification tasks. This is performed in order to ensure the validity of our results across folds, but also to
demonstrate the generality of the network design proposed here; the same architecture performs well across different, related problems. We compare previously reported results to the results computed by our network. Our results include both methods for testing: center-crop and over-sampling (Section 3).

4.**2. Results**

Table 2 and Table 3 presents our results for gender and age classifification respectively. Table 4 further provides a confusion matrix for our multi-class age classifification results. For age classifification, we measure and compare both the accuracy when the algorithm gives the exact age-group classifification and when the algorithm is off by one adjacent age-group (i.e., the subject belongs to the group im
mediately older or immediately younger than the predicted group). This follows others who have done so in the past, and reflflects the uncertainty inherent to the task – facial features often change very little between oldest faces in one age class and the youngest faces of the subsequent class. Both tables compare performance with the methods described in [10]. Table 2 also provides a comparison with [23] which used the same gender classifification pipeline of [10] applied to more effective alignment of the faces; faces in their tests were synthetically modified to appear facing forward.

Evidently, the proposed method outperforms the reported state-of-the-art on both tasks with considerable gaps. Also evident is the contribution of the over-sampling approach, which provides an additional performance boost over the original network. This implies that better alignment (e.g., frontalization [22, 23]) may provide an additional boost in performance. We provide a few examples of both gender and age misclassifications in Figures 4 and 5, respectively. These show that many of the mistakes made by our system are due to extremely challenging viewing conditions of some of the Adience benchmark images. Most notable are mistakes caused by blur or low resolution and occlusions (particularly from heavy makeup). Gender estimation mistakes also frequently occur for images of babies or very young children where
obvious gender attributes are not yet visible.

Conclusions

Though many previous methods have addressed the problems of age and gender classifification, until recently, much of this work has focused on constrained images taken in lab settings. Such settings do not adequately reflflect appearance variations common to

the real-world images in social websites and online repositories. Internet images, however, are not simply more challenging: they are also abundant. The easy availability of huge image collections provides modern machine learning based systems with effectively endless training data, though this data is not always suitably labeled for supervised learning. Taking example from the related problem of face recognition we explore how well deep CNN perform on these tasks using Internet data. We provide results with a lean deep-learning architecture designed to avoid overfifitting due to the limitation of limited labeled data. Our network is “shallow” compared to some of the recent network architectures, thereby reducing the number of its parameters and the chance for overfifitting. We further inflflate the size of the
training data by artifificially adding cropped versions of the images in our training set. The resulting system was tested on the Adience benchmark of unfifiltered images and shown to signifificantly outperform recent state of the art. Two important conclusions can be made from our results. First, CNN can be used to provide improved age and gender classifification results, even considering the much smaller size of contemporary unconstrained image sets labeled for age and gender. Second, the simplicity of our model implies that more elaborate systems using more training data
may well be capable of sub

Acknowledgments

This research is based upon work supported in part by the Offifice of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA 2014-14071600010. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the offificial policies or endorsements, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright annotation thereon.
-

2.2 Human Age And Gender Classification using Convolutional Neural Network

Abstract

Pattern recognition and automatic classification are very active research areas, their main objectives are to develop intelligent systems able to achieve efficiently learning and recognizing objects. An essential section of these applications is attached to biometrics, which is used for security purposes in general. The facial modality as a fundamental biometric technology has become increasingly important in the field of
research. The goal of this work is to develop a gender prediction and age estimation system based on convolutional neural networks for a face image or a real-time video. In this paper, three CNN network models were created with different architecture (the number of filters, the number of convolution layers...) validated on IMDB and WIKI dataset, the results obtained showed that CNN networks greatly improve the performance of the system as well as the accuracy of the recognition.

INTRODUCTION

Biometrics, is the science of analyzing the physical or behavioral characteristics of each individual that enable the authentication of their identity in a reliable manner, it offers significant advantages over conventional identification methods, such as passwords and cards, are not transferable, exclusive to each person and are not lost or stolen, particularly because of biometric features. The range of biometric solutions
relies on user approval, security, cost and time for implementation...etc. Recently, face recognition has been one of the most interesting tasks in pattern recognition, many applications use this technique because the human face is considered a very rich source of information. In particular, gender and age are facial features that can be very useful for a multitude of applications, for example an automatic gender and age prediction system is used to profile customers who are interested for a product or for
target advertising. The areas of age and gender classification have been studied for decades. Until detailing the methods used in this article, we will first provide a summary of the facial recognition experiments carried out by scholars, which can be
grouped into tree classes of interest. [1]. Local strategies extract facial characteristics by focusing on the key points of the face, such as nose, mouth, eyes, which will offer more information. Global approaches their concept is to use the full surface of the face as a source of information independent of local characteristics such as eyes, lips ... etc. Hybrid techniques they merge all sorts of strategies, theoretically giving the best of all [1] [2]. Many techniques were applied for gender prediction from
face images, Hui-Cheng Lian et al [2] proposed gender recognition taking into account both form and texture information from facial images. This last is divided into small regions, from which local binary histograms are extracted and concated into a single vector representing the facial image, then the support vector machine (SVM) is applied for gender prediction. Jing Wu et al [3] proposed a gender classification using the form shading (SFS) and multi layers perceptron (MLP)
was applied for this study by Golomb et al [4]. Khan et al [5] applied classifier reinforcement in particular adaboost for gender prediction during this period. Among age prediction studies, Yamaguchi et al [6] confirmed that differences between the characteristics of an adult's face and a child's include the length of the face and the ratio of each side. Burt and Perrett [7] studied the age estimate based on the use of average faces of people between 25 and 60 years of age, they used approach that
generalize facial texture and shape, also Ueki and Coll [8] reported a method of classifying age groups by linear discriminating analysis. Although the SVM has been tested for age classification several times [9]. Kwon and Lobo [10] defined a method for classifying input images into one of three age groups: child, young and old using texture information. However, almost all previous research has been based on craniofacial development method and analysis of skin wrinkles. During the last few years, a convolution neural network centered on deep learning [11], according to the powerful ability to estimate and extract features to enhance the precision of image classification, state-of-the-art achievements have been achieved in large areas. In this paper, we will first introduce the basic structure of the convolutional neural network. Next, we will

describe models CNN for training data to classify gender and age, then we will present the results obtained using a model trained by these data,
and finally, our conclusion.

EXPERIMENTAL WORK AND RESULTS

A. Experiment and result for gender prediction

The CNNs models proposed in table 1 was build using Keras which has many advantages to improve efficiency of the model. We input 2500 images of male and female separately 2000 images for train and 500 images for the test. The CNNs models were trained for 1500 epochs, after every epoch the accuracy was calculated, which is the count of predictions where the predicted value is equal to the true value, it is typically expressed as a percentage. The input is passed through a pile of convolutional and maxpooling layer, the non-linear activation function (ReLu) was
used, in output result we applied a sigmoid function as shown in table 1, for all models, RMSpro was used as an optimizer.

D. Discussion

In this work, we aimed to automate a system for gender prediction and age estimation by using CNN and deep learning techniques, first, we build three models: CNN1, CNN2, CNN3 as described in table 1, these models were trained on IMDB dataset, we noticed that the CNN 3 present best results compared to the CNN 2 and CNN 1, due to the depth of the network. In CNN 3 we used three convolutional layer but in CNN 2 and CNN 1 we used only two layers of convolution with various filter size,16 filters were used at the 1st convolution layer in CNN 1, 32 filters were applied in the 1st convolutional layer in CNN 2, when the number of the filter was large the performance of system increase. In other word, the depth of network and the number of the filters have a great influence in creating an efficient convolutional network ranking. For age estimation, we used the model CNN 3 to classify age in three categories; young (20-39 years), middle (40-59 years), old (more than 60 years), this kind of classification will eventually be useful for marketing to identify customers. After training model CNN 3, we noticed that this CNN model can obtain a perfectly acceptable result, as show in figures 6 and 7. Furthermore, the rate of classification growth with the number of epochs, this reflects that with each epoch the model
learns more information.

Fig.8. Example of age estimation form IMDB dataset.
Fig.9. Example of gender prediction form WIKI dataset.
Fig.10. Example of age and gender classification from IMDB dataset.

VII. CONCLUSION

In this article, we analyzed the implementation of deep convolutional neural network for human age and gender prediction using CNN. During this study various design was

developed for this task, age and gender classification is one of the key segments of research in the biometric as social applications with the goal that the future forecast and the information disclosure about the particular individual should be
possible adequately. In this study, the main conclusion that can be drawn is that age and gender from face recognition are very popular among panels to implement an intelligent system that can achieve good and robust results in the accuracy of recognition, we employed a deep learning algorithm, as a convolutional neural network to propose a simple study contain various CNNs model in gender
classification, trained in well-known datasets IMDB-WIKI, then we applied an efficient model for age estimation, the different results obtained in terms of precision, compared with those cited in the state of the art, have shown that the depth of the
convolutional networks used in this work is an important factor in achieving better precision. The interpretation of the figure (4- 5-6-7) and the outcome of tables 3 and 4 was based on parameter settings in our experiment described in table 2. The proposed network provides significant precision improvements in age and gender classification, but takes considerable time to train the network to implement the correct prediction.
Finally, as a perspective, an extension of this work can be envisaged by creating a face detection and recognition system based on CNNs as a feature extractor and the machine vector support as a classifier, another perspective would be the tests our
approach on other facial databases showing strong variations in lighting and pose.

2.3 Convolutional Neural Network for age and gender classification

Abstract

This paper focuses on the problem of gender and age classifification for an image. I build off of previous work [12] that has developed effificient, accurate architectures for these tasks and aim to extend their approaches in order to improve results. The fifirst main area of experimentation in this project is modifying some previously published, effective architectures used for gender and age classifification [12]. My
attempts include reducing the number of parameters (in the style of [19]), increasing the depth of the network, and modifying the level of dropout used. These modifications actually ended up causing system performance to decrease (or
at best, stay the same) as compared with the simpler architecture I began with. This verifified suspicions I had that the tasks of age and gender classifification are more prone to over-fitting than other types of classification. The next facet of my project focuses on coupling the architectures for age and gender recognition to take adwe must make do with the nature of this problem we are apvantage of the gender-specifific age characteristics and age specifific gender characteristics inherent to images. This stemmed from the observation that gender classifification is an inherently easier task than age classifification, due to both the fewer number of potential classes and the more prominent intra-gender facial variations. By training different age classififiers for each gender I found that I could improve the performance of age classification, although gender classification did not see any significant gains.

1. Introduction

Over the last decade, the rate of image uploads to the Internet has grown at a nearly exponential rate. This new found wealth of data has empowered computer scientists to tackle problems in computer vision that were previously either irrelevant or intractable. Consequently, we have witnessed the dawn of highly accurate and effificient facial detection frameworks that leverage convolutional neural networks under the hood. Applications for these systems include everything from suggesting who to “tag” in Facebook photos to pedestrian detection in self-driving cars. However
the next major step to take building off of this work is to ask not only how many faces are in a picture and where they are, but also what characteristics do those faces have. The goal of this project do exactly that by attempting to classify the age and gender of the faces in an image. Applications for this technology have a broad scope and
the potential to make a large impact. For example, many languages have distinct words to be used when addressing a male versus a female or an elder versus a youth. Therefore automated translation services and other forms of speech generation can factor in gender and age classifification of subjects to improve their performance. Also, having an idea about the age and gender of a subject makes the task of
recognizing that subject signifificantly easier. This could be used to aid assisted vision devices for those with deteriorating, or lost, eyesight. Social media websites like Face book could use the information about the age and gender of the people to better infer the context of the image. For example, if a picture contains many people studying together, Facebook might be able to caption the scene with “study session.” However if it can also detect that the people are all men in their early 20s and that some are wearing shirts with the same letters, it may predict “College students in a fraternity studying.” Age and gender classifification is an inherently challenging problem though, more so than many other tasks in computer vision. The main reason for this discrepancy in difficulty lies in the nature of the data that is needed to train these types of systems. While general object classification tasks can often have access to hundreds of thousands, or even millions, of images for training, datasets with age
and/or gender labels are considerably smaller in size, typically numbering in the thousands or, at best, tens of thousands. The reason for this is that in order to have labels for such images we need access to the personal information of the subjects in the images. Namely we would need their date of birth and gender, and particularly the date of birth is a rarely released piece of information. Therefore, proaching and tailor network architectures and algorithmic approaches to cope with these limitations. These reasons are the primary motivation behind [12] choosing to implement a relatively shallow architecture for age and gender classifification using convolutional neural networks, and we have followed this pattern. The input to my algorithm is an image of a human face of size 256x256 that is then cropped to 227x227 and fed into
either the age classififier, gender classififier or both. The age classififier returns a integer representing the age range of the individual. There are 8 possible age ranges (see Section 4), so the age classififier returns an integer between 0 and 7. The gender classifier returns a binary result where 0 indicates male and 1 represents female.

Methods

3.1. Network Architecture

The network architecture used throughout my project is based off of the work in [12]. As mentioned toward the end of Section 2, this network design is intended to be relatively shallow so as to prevent over-fifitting the data. Figure 1 visualizes the network, which is explained below. An RGB image being input to the network is fifirst scaled to 3x256x256 and then cropped to 3x227x227. The types of cropping are described further in Section 3.2. There are 3 convolution layers, followed by 3 fully connected layers. The convolution layers are:

1. Conv1- 96 fifilters of size 3x7x7 are convolved with stride 4 and padding 0, resulting in an output volume size of 96x56x56. This is followed by a ReLU, max pooling pooling which reduces the size to 96x28x28, and a local-response normalization (LRN).

2. Conv2- 256 fifilters of size 96x5x5 are convolved with stride 1 and padding 2, resulting in an output volume size of 256x28x28. This is also followed by a ReLU, max-pool, and LRN, reducing the output size to 256x14x14.

3. Conv3- 256 fifilters of size 256x3x3 are convolved with stride 1 and padding 1, followed by a ReLU and maxpool, resulting in an output volume of 256x7x7. The fully connected layers are
1. FC6- 512 neurons fully connected to the 256x7x7 output of Conv3, followed by a ReLU layer and dropout layer.
2. FC7- 512 neurons fully connected to the 1x512 output of FC6 followed by a ReLU layer and dropout layer.
3. FC8- 2 or 8 neurons fully connected to the 1x512 output of FC7, yielding the un-normalized class scores for either gender or age, respectively.

And finally there is a softmax layer that sits on top of FC8, which gives the loss and fifinal class probabilities.

3.2. Training and Testing

The authors of [12] split the images (as described in Section 4) into 5 folds and then perform a subject-exclusive cross-validation protocol, as fifirst developed by [2]. The reason this type of protocol is necessary is because of the nature of the dataset being used, which contains multiple pictures of the same subjects. Therefore if the images were simply randomly shufflfled and divided into fifths, the same subjects could potentially appear in both the training and test folds, thereby skewing the results to seem more promising than they are in reality. This protocol therefore ensures that all the images of a given subject appear in a single fold to avoid this issue.
For my project, I divided the dataset into 6 subject exclusive folds, but then further divided each of those folds into males, females, and each of the 8 age groups. As de scribed in Section 3.3, this was necessary for the types of classifiiers I was aiming to build. This resulted in a total of 66 “sub-folds”, where each of the original 6 folds were

broken up into 11 groups, based on the types of classififiers I would be training. All of the sub-folds coming from the 6th original fold were separated as test data, never to be trained on or validated against. Then the remaining 5 folds, and their sub-folds, were used for training, and cross-validation. So if, for example, at a given time I was training a male age classififier, I would use 4 of the male age sub-folds as the
training set and the 5th male age sub-fold as the validation set. This would rotate between the fifirst 5 sub-folds for every possible assignment of the validation fold, and the resulting classififier would be tested against the 6th (previously separated) male age sub-fold. This altered fold distinction I implemented prevented me from being able to use the pretrained weights provided by [12] because by construction their training folds could overlap with my test folds. At fifirst I did not see this issue and then initial experiments showed uncharacteristically high accuracies that led to further investigation and the discovery of this inherent problem.

Finally, [12] proposes 2 types of sampling of an input image when being classified. One is to simply take a center crop of 227x227 out of the 256x256 image and classify
that. The other is to take 5 such crops, one from each of the corners and one from the center, classify them all, and take the majority classifification from between them. While they found that the latter technique can improve accuracy slightly, for the sake of reducing testing time, I used the first approach for this project.

3.3. Goals

My fifirst objective in this project was to determine if the proposed network architecture (see Section 3.1) was indeed optimal. Although the authors of [12] claimed that any deeper network would suffer from over-fifitting, I wanted to verify this for myself. To this end I experimented with adding additional convolution layers, removing fully connected layers (in the style of [19]), and modifying the parameters used for dropout as well as LRN. The primary goal, however, was to experiment with a new higher-level approach for composing these classifiers to improve performance. The observation I made early on was that gender classifification is an inherently easier task than age classifification, both due to the fewer number of classes to distinguish between and the more marked differences that exist between genders than between many age groups. This then led me to the conclusion that while it is
reasonable to assume one should be able to ascertain someones gender apart from knowing their age, or vice versa, there is also some plausibility of using one of these attributes to better inform the prediction of the other. For example, the amount of hair on a man's head can often be a useful indicator of age, but the same is not true for women.

Furthermore, separating the tasks of classifying men's age and women's age should, in theory, give the networks more expressive power by freeing them from having to learn a gender-neutral concept of age. Therefore, I proposed that training separate age classifiers for men and women could simulate the added power of deeper networks while avoiding the danger of over-fitting.

3.4. Technical Details

In this section, I elaborate on some of the technical details of the network architecture and how it is trained. **Local Response Normalization (LRN).** After the first 2 pooling layers, there are local response normalization (LRN) layers. LRN is a technique that was first introduced in [9] as a way to help the generalization of deep CNNs. The idea behind it is to introduce lateral inhibition between the various fifilters in a given convolution by making them "compete" for large activations over a given segment of their input. Effectively this prevents repeated recording of the same
information in slightly different forms between various kernels looking at the same input area and instead encourages fewer, more prominent, activations in some for a given area. If *ai x,y* is the activation of a neuron by applying kernel *i* at position (*x, y*), then it's local response normalized activation *bi x,y* is given by
bi x,y = ai x,y/

k + *α*
min(*N*
−1*,i*+*n/*2)
X
j=*max*(0*,i*−*n/*2)
(*aj x,y*)2

β
where *k*,*n*,*α*, and *β* are all hyper-parameters. The parameter *n* is the number of "adjacent" kernel maps (filters) over which the LRN is run, and N is the total number
of kernels in that given layer. The values used for these
hyperparameters are the same as those used in [9]. **Softmax.** At the top of the proposed architecture lies a softmax layer, which computes the loss term that is optimized during training and also the class probabilities during a classifification. While some loss layers like multiclass SVM loss treat the output of the fifinal fully connected layer as the class scores, softmax (also known as multinomial logistic regression) treats these scores as the unnormalized log probabilities of the classes. That is, if we have *zi* is the score assigned to class *i* after the fifinal fully connected layer, then the softmax function is
fj (*z*) =
ezj
P *k ezk*
Because we want to maximize the log likelihood of the correct class, the term we want to minimize is the negative log likelihood.
Li = −*log*(
efyi
P *j efj*
)
Because the softmax function takes real-valued scores being output from *f* and normalizes them by their exponentiated sum, it guarantees that the sum of all softmax scores is 1, thereby allowing it to be interpreted as a true class probability. It should be noted that the softmax loss is actually a particular form of a cross-entropy loss. More specifially, the

cross-entropy between an actual distribution p and an approximate distribution q is deffined as
$H(p, q) = -X$
x
$p(x) log q(x)$
From this it can be seen that the softmax classififier is really just minimizing the cross-entropy between the estimated class probabilities and the real distribution, which would look like 1 predicted for the actual class and 0 predicted for everything else.

Stochastic Gradient Descent. Now that we know how to calculate the loss, we need to know how to minimize it in order to train an accurate classifier. The type of optimization used in this experiment is stochastic gradient descent. In order to explain this, first I will elaborate on the more generalized form of gradient descent. The gradient of a function is really just its derivative, and therefore by definition it is the direction of steepest ascent (or descent if you move backwards along it). Therefore if we compute the gradient of the loss function with respect to all of the system variables/weights (in CNNs there can be up to millions of these), we will have the direction along which we can move toward our minimum loss most quickly by following the negative of the gradient. Each time we compute the gradient we take a small step (governed by a hyperparameter) in the opposite direction, and we re-evaluate the loss, re-compute the gradient, and repeat. The hope (and in fact the reality) is that by repeating this process we will iteratively decrease our loss function, which is reflective of the model becoming iteratively better at its classification task.
Mathematically, we can write this as
$\mathbf{w} = \mathbf{w} - \eta \nabla \mathbf{w} L$
where η is the learning rate, also sometimes called the step size and $\nabla \mathbf{w} L$ is the gradient of the loss term with respect to the weight vector **w**. While this is theoretically great, the truth is that computing the gradient across the entire training set in order to make an incremental update to the weights is prohibitively computationally expensive. Therefore alternate approaches have been invented that evaluate the gradient of the loss 4function over a sample of the training data, and use that approximate gradient to make the update. The reason this gradient is approximate is that although it is the optimal direction to travel down given the sample of images it was computed over, there is no telling what kinds of images it did not look at when computing the gradient. Therefore making this form of mini-batch gradient descent, as it is called, will still usually reach the minimum loss over time (or at least a local minimum), but it will require many more iterations on average. However the time it takes to evaluate the gradient drops so dramatically when we operate on a mini-batch that it is actually significantly faster to perform many more mini-batch gradient updates than a few full gradient updates. Finally, stochastic gradient descent is a special form of gradient descent in which the mini-batch size is 1. This is extremely fast to compute since it only requires passing 1 image forward (to calculate the loss) and backward (to calculate the gradient) through the network, but the gradients are even less globally optimal than mini-batch gradient descent, therefore a smaller step size is required at each iteration and many more iterations are required.

4. Dataset

The dataset used for training and testing for this project is the Adience face dataset, which comes from the Face Image Project[12] from the Open University of Israel (OUI). This dataset contains a total of 26,580 photos of 2,284 unique subjects that are collected from Flickr. Each image is annotated with the person's gender and age-range (out of 8 possible ranges). The images are subject to various levels of occlusion, lighting, and blur, which reflflects real-world circumstances. I used those images which were mostly front facing, which limited the total number of images to around
20,000. Table 1 includes details regarding the distribution of images in each gender and age range. Figure 2 shows some examples of images of both males and females in the dataset of various ages. The images were originally of size 768x768, so they were preprocessed by all being resized down to 256x256.

Figure 2. **Adience image dataset examples**. Top row: 6 males of various ages. Bottom row: 6 females of various ages.

5. Experiments

The training and testing for this project were done exclusively using Caffe [6], running on Amazon EC2 using be- 0-2 4-6 8-13 15-20 25-32 38-43 48-53 60+ Total

Male 745 928 934 734 2308 1294 392 442 8192
Female 682 1234 1360 919 2589 1056 433 427 9411
Both 1427 2162 2294 1653 4897 2350 825 869 19487

Table 1. **Adience image dataset distribution**. Number of images for each gender and age range between 1 and 3 instances at a time, each with 1,536 CUDA cores and 4GB of video memory. As described in Section 3.2, a 6-fold subject-exclusive cross validation protocol was used to provide added robustness to the training, while
preserving the reliability of results. Although much of my network architecture was built off of the work in [12], my networks were trained from scratch without using any pretrained weights provided by that, or any other, project. The reason for this had to do with the way that I divided up the dataset for the purposes of my project (again, see Section 3.2 for more information). The fifirst step I took was to attempt to reproduce the results of [12] as a baseline since I had their network architecture and training data. I attempted to replicate their experiment as closely as possible, so I also used SGD with a learning rate of 1e-3 that decays by a factor of 10 every 10,000 iterations and a batch size of 50 images. This proved quickly successful, and within a few hours of training I reached accuracies that were very close to their results for both age and gender classification. These results are recorded in Table 2. Note that their slightly higher accuracies are likely due to the oversampling they do of the input images followed by taking the majority classifification of the various samples. For the sake of faster iteration in my model, I avoided this technique tribute much to the overall performance and that depth in the convolutional layers is actually preferable. To this end I removed 1 and then 2 of the fully connected layers (out of 3) and added 1 and then 2 additional convolution layers (on top of the existing 3). I attempted 5 different combinations of these modified architectures, but like with the attempt at

using Adam, after multiple days there was no clear benefifit, and if anything added complexity was making the system perform the same or, sometimes, worse.

At this point I chose to focus my efforts on the main insight that motivated this project, which was that coupling the architectures for gender and age classifification could give more expressive power to the classififiers, particularly for age. As a sort of "proof of concept", I attempted to train classififiers on each gender separately to see what would happen. The results pleasantly surprised me and are summarized in Figure 3. I saw that when training a classififier from the ground up only on male images, the accuracy when predicting the age of men increased. Conversely, I saw the accuracy of classifying women's ages decreases over the average (which may or may not be taken as a social commentary on how women are more effective at hiding their age)fifiers for each age group, and those results are summarized in Table 3. These results are less striking than those of Figure 3, but there is some reassurance in how intuition lines up with the results. Namely it can be seen that the age range in which it is most diffificult to predict gender, with just a 27% accuracy, is 0-2 years old. Of course though that makes perfect sense as gender-specifific features are not usually present at such a young age, or at least not as much as later in life. Also the age range in which gender prediction is the best is 15-20, which also seems reasonable since that is the time when there is the most development of gender-specifific features.
Given these results, it seemed most promising to use the remaining time I had to develop and train a chained gender age network that would fifirst classify gender (as before)

Conclusion

Although many previous methods have tackled the problem of age and gender classification of images, in this paper. I establish a benchmark for the task based on state-of-the-art network architectures and show that chaining the prediction of age with that of gender can improve overall accuracy. If there had been more time, I would have dedicated more effort towards fine-tuning the parameters and the modified architectures I experimented with. Specifically, I would have liked to get the Adam learning algorithm in place with equal or improved performance to SGD, and I would have liked to replace the multiple fully connected layers at the end of the architecture with only one and instead shifted those parameters over to additional convolutional layers. By far the most difficult portion of this project was setting up the training infrastructure to properly divide the data into folds, train each classifier, cross-validate, and combine the resulting classifiers into a test-ready classifier. I foresee future directions building off of this work to include using gender and age classification to aid face recognition, improve experiences with photos on social media, and much more. Finally I hope that additional training data will become available with time for the task of age and gender classification which will allow successful techniques from other types of classification with huge datasets to be applied.

2.4Age and Gender Detection using Python

ABSTRACT

In this paper, the author has worked on a technique for age and gender classification using python algorithm. Human identification and classification are being utilized in various field for a very long time. Fields like Government ID Cards, Verification procedures etc. We have already developed techniques like retina scan, iris scans, fingerprint and other sophisticated systems such as DNA fingerprinting to identify the individuals. Although these already built methods works efficiently, the hardware, software and human proficiency requirement are way too demanding for several simpler task which may or may not require a professional efficiency. Technique reported in this paper is simple and easy for human classification which can be performed using only a webcam and a decent computer system.

INTRODUCTION

Human Classification is an age-old procedure and being done in various fields and technology such as biometrics, forensics sciences, Image processing, Identification system, etc. With the development of Artificial Intelligence and techniques such as Neural Network and Deep Learning, it has become increasingly easier to classify human. These new technologies help identification, classification of Individuals without the need of another professional or Individual records. Also Being immensely fast, these technologies can classify millions of individuals way faster than a professional. Human Facial Image Processing provides many clues and cues applicable to industries such as security, entertainment, etc [1]. Human Face can provide immense amount of information like their emotional state, slightest agreement or disagreement, irony or anger, etc. This is the reason why faces have been long research topic in psychology [2]. This data (or in our case digital data) is very valuable as they help recognition, selection or identification of individual according to the requirement. Age and Gender Detection can alone provide a lot of information to places such as recruitment team of organizations, Verification of ID cards, example: Voter ID cards which millions of individual uses to cast their vote at the time of election, etc. Human Facial Image processing eases the task of finding ineligible or counterfeit individuals.

PROCEDURE

Since the technique is implemented, we can start testing it for its accuracy. The general procedure to be followed is • Input the data. • Create a frame. • Detect the face. • Classify the Gender. • Classify the Age Group. • Attach the result in the image. • Output the image in specified location. 4. TEST RUN To verify the efficiency of the technique we collected some of human face along with their mentioned ages when the photo was captured and fed them to the program. The performance can be judged using the chart human face like non-human object was provided as an input and since the no data for the non-human object was stored in training datasets it gave the inaccurate results.

5. KEY FEATURES

Main aim of this technique is to provide faster and cost-effective method of age and gender classification of human. Key Features of this model are: • There is no need of high precision hardware or software. It can process the image directly through the camera device such as webcam. Although a better device will provide more efficient result. • This technique is easy to use, it does not require a professional level knowledge. A normal computer knowledge is enough. • It can process and store hundreds of faces along with the corresponding result without any lag or delay

. USE CASES Several uses cases for this project includes the following: • Identification of the target audience in marketing organisation. • In Recruitment procedure, to verify legitimacy of the applicants. • Verification of authentic person applying for government IDs. • Classification of human resources in bulk.

. **CONCLUSION**

"Human Age and gender classification" are two of the many important information gathering resource from and individual. Human faces provide enough data which may be used for many purposes. In order to reach the correct audience human age and gender classification is very essential. Here we tried to do the same process but with general equipment. The efficiency of the algorithm depends on several factor but the main motif of this project is being easy and faster while also being as accurate as possible. Work is being done to the improve the efficiency of the algorithm. Some future improvements include discarding the face like non-human objects, more datasets for people belonging to different ethnic groups and more granular control over the workflow of the algorithm.

CHAPTER 3

SYSTEM DESIGN

OBJECTIVE

- From the camera sources, from satellites, aeroplanes, and the images caught in everyday lives is called picture processing.

- Image processing has two main steps followed by simple steps. The improvement of an image with the end goal of more good quality pictures; that can be adopted by other programs are called picture upgrades.

- The other procedure is the most pursued strategy utilized for the extraction of data from a picture. The division of images into certain parts is called segmentation.
- The evolving of ideas helps in figuring certain boundaries. Age assessment is a multi-class issue in which the years; are categorized into classes. Individuals of various ages have various facials, so it is hard to assemble the pictures.

- To identify the age and gender of several faces' procedures, are followed by several methods. From the neural network, features are taken by the convolution network. In light of the prepared models, the image is processed into one of the age classes. The highlights are handled further and shipped off the preparation frameworks.

EXISTING SYSTEM

- ecascaded Adaboost learning algorithm in face detection and achieved the age estimation mechanism using Gabor wavelets and OLPP.

- This paper is organized in the following sections. First, our presented face detection system includes histogram lighting normalization, feature selection, the cascaded Adaboost classifier and the region‑based clustering algorithm.
- The age estimation process, including the feature extraction using Gabor wavelets, feature reduction and selection, and age classification, is then introduced.
- Finally, the experimental results and conclusions are provided and summarized

DISADVANTAGE

The downside to Haar cascades is that they tend to be prone to false-positive detections, require parameter tuning when being applied for inference/detection, and just, in general, are not as accurate as the more "modern" algorithms

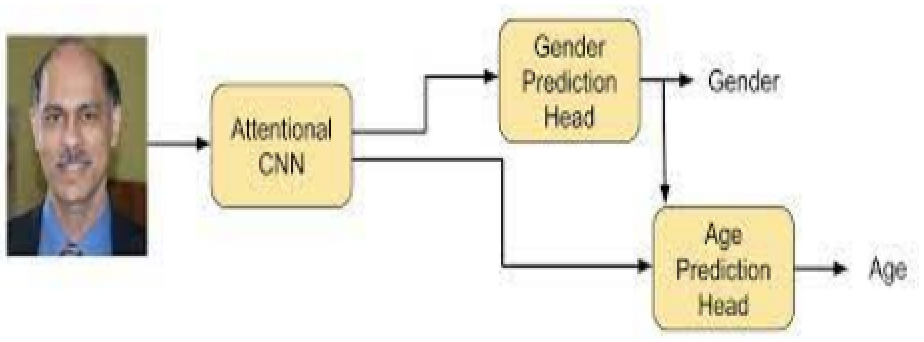
PROPOSED SYSTEM

- Age and Gender Detection, Deep EXpectation (DEX) – is used for age estimation which can be seen in image classification [5, 32, 47] and object detection fuelled by deep learning. From the deep learning concept we learn four key ideas that we apply to our solution:
- the deeper the open cv(by sheer increase of parameters / model complexity) the better is the capacity to model highly non-linear transformations - with some optimal depth on current architectures;
- the larger and more diverse the datasets used for training, the better the network learns to generalize and the more effective it becomes to over-fitting;
- the alignment of the object in the input image impacts the overall performance;
- when the training data is small that is when we must finetune a network pre-trained for comparable inputs and goals which would benefit us from the transferred knowledge.

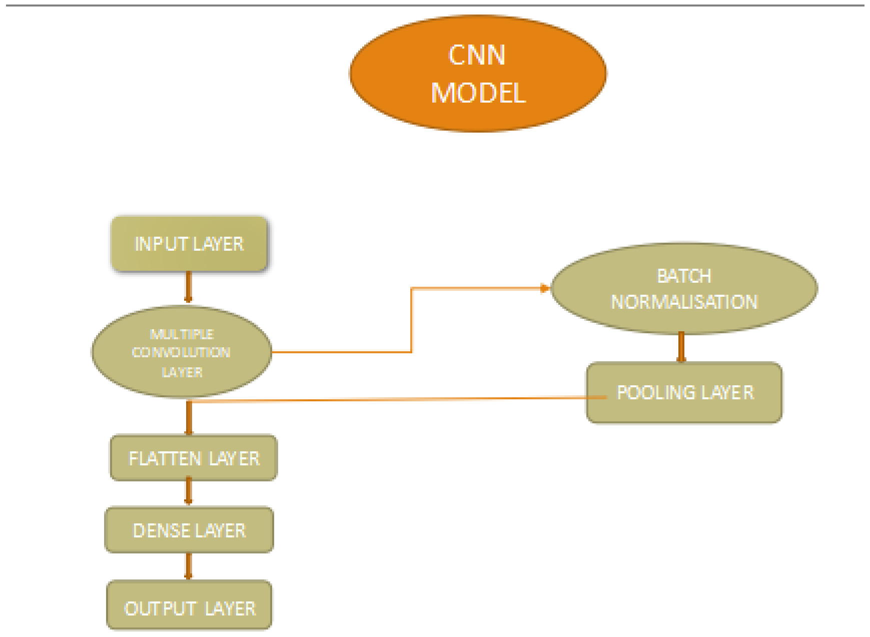
ADVANTAGE

This gives us two advantages: first, the code is as fast as the original C/C++ code (since it is the actual C++ code working in background) and second, it easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

Block diagram



FLOW DIAGRAM



Software Prerequisites

Keras (with TensorFlow backend)
OpenCV
Python 3.5 (TensorFlow not supported in higher versions)
NumPy
TensorFlow
. h5py (for Keras model serialization)

Hardware

Intel core processor with high GPU power & frequency

CHAPTER 4
DEEP LEARNING

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

How does deep learning attain such impressive results?

In a word, accuracy. Deep learning achieves recognition accuracy at higher levels than ever before. This helps consumer electronics meet user expectations, and it is crucial for safety-critical applications like driverless cars. Recent advances in deep learning have improved to the point where deep learning outperforms humans in some tasks like classifying objects in images.

While deep learning was first theorized in the 1980s, there are two main reasons it has only recently become useful:

1. Deep learning requires large amounts of **labeled data**. For example, driverless car development requires millions of images and thousands of hours of video.

2. Deep learning requires substantial **computing power**. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

Examples of Deep Learning at Work

Deep learning applications are used in industries from automated driving to medical devices.

Automated Driving: Automotive researchers are using deep learning to automatically detect objects such as stop signs and traffic lights. In addition, deep learning is used to detect pedestrians, which helps decrease accidents.

Aerospace and Defense: Deep learning is used to identify objects from satellites that locate areas of interest, and identify safe or unsafe zones for troops.

Medical Research: Cancer researchers are using deep learning to automatically detect cancer cells. Teams at UCLA built an advanced microscope that yields a high-dimensional data set used to train a deep learning application to accurately identify cancer cells.

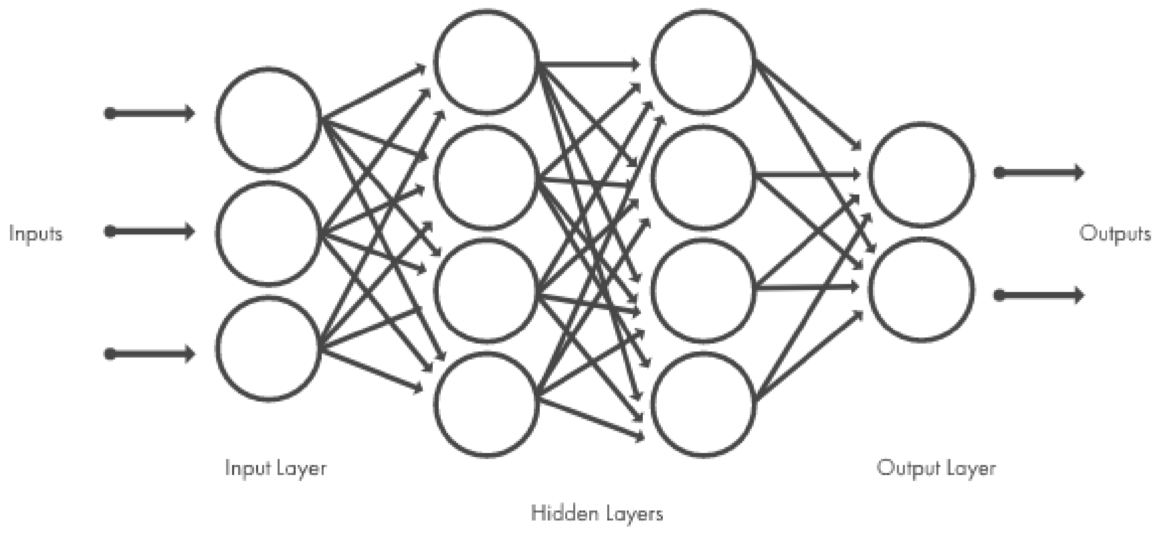
Industrial Automation: Deep learning is helping to improve worker safety around heavy machinery by automatically detecting when people or objects are within an unsafe distance of machines.

Electronics: Deep learning is being used in automated hearing and speech translation. For example, home assistance devices that respond to your voice and know your preferences are powered by deep learning applications.

Most deep learning methods use **neural network** architectures, which is why deep learning models are often referred to as **deep neural networks**.

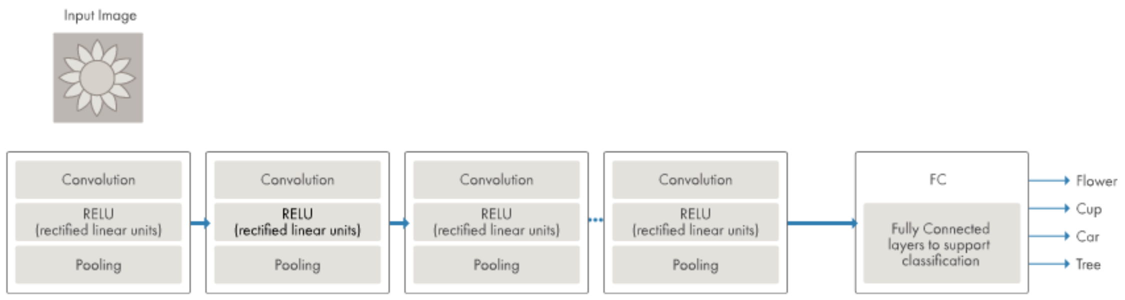
The term "deep" usually refers to the number of hidden layers in the neural network. Traditional neural networks only contain 2-3 hidden layers, while deep networks can have as many as 150.

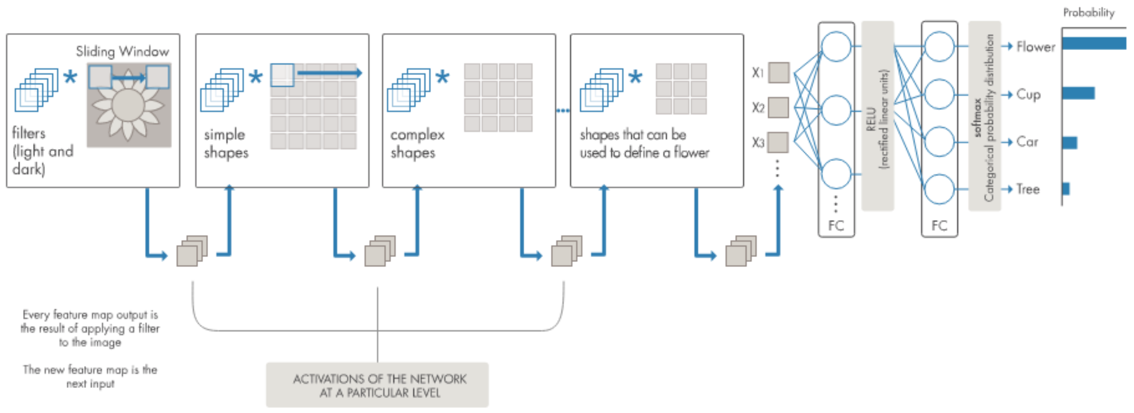
Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction.



One of the most popular types of deep neural networks is known as convolutional neural networks **(CNN** or **ConvNet)**. A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features directly from images. The relevant features are not pretrained; they are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification.


Input Image
Convolution
RELU (rectified linear units)
Pooling
Convolution
RELU (rectified linear units)
Pooling
Convolution
RELU (rectified linear units)
Pooling
Convolution
RELU (rectified linear units)
Pooling
FC
Fully Connected layers to support classification
Flower
Cup
Car
Tree


Sliding Window
filters (light and dark)
simple shapes
complex shapes
shapes that can be used to define a flower
X1
X2
X3
FC
RELU (rectified linear units)
FC
softmax
Categorical probability distribution
Probability
Flower
Cup
Car
Tree
Every feature map output is the result of applying a filter to the image
The new feature map is the next input
ACTIVATIONS OF THE NETWORK AT A PARTICULAR LEVEL

CNNs learn to detect different features of an image using tens or hundreds of hidden layers. Every hidden layer increases the complexity of the learned image features. For example, the first hidden layer could learn how to detect edges, and the last learns how to detect more complex shapes specifically catered to the shape of the object we are trying to recognize.

What's the Difference Between Machine Learning and Deep Learning?

Deep learning is a specialized form of machine learning. A machine learning workflow starts with relevant features being manually extracted from images. The features are then used to create a model that categorizes the objects in the image. With a deep learning workflow, relevant features are automatically extracted from images. In addition, deep learning performs “end-to-end learning” – where a network is given raw data and a task to perform, such as classification, and it learns how to do this automatically.

Another key difference is deep learning algorithms scale with data, whereas shallow learning converges. Shallow learning refers to machine learning methods that plateau at a certain level of performance when you add more examples and training data to the network.

A key advantage of deep learning networks is that they often continue to improve as the size of your data increases.

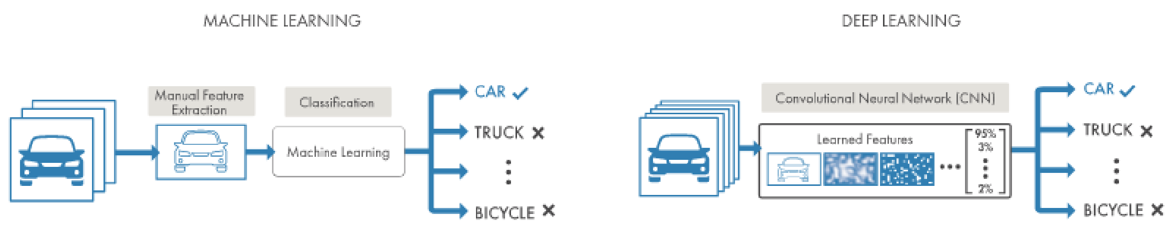


Figure 3. Comparing a machine learning approach to categorizing vehicles (left) with deep learning (right).

Choosing Between Machine Learning and Deep Learning

Machine learning offers a variety of techniques and models you can choose based on your application, the size of data you're processing, and the type of problem you want to solve. A successful deep learning application requires a very large amount of data (thousands of images) to train the model, as well as GPUs, or graphics processing units, to rapidly process your data.

When choosing between machine learning and deep learning, consider whether you have a high-performance GPU and lots of labeled data. If you don’t have either of those things, it may make more sense to use machine learning instead of deep learning. Deep learning is generally more complex, so you’ll need at least a few thousand images to get reliable results. Having a high-performance GPU means the model will take less time to analyze all those images.

How to Create and Train Deep Learning Models

The three most common ways people use deep learning to perform object classification are:

Training from Scratch

To train a deep network from scratch, you gather a very large labeled data set and design a network architecture that will learn the features and model. This is good for new applications, or applications that will have a large number of output categories. This is a less common approach because with the large amount of data and rate of learning, these networks typically take days or weeks to train

Transfer Learning

Most deep learning applications use the transfer learning approach, a process that involves fine-tuning a pretrained model. You start with an existing network, such as AlexNet or GoogLeNet, and feed in new data containing previously unknown classes.

After making some tweaks to the network, you can now perform a new task, such as categorizing only dogs or cats instead of 1000 different objects. This also has the advantage of needing much less data (processing thousands of images, rather than millions), so computation time drops to minutes or hours.

Transfer learning requires an interface to the internals of the pre-existing network, so it can be surgically modified and enhanced for the new task. MATLAB® has tools and functions designed to help you do transfer learning.

In machine learning, you manually choose features and a classifier to sort images. With deep learning, feature extraction and modeling steps are automatic.

Feature Extraction

A slightly less common, more specialized approach to deep learning is to use the network as a **feature extractor**. Since all the layers are tasked with learning certain features from images, we can pull these features out of the network at any time during the training process. These features can then be used as input to a machine learning model such as support vector machines (SVM).

Accelerating Deep Learning Models with GPUs

Training a deep learning model can take a long time, from days to weeks. Using GPU acceleration can speed up the process significantly. Using MATLAB with a GPU reduces the time required to train a network and can cut the training time for an image classification problem from days down to hours. In training deep learning models, MATLAB uses GPUs (when available) without requiring you to understand how to program

Chapter 5

Modules

- image data
- pre- processing
- segmentation image
- feature extraction
- data training and testing
- deep learning algorithm
- detection

Dataset collection

- Collecting data heavy use of collections of images called datasets. A dataset in computer vision is a curated set of digital photographs that developers use to test, train and evaluate the performance of their algorithms.
- Data can be gathered by different means like scraping from the web, gathering from third-party sources or you could even buy datasets from re-sellers etc.
- Autoecncoders work best for image data.
- Support file type filters.
- Support Bing.com filterui filters.
- Download using multithreading and custom thread pool size.
- Support purely obtaining the image URLs.

Data Cleaning

- Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.
- When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled.
- data cleaning or data scrubbing, is the process of fixing incorrect, incomplete, duplicate or otherwise erroneous data in a data set.
- It involves identifying data errors and then changing, updating or removing data to correct them.

Feature Extraction:

- Feature extraction is a part of the dimensionality reduction process, in which, an initial set of the raw data is divided and reduced to more manageable groups.
- So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables.

- These variables require a lot of computing resources to process. So Feature extraction helps to get the best feature from those big data sets by selecting and combining variables into features, thus, effectively reducing the amount of data.
- These features are easy to process, but still able to describe the actual data set with accuracy and originality.
- Image Processing –Image processing is one of the best and most interesting domain. In this domain basically you will start playing with your images in order to understand them.

Model training

- Plan and simplify. In the beginning we must think about how does the computer sees the images. ...
- Collect. For all the tasks try to get the most variable and diverse training dataset. ...
- Sort and upload. You have your images ready and it's time to sort them. ...
- Train and precise.
- Load and normalize the CIFAR10 training and test datasets using torchvision.
- Define a Convolutional Neural Network.
- Define a loss function.
- Train the network on the training data.
- Test the network on the test data.

Testing model:

- In this module we test the trained deep learning model using the test dataset
- A type of test that makes detailed pictures of areas inside the body. Imaging tests use different forms of energy, such as x-rays (high-energy radiation), ultrasound (high-energy sound waves), radio waves, and radioactive substances. They may be used to help diagnose disease, plan treatment, or find out how well treatment is working.
- Examples of imaging tests are computed tomography (CT), mammography, ultrasonography, magnetic resonance imaging (MRI), and nuclear medicine tests. Also called imaging procedure

Performance Evaluation

- In this module, we evaluate the performance of trained deep learning model using performance evaluation criteria such as F1 score, accuracy and classification error.
- To evaluate object detection models like R-CNN and YOLO, the mean average precision (mAP) is used. The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections.
- Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses.

- Model evaluation is important to assess the efficacy of a model during initial research phases, and it also plays a role in model monitoring.

Detection

- Object detection is a process of finding all the possible instances of real-world objects, such as human faces, flowers, cars, etc. in images or videos, in real-time with utmost accuracy.
- The object detection technique uses derived features and learning algorithms to recognize all the occurrences of an object category.
- First, we take an image as input
- Then we divide the image into various regions
- We will then consider each region as a separate image.
- Pass all these regions (images) to the CNN and classify them into various classes.

STEPS FOR CNN

Step 1: Choose a Dataset. ...
Step 2: Prepare Dataset for Training. ...
Step 3: Create Training Data. ...
Step 4: Shuffle the Dataset. ...
Step 5: Assigning Labels and Features. ...
Step 6: Normalising X and converting labels to categorical data. ...
Step 7: Split X and Y for use in CNN.

Open cv:

- OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.
- OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel).
- OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. ..
- . Some of these functions are really common and are used in almost every computer vision task.

Output:
In the Output phase, we apply the same feature extraction process to the new images and we pass the features to the trained machine learning algorithm to predict the label.

WORK FLOW

- The project has to show a final product as a website that accepts your input data as a picture and then tells us age and gender.

- The website will be an instantaneous website that will convert the information right away, yet cannot store or reproduce the same information again. Just acts as an end-to-end volatile conversion interface.

- Following the conversions, the user can also avail some other functions in the website.

CHAPTER 6

APPENDIX:
<u>Source Code</u>

```
import cv2
import numpy as np
import math
import argparse
from flask import Flask, render_template, Response, request
from PIL import Image
import io

UPLOAD_FOLDER = './UPLOAD_FOLDER'
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER


def highlightFace(net, frame, conf_threshold=0.7):
    frameOpencvDnn = frame.copy()
    frameHeight = frameOpencvDnn.shape[0]
    frameWidth = frameOpencvDnn.shape[1]
    # Grab the frame dimensions and convert it to a blob.
    blob = cv2.dnn.blobFromImage(frameOpencvDnn, 1.0, (300, 300), [
                               104, 117, 123], True, False)
    # Pass the blob through the network and obtain the detections and predictions.
    net.setInput(blob)
    # net.forward() method detects the faces and stores the data in detections
    detections = net.forward()

    faceBoxes = []

    # This for loop is for drawing rectangle on detected face.
    for i in range(detections.shape[2]):    # Looping over the detections.
        # Extract the confidence (i.e., probability) associated with the prediction.
        confidence = detections[0, 0, i, 2]
        if confidence > conf_threshold:   # Compare it to the confidence threshold.
            # Compute the (x, y)-coordinates of the bounding box for the face.
            x1 = int(detections[0, 0, i, 3]*frameWidth)
            y1 = int(detections[0, 0, i, 4]*frameHeight)
            x2 = int(detections[0, 0, i, 5]*frameWidth)
            y2 = int(detections[0, 0, i, 6]*frameHeight)
            # Drawing the bounding box of the face.
            faceBoxes.append([x1, y1, x2, y2])
            cv2.rectangle(frameOpencvDnn, (x1, y1), (x2, y2),
                          (0, 255, 0), int(round(frameHeight/150)), 8)
    return frameOpencvDnn, faceBoxes


# Gives input img to the prg for detection.
# Using argparse library which was imported.
parser = argparse.ArgumentParser()
# If the input argument is not given it will skip this and open webcam for detection
parser.add_argument('--image')

args = parser.parse_args()

'''
Each model comes with two files: weight file and model file
weight file stores the data of the deployment of the model
model file stores actual predication done by the model
We are using pre trained models

The .prototxt file(s) which define the model architecture (i.e., the layers themselves)
The .caffemodel file which contains the weights for the actual layers
Both files are required when using models trained using Caffe for deep learning.
'''

def gen_frames():
    faceProto = "opencv_face_detector.pbtxt"
    faceModel = "opencv_face_detector_uint8.pb"
    ageProto = "age_deploy.prototxt"
    ageModel = "age_net.caffemodel"
    genderProto = "gender_deploy.prototxt"
    genderModel = "gender_net.caffemodel"

    MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
    # Defining age range.
    ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)',
               '(25-32)', '(38-43)', '(48-53)', '(60-100)']
    genderList = ['Male', 'Female']

    # LOAD NETWORK
    faceNet = cv2.dnn.readNet(faceModel, faceProto)
    ageNet = cv2.dnn.readNet(ageModel, ageProto)
    genderNet = cv2.dnn.readNet(genderModel, genderProto)

# Open a video file or an image file or a camera stream
    video = cv2.VideoCapture(0)
    padding = 20
    while cv2.waitKey(1) < 0:
        # Read frame
        hasFrame, frame = video.read()
        if not hasFrame:
            cv2.waitKey()
            break

    # It will detect the no. of faces in the frame
        resultImg, faceBoxes = highlightFace(faceNet, frame)
        if not faceBoxes:   # If no faces are detected
            print("No face detected")   # Then it will print this message

        for faceBox in faceBoxes:
            # print facebox
            face = frame[max(0, faceBox[1]-padding):   # Face info is stored in this variable
                         min(faceBox[3]+padding, frame.shape[0]-1), max(0,
faceBox[0]-padding):min(faceBox[2]+padding, frame.shape[1]-1)]

        # The dnn.blobFromImage takes care of pre-processing
        # which includes setting the blob  dimensions and normalization.
            blob = cv2.dnn.blobFromImage(
                face, 1.0, (227, 227), MODEL_MEAN_VALUES, swapRB=False)
            genderNet.setInput(blob)
        # genderNet.forward method will detect the gender of each face detected
            genderPreds = genderNet.forward()
            gender = genderList[genderPreds[0].argmax()]
            print(f'Gender: {gender}')  # print the gender in the console

            ageNet.setInput(blob)
        # ageNet.forward method will detect the age of the face detected
            agePreds = ageNet.forward()
            age = ageList[agePreds[0].argmax()]
            print(f'Age: {age[1:-1]} years')    # print the age in the console

        # Show the output frame
            cv2.putText(resultImg, f'{gender}, {age}', (
                faceBox[0], faceBox[1]-10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 255), 2, cv2.LINE_AA)
        #cv2.imshow("Detecting age and gender", resultImg)

            if resultImg is None:
                continue

            ret, encodedImg = cv2.imencode('.jpg', resultImg)
            #resultImg = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: image/jpeg\r\n\r\n' + bytearray(encodedImg) + b'\r\n')


def gen_frames_photo(img_file):
    faceProto = "opencv_face_detector.pbtxt"
    faceModel = "opencv_face_detector_uint8.pb"
    ageProto = "age_deploy.prototxt"
    ageModel = "age_net.caffemodel"
    genderProto = "gender_deploy.prototxt"
    genderModel = "gender_net.caffemodel"

    MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
    # Defining age range.
    ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)',
               '(25-32)', '(38-43)', '(48-53)', '(60-100)']
    genderList = ['Male', 'Female']

    # LOAD NETWORK
    faceNet = cv2.dnn.readNet(faceModel, faceProto)
    ageNet = cv2.dnn.readNet(ageModel, ageProto)
    genderNet = cv2.dnn.readNet(genderModel, genderProto)

# Open a video file or an image file or a camera stream

    frame = cv2.cvtColor(img_file, cv2.COLOR_BGR2RGB)
    #frame = img_file
    #hasFrame, frame = img_file.read()
    #ret, frame = cv2.imencode('.jpg', img_file)
    #video = cv2.VideoCapture(img_file)
    padding = 20
    while cv2.waitKey(1) < 0:
        # Read frame
        #hasFrame, frame = video.read()
        # if not hasFrame:
        # cv2.waitKey()
        # break

        # It will detect the no. of faces in the frame
        resultImg, faceBoxes = highlightFace(faceNet, frame)
        if not faceBoxes:   # If no faces are detected
            print("No face detected")   # Then it will print this message

        for faceBox in faceBoxes:
            # print facebox
            face = frame[max(0, faceBox[1]-padding):   # Face info is stored in this variable
                         min(faceBox[3]+padding, frame.shape[0]-1), max(0,
faceBox[0]-padding):min(faceBox[2]+padding, frame.shape[1]-1)]

        # The dnn.blobFromImage takes care of pre-processing
        # which includes setting the blob  dimensions and normalization.
            blob = cv2.dnn.blobFromImage(
                face, 1.0, (227, 227), MODEL_MEAN_VALUES, swapRB=False)
            genderNet.setInput(blob)
        # genderNet.forward method will detect the gender of each face detected
            genderPreds = genderNet.forward()
            gender = genderList[genderPreds[0].argmax()]
            print(f'Gender: {gender}')  # print the gender in the console

            ageNet.setInput(blob)
        # ageNet.forward method will detect the age of the face detected
            agePreds = ageNet.forward()
            age = ageList[agePreds[0].argmax()]
            print(f'Age: {age[1:-1]} years')    # print the age in the console

        # Show the output frame
            cv2.putText(resultImg, f'{gender}, {age}', (
                faceBox[0], faceBox[1]-10), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 255), 2, cv2.LINE_AA)
        #cv2.imshow("Detecting age and gender", resultImg)

            if resultImg is None:
                continue

            ret, encodedImg = cv2.imencode('.jpg', resultImg)
            #resultImg = buffer.tobytes()
            return (b'--frame\r\n'
                    b'Content-Type: image/jpeg\r\n\r\n' + bytearray(encodedImg) + b'\r\n')


@app.route('/')
def index():
    """Video streaming home page."""
    return render_template('index.html')

@app.route('/video_feed')
def video_feed():
    # Video streaming route. Put this in the src attribute of an img tag
    return Response(gen_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')

@app.route('/webcam')
def webcam():
    return render_template('webcam.html')

@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['fileToUpload'].read()
        img = Image.open(io.BytesIO(f))
        img_ip = np.asarray(img, dtype="uint8")
        print(img_ip)
        return Response(gen_frames_photo(img_ip), mimetype='multipart/x-mixed-replace;
boundary=frame')
        # return 'file uploaded successfully'

if __name__ == '__main__':
    app.run(debug=True)
```

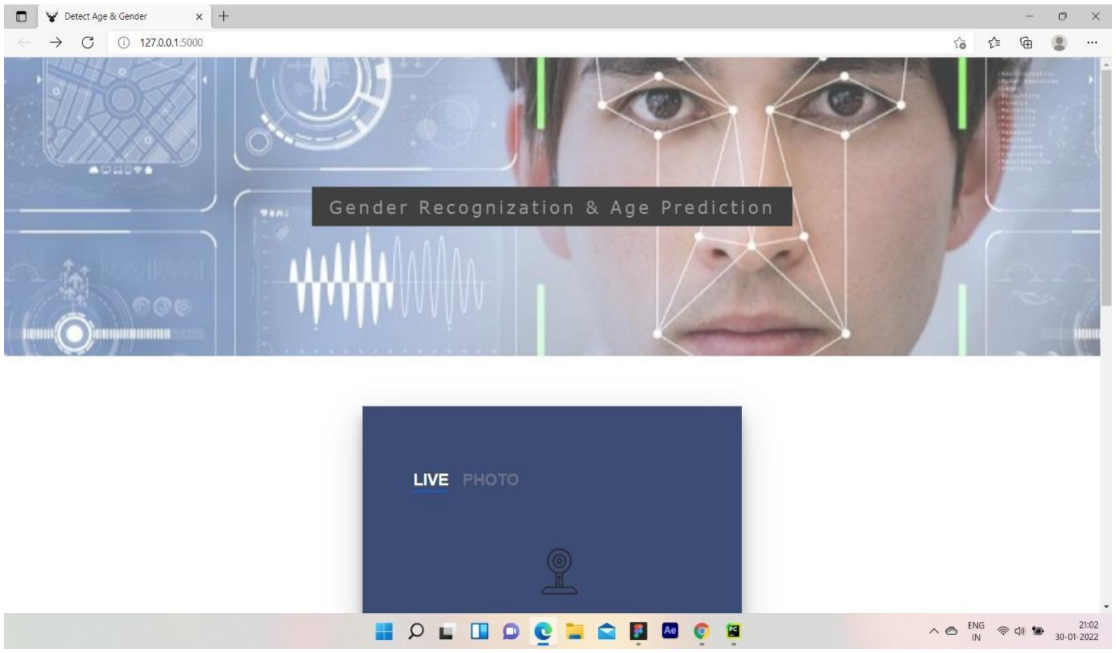
HTML CODE

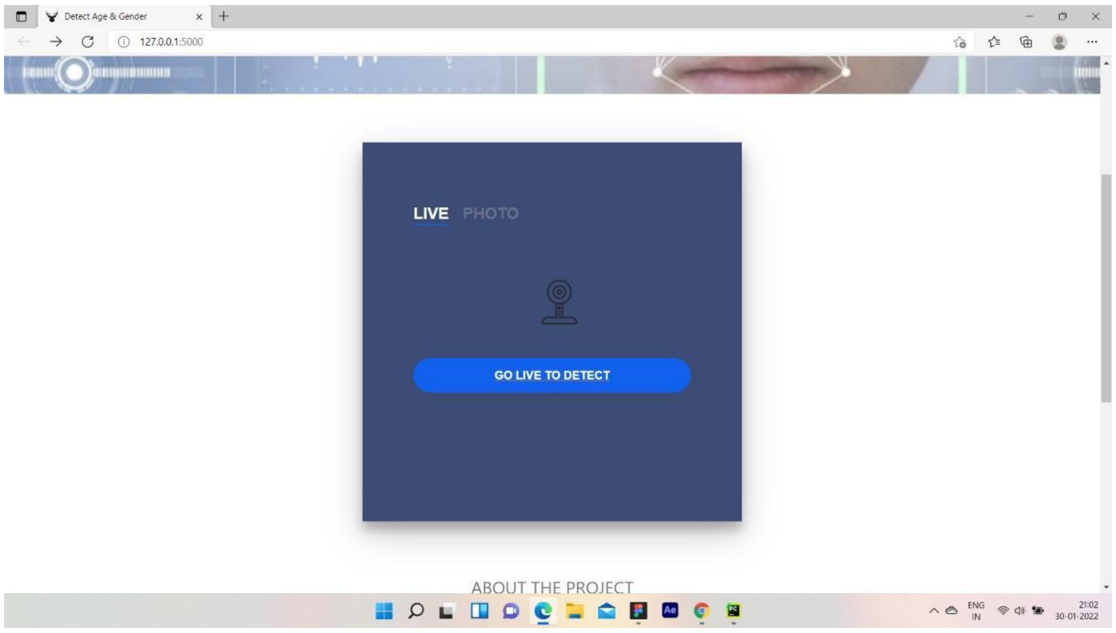
```
<!DOCTYPE html>
<html>
<title>Detect Age & Gender</title>
<link rel = "icon" href =
"https://cdn.pixabay.com/photo/2019/06/23/05/32/deer-head-4292868_1280.png" type =
"image/x-icon">
<meta charset="UTF-8">
```

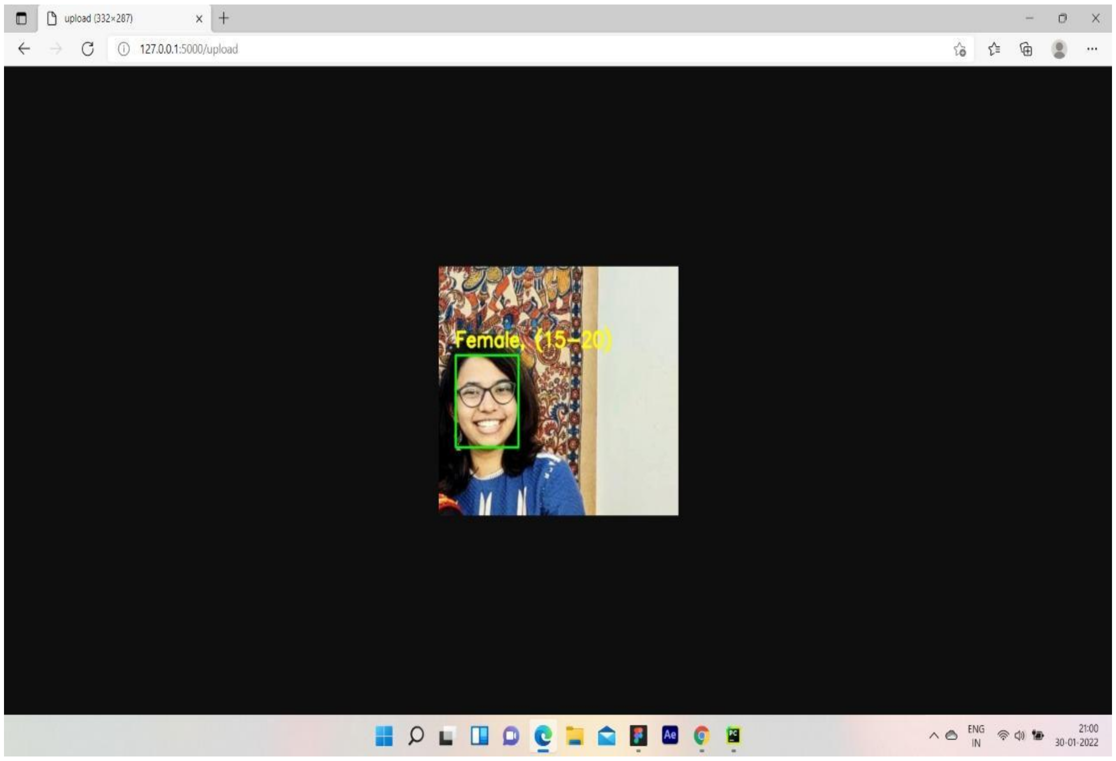
```
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="../static/styles/style.css">
<link rel="stylesheet" href="../static/styles/aj.css">
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- First Parallax Image with Logo Text -->
<div class="bgimg aj-display-container aj-opacity-min" id="home">
  <div class="aj-display-middle" style="white-space:nowrap;">
    <span class="aj-center aj-padding-large aj-black aj-xlarge aj-wide aj-animate-opacity">Gender<span
class="aj-hide-small"> Recognization</span> & Age Prediction</span>
  </div>
</div>
<br><br><br>
<div class="action-wrap">
  <div class="action-html">
    <input id="tab-1" type="radio" name="tab" class="live" checked><label for="tab-1"
class="tab">Live</label>
    <input id="tab-2" type="radio" name="tab" class="photo"><label for="tab-2"
class="tab">Photo</label>
    <div class="action-form">
      <div class="live-htm">
        <div class="group">
          <br><br>

&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbs
p;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&n
bsp;&nbsp;
          <img src="https://cdn.pixabay.com/photo/2021/03/23/09/01/webcam-6116845_1280.png"
height="120" width="120">
          <br><br>
          <a href='/webcam'><input type="submit" class="button" value="Go live to detect"></a>
        </div>
      </div>
      <div class="photo-htm">
        <form action="/upload" method="POST" enctype="multipart/form-data" style="color: #aaa">
          <br><br>Select image to upload:<br><br><br>
          <input type="file" name="fileToUpload" id="fileToUpload">
          <br><br><br><br>
        <div class="group">
          <input type="submit" class="button" value="Upload and detect" name="submit">
        </div>
        </form>
      </div>
    </div>
  </div>
</div>

<!-- Container (About Section) -->
<div class="aj-content aj-container aj-padding-64" id="about">
```

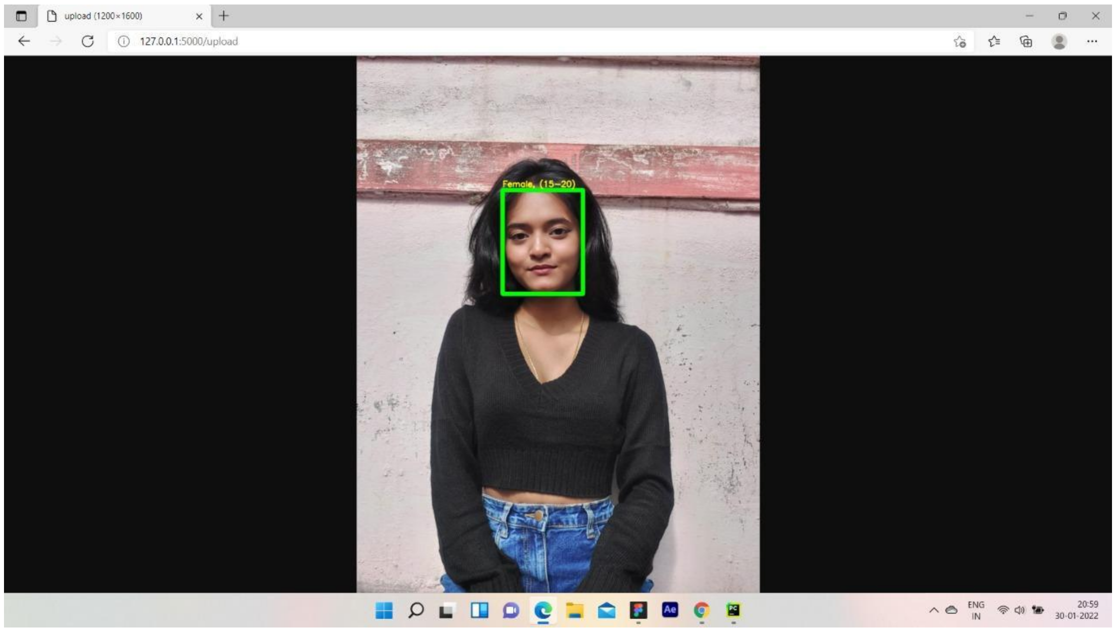
```
 <h3 class="aj-center">ABOUT THE PROJECT</h3>
 <p class="aj-center"><em>ML Project</em></p>
 <p>We will use Deep Learning to accurately identify the gender and age of a person from a single image
of a face.</p>
 </div>


<!-- Footer -->
<footer class="aj-center aj-black aj-padding-64 aj-opacity aj-hover-opacity-off">
 <a href="#home" class="aj-button aj-light-grey"><i class="fa fa-arrow-up aj-margin-right"></i>To the
top</a>
 <div class="aj-section" style="text-align: left; padding-left: 10px">
  <p>Project by :</p>
  <div class="hr"></div>
  <table>
   <tr>
    <th><p>Anchitya</th>
    <th>
     <a href=""><i class="fa fa-facebook-official aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-instagram aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-snapchat aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-twitter aj-hover-opacity"></i></a>
     <a href="https://www.linkedin.com/in/anchitya/"><i class="fa fa-linkedin
aj-hover-opacity"></i></a></th>
    </p>
   </tr>
   <tr>
    <th><p>Rhys Rahman</th>
    <th>
     <a href="https://www.facebook.com/rhysrocks200"><i class="fa fa-facebook-official
aj-hover-opacity"></i></a>
     <a href="https://www.instagram.com/rhys_tris"><i class="fa fa-instagram
aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-snapchat aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-twitter aj-hover-opacity"></i></a>
     <a href="https://www.linkedin.com/in/rhys-rahman-775051189/"><i class="fa fa-linkedin
aj-hover-opacity"></i></a>
    </th>
    </p>
   </tr>
   <tr>
    <th><p>Shaadmaan Adil Raza</th>
    <th>
     <a href=""><i class="fa fa-facebook-official aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-instagram aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-snapchat aj-hover-opacity"></i></a>
     <a href=""><i class="fa fa-twitter aj-hover-opacity"></i></a>
     <a href="https://www.linkedin.com/in/shaadmaan-adil-raza-48545916b/"><i class="fa fa-linkedin
aj-hover-opacity"></i></a>
    </th>
    </p>
   </tr>
  </table>
  </div>
  <div style="text-align: right; padding-right: 10px">
    <p>Guided by : Ms. Saswati Bhattacharya</p>
  </div>
</footer>

<script>
// Modal Image Gallery
function onClick(element) {
  document.getElementById("img01").src = element.src;
  document.getElementById("modal01").style.display = "block";
  var captionText = document.getElementById("caption");
  captionText.innerHTML = element.alt;
}

</script>

</body>
</html>


<!doctype html>
<html lang="en">
<head>
    <link rel = "icon" href =
"https://cdn.pixabay.com/photo/2021/03/23/09/01/webcam-6116845_1280.png" type = "image/x-icon">
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="static/css/style.css">

    <title>Photo Detection</title>
    <style>
        html {
            background: #100a1c;
            background-image:
                radial-gradient(50% 30% ellipse at center top, #201e40 0%, rgba(0,0,0,0) 100%),
                radial-gradient(60% 50% ellipse at center bottom, #261226 0%, #100a1c 100%);
            background-attachment: fixed;
            color: #6cacc5;
        }
        img {
            margin-left: 110px;
            padding: 0;
            width: 80vw;
            height: 80vh;
            border: 2px solid #6cacc5;
            border-radius: 4px;
        }
        button {
            float: left;
        }
        h2 {
            text-align: center;
            font-family: 'EB Garamond', serif;
           /*color: #ff1a1a;*/
            text-shadow: 2px 2px 4px #000000;
        }
        /* --- STYLING THE BUTTONS --- */
        button {
            border: 0;
            background: rgba(42,50,113, .28);
            color: #6cacc5;
            cursor: pointer;
            font: inherit;
            margin: 0.25em;
            transition: all 0.5s;
            border-radius: 4px;
        }
        /* --- WHEN THE CURSOR HOVERS OVER THE BUTTONS THE COLOR CHNAGES --- */
        button:hover {
            background: #201e40;
        }
    </style>
</head>
<body>
    <div>
        <div class="header">
            <a href="/"><button>Home</button></a>
            <h2>Detecting Age and Gender from Photo</h2>
        </div>

        <div class="container">
            <img src="{{ url_for('upload_file') }}" width="100%">
        </div>
    </div>
</body>
</html>


<!doctype html>
<html lang="en">
<head>
    <link rel = "icon" href =
"https://cdn.pixabay.com/photo/2021/03/23/09/01/webcam-6116845_1280.png" type = "image/x-icon">
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="static/css/style.css">

    <title>Live Detection</title>
    <style>
        html {
            background: #100a1c;
            background-image:
                radial-gradient(50% 30% ellipse at center top, #201e40 0%, rgba(0,0,0,0) 100%),
                radial-gradient(60% 50% ellipse at center bottom, #261226 0%, #100a1c 100%);
            background-attachment: fixed;
            color: #6cacc5;
        }
        img {
            margin-left: 110px;
            padding: 0;
            width: 80vw;
            height: 80vh;
            border: 2px solid #6cacc5;
            border-radius: 4px;
        }
        button {
            float: left;
        }
        h2 {
            text-align: center;
            font-family: 'EB Garamond', serif;
           /*color: #ff1a1a;*/
            text-shadow: 2px 2px 4px #000000;
        }
        /* --- STYLING THE BUTTONS --- */
        button {
            border: 0;
            background: rgba(42,50,113, .28);
            color: #6cacc5;
            cursor: pointer;
            font: inherit;
            margin: 0.25em;
            transition: all 0.5s;
            border-radius: 4px;
        }
        /* --- WHEN THE CURSOR HOVERS OVER THE BUTTONS THE COLOR CHNAGES --- */
        button:hover {
            background: #201e40;
        }
    </style>
</head>
<body>
    <div>
        <div class="header">
            <a href="/"><button>Home</button></a>
            <h2>Detecting Age and Gender</h2>
        </div>

        <div class="container">
            <img src="{{ url_for('video_feed') }}" width="100%">
        </div>
    </div>
</body>
</html>
```

OUTPUT:


Detect Age & Gender
127.0.0.1:5000
Gender Recognization & Age Prediction
LIVE
PHOTO


Detect Age & Gender
127.0.0.1:5000
LIVE
PHOTO
GO LIVE TO DETECT
ABOUT THE PROJECT





CONCLUSION:

- The task of recognizing age and gender, nonetheless, is an innately troublesome issue, more so than numerous other PC vision undertakings.

- The fundamental justification for this trouble hole lies in the information needed to prepare these kinds of frameworks.
- Python obtained images and the Model did not do well much in the accuracy rate, further, improvement is required in the model algorithm.

REFRENCES

- Aurélien Géron (2019). Hands-on Machine Learning with Scikit- Learn, Keras, and TensorFlow: Second Edition.
- Hisham, A., Harin, S. (2017). Deep Learning – the new kid in Artificial Intelligence
- Robin Nixon (2014). Learning PHP, MySQL, JavaScript, CSS & HTML5: A Step-by-Step Guide to Creating Dynamic
- Choi, S.E.; Lee, Y.J.; Lee, S.J.; Park, K.R.; Kim, J. Age Estimation Using a Hierarchical Classifier Based on Global and Local Facial Features. Pattern Recognition
- Ricanek, K.; Tesafaye, T. Morph: A Longitudinal Image Database of Normal Adult Age-Progression. In Proceedings of the Seventh International