Masaryk University
Faculty of Informatics

Gender and Age Classification in Camera Data

Master's Thesis

Bc. Matúš Námešný

Brno, Spring 2019

Masaryk University
Faculty of Informatics

Gender and Age Classification in Camera Data

Master's Thesis

Bc. Matúš Námešný

Brno, Spring 2019

This is where a copy of the official signed thesis assignment and a copy of the Statement of an Author is located in the printed version of the document.

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Bc. Matúš Námešný

Advisor: doc. RNDr. Aleš Horák Ph.D.

Acknowledgements

I would like to thank my advisor doc. RNDr. Aleš Horák Ph.D. for his advice and guidence. I would also like to thank Pavel Dvořák for his patience and advice. I would also like to acknowledge Konica Minolta and Metacentrum VO for providing necessary computational resources.

Abstract

Age and gender prediction from images is an important application of computer vision. There are many approaches to solve this problem. We evaluate three different methods. We combine publicly available datasets and one manually labelled dataset into a large set and train the best method. We further extend the data by adding a colour channel to the images and train the best method. We show that training a network with a large dataset improves the performance, however adding additional colour channel does not. Based on our results we develop an application for age and gender classification.

Keywords

machine learning, computer vision, classification, image analysis

Contents

List of Tables

List of Figures

Introduction

Deeper understanding of how biological neurons in the visual cortex of the brain work enabled the inception of the field of computer vision. The important takeaway from biology is that processing of visual input starts with recognition of simple shapes such as edges and then moves to more complex structures. During the early years of computer vision the main focus was on extracting 3 dimensional shapes from 2 dimensional images.

The field took a big step forward when Japanese researcher Kunihiko Fukushima invented Neocognitron[1], a multilayered artificial neural network, which contained convolutional layers. Then in 1989, scientist Yann LeCun applied backpropagation algorithm to Neocognitron and released the first convolutional neural network LeNet-5[1]

The next breakthrough came with the introduction of AlexNet by Krizhevsky et al. [2]. AlexNet is an 8 layer convolutional neural network which contains neurons with non-saturating ReLU activation. The network won the 2012 ImageNet Large Scale Visual Recognition Challenge achieving top-5 error of 15.3%. AlexNet is considered the most influential neural network architecture.

Many attributes can be predicted from an image of a face, such as age, gender, expression, ethnicity or emotion. One of the most significant attributes are age and gender. It is easy to imagine the possible applications, from human-computer interaction to marketing to security systems.

There are various methods to solve the problem of accurately estimating age and gender. The first approach relies on manually extracting features such as the size of the head, position of eyes or length of the nose. Another approach is based on end-to-end deep learning models. The two methods can be combined to form a new mixed approach. Most modern methods use deep learning approach.

To train age and gender models it is necessary to have labelled datasets. There are many publicly available datasets with age or gender annotations. Another option is to create a new dataset by manually labelling face images.

---

1. http://yann.lecun.com/exdb/lenet/

The motivation behind this thesis was to build an application for age and gender classification using a model that is suitable for real life predictions. Many models are focusing on datasets with constrained faces and are not suitable for in-the-wild estimation. In this thesis we will focus on deep learning end-to-end methods. We will evaluate three different deep learning models.

In Chapter 1 we describe current methods for face detection and face landmark detection. We also describe the datasets used for evaluation. In Chapter 2 we discuss our selection of method implementations for evaluation. Then in Chapter 3 we detail the chosen models and evaluate them. In Chapter 4 we discuss the implementation details of an application for age and gender estimation. In Chapter 5 we train one model on combined dataset and evaluate it. Finally in Chapter 6 we document the application for age and gender classification.

1 Age and gender prediction methods

Predicting age and gender from a picture involves several steps. First we need to detect and extract face from the image. Optionally we can recognise face features such as eyes, mouth and nose and using those we can align the detected face. Then we pass the detected and aligned face through our model to get the prediction.

1.1 Face detection

In computer vision face detection identifies human faces on images. There are numerous approaches for unconstrained face detection. Some earlier methods rely on detecting edges of the face.

The most commonly used algorithm was first presented by Viola and Jones [3]. The algorithm divides the input image into rectangular sections. Each section is then passed through a cascade of weak classifiers that are detecting the presence of simple haar-like features. Haar-like feature is the difference between the sum of pixel intensities in different adjacent regions. If the section passes through all stages of the cascade it is classified as containing face, otherwise it is rejected. This process is repeated for different sizes of rectangular sections.

The classifier is trained using AdaBoost. The main advantage of the algorithm is that due to the use of integral images the time needed to calculate haar-like feature is constant. An integral image, or summed-area table, is a data structure where the value of each cell is the sum of cells above and to the left. The summed-area table can be computed in a single pass.

Haar cascade face detection is used in many popular computer vision libraries such as OpenCV[1]

A different approach uses HOG or histogram of oriented gradients feature descriptor. It was invented by authors Dalal and Triggs [4] originally for person detection. The algorithm computes the gradient of pixel intensities for each pixel of image. Next the image is split into smaller regions. In each region a histogram of the gradients is created. Then only the most prominent gradient is saved. Finally the resulting

1. https://opencv.org/

HOG image is classified using Support Vector Machines. The popular software library Dlib[2] uses HOG based face detection.

A deep learning algorithm for face detection uses Multitask Cascaded Convolutional Neural Networks (MTCNN), first introduced by Zhang K. et al. [5]. The algorithm detects face and facial landmarks in one pass. It uses a cascade of convolutional neural networks.

First, the input image is resized to form an image pyramid. An image pyramid is a multi-scale representation of an image. It enables finding objects in images at different scales. This is often combined with sliding window which can find objects in different locations.

The image pyramid is then fed to the first convolutional neural network called Proposal Network (P-Net). This network finds candidate bounding boxes. The candidates are the input to the next network called Refine Network (R-Net), which filters out false positives. During both stages overlapping bounding boxes are merged using non-maximum suppression.

Non-maximum suppression is a method used in edge detection to eliminate redundant edges. It suppresses all gradient values except the local maxima.

Last stage, the Output Network (O-Net) further refines the candidates and performs simple facial landmarks localisation. Compared to other facial landmark detection techniques, the MTCNN will only find 5 landmarks: both eyes, both mouth corners and the tip of the nose.

1.2 Facial Landmarks Detection

Detecting facial landmarks is a crucial step for face alignment. Facial landmarks are used to localise facial structures such as nose, eyebrows, eyes, mouth and jaw. The Dlib library uses an implementation of a very fast algorithm invented by Kazemi et al. [6]. It uses a cascade of regression trees, where each tree updates a vector of facial landmark coordinates. Feature selection at each node of the tree is based on the difference of pixel intensities.

An alternative method which uses deep learning was presented by Bulat and Tzimiropoulos [7]. They introduce a new neural network

2. http://dlib.net/

called simply Face Alignment Network (FAN). The network consists of four stacked "Hourglass networks" with an updated bottleneck block.

The Hourglass network was invented by Newell et al. [8]. It is intended for human pose estimation. One Hourglass network first scales down the image to a very low resolution and then combining the features across different resolution the image is up-sampled.

1.3 Age Prediction

In the past, the problem of age prediction has been studied as a sub-problem of facial ageing. Many approaches developed in facial ageing research are directly transferable to the problem of age classification.

One of the first research into age classification was done by authors Kwon and Lobo [9]. In their proposed approach they first find an initial oval of the face and eyes. Then using the boundaries of the initial oval, they find the chin and the sides of the face. After finding all facial features various ratios are computed. Using the computed ratios the face is then classified. The approach also uses the presence of wrinkles in an area of face to infer the age.

More recently authors Rothe et al. [10][11] proposed a method they call Deep EXpectation (DEX). The method starts by rotating images with steps of 5°, detecting faces and selecting the image with the highest face score. This procedure eliminates the need for facial landmarks detection. Additionally the DEX method leaves a 40% margin around the detected face. The detected face is then fed into convolutional neural network. The method achieved mean absolute error of 3.221 on IMDB-WIKI dataset.

A different approach would be to handle age estimation as classification problem. It is often not necessary to know the exact age of a person and being able to predict the age group is sufficient. A method introduced by Eidinger et al. [12] uses a dropout Support Vector Machine to classify faces into age groups. First they localise facial landmarks on the detected face. Using the coordinates of landmarks they apply an affine transformation to align the face. The aligned face is classified using linear Support Vector Machine. To prevent overfitting they propose using dropout similar to dropout used in neural networks. Instead of randomly omitting neurons, their method

randomly sets some features to zero. On the Adience dataset they achieved 45.1% exact accuracy and 80.7% one-off accuracy.

Aother classification method was introduced by Levi and Hassner [13]. To classify age into 8 age groups it uses convolutional network.It consists of only 3 convolutional layers and 2 fully connected layers. They argue that such a small network will reduce the risk of overfitting. The image is first resized to 256×256 pixels. Then a center crop of 227×227 pixels is fed to the network. Using the Adience dataset this method achieved 50.7% exact accuracy and 84.7% one-off accuracy. This method also addresses the problem of gender classification where they achieved accuracy of 86.8%.

Many real world applications such as embedded systems require a more efficient neural network architectures. Authors Yang et al. [14] introduce a network architecture called Soft Stagewise Regression Network (SSR-Net). The method classifies age in stages where each stage refines the classification made by previous stage. The best result this method achieved was mean absolute error of 3.16 using model trained on IMDB-WIKI dataset and evaluated on MORPH2 dataset.

1.4 Gender Prediction

The problem of gender prediction and classification is closely related to face detection.

One of the first method for gender classification was SEXNET introduced by Golomb et al. [15]. It classified images sampled at 30 × 30 pixels using a fully connected neural network with 3 layers of 900, 40 and 900 neurons.

Another method introduced by Moghaddam and Yang [16] uses Support Vector Machines to classify gender. The method used subsampled images of 21 × 21 pixels and classifed them using SVM with radial basis function kernel.

More recently the gender classification problem has been treated as a subproblem of age prediction. Current approaches use the same neural network architecture for age as well as gender prediction. Gender can be easily modelled as a binary classification problem. If the gender is modelled using regression we can view the output number as confidence.

1.5 Datasets

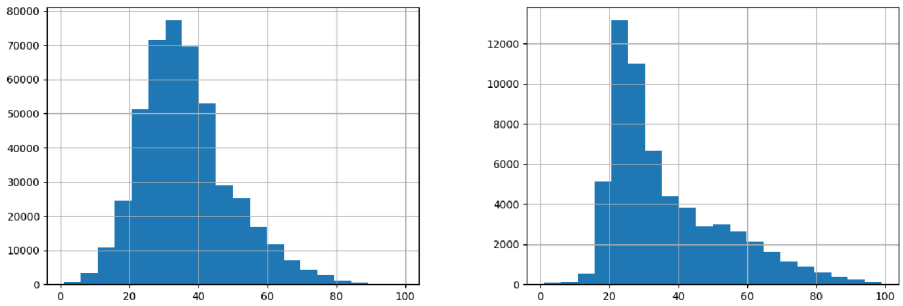
There are various publicly available datasets suitable for age and gender estimation. A comprehensive comparison of datasets is shown in Table 1.2.

1.5.1 IMDB-WIKI Dataset

IMDB-WIKI is the largest publicly available dataset with age and gender labels. It was created for the evaluation of the DEX (Deep EXpectation) method, introduced by Rothe et al.[10][11].

The dataset was created by automatically by crawling IMDb (Internet Movie Database) and Wikipedia. The authors crawled the most popular actors on IMDb getting date of birth and gender along with all the images associated with the actor. Each image contained the date, when it was taken, in the metadata. They crawled 20,284 celebrities and got 460,723 labelled images, 57% of which are male faces.

Similarly they crawled Wikipedia pages of famous people and got the same data. The Wikipedia subset contains 62,328 images, 75% of which are male faces. The total number of images in the IMDB-WIKI dataset is 523,051. Figure 1.1 shows histogram of ages in the IMDB and WIKI subsets.

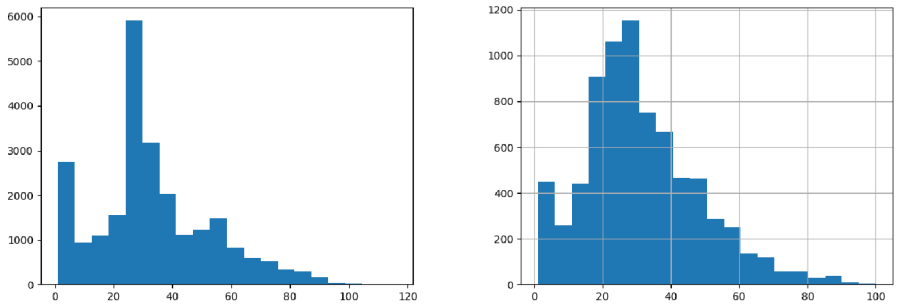


(a) IMDB dataset age distribution (b) WIKI dataset age distribution

Figure 1.1: Histogram of ages in IMDB-WIKI dataset

1.5.2 UTKFace Dataset

The UTKFace dataset contains 24,107 faces with ages ranging from 0 to 116. 52% of images show male subject. Each image is annotated by age, gender and ethnicity. Additionaly the dataset contains 68 landmarks for each face. On Figure 1.2a we can see the distribution of ages in the UTKface dataset.



(a) UTKFace dataset (b) APPA-REAL dataset

Figure 1.2: Histogram of ages in UTKFace and APPA-REAL datasets

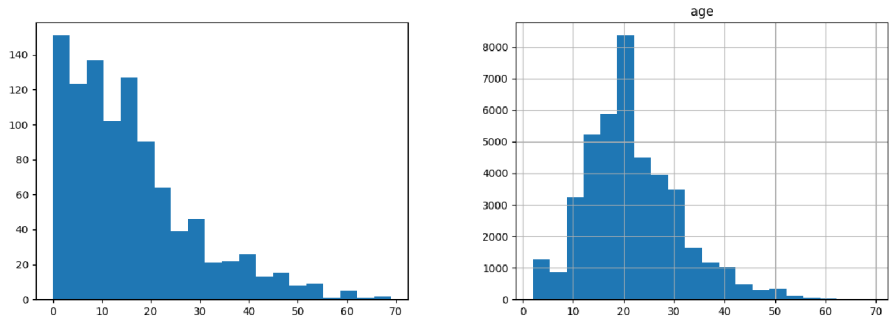
1.5.3 APPA-REAL Dataset

The APPA-REAL dataset is a relatively small dataset consisting of over 7000 images each labelled by real and apparent age. The apparent age label was decided by voting. On average 38 votes decided the apparent image. The total number of votes is around 250,000. The dataset is provided with train, validation and test splits containing 4113, 1500 and 1978 images respectively. There are 3818 images of male subjects and 3773 images of female subjects. All labels and votes are provided with the dataset. The age distribution is displayed on Figure 1.2b

The dataset was created for paper by Agustsson et al. [17]. Additional labels include gender, ethnicity and expression which were used in paper by Clapés et al. [18]

1.5.4 FG-NET Dataset

The FG-NET (Face and Gesture Recognition) is a face ageing dataset containing faces. The images are annotated by person identification number and age. Additionally for each image the dataset contains 68 manually annotated facial landmarks. Each person has multiple images at various ages. The dataset does not contain gender labels.



(a) FG-NET dataset (b) MegaAge dataset

Figure 1.3: Histogram of ages in FG-NET and MegaAge datasets

1.5.5 MegaAge Dataset

The MegaAge dataset contains over 41,941 face images annotated with posterior age distributions. The dataset is a result of a method introduced by Zhang Y. et al.[19]. Additionally the authors provide MegaAge-Asian dataset which contains 40,000 images of Asian faces. The dataset does not contain gender labels.

1.5.6 Adience Dataset

The dataset consist of over 26,580 images of 2,284 subjects. Each image is annotated with gender and one of 8 age groups. The dataset was created by crawling publicly available Flickr albums released under Creative Commons license. Adience dataset was created for paper by Eidinger et al.[12]. The dataset contains 53% images of female subject. The largest age group in the dataset is 25 to 32 years old.

Figure 1.4: Examples of images in the manually labelled dataset

1.5.7 Manually Labelled Dataset

We received camera data from Konica Minolta. We processed each frame of the data by extracting faces using MTCNN based face detection. Then we manually labelled the extracted faces by age and gender. It is difficult to guess a persons exact age, therefore we used a precision of 5 years.

The dataset contains 592 images of cropped faces with ages spanning 30 to 60 years. The dataset includes images from approximately 60 people. Out of all 592 images, 518 contain male subject and 74 contain female subjects. Table 1.1 shows the size of all age categories in manually labelled dataset. On Figure 1.4 we can see examples of images from the dataset.

Table 1.1: Overview of the manually labelled dataset

Age	Size
30	17
35	87
40	212
45	77
50	73
55	42
60	84

Table 1.2: Overview of all datasets used

Dataset	Number of Subjects	Precise Age	Gender Labels
IMDB	460,723	Yes	Yes
WIKI	62,328	Yes	Yes
UTKFace	24,107	Yes	Yes
APPA-REAL	7591	Yes	Yes
FG-NET	1002	Yes	No
MegaAge	41,941	Yes	No
Adience	26,580	No	Yes
Manual	592	Yes	Yes

2 Analysis

In this Chapter we will describe the selection of candidate models for evaluation. We will also detail the process of choosing the implementation frameworks and architecture.

To select candidate models we searched for existing open-source implementations of methods we discussed in Chapter 1. We focused on implementations using TensorFlow [1] machine learning framework. This library is the most popular open-source machine learning framework. Another advantage of TensorFlow is the possibility to serve trained models through an API, although we did not focus on this aspect in this thesis.

Another criterion for implementation selection was the availability of pretrained models. We wanted to first focus on the evaluation on available datasets. Furthermore, we wanted the age and gender classification to be done using the same neural network architecture.

During the selection we noticed that the majority of implementations serve only as proof of concept and are not suitable for use in production. The main issue was that they did not provide a way to initialise the model at the beginning and then feed the initialised network with images on demand.

1. http://www.tensorflow.org/

2.1 Implementation Selection

We found 3 implementations that fit our criteria. The first project [2] implements the DEX method introduced by Rothe et al. [10]. It uses Inception-ResNet-v1 convolutional neural network architecture. The pretrained models are trained on the IMDB-WIKI dataset.

The second project [3] implements the method introduced by Levi and Hassner [13]. The pretrained model uses Inception-v3 convolutional neural network architecture. It was trained using the Adience dataset.

The final project [4] we selected for our evaluation was an implementation of the Soft Stagewise Regression method introduced by Yang et al. [14]. The authors provide models pretrained on either IMDB or WIKI datasets. This implementation uses Keras [5] machine learning library running on top of TensorFlow.

2. https://github.com/BoyuanJiang/Age-Gender-Estimate-TF
3. https://github.com/dpressel/rude-carnie
4. https://github.com/shamangary/SSR-Net
5. http://keras.io/

3 Method Evaluation

We used three different open source implementations of methods. Each method is based on a paper presenting a solution to the age prediction problem.

3.1 Deep Expectation Approach

The DEX method combines classification and regression. The convolutional neural network is first trained for classification, classifying age into 101 age labels. The resulting age is then calculated as follows:

$$E(O) = \sum_{i=0}^{100} y_i o_i \tag{3.1}$$

where o_i represents softmax output probability of i-th age label and y_i is the age corresponding to class i. The pipeline of the DEX method is shown on Figure 3.1.

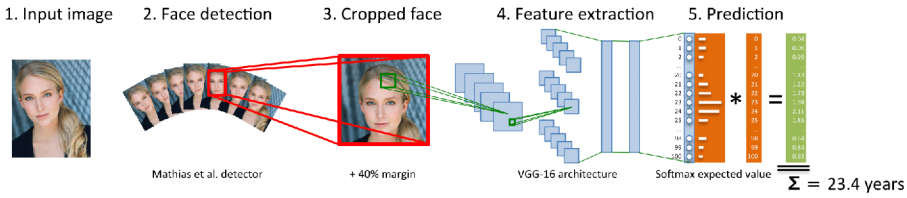


Figure 3.1: Pipeline of the DEX [10] method

DEX uses VGG-16 [20] neural network architecture. It is composed of 13 convolutional layers and 3 fully connected layers. Compared to previous network architectures the VGG-16 uses small 3x3 convolutions with the stride of 1 pixel. The diagram of the VGG-16 neural network is shown on Figure 3.2. The network achieves top-5 accuracy 92.7% in ImageNet. The main disadvantage of the VGG-16 network architecture is that it is very slow to train and uses a lot of resources.

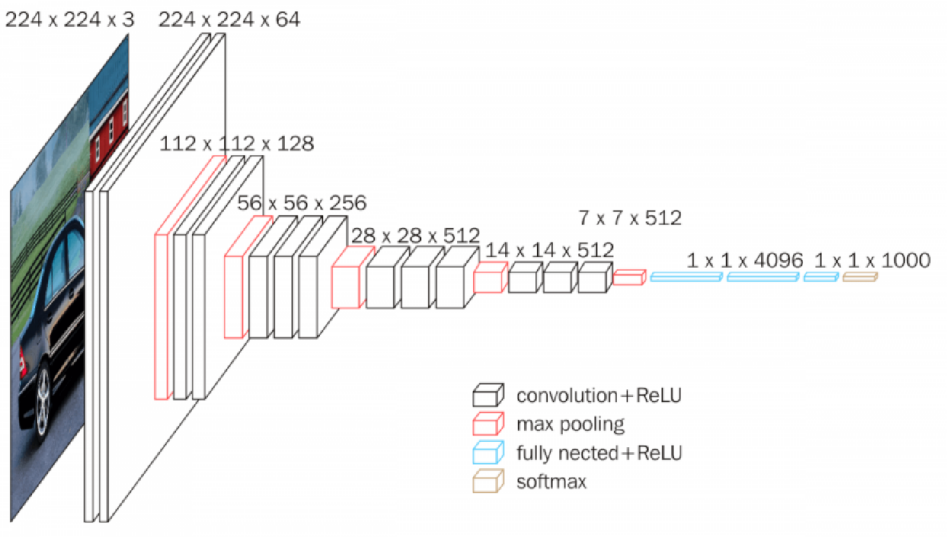


Figure 3.2: The architecture of VGG-16 [21] convolutional neural network

For this reason the open source implementation[1] of the DEX method uses Inception-ResNet-v1 [22], a different neural network architecture.

Inception-ResNet-v1 is a hybrid network consisting of inception blocks and residual connections. Inception blocks were introduced by Szegedy et al. [23].The problem they were trying to solve is that the important part of the image can vary in size. Therefore it is crucial to apply convolutional operations with different kernel sizes to the image. However just stacking convolutional layers can be very computationally expensive. The presented solution is to put convolutions next to each other and concatenate their output instead of stacking them on top one another.

In He et al.[24] authors were trying to address the issue of degradation of training accuracy. The accuracy of network gets saturated and adding more layers results in even worse performance. This is counter intuitive since we would expect the deeper model to perform at least as well as the model with less layers. The researchers suggest that the reason for this is that the solvers struggle to approximate identity

1. https://github.com/BoyuanJiang/Age-Gender-Estimate-TF

mappings. The proposed solution uses shortcut connections which skips one or more layers.

Inception-ResNet-v1 combines these two approaches into residual inception blocks. An example of residual inception block is shown on Figure 3.3. Inception-ResNet-v1 has approximately 50 million parameters, less than half of the number of parameters of VGG-16 convolutional neural network. VGG-16 contains 138 million parameters.

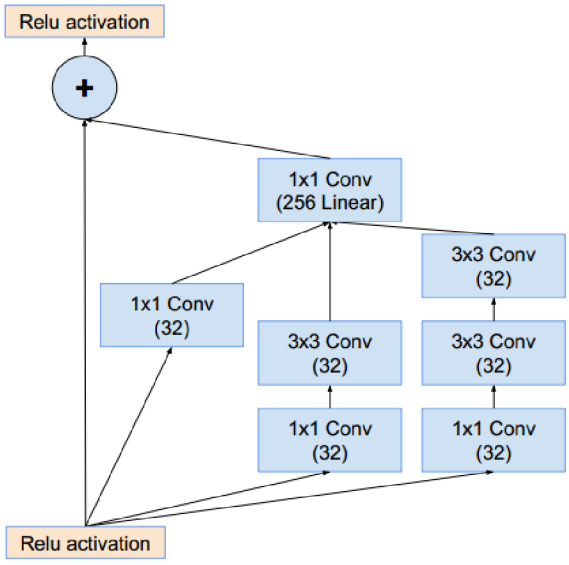


Figure 3.3: The schema of Inception-ResNet-v1 [22] network's 35×35 grid residual inception module

3.1.1 Model Evaluation

We evaluated the model using pretrained weights made available by the authors of the open source implementation of the DEX method.

The first experiment involved resizing images to various sizes and observing the performance on IMDB-WIKI dataset. The results are shown in Table 3.1. We can observe that with the image size getting smaller the results are getting worse. While with WIKI dataset and image size 160×160 pixels the gender error rate is 10.73% it gets more than 3 times worse when the image size is 80×80 pixels.

Table 3.1: DEX method results at various input image sizes

Dataset	Image Size	Age Mean Absolute Error	Gender Error
wiki	160	9.46	10.73
wiki	140	10.12	12.11
wiki	120	11.68	15.07
wiki	100	13.58	26.07
wiki	80	14.38	36.25
imdb	160	8.68	25.06
imdb	140	9.49	26.26
imdb	120	10.09	28.72
imdb	100	11.13	36.56
imdb	80	12.23	42.02

With these results in mind we tried to find out whether such different results are due to the image resolution or whether some valuable information is lost during the resizing process. Therefore we first resized the input image to 80×80 pixels and immediately resized it back to 160×160 pixels. The results shown in Table 3.2 prove that the poor results at 80×80 pixels are due to the low resolution.

Table 3.2: DEX results with images resized to 80×80 pixels and then back to 160×160 pixels.

Dataset	Age (MAE)	Gender Error
wiki	10.27	10.66
imdb	8.72	25.04

The open source implementation uses face detection with Dlib library before feeding the image to the network. Dlib implements face detection using histogram of oriented gradients described in Section 1.1. However we used face detection method based on Multi-task Cascaded Convolution Networks. We noticed some differences in model performance when using either Dlib or MTCNN for face detection. An important attribute to consider is the speed of face de-

tection. The results for IMDB dataset are shown in Table 3.3 while the results for WIKI dataset are displayed in table 3.4. We can observe that MTCNN based face detection is significantly faster however it does have lower gender accuracy and higher age mean absolute error. This might be due to how the dataset was preprocessed before the network was trained.

Table 3.3: DEX results with different face detection on IMDB dataset

Face Detection	Average Time (ms)	Age MAE	Gender Error
Dlib	240.80	5.79	15.09
MTCNN	83.92	7.68	22.71

Table 3.4: DEX results with different face detection on WIKI dataset

Face Detection	Average Time (ms)	Age MAE	Gender Error
Dlib	283.09	5.21	5.99
MTCNN	166.87	7.02	9.52

Before the images are fed to the network they are aligned using Dlib facial landmark detection. We noticed that some face images after face alignment are distorted or misaligned. We reasoned that the misalignment might be due to face bounding boxes not being perfectly square. Therefore we tried first to reshape the face bounding boxes to perfect square before passing them to face alignment. We also tried skipping the face alignment step altogether. The results with no reshaping before face alignment were already presented in previous paragraph, the rest of the results are presented in Tables 3.5 and 3.6 We can observe that in general simple face alignment without reshaping gives the best results compared to no face alignment.

Another dataset we tested this model on, was the Adience dataset. It is labelled with age groups and therefore it is more suitable for classification. We calculated mean absolute error by taking the minimum of difference between the predicted age and the bounds of the age group. If the predicted age fell within an age group the error was set to 0.

Table 3.5: DEX results with various combination of face detection, reshaping and aligning on IMDB dataset

Experiment	Avg Time (ms)	Age MAE	Gender Err
MTCNN-shape-align	86.89	7.75	22.43
MTCNN-shape	78.40	8.54	23.28
MTCNN	78.14	9.22	22.47
Dlib-shape-align	260.79	5.85	15.07
Dlib-shape	238.15	6.63	16.27
Dlib	241.34	5.99	15.7

Table 3.6: DEX results with various combination of face detection, reshaping and aligning on WIKI dataset

Experiment	Avg Time (ms)	Age MAE	Gender Err
MTCNN-shape-align	166.78	6.97	9.05
MTCNN-shape	160.43	8.29	10.22
MTCNN	158.48	7.46	7.9
Dlib-shape-align	285.40	5.32	6.07
Dlib-shape	281.58	5.63	5.99
Dlib	279.48	5.82	5.98

Finally we tested the model on all our datasets. The results are shown in Table 3.7. For IMDB and WIKI dataset we took only the best results. We can observe that the performance on IMDB and WIKI dataset is considerably better than on other datasets. This is most likely due to the fact that this model was trained using IMDB and WIKI datasets.

Apart for experimenting on labelled datasets we evaluated the model on datasets for facial recognition. One such dataset is the YouTube faces dataset[25]. While the dataset does not have age and gender labels it is useful for quickly visually evaluating the prediction. The dataset is composed of multiple frames from one video with a famous person. We noticed that averaging the predictions for one

Table 3.7: DEX model results on all datasets

Dataset	Age (MAE)	Gender Error
IMDB	5.79	15.07
WIKI	5.21	5.98
UTKFace	10.86	16.58
Adience	8.64	17.34
FG-NET	16.26	N/A
MegaAge	9.99	N/A
AppaReal	10.53	14.20
Manual	12.65	27.70

person improve the quality of the prediction. This could mean that combined with person tracking the age and gender prediction could achieve improved results.

Another useful resource to visually evaluate the model predictions are videos from the internet. We noticed that the prediction gets progressively worse the more the face is turned away from the camera. In a real world application a facial landmark detection method could be used to detect the face orientation.

To conclude the evaluation, the model is worse performing than the original DEX method. This may be improved with additional training. However an important attribute for real world applications is the speed of prediction. This is inherent to the neural network architecture.

3.2 Classification Using Convolutional Neural Networks

The method introduced by Levi and Hassner [13] uses small convolutional neural network to classify face image into 8 age classes and 2 gender classes. The network consists of only 3 convolutional layers and 2 fully connected layers. Authors argue that such a small network will reduce the risk of overfitting.

The method primarily works with 227×227 pixels center crop which is fed into the network. Alternatively the authors propose over-

sampling the image, where 5 crops of 227 × 227 pixels, one from each corner and one from the center, are presented to the network. In case of over-sampling the result is computed as the average of all 5 predictions. The authors claim that the over-sampling should compensate for misalignment of faces in a dataset.

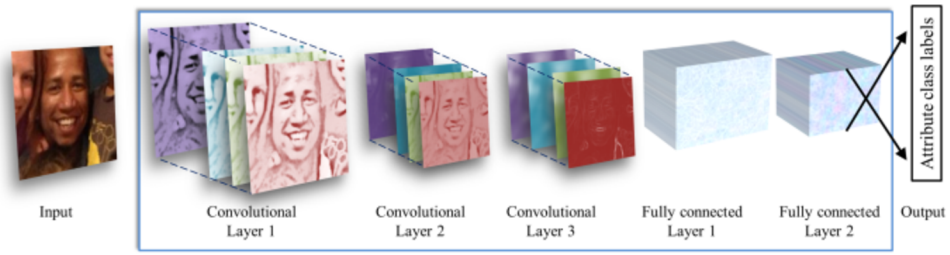


Figure 3.4: Illustration of neural network architecture used in classification method [13]

The open source implementation[2] of this model uses Inception-v3 architecture, introduced by Szegedy et al. [26]. In the paper, the authors introduce Inception-v2 and Inception-v3 architectures. Inception-v2 improves upon Inception-v1[23] by factorizing convolutions with large filter sizes, which are computationaly expensive. The Inception-v2 architecture replaces large convolutions with network of smaller convolutions. Figure 3.5 shows factorization of 5 × 5 convolution into a network of smaller convolutions. The Inception architecture contains an auxiliary classifier. In Inception-v3 batch normalization is applied not just to convolutional layers but also to the fully-connected layer of the auxiliary classifier.

3.2.1 Model Evaluation

While the authors of classification model use custom convolutional neural network architecture the open source implementation provides pretrained checkpoint for the Inception-v3 architecture. We used both mean absolute error and accuracy to evaluate this model. While evaluating the previous model we discovered that the MTCNN based face detection method has better performance than other methods.

2. https://github.com/dpressel/rude-carnie

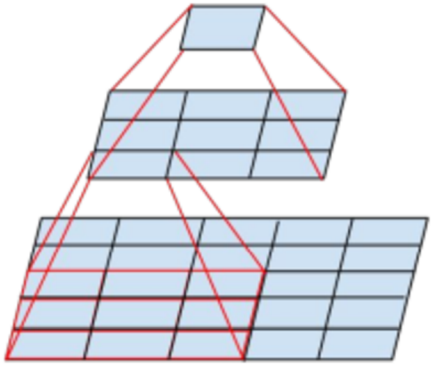


Figure 3.5: Illustration of factorization of 5×5 convolution[26]

Therefore for this model we only used MTCNN. This method requires two separate networks for age and gender respectively.

Table 3.8: Classification model results for IMDB-WIKI dataset at various image sizes

Dataset	Image Size	Age MAE	Gender Error
wiki	80	16.82	74.06
wiki	100	16.37	74.6
wiki	120	13.99	68.94
wiki	140	14.17	64.1
wiki	160	12.66	46.16
imdb	80	15.0	56.56
imdb	100	14.19	57.72
imdb	120	12.14	56.04
imdb	140	12.11	51.74
imdb	160	12.01	44.6

First we evaluated the model on IMDB-WIKI dataset. As evaluation metric we used mean absolute error. If the age label in the dataset was within the predicted age group, the error was set to 0. Otherwise it was set to the minimum difference between the label and boundaries

of predicted age group. Similarly to the evaluation of previous model we tested different image resolutions. The results are displayed on table 3.8. We can see that the gender prediction becomes worse than random guessing at image size 140×140 pixels.

This model, instead of aligning faces, uses so called over-sampling. Instead of feeding a single image it takes multiple crops of the image. The results are displayed in Tables 3.9 and 3.10. We can observe that over-sampling and reshaping do not give a boost in performance. On the other hand over-sampling has significant overhead and prediction can get more than 3 times slower.

Table 3.9: Results of classification model with reshaping and over-sampling on IMDB dataset

Experiment	Avg Time (ms)	Age MAE	Gender Error
Both	831.31	14.92	45.23
Reshape	267.46	14.48	44.58
Over-sample	834.37	14.88	44.16
None	271.1	16.23	44.0

Table 3.10: Results of classification model with reshaping and over-sampling on WIKI dataset

Experiment	Avg Time (ms)	Age MAE	Gender Error
Both	701.38	13.43	42.38
Reshape	220.56	13.29	41.37
Over-sample	690.29	13.28	38.89
None	222.20	12.86	38.45

Next we used the Adience dataset. The dataset is labelled with the same age groups that this model uses. Therefore we could evaluate the model using accuracy and off-by-one accuracy. As with the previous dataset we tried different combinations of reshaping and over-sampling. The results are displayed in Table 3.11

While the Adience dataset is more suited for accuracy metric we also used mean absolute error. The results are displayed in Table

Table 3.11: Results of classification model with reshaping and over-sampling on Adience dataset

Experiment	Age Acc	Age Off-by-one Acc	Gender Acc
Both	30.65	58.46	60.66
Reshape	32.05	62.44	61.80
Overample	33.45	62.96	63.47
None	34.0	65.43	63.82

3.12. The model performs significantly better on the Adience dataset compared to IMDB-WIKI datasets, however the reason might be that the model was trained on the Adience dataset.

Table 3.12: Results of classification model on Adience dataset using mean absolute error

Experiment	Age MAE
Both	7.65
Reshape	8.49
Over-sample	7.11
None	8.12

Finally we evaluated the model on all other datasets. The results are displayed in Table 3.13. For IMDB, WIKI and Adience datasets we show only the best results.

This model does not perform as well. While for some datasets the age mean absolute error is competitive (below 10) the gender accuracy is significantly lacking. The best result this model achieved was age mean absolute error of 7.11 and 24.66 gender error.

3.3 Soft Stagewise Regression Network

Yang et al. [14] introduce a novel approach for age prediction. Building on the DEX method they create Soft Stagewise Regression Network (SSR-Net). The disadvantage of the DEX method is that the convolutional neural network has 101 neurons in the last classification layer,

Table 3.13: Classification model results on all datasets

Dataset	Age (MAE)	Gender Error
IMDB	14.48	44.16
WIKI	12.86	38.45
UTKFace	12.42	50.3
Adience	7.11	36.18
FG-NET	7.34	N/A
MegaAge	7.89	N/A
AppaReal	13.76	62.23
Manual	12.41	24.66

which leads to a large number of connections and therefore a large number of parameters to train. The SSR-Net addresses it by employing a "coarse-to-fine strategy". Each stage classifies into only a small number of classes.

The network is composed of 2 parallel heterogeneous streams. Each stream consists of convolutional blocks. At each stage, features from both streams are combined in a fusion block. The schema of the SSR-Net architecture is shown in Figure 3.6. SSR-Net uses 3 stages each with 3 classes. The authors provide their own implementation of the network[3].

3.3.1 Model Evaluation

The authors provide models pretrained on either WIKI or IMDB dataset. The models base image resolution is 64×64 pixels, compared to 80×80 pixels, the smallest resolution that can be fed into previous models. Therefore it was not necessary to resize the input images to different sizes.

We tested each pretrained model on each of our datasets. The results for age estimation are displayed in Table 3.14. Table 3.15 shows the results for gender estimation. Each prediction took on average 10ms.

3. https://github.com/shamangary/SSR-Net

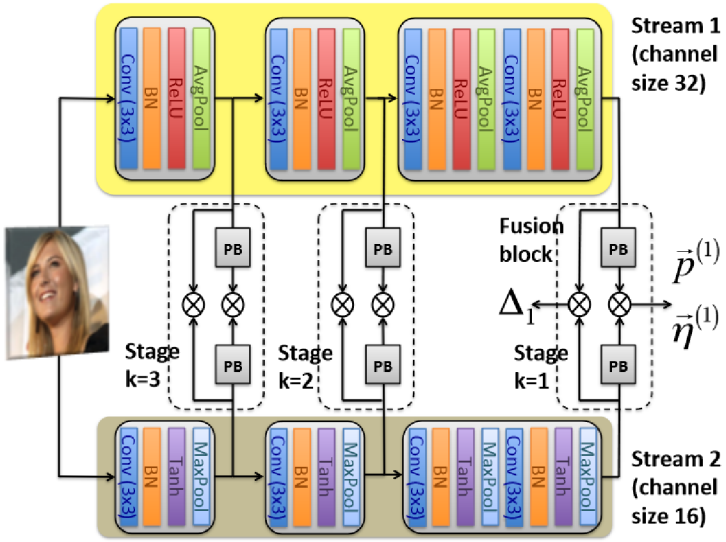


Figure 3.6: Schema of SSR-Net [14] architecture

We can observe that the mean absolute error and gender error are considerable higher on UTKFace and Adience datasets compared to the evaluation errors on IMDB and WIKI datasets. This might be due to the fact that the model was trained only on either IMDB or WIKI datasets. However using the AppaReal dataset we achieved similar results to IMDB and WIKI datasets.

Finally we evaluated the pretrained model on our manually labelled dataset. We can observe that despite the fact that the images were not taken in a controlled environment and are of lower quality than in other datasets the model performed very well. The reason may be that the age and gender distributions of the manually the dataset are very close the distribution that the model has learned.

The evaluation results of the pretrained SSR-Net model show that the error depends a lot more on the evaluation data rather than the data that the model has been trained on. Other models have achieved better results on some datasets, however it is important to note that, unless stated otherwise, we evaluated these models using the default image resolution. For SSR-Net it is 64×64 pixels, for Inception-ResNet-v1 (DEX) it is 160×160 pixels and for Inception-v3 (Classification method) it is 227×227 pixels. Our evaluation of these

Table 3.14: Comparison between SSR-Net age results trained on WIKI and IMDB dataset

Dataset	Age WIKI MAE	Age IMDB MAE
IMDB	8.48	7.44
WIKI	5.38	6.04
UTKFace	11.37	11.27
Adience	8.08	16.44
FG-NET	12.85	13.73
MegaAge	8.15	8.5
AppaReal	9.01	8.97
Manual	12.42	9.95

models on IMDB-WIKI dataset using different sizes of images proves that the lower the resolution the worse the performance of the model.

3.4 Method Evaluation Conclusion

We evaluated 3 different pretrained models. Table 3.16 shows detailed comparison of age estimation between all pretrained models on each dataset. Table 3.17 shows the comparison for gender prediction.

The first model which implements the DEX method using the Inception-ResNet-v1 architecture does give good results when the image size is 160×160 pixels. For age prediction the best result the model achieved was mean absolute error of 5.21 on the WIKI dataset using face aligning and Dlib face detection. For gender prediction the best result was error rate of 5.99% on the WIKI dataset using Dlib face detection and no face alignment. However as the resolution of the image decreased the performance of the model declined. At the resolution of 80×80 pixels the model achieved age mean absolute error of 12.23 on the IMDB dataset and gender error rate of 36.25% on the WIKI dataset.

Another disadvantage of this model is the prediction speed. With the best prediction time being approximately 100ms this would mean that a real world application could only process 10 frames per second.

Table 3.15: Comparison between SSR-Net gender results trained on WIKI and IMDB dataset

Dataset	Gender WIKI Error	Gender IMDB Error
IMDB	17.9	17.15
WIKI	5.29	6.39
UTKFace	24.85	20.45
Adience	15.91	16.44
AppaReal	10.75	10.97
Manual	10.48	10.48

The second model classifying age into 8 different age groups performed noticeably worse. The best result the pretrained Inception-v3 model achieved was age mean absolute error of 7.11 on the Adience dataset and gender error rate of 36.18% on the Adience dataset. The model was trained using the Adience dataset, therefore its performance was worse on other dataset such as IMDB-WIKI. The best age classification accuracy on the Adience dataset that the model reached was 34% and best the off-by-one accuracy was 65.43%. The best gender classification accuracy was 75.36%. Feeding the network multiple crops of the image does not improve the accuracy but it does significantly reduce the prediction speed. The results reported by the authors of the method were 50.7% age accuracy, 84.7% off-by-one age accuracy and 86.8% gender accuracy. While the off-by-one accuracy is good, the difference between real age and predicted age group can be significant.

The final test model, Soft Stagewise Regression Network performed the best from all tested models. The best result it achieved was age mean absolute error of 5.38 and gender error of 5.29%. The worst results were achieved on the UTKFace dataset, however these results were still competitive with the other two models. Furthermore these results were achieved at resolution of 64×64 pixels which other models were not able to process. The prediction speed is another strength of the model. The average time to process age and gender was under 10 milliseconds. This means that the model is able to process video stream in real time.

Table 3.16: Comparison of best age prediction results of all evaluated methods using mean absolute error

Dataset	DEX	Classification	SSR-Net
IMDB	5.79	14.48	7.44
WIKI	5.21	12.86	5.38
UTKFace	10.86	12.42	11.27
Adience	8.64	7.11	8.08
FG-NET	16.26	7.34	12.85
MegaAge	9.99	7.89	8.15
AppaReal	10.53	13.76	8.97
Manual	12.65	12.41	9.95

4 Implementation

Based on our evaluation in Chapter 1 we selected SSR-Net method as a basis for our application. In this Chapter we discuss the implementation of the application for age and gender estimation.

4.1 Application Overview

Our application provides two different ways to use it. First, the application is provided as a Python package, where it provides several objects which can be imported in another Python project. The second way of interacting with the application is through a command line interface. Users are able to interact with each object directly through command line.

The application has several parts. When a user wants to classify an image, first a face is detected. Optionally using facial landmark detection we can filter out faces too tilted in x axis. Finally detected faces are passed to our model and classified. The estimation result can be displayed with face bounding boxes and age and gender labels drawn on the input image.

Table 3.17: Comparison of best gender prediction results of all evaluated methods

Dataset	DEX	Classification	SSR-Net
IMDB	15.07	44.16	17.15
WIKI	5.98	38.45	5.29
UTKFace	16.58	50.3	20.45
Adience	17.34	36.18	16.44
AppaReal	14.2	62.23	10.97
Manual	27.7	24.66	10.48

When the application is used as a Python module it is possible to pass a person id along with their face. Then, the estimation result will be stored in memory and the next predictions for the same person id will be computed as average of all previous predictions. This way, the user can take advantage of person tracking system in conjunction with our application.

4.1.1 Face Detection

We discussed various methods for face detection in Section 1.1. In our application we provide two face detection method. The first method uses classifier based on histogram of oriented gradients implemented by open-source library Dlib [1]. The second method uses Multitask Cascaded Convolutional Neural Network (MTCNN) [5]. We recommend using this method as it is faster and more precise than HOG.

Each face the detector detects in an image is resized to the desired image size. For our models the desired size is 64×64 pixels. The detector returns resized faces as well as coordinates of their bounding boxes in the original image.

4.1.2 Facial Landmarks Detection

In our application we do not use face alignment using the detected landmarks. However it is useful to know the position of eyes, nose and

1. www.dlib.net

mouth to determine the tilt of a face. As we discussed in Section 3.1 we noticed that the estimation got progressively worse, the more the face was turned away from the camera. Therefore we implemented optional face filtering using facial landmark detection.

We decided to use Face Alignment Network (FAN) [7] as it gave us good results during our experiments. After the faces in an image are detected, they are then passed to the facial landmark detector. The facial landmark detector localises 68 facial landmarks such as outline of eyes, nose, mouth, eyebrows and jawline. Using the points on the outline of eyes we compute the center of each eye. Then we fit a straight line through the points on the nose. We then compute the distance from each eye to the nose line. Finally we compute the ratio of the distances. If the ratio is bellow a certain threshold the face can be passed to age and gender predictor.

We experimented with various thresholds but the best visual result was achieved using threshold of 1.2.

4.1.3 Age and Gender Prediction

The final stage is age end gender estimation. For our application, we decided to train our own models. We discuss the training process and evaluation of trained models in Chapter 5.

Additionally we chose to train a different set of models with 4 colour channels. As we discussed in Section 1.1 many popular face detection methods use integral images to find faces. We add the integral image as an additional colour channel. In our application we provide both standard models and models with 4 colour channels.

After face detection the detected faces are first fed to the age estimation model and then to the gender estimation model. The results can be displayed on the original image. Otherwise the result is returned by the prediction function.

5 Model Training

Based on our results from experiments with pretrained models, we decided to further train and test the SSR-Net method.

5.1 Data Preprocessing

Each dataset described in chapter 1.5 uses different style of labelling, some datasets contain mislabelled images or do not contain recognisable face. The SSR-Net method requires separate networks for age and gender predictions. Therefore it was necessary to prepare the data for training.

For training the age prediction we processed the datasets with images labelled with age labels. Each image was passed through MTCNN face detector described in chapter 1.1. If a face was detected with confidence greater than 95% and labels were not missing, the image was added to filtered dataset. Each filtered dataset was then split to training, validation and test subsets with ratios 80%, 10% and 10% respectively.

For gender prediction training, the process was similar. The FG-NET dataset described in section 1.5.4 and the Megaage dataset described in section 1.5.5 do not have gender labels therefore they were only used for training of age prediction model.

The filtered train, validation and test subsets were then combined to form a large dataset. Table 5.1 provides an overview of the sizes of subsets of the large training dataset. The histogram of ages of the combined dataset is displayed in Figure 5.1. Table 5.2 shows the number of male and female subjects in combined dataset for gender training.

We trained the model with different optimizer and different starting learning rate. For optimizers we use standard gradient descent, RMSprop and Adam optimizer. For starting learning rate we used $1 \times 10^{-k}, k = 1 \dots 4$.

The main disadvantage of standard gradient descent is that it is not adaptive. The choice of proper learning rate is cruicial and the same learning rate applies to all parameters. Another issue arises when the error function is highly non-convex and the model can get trapped in a local minima or in a saddle point where the gradient is close to zero.

Table 5.1: Overview of the sizes of subsets of the combined dataset

Class	Subset	Size
Age	Train	200,469
Age	Validation	26,044
Age	Test	26,527
Gender	Train	166,116
Gender	Validation	21,750
Gender	Test	22,231

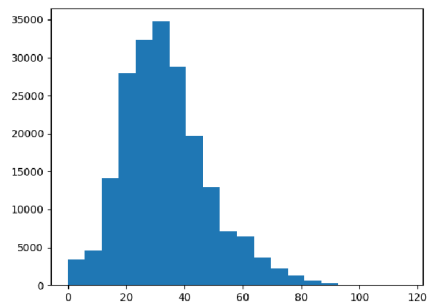
Table 5.2: Number of male and female subjects in combined dataset for gender training

Subset	Males	Females
Train	93,428	72,688
Validation	12,254	9,496
Test	12,369	9,862

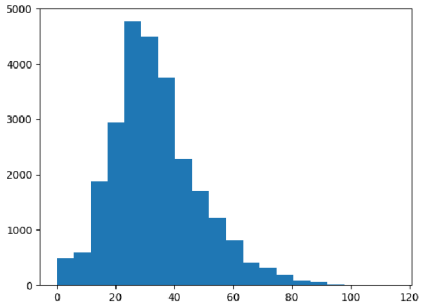
RMSprop or Root Mean Square Propagation tries to address the issues of gradient descent by dividing the learning rate by an exponentially weighted average of squared gradient. Let's assume a mini-batch of size T. Equation 5.1 shows the update of weight w_{ji} at time $t+1$. The author of RMSprop suggests for γ value 0.9. The parameter δ is a smoothing term to avoid division by zero.

$$\begin{aligned} w_{ji}^{(t+1)} &= w_{ji}^{(t)} + \Delta w_{ji}^{(t)} \\ \Delta w_{ji}^{(t)} &= -\frac{\eta}{r_{ji}^{(t)} + \delta} \times \sum_{k \in T} \frac{\partial E_k}{\partial w_{ji}}(\vec{w}^t) \\ r_{ji}^{(t)} &= \gamma r_{ji}^{(t-1)} + (1-\gamma)(\sum_{k \in T} \frac{\partial E_k}{\partial w_{ji}}(\vec{w}^t))^2 \end{aligned} \tag{5.1}$$

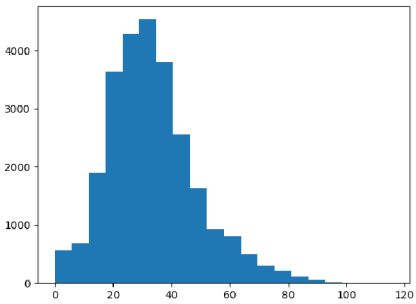
Momentum is used with gradient descent to smooth the update. It adds an exponentially weighted average of past updates to the current update. Adam or Adaptive Moment Estimation was introduced by



(a) Train subset



(b) Validation subset



(c) Test subset

Figure 5.1: Histogram of ages in combined dataset for age training

Kingma and Ba [27]. It combines the approach of RMSprop with momentum. Let g_t be the gradient at step t. Equation 5.2 shows how Adam optimizer computes the exponentially decaying average of past gradients and past squared gradients. The original paper suggests values for β_1 and β_2 to equal 0.9 and 0.999 respectively.

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \end{aligned} \tag{5.2}$$

The equation 5.3 shows update for parameter (i, j). The parameter ϵ is a smoothing term to avoid division by zero.

$$w_{ji}^{(t+1)} = w_{ji}^{(t)} - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} m_t \tag{5.3}$$

5.2 Training Process

We trained each combination of optimizer and learning rate for 60 epochs, using mean absolute error as the loss function. During the training we monitored the training loss as well as validation loss. The training was done on 4 CPUs, 30GB of RAM and 1 GPU on the infrastructure made available by Metacentrum VO [1]. The training of one combination took approximately 6 hours. We first tried learning rate of 0.1, however, for each optimizer the model diverged.

5.2.1 Age Network

The best learning rate for Adam optimizer was 0.001 achieving mean absolute error of 6.48. For RMSprop optimizer the best learning rate is 0.001 with mean absolute error of 6.65. Using the stochastic gradient descent optimizer we achieved mean absolute error of 6.88 with learning rate equal 0.01.

Table 5.3 shows comparison of validation losses after 60 epochs for each combination.

The best optimizer, learning rate combination based on validation loss was Adam optimizer with learning rate 0.001. We evaluated this model on the test dataset. Since RMSprop optimizer with learning rate 0.001 and gradient descent optimizer with learning rate 0.01 achieved similar results we also evaluated these models on the test dataset. The results of the evaluation on validation and test subsets are displayed on Table 5.4. Based on both results the best combination of optimizer and learning rate is Adam with 0.001 learning rate.

5.2.2 Gender Network

We trained the gender estimation networks similarly to age estimation networks. The best learning rate for Adam optimizer is 0.001 achieving

1. https://www.metacentrum.cz

Table 5.3: Validation loss for each combination for training the age network

Optimizer	Learning Rate	Validation Loss
Adam	0.01	7.34
Adam	0.001	6.48
Adam	0.0001	7.49
RMSprop	0.01	8.32
RMSprop	0.001	6.65
RMSprop	0.0001	7.57
SGD	0.01	6.88
SGD	0.001	7.54
SGD	0.0001	8.92

Table 5.4: Test loss for best combinations for age network training.

Optimizer	Learning Rate	Val Loss	Test Loss
Adam	0.001	6.48	7.08
RMSprop	0.001	6.65	7.23
SGD	0.01	6.88	7.41

validation loss of 0.09. For RMS prop optimizer the best result we achieved is validation loss of 0.09 for learning rate of 0.001.

Finally, with gradient descent optimizer we achieved validation loss of 0.1 for learning rate 0.01. It is interesting to note that learning rate of 0.0001 was too low and the model achieved validation loss as high as 0.47.

On Table 5.5 we can see comparison of validation loss after 60 epochs for each combination.

The best optimizer and learning rate combination was RMSprop with learning rate equal to 0.001. However Adam optimizer with learning rates 0.001 and 0.0001 as well as RMSprop optimizer with learning rate 0.0001 and gradient descent optimizer with learning

Table 5.5: Validation loss for each combination for training the gender network

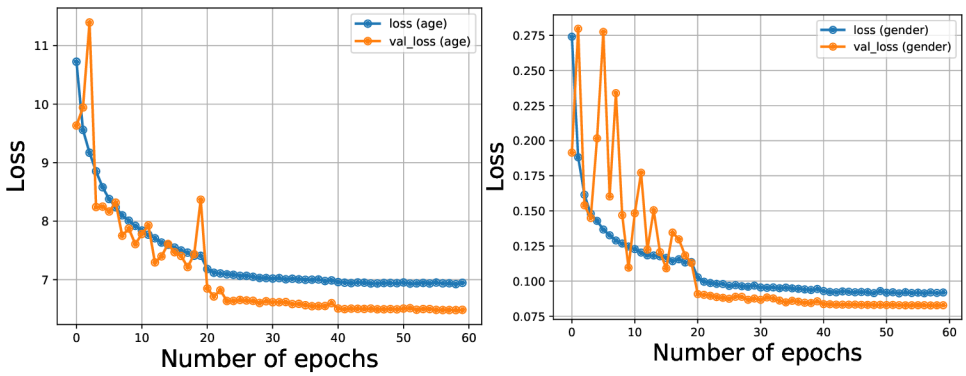
Optimizer	Learning Rate	Validation Loss
Adam	0.01	0.16
Adam	0.001	0.09
Adam	0.0001	0.10
RMSprop	0.01	0.15
RMSprop	0.001	0.08
RMSprop	0.0001	0.10
SGD	0.01	0.10
SGD	0.001	0.19
SGD	0.0001	0.47

rate 0.01 achieved results that were close to the best. Therefore we evaluated all these combinations on the test dataset.

Table 5.6: Test loss for best combinations for gender network training.

Optimizer	Learning Rate	Val Loss	Test Loss
Adam	0.001	0.09	0.10
Adam	0.0001	0.10	0.12
RMSprop	0.001	0.08	0.10
RMSprop	0.0001	0.10	0.12
SGD	0.01	0.10	0.12

The best performing combination of optimizer and learning rate based on the evaluation on validation and test datasets was RMSprop with learning rate 0.001. The results of all evaluated combinations are displayed in Table 5.6.



(a) Age model

(b) Gender model

Figure 5.2: Graphs of training and validation loss for best age and gender models

5.3 Evaluation of Best Trained Models

Similarly to our evaluation of pretrained SSR-Net model we evaluated our best models trained on the combined dataset. The model chosen for age evaluation was the model trained using Adam optimizer with learning rate 0.001. The graph of training loss and validation loss for the model is displayed in Figure 5.2a

The best performing gender model combinations of optimizer and learning rate based on both validation and test results are Adam and RMSprop both with learning rate equal to 0.001. RMSprop has slightly better performance, therefore we chose RMSprop with learning rate equal to 0.001 for evaluation on individual datasets. The graph of training loss and validation loss for the model is displayed in Figure 5.2b

On Table 5.7 we can see comparison of best pretrained SSR-Net age models and out age model trained on the combined dataset. We can observe that the mean absolute error improved on all datasets. On some datasets like UTKFace or FG-NET it improved significantly.

Table 5.8 shows comparison between best pretrained gender SSR-Net models and our gender model trained on the combined dataset. It is interesting to observe that the gender error significantly improved on some dataset like UTKFace, it worsened on other dataset like IMDB and WIKI.

Table 5.7: Comparison of trained age models with best pretrained SSR-Net models

Dataset	Age Pretrained MAE	Age Trained MAE
IMDB	7.44	7.20
WIKI	5.38	5.25
UTKFace	11.27	6.49
Adience	8.08	5.69
FG-NET	12.85	6.61
MegaAge	8.15	5.58
AppaReal	8.97	7.59
Manual	9.95	5.66

5.4 Evaluation on Manually Labelled Dataset

In this Chapter we discuss the detailed evaluation of our implemented method and trained models on manually labelled dataset. The dataset consists of 592 images of approximately 60 people of which 74 are female and 518 are male. We introduced the dataset in greater detail in Section 1.5.7.

5.4.1 Age Prediction Evaluation

For training and evaluation of age prediction model we used mean absolute error (MAE). Evaluating on the entire dataset we achieved MAE of 5.98. Next we computed the error for each age label. The results are shown in Table 5.9.

We can observe that the best MAE, 3.96 is achieved with age label 45. The largest group, 40 years old, achieve a very good MAE of 4.73. This shows that the best MAE was achieved at or near peak of age distribution in the combined training dataset. It is interesting to note that the worst MAE of 10.63 corresponds with age 50. To explain this we look further into the MAE accross the gender split.

The results displayed in Table 5.10 show that for males the MAE is similar to age labels 55 and 60. However, for females the MAE is much higher but with fewer examples. There might be few different reasons

Table 5.8: Comparison of trained gender models with best pretrained SSR-Net models

Dataset	Gender Pretrained Error	Gender Trained Error
IMDB	17.15	17.4
WIKI	5.29	6.32
UTKFace	20.45	10.6
Adience	15.91	15.36
AppaReal	10.75	9.98
Manual	10.48	10.08

Table 5.9: Mean absolute error for each age label

Age Label	Mean Absolute Error	Number of Examples
30	6.71	17
35	4.52	87
40	4.72	212
45	3.96	77
50	10.63	73
55	7.64	42
60	7.51	84

for this. From human annotator error to a particularly difficult set of images for the model.

5.4.2 Gender Prediction Evaluation

Since gender prediction is a binary classification, we have more metrics available to evaluate the performance of our trained model. Two usefull metrics are precision and recall. Precision is the ability of a model not to label positive example as a negative. It is calculate by dividing the number of true positive examples by the number of all positive examples.

On the other hand, recall is the ability of a model to find all positive examples. It is calculated by dividing the number of true positive

Table 5.10: Mean absolute error for age label 50

Gender	MAE	Number of Examples
Male	7.78	60
Female	16.91	13

examples by the sum of the number of true positives with the number of false negative examples.

Another metric closely related to precision and recall is F1 score. It is a weighted average of precision and recall. Equation 5.4 shows the calculation of F1 score.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{5.4}$$

Confusion matrix shows the number of examples. The row of the matrix signifies the true label and the column signifies the predicted label.

Table 5.11: Precision, recall and F1 score for each gender class

Gender	Precision	Recall	F1	Size
F	0.95	0.49	0.64	74
M	0.93	1.00	0.96	518

Table 5.11 shows precision, recall and F1 score for each gender class. We can see that the model is able to correctly identify all male examples. On the other hand only 49% of females is correctly identified. Confusion matrix, displayed in Table 5.12 shows that 38 females have been incorrectly classified as males. The reason may be that the gender distribution in this particular dataset is heavily skewed towards males.

To investigate this further we evaluated female gender error for each age label. Table 5.13 shows the results. We can observe that the largest contributor to the gender error is age 40. To explain this we looked through the images. We found that the majority of the images in this age and gender category are of one female subject. The model might have difficulty with this particular subject. Furthermore the

Table 5.12: Confusion matrix of gender classification

	Female	Male
Female	36	38
Male	2	516

image was taken from an angle where it is difficult to see any distinctive female features.

Table 5.13: Gender error for female subjects by age

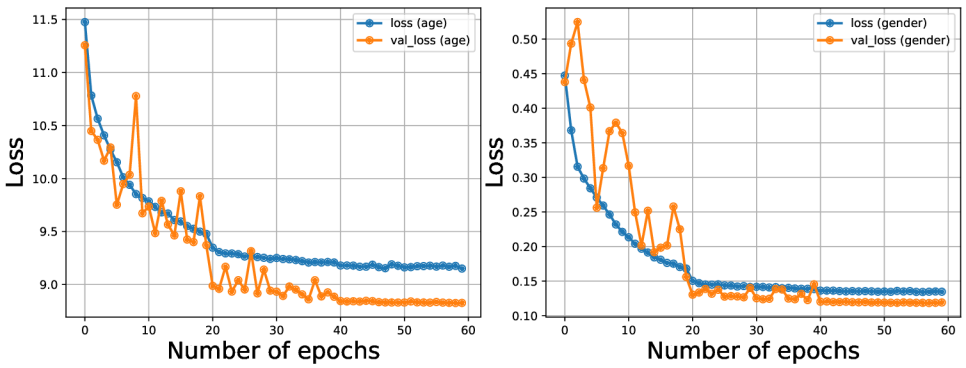
Age	Gender Error	Size
35	25.0	8
40	73.58	53
50	15.38	13

5.5 Extending Data

As we discussed in Chapter 1 many popular face detection implementations are based on the use of integral image, or summed-are table. As far as we are aware there is no research which uses integral images in age or gender estimation. We decided to try and train the model using integral image as fourth channel, in addition to standard red, green and blue channels.

First it was necessary to preprocess the data again. The process was similar to what we described in Section 5.1, but instead of adding images directly to filtered dataset, it was necessary to first convert the image to grey scale, then compute the integral image and add it as additional colour channel to the original image.

The extended dataset was considerably larger and required lot more resources for training. Therefore we trained only the optimizer learning rate combinations we found most successful on the non-extended dataset. The graphs showing training loss and validation loss is displayed on Figure 5.3. The training was done on 4 CPUs, 120GB of RAM and 1 GPU on the Metacentrum VO infrastructure.



(a) Age training (b) Gender training

Figure 5.3: Training loss and validation loss for age and gender training with integral image

Table 5.14 shows comparison of validation loss and test loss after 60 epochs for models using standard images and models using integral image as an additional colour channel. We can observe that the addition of integral image did not improve validation nor test loss.

Table 5.14: Comparison of validation loss and test loss for training standard models and models with integral images

Training	Val Loss	Test Loss
Age Integral	8.83	9.16
Age	6.48	7.08
Gender Integral	0.12	0.36
Gender	0.08	0.10

On Table 5.15 we can see the comparison between age models we trained earlier and models trained with integral images. We can observe that adding integral image as an additional channel did not improve the results. Similarly on Table 5.16 we can see the same result for gender models. The gender prediction accuracy did not improve by adding integral images.

Table 5.15: Comparison of trained age models with standard images and integral images

Dataset	Age MAE	Age Integral MAE
IMDB	7.20	7.69
WIKI	5.25	7.14
UTKFace	6.49	9.14
Adience	5.69	9.83
FG-NET	6.61	10.44
MegaAge	5.58	7.08
AppaReal	7.59	11.46
Manual	5.66	6.74

5.6 Model Evaluation Conclusion

We trained the SSR-Net networks with combined dataset which is comprised of 7 publicly available dataset and our manually labelled dataset. Training of age network improved the mean absolute error on each of the tested datasets. Gender network training improved the gender error on most of the datasets, however it worsened the results on IMDB and WIKI datasets.

We evaluated our trained model on manually labelled dataset. We found that age estimation works well, however the model tends to predict age that is close to the peak of the age distribution in the training dataset. The gender model is skewed towards males. The reason may be that this dataset contains a lot more male subjects than female subjects. To correct this behaviour it is necessary to collect more images with more equal gender distribution and train the model with those.

Adding integral image as an additional colour channel did not prove useful as both age mean absolute error and gender error worsened on all tested datasets. It is possible that training with integral images requires completely different set of neural network hyperparameters or a different neural network architecture.

Table 5.16: Comparison of trained gender models with standard images and integral images

Dataset	Gender Error	Gender Integral Error
IMDB	17.4	20.48
WIKI	6.32	7.5
UTKFace	10.6	14.0
Adience	15.36	23.64
AppaReal	9.98	16.15
Manual	10.08	12.5

6 Documentation

6.1 User Documentation

In this Section we describe how to use our application. We describe both the use as a Python module and the use through command line interface.

6.1.1 Installation

First it is necessary to install all the dependencies. We suggest installing them inside a virtual environment. The easiest way to do it is to use Anaconda[1] Python 3.7 distribution. To install all dependencies run `pip install -r requirements.txt` inside a virtual environment. After successful installation the application is ready to be used.

6.1.2 Python Module

The simplest way to use our application is to import class `Predictor` from `predict` module. Creating an instance of the object without any arguments will initialise the object with our trained models. Alternatively the user can specify arguments:

1. http://www.anaconda.com/distribution/

- `age_model_path` - The path to trained weights for age prediction model
- `gender_model_path` - The path to trained weights for gender prediction model
- `integral` - Specifies whether to use models trained with integral images for prediction

The `Predictor` class contains a single method call `predict`. The method accepts these arguments:

- `image` - The original image
- `faces` - A list containing detected faces
- `boxes` - A list of bounding box coordinates for each face on the original image
- `draw` - Boolean value whether to draw predicted values and bounding boxes on the original image
- `person_ids` - Identification numbers of people in the picture if the prediction module is used in conjunction with person detection method.

The images and faces are accepted as NumPy[2] arrays in blue, green, red ordering. To load the images in this way we suggest using OpenCV[3] library and function `cv2.imread()`.

To first detect faces on the image it is necessary to import `Detector` class from `face_detector` module. Instantiating the `Detector` object without any arguments will initialise the object with MTCNN face detection and disabled face filtering. Alternatively the user can specify arguments:

- `detector` - Specifies which face detector to use. Possible values are "dlib" or the default value "MTCNN"

2. http://www.numpy.org
3. http://opencv.org/

- `filter` - The threshold to use for filtering faces. The default value is `None` which disables face filtering.
- `img_size` - The size of the image required by `Predictor` object. The default value is 64×64 pixels.

The detector object has only one public method called `detect()` which accepts an image as a NumPy array and returns a list of detected faces and a list of bounding boxes coordinates. If the face filtering is turned on the return values will contain only faces and bounding boxes which pass the filter process.

We also provide a simple `demo.py` script which shows a simple usage of the application as python module.

6.1.3 Command Line Interface

In addition to Python module we provide command line interface. The interface is meant as a one-off prediction. First, it calls face detection and then it estimates age and gender on the detected faces. It does not do any face filtering nor it does accept person id. The results of prediction are printed to standard output and drawn on the image and saved. To use the command line interface call `predict.py` with following arguments:

- `detector` - Face detection method to use
- `age_model_path` - Path to trained weights of the age estimation model
- `age_model_path` - Path to trained weights of the gender estimation model
- `image` - Path to image to use for face detection and age and gender prediction
- `output` - Path where to save the output image with drawn bounding boxes and predictions
- `integral` - Specifies whether to use models trained with integral images for prediction

For example a prediction using command line interface could look like this:

```
python predict.py --detector MTCNN
                  --age_model_path ./age.h5
                  --gender_model_path ./gender.h5
                  --image ./image.jpg
                  --output ./result.jpg
```

6.1.4 Training Models

We provide a simple way to train custom models. First it is necessary to preprocess the datasets into a format suitable for training. For this we provide utility script `preprocess.py`. It accepts the following command line arguments:

- `data_path` - The base path where all datasets are located
- `output_path` - Specifies where to save processed data
- `img_size` - Image size for training
- `gender` - Specifies that we are training gender models
- `integral` - Specifies that the images need to be expanded with integral image as an additional color channel

After the data preprocessing is done, to train a model, it suffices to call `train.py`. The script accepts following command line arguments:

- `learning_rate` - Sets the initial learning rate.
- `output_path` - Path where to save the trained model
- `epoch` - Number of training epochs
- `batch_size` - Batch size for training
- `data` - Path to preprocessed data
- `optimizer` - The optimizer to use for training

- `gender` - Specifies that we are training gender model
- `integral` - Specifies that we are training model where images have integral image as an additional colour channel

6.2 Technical Documentation

The application is written in Python programming language. It requires Python version 3.7. The support for neural networks is done through Keras and TensorFlow libraries. We provide full list of dependencies in `requirements.txt`

6.2.1 Application Scripts

In this Section we will detail the important Python scripts of the application.

predict.py

Contains class `Predictor`. It is responsible for loading the models, running prediction on the input and drawing the result on the input image. Additionally it is responsible for parsing command line arguments and processing the input from command line interface. If the application is used in conjunction with person tracking, `Predictor` class is responsible for computing the average value of all previous prediction for one person.

face_detect.py

Contains class `Detector`. It is responsible for detecting faces on the input image. If the face filtering is switched on then the `Detector` class will pass the detected faces to `face_filter.py`. `Detector` returns detected faces and the coordinates of their bounding boxes on the original image.

face_filter.py

`FaceFilter` class is responsible for finding the center of each eye and nose line using facial landmarks. It then computes the ratio of distances

from each eye to nose. It returns only the indexes of faces with ratio below a threshold.

SSRNet_model.py

Contains Soft Stagewise Regression Network implemented in Keras machine learning library. It consists of both age network and more general gender network.

6.2.2 Third Party Libraries

In our application we used several open-source third party libraries. In this section we detail which libraries we used.

Keras

Keras is a high level neural-network library. It is capable of running on top of many lower level neural-network libraries such as TensorFlow. The library is released under MIT open-source license. In our application, the Soft Stagewise Regression Network is written in Keras.

TensorFlow

TensorFlow is a highly optimised library designed for graph computations. It was originally developed by Google and is licensed under Apache License 2.0. It serves as a back-end for Keras.

Scikit-learn

Scikit-learn is a general machine learning library. It contains various algorithms such as Random Forests, Support Vector Machines of K-means. It is released under BSD 3-clause license. We used scikit-learn for train, test and validation split during data preprocessing.

Scikit-image

Scikit-image is a image processing library. It includes various utility functions for image manipulation and transformation. It is released

under BSD open-source license. In our application scikit-image is used to compute integral image.

NumPy

NumPy is a Python library for mathematics. It provides support for large multidimensional matrices. It also includes various linear algebra functions. It is licensed under BSD 3-clause license. NumPy array is a primary mode of storing and manipulating with images.

Pandas

Pandas is a Python data analysis library. It contains data structures and functions for manipulating with tables. It is released under BSD 3-clause license. We use pandas during data preprocessing to parse dataset labels.

OpenCV

OpenCV is an open-source computer vision library. It provides functions for image manipulation as well as various computer vision algorithms. It is licensed under BSD license. We use OpenCV for loading, resizing and saving images.

Dlib

Dlib is a general purpose software library written in C++ with a Python interface. The library is released under Boost Software License. We use Dlib as an alternative face detection method.

MTCNN

MTCNN is an implementation of Multitask Cascaded Convolutional Networks which we discussed in Section 1.1. The Python library is released under MIT license. We use MTCNN as a primary face detection method.

Face_alignment

Face_alignment is a Python library for facial landmark detection and face alignment. It is an implementation of Face Alignment Network which we discussed in Section 1.2. We used face_alignment library to filter out faces which are turned away from the camera.

7 Conclusion

We implemented an application for age and gender prediction. The application can be used both as a Python module imported into another project and as a standalone application with command line interface. The application provides two different face detection method. One is based on computing histogram of oriented gradients, the second is based on cascade of convolutional networks.

Additionally the application provides a way to filter faces which are turned away from the camera. We also enable the usage of our application in conjunction with person detection tools by accepting a person id parameter and computing the average result.

To select the basis of our application we evaluated 3 different methods. We then trained our own model that implements the best method using a dataset made by combining all evaluated publicly available datasets and the dataset we manually labelled. The trained model performed better on all dataset in age estimation. The most significant improvement was on FG-NET dataset where mean absolute error improved from 12.85 to 6.61. In gender estimation the model performed better on most of the datasets. On UTKFace dataset it improved from 20.45% error rate to 10.6%.

Furthermore, we evaluated in detail the trained model on the manually labelled dataset. We achieved age mean absolute error of 5.66. In some cases the mean absolute error gets as low as 3.96. The model is also able to correctly classify all male subjects. We demonstrated that our implementation is able to estimate age and gender in a real world situation.

We also trained one age model and one gender model using integral image as an additional colour channel. This did not improve the performance and the results on most datasets were worse that the pretrained models. It is possible that training with integral images requires a different neural network architecture or different set of hyperparameters.

7.1 Possible Future Research

There are many possibilities in age and gender estimation research. An immediate idea would be to look more deeply into training models with integral images as additional colour channels using more varied neural network architectures.

Another idea would be to use more varied neural network architecture specifically for gender prediction. Many current tools use the same architecture for both age and gender prediction.

Bibliography

1. FUKUSHIMA, Kunihiko. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*. 1980, vol. 36, pp. 193–202. Available from DOI: 10.1007/BF00344251.
2. KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (eds.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105. Available also from: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.
3. VIOLA, Paul; JONES, Michael. Robust real-time face detection. *International Journal of Computer Vision*. 2004, vol. 57, pp. 137–154.
4. DALAL, Navneet; TRIGGS, Bill. Histograms of Oriented Gradients for Human Detection. In: *In CVPR*. 2005, pp. 886–893.
5. ZHANG, K.; ZHANG, Z.; LI, Z.; QIAO, Y. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters*. 2016, vol. 23, no. 10, pp. 1499–1503. ISSN 1070-9908. Available from DOI: 10.1109/LSP.2016.2603342.
6. KAZEMI, Vahid; SULLIVAN, Josephine. One Millisecond Face Alignment with an Ensemble of Regression Trees. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2014, pp. 1867–1874. CVPR '14. ISBN 978-1-4799-5118-5. Available from DOI: 10.1109/CVPR.2014.241.
7. BULAT, Adrian; TZIMIROPOULOS, Georgios. How far are we from solving the 2D & 3D Face Alignment problem? (and a dataset of 230,000 3D facial landmarks). In: *International Conference on Computer Vision*. 2017.
8. NEWELL, Alejandro; YANG, Kaiyu; DENG, Jia. Stacked Hourglass Networks for Human Pose Estimation. *Computing Research Repository (CoRR)*. 2016, vol. abs/1603.06937. Available from arXiv: 1603.06937.

9. YOUNG HO KWON; DA VITORIA LOBO. Age classification from facial images. In: *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1994, pp. 762–767. ISSN 1063-6919. Available from DOI: 10.1109/CVPR.1994.323894.

10. ROTHE, Rasmus; TIMOFTE, Radu; GOOL, Luc Van. DEX: Deep EXpectation of apparent age from a single image. In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2015.

11. ROTHE, Rasmus; TIMOFTE, Radu; GOOL, Luc Van. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision (IJCV)*. 2016.

12. EIDINGER, Eran; ENBAR, Roee; HASSNER, Tal. Age and Gender Estimation of Unfiltered Faces. *IEEE Transactions on Information Forensics and Security*. 2014, vol. 9, pp. 2170–2179.

13. LEVI, Gil; HASSNER, Tal. Age and Gender Classification Using Convolutional Neural Networks. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) workshops*. 2015. Available also from: %5Curl%7Bhttps://osnathassner.github.io/talhassner/projects/cnn_agegender%7D.

14. YANG, Tsun-Yi; HUANG, Yi-Hsuan; LIN, Yen-Yu; HSIU, Pi-Cheng; CHUANG, Yung-Yu. SSR-Net: A Compact Soft Stagewise Regression Network for Age Estimation. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 1078–1084. Available from DOI: 10.24963/ijcai.2018/150.

15. GOLOMB, Beatrice A.; LAWRENCE, David T.; SEJNOWSKI, Terrence J. SEXNET: A Neural Network Identifies Sex From Human Faces. In: *NIPS*. 1990.

16. MOGHADDAM, B.; MING-HSUAN YANG. Gender classification with support vector machines. In: *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*. 2000, pp. 306–311. Available from DOI: 10.1109/AFGR.2000.840651.

17. AGUSTSSON, E; TIMOFTE, R; ESCALERA, S; BARO, X; GUYON, I; ROTHE., R. Apparent and real age estimation in still images with deep residual regressors on APPA-REAL database. In: *12th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), 2017*. 2017.

18. CLAPÉS, Albert; BILICI, Ozan; TEMIROVA, Dariia; AVOTS, Egils; ANBARJAFARI, Gholamreza; ESCALERA, Sergio. From apparent to real age: gender, age, ethnic, makeup, and expression bias analysis in real age estimation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 2373–2382.

19. ZHANG, Yunxuan; LIU, Li; LI, Cheng; LOY, Chen Change. Quantifying Facial Age by Posterior of Age Comparisons. In: *British Machine Vision Conference (BMVC)*. 2017.

20. SIMONYAN, Karen; ZISSERMAN, Andrew. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. Available from eprint: `arXiv:1409.1556`.

21. HEURITECH. *Atomic force microscopy*. 2013. Available also from: `https://blog.heuritech.com/2016/02/29/a-brief-report-of-the-heuritech-deep-learning-meetup-5/`. [Online; accessed May 15, 2019].

22. SZEGEDY, Christian; IOFFE, Sergey; VANHOUCKE, Vincent. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *CoRR*. 2016, vol. abs/1602.07261. Available from arXiv: `1602.07261`.

23. SZEGEDY, Christian; LIU, Wei; JIA, Yangqing; SERMANET, Pierre; REED, Scott; ANGUELOV, Dragomir; ERHAN, Dumitru; VANHOUCKE, Vincent; RABINOVICH, Andrew. *Going Deeper with Convolutions*. 2014. Available from eprint: `arXiv:1409.4842`.

24. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. *Deep Residual Learning for Image Recognition*. 2015. Available from eprint: `arXiv:1512.03385`.

25. WOLF, Lior; HASSNER, Tal; MAOZ, Itay. Face recognition in unconstrained videos with matched background similarity. In: *in Proc. IEEE Conf. Comput. Vision Pattern Recognition*. 2011.

26. SZEGEDY, Christian; VANHOUCKE, Vincent; IOFFE, Sergey; SHLENS, Jonathon; WOJNA, Zbigniew. Rethinking the Inception Architecture for Computer Vision. *CoRR*. 2015, vol. abs/1512.00567. Available from arXiv: 1512.00567.

27. KINGMA, Diederik P.; BA, Jimmy. *Adam: A Method for Stochastic Optimization*. 2014. Available from eprint: arXiv:1412.6980.

A Attachments

The attached archive contains source code of the Python application described in Chapter 4 and Chapter 6.