MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Ćirić Stefan** Personal ID number: **439631**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Computer Science**

Study program: **Open Informatics**

Branch of study: **Artificial Intelligence**

II. Master's thesis details

Master's thesis title in English:

Learning CNNs from Weakly annotated facial images

Master's thesis title in Czech:

Učení konvolučních neuronových sítí ze slabě anotovaných obrázků tváří

Guidelines:

Learning Convolutional Neural Networks (CNNs) for face recognition requires large sets of annotated face images. Manualannotationof faces is laborious workandresultsareoftenimprecise. Thegoalof thisproject is toimplement a method that can learn CNNs from a database of weakly annotated images which is created by fully automated process. The method will be used to learn a CNN for age and gender recognition from weakly annotated IMDB database that contains 300k faces of movie celebrities. Each image in the database is annotated by the name, biological age and gender of the captured celebrity. The images can contain multiple faces and it is unknown to which face the annotation should be linked. The proposed method has to establish the missing link between the images and the annotations. The precision of the learned CNN as well as the precision of the automatically crated annotation will be statistically evaluated.

Bibliography / sources:

Franc, Cech. Learning CNNs from Weakly Annotated Facial Images. Image and Vision Computing, 2018.
- Moschoglou et al. Manually annotated age database, CVPR-W 2017.
- Agustsson et al. Apparent and real age estimation in still images with deep residual regressors on APPA-REAL database, FG 2017.

Name and workplace of master's thesis supervisor:

Ing. Vojtěch Franc, Ph.D., Machine Learning, FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **09.07.2019** Deadline for master's thesis submission: _____

Assignment valid until: **19.02.2021**

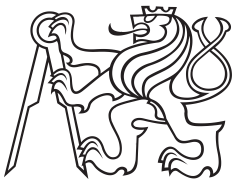
Ing. Vojtěch Franc, Ph.D. Supervisor's signature	Head of department's signature	prof. Mgr. Petr Páta, Ph.D. Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt	Student's signature

Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Computer Science

Learning CNNs from weakly annotated images

Stefan Ćirić

Supervisor: Ing. Vojtech Franc, Ph.D.
January 2020

Acknowledgements

I thank Ing. Vojtech Franc for his continued support and guidance during my work on the the thesis. I thank my friends Božidar, Mihajlo, Vukašin, David and Stefan and Mykyta for assisting me in the annotation. I especially want to thank my dearest Lidija Lukić for cheering me on every step of the way, for her assistance with the illustrations and annotation. Most of all I would like to thank my parents, for being unwavering pillars of encouragement and love, and supporting me fully in my goals.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, 06. Jan 2019

Abstract

Age prediction from facial images is a long standing problem. Various automated facial recognition applications and models have been developed and researched to tackle it. State of the art models for age prediction and face recognition are based on convolutional neural network. The main weakness of such models is the requirement of large annotated datasets. Many state of the art models are learned from automatically annotated IMDB database containing over 500k images. The major weakness of the IMDB database is a large amount of noisy labels produced by the used annotation algorithm.

The thesis aims to develop an annotation algorithm which produces a cleaner annotation of the IMDB database. The proposed algorithm builds identity models of the actors appearing in the database and uses the identity models to assign age to the correct faces, and as such accurately annotate the data. We present the proposed annotation method for building the identity models and the fine tuning of the necessary parameters surrounding the annotation. We evaluate the model on two manually independent datasets, and compute two different types of errors: overlooked and misclassified samples. For the purpose of manual annotation we develop a tool which is used for efficient collection of the labels from the data.

Keywords: age prediction, face recognition, CNN, keras, VGGFace, identity model, IMDB, IJB, annotation

Supervisor: Ing. Vojtech Franc, Ph.D.

Abstrakt

Predikce věku ze snímků obličejů je dlouhodobě studovaný problém, pro který byla navržena řada různých postupů. V oblastech rozpoznávání tváří a predikci věku dosahují v současné době nejlepších výsledků metody založené na konvolučních neuronových sítích. Jejich nevýhodou je nutnost učení z velkých anotovaných kolekcí, které bývá obtížné zajistit. Obvykle se používá automaticky vygenerovaná databáze ze serveru IMDB obsahující přes 500 tisíc snímků, která ale vlivem použitého anotačního algoritmu obsahuje velké množství špatných označení.

Představíme anotační algoritmus dosahující lepších výsledků na IMDB datasetu. Metoda je založená na modelování identit herců v databázi a následném přiřazení věku k obličejům. Kromě metody samotné je také diskutováno optimální nastavení parametrů pro dosažení maximální přesnosti. Za pomoci vlastního nástroje pro ruční anotaci je model vyhodnocen na dvou nezávislých datasetech, a to ze dvou hledisek. Nejprve uvažujeme počet špatně klasifikovaných vzorků, dále také počet osob jimž nebyl přiřazen žádný věk z důvodu chybějícího modelu identity.

Klíčová slova: predikce věku, rozpoznávání obličejů,CNN, keras, VGGFace, model identity, IMDB, IJB, anotace

Překlad názvu: N/A

Contents

Figures

Tables

Chapter 1

Introduction

Age prediction from facial images is an important long standing problem [22, 27, 25]. The need for predicting age from images has prompted various face recognition systems. The automation of such processes could greatly contribute even to our every day lives. Such as updating various database records, which would eliminate the need for changing and renewing identification documents or passports, by having simulated records representing the most recent and accurate appearance. However, the aging process influences the facial landmarks in severe ways, making it difficult, for both human and machine, to predict the age. It is affected by many factors such as health, gender, living conditions and lifestyle. Facial aging essentially represents a new dimension to the face recognition problem.

State of the art methods are based on deep convolutional neural networks, which require a large amount of training data [29, 5, 9, 23]. The models based on CNN require training examples of pairs of input faces and corresponding biological age of captured person.

Due to the difficulty in collecting and labelling sufficiently large datasets to increase the performance, the progress in age prediction seems somewhat slower than in other face recognition tasks for which large datasets are available. The challenge is to introduce a method that can bypass the problem of small databases, or the fact that the automatically created annotation of large datasets is noisy.

Many current state of the art methods use automatically collected IMDB database [29]. The IMDB database is large (500K images), but unfortunately since it has been crawled from both Wikipedia and IMDB, the annotation is noisy, so it is usually used only for pre-training of CNN followed by fine-tuning on accurately annotated data. The effort of collecting a similar database through manual means would be very challenging, as the annotation process is very labor-intensive. One of the primary goals of the thesis is to come up with a clean annotation of the IMDB database.

The thesis tries to develop an annotation algorithm which uses the identity

model of celebrities to assign the age to faces. The main contributions of the thesis are the proposed algorithm for annotation, a tool developed for efficient and easy annotation of the database, and collected manual annotation of a subset of the IMDB database.

Chapter 2

Related methods

There are multiple databases for visual age prediction. A list can be found e.g. in the survey paper [8]. Most of the existing databases are either small to train deep neural networks or they come with significant amount of noisy labels.

The IMDB database has been introduced in [29]. The authors propose a simple heuristic to assign annotation to faces. The images in which the face detector finds a single face, or single dominant face, are assumed to contain the target identity. This annotates around 40% of images while the rest of images is not used. In addition large portion of the labels is incorrect as the single detected faces does not correspond to the annotated identity.

The goal of the thesis to develop a method wastes much less faces and produces cleaner annotation. In [16] the authors propose a method to learn age predictor from the weakly annotated faces contained in the IMDB database. They propose complicated statistical model tailored to the weak annotation of the IMDB database and they learn the model parameters by computationally demanding instance of the Expectation-Maximization algorithm. The method outputs a CNN based age predictor and, as a byproduct a database of face images annotated by age, gender and identity. This thesis aims to develop an cheap alternative approach which will be much simpler yet producing comparable results.

Majority of approaches for automated age prediction use supervised machine learning methods, for example, [22, 18, 12, 19] etc. The supervised methods require precise age annotation for each training face. Learning from weakly annotated data in the context of age prediction is rare. E.g. [31, 10] consider learning from interval age annotation while [17] considers faces annotated in form of age distribution. Weak learning of general classifiers has been studied much more intensively which produced many different approaches [15, 20, 21, 14, 33, 13]. Majority of the existing methods consider a setting when input instances are annotated by a set of candidate labels, assuming that only one is correct. Another related scenario is the multi-instance learning (MIL) [6, 30, 32]. The MIL considers the training instances to come in bags that are labeled rather than having label for each instance.

Database	No. of subjects	Database size	Age range (years)
FG-NET [21]	82	1002	0-69
MORPH [148]	13,618	55,134	27-68
Yamaha gender and age (YGA) [12,68]	1600	8000	0-93
Waseda human-computer interaction technology[134]	26,222	5500	3-85
AI & R Asian [150]	17	34	22-61
Burt's Caucasian Face database [151]	-	147	20-62
Lotus Hill Research Institute (LHI) database [152]	–	50,000	9–89
Human and object interaction processing (HOIP) [11]	300	306,600	15–64
Iranian face database [153]	616	3600	2–8
Gallagher's Web-Collected database [4]	–	28,231	0–66
Ni's Web-Collected database [154,155]	–	219,892	1–80
Kyaw's Web-Collected Database [156]	–	963	3–73
BERC database [214]	95	5910	3–83
3D morphable database [69,157]	438	–	–

Table 2.1: Summary of facial aging databases[7] showing the number of subjects in each database, as well as the database size and the span of years of the participants.

The assumption is that the bag label is correct at least for one of the instance in the bag. None of the existing weak label scenario fits exactly to the problem addressed in this thesis.

Chapter 3

Data

The collected datasets which we were dealing with are the IMDB [29] and IJB[2]. The IMDB dataset spans over 850 000 faces.The IJB data has less, about 97 548 faces.

The IMDB dataset has been automatically generated, hence it introduces a type of weak annotation. In other words, the images gathered for each of the celebrities in the database may, or may not, be associated to the celebrity. Furthermore, we do not have any labels for the detected images. This means that we do not know for any particular image whether the celebrity is contained. And if it is contained, we do not know which of the detected identities belongs to the celebrity.

The automated process of generating the IMDB dataset produces some information for each of the images. Namely, it assigns an age, gender and proposed identity to each image. However, each image can have multiple faces, which are detected by a face detector, so a single image can be described by multiple entries each having the same labels for age, gender and name, but different labels for the position of the face it is describing. The challenge is to link the annotation with the correct face. This representation of the dataset holds for IMDB, however, in the case of IJB, which is fully annotated, we additonally have a label for the correct face in the image. We use it to generate the weak annotation of the same kind as presented in IMDB.

Since the IMDB database is weakly annotated, we use subsets of the data to perform various tests. Namely we split the data in several parts 3.1

3.1 Properties of the data

Due to the high sparsity in the amount of images per celebirty in the IMDB database, it was necessary to decide how to select the best candidate images for the identity models. In the best case scenario there would be the correctly annotated faces which we would build the model from. Since that would require the manual annotation of the database, we opted for a method which would mix in the least amount of incorrect images. So to make the identity

Dataset	Description	Size
imdb_small	A subset of 100 celebrities	12 000
imdb_full	The whole imdb database	860 000
imdb_annotated	A small subset of only correctly annotated images	7765
imdb_annot_trn	A subset of all images from celebrities which can be found in the imdb_annotated set	401 889

Table 3.1: Various subsets from the data used for the algorithm tuning

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	file_name	celeb_name	age	gender	top_left_	top_left_	top_right_	top_right_	bot_right_	bot_right_	bot_left_	bot_left_	ground_truth
2	42/nm0704	'Lee' George Quinones	23	M	562	567	1008	567	1008	1013	562	1013	-1
3	48/nm0946	'Weird Al' Yankovic	43	M	139	47	184	47	184	92	139	92	-1
4	48/nm0946	'Weird Al' Yankovic	29	M	208	119	378	119	378	288	208	288	-1
5	48/nm0946	'Weird Al' Yankovic	29	M	277	203	413	203	413	339	277	339	-1
6	48/nm0946	'Weird Al' Yankovic	29	M	548	124	702	124	702	277	548	277	-1
7	48/nm0946	'Weird Al' Yankovic	51	M	239	628	880	628	880	1270	239	1270	-1
8	48/nm0946	'Weird Al' Yankovic	55	M	823	197	1036	197	1036	410	823	410	-1
9	48/nm0946	'Weird Al' Yankovic	55	M	557	181	772	181	772	396	557	396	-1
10	48/nm0946	'Weird Al' Yankovic	55	M	1088	164	1338	164	1338	414	1088	414	-1
11	48/nm0946	'Weird Al' Yankovic	55	M	82	368	158	368	158	443	82	443	-1
12	48/nm0946	'Weird Al' Yankovic	55	M	648	247	857	247	857	457	648	457	-1
13	48/nm0946	'Weird Al' Yankovic	53	M	1103	332	1276	332	1276	506	1103	506	-1
14	48/nm0946	'Weird Al' Yankovic	53	M	364	312	578	312	578	527	364	527	-1
15	48/nm0946	'Weird Al' Yankovic	54	M	928	489	992	489	992	554	928	554	-1
16	48/nm0946	'Weird Al' Yankovic	54	M	1057	451	1132	451	1132	526	1057	526	-1
17	48/nm0946	'Weird Al' Yankovic	54	M	618	464	684	464	684	531	618	531	-1
18	48/nm0946	'Weird Al' Yankovic	54	M	499	518	562	518	562	581	499	581	-1

Figure 3.1: The data represenation as shown in Excel. Ordered by columns, starting with the file name, celebrity name, age, gender, the bounding box specifications and the ground truth labels describing the correct detected face in the image (1 represents the correct label, 0 an incorrect label, and -1 means that the label has not been input, i.e. it is unknown)

model we were taking only images associated to the celebrity which had a single face detected on them.

The database was created automatically, with the goal of associating the images to the celebrity. So while there definitely are mistakes, the likelihood of an image being wrongly associated should exponentially increase with the number of persons present on it.

In Figure 3.2 we can see that the disparity in number of images is quite high. In fact most of the celebrities fall in the range from 0 to 50 images. To get a better idea of the data we split the graph into two parts, the celebrities which would have under 50 images associated to them, and those that would have more.

Over almost 90% of the celebrities fall in the range of 0 to 50 total images. That means that a smaller subset of those would be the single detection images. So, it could happen that a celebrity would have no single detections present in the database, which would essentially prevent us from building the model for that person. In fact, there were about 2100 identities which

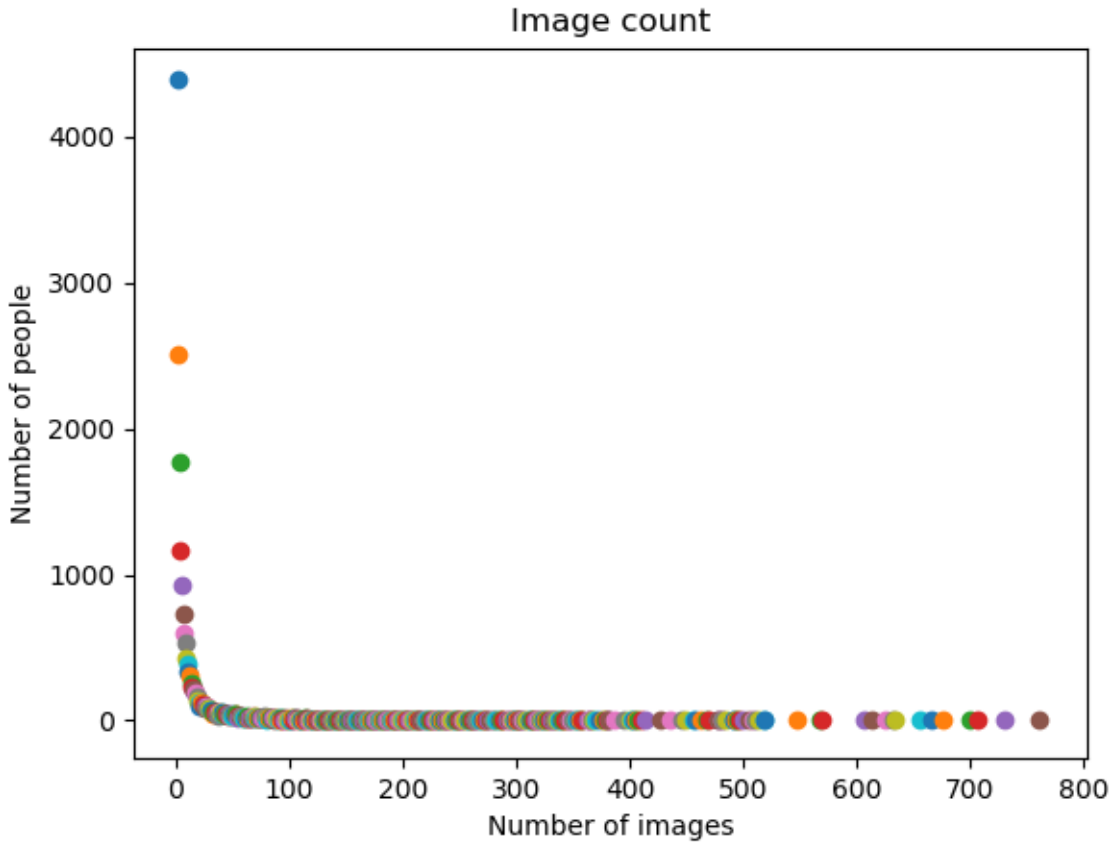


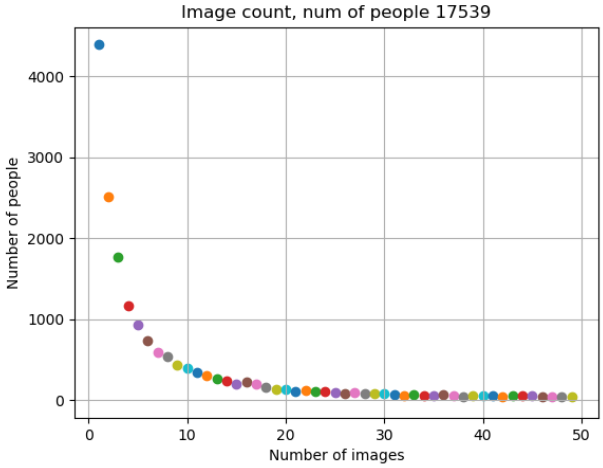
Figure 3.2: The number of images associated with each celebrity in the database

had 0 single detections. Under such circumstances we were using all of the images associated to the celebrity to make the models, meaning every face detected in the images would play a factor in building the model. As long as the celebrity is the dominant face among the collected images the resulting model would be sufficiently good. The reason for this is that when creating the model if we were to extract the features from all of the images, combined them and compute the median of coordinate-wise pairs, the model would lean towards the identity which appears the most times in the images.

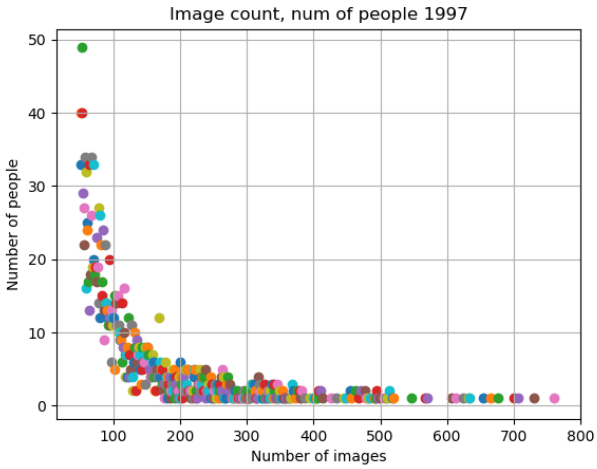
Additionally, it is relevant to know the distribution of the amount of single detection per celebrity. As described this greatly influences the model, and at the same time the accuracy. In Figure 3.4 we observe that the vast majority of the collected data has a very small amount of single detections. Most of them have only a single image. To this end, we evaluate different restrictions when building the model. Namely, we differentiate between choosing, or discarding, celebrities which have less single detections than some fixed threshold.

In 3.5 we can see that most of the celebrities are in their mid thirties. When

assessing the gender of the celebrities, the male population is slightly higher than the female. There are 11068 recorded male celebrities, and 7306 female celebrities. A small part of the celebrities have an undefined gender, only 1162. The distributions of age and gender play a role when we are choosing the celebrities for evaluating our model, or when picking samples for manual annotation. To avoid any bias towards genders, it was necessary to represent both of the genders as equally as possible.



(a)



(b)

Figure 3.3: The number of images associated with celebrities which have under and over 50 images. Figure (a) shows the image count of celebrities which have less than 50 total images in the database. Figure (b) shows the image count of celebrities which have more than 50 total images in the database.

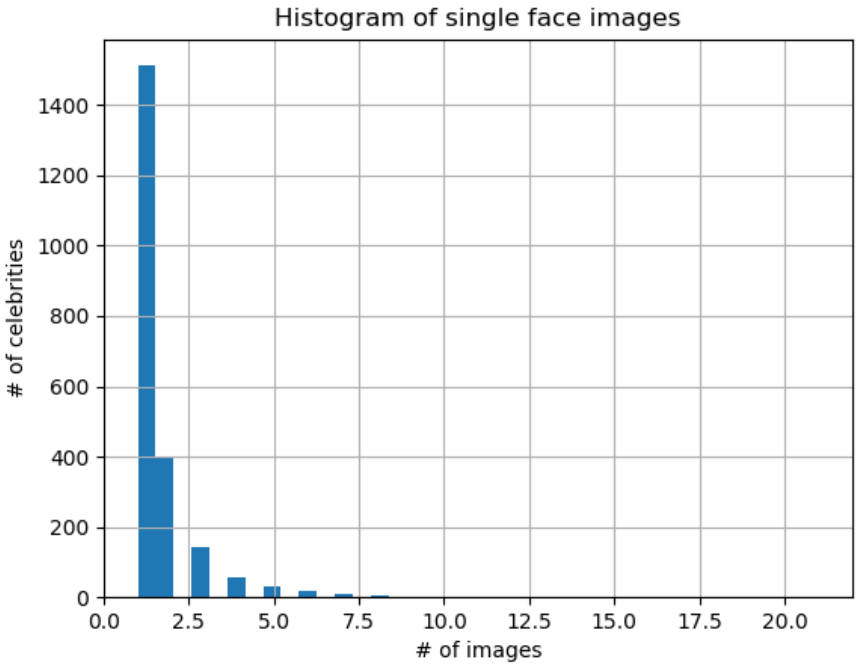


Figure 3.4: Histogram of single detections

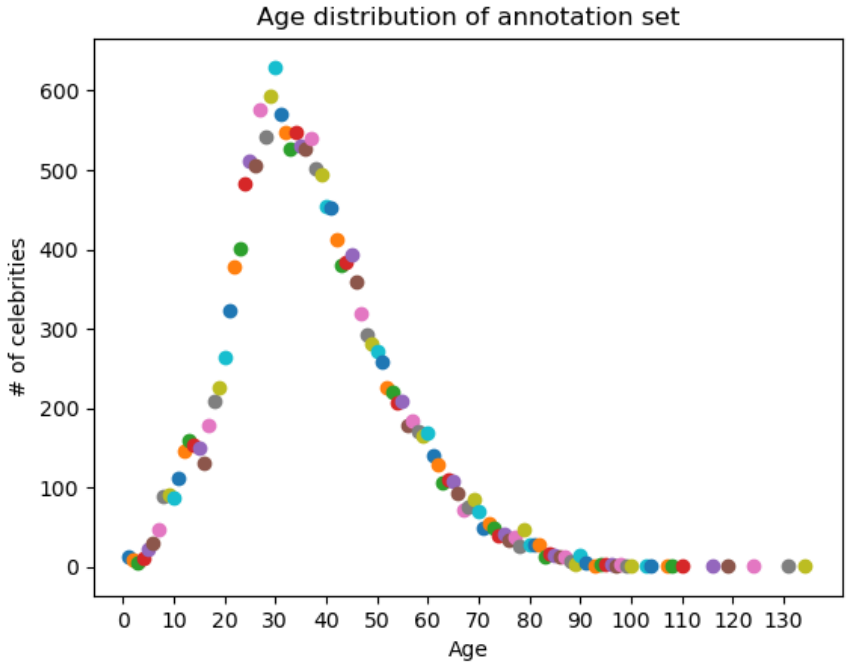


Figure 3.5: The age distribution of the celebrities in the IMDB dataset. On the X curve the range of ages is represented. The Y curve how represents the number of celebrities of that age.

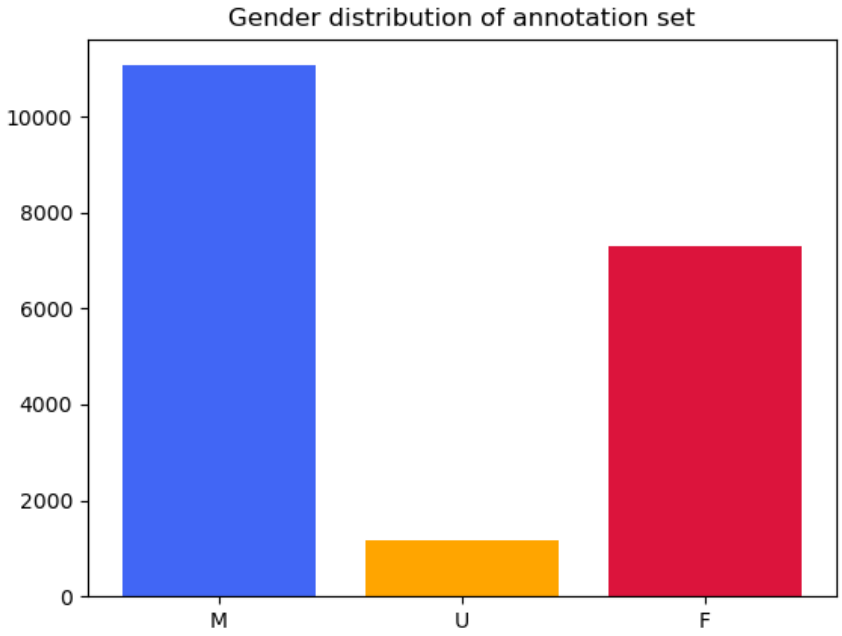


Figure 3.6: The gender distribution of the celebrities found in the IMDB dataset. Representing how many of the celebrities are identified as male, female or undefined.

Chapter 4

Method

4.1 Problem definition

The input is a set of images containing faces. Each image is annotated by biological age, gender and identity of a celebrity. The faces in each image are found automatically by a face detector. It may happen that the annotated identity is not among the detected faces due to face detector failure or due to incorrect annotation. The task is to associate the annotation (age, gender, identity) with the correct face or to decide the face is not contained in the image.

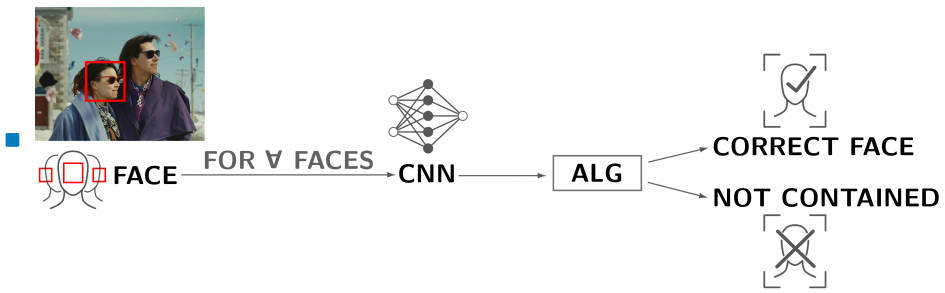


Figure 4.1: Brief illustration of the annotation process.

- Let the training set T have $\mathbf{N}$ samples, where each sample τ^j ($\mathbf{j} \in \mathbf{N}$) is a tuple $(X, \ell, \mathbf{n})$ describing a single image. The X is a vector which holds the extracted facial images from the original image $X = (\chi^1, \chi^2, \ldots, \chi^m)$ where each unique person found by the face detector. The ℓ is the label representing the age and gender of the identity, and the $\mathbf{n}$ is the name of the identity.

4.2 Annotation method

To annotate the images the idea is to build a model of each identity by exploiting the fact that, for each identity we would have multiple images on

which the person is captured. The identity model is obtained by combining carefully chosen images, and then combining distinctive features of the images into a model, by means of a median. The identity model is then used to determine whether the identity is contained in the image, or to declare if it is not. If the celebrity is present, then the appropriate face would be linked with the annotation.

To acquire an annotation of a certain face found on an image to a specific celebrity, we need to pre-process the data and compute an identity model before hand, for the person we are looking to associate the face with. When we're building the model of a celebrity we have two hyper parameters which we need to take into account.

First, since the faces extracted by the face detector could be insufficiently big, we might need to expand the borders of the cropped face, i.e. the margin of the face Θ.

Second, once we have the obtained the identity model, we need to be able to evaluate. When we are checking to see if a specific face belongs to the celebrity. Let Φ be a threshold, or score, to which we compare the difference between the model and target face. If the value is greater, i.e. the difference between the computed features of the face and model are higher than Φ, we reject the face. Otherwise, we accept.

4.3 Algorithm

When the algorithm is deciding whether two faces belong to the same person, it computes the distance of the features between them and compares it against the classification threshold. However, when we are trying to determine whether a certain image has a celebrity, we require the model of the celebrity to compute the distances against the features of the detected face(s).

There are 5 cases when the algorithm is detecting whether an image contains the target celebrity. The algorithm is evaluating whether "Al Pacino" is present on the images 4.2, visualizes what happens when it makes a correct or wrong decision.

By obtaining the best margin we can build the model for each celebrity. To compute the model we used the "IMDB_annot_trn" set, so we would only have the celebrities which are also in "IMDB_annotated", because we would like to evaluate the model later on that subset. For the feature computation we only use the images with single detections of each celebrity. This brings an additional parameter, the amount of single detections that are sufficient in order to acquire the model, and we can see that by increasing the minimum amount of single detections, we can increase the accuracy by discarding those

Figure 4.2: Algorithm detection cases

celebrities that do not have enough images. To obtain the model we compute the median of features of all single detection images assigned to a celebrity.

4.4 Transforming the data

Multiple faces found in the same image would have the same file name, and celebrity name associated to them. The only distinctive part would be the positions of the margins extracted from the face detector. To traverse the data in an easier and more coherent manner, we associate each unique image name to the celebrity it belongs to. Then we associate to that image each unique face found on the image, i.e. each line in the data file that has the same filename and celebrity name. We store the information of each face in a small wrapper class "Info". The object essentially represent each unique line in the file describing the images.

Once we have loaded the data into a more optimal medium for operation, we need to store some metadata as well. Mainly we want to know the number of single detection images for each celebrity, i.e. the images where only one face has been detected by the face detector. It is also beneficial to know the number of images the celebrity possibly appears in. This information is later used to ease the choice of suitable candidate images for the model computation, as well as to speed up the entire process.

4.5 Hyper-parameters

The hyper-parameters which are necessary for the algorithm to operate are the bounding box ratio of images and a classification threshold.

The bounding box ratio $\mu \in [0\%, 100\%]$ is the relative amount by which we need to extend the borders of the detected face before passing it to VGGFace CNN. The face retrieved from the face detector may not cover a sufficient part of the face for feature extraction. By extending it we are able to get more accurate descriptors.

The classification threshold θ is used to decide whether a face belongs to a certain person. Alternatively if we are trying to determine whether two faces belong to the same person. When evaluating a face we require the feature descriptor of the target face, and either the identity model of the person we're comparing it to, or a feature descriptor of a different face. We calculate the difference between the feature descriptors and subtract the classification threshold. If the difference of the features is lower than the threshold we accept the clasification. Otherwise, we would reject.

4.5.1 Tuning the margin

In order to get the best results we need to choose an optimal size of the extracted faces which will yield the highest descriptive features. One of pitfalls with the face detector is that it might not capture the whole face. The detected faces from the face detector are all of the same size. We want to increase the retrieved bounding box by some margin. If we increase too little we won't increase the information gain from the cropped face, and similarly if we increase it by too much we will decrease the information extracted from the face.

To tune the margin we used the IMDB_small database for the initial overview of its behaviour, and later the IJB-A database. The main idea of tuning the margin is by generating pairs of faces. Each such face pair would have a label denoting whether the two faces belong to the same identity or not. Then to evaluate the margin we would compute the accuracy on the face pairs, and pick the margin which minimizes the error.

Let P denote a vector of face pairs, where a face pair is $\rho = (\sigma_A, \sigma_B, \iota)$. σ_A and σ_B (A and B are unique faces from the database) are instances of a wrapper object containing all of the information of the face, such as bounding box, age, gender and name, and ι is a boolean value describing the relationship of the faces, i.e. if they belong to the same identity. We split P into two parts, P_{trn} and P_{tst}, by 70% and 30% respectively, such that:

Let M be a vector of candidate margin values = [-0.4, -0.2, 0.0, 0.1, 0.2, 0.3, ... , 1.0]. For each m $\in M$ we compute the feature descriptors of the faces A and B in $\rho_{i,trn}$.

$$\begin{aligned} X(A) &= \frac{Z(\rho_{trn}[\sigma_A])}{||Z(\rho_{trn}[\sigma_A])||_2} \\ X(B) &= \frac{Z(\rho_{trn}[\sigma_B])}{||Z(\rho_{trn}[\sigma_B])||_2} \end{aligned} \tag{4.1}$$

Z represents the pre-trained CNN which computes the feature descriptor of a a face, $X(A)$ and $X(B)$ represent the normalized feature descriptors. Once we compute them for a pair, we compute the normalized difference between them.

$$dist(A,B) = ||\frac{X(A)}{||X(A)||_2} - \frac{X(B)}{||X(B)||_2}||_2 \tag{4.2}$$

For each distance computed between the faces in a face pair, we store it paired with the label from that face pair into a new vector K.

$$K = [(dist(A_1,B_1),\iota_1),(dist(A_2,B_2),\iota_2),...,(dist(A_n,B_n),\iota_n)] \tag{4.3}$$

Once we are done with all of the face pairs in P_{trn}, we order the values in K such that:

$$\begin{aligned} &\forall \kappa_i \kappa_j \in K, i < j \\ &\quad \kappa_i[dist] < \kappa_j[dist] \end{aligned} \tag{4.4}$$

4.5.2 Tuning the classification threshold

Apart from the bounding box margin, the other hyper-parameter which is necessary to decide the label of faces is the classification threshold, i.e. the maximum distance allowed between two faces for which we decide whether they are part of the same class. That is, whether they have the same label.

First, we need to differentiate the errors, the overlooked and misclassified errors which can occur 4.2, to get a better understanding of what type of problem is associated with the different thresholds. Additionally, we plot a histogram of the pre-computed face scores, from which we will pick our best threshold 5.3. It is necessary to note that the best threshold μ may vary depending on the type of result we want to obtain. If we want to ensure that the % of misclassified samples is not more than 5%, the best choice may well be e.g. μ_i, however, if the desired result is to have the % of overlooked samples be less than 7%, that may be μ_j, where $\mu_i \neq \mu_j$.

While we were tuning the margin, we were computing a new classification threshold every time. However, this threshold is depends on the margin we were tuning. While that threshold does produce the lowest total error, i.e. we weren't differentiating between the types of errors the annotation algorithm could make, we need to independently compute a new threshold to evaluate the algorithm. We were tuning the classification threshold on the previously manually annotated part of the database Table 3.1. The reason for this is that during the computation of the errors we needed to know whether each image contained the real labels of the faces, and whether the ground truth celebrity was contained. Then we could check the decisions which would have been made by the algorithm and the scores computed, and properly evaluate them.

From the 5.3 we observe that the majority of scores computed fall between the range [0.6, ... , 0.8] $\subseteq M$. We introduce two variables *overlooked_samples* and *misclassified_samples* instantiated to 0 for each of the $\mu_i \in M$. Then for each μ_i we go through the annotated part of the dataset. For each image I_i = $[f_1, f_2, ..., f_n]$, we need to check whether the identity is contained in the image. If it is, we need to mark the appropriate face describing the ground truth face as f_{gt}.

$$f_{gt} = \begin{cases} f_{i,j} \in I_i : f_{i,j}[label] = 1 \\ \\ \emptyset \end{cases}$$

Then we need to mark the face chosen by the algorithm. In other words, from the faces present in the image, we pick the face which has the lowest score, i.e. it will be the closest to the identity model of the celebrity. We will denote the face chosen by the algorithm as f_{alg}.

$$f_{alg} = \underset{score}{argmin} f_{i,j} \in I_i \tag{4.5}$$

When both of the faces f_{gt} and f_{alg} have been marked we compute the difference Y between the scores and proceed with the evaluation of errors.

$$Y = sgn(f_{alg} - \mu_i) \tag{4.6}$$

The number of misclassified samples is equal to the number of samples where the algorithm either chooses some face $f_{alg,i}$ in an I_i, and the ground truth is present, but it does not match the chosen face. Or the algorithm chooses a face, but the ground truth face is not contained.

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Figure 4.3: The layers of the VGGFace CNN. The FC layers are listed as convolutions. For each layer the stride, padding, number of filters and filter size are indicated. The image is from the original VGG Face paper [26]

$$misclassified\_samples = \sum_{i=1}^{n} [f_{alg,i} \neq f_{gt,i} : Y_i < 0, f_{gt,i} = 1 \ \ \& \ \ Y_i < 0 : f_{gt} = \emptyset] \tag{4.7}$$

The number of overlooked samples shares the case with misclassified samples, when the algorithm detects a face, but it does not match the ground truth face. But also it includes the case when the algorithm does not choose a face, because they were below the boundary, and yet the ground truth face was contained in the image.

$$overlooked\_samples = \sum_{i=1}^{n} [f_{alg,i} \neq f_{gt,i} : Y_i < 0, f_{gt,i} = 1 \ \ \& \ \ Y_i > 0 : f_{gt} = 1] \tag{4.8}$$

4.6 Framework

To develop the algorithm for computing identity models, we were using one of the pretrained CNN's,[26], developed by Oxford's renowned Visual Geometry Group[1]. It is available in the keras-vggface module[4]. It is a implementation in python with the Keras framework. The goal of VGGFace was to provide accurate face recognition from photographs or sets of faces using the very deep architecture of convolutional neural networks. The VGGFace network Figure 4.3 is comprised of 11 blocks. Each such block is followed by one or multiple non-linear layers, such as max pooling or ReLu layers. One of the main properties of the network is that the filter size matches the input data. This allows the networks' filters to discern the data from the whole image. The input to all of the networks is of size 224x224.

To perform various image operations and transformations, we opted for the OpenCV module. It is an open-source library for various computer vision and machine learning tasks. While the functionality of the library is fairly

Layer type	Configuration
Output	Distribution over Y outputs
Soft-Max	
Convolution	filt: Y, k: 1x1, s: 1, p: 0
ReLU	
Convolution	filt: 2048, k: 1x1, s: 1, p: 0
ReLU	
Convolution	filt: 128, k: 5x5, s: 1, p: 0
ReLU	
Convolution	filt: 128, k: 4x4, s: 1, p: 0
ReLU	
Convolution	filt: 128, k: 3x3, s: 1, p: 0
MaxPool	2x2, s: 2, p: 0
ReLU	
Convolution	filt: 64, k: 3x3, s: 1, p: 0
MaxPool	2x2, s: 2, p: 0
ReLU	
Convolution	filt: 64, k: 3x3, s: 1, p: 0
MaxPool	2x2, s: 2, p: 0
ReLU	
Convolution	filt: 32, k: 3x3, s: 1, p: 0
ReLU	
Convolution	filt: 32, k: 3x3, s: 1, p: 0
Input	100x100 gray-scale image

Table 4.1: Configuration of the CNN used to predict label from a facial image. The second column describes the number of filters 'filt', the filter size 'k', stride 's' and padding 'p'.

vast, spanning between classifying human actions, tracking human actions and removing red eyes from images, it has a lot of useful image processing tools. Mainly the options to perform quick affine transformations to your data [11].

Aside from the the aforementioned modules, we used the standard machine learning modules such as numpy, pickle, scipy and defaultdict for processing, storing and transforming the data. When computing the features and models, due to their potential large sizes, it is recommended computing them up front and storing them with the pickle module. Then when running the algorithm we can fall back on the already precomputed variables.

To evaluate and check the quality of the results we learned a CNN for predicting the age. The accuracy was tested in terms of two errors, Mean Absolute Error and CS5. The detailed configuration of CNN and its filters was described in [16]. The detailed configuration of the described CNN is described

Figure 4.4: Layers of the VGGface CNN from [26]

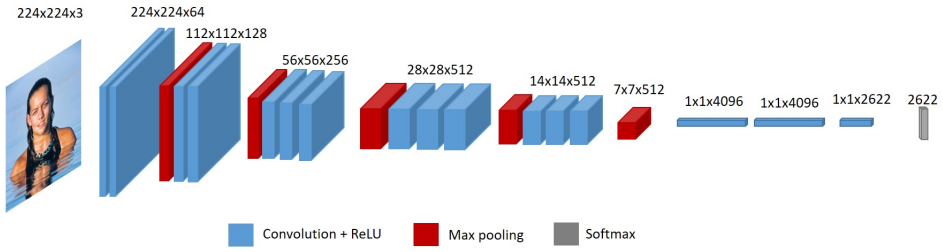


Figure 4.5: Visualization of the deep architecture of VGG.

4.7 Identity models

In order to create our identity models it is necessary to use the aforementioned keras VGGFace CNN.

The network 4.4 has 11 layers and uses 3x3 convolution and 2x2 pooling throughout the entire network. This demonstrates that the depth of the network is very important in terms of results.

4.7.1 Using keras-vggface

Keras-vggface is a pre-trained CNN for identity recognition [26]. Before using the network, we need to download the weights of the network. We can choose between different pooling options (none, avg or max). The models can be used either for feature extraction and prediction. Additionally, we can include custom parameters and fine-tune the network by training the model again. In the proposed method, we use the pre-trained network with the downloaded weights to compute the features of faces.

The CNN returns a vector $\upsilon \in \Re^{1x512}$ which is the feature descriptor of the input face. The features describe the identity with age and gender. Next we need to normalize the data by the norm to get a meaningful descriptor of the face.

Let I be an image with $\mathbf{n}$ detected faces $= \iota_1, \iota_2, ..., \iota_n$, Z be the CNN.

Then a feature descriptor ϕ_j is equal to χ_j where:

$$\phi_j = \frac{Z(\iota_j)}{||Z(\iota_j)||_2} \tag{4.9}$$

The feature descriptors are used to determine the similarity of two faces. For any two faces we want to compare, we compute their feature descriptors eg. ϕ_1 and ϕ_2. We subtract the feature descriptors to obtain their difference. This difference represents the how different the faces are. The higher it is, the less likelihood of the two faces belonging to the same person.

4.7.2 Building the model

In this section we describe the process of computing an identity model for a single celebrity. We deal with two main cases. First, the celebrity has a sufficient amount of single detections in the dataset. In this case, the model is computed as a coordinate-wise median from the identity features previously extracted by VGGFace from the detected faces. Second, the celebrity has less than the required amount of single detections. In this case, we either discard the celebrity, i.e. we do not compute the model and skip it. Or, we build the model by computing the feature descriptors for all the faces detected in all of the images we have associated with the celebrity, and then taking the coordinate-wise median to obtain the model.

In order to complete the model we need to compute and store all of the feature descriptors of our candidate images. Let Γ be a set of ϕ^{ij} where i denotes the image in the candidate set A, and j denotes a specific face in image α_i.

Ideally there should be only one face associated to each image. In case we do not have single detection images, or an insufficient amount of them, we would need to take additional images with multiple faces.

When all of the faces have been processed by the CNN, to build the model we need to compute the median value of each of the values present in the feature descriptors.

Let

$$\phi = [\rho^1, \rho^2, ..., \rho^{512}] \tag{4.10}$$

and the identity model

$$\Pi = [\hat{\rho^1}, \hat{\rho^2}, ..., \hat{\rho^{512}}] \tag{4.11}$$

where

$$\hat{\rho^j} = median(\rho_1^1, \rho_2^1, \rho_1^2, ..., \rho_l^k) \tag{4.12}$$

assuming that all of the ρ present in the median function all point to the indice **j**. Then **k** denotes the image, and **l** represents the feature descriptor of the face in the image.

What we get is a vector $\Pi \in \Re^{1x512}$ representing our identity model. This model is later used to decide whether some face belongs to the celebrity. This is done by computing the difference between the identity model and the feature descriptor of a face we are trying to annotate.

Chapter 5

Experiments

In order to find the optimal hyper-parameters which produce the most accurate feature descriptors, as well as the best accuracy in classification, we were tuning the parameters in the following way. We evaluate the model on two different datasets, the IMDB_annotated, and on the manually annotated data we collect. We present the results in terms of misclassified and overlooked errors, and what is their trade off. The accuracy is computed for different minimal requirements on the single detections. Additionally, we evalute the accuracy by training a age prediction CNN and compare the results with the single-detection heuristic.

5.1 Bounding box margin

First, we extracted all of the images with only single detections and grouped them by celebrities, i.e. for each celebrity we had a subset of only images with a single face on them. Once we had all the images we split them into two equal parts for the imbd_small dataset, and the IJB-A dataset by 70-30 into train and test parts.

For each set of images we created FacePair objects, whose purpose was to keep information of two images (faces) and also a label denoting whether the images were extracted from the same celebrity (face1, face2, label). For each celebrity we created, at random, 25 positive and negative instances. A positive instance represents two different faces extracted from the same identity. A negative instance represents a face from the target celebrity, and a random face that appears in images associated with the celebrity. The idea is that by tuning the margin on the negative instances from the identities which occur in the images associated to the celebrity, we could increase the accuracy when later evaluating the identity model. However, in the case the celebrities would not have enough identities that would be present in the images, we would randomly collect from the remaining images in the database.

In the training part we went through all the face pairs computing the distance between their feature descriptors. We then stored each distance with the class of the two faces. Once we went through all of the distances from

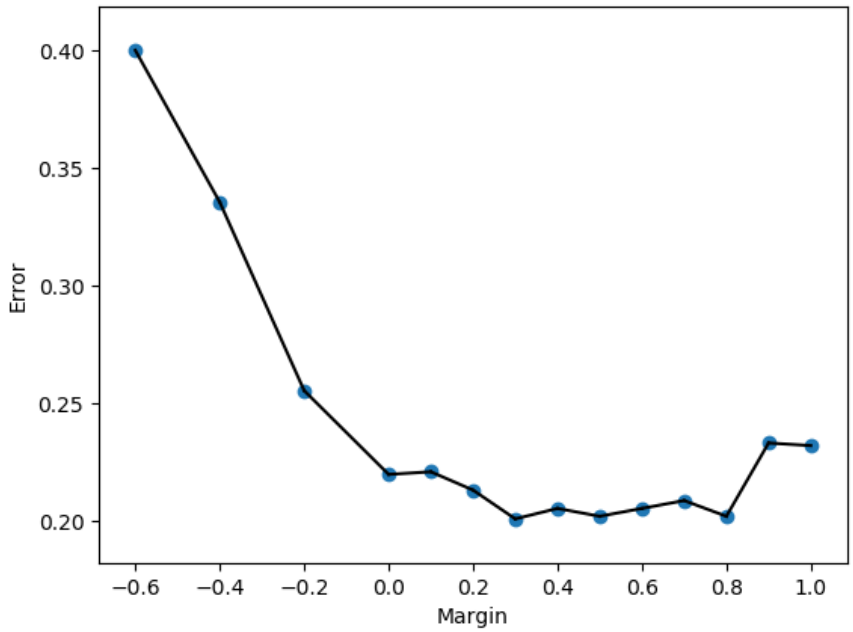


Figure 5.1: Plot of the error on the dataset of one face images for different margin.

the training examples we could compute the optimal classification threshold, i.e. the one which would best separate the negative and positive classes.

To compute the best threshold we initialized two vectors: positive_error and negative_error, each equal to the size of samples. Each entry in the positive_error was initialized to the number of samples, while the negative_error to zero. Then going through the sorted distances of the face pairs, if the distance was associated with the positive class we would decrease the positive_error by one for that distance and leave the negative_error the same. Similarly for the opposite case. This would clearly show the relation between positive and negative errors made by the algorithm based on the distance. For the smallest distance (score) there would be the most positive errors because we would accept all of the faces in the classification. In the end we would use the threshold which has the smallest sum of both the positive and negative errors.

For the IMDB_small database 5.1 the smallest error (0.2011%)is obtained for margin 0.3.

Then, to obtain the best classification threshold for the previously chosen m, we initialize two vectors, namely "positive_error" (α) and "negative_error" (β) $\in \Re^n$. We initialize all entries in the "negative_error" to 0, and all entries in the "positive_error" to n.

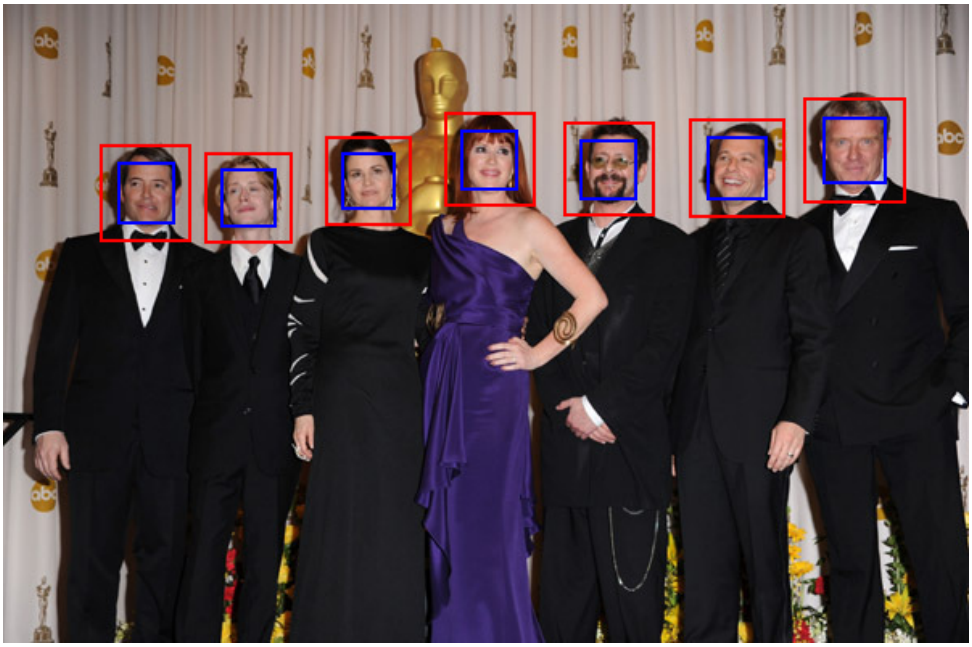


Figure 5.2: Bbox detected by face detector (blue), and optimal bbox from algorithm (red)

Since K has been ordered by the score, the entries at e.g. j $\in K$ in α and β correspond to number of positive and negative errors made for the distance $\kappa_j[dist]$. The positive errors represents the number of samples which would be overlooked for a specific distance. In other words the threshold for accepting the face would be so low, that we would not select any face as the correct one.

Similarly, the negative errors corresponds to the number of mislcassifications that would occur for that distance. That is, the threshold would be so low that since we would not choose any face, we would not have the opportunity to misclassify any of the faces, hence the number of misclassified samples would be very low.

The idea is that by increasing the distance, i.e. the classification threshold for accepting or rejecting the faces, we would incrementally decrease the number of positive errors, but eventually increase the negative error. We are looking for the distance which would minimize both of these values.

So to update α and β we traverse through all the recorded samples in K. For each record we check if the label ι is positive (True) or negative (False). If the label is positive, meaning both of the faces do belong to the same person, we would decrement the positive error for that entry, i.e. classification threshold. Alternatively, if the label is negative, meaning the two faces do not belong to the same identity, we would increment the negative error.

The best classification threshold μ w.r.t. the chosen m is the one that minimizes both of the errors. Once we obtain it, we then use it to compute the real error on P_{tst}. For each $\rho_{tst} \in P_{tst}$ we compute the $X(A)$ and $X(B)$ and we subtract the classification threshold μ. If the value is less than 0, we accept the face, otherwise we reject. If we accepted the face and the ι of the $\rho_{tst}[\iota]$ were not equal, we increment the total real error. Otherwise, if we reject, and the ι was equal to $\rho_{tst}[\iota]$ we also increase the error.

$$\iota_{\mu}(A, B) = sgn(dist(A, B) - \mu) \tag{5.1}$$

We plot the computed errors for each of the m in a single graph 5.1. We observed that in fact the margin detected by the face detector covers an insufficient surface of the faces, i.e. it misses some of the facial landmarks necessary for a more accurate classification. After some point (e.g. 70%, 80%) by increasing the margin the error starts increasing, as the facial descriptors become too general. This is due to the majority of the cropped face becomes the surrounding background and possibly other people which are present in the image. From the plot we gathered that margin should be increased by 30% to obtain the most accurate results. With the deduced margin we performed the remaining part of our experiments.

5.2 Classification threshold

From Figure 5.3 we can see the histogram of all the computed distances from the dataset. The histogram represents the smallest, and highest gap in similarity between the faces of the model, and the faces associated to the celebrities.

When we are looking for the best classification threshold, we are observing three cases 4.2(2,3 and 4), and we compute the errors according to them. If the ground truth face is contained in the image, we check whether the algorithm chooses the face or not. And if it does, we check if the face deduced from the algorithm matches the ground truth annotation. If the face is not contained in the image, we inspect whether the algorithm chooses a face.

From the plots 5.6 we can see the way the overlooked and misclassified errors behave when we slide the boundary for deciding faces along the computed distances of the faces. The two errors intersect 0.72.

These errors are obtained by computing the identity models from single detections of the celebrities, without a minimum requirement for the number of single detections. Which means that if the celebrity had only one single detection, that image would solely present the identity model. We can improve the result by imposing a minimum requirement on the number of

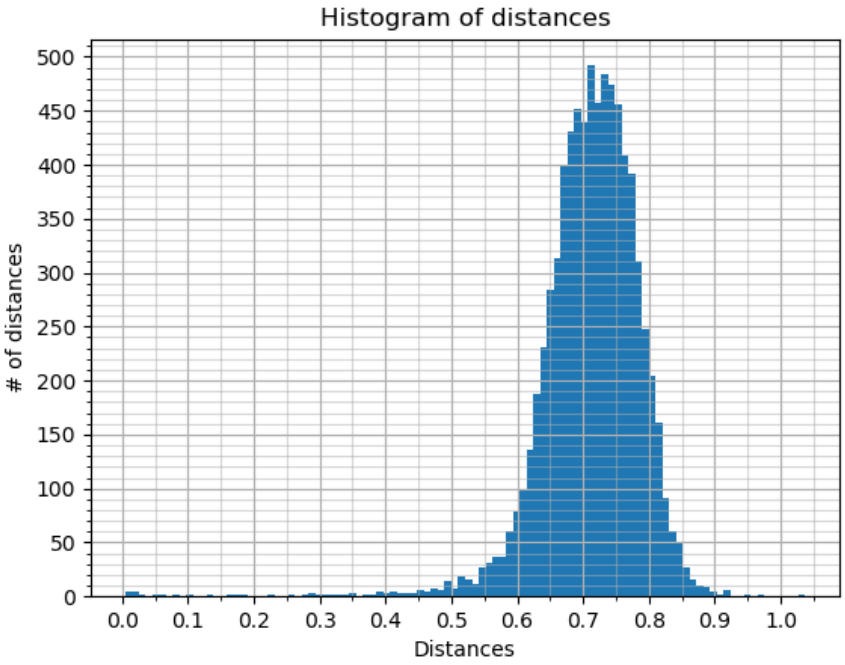


Figure 5.3: Histogram representing the distances between the models and features of the annotated samples.

single detections required for computing the model, and discarding those celebrities which do not meet the requirement. In the 5.6 the models were computed for all of the celebrities, i.e. a total of 3253 celebrities. If there were no single detections recorded for the celebrity, the model would be computed as a median of all the images associated with the celebrity.

In the following two examples, 5.7 and 5.8 we were discarding the celebrities which didn't have at least 10 or 50 single detections recorded respectively. In the first example point of intersection falls down to 0.73, and the number of samples included to 2786. In the second, the point of intersection is at 0.72, and the number of included celebrites to 1498.

5.3 Evaluation based on results from CNN

The proposed method is used to collect training faces from the weakly annotated IMDB database. The proposed method identifies a single face that is most similar to the annotated identity from each image in the database. Only those faces whose similarity to the respective identity template is above a fixed threshold are included in the training set. We varied the similarity threshold so that $25\%, 50\%, \ldots, 100\%$ portion of images is included.

Additionally, in order to guarantee the quality of the annotation, we filtered

num of examples	181k	75k	167k	258k	275k	310k	340k
used images [%]	-	25	50	75	80	90	100
MAE	7.18	8.04	6.92	6.92	6.92	6.97	7.00
CS5	0.48	0.43	0.49	0.49	0.49	0.49	0.49

Table 5.1: Comparison of the proposed annotation algorithm and the single-detection heuristic. The first row shows the number of annotated faces the algorithm produces. The proposed method is set to harvest faces from a given portion of images (second row). The produced training faces were used to learn a CNN predicting biological age. The accuracy was measured on separate AgeDB dabatse. The prediction accuracy measured in terms of MAE and CS5 is shown in the third and the fourth row, respectively.

out identities with less than 5 images with a single detection which we use to build the identity threshold. The created training set was used to learn CNN predicting age with the architecture proposed in [16]. We evaluated the accuracy of the CNN in terms of the Mean Absolute Error (MAE) and CS5 score, with the latter representing the portion of test examples with prediction error not higher than 5 years. We opted for challenging cross-database testing. To this end, we evaluated the accuracy on AgeDB [24].

The results are summarized in Figure 5.4 and Table 5.1. When the results obtained from the proposed algorithm are compared with the single-detection heuristic [29], we can see that the proposed method allows us to harvest a higher number of faces than the single-detection heuristic would. Moreover, the quality of the annotation is sufficient to allow learning CNN with higher prediction accuracy.

5.4 Statistics of IMDB annotated subset

In 5.6 we compute two errors, misclassfied and overlooked. The misclassified errors represents the cases when we selected the wrong face, while the ground truth face was present. The cases 2 and 4 Figure 4.2, and the overlooked errors represent cases when we did not choose a celebrity, or we chose the wrong one, when the ground truth face was present. Namely the cases 2 and 3.

After the computation of the errors, the algorithm produces an HTML result page Figure 5.5 with an overview of the celebrities, their computed distance and orders them according from smallest to highest. The HTML results show a clear overview of the behaviour of the classification threshold and how it influences the annotation accuracy. It is quite useful to see at what point of the threshold do the misclassified samples start to occur. By sorting the distances from lowest (best) to highest (worst) we could potentially see which landmarks affect the result the most.

We can see that by discarding 5.7 and 5.8 some celebrities due to an insufficient amount of images with single detections, we increase the accuracy. This means that when we don't have a sufficient amount of images, and we take all of the images associated to celebrity to build the model, the accuracy falls off noticeably.

5.5 Statistics of manual annotation

To select a batch from the database for manual annotation we wanted to pick the subset in some meaningful way. Initially we first checked the distribution of images per celebrity 3.2. To see the spread of the distributed images by celebrities. This was one of the main factors in deciding the subsets from. What we can gather from the data is that most of the celebrities fall in the range of having either 0 to 10 images in the dataset. When looking at the age distribution Figure 5.9 the average year of the celebrities falls somewhere between 28 and 34 years. This is just a bit above the average age of the world population, which is approximately 29.6 years[28].

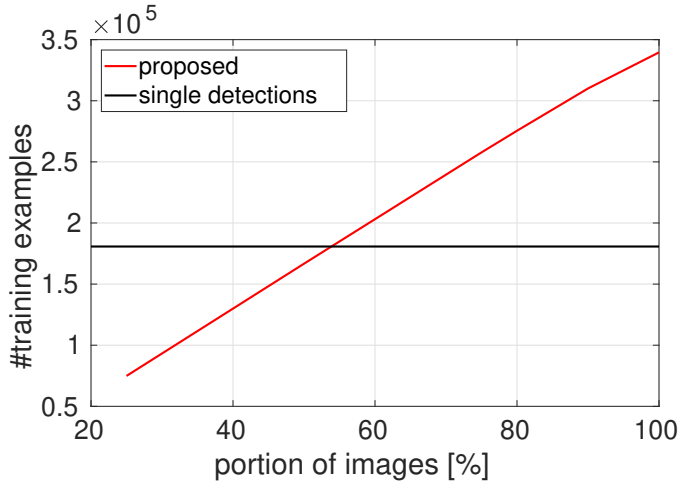
The last deciding factor for the annotation set was the amount of images associated to the celebrities. We decided to split the dataset by celebrities into subsets, such that each subset will represent celebrities which will have the number of faces in some range. The ranges were as such $ranges = [(1,5), (6,10), (11,20), \ldots , (99,100+)]$. From each range we picked between 70-100 celebrities at random, such that both of the requirements of age and gender would be satisfied.

Let $S_{(x,y)}$ be a subset of celebrities c s.t. $(x, y) \in ranges$ where each c has the number of images associated to it in (x, y). Let S_{ann} be the annotation set. Then

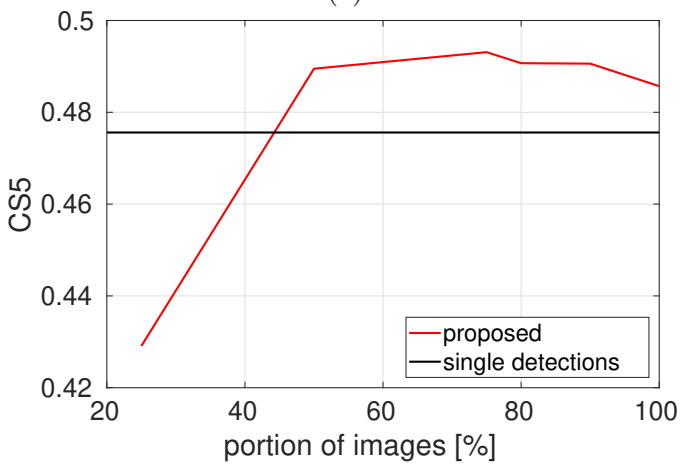
$$S_{ann} = \sum_{x,y}^{ranges} [sample(S_{x,y}, 100)] \tag{5.2}$$

The resulting subset S_{ann} contains a total of 656 celebrities. The age distribution 5.9 remained relatively the same as in the original dataset. And so did the gender distribution 5.10 with there being 360 male celebrities, 276 female celebrities, and 20 celebrities whose gender is missing or undefined.

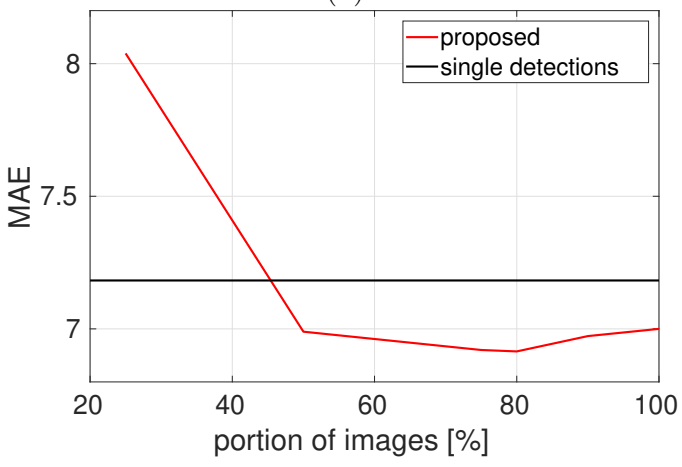
We tested our method on this manually collected annotation with the same parameters, looking at the misclassification and overlooked samples in Figure 5.11 we can see that both of the error curves highly resemble the errors from the imdb_annotated set. The intersecting point for the curves occurs for the threshold of: 0.69. By obtaining almost identical results from two independent datasets we can conclude that the accuracy and results are consistent, and not biased towards gender or age.



(a)



(b)



(c)

Figure 5.4: Figure (a) shows the number of training faces obtained when using the proposed method with the quality score threshold set to accept given portion of images. Figure (b) and (c) show the MAE and the CS5 score obtained when training CNN from faces originating from a given portion of images. The corresponding results for the single-detection heuristic is denoted by horizontal black line.

Figure 5.5: The HTML page created by the algorithm after computing the identity models and evaluating the faces against it. The HTML shows the number of celebrities, number of faces, images and gender. The presented faces are sorted from lowest to highest distance.

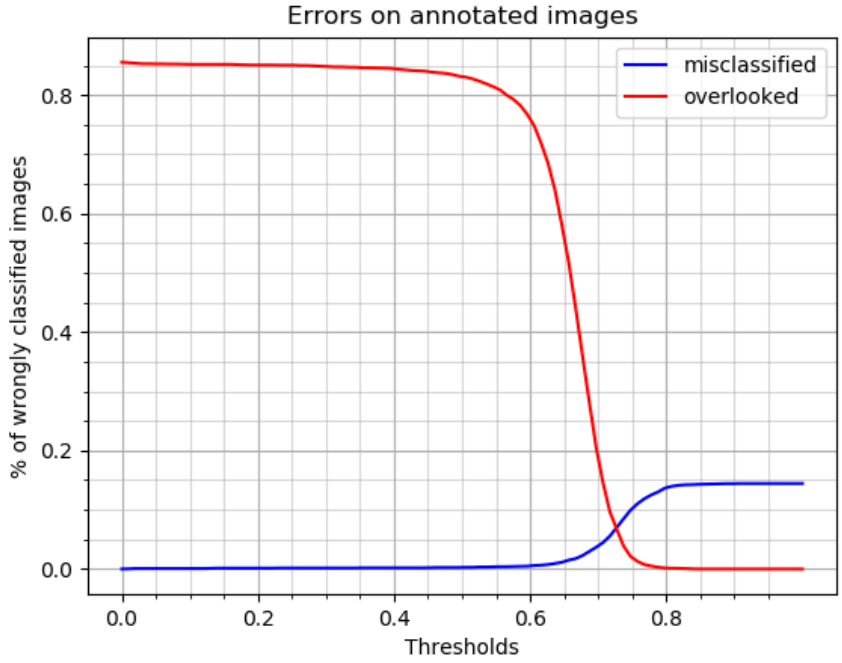


Figure 5.6: Misclassified and overlooked samples on 3523 total samples when using the thresholds for single detections as 0.

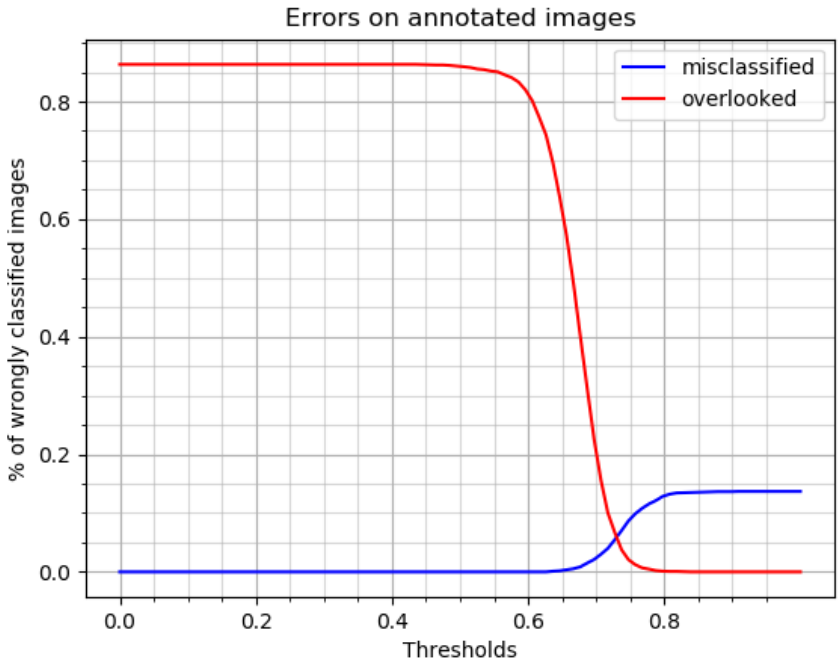


Figure 5.7: Misclassified and overlooked samples on 2786 total samples when using the thresholds for single detections as 10.

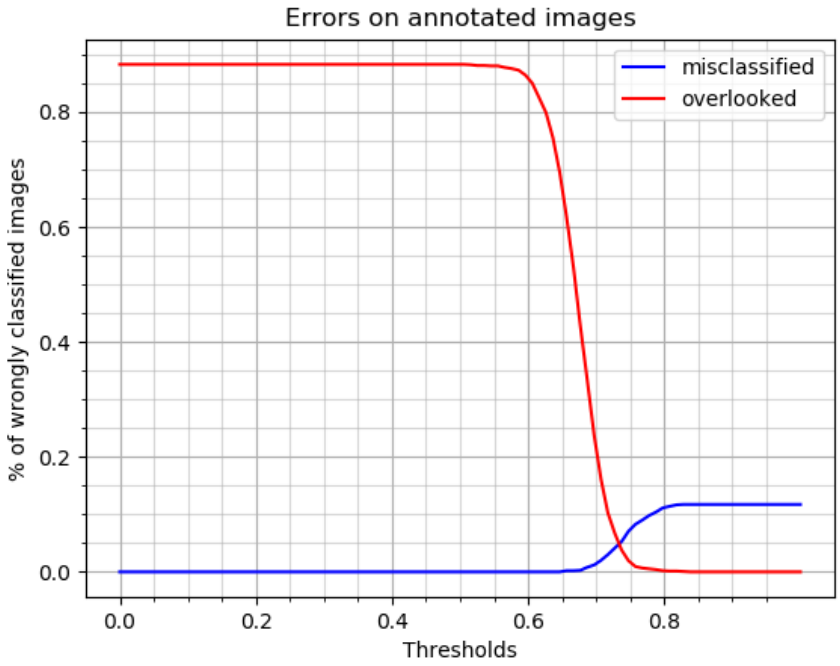


Figure 5.8: Misclassified and overlooked samples on 1498 total samples when using the thresholds for single detections as 50.

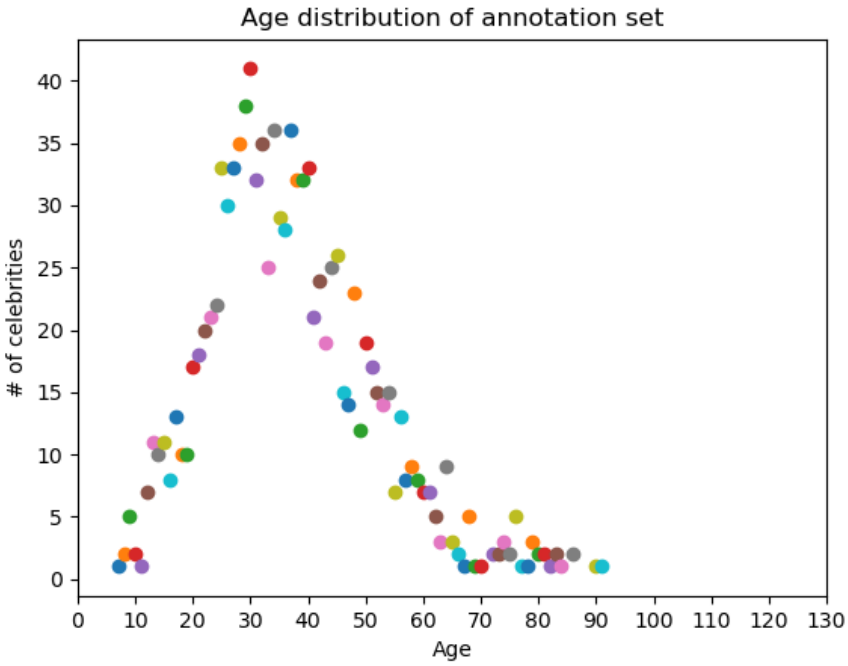


Figure 5.9: Age distribution of the celebrities we picked for the annotation set. Chosen w.r.t. the number of images, age and gender distribution in the original set. The X curve represents the age of the celebrities. The Y curve represents the number of celebrities of that age.

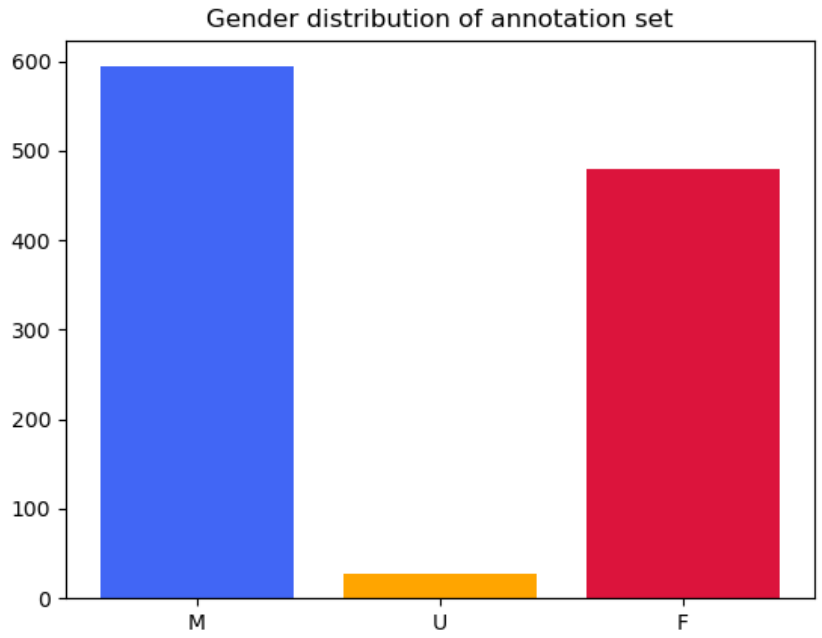


Figure 5.10: The gender distribution of celebrities we picked for the annotation set. The subset was chosen w.r.t. to the number of images associated to celebrities, their age and gender distribution in the original dataset.

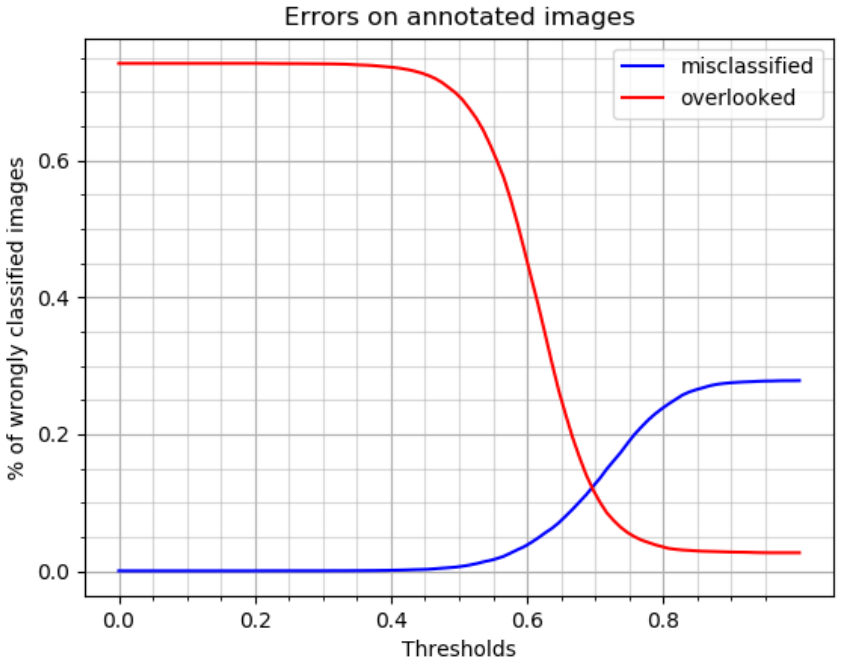


Figure 5.11: Misclassified and overlooked samples on 656 total samples when using the thresholds for single detections as 0.

Chapter 6

Annotation tool

In order to test our model we need a subset of ground-truth annotation. We can then use it to evaluate the accuracy of the proposed automated annotation algorithm. However, since we are dealing with the weakly annotated dataset, to obtain such subset it is necessary to manually annotate. This tool was designed with the idea in mind of collecting the manual annotation quickly and effectively. The process of manual annotation can be quite tedious and prone to human error due to the monotone nature of the task, so it is important to make it as practical as possible.

6.1 Motivation

Regardless of the algorithm and the method used for annotation, there is an inevitable room for error. In order to be efficient and best circumvent this it is necessary to use algorithms which can swiftly and accurately process a large number of images. Then in order to ensure the best results manually check the outliers and inaccuracies. This is a taxing and mentally challenging process due to the inherent automation and monotonic actions required by the user.

By going through many images of the same person it is difficult to remain focused and pay enough details to each individual image. Which leads to overlooked errors. Our annotation tool looks to use the data acquired by the computation of the identity models to ease the process of manual annotation. It does so by ordering the results for each image according to the pre-computed results from the identity models in the data. This would allow the annotater to cover a larger amount of images in the same time with less mistakes, by being offered the potential candidate immediately.

6.2 Framework

The tool is built purely in python. It uses an extremely efficient GUI module [3] in order to make the graphical user interface of the application. It

is possible to compile the application for both Linux, Windows and Mac operating system, and just distributing a single executable file to be run without any prior installation of any python packages. The data is necessary to come separately, as the application reads the images on the fly.

Aside from the GUI, the application uses the same modules for image processing in numpy and openCV, as well as some additional ones like webbrowser, math, and collections.

Essentially the application has three main components that are necessary for running it properly. Namely those are a text file used in the annotation algorithm which holds the information of all the celebrities and images, but updated with the score from the annotation algorithm. Meaning, each face described in the text file will have a value representing how far away it was from the identity model. It needs the images stored in a sub-directory from the main directory of the app, and the permission to access it in order to open the images and process the face to display it in the application. And lastly in needs the application in executable form which can be distributed.

6.3 Layout

The application has two main windows that are displayed to the annotater, 6.1the main layout and the annotation layout. The main layout presents the user with a list of all the celebrities found in the aforementioned descriptive text file provided with the application. The annotater can freely scroll and select a celebrity from the list just by clicking on the desired name.

The annotation layout 6.2 is used for the actual manual annotation of images. When the user selects a celebrity, the application finds all the images associated with the celebrity in the text file. Then in a specific order it presents an image by image, that is it displays all of the faces described on each image in order by the previously computed score.

The annotater is faced with three main cases when annotating the images: 1) The face is present in the image and correctly marked, 2) The face is present in the image, but it is not correctly marked, 3) The face is not present in the image. The first face presented by the application is the face which has the lowest score, i.e. the highest probability of being the correct face. This face is emphasized to the user by its' red border that it should be the correct face. The other faces are displayed consecutively in order. The user is able to confirm, i.e. verify the annotation presented by the algorithm by clicking the "Verify Annotation" button. This will mark the face presented in the first block as the ground truth face, while mark all the other with the negative annotation. Upon clicking the button, the user will see a visual change in colors of the buttons presented under each of the faces, denoting which faces are incorrect (red), and which is the ground true face (green).

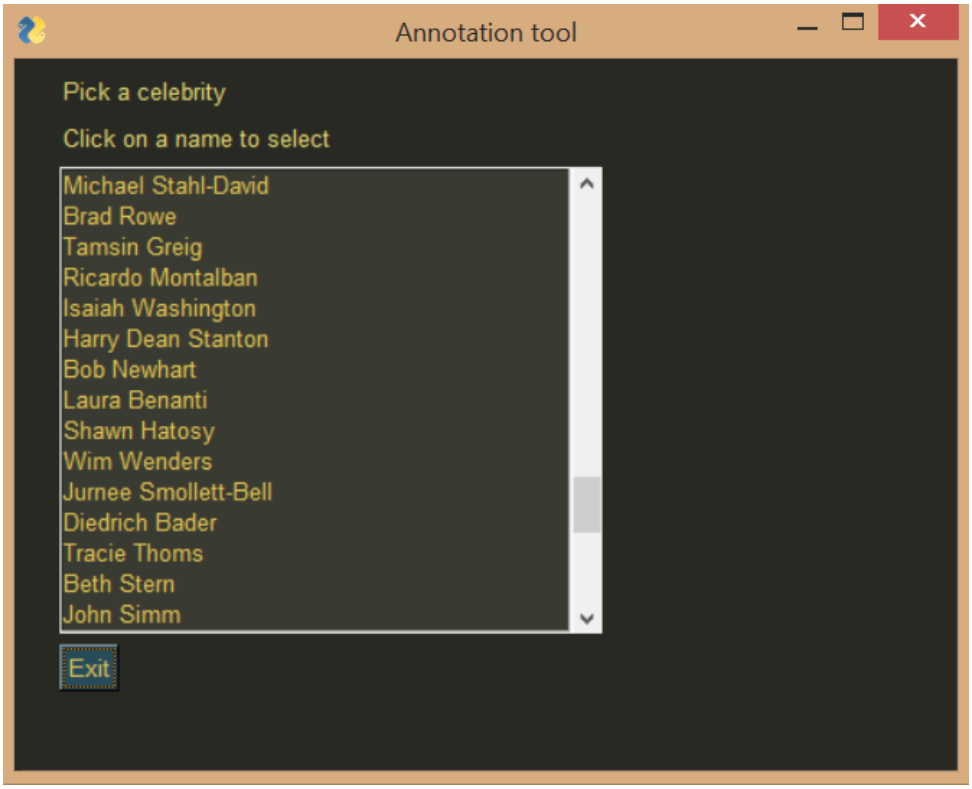


Figure 6.1: The main layout of the annotation tool

In the event that the presented order is incorrect, i.e. the real ground truth face is detected in the image, but it is not on presented on the first place, the user is able to instead of verifying the annotation, click the button under the face to mark it as the correct face. This will set all of the other present faces as incorrect. Alternatively, if the real face is not detected, the user should select the "Not Present" button which will mark all of the faces on the image as incorrect.

6.4 Functionality

The core of the application resides within an event loop, which is common for most GUI applications. While the application is running, so is the loop, listening to the user input and actions. There are two loops for processing the requests. The main loop which takes care of creating the layouts in the application, and an inner loop which deals with the actions when annotating images. In the PySimpleGUI all actions and inputs are considered events. An Event is any action or change that can be interpreted by the interface, which can be retrieved through the framework. Each layout is connected to the individual event loop.

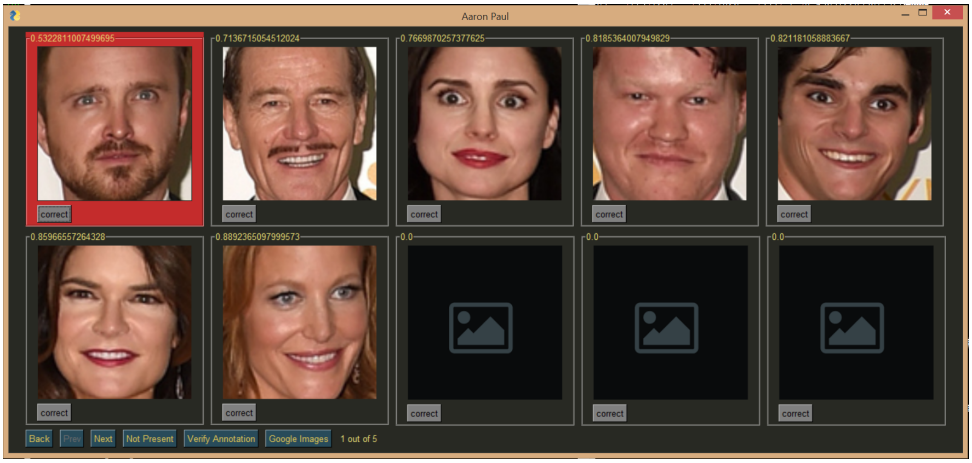


Figure 6.2: Annotation layout for Aaron Paul

In PySimpleGUI when designing a layout, we need to specify the elements that will appear in a python list. We add them in a top-bottom, left-right order, which represents the way they will appear on the screen. Then we can create an instance of the "Window" object in PySimpleGUI which encapsulates all of the data together. Additonally it allows for customization of the window, such as changing the shape, color, size, setting the title and making it resizable.

The "Window" object has a unique ID in the application. During the windows lifetime, once it is closed it cannot be used anymore. It is necessary to create a new window every time. In the application we want to retain the list of celebrities, and do not want additional overhead in creating the same window over and over again, it is possible to keep track of windows that are currently shown or hidden. While the outer event loop is running, the main layout window is shown to the user awaiting for input, and once we enter the inner loop it is hidden.

In the inner event loop the annotation window is created for the chosen celebrity. From the on-click event we get the name of the celebrity. All of the descriptive information of images is loaded into a similar dictionary as in the annotation algorithm, and when creating the annotation window we pull only the images associated to the celebrity. The annotation window is created once for each celebrity, and only the images displayed are changed. To execute the tasks there are two classes used by the application. The "DataIterator" which is responsible for extracting the images from the database and displaying them to the user, updating their annotation, and later storing them to the database. The second class used to encapsulate the data of images is "ImageInfo" which describes a single face. As soon as the annotater changes scenes, from either the main layout to the annotation layout, or by switching celebrites, the changes are permanently added by the DataIterator. If the user were to go back to the previously annotated celebrity, all of the changes made would be

displayed. The correct images would have the buttons colored in green, and the incorrect ones in red.

The DataIterator notes the number of images present for the celebrity, the location of all the images on the machine, position of the detected faces on the images and the current image, i.e. page that is being shown on the display. It checks what is the maximum number of detected faces in the selected images. Since we want to create the layout only once and update the faces as we go, we need to create the layout such that it can display all of the images. When displaying an image which has less than the maximum number of faces detected the remaining frames will load a default image indicating that there are no more faces.

6.4.1 User Input

The main layout deals with on-click events from the list. There are no buttons used for user input aside from the mandatory "Exit" button for shutting down the application. Clicking the X button of the application will perform the same action as the "Exit" button. All of the data work will still be saved.

In the annotation layout for celebrities the buttons shown to the user are buttons to control the iteration of images, namely "Prev" and "Next". These buttons task the DataIterator to change the faces from one image to another, by updating the current page, i.e. the image we are currently observing. The "Next" button is disabled once the last image is reached, and the "Prev" button on the first image. There are the buttons to accept and reject the proposed annotation by the algorithm, "Verify Annotation" which assumes the presented image by the algorithm is the correct one. "Not Present" for the case when the image is missing on the image, marking all of the faces as incorrect. The buttons under each face mark that face as the ground truth face, and all the other faces as incorrect. The functionality of the "Verify Annotation" and "correct" button under the first face, the one proposed by the application in the red border, is the same. There is a button to open the users preferred web browser to the Google image result page of the celebrity, in case the user is unfamiliar with the celebrity.

Chapter 7
Conclusions

The thesis aimed at developing a method for effectively automating the process of assigning the correct age to faces. The method uses the idea of building identity models from the data to perform the annotation.

We show the behaviour and how tuning the parameters for the annotation algorithm affects the result. The hyper-parameters, margin and classification threshold, have been tuned to yield the best results. We show the optimal bounding box margin, c.f. Figure 5.1. Error curves have been computed for the two types of errors: misclassfied and overlooked, with the optimal margin to demonstrate the trade-off between misclassification and overlooking when annotating, c.f. Figure 5.6. Additonally, we present how the errors change when increasing the minimal requirement of single detections for building identity models, c.f. Figure 5.75.8.

For the purpose of collecting manual annotation, we develop a tool to effectively and swiftly acquire annotation labels of the data. We carefully select a subset from the IMDB database, for which we collect the labels of 650 celebrities with the help of the tool. The collected annotation has been used to evaluate the accuracy of the proposed methods and model.

7.1 Future work

The identity models computed from the pre-trained CNN currently pass the responsibility of extract age, gender and facial landmarks solely on the CNN. An interesting path would be to try and additionally filter the results by some age and gender dedicated recognition systems. This could provide more insight to the computed features, and allow for more accurate selection of faces for the model. Namely if the identity model is being computed from single detections for a male celebrity, if we were to encounter a detection which we would have a high certainty that it's of a female identity, we could exclude it from the model. In the same manner for the age, if some of the detections would show a vast difference in the biological age, we could decide to discard the result. Or, with sufficient enough data, we could build an identity model for the celebrity for different age groups.

The annotation tool provides means for efficient annotation of large datasets. However, there may be special cases when the annotation would be more effective for a different layout. The idea is that instead of annotating images of the celebrity one by one, we would extract the recommended faces by the algorithm from 10 or 15 images, and show them all together. For the data which we have high accuracy this would significantly speed up the process, allowing to instantly annotate multiple images of the celebrity at once. However, such an approach would be harder for the annotator with an increasing number of wrongly presented faces.

Bibliography

[1] Oxford Visual Geometry Group. `https://www.robots.ox.ac.uk/~vgg/`.

[2] *Pushing the Frontiers of Unconstrained Face Detection and Recognition: IARPA Janus Benchmark A.*

[3] PySimpleGUI Module. `https://pysimplegui.readthedocs.io/en/latest/`.

[4] VGGFace implementation with keras framework. `https://github.com/rcmalli/keras-vggface`.

[5] E. Agustsson, R. Timofte, S. Escalera, X. Baro, I. Guyon, and R Rothe. 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.

[6] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Proc. of Neural Information Processing Systems*, 2002.

[7] Raphael Angulu, Jules R. Tapamo, and Aderemi O. Adewumi. Age estimation via face images: a survey. *EURASIP Journal on Image and Video Processing*, 2018(1):42, 2018.

[8] Raphael Angulu, Jules R. Tapamo, and Aderemi O. Adewumi. Age estimation via face images: a survey. *EURASIP Journal on Image and Video Processing*, (42), 2018.

[9] G. Antipov, M. Baccouche, S.-A. Berrani, and J.-L. Dugelay. Apparent age estimation from face images combining general and children-specialized deep learning models. In *CVPR workshop, Looking at People Challenge*, 2016.

[10] K. Antoniuk, V. Franc, and V. Hlavac. V-shaped interval insensitive loss for ordinal classification. *Machine Learning*, 2016.

[11] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[12] Kuang-Yu Chang, Chu-Song Chen, and Yi-Ping Hung. Ordinal hyperplane ranker with cost sensitivities for age estimation. In *CVPR*, 2011.

[13] C.H. Chen, V.M. Patel, and R. Chellappa. Matrix completion for resolving label ambiguity. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[14] Y.-C. Chen, V.-M. Patel, R. Chellapa, and P.-J. Phillips. Ambiguously labeled learning using dictionaries. *IEEE Transactions on Information Forensics and Security*, 9(12):2076–2088, 2014.

[15] Timothee Cour, Benjamin Sapp, and Ben Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 12:1225–1261, 2011.

[16] Vojtech Franc and Jan Cech. Learning cnns from weakly annotated facial images. *Image and Vision Computing*, 2018.

[17] X. Geng, K. Smith-Miles, and Z.H. Zhou. Facial age estimation by learning from label distributions. In *Proc. of Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[18] X. Geng, Z.H. Zhou, and K. Smith-Miles. Automatic age estimation based on facial aging patterns. *IEEE Trans. Pattern Analysis and Machine Learning*, 29(12):2234–2240, 2007.

[19] Hu Han, Charles Otto, and Anil K. Jain. Age estimation from face images: Human vs. machine performance. In *International Conference on Biometrics (ICB)*, 2013.

[20] L. Jie and F. Orabona. Learning from candidate labeling sets. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*, pages 1504–1512, USA, 2010. Curran Associates Inc.

[21] R. Jim and Z. Ghahramani. Learning with multiple labels. In *Proc. of NIPS*, 2002.

[22] A. Lanitis, C.J. Taylor, and T.F. Cootes. Toward automatic simulation of aging effects on face images. *IEEE Trans. Pattern Analysis and Machine Learning*, 24(4):442–455, 2002.

[23] S. Lapuschkin, A. Binder, and K.-R. Muller. Understanding and comparing deep neural networks for age and gender classification. In *Proceedings of the ICCV'17 Workshop on Analysis and Modeling of Faces and Gestures (AMFG)*, 2017.

[24] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S. Zafeiriou. Agedb: the first manually collected, in-the-wild age database. In *Proceedings of IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR-W 2017)*, Honolulu, Hawaii, June 2017.

[25] Ramanathan N. and R. Chellappa. Computational methods for modeling facial aging: Asurvey. *Journal of Visual Languages and Computing*, 2009.

[26] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[27] K. J. Ricanek and T. Tesafaye. Morph: A longitudinal image database of normal adult age-progression. In *IEEE 7th International Conference on Automatic Face and Gesture Recognition*, pages 341–345, Southampton, UK, April 2006.

[28] Hannah Ritchie and Max Roser. Age structure. *Our World in Data*, 2019. https://ourworldindata.org/age-structure.

[29] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, December 2015.

[30] A. Shrivastava, V.-M. Patel, J.-K. Pillai, and R. Chellappa. Generalized dictionaries for multiple instance learning. *International Journal of Computer Vision*, 114(2–3):288–305, 2015.

[31] S. Yan, H. Wang, X. Tang, J. Liu, and T.S. Huang. Regression from uncertain labels and its applications to soft biometrics. *IEEE Transactions on infromation forensics and security*, 3(4):698–708, 2008.

[32] Z. Zeng, S. Xiao, T.-H. Jia, K. Chan, S. Gao, and Y. Ma. Learning by associating ambiguously labeled images. In *Proc. of the Internationa Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2013.

[33] M.-L. Zhang and F. Yu. Solving the partial label learning problem: An instance-based approach. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 4048–4054. AAAI Press, 2015.