Children safety and privacy protection through facial characteristics censorship in photos

Final project in Deep Neural Networks course at DSIT @ UoA Postgraduate Program

Alexandros Zerntev
National and Kapodistrian University of Athens
alexzerntev @ di.uoa.gr

Nikolaos Episkopos
National and Kapodistrian University of Athens
episko @ di.uoa.gr

Spyros Nikolakis
National and Kapodistrian University of Athens
snikola @ di.uoa.gr

Abstract

Social media online services have invaded our lives throughout the globe during the last decade. However, with the rise of the use of social networks and other web services, new types of privacy invasion and cybercriminal activities have been introduced. A huge effort is being made both through legislation changes and through frequently updating the terms of service in these services to protect the users' privacy and prevent exploiting these services for criminal activities. Such an activity revolves around child abduction and blackmail. In this project we try to introduce a censorship system which will protect the identities of children in photographs which are uploaded on the web.

1. Introduction

As the years go passing by, technology advances enabling more and more people to have internet access. The internet has opened a whole new potential for revolutionary web services which resulted in the business sector evolving around the internet shifting its focus towards providing revolutionary web services. Some of these services, which nowadays are called social media or social networks, aim at making people more social and helping them communicate with their friends and relatives in a more direct and live way compared to what the phone call offers. Chatting, posting and content sharing are the fundamental ways of communication in social media and a lot of individuals and corporations utilize these capabilities to promote themselves and their work.

However, this whole data sharing generates some questions of several importance, like "Who does this data belong to?", "Who can access this data?", "How can this data be protected and/or deleted?" and "Who can exploit this data and how?". More and more people ask questions of this type and a lot of actions have been taken. A recent example of such actions is the European Union's General Data Protection Regulation. However, in several cases where the safety and bodily integrity of individuals are at stake, more drastic measures must be taken. For example, for identity verification, Facebook may force its users to provide a digital copy of their national identity cards, in order to make sure that the details of a profile/account are correct and that this account is not used maliciously.

Furthermore, in order to upload a picture in which one or more other individuals are present, in any web service out there which makes this image public, all the individuals that appear in the picture must give their consent to the uploader before the upload begins. In the context of children protection, this means that small children who are not aware of any of the aforementioned stuff and can not give their consent to other people for anything, solely rely on their parents' decisions. However, in the future, when these children grow up and realize the value and the importance of privacy as well as the dangers of public exposure, they may not agree with their parents' decisions and may want their pictures taken down.

In this context, several criminals collect data from potential victims and stalk them for some time before they proceed to the actual crime, which might even be abducting a member of the victim's family, in which case the easiest target is a child. That is the reason why people are advised against sharing pictures of their children in social networks, so that they hide the fact that they have children and also protect the identities and the appearance of their children.

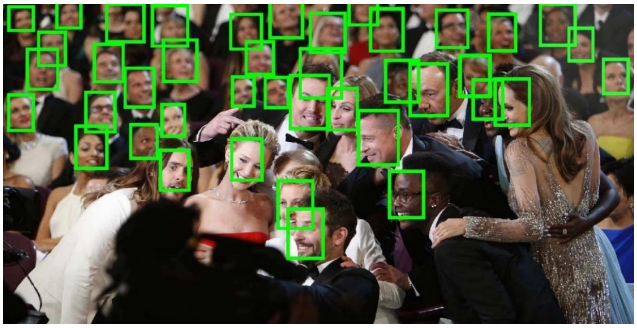

Figure 1. Face detection in an image with multiple human individuals [2].

However, since people sometimes prefer not to follow the recommended guidelines for their safety and as we all know prevention is better than cure, we can install automated systems for non negotiable enforcement of safety standards. In the context of children protection, we propose an automated censorship system, which censor all pictures that are uploaded in a web service and feature young children.

So, the type of censorship we chose to implement is covering the faces of young children, more specifically under the age of 5, with emojis, in order to hide their facial characteristics so that the family's privacy is protected and the children's safety is maintained, since no one who views these censored photographs will be able to recognise any of these children when they see them in close proximity.

2. Related Work

2.1. Face detection

Face detection is a computer technology that determines the location and size of a human face in digital images. Given an image, the goal of facial recognition is to determine whether there are any faces and return the bounding box of each detected face. Other objects like trees, buildings, and bodies are ignored from the digital image. Face detection can be regarded as a specific case of object-class detection, where the task is finding the location and sizes of all objects in an image that belongs to a given class. Face detection is the necessary first step for all facial analysis algorithms, including face alignment, face recognition, face verification, and face parsing. Also, facial recognition is used in multiple areas such as content-based image retrieval, video coding, video conferencing, crowd surveillance, and intelligent human-computer interfaces.

Face detection with a deep convolutional network, achieving high recall of faces even with severe occlusions and head pose variations. The detecting of human faces is a difficult computer vision problem. Mainly because the human face is a dynamic object and has a high degree of variability in its appearance. In recent years, facial recognition techniques have achieved significant progress. However, high-performance face detection remains a challenging problem, especially when there are many tiny faces. There are two types of approaches to detect facial parts, (1) feature-based and (2) image-based approaches.

1. Feature-based approach.

   - **Technique:** Feature-based methods try to find invariant features of faces for detection. The underlying idea is based on the observations that human vision can effortlessly detect faces in different poses and lighting conditions, so there must be properties or features which are consistent despite those variabilities. A wide range of methods has been proposed to detect facial features to then infer the presence of a face.
   - **Examples:** Edge detectors commonly extract facial features such as eyes, nose, mouth, eyebrows, skin color, and hairline. Based on the extracted features, statistical models were built to describe their relationships and verify a face's presence in an image.
   - **Advantages:** Easy to implement, the traditional approach.
   - **Disadvantages:** A major problem of feature-based algorithms is that the image features can be severely corrupted due to illumination, noise, and occlusion. Also, feature boundaries can be weakened for faces, and shadows can cause strong edges, which together render perceptual grouping algorithms useless.

2. Image-based approach.

   - **Technique:** Image-based methods try to learn templates from examples in images. Hence, appearance-based methods rely on machine learning and statistical analysis techniques to find the relevant characteristics of "face" and "no-face" images. The learned characteristics are in the form of distribution models or discriminant functions that is applied for face detection tasks.
   - **Examples:** Image-based approaches include neural networks (CNN), support vector machines (SVM), or Adaboost.
   - **Advantages:** Good performance, higher efficiency.
   - **Disadvantages:** Difficult to implement.

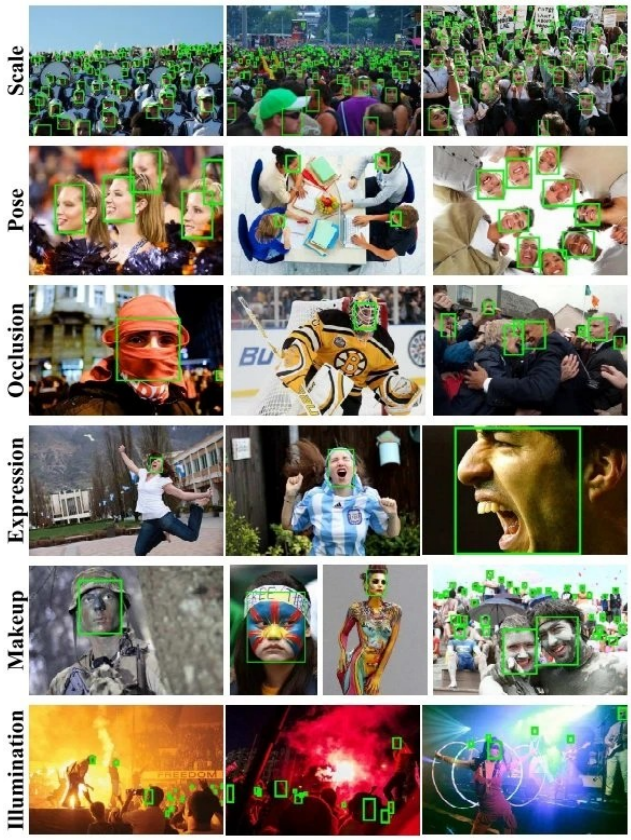

Figure 2. The face detection dataset WIDER FACE has a high degree of variability in scale, pose, occlusion, expression, appearance, and illumination [2].

2.2. Age estimation

Raising the ability of a machine to recognize and interpret faces and facial traits such as age in real time can improve the interaction between humans and machines. Many researchers pay attention to the automatic interpretation of face images. Consequently, systems to identify faces and gender, estimate age and recognize emotions, have been developed.

However, faces change with age: as we get older, the skin becomes thicker and its colour and texture change, the tissue composition begin to be more sub-cutaneous and the facial skeleton lines and wrinkles appear. The process of ageing is very complicated and varies greatly for different individuals.

Thus, Automatic Age Estimation (AAE) from face images is a challenging topic because of the large facial appearance variations. It is due to a mixture of extrinsic and intrinsic factors. The extrinsic factors are mainly determined by living environment, health conditions, lifestyle, etc., while intrinsic factors include physiological elements, such as genes. Robust AAE systems based on facial images should deal with facial expressions and appearance changes.

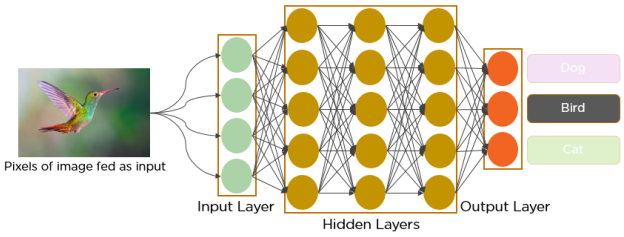

Figure 3. Deep Neural Network generic architecture for image classification [14].

AAE systems have a wide range of applications in Human–Computer Interaction (HCI), in surveillance and web content filtering and in Electronic customer relationship management (E-CRM). They are needed mainly because humans fail to perform age estimation accurately.Thus, it is crucial to develop AAE systems that outperform human performance.

2.3. State-of-the-Art

Deep learning and Deep Neural Networks (DNN) which are its applications, have recently become a hot research topic and have achieved state-of-the-art performance for a variety of applications [5]. Deep learning attempts to capture high-level abstractions through hierarchical architectures of multiple nonlinear transformations and representations. The basic architecture of a DNN is shown in Fig.3.

2.3.1 Convolutional Neural Networks

Convolutional Neural Network (CNN) is inspired by visual mammalian cortex of simple and complex cells [12]. It consists of 4-8 layers with image processing tasks incorporated into the design. CNN applies three architectural concepts in its architecture namely shared weights, local receptive field and subsampling. Shared weights are formed by convolution kernel in order to reduce a number of free parameters, improving the training performance of the forward BP algorithm. These convolution kernels are randomly initialized and form a noise filter as well as edge detector for the feature maps at the respective layer. Local receptive field is applied by both convolution and subsampling kernel by taking a group of neighbourhood pixels for further processing before passing the result of a coarser resolution to the following CNN layers. Subsampling process forms local averaging while reduces the feature map size at the respective layer. At this point, the exact locations of the features are not important anymore. This process adds the robustness of CNN towards handling deformation of input images such as translation, rotation and scaling [12].

CNN is unique due to the architecture itself. It performs segmentation, feature extraction and classification in one processing module with minimal pre-processing tasks on the input image, as shown in Fig.4. Minimal domain knowledge of the problem at hand is sufficient to perform efficient pattern recognition tasks. This has been proven by a wide range of applications that are using CNN such as face detection [6], face recognition [3], gender recognition [16], object recognition [10], character recognition [1], texture recognition [23] and so forth. This is completely in contrast with the conventional pattern recognition tasks in which prior knowledge of the problem at hand is needed in order to apply a suitable algorithm to extract the right features. In addition, whenever the problem domain changes the whole system needs to change requiring a re-design of the algorithm from the start [12].

As shown in Fig.5, CNNs have been used as basic blocks in building a significant number of interesting face-related DNN architectures, like AlexNet [11], DeepFace [21], VGGNet [19], GoogleNet [20], Facenet [18], VGGFace [15], ResNet [7], SphereFace [13], SENet [9] and VGGFace2 [15]. Each one of these architectures has been know as the state-of-the-art during the period they were published and clearly show how the CNN has inspired researchers to develop better and more accurate models.

2.3.2 Vision Transformers

Another architecture that became very popular nowadays is Vision Transformers[4]. This architecture is based on Transformers[25] architecture for NLP. The main advantage of this architectures is that they are able to understand the context of either a sentence or image. Another advantage of this architectures is that they are able to run parallelly for sequential data, something that until now was a bottleneck in language tasks. Since we used this architecture in our application, more detailed analysis will be presented in section 5.2.

3. Dataset

The dataset used in this application is a product of data mining from various web-based sources as well as images of our relatives with their families. The best part of the collected data was found from social networks and web search engines, namely Google and DuckDuckGo.

There have been generated three different sets of data:

- training
- validation
- testing

Each dataset contains two different directories inside, namely *baby* and *not-baby*.

In order to generate the datasets both algorithmic and manual work was necessary. The images were hand-picked from the aforementioned sources and stored to a specific directory locally. Then this directory was defined in the face extraction algorithm (provided by the YOLOFace software), which is responsible to read stored images from a given directory, extract the faces detected from it into a directory specified. Finally, those faces extracted were manually divided in the *baby* and *not-baby* directories based on what they were depicting.

The images used, contained a various number of faces, both infants and not-infants in different scenery and with different objects surrounding the faces but these were rejected.

4. Face Detection Module

Before we delve into the specifics of the model deployed for our application, it is best to describe the whole process right from the beginning. The dataset used, consists of a number of images portraying people and infants in various scenes. The images depict multiple objects and in our case, we only need to detect faces and in order to achieve that we have introduced a face detection module. After that, the face that has been cropped from the full image, is fed to our model in order to train it or predict whether the person depicted is an infant or not using the already trained neural network.

4.1. Viola-Jones

The first module of our proposed pipeline is the Face Detection module. It is responsible to locate and separate the faces from their surroundings which are of no use in our work.

Face recognition is certainly not a new task, having been introduced in the mid-1960's from Woodrow Bledsoe, Helen Chan and Charles Bisson who are considered as the pioneers of the field. However, in recent years the field has flourished as the interest in it seems to be high and the number of applications that can be improved with its help are vast. The first implementations required some human effort while automatic recognition has improved a lot since 1993, allowing commercial products to be built upon such methods. [22]

Haar cascades were introduced by Viola and Jones to achieve the detection of faces in their proposed Viola-Jones face detector. [26] The method they followed is described in more detail in their publication and is basically the following:

- First, they have introduced a new representation of images called "Integral Image" and it allowed quick calculations on the image by their detector.
- Because there were plenty of visual features that helped describe a face, they were aware that they could not make it work on live applications except if they managed to reduce the number of features. To do so

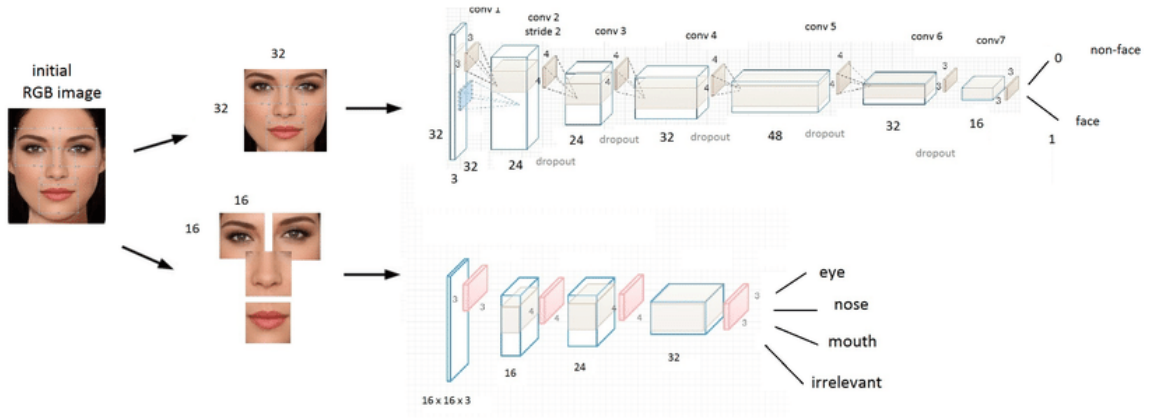

Figure 4. Up: The CNN trained for the task of full face detection. Down: The CNN trained for the task of facial parts detection [24].

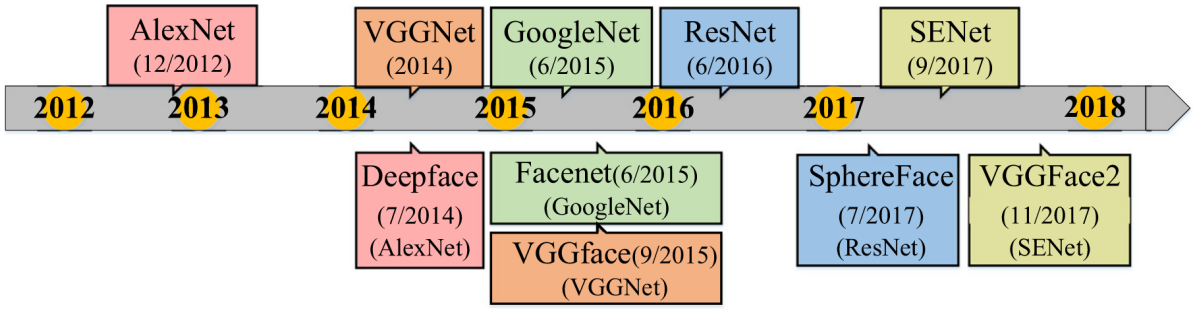

Figure 5. The top row presents the typical network architectures in object classification, and the bottom row describes the well-known FR algorithms that use the typical architectures. We use the same color rectangles to represent the algorithms using the same architecture. It is easy to find that the architectures of deep FR have always followed those of deep object classification and evolved from AlexNet to SENet rapidly [27].

Figure 6. An example of the custom dataset created for the purposes of this application. This image contains two faces, one corresponding to an infant and the other one corresponds to a man.

they have used a machine learning model (namely AdaBoost) which helped at discarding features that were irrelevant to the face features.

- Their last contribution was a way for combining increasingly more complex classifiers in a "cascade" that allows the fast disregarding of the background which allowed faster computations from the detector on the promising regions of the images instead of the whole image.

The usage of Haar cascades is nowadays pretty straightforward: the OpenCV library contains plenty of pre-trained classifiers for face recognition and thus one would not have to train a model to achieve that. Those classifiers are stored in XML format and can be easily used.

4.2. Introduction to YOLOv3

A more modern way of detecting objects is provided by YOLOv3. [17] In a nutshell, YOLOv3 is a blazing fast object detector but in more detail, it performs the following specific tasks:

- **Bounding box prediction:** It predicts bounding boxes using dimension clusters as anchor boxes. Then, it predicts an objectness score for each bounding box which should be equal to 1 if the bounding box prior overlaps

a ground truth object by more than any other bounding box prior.

- **Class prediction:** For each predicted bounding object, the classes that correspond to it are predicted in a multilabel way. An interesting point in this task is that instead of using softmax as the loss function during the training, the binary cross-entropy loss function is utilized. Thus, there is not the assumption that each box has exactly one class but multiple classes for each box may be found.
- **Usage of 3 different scales:** It predicts boxes at 3 different scales from which it extracts features in a similar fashion to how feature pyramid networks work.
- **Feature extraction:** It uses DarkNet-53 (it contains 53 convolutional layers) which is more powerful than Darknet-19 (used by YOLOv2) and more efficient than ResNet-101 and ResNet-152.

4.3. The YOLOFace open-source project

For our own application we have used an open-source project named "YOLOFace". (https://github.com/sthanhng/yoloface) Briefly, this is a YOLOv3 based object detected, trained to detect faces (using this dataset [28]) and the authors provide the already trained network weights in order to facilitate quick development.

In more detail, YOLOFace authors make available the model architecture used and the weights as modified by the procedure of the model training. It is fairly easy for a software engineer to load all this in a matter of minutes and use the powerful detector in their applications.

For our case, we used the forward pass of the pre-trained model to get the predicted bounding boxes that contained faces. Those predicted bounding boxes are later passed through a procedure that rejects overlapping boxes and boxes that have less than 0.5 probability to contain a face. The proposed values by the authors are used for those tasks.

The rejection of overlapping boxes is handled by the non-maximum suppression algorithm which basically, suppresses detections that overlap but have a less detection probability than the one that has the maximum detection probability.

5. Classification Module

5.1. Introduction

After faces extraction our pipeline suggests that we classify the extracted faces into two categories - baby and not baby. The challenge that emerges here is that there is no available dataset of babies and adults in same arbitrary picture so we had to gather everything ourselves, but more on that in the dataset section. Now because the data that one can gather on the internet is limited, conventional deep learning architectures like *CNNs*, *Feed Forward Neural Nets* and even *ResNet* fail to generalize (they overfit on the train data). The solutions that we chose are two: Vision Transformers[4] and VGGFace2 which is an improved version of the original VGGFace. They both had good enough results for our application but Vision Transformers outperformed VGGFace2 in our case.

5.2. Vision Transformers

The first architecture that will be analyzed is the initial publication that presented the Transformers architecture called **Attention Is All You Need [25]** and then its application to Image Classification task by **An Image is Worth 16x16 Words: Transformer for Image Recognition at Scale [4]**.

The reason for that order is that the Vision Transformers architecture has hard dependency on the Transformers architecture.

5.2.1 Attention Is All You Need [25]

Until before this paper, LSTM and RNN were established as state of the art approaches for task like machine translation and language modeling. One of the biggest problems that this architectures anticipated is the inability of parallelization due to the fact that an the current input of a layer depends on a previous input. Another computational problem of this task is that there is no 1-1 correspondence between input and output. (the Greek "gia su" has different count of tokens with "hi"). Both of this problem are solved with transformer architecture by having always a constant amount of operations while running the model. Thanks to the attention mechanism, the transformer architecture is also context (of the sentence) aware. It outperforms previous architectures by significant amount.

The Architecture

On Fig.7 the overall transformer architecture can be seen. We will analyze each of its modules step by step by "zooming" in. The task for which we will apply this architecture is text generation.

Encoder and Decoder

The left side of Fig.7 is called the Encoder and the right side is called a Decoder. They are almost identical. The difference is that the decoder part masks and shifts its input(which is the overall expected output) and at some level the decoder also combines the input and the output in order to generate the final result. But we will discuss the differences more analytically in the next sections.

Preprocessing

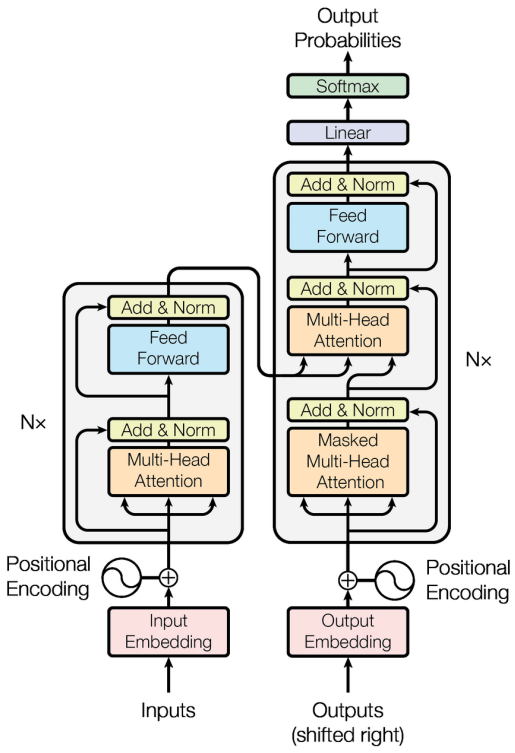

Figure 7. The Transformer - model architecture.

Input: The encoder takes the whole sentence as an input but before that the words of the sentence are converted to word embeddings. The word embeddings are the mapping of the words to some vector space in such way that the semantically similar words have smaller distance between them. A toy example would be for words Dog, Wolf, Car, Truck, and in a two space embedding space their embeddings could be Dog $[0, 0]$, Wolf[0.1,0.3], Car $[0.8, 0.6]$, Truck $[1, 1]$. If we consider an euclidean distance, the distance between a dog and a wolf is very small because they are semantically similar, and the same holds for car and truck. The embedding space could be pretrained independently of the rest of the architecture or done by already trained architectures like word2vec.

Positional encoding: Since the embbedings are fed to the transformer simultaneously, we need a mechanism to encode the position of the words. The publication proposes the below functions

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

The reason to use this specific functions is to be sure that the positions don't overlap, but it could be any other reasonable function as well. The above functions are sinusoidal functions and the i variable represents the frequency of the

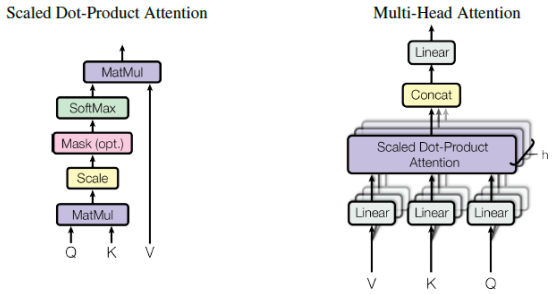

Figure 8. The Multi-Head Attention.

sinusoid, so in every period the same x will never be maped to the same y. By adding the Positional Encoding and the embeddings we get a vectors that both encode the position of the word and its semantics.

Encoder

Multi-Head Attention: The role of this part is to determine which word of the sentence is more important than others, it helps to determine the context of the sentence. The idea here is to find out which words here are more relevant to which other words within the same sentence. We will "zoom in" even more here in order to see how it works.

The right part of Fig.8. shows the Multi-Head Attention. The Linear layer consist of some fully connected neurons without activation functions. Their role is to i) Map inputs to outputs ii) Change the dimensions of vectors. Now lets zoom in to the Scaled Dot-Product Attention part (it can be seen on the left part of Fig.8). We have three linear layers with different weights (this weights are trained during the training process). Each layer outputs a **K**ey, **Q**uery, **V**alue respectively. The idea here is for given Query and Key, find the most relevant key to the query and then use the result as filter (weighted matrix multiplication) on the value. By applying dot product to Query and Key and then re-scaling we essentially get the cosign similarity (not exactly because the scaling factor that we divide is different but it serves the same purpose). Because we feed matrices and not vectors to the Multi-Head Attention, we get the similarity from each vector to each vector. Then the softmax function is applied in order to map the vectors sums to 0-1. This part can be called attention filter. Now we take the attention filter and multiply it with the Value (the reason is to focus on the most valuable information). The result we get is a filtered Value which is the final output of our multi-head attention layer. More formally the whole process can be written as follows:

$$\text{Attention}(Q, K, V) = softmax(\frac{Q \cdot K^T}{\sqrt{d_k}}) \cdot V$$

Now the final step is because we have multiple Scaled

Dot-Product Attention we concatenate them and feed them again to a Linear layer for remapping purposes. The reason for multiple Scaled Dot-Product Attentions (therefore multiple attention filters) is for not focusing only on one case of "valuable information" but on multiple.

Add & Norm

This steps purpose is i) Knowledge preservation and ii) elimination the Vanishing gradient problem. It is done by adding a residual connection from before the Multi-Head Attention. So now the output of Multi-Head Attention is added to the previously encoded input embedding. Then a normalization happens.

Feed Forward

This layer is a feed forward neural net which consist of a linear layer, then a RELU activation function and then again a Linear layer. Then again the output is added to a residual connection and finally layer normalized.

Decoder

As it was mentioned in the 2.1.1 section the decoder part is very similar to the encoder one. Here we will discuss the differences. In general the encoder part takes as input some matrix and outputs a new matrix which is some vectorized representation. The decoder part gets this matrix and converts it to new text. So the main difference between the encoder and decoder is that the former takes one input and the latter takes two. It can be seen on Fig.8. that the output of the encoder is fed to a Multi-Head Attention as both Query and Key in the middle of the decoder as discussed previously. The output of the decoder part before the middle is passed as a value to the Multi-Headed Attention in the middle. Afterwords everything is identical to the encoder.

Lets analyze the part before the middle now. Lets focus now on the Masked Multi-Head Attention layer. Firstly the decoder input (Output Embedding) is shifted one position to the right. The reason for that is for the whole model to predict every next token. For example if we put as input to the encoder "Hello how are you?" then we want the overall model to predict the answer. To do that firstly the input of the decoder is a special token which points as a start of a sentence. So with that input the transformer have to predict the next word of the answer (lets say I'm). If the prediction is wrong, here is where the training happens, so we need the right prediction. Afterwards the start token + the first predicted word comes as input to the decoder in order to predict the next word. This process is repeated until a special token that points to the end of a sentence comes as input. Now to get rid of the repetition, the input array is duplicated multiple times and masked, in such way that the matrix that comes in as an input, on each row has "hidden" (masked) every next sub-string of the input. So for our example "I'm fine", the first row of the input will be only the special start token, and the rest zeros. The second row will have the start token and the "I'm" token, the third will have the start token, the "I'm" word, the "fine" words and the rest are zeros etc... Now we have a vectorized training mechanism to learn the next word prediction.

Linear and Softmax

In the final layer there is a linear fully connected layer that in combination with softmax as an activation function, outputs the final prediction of the next word. out of our vocabulary, which essentially is a classification task that outputs the word with the highest probability.

5.2.2 An Image is Worth 16x16 Words: Transformer for Image Recognition at Scale [4]

So we described the Transformers architecture in previous section. Now we have to apply it on image classification task. The naive approach would be to apply the attention mechanism pixel-wise, but doing that on some modern picture with great size and 3 color channels would be unrealistic for today's available hardware.

The idea that authors propose is to see an image as a sequence of concatenated words in 2 dimensions, where each word is a patch (subimage) of fixed size. But lets analyze it further. The task that we are going to apply this architecture is classification.

Preprocessing

Patches creation: Firstly the input image is separated into fixed sized smaller images. This are the patches which will be treated similarly as words in a sentence, where the patches are the words and the whole image is a sentence.

Patches flattening: Here the authors propose either to directly flatten the images or apply CNN (A hybrid model) filters in order to obtain the desired result.

Embedding mapping: Now the flattened vectors are mapped to some embedding space. In addition to that there is a learnable [class] embedding which is treated as a simple embedding and it exists in order to help the MLP-Head do the classification. The [class] is constant after the training and it is always prepended to the embeddings.

Positional encoding: Afterwards the positional encoding of where in the image the specific patch was. The authors tried a 2D-aware positional encoding, but they didn't show significant results so finally a learnable 1D positional encoding is used. The [class] token has always a fixed position.

Transformer Encoder

The transformer encoder is exactly the encoder of section 2.4. since it takes a set of embeddings as an input.

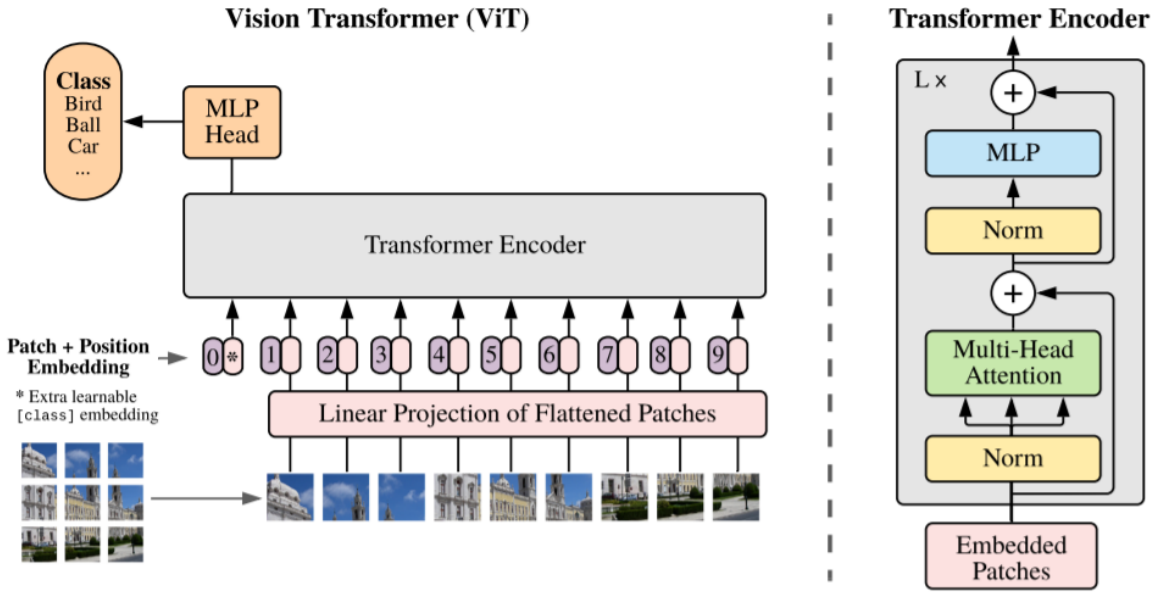

Figure 9. Vision Transformer.

MLP Head

Here is an MLP that does the final classification based on the class token. Specifically the transformer encoder takes a matrix as an input and outputs another matrix. Based on first embedding of the output a classification is done with a simple fully connected neural net.

Pretraining

The transformers are normally pretrained on large general datasets (for NLP it would be milions of texts, for Computer Vision it would be millions of images), then fine tuned to a specific dataset and then applied to tasks. And that is what the authors done before the evaluation.

The result of the attention mechanism can be seen on Fig.10. The left images are the initial images, and the right images are the attention map. The lighter the color the more significant the pixel is.

5.3. VGGFace2

When we use the term "VGGFace2 model" in reality we mean the architecture of the ResNet-50[8] model trained upon the VGGFace2 dataset. The VGGFace2 dataset includes over nine thousand human identities of different countries and ages, with between 80 and 800 images for each identity, and more than 3M images in total, all collected using web search engines. This dataset, combined with ResNet-50 or similar deep network architectures, are still some of the most popular and wildly used methods to perform face-related tasks, like face identification, identity

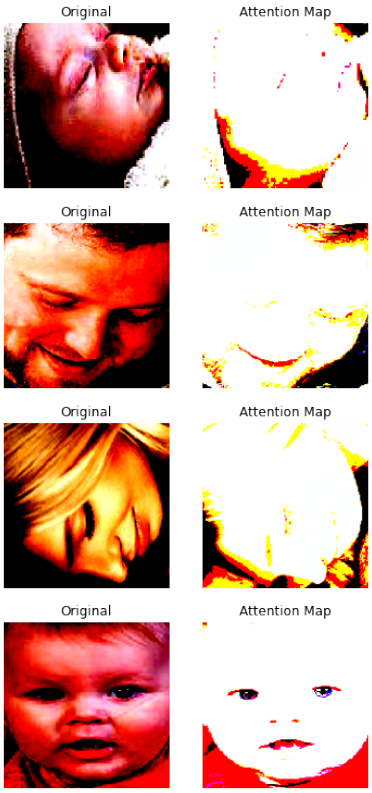

Figure 10. Attention maps (normalized images).

Figure 11. An example of the application's output. The face of the infant is now hidden while the man's face remains intact.

classification, etc..

Experiments of this model show very good results in various applications. Recently, this model has been used to detect identities of people who are wearing surgical masks and the results has a really high confidence. Since this model was considered to be the state-of-the-art until not so long ago, we are going to compare evaluate our proposed model to VGGFace2. Since this model is widely used and has been the state-of-the-art for face-related tasks, we will consider it as our baseline model.

6. Model and Methodology Overview

The application created consists of numerous modules working together in order to solve the problem as stated in the introductory section of the present document. The modules are basically three:

- The **Face Detection** module that finds all the faces in an image.
- The **Classification Network** that classifies whether a given face corresponds to an infant or not.
- The **Face Censor** module that uses the prediction of the **Classification Network** module and hides the face if it corresponds to an infant.

The algorithm expects a directory containing images. For each one of those images it locates the faces and returns the bounding boxes containing them. Then, for each face detected, it passes it through the classification deep neural network and predicts whether is belongs to an infant or not. If it belongs to an infant, it needs to hide the face by using an emoji picture, randomly selected from those downloaded to a given directory. The **Face Censor** module is responsible to select and resize accordingly a randomly selected emoji, and then place it correctly in the image in order to hide the face of an infant.

Figure 12. The high-level architecture of the application. An image passes through the Face Detector which finds all the faces in the image. Each face detected passes through the Classification Network to decide whether the face belongs to an infant or not and finally, if the face belongs to an infant, an emoji hides the face.

7. Experiments

Two distinct sets of experiment were performed. In the first set, we used three distinct face detection models, in order to decide upon the best one, which will be used to extract faces for the seconds set of experiments. In the second set, we used two distinct models in order to decide upon the age of the individual portrayed in the picture.

7.1. Face Detection and Extraction

For this set of experiments, we used the following six face detection models:

- OpenCV (Haar cascade face detector)
- OpenCV (DNN face detector)
- HOG + Linear SVM
- MMOD + CNN
- MTCNN [29]
- YOLOFace

As far as OpenCV is concerned, we used the pre-trained Haar cascades and it mostly failed to detect all the faces that were not facing the camera. There are multiple Haar cascades but one needs to use all of them in an image in order to get all the faces depicted and it will probably fail to detect some of them because of different angles of faces. OpenCV DNN face detector performed better than Haar cascades, however in some cases it considered the area around the faces as part of the faces. Both the HOG + Linear SVM and MMOD + CNN detectors performed worse than OpenCV DNN face detector.

MTCNN, basically promises to perform both faces detection and faces alignment. However, in our case it failed to perform well and practically it failed to detect any faces at all in some sample images we used against it.

Finally, we decided to use YOLOFace which performed pretty well and was able to detect most of the faces in the images provided in relatively small time, as depicted in Fig.13 and Fig.14. More about this, can be found in the dedicated section above.

7.2. Classification

Due to the fact of small dataset, seven classification models were tested for the specific task.

- ResNet-50 (both fine-tuned and retrained)

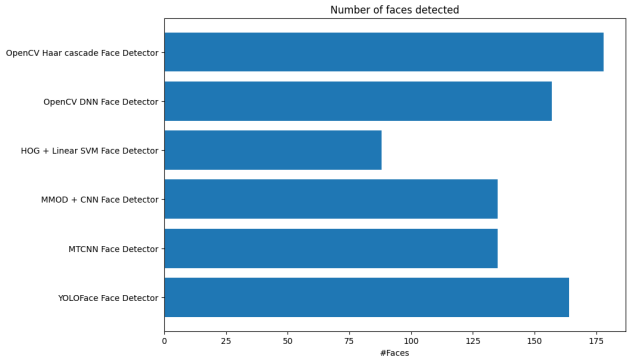

Figure 13. Detected faces counts of the evaluated face detectors.

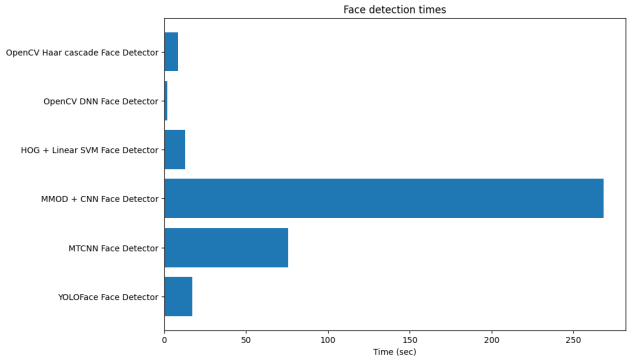

Figure 14. Face detection times of the evaluated face detectors.

- VGGFace based on ResNet-50 (both fine-tuned and re-trained)
- VGGFace based on VGG16 (both fine-tuned and re-trained)
- Vision Transformer

For every model, different hyperparameters, activation functions and optimizers were tested and finally the best one was used for the final model. The ResNet-50 related models failed in this task. The reason for that is that the dataset is small therefore powerful models such as ResNet, overfits the data. But Vision Transformers is context aware, it actually ”understands” what face features are and it doesn't overfit the data so its performance was exceptional for the specific task.

8. Overall Model Results

As aforementioned, we performed a number of different experiments with various networks and the results varied as expected. For the training dataset all models performed well. However, for the validation and test datasets that were not previously fed to the network, only the ViT and VGGFace (VGG16) models did pretty well. As shown on Fig.15 and Fig.16 the best model both in Training, Validation and Test accuracy was the ViT, which also generalizes well for arbitrary images taken by the authors in the wild.

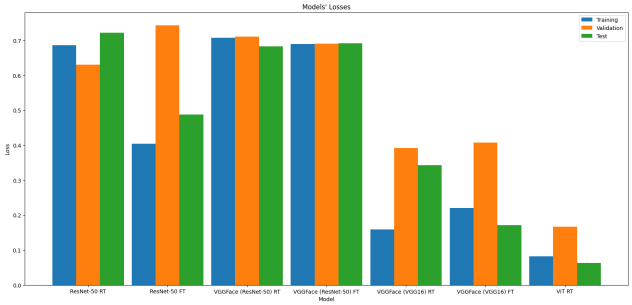

Figure 15. Loss of the tested models.

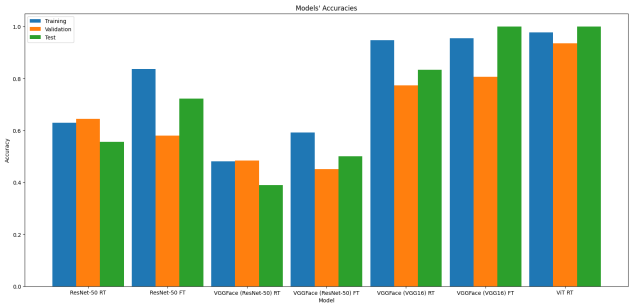

Figure 16. Accuracy of the tested models.

In Fig.17 we can see a family of two parents and two children, in which only the younger child's face has been censored, which is exactly was our system was designed to do. However, in Fig.18 which features four adults and four children, we can see that although our system correctly censored all four children, it also mistakenly censored one of the adults, because the angle in which we was captured was not used in the pictures with which we trained our models, so this misclassification was expected. This also means that there is still room for improvement. Overall, this is a great starting point for future work and improvements which can further fine-tune the models and produce more accurate results.

9. Conclusions and Future Work

It is fascinating how powerful the tested models are. YOLOFace managed to extract faces from any arbitrary image we tried with any pose, angle, brightness etc. On top of that, despite the fact that we had small dataset, Vision Transformer architecture managed to learn and generalize even with this size. On the other hand VGGFace proved to be sensitive to image cropping and augmentation, the CNN models tend to overfit.

There are a lot of things that can be done in order to improve such models. First of all, a bigger dataset needs

Figure 17. A properly classified image from Wikipedia.

Figure 18. A poorly classified image - the person on the far right is an adult.

to be used, in order to have more faces for the training, can have significantly different facial characteristics, skin tone, hair length, face shape, etc.. Furthermore, the data needs to include faces of both children and adults in multiple angles, in order to learn on people who may even be looking away from the camera.

Also, several tricks can be performed during pre-processing, in order to take in account different angles, distances from the camera, etc.. Face alignment is such a trick. Furthermore, face extraction needs some fine-tuning too, since YOLOFace detects and extracts faces through bounding boxes, which are shaped as rectangles. If these rectangles are too large, they may include unnecessary information which may not be helpful with the classification procedure and may even introduce noise, resulting in worse results. A more accurate face detection, with the proper preprocessing resizing and padding is more likely to improve the quality of the input data.

References

[1] Sajjad Ahranjany, Farbod Razzazi, and Hassan Ghassemian. A very high accuracy handwritten character recognition system for farsi/arabic digits using convolutional neural networks. In *Proceedings 2010 IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications, BIC-TA 2010*, pages 1585–1592, 09 2010.

[2] Gaudenz Boesch. Face detection in 2021: Real-time applications with deep learning. *viso.ai*, 03 2021.

[3] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[5] L. Deng et al. Deep learning: methods and applications, 2014.

[6] C. Garcia and M. Delakis. Convolutional face finder: a neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1408–1423, 2004.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[9] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

[10] Fu Jie Huang and Yann LeCun. Large-scale learning with svm and convolutional nets for generic object categorization. In *Proceedings - 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2006*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 284–291, 2006. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2006 ; Conference date: 17-06-2006 Through 22-06-2006.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[13] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. *CoRR*, abs/1704.08063, 2017.

[14] Manav Mandal. Introduction to convolutional neural networks (cnn). *analyticsvidhya.com*, 05 2021.

[15] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[16] Feeza Radzi, Mohamed Khalil-Hani, Shan Sung Liew, and Rabia Bakhteri. Convolutional neural networks with fused layers applied to face recognition. *International Journal of Computational Intelligence and Applications*, 14:1550014, 09 2015.

[17] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[18] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823. IEEE Computer Society, 2015.

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[21] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[22] Smriti Tikoo and Nitin Malik. Detection, segmentation and recognition of face and its features using neural network. *arXiv preprint arXiv:1701.08259*, 2017.

[23] Fok Hing Chi Tivive and Abdesselam Bouzerdoum. Texture classification using convolutional neural networks. In *TENCON 2006 - 2006 IEEE Region 10 Conference*, pages 1–4, 2006.

[24] Danai Triantafyllidou and Anastasios Tefas. Face detection based on deep convolutional neural networks exploiting incremental facial part learning. pages 3560–3565, 12 2016.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[26] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. Ieee, 2001.

[27] Mei Wang and Weihong Deng. Deep face recognition: A survey. *CoRR*, abs/1804.06655, 2018.

[28] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[29] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.