University of Barcelona
Master in Artificial Intelligence (UPC, URV, UB)
Faculty of Mathematics and Computer Science

Deep Learning for Universal Emotion Recognition in Still images

Juan Luis Rosa Ramos

Advisors: Sergio Escalera, PhD in Multi-class visual categorization systems at Computer Vision Center
and Andrés Cencerrado, PhD in High Performance Computing at University Autonoma Barcelona

Abstract

This work propose a methodology for still image facial expression. The proposed method contains a face detection and alignment module followed by a deep convolutional neural network (CNN) that outputs a seven emotions probability vector. The input images are human faces cropped and aligned thus, eliminating the variability of the dataset. Three CNN models where trained and tested with the purpose of recommending the best one in function of its the deployment environment so, their trade-off were take into account. They were build following a transfer learning approach, from pre-trained on face recognition models that were fine-tuned with a new face emotions dataset that was build from a collection of free, for research, available ones. There is also a data augmentation process made with the purposes of analyzing his influence. Experimental results showed that the three models got more than acceptable accuracy and each one has his own advantages depending on the needs. More complicated models expend more training and inference timings but have more generalization capacities. The work also aims to compile an extensive information in face analysis datasets, training and inference times in two different Nividia GPUs. It explains the fine-tuning key points and insists in overfitting avoidance methods. It also has the purpose to serve as a software workflow guide for producing computer vision pipelines from zero to deployment helping to solve to other object classification problems. The best model is going to be deployed in a new driver awareness technology product that aims to avoid driving accidents.

Este trabajo presenta una metodología para el reconocimiento de sentimientos humanos a través del análisis de expresiones faciales. Para ello, se resuelve un problema de clasificación automática de imágenes. La metodología propuesta contiene un módulo de detección, recorte y alineamiento de caras (eliminando la variabilidad) seguido de una red neuronal convolucional (CNN) que devuelve un vector de probabilidades de siete emociones. En total, se han escogido, entrenado y testeado tres modelos diferentes de CNN con el propósito de poder hacer recomendaciones a usar en función de las necesidades de puesta en producción. Las redes fueron entrenadas usando metodología basada en la transferencia de conocimiento desde otras redes pre entrenadas con caras humanas. Fueron re entrenadas con un dataset construido ex profeso y con otro aumentado para testear la influencia de usar mas datos y clases más balanceadas. Una serie de experimentos -en mismas condiciones y mismos parámetros- nos enseñan que los tres modelos tienen niveles muy aceptables de acierto y que cada uno tiene sus particulares ventajas en función de las necesidades: modelos mas complejos tardan muchas más horas en entrenar pero tienen mejores capacidades de acierto con nuevos tipos de datos. Este trabajo también compila una extensiva información sobre datasets de análisis facial, tiempos de entrenamiento e inferencia con distintas GPUs e incluso con CPU. También incluye explicaciones sobre cuáles son los puntos clave en metodologías de transferencia de conocimiento e insiste en como evitar que nuestras redes no tengan capacidad de generalización. El propósito de este trabajo también pretende servir una guía para aplicar la metodología descrita a problemas diferentes al de análisis de caras y que se puedan resolver con clasificación automática de imágenes ya que el flujo descrito va desde cero, hasta consejos de puesta en producción. El mejor modelo será utilizado en una tecnología de monitorización de conductores de vehículos que logrará evitar accidentes en carretera.

Aquest treball presenta una metodologia per al reconeixement de sentiments humans a través de l'anàlisi d'expressions facials. Per a això, es resol un problema de classificació automàtica d'imatges. La metodologia proposada conté un mòdul de detecció, retallada i alineament de cares (eliminant la variabilitat) seguit d'una xarxa neuronal convolucional (CNN) que retorna un vector de probabilitats de set emocions. En total, s'han escollit, entrenat i testejat tres models diferents de CNN amb el propòsit de poder fer recomanacions a fer servir en funció de les necessitats de posada en producció. Les xarxes van ser entrenades usant metodologia basada en la transferència de coneixement des d'altres xarxes pre entrenades amb cares humanes. Van ser re entrenades amb un dataset construït expressament i amb un altre augmentat per testejar la influència d'usar mes dades i classes més balancejades. Una sèrie d'experiments -en mateixes condicions i mateixos paràmetres- ens ensenyen que els tres models tenen nivells molt acceptables d'encert i que cada un té els seus particulars avantatges en funció de les necessitats: models més complexos triguen moltes més hores a entrenar però tenen millors capacitats d'encert amb nous tipus de dades. Aquest treball també compila una extensiva informació sobre datasets d'anàlisi facial, temps d'entrenament i inferència amb diferents GPUs i fins i tot amb CPU. També inclou explicacions sobre quins són els punts clau en metodologies de transferència de coneixement i insisteix com evitar que les nostres xarxes no tinguin capacitat de generalització. El propòsit d'aquest treball també pretén servir una guia per aplicar la metodologia descrita a problemes diferents al d'anàlisi de cares i que es puguin resoldre amb classificació automàtica d'imatges ja que el flux descrit va des de zero, fins a consells de posada en producció. El millor model serà utilitzat en una tecnologia de monitorització de conductors de vehicles que aconseguirà evitar accidents a la carretera.

Keywords Facial Expression Recognition; Convolutional Neural Networks; Overfitting; Face analysis dataset

Contents

Abstract **i**

1 Introduction **1**

1.1 Facial Expressions and Emotions 1

1.2 Some applications 2

1.3 Proposal 3

2 Related work: emotions recognition **4**

2.1 Early studies 4

2.2 Description of emotions 5

2.2.1 Categorical model 6

3 Related work: automatic emotions analysis **7**

3.1 Affective computing 7

3.2 Automated face analysis with computer vision 8

3.3 Related work on automated face analysis 9

3.3.1 Main challenges 9

3.3.2 Face detection and alignment 10

3.3.3 Face analysis classification algorithms 10

3.3.4 Inference in ConvNets 11

4 Related Work: background on 2D CNN 13

4.1 Choosing a classification algorithm 13

4.2 Main layers of a ConvNet 14

4.2.1 Input layers 14

4.2.2 Convolutional layers 14

4.2.3 Dense layers 14

4.2.4 Output layers 15

4.3 CNN choosing a framework tool 15

5 Contribution of this work 19

6 Dataset collection and augmentation 21

6.1 Related work: existing datasets 21

6.2 Selected datasets 25

6.2.1 The Japanese Female Facial Expression (JAFFE) Database 25

6.2.2 The Extended Cohn-Kanade Dataset (CK+) 26

6.2.3 Karolinska Directed Emotional Face(Kdef) 26

6.2.4 Radboud Faces Database (Radboud) 27

6.2.5 Pain expressions 28

6.2.6 Oulu-CASIA 29

6.2.7 Dartmouth Database of Children's Faces 30

6.2.8 California Facial Expressions (CAFE) 30

6.2.9 Face Place (Face) 31

6.3 Constructed dataset 31

6.4 Data augmentation 32

7 Data preprocessing 36

7.1 Face detection and cropping 36

7.2 Face alignment 37

7.3 Dataset normalization 39

8 Deeplearning: Training 2D convolutional networks 41

8.1 Fine-tuning CNN 41

8.2 Avoiding overfitting 42

8.3 Fine-tuning with Caffe 43

8.4 ConvNets: pre-trained models 44

8.4.1 Alexnet 44

8.4.2 Alexnet weights 45

8.4.3 VGG16 45

8.4.4 VGG16 weights 46

8.4.5 Microsoft ResNet 46

8.4.6 Resnet weights: ResFace101 47

8.4.7 ImageNet dataset 47

8.5 Methodology for training 48

8.6 Computing resources 48

8.7 Software resources 50

8.8 Fine-tuning details 51

8.9 Image inference 52

9 Experimental methodology 56

9.1 Experimental settings 56

9.2 Data distribution 57

9.2.1 Original dataset 57

9.2.2 Augmented dataset 57

9.2.3 Test set 57

9.2.4 Overfitting Test set 58

9.3 Evaluation metrics 58

9.4 Method details 59

9.4.1 Parameters 59

9.5 Analysis 59

10 Dealing with overfitting 63

10.1 New experiments 63

10.2 New test results 64

11 Testing inference in real drivers **66**

12 Conclusion **74**

12.1 Summary of Thesis Achievements 74

12.2 Biased ConvNets 75

12.3 Future Work 75

Bibliography **76**

List of Tables

6.1 Total numbers and class division of the images compiled for the dataset 32

6.2 Augmented dataset final numbers 35

8.1 Differences training with two different Nvidia Tesla GPUs 50

8.2 Virtual machines used for training 50

8.3 Software used for training 50

8.4 Same parameters values used for training the three networks 52

8.5 Comparison of the models time training with original and augmented dataset in a Tesla P100 GPU of 12 Gigas. 52

9.1 Final number of the dataset distributions. 58

9.2 Results of the six experiments 60

9.3 Inference of 742 images time comparison using NvidiaP100 GPU or CPU 61

10.1 Comparison of the accuracies of the three models using different weights for training 64

List of Figures

2.1 Primary emotions expressed on the face. From left to right: neutral, right, anger, disgust, happy, sadness, surprise. 5

4.1 SOA of open source deep learning frameworks in 2017 17

6.1 Occlusions caused by hands in spontaneous faces 22

6.2 Differences and intensity in poses interpreting anger 23

6.3 Some examples of FACS Action Units coding system 23

6.4 Translation results on the CelebA dataset by transferring knowledge learned from the RaFD dataset 24

6.5 Transferring expressions in real time RGB videos from an human source to a human target in a youtube video 24

6.6 Some samples of JAFFE database 26

6.7 Some samples of CK+ database 27

6.8 Some samples of Karolinska face Database 28

6.9 Some samples of Radboud Faces Database 29

6.10 Some samples of Pain expressions dataset 30

6.11 A sample of Oulu-CASIA database. Built with three different cameras. 31

6.12 The final dataset has unbalanced distribution of clasess 32

6.13 Data augmentation: Some examples of affine transformations 34

7.1 Localization in the human face of 68 facial coordinate points 38

7.2 Examples of face rotation and cropping finishing in face alignment 40

8.1 VGG16 architecture image by Davi Frossard. 46

8.2 Mean image of the built dataset 49

8.3 VGG16 fine-tuning and LR decreasing. 54

8.4 Loss curve stability differences between different LR (0,01 vs 0.001 in the bottom) 55

9.1 Some of the variety examples of the new dataset that were selected to trick the models 62

10.1 Loss curves differences training Alexnet with different weights and same architecture, dataset and LR 65

11.1 Bad classification: the pose was tagged as anger but classified as sad 68

11.2 No winning class in this predictions 69

11.3 Extreme angle. Image was tagged as neutral 70

11.4 Image tagged as happy. 71

11.5 Two good predictions. 72

11.6 Good predictions: happy and neutral classes were tagged 73

12.1 Wrong classifications in biased classifiers. 76

Chapter 1

Introduction

1.1 Facial Expressions and Emotions

Emotions play a fundamental role in human cognition and they are an essential field in studies of cognitive sciences like neuroscience or psychology. They also play a role in marketing or product management as detecting happiness from a customer can be an evaluation that his/her goals are satisfied. Human emotions produce physiological changes (heart rate, breathing rate, perspiration, hormone levels or facial expressions) that can be automatically detected and interpreted. More specifically emotions can be estimated by facial inference since there is undoubtedly universal connections between emotions and facial expressions, independently of races, genders, ages or cultures. Spontaneous emotions arise even in blind individuals producing the same facial expressions as sighted individuals do [CMD04] and those facial expressions considered to be universal among humans have been observed in nonhuman [PDW07] primates. Human facial expressions communicates emotions and intentions studying them can infer us the observed mood and emotional states that affects our judgment and our decision process system.

1.2 Some applications

Automated study of facial expressions can help studies of visual perception, social interactions and disorders such as schizophrenia and autism leading to discoveries in areas that include detection of pain, frustration, emotions intensity, depression and psychological distress. Automated face analysis can be crucial for Affective Computing a computing discipline research field covering a wide range from robotics to marketing or entertainment. In example, researchers have used facial analysis to spot struggling students in computer tutoring sessions making possible to recognize different levels of engagement or frustration where the works were too easy or too difficult and thus, improving the learning experience in online platforms. Related to face analysis with computer vision techniques there has been some surprising experiments like the automatic identification of criminals [WZ16]. Researchers have developed a deep learning model trained with id photos and able to successfully identify around 83 per cent of the criminals. Obviously, it is a nonsense to condemn someone by it's ID photo but the results of those experiments linked to popular theories of physiognomy and creationism of the 17th century that assumed that a person's character or personality could be judged by their outer appearance, especially the face. Furthermore, it also brings some criticism on the need and importance of policing artificial intelligence research and focus on the variety of datasets. Another application of face analysis should be related driver's attention to the roads, crucial within the increase of assistance to driving systems. Driver drowsiness is the leading causes of traffic accidents and driver monitoring and risk alert must be more general and accessible. Nowadays, it is not the case and some techniques can only be used in certain driving conditions (breaking, lane departure) or make use of complicated sensors like electrical bio-signals that have no practical application. Computer Vision techniques that mainly concentrate on detecting eye closure, yawning patterns and the overall expression of the face or movement of the head are less intrusive and more accessible to the general public as it can be integrated, in example, in a mobile phone.

1.3 Proposal

The conclusions presented after the 5th EmotiW Challenge that took place during the 19th ACM international conference on Multimodal Interaction and can bee seen here were:

- Face and facial parts detection is a bottleneck.
- Deep learning based methods outperform traditional vision, machine learning methods.
- Transfer learning shown useful in several task.
- More labeled data and different emotion labeling strategies should be tested with.

This work aims to fine-tune three different convolutional neural networks (ConvNet) architectures with a new dataset for face analysis that was created expressly for this work. Knowing that there is no public pretrained ConvNet models for face expressions analysis and being emotion recognition a hot topic in disciplines like Human Computer Interaction or Affective Computing it is necessary to establish a baseline for future works. So, it will be built a dataset collecting the publicly available ones. This datasets will be used for training and testing different ConvNets architectures exploring throughout experimentation the best recommendations and methods for fine tuning and measuring times in different hardware configurations. The presented work provides a methodology that covers from collecting data to inference systems passing through face detection and alignment methods. This methodology can be extended to other different purposes that can be solved with images or videos datasets. The best model was tested and integrated in a demo prototyping for driver assistance in a new product of a startup focused on mobility and road-safety in Barcelona.

Chapter 2

Related work: emotions recognition

2.1 Early studies

Studies of the face were greatly influenced in premodern times by popular theories of physiognomy and creationism. Physiognomy, without scientific support, assumed that a person's character or personality could be judged by their outer appearance, especially the face. In the 17th century in England, John Buwler studied human communication with particular interest in the sign language of persons with hearing impairment. His book *Pathomyotomia or Dissection of the significant Muscles of the Affections of the Mind* was the first consistent work in the English language on the muscular mechanism of Facial Expressions. About two centuries later and influenced by creationism, Sir Charles Bell investigated facial expressions as part of his research on sensory and motor control. Subsequently, Duchenne de Boulogne conducted systematic studies on how Facial Expressions are produced publishing photos of facial expressions obtained by electrically stimulating facial muscles. In the same historical period, Charles Darwin firmly placed facial expressions in an evolutionary context. He was one of the firsts to suggest the universality of facial expressions, his ideas about emotions were a centerpiece of his theory of evolution, suggesting that emotions and their expressions were biologically innate and evolutionarily adaptive, and that similarities in them could be seen phylogenetically. At the end of the 19th century he wrote *The Expression of the emotion in Man and Animals*, which largely inspired the study of Facial Expressions of emotions. Darwin proposed that facial expressions are the residual actions of more complete behavioral responses to environmental challenges in example, constricting the nostrils in disgust served to reduce inhalation of noxious or harmful substances or widening the eyes in surprise increased the visual field to see an unexpected stimulus. In the XX century during the 70's, Paul Ekman and Carroll Izard conducted what is known today as the "universality studies" demonstrating high cross-cultural agreement in judgments of emotions in faces by people in both literate and preliterate cultures (indigenous tribes). Friesen's (1972) study documented that the same facial expressions of emotion were produced spontaneously by members of very different cultures in reaction to emotion-eliciting films.

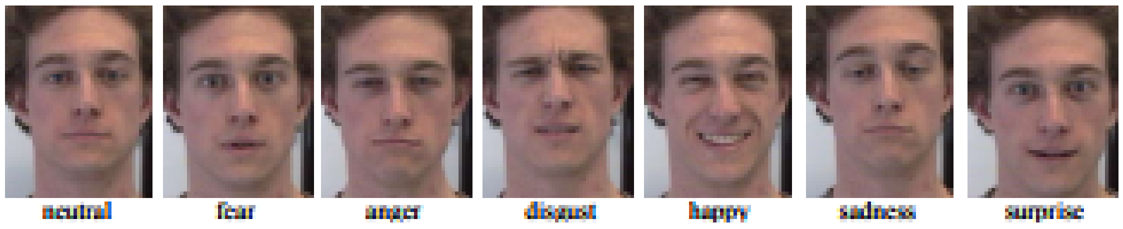


Figure 2.1: Primary emotions expressed on the face. From left to right: neutral, right, anger, disgust, happy, sadness, surprise.

2.2 Description of emotions

The psychologist Paul Ekman studies (beginning on the 60's) stated that emotions are biologically innate, universal to all humans and displayed through facial expressions categorizing six basic emotions (scaling it up to twenty one lately). But, recent studies published by the Institute of Neuroscience and Psychology at the University of Glasgow [JGS14] affirms that the range of humans emotions may be limited to four so, there is not a definite classification model. In facial expressions analysis -in cognitive science and neuroscience- there has been two leading models for describing how humans perceive and express emotions: the continuous and the categorical model. The continuous model defines each facial expression as a feature vector in a face space having the ability of explaining different intensities of emotions. In contrast, the categorical model consists of C classifiers, each tuned to a specific emotion category in general seven emotions classes: neutral, anger, disgusted, anger, happy, fear, sadness and surprise but without having the capacity of detect transitions between those states or the intensity of that emotions. Those six emotions plus the neutral one (or no expression) can be seen, represented by an actor, in the figure 2.1. Both models are used with good results but, science still struggling to define emotions Jaak Panksepp, the father of Affective Neuroscience, during the study of neural mechanisms of emotions, conceptualized affect as a set of seven distinct and basic categories: seeking, rage, fear, lust, care, panic and play. He concluded that, those emotions emerges not from the cerebral cortex but from deep ancient brain structures. On the other side, the social psychologist James Russell stated that all affects can be decomposed into two underlying dimensions: pleasure versus displeasure and low arousal versus high arousal.

2.2.1 Categorical model

Being the objective of this work to automatize the analysis and recognition of emotions with computing software it can be computationally less effective to choose a model that deals with fuzzy classes. The aim of the categorical descriptors are to classify emotions into a set of classes, easy to recognize and described in a daily language. Mainly because of its simplicity and its universality claim those basic descriptors are extensively exploited in affective computing and vastly used in research trying to detect affect of people from Facial Expressions. Although, the categorical model is not valid when it is needed to recognize expressions at different intensities or modes and it has problems explaining combinations of emotion such as happily surprised, angrily surprised or a morphing sequence (video in example) between a happy and a surprise face that are perceived as either happy or surprise but not something in between those.

Chapter 3

Related work: automatic emotions analysis

3.1 Affective computing

It is an interdisciplinary field spanning computer science, psychology, and cognitive science that facilitates the study and development of systems and devices that can recognize, interpret, process and simulate human affects. Affective computing addressed the question of digitization and transmission of emotional experiences through monitoring and interpretation of physiological signals via sensors, microphone, cameras and software logic measuring signals like heart rate, blood pressure, blood-pulse volume,respiration, temperature, pupil dilation, skin conductivity and muscle tension. Combined measures of multiple autonomic signals that infers promising results as components of an emotion recognition system. Picard, Vyzas and Healey, for example, achieved 81% percent recognition accuracy on eight emotions through combined measures of respiration,blood pressure volume and skin conductance, as well as facial muscle tension. Experiments have been conducted throughout multimodal interaction like video plus audio where the subjects are exposed to stimuli and cardiovascular signals and electrodermal activities of the subjects (like galvanic skin response or skin temperature) are recorded. So far, this kind of evaluation techniques are only available in advanced research centers. However, the advances in micro sensing technologies and research made the evaluation of human experience considerably easier and more accessible to individuals without extensive background in psychophysiology. Thus, advances in pervasive computing and human computer interaction made affordable user-friendliness devices that helps development of cognitive engineering and benefits emotions detections being more effective. Into this fields of action, computer vision can help as a non intrusive technique, cameras are everywhere and inferring face emotions can help producing more enhanced products.

3.2 Automated face analysis with computer vision

Machine learning and computer vision researchers are creating computational models helping the studies and experimentation in social sciences. Furthermore, those models are essential for the development of artificial intelligence and robotics, also in human computer interaction (HCI) systems. The first works on automatically analyzing facial expressions were published in the late 70's mostly of them -due to a poor face detection algorithms and limited computational power- received little attention throughout the next decade but the subject of study had a revival in the late 90's when it was published the first big enough [LCK+10] public dataset CK that is still in use in most of the works related to this subject. For achieving good results it is required strong preprocessing before classification algorithms. Multiple steps like face detection and face frontalization or alignment. This implies numerous challenges including some classical computer vision problems like non-frontal faces, occlusions (like sun glasses) different illuminations and also non inherit to computer vision problems like the fact that many facial expressions are inherently subtle making them difficult to model even for humans. Despite this challenges, machine learning and computer vision should proceed to successfully emulate the human capacity of emotions detection through the face and this should be achieved through computers because without any doubt it can aid visual perception, social interactions and disorders such as schizophrenia and autism.

3.3 Related work on automated face analysis

3.3.1 Main challenges

Recently the Emotion Recognition in the Wild(EmotiW) challenge used categorical approach classification with imaging conditions close to real-life, including low and uneven illumination, low resolution, occlusions, non-frontal head-poses, and motion blur. Another difficulties mentioned were relatively small sizes of dataset is, which makes it difficult to train large-scale models and thus, proning to overfitting (it will be demonstrated in this work). Computer vision community is switching to deeplearning as it can be seen in the conclusions presented after the 5th EmotiW Challenge that took part during the 19th ACM international conference on Multimodal Interaction:

- Face and facial parts detection is a bottleneck.
- Deep learning based methods outperform traditional vision, machine learning methods.
- Transfer learning shown useful in several task.
- More labelled data and different emotion labelling strategies should be tested with.

The three winners team of the 3rd EmotiW challenge used deep learning networks in their methods. In the present work those four points will be explored: a new dataset will be build and used for fine tuning a deep neural network that will be used to enhance a mobile application used for increasing security in car drivers. Deep neural networks have evolved to be the state-of-the-art technique for machine learning tasks ranging from computer vision and speech recognition to natural language processing. But before being analized by a ConvNet an automated face analysis begins with face detection regardless of illuminations, occlusions, facial pose, orientation and expression.

3.3.2 Face detection and alignment

Face detection and alignment are challenging due to various poses, illuminations, scales and occlusions. In the case of relatively frontal pose, the Viola and Jones (2004) face detector is probably the most widely used at the present. Recent studies showed that deep learning approaches can achieve impressive performance on those two tasks. Recently, scientists are reaching great levels of accuracy using deep learning approaches. We have, for example, the work of [ZZLQ16] Kaipeng Zhang and Zhanpeng Zhang that proposed a deep cascaded multi-task framework for face detection. Previous widely used techniques were the use of cascade of classifiers like Viola and Jones algorithm that used AdaBoost for detecting possible faces identified by Haar-Like features which was specially efficient and thus widely adopted in digital cameras but it struggles with no full front view of the human face. Another approach for face detection, very useful for face alignment (the contours of the eyes and other permanent facial features) was the deformable parts models. An old idea (Fischler & Elschlager '73) requiring high computational resources that works combining modern features and machine learning. It is also necessary to mention a very efficient approach related to Computer Vision techniques that consists in using HOG features and a linear classifier like Support Vector Machine for detecting faces. This technique as been choose for building the dataset related to this work. In part-based models the subtle idea was that local appearances are more easier to model than the global ones. However, there are other emerging techniques specially for automatic face recognition that focus on the entire image rather than local features. An image is considered as a high dimensional vector and statistical methods are frequently used to reduce the basis vectors where the face vector is projected. Linear methods like Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) reduces the dimension of the training set and facilitates the comparison of the sample image.

3.3.3 Face analysis classification algorithms

On this work it will be choose a deep convolutional model that will be more detailed further on. Among the many methods proposed in the literature, there is a distinction between the ones that do not use deep learning, which are referred as shallow, from ones that are called deep. Shallow methods start by extracting a representation of the face image using handcrafted local image descriptors such as SIFT or HOG then they aggregate such local descriptors into an overall face descriptor by using a pooling mechanism, for example the Fisher Vector. There are a large variety of such methods but this work is concerned mainly with deep architectures as they represent the SOA in others face classification tasks like face recognition. The defining characteristic of such methods is the use of a Convolutional Neural Network (CNN) feature extractor, a learnable function obtained by composing several linear and non-linear operators. Deep CNNs have been introduced more than three decades ago, it is only recently that they become a predominant method in image classification tasks. With the emergence of very large classification datasets, the increase in computation power and algorithmic improvements in training those models, the huge number of model parameters is no longer a limiting factor in applying CNNs in practical settings. Thus, in recent years, deep CNNs have been applied in various image classification problems, including, object recognition, scene recognition, face verification, age and gender classification and more.

3.3.4 Inference in ConvNets

Despite bein powerful on various classification vision related tasks, deeplearning is a burden to deploy in ractical applications or embedded systems since models size of deep learning are generally large and high computational complexity is required. Therefore, in the recent years algorithms to reduce deep trained models size and improve speed have been proposed. Methods to reduce models size consisting in detecting and discarding redundant weights by applying pruning without accuracy drop. Deep compression significantly reduces the computation and storage required by neural networks. For example, for a CNN with fully connected layers, such as Alexnet and VGGnet, it can reduce the model size by 35 to 49 times and in more complex models such as GoogleNet and SqueezeNet, deep compression can still reduce the model size by 10 times. Most of this compression is achieved during training time an example of this technique is [HPN$^+$16] DSD or Dense-Sparse-Dense a novel training method that first regularizes the model through sparsity constrained optimization, and improves the prediction accuracy by recovering and retraining on pruned weights without no loss of prediction accuracy. To reduce model size further, quantization techniques have been introduced such as bit-quantization. In bit-quantization, the least number of bits are utilized for representing information of model while minimizing accuracy loss.

Chapter 4

Related Work: background on 2D CNN

4.1 Choosing a classification algorithm

Deep learning is a popular name for artificial neural network architectures built with a long sequence of hidden layers and used with tremendous success in computer vision challenges. The goal of training such models is to find the parameters of the network that minimise the average prediction log-loss after the output softmax layer. A typical architecture of a convolutional neural network will contain an input layer, some convolutional layers, some dense layers (fully-connected layers), and an output layer. The hidden blocks contains linear operators followed by one or more non-linearities such as ReLU or max pooling. The convolutional blocks works as the linear operator with a bank of linear filters (linear convolution) and the last three blocks are instead called Fully Connected (FC) and are the same as a convolutional layer, but the size of the filters matches the size of the input data, such that each filter "senses" data from the entire image.

4.2 Main layers of a ConvNet

4.2.1 Input layers

The input layer has predetermined fixed dimensions so the image has to be preprocessed before it can be fed into the layer. The input to the networks used in this work, is a face image of size 224x224 with the average face image (computed from the training set) subtracted. This is critical for the stability of the optimisation algorithm.

4.2.2 Convolutional layers

The numpy array gets passed into the Convolution2D layer where it is specify the number of filters as one of the hyperparameters. The set of filters or kernels are unique with randomly generated weights. Those receptive fields, slides across the original image with shared weights to output a feature map. So, convolution generates feature maps that represent how pixel values are enhanced, for example, edge and pattern detection. Other filters are applied one after another creating a set of feature maps. Pooling is a dimension reduction technique usually applied after one or several convolutional layers. It is an important step when building CNNs as adding more convolutional layers can greatly affect computational time. Most original models used a popular pooling method called MaxPooling2D that uses (2, 2) windows across the feature map only keeping the maximum pixel value. The pooled pixels forms an image with dimensions reduced by 4.

4.2.3 Dense layers

The dense layer (aka fully connected layers), is inspired by the way neurons transmit signals through the brain. It takes a large number of input features and transform features through layers connected with trainable weights. These weights are trained by forward propagation of training data then backward propagation of its errors. Back propagation starts from evaluating the difference between prediction and true value, and back calculates the weight adjustment needed to every layer before. We can control the training speed and the complexity of the architecture by tuning the hyper-parameters, such as learning rate and network density. As we feed in more data, the network is able to gradually make adjustments until errors are minimized. Essentially, the more layers/nodes we add to the network the better it can pick up signals. As good as it may sound, the model also becomes increasingly prone to overfitting the training data. One method to prevent overfitting and generalize on unseen data is to apply dropout. Dropout randomly selects a portion (usually less than 50%) of nodes to set their weights to zero during training. This method can effectively control the model's sensitivity to noise during training while maintaining the necessary complexity of the architecture.

4.2.4 Output layers

This output presents itself as a probability for each emotion class. Therefore, the model is able to show the detail probability composition of the emotions in the face. Our expressions are usually much complex and they contain a mix of emotions that could be used to accurately describe a particular expression.

4.3 CNN choosing a framework tool

The deep learning area had a first widely adopted framework called Theano. Unfortunately, there will be no further development work on Theano in 2018. In the past years, different open source python deep learning frameworks were introduced, often developed or backed by one of the big tech companies currently TensorFlow by Google is expected to dominate the market for years. PyTorch, introduced by Facebook in January 2017 it is a port to the popular Torch framework (implemented in C with a wrapper in Lua) with the binaries wrapped in GPU accelerated Python. Next to the GPU acceleration and the efficient usages of memory, the main driver behind the popularity of PyTorch is the use of dynamic computational graphs. The advantage of these dynamic graphs is that they are defined on the run instead of the traditional define and run and this is especially important for cases where the input can vary for example, with unstructured data like text. Other tech giant company Microsoft, developed an internal deep learning framework called CNTK and officially launched the 2.0 version in 2017 after renaming it to the Microsoft Cognitive Toolkit. In 2017, Facebook also launched Caffe2 the successor of the well known Caffe framework. The original Caffe framework, developed by the Berkeley Vision and Learning Center is extremely popular for it's community, it's application in computer vision and it's Model Zoo, a selection of pretrained models. It is also the framework most supported by Nvidia the leader of the GPUs ecosystem. Another popular deep learning framework is MXNet, supported by Microsoft and Amazon that it actually supports many languages, from C++ to Python, JavaScript, Go, and, indeed, R. MXNet stands-out is it's scalability and performance. Close to all these frameworks, we also have interfaces that are wrapped around one or multiple frameworks. The most well know is Keras a high-level deep learning API, written in Python and created by François Chollet. Google announced in 2017 that Keras has been chosen to serve as the high-level API of TensorFlow and Keras will be included in the next TensorFlow release. Next to TensorFlow, Keras can also use Theano, Caffe or CNTK as backend. Hence, even for senior researchers and developers it is hard to keep up with the latest developments. There is a compilation of state of the art frameworks in figure 4.1. A positive reaction from some companies to all this constantly updates and releases of deep learning frameworks is the release of the **Open Neural Network Exchange (ONNX)**. Announced in September of 2017 ONNX is an open format to represent deep learning models. This allows users to more easily move models between different frameworks. For example, it allows you to build a PyTorch model and run the model for inference using MXNet. ONNX is launched by Microsoft, Amazon and Facebook amongst others so ONNX supports Caffe2, Microsoft Cognitive Toolkit, MXNet, and PyTorch from the start and Tensorflow through a open source converter.

In this work, the framework chosen is Caffe, a deep learning framework made with speed and modularity in mind. It is developed by the Berkeley Vision and Learning Center (BVLC), as well as great community contributors and it is popular for computer vision purposes. Caffe supports Nvidia cuDNN (framework for convolutional operations in GPUs) for GPU acceleration and the

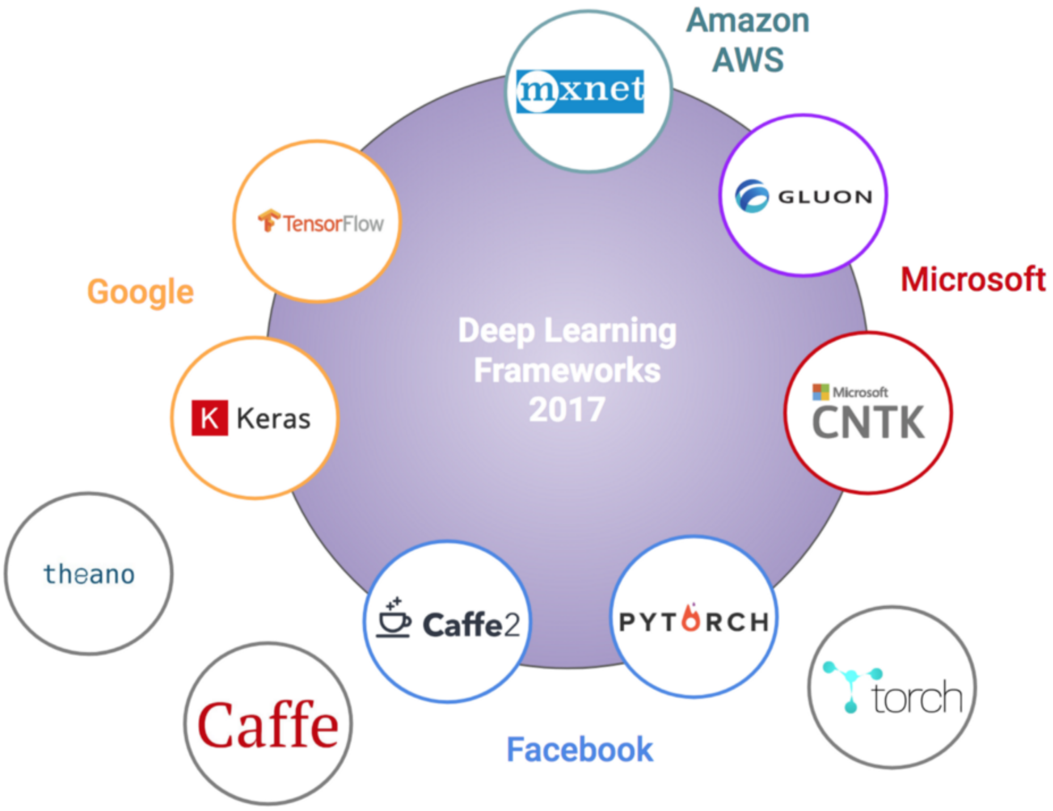


Figure 4.1: SOA of open source deep learning frameworks in 2017

supported interfaces are C, C++, Python, MATLAB and the linux command line interface. Models and optimization are defined by configuration without hard-coding and the switch between CPU and GPU is as simple as setting a single flag to train on a GPU machine. The execution speed makes Caffe perfect for research experiments and industry deployment as it can process an image in milliseconds for learning and faster in inference mode specially using recent library versions adapted to recent hardware like the Nvidia implementation for inference called TensorRT. Caffe (recently surpassed by Tensorflow) has the greatest community and powers lots of academic research projects, startup prototypes, and even large-scale industrial applications in vision, speech, and multimedia. This framework has also been choose because of pre-trained models available for a variety of domains becoming great for finetuning networks.

For using Caffe a GUI tool has been choose for training, analyzing, visualization and tuning the networks. This tool is developed by Nvidia and it is called Digits. Nvidia provides tools for all the deeplearning pipeline and they are optimized for using CUDA (that are the libraries or drivers that connect Caffe with the GPU) and allows the parallelization and the speed up the mathematical computations (specially matrices). Nvidia Digits and Caffe framework also use the Nvidia CUDA Deep Neural Network library (cuDNN) a GPU-accelerated library of primitives for deep neural networks such as forward and backward convolution, pooling, normalization, and activation layers.

Chapter 5

Contribution of this work

This work aims to be a proposal of a working methodology for solving specific problems related to computer vision. The work also makes some recommendations for fine-tuning ConvNets and specially how to deal with the overfitting problem. A new dataset, related to face analysis and emotion recognition through face analysis, has been built. The CNN model built will try to contribute to a company software technology that pursues to assist car drivers with the intention of reducing driving accidents. So, it was needed to analyze the face of the driver and this occurs in a relatively controlled environment. Apart of face analysis, this working workflow can be used for solving other computer vision related problems like: obscene image classification in second hand selling apps or in social networks. It also can be used for tracking sell products in supermarkets or monuments recognition in tourism guides. The pipeline proposed in this work included the following points:

- The main datasets for face analysis are explored and explained their advantages and constraints.
- There is a compilation of already trained (with human faces) deeplearning networks and why they were choose.
- There is an explanation for how to choose a CNN framework that can cover all the deeplearning cycle including training and inference.

- There is a compilation of how to create a face analysis dataset and how to get it reliably labeled.
- There is also a part for how to build a solution for preprocessing the data that crops and align faces made with python standard libraries.
- There is also a part that explain how to augment our dataset and achieve a more balanced classes division.
- There is an explanation about how the data sets are divided between training, validation and testing sets.
- There is a fine tuning part for training CNN with small datasets where it can be found a parameters compilation and some useful tricks learned after training.
- After six experiments that were evaluated with an isolated test set there is a model recommendation.
- There is a guide recommendation for software and hardware environments with training times in function of the GPU.
- There is a deployment recommendations part with a python recommendation for inference and time comparison between using GPU or CPU.
- There is a part that deals with the generalization capacities of the models and the problem of overfitting.
- And finally there is a chapter that shows the results of test inference in real driving conditions.

In summary, the methodology proposed in this work involves all the cycle of a image classification pipeline from data collection to deploying a CNN model including recommendations for fast training and tricks for avoiding non evidence problems like non generalization models.

Chapter 6

Dataset collection and augmentation

With the ubiquity of cameras on computers and mobile devices, there is a growing interest in bringing face analysis applications to the real-world. To do so, data collected and labeled from real-world environments or laboratories is needed. In this sense, public datasets truly helps accelerate research, not just because they provide a benchmark, or a common language for researchers that can compare their different algorithms in an objective manner, but also because compiling such a corpus and getting it reliably labeled, is tedious work requiring a lot of effort which many researchers may not have the resources to do.

6.1 Related work: existing datasets

In other specific problems related to object recognition there exists public projects like the Imagenet dataset or KITTI Vision Benchmark Suite. In face analysis there is no such a quantity public datasets available for face analysis and moreover, the existing ones lacks of standardization between them. The available data have substantial differences between them specially in how they are organized and labeled making some of them not useful for the purpose of this work because they demand high time and computing adaptations. In general, those datasets where built (and make publicly available) for solving specific problems and they lack of general methods and standards that help their reuse for other purposes. Some of the basic differences existing in the available datasets are:

- Datasets built in a laboratory or controlled environment versus naturalistic ones that reflects the conditions find by real world applications. Things like lighting or occlusions (sunglasses or hands on the face) as it can be seen in figure 6.1.
- Usage of professional actors (or scientists making the job of an actor) interpreting the face expression versus spontaneous reactions to videos or images selected for provoking those same reactions as it can be seen in figure 6.2. In general interpreted poses are more easy to detect as they are more intense than spontaneous ones.
- Differences in how the labeler interprets what a specific pose is because, sometimes there are so subtle differences that it is difficult to classify from a human perspective but ConvNets can pay attention to those details.
- Differences in labeling and annotations. Some datasets make use of the seven emotions classes but most of them make use of FACS action units as it allows different uses of the same dataset.
- 3D models versus 2D models.

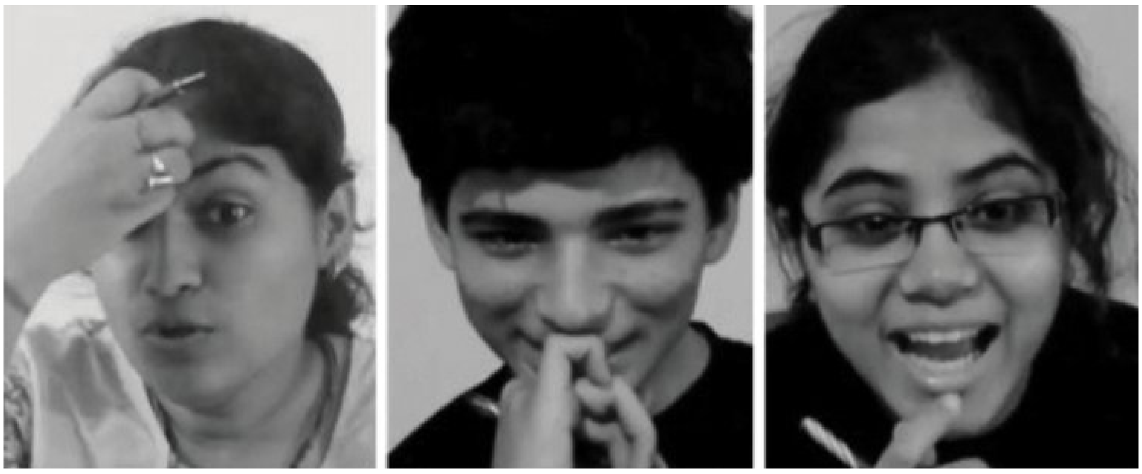


Figure 6.1: Occlusions caused by hands in spontaneous faces

At this point it is necessary to write about the coding system called FACS and why it is not useful for this work. FACS was developed by Ekman and Friesen [EF71] to measure the facial

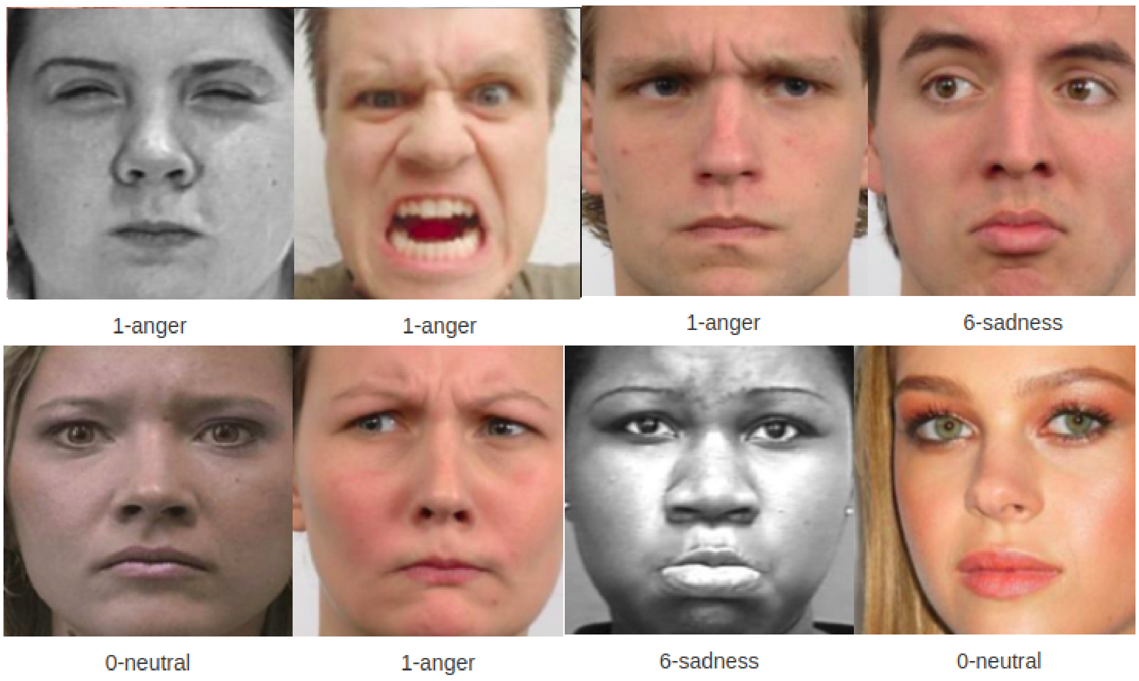


Figure 6.2: Differences and intensity in poses interpreting anger

behavior and it codes different facial movements into Action Units (AU). Some examples can be seen in figure 6.3

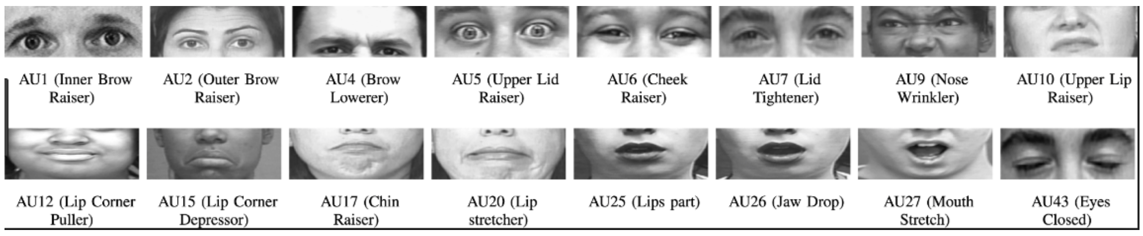


Figure 6.3: Some examples of FACS Action Units coding system

Based on the underlying muscular activity that produces momentary changes in the facial expressions it allows an expression to be recognized by correctly identifying the combination of action units related to a particular expression. The use of FACS makes the dataset more generally applicable and can be applied to different uses (other than analyzing emotions). But, the conversion between FACS and emotions is a computation difficulty that can't be slighted. Meanwhile, it should be mentioned that FACS poses a new utility of the dataset: expressions can be generated by computers. In recent years, facial animations attracted a lot of attention in the computer graphics community and a number of successful solutions have been proposed, making use of motion capture techniques to record facial expressions portrayed by actors. Algorithms can render high quality animations reproducing the facial expressions recorded. However, despite the high quality graphics that can be produced, the technical investment is such that it is very unlikely to appeal to researchers in psychology or neuroscience. Recently, in 2017 there has been presented a work [CCK+17] that propose StarGAN, a generative adversarial network capable of synthesizing facial expressions of face images and also generating facial attributes from still images. Spectacular results can be seen in figure 6.4 Another recent work is [TZS+16] the so called Face2Face where it is presented

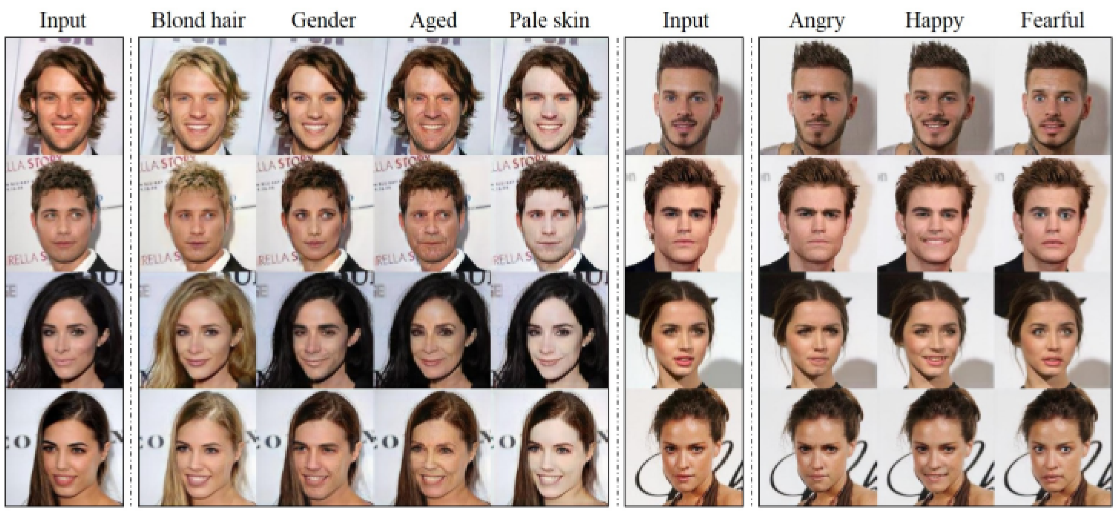


Figure 6.4: Translation results on the CelebA dataset by transferring knowledge learned from the RaFD dataset

an approach for real-time facial reenactment where they animate and manipulate the face of a target video by a source actor analyzing in real time the face expressions of a source video and transferring them to a target video re-targeting those expressions in a very realistic manner. A clear example can be seen in figure 6.5.

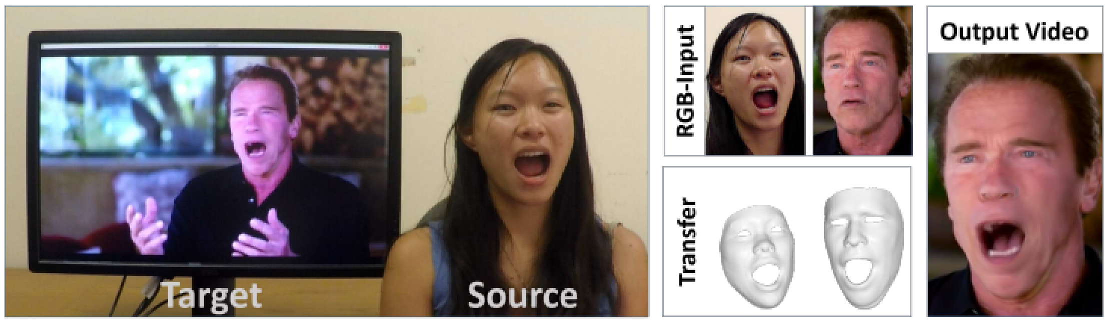


Figure 6.5: Transferring expressions in real time RGB videos from an human source to a human target in a youtube video

Pattern analysis depends hugely on the number of training examples or the result will often leads to over-fitting. Recently,it is possible to find some datasets of spontaneous facial expressions that had been collected in naturalistic settings (viewers watching video content) but they lack of precise labeling specially for indicating when a pose begins and finish. It is not clear when a pose morph to another (from neutral to fun). Those spontaneous datasets are not so useful in terms of productivity (time/results). On the other hand, the datasets built in studios with actors have low numbers of models and examples as most of them were built in the late 90's or early 2000's. So, in further works it is needed to consider the evolution and inclusion of spontaneous face detection and alignment pipelines for covering the needs of commercial and industrial systems. Analysis systems must learn to model not just inter-class appearance variations (differences between different people) but also intra-class variations (same person appearance variations) and this appears to remain a challenge for data collection efforts.

6.2 Selected datasets

In this work it has been built a new dataset collecting public existing ones. It has been done through an extensive research of public (for research purposes) available datasets in face emotions analysis. From the available datasets it has been discarded all the databases that were built or prepared around 3D models of the human face as the preprocessing of our data has not been done with 3d masques models. It has also been discarded those labeled with FACS units due to computing complexity and it has also been discarded videos datasets (mostly spontaneous faces) most of them are built with poor illumination changes or angles variations and poor labeling. The finally incorporated datasets were:

6.2.1 The Japanese Female Facial Expression (JAFFE) Database

Collected from the [LAK+98]paper the database contains two hundred and thirteen images of seven facial expressions posed by ten Japanese female models. The photos were taken at the Psychology Department in Kyushu University and some samples can be seen in the figure 6.6.

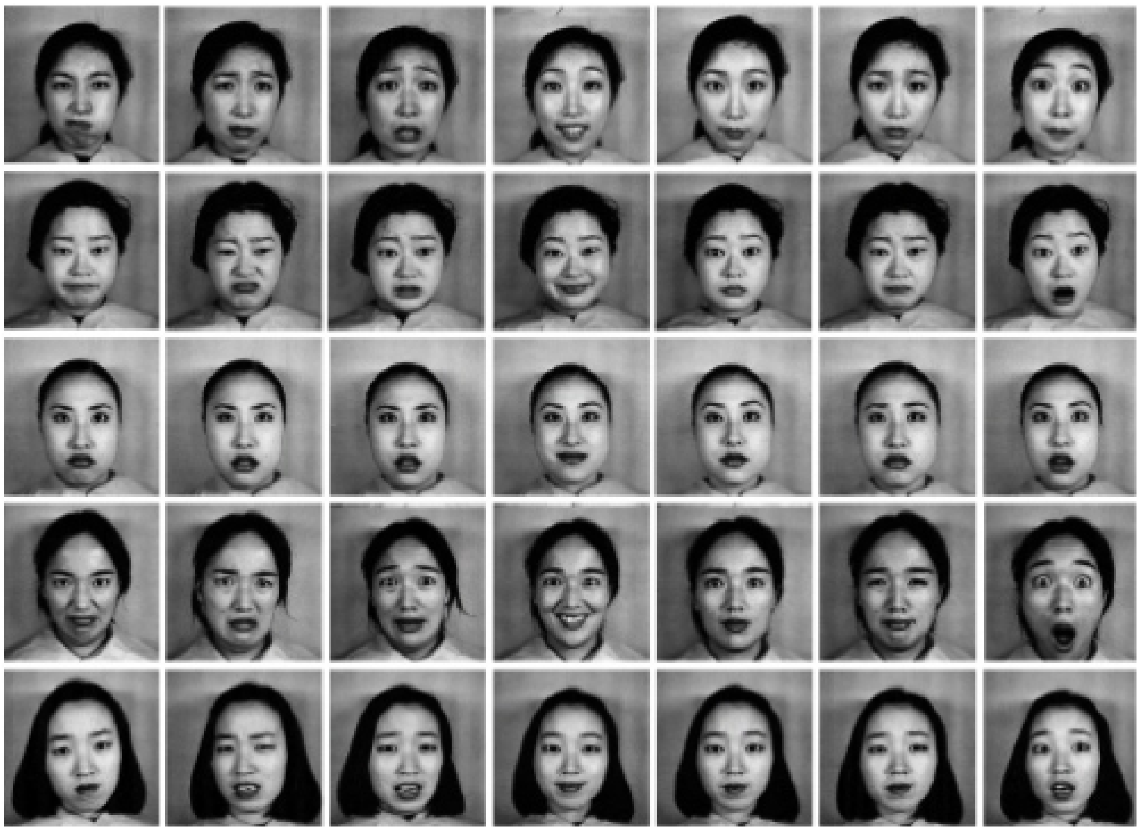


Figure 6.6: Some samples of JAFFE database

6.2.2 The Extended Cohn-Kanade Dataset (CK+)

The [LCK+10]CK database is one of the most widely used dataset for face analysis (AU and emotions detection) but, it has one great limitation for the purpose of emotional face analysis, while AU codes are well validated, emotion labels are not. The Extended Cohn-Kanade (CK+) database completes this dataset with emotion labels been revised and validated. Some samples can be seen in figure 6.7

6.2.3 Karolinska Directed Emotional Face(Kdef)

This [CL08] database is built by scientists Ellen Goeleven, Rudi De Raedt, Lemke Leyman, and Bruno Verschuere from Ghent University in Belgium.It is a subset of the original KDEF database that had 490 JPEG pictures showing 70 individuals (35 women and 35 men) displaying 7 different emotional expressions. Each expression is viewed from 5 different angles. All the individuals were trained amateur actors between 20 and 30 years of age. For participation in

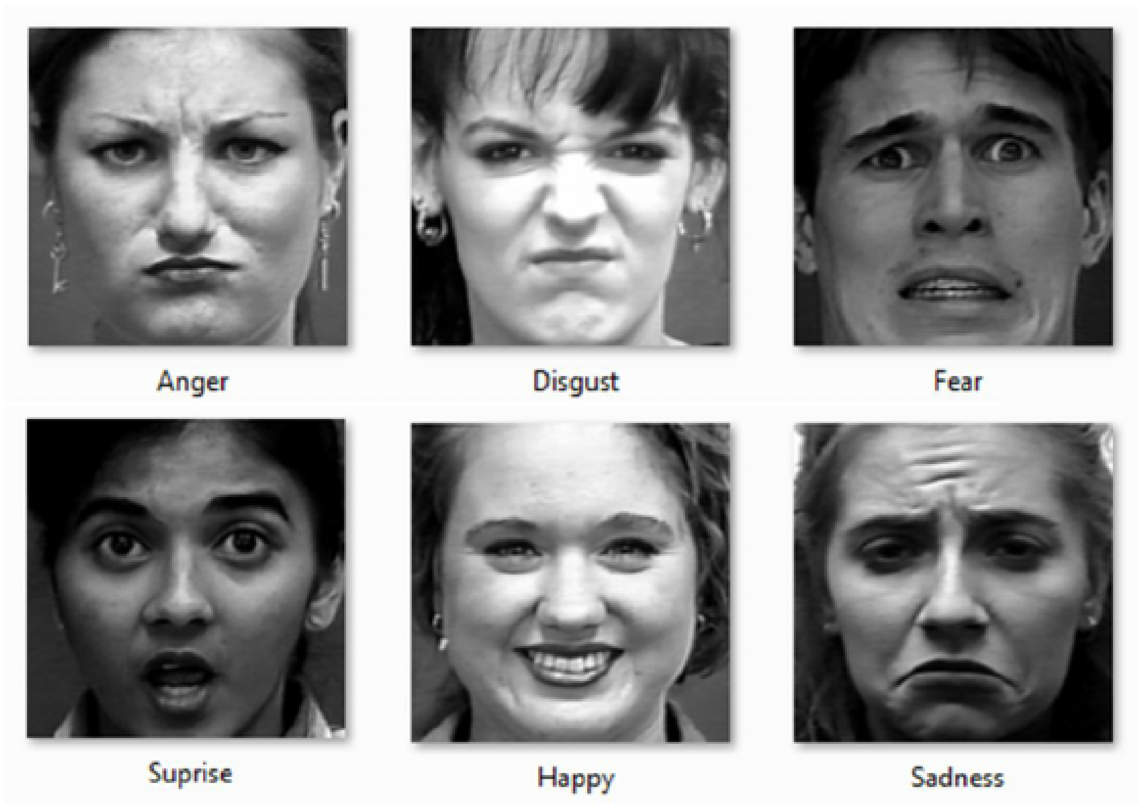


Figure 6.7: Some samples of CK+ database

the photo session, beards, moustaches, earrings, eyeglasses, and visible make-up were exclusion criteria. All the participants were instructed to try to evoke the emotion. Some angles of the pictures can not be aligned so they was discarded as it can be seen in figure 6.8

6.2.4 Radboud Faces Database (Radboud)

The Radboud Faces Database (RaFD) [LDB$^+$10] is an initiative of the Behavioural Science Institute of the Radboud University Nijmegen, (the Netherlands) from 2010 and including a set of pictures of 67 models (including Caucasian males and females, Caucasian children, both boys and girls, and Moroccan Dutch males) displaying 8 emotional expressions as it can be seen in figure 6.9. So, one emotion representation had to be discarded.

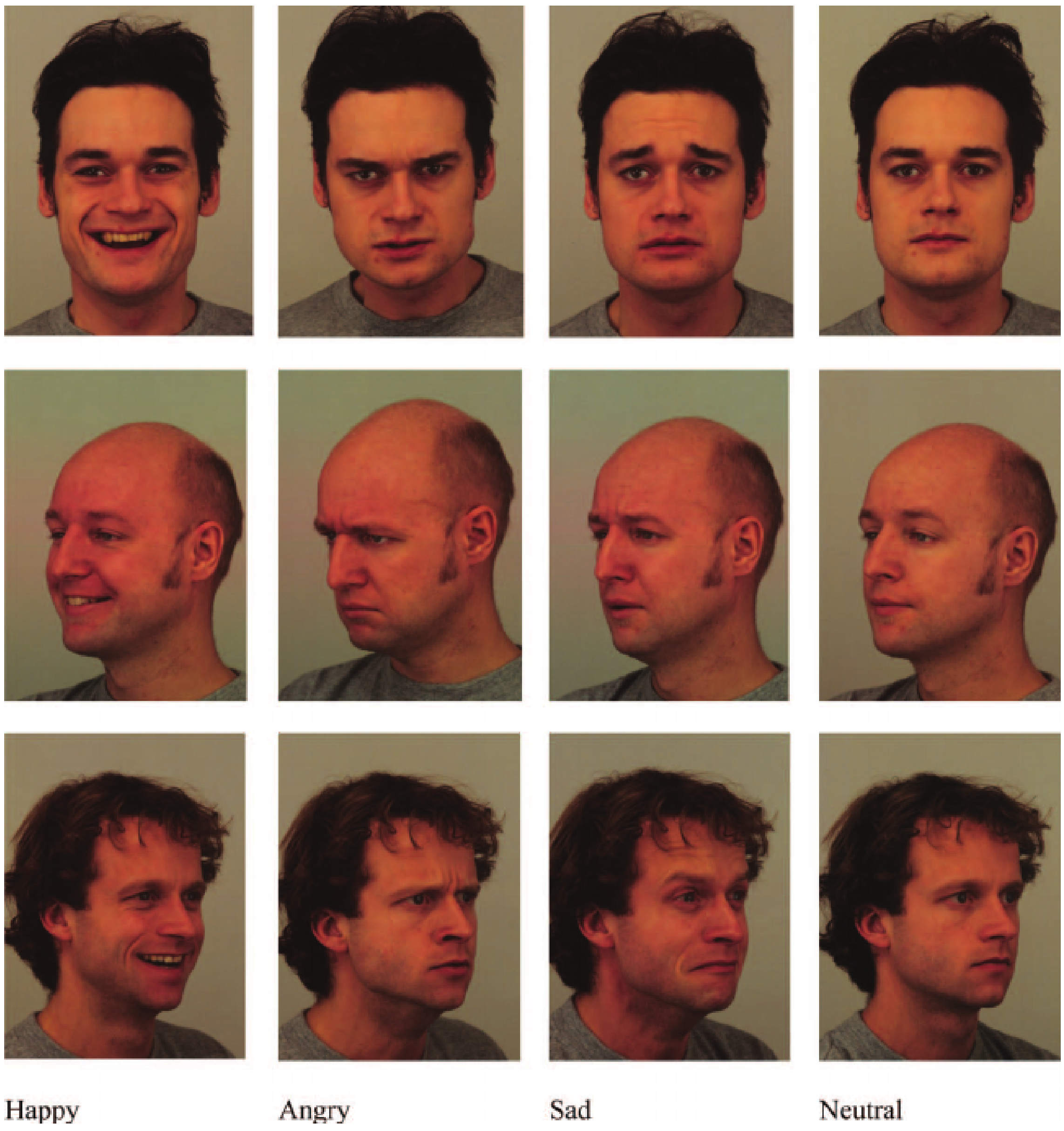


Figure 6.8: Some samples of Karolinska face Database

6.2.5 Pain expressions

Those are images prepared by the [Psy99] Psychology, School of Natural Sciences from the University of Stirling. This is a collection of images useful for conducting experiments in psychology and it has images of the seven emotions. It was specially useful for augmenting the intra-variability of the classes but, they employed only a few actors so unfortunately not enough samples, some of then can be seen in figure 6.10.

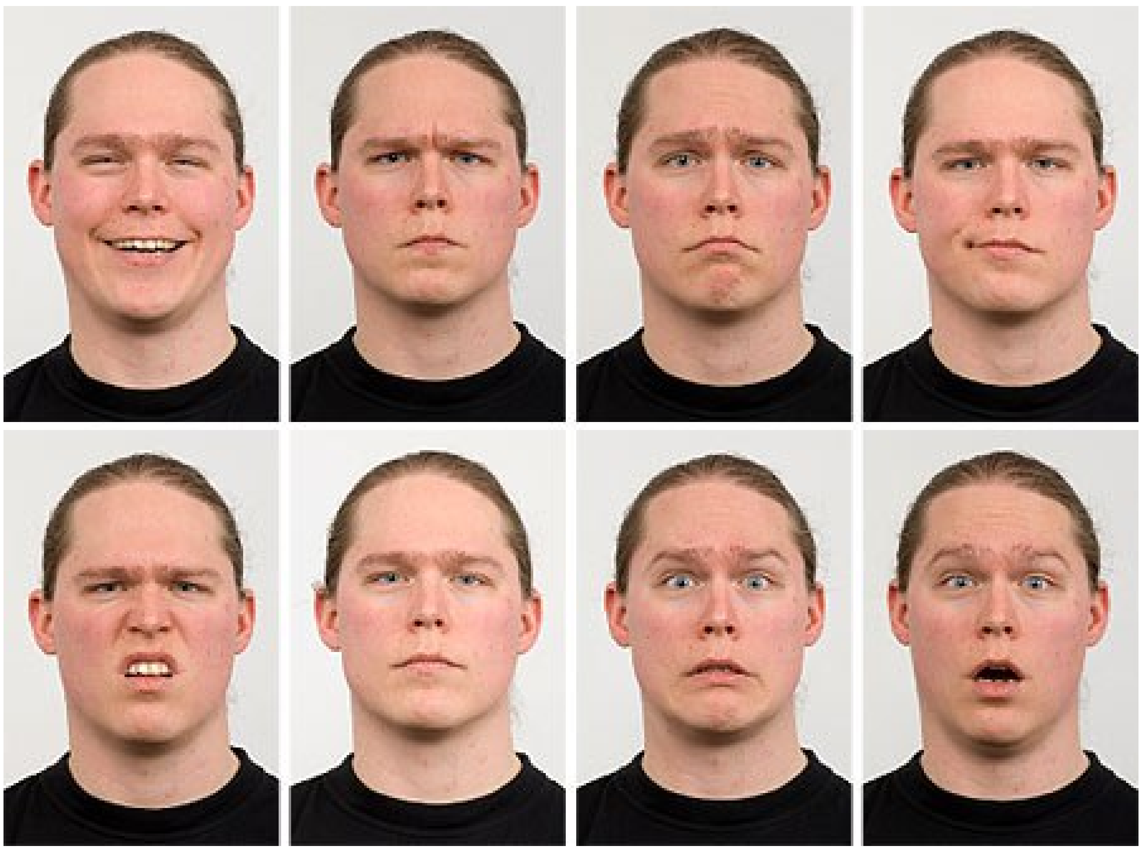


Figure 6.9: Some samples of Radboud Faces Database

6.2.6 Oulu-CASIA

The facial expression database named [ZHT$^+$11] the Oulu-CASIA NIR-VIS database consists of videos with the six expressions (surprise, happiness, sadness, anger, fear and disgust) from 80 subjects captured with two imaging systems, NIR (Near Infrared) and VIS (Visible light), under three different illumination conditions: normal indoor illumination, weak illumination (only computer display is on) and dark illumination (all lights are off). The whole database includes two parts, one was taken in 2008 in Oulu by the Machine Vision Group of the University of Oulu, consisting 50 subjects and most of them are Finnish people. The other was taken in April 2009 in Beijing by the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, consisting of 30 subjects and all of them are Chinese people. The authors also provide still images of the video sequence so we can incorporate this images. All the images will be incorporated disregarding type of camera, light conditions. This differences can be seen in figure 6.11

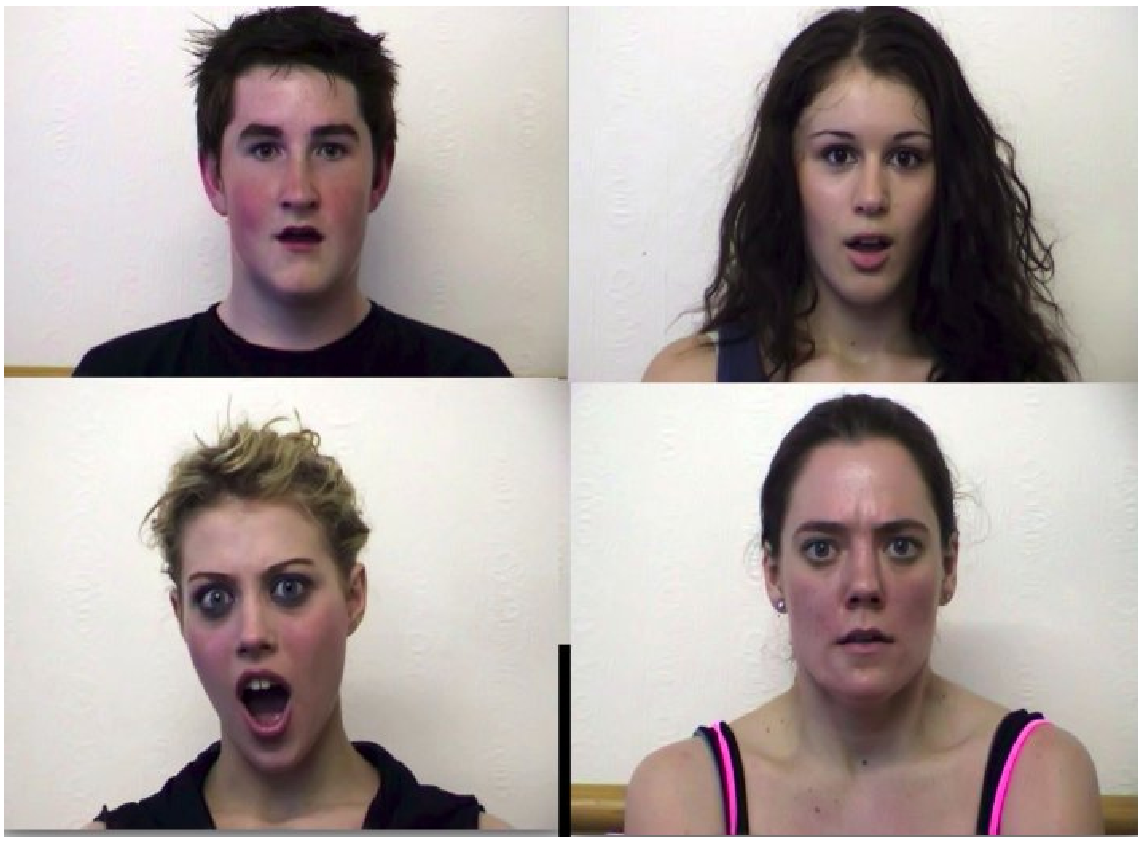


Figure 6.10: Some samples of Pain expressions dataset

6.2.7 Dartmouth Database of Children's Faces

The [DGD13] Database contains images of forty male and forty female models between the ages of 6 and 16. This dataset has to had a very special treatment due to the privacy requirements because the models are minors so there will be no public images. They are photographed on a black background and are wearing black bibs and black hats to cover hair and ears and from five different camera angles and posing in eight different facial expressions.

6.2.8 California Facial Expressions (CAFE)

This dataset [DCR01] was created with the support of the National Institute of Mental Health in 2001 by Matthew Dailey, Garrison W. Cottrell, and Judith Reilly. It is only 146 images so it will be used for testing purposes.

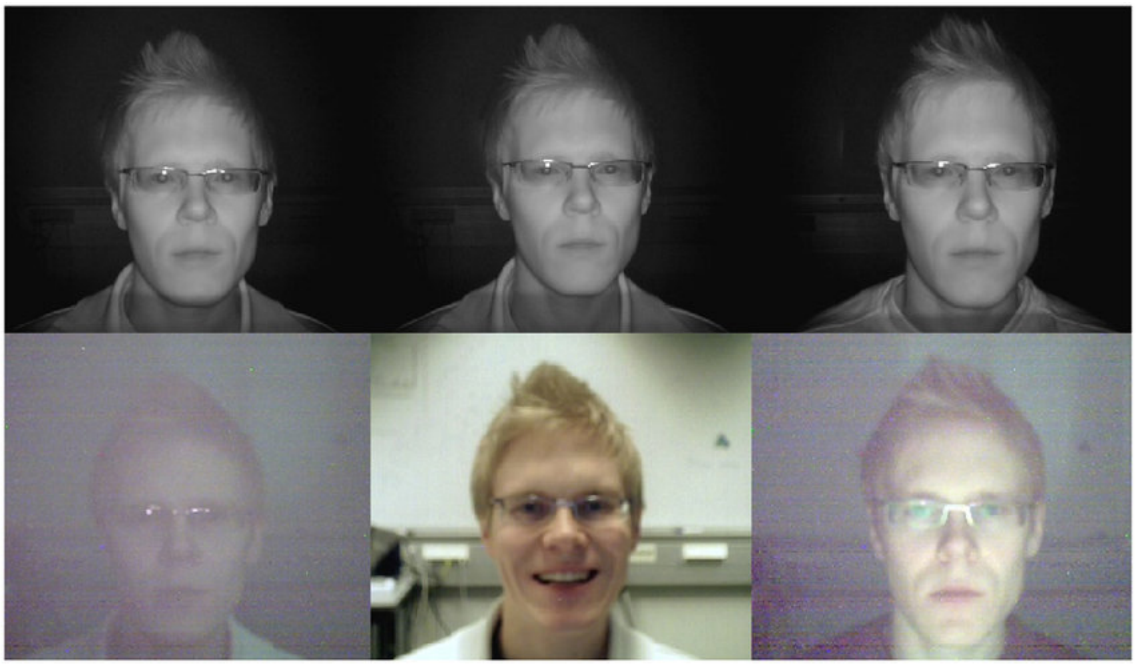


Figure 6.11: A sample of Oulu-CASIA database. Built with three different cameras.

6.2.9 Face Place (Face)

This face database was created by the Tarrlab at Brown University (Tarrlab is now at Carnegie Mellon) in 2008 and it includes multiple images for over 200 individuals of many different races with consistent lighting, multiple views, real emotions, and disguises in jpeg format, 250x250 72 dpi 24 bit color. Stimulus images courtesy of Michael J. Tarr, Center for the Neural Basis of Cognition and Department of Psychology, Carnegie Mellon University. Emotions poses could not be found and 541 neutral images could be saved and used for testing the generalization capacieties of the trained networks.

6.3 Constructed dataset

The total images collected are **37848** and in the table 6.1 it can be seen how they are divided and the sources dataset of each class. As it can be seen in the table the principle source is the dataset Oulu-CASIA and this is caused because it was built with three different cameras so we have three different samples of the same pose and same actor as it can be seen in figure 6.11 that augmented the intra-class variations (same person appearance variations). The final

Class	Total	JAFFE	CK+	Kdef	Radboud	Internet	Pain	CASIA	Dartmouth
Anger	6647	30	546	111	201	375	46	5178	160
Fear	6231	32	293	111	201	47	273	5114	160
Surprise	6059	30	730	112	201	173	92	5011	160
Disgust	4385	29	479	111	201	112	48	3245	160
Happy	3556	31	714	111	201	355	46	1938	160
Sadness	2949	31	297	112	201	180	0	1968	160
Neutral	2566	30	1633	111	201	383	48	0	160
SUM	32843	213	4692	779	1407	1625	553	22454	1120

Table 6.1: Total numbers and class division of the images compiled for the dataset

dataset is quite unbalanced as it can be seen in figure 6.12 and it that has been compensated doing data augmentation.

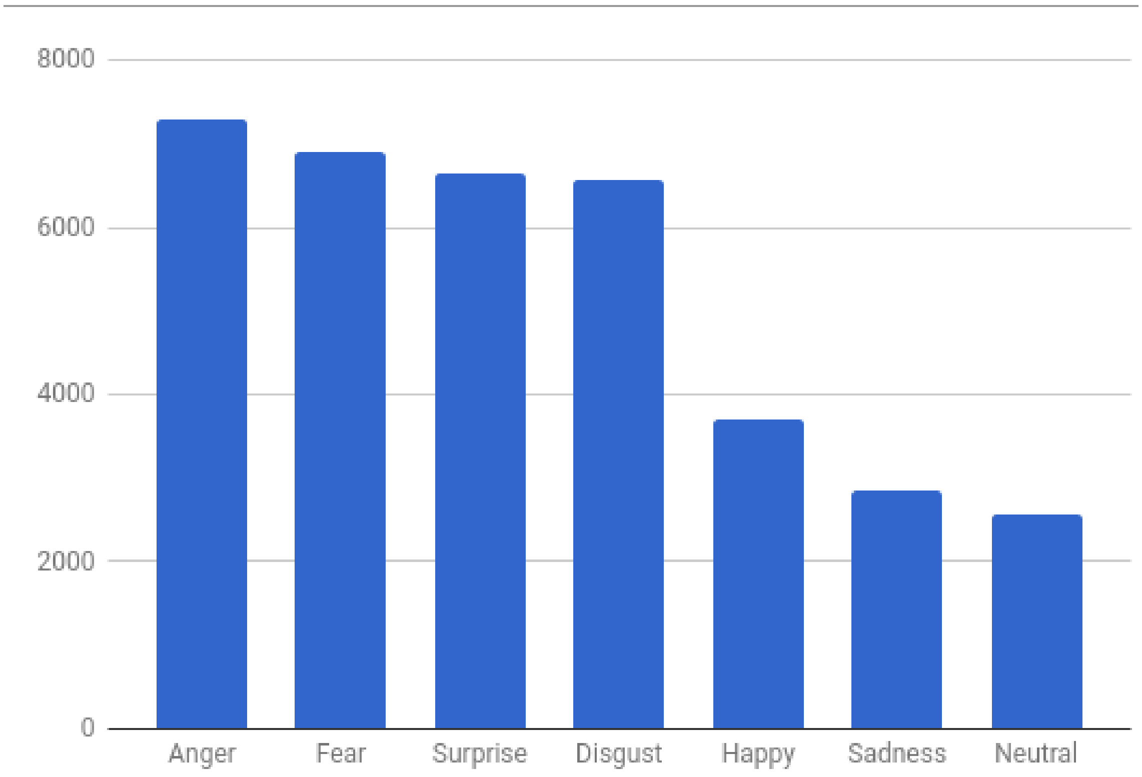


Figure 6.12: The final dataset has unbalanced distribution of clasess

6.4 Data augmentation

Deep artificial neural networks require a large corpus of training data in order to effectively learn and in the case of transfer learning for avoiding overfitting. Collection of such training data is often expensive and laborious. The problem with small datasets is that models trained with them do not generalize well to data from the validation and test set. It is common knowledge that the more data an ML algorithm has access to the more effective it has to be even, when the data is of lower quality. Algorithms can actually perform better, as long as useful data can be extracted by the model from the original data set. Previous works [WP17] had demonstrated the effectiveness of data augmentation through simple techniques, such as cropping, rotating, and flipping input images. A very generic and accepted current practice for augmenting image data is to perform geometric and color augmentations, such as reflecting the image, cropping and translating the image even changing the color palette of the image.

In this work the augmented data will be generated before training the classifier as computing resources have to be free for training CNN purposes. Despite that, in most of the deeplearning frameworks those transformations can be made during training time with the advantage that it generates the necessary images online and those will never be repeated during the training experiments. The problem is that this online generation takes time and resources that will increase the final financial costs of the experiments as they are realized in a private cloud provider. On the case of the dataset of this work geometric methods had been applied: flipping, rotation and blurring. The augmentation has been done with a python library called Augmentor. In principle, Augmentor consists of a number of classes for standard image manipulation functions, such as the rotate class or the crop class. Augmentor applies operations to images stochastically as they pass through the pipeline, according to a user-defined probability value for each operation. Four actions have been choose and the library will randomly apply them in execution time the change are very subtle as the training images were already heavily preprocessed:

- Flip horizontally, rotate the image 2 degrees to the left, apply blur and a small quantity of noise.
- Flip horizontally, rotate the image 3 degrees to the left and a small quantity of noise.
- Flip horizontally, rotate the image 2 degrees to the right, apply more blur and a high quantity of noise.

- Flip horizontally, rotate the image 2 degrees to the left and a higher quantity of noise.

Some examples of this transformations can be seen in figure 6.13 where there has been applied horizontal flipping a subtle degrees of rotation and different levels of noise and blur. The final numbers of the transformed images can be seen in the table 6.2 The deeplearning networks

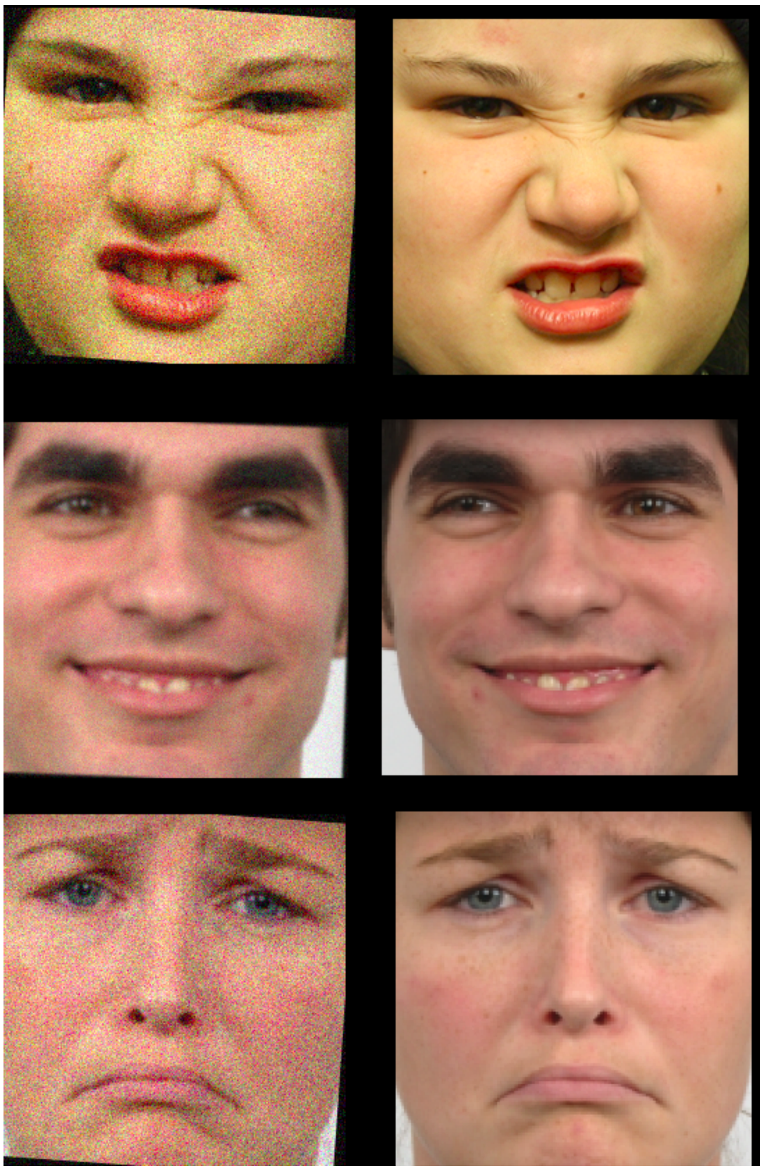


Figure 6.13: Data augmentation: Some examples of affine transformations

have all been trained (fine tuned) with both datasets the original and the augmented in the same conditions to test the effects of data augmentation. The data augmentation has not been applied in the test set, a partition that has been created and isolated for testing purposes.

Class	Original images	Augmented	Total
Anger	5797	0	5797
Fear	5275	469	5744
Surprise	5082	547	5629
Disgust	5032	545	5577
Happy	2839	2466	5305
Sadness	2183	1883	4066
Neutral	1971	1652	3623
SUM	28179	7562	35741

Table 6.2: Augmented dataset final numbers

Chapter 7

Data preprocessing

Many facial recognition algorithms, including Eigenfaces, Fisherfaces and deeplearning methods can all benefit from applying face alignment. This is due to the fact that a CNN learns by continually adding gradient error vectors computed in backpropagation method (multiplied by a learning rate) to various weight matrices throughout the network, as training examples are passed through. If input training vectors aren't scaled, the ranges of distributions of feature values would likely be different for each feature. Thus, the learning rate would cause corrections in each dimension that would differ (proportionally speaking) from one to another. We might be over compensating a correction in one weight dimension while under compensating in another. It is very common to [HRBLM] align the faces in a dataset before training a face recognizer and by performing this process, we obtain higher accuracy in the face recognition models.

7.1 Face detection and cropping

It has been used a code provided by the [Ros17] the author Adrian Rosebrock on his website that makes cropping and alignment. To extract faces from images it is used the Ddlib face detector [Kin09]. Dlib is a a popular C++ toolkit that contains pretrained models for face detection in an image. It provides the coordinates of a rectangular frame that fits the face. The frame coordinates are then used to crop faces from images. The face has to be detected from the input image and for that purpose it will be used a **HOG face detector**. The histogram of oriented gradients is a method from 2005 that starts converting the image to grayscale then it looks at every single pixel in the image one at a time. For every single pixel, it looks at the pixels surrounding it with the goal is to figure out changes of intensity between those pixels. Then it draws an arrow showing in which direction the image is getting darker and repeating the process for every single pixel in the image it ends up with an image called gradients that show the flow from light to dark across the entire image. After this process it will divide the image into small squares (normally 16x16 pixels each) and count in each square how many gradients points in each major direction and draw an arrow of the strongest direction. The resulting is a much more simple representation that captures the basic structure of a patterns existing in our image. To find faces in this HOG image it will be compared to a known HOG patterns extracted from other training faces

7.2 Face alignment

The process of face alignment works in this way:

1. Identifying the geometric structure of faces in digital images.
2. Attempting to obtain a canonical alignment of the face based on translation, scale, and rotation.

Others alignment methods than the proposed one, had been discarded like: imposition of 3D models for applying a transformation to the input image. After identifying faces and draw a bounding box rectangle around the detected face the algorithm move to apply landmark prediction to detect the basics characteristics that all the human faces have in common. We commonly all have mouse, nose and two eyes so the algorithm has to detect, label and extract face regions including: mouth, right eyebrow, left eyebrow, right eye, left eye, nose and the jaw. The facial landmark detector implemented inside C++ computer vision library Dlib produces 68 mapping points using a shape predictor trained on the labeled *iBug 300-W* dataset, see figure 7.1 for a sample of this points. This predictor is called a facial landmark predictor model and called from Dlib library code for predicting the situation of principal face landmarks. Once we have the landmarks detected the different face regions can be accessed through Dlib

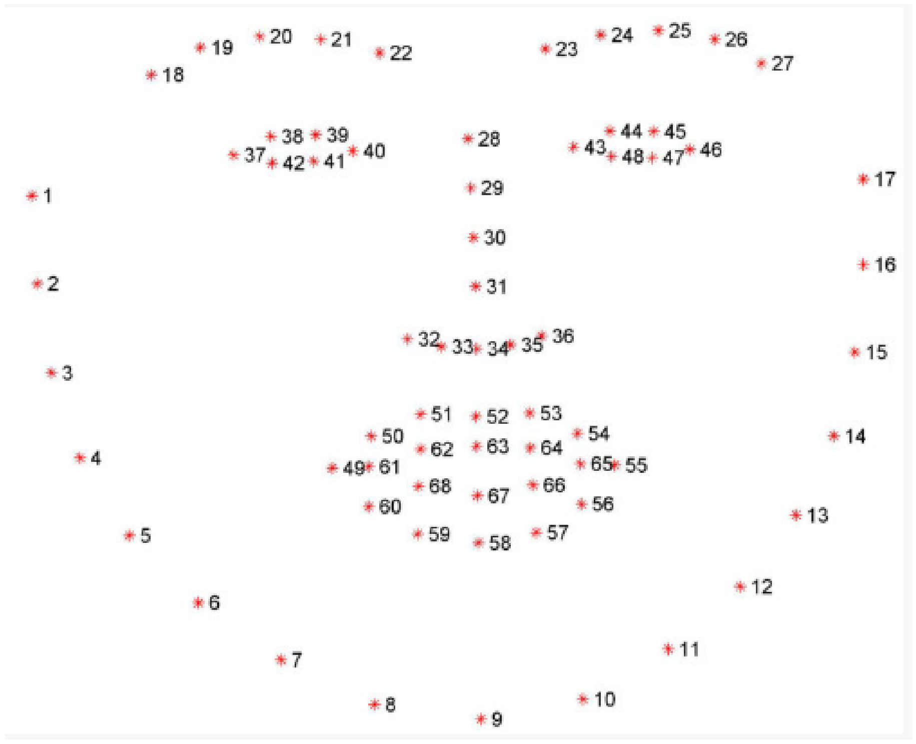


Figure 7.1: Localization in the human face of 68 facial coordinate points

standardized points. For the purpose of centering the face, it is going to be used the two eyes and compute the center of each eye. The centroid or center of mass as well as the angle between the eye centroids. This angle serves as the key component for aligning the image. Given the eye centers it is possible to compute differences in x,y coordinates and take the arc-tangent to obtain angle of rotation between eyes determined by finding difference between right and left eyecenter. After that, it is computed the angle in the y and the angle in the x directions and the desired eyes angle are substracted to the actual ones. The scale of the image is obtained by taking the ratio of the distance between the eyes in the current image and the distance between the eyes in the desired image. Having the rotation angle and the scale of the face it is needed to compute the midpoint between the eyes (that has to be the top of the nose) and a rotation matrix that will be used to apply affine transformations to the images. So, the parameters that will get a rotation 2d function will be:

- The center of the eyes where the rotation will occur
- The angle of rotation for obtaining a horizontal line between the eyes
- The percentage of scalation of the image for having the desired size (up or down)

The warp transformations will be made using the CV2 C++ library. It is needed to pass to the warp function the translation, rotation and scaling matrix, the desired width and height of the output face and the interpolation algorithm used for the warp. An example can be seen in figure 7.2

Finally, the face will be scaled to the desired size (depending on the input layer of the network) and cropped always after obtaining the aligned face and not before. For the purpose of training and validating a convolutional neural network is is not recommendable to distract the network with other elements that did not contribute to the necessary purpose so it is necessary to adjust to the minimal features necessary to obtain the human face. The previously commented C++ library Dlib can crop (in a parameterized manner) the image on this case it is needed to just keep a 25% of the bounding box image.

7.3 Dataset normalization

Normalization in deeplearning consists in subtracting the mean of the whole dataset images before feating the network. The mean image of the built dataset can be seen in the figure 8.2. Subtracting the mean is a particular technique of data normalization and it helps the learning process or finding the global minimum of the cost function. There are many factors that affects the learning process and a crucial one is how well our data was pre-processed. The better it is pre-processed, the more likely our model will learn faster and better. The goal of computing the mean image is to make our data have zero mean. In images, this normalization has the property of removing the average brightness (intensity) of the data point. In many cases, we are not interested in the illumination conditions of the image, but more so in the content removing the average pixel value per data point makes sense here. Mean subtraction

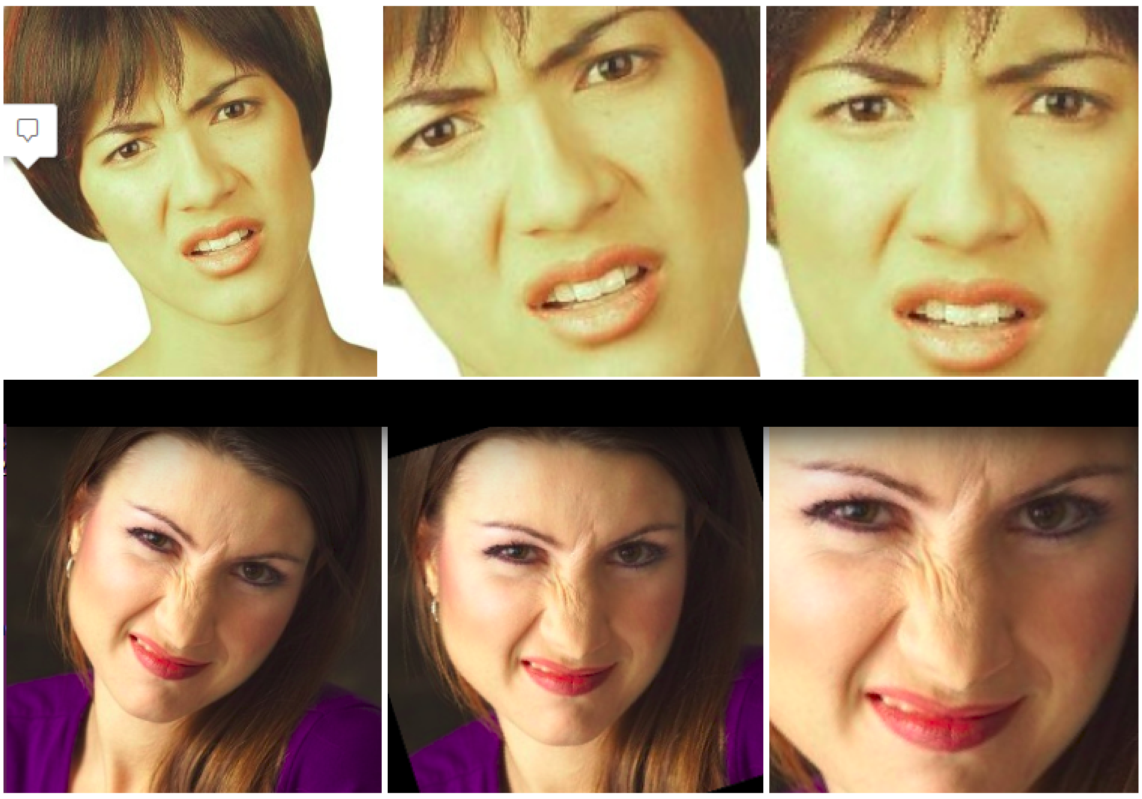


Figure 7.2: Examples of face rotation and cropping finishing in face alignment

is done by each channel. When we are ready to pass an image through our network (whether for training or testing), we subtract the mean from each input channel of the input image the mean Red, Green, and Blue values. This process have been computed channel-wise rather than pixel-wise, resulting in an MxN matrix. In this case, the MxN matrix for each channel is then subtracted from the input image during training/testing.

Chapter 8

Deeplearning: Training 2D convolutional networks

8.1 Fine-tuning CNN

Deep neural networks like Covnets compute a huge number of parameters per layer (often in the range of millions) those parameters get specially concentrated in the last fully connected layers (FC). For an Alexnet network trained in ImageNet with more than 4000 classes we have 37,752,832 learned parameters in the first FC and 16,781,312 learned parameters just in the second FC. In the case of the last FC retrained only with 7 emotions the last FC layer learns 28,679 parameters. Training a Convnet on a small dataset (one that is smaller than the number of parameters) greatly affects the Covnet's ability to generalize, often resulting in overfitting. So, it is necessary to work with existing networks trained on a large dataset like the ImageNet (or better if they were trained with human faces) by continue training it (i.e. running back-propagation) on the smaller dataset that we have. As we will see, this practice works if the new dataset is not drastically different in context to the original dataset. Some learned general guidelines for fine-tuning implementation are:

- Truncate the last FC layer of the pre-trained network and replace it with our new FC layer that is relevant to resolving our classification problem (seven emotions).

- Use a smaller learning rate (LR) to train the network than the LR used for training the original network. Since it is expected that the pre-trained weights will be quite good already or at least better than randomly initialized weights, it is not recommended to distort them too quickly and too much. A common practice is to make the initial learning rate ten times smaller than the one used for scratch training.
- More than smaller LR it is also a common practice to freeze the weights of the first few layers of the pre-trained network. This is because the first few layers capture universal features like curves and edges that are also relevant to our new problem.

8.2 Avoiding overfitting

Deep artificial neural networks require a large corpus of training data in order to effectively learn, the collection of such training data is often expensive and laborious. The problem with small datasets is that models trained with them do not generalize well data from the validation and test set. Hence, having a small dataset (less samples than network parameters) can finish in overfitting, all those methods will be experimented in this work for trying to reduce it:

- Dropout
- Data augmentation
- Transfer learning

Dropout works by probabilistically removing a neuron from designated layers (FC in our case) during training or by dropping certain connections and batch normalization, which normalizes layers and allows us to train the normalization weights. **Transfer learning** a technique in which we take pre-trained weights of a neural net trained on some similar or more comprehensive data and fine tune certain parameters to best solve a more specific problem. In this work it is going to be tested the effectiveness of retrain the parameters of the last fully connected layer versus retraining all the fully connected layers and the last convolutional. **Data augmentation** is another way we can reduce overfitting on models, where we increase the amount of training data. The influence in the final accuracy of augmented data had been tested in this work

8.3 Fine-tuning with Caffe

For the experiments Caffe framework with Nvidia Digits GUI are used. Deeplearning operations are orchestrated by the Caffe Solver. This functionality coordinates the network's forward inference and backward gradients to form parameter updates that attempts to improve the loss function. The responsibilities of learning are divided between the Solver library for overseeing the optimization and generating parameter updates and the Net library for yielding loss and gradients. The standard Caffe solvers algorithms used will be Stochastic Gradient Descent (type: "SGD"). The Solver library creates the training network for learning and test the network for evaluation by iteratively calling forward / backward and updating the parameters. It evaluates the tested networks with the validation input set and it can also snapshot the model and solver state throughout the optimization for testing purposes or in case of shutdowns. In each iteration, it forwardly computes the network output and recomputes network gradients (layers weights) backward to incorporate the gradients into parameter updates according to the solver method (usually backpropagation) and according to a learning rate (configurable) and modifiable in execution time.

Stochastic Gradient Descent The chosen learning algorithm for training the networks in this work has been **Stochastic Gradient Descent** which updates the weights by a linear combination of the negative gradient and the previous weight update. The learning rate (LR) is the weight of the negative gradient. The momentum is the weight of the previous update. The rules of thumb for setting the learning rate and momentum with SGD is to initialize the learning rate to a value around 0.01 and 0.001 and dropping it by a constant factor throughout training when the loss begins to reach an apparent plateau, repeating this several times. Another parameter to choose is momentum that smoothes the weights updates across iterations, momentum tends to make deep learning with SGD both stabler and faster it is recommended a 0,9 value as it was the strategy used by Krizhevsky in the winning CNN entry to the ILSVRC-2012 competition. The final learning rate policy choose is to drop the learning rate in steps by a factor of gamma every stepsize iterations as it will be seen in the experiments section it increments the accuracy. After each iteration the weight update is made by the solver then applied to the net parameters incorporating any weight decay into the weight gradients (which currently just contain the error gradients) to get the final gradient with respect to each network weight. Then, those gradients are scaled by the learning rate.

8.4 ConvNets: pre-trained models

Choosing Caffe as a deeplearning framework gets access to the Caffe Model Zoo where lots of researchers and engineers have made Caffe models for different tasks with all kinds of architectures and distributed throught Github Gists and packaged in a standard format and released for unrestricted use, since none of the original images are distributed in whole or in part. A Caffe model contains information about the architecture of the model in prototxt format and information of the hyperparametrization in the solver.prototxt. It is interesting to pay attention to Caffe version that was used to train the model as sometimes layers implementations and it's parametrization can change between different version. From the model Zoo and in chronological paper publication order the following architectures were chosen:

8.4.1 Alexnet

The Alex Krizhevsky [KSH12] modifications of LeNet with the added insight of stacking multiple convolution layers before pooling and activation layers. Those modifications leads in 2012 to a network having an unprecedented size and trained in GPUs at an unprecedented speed (six days). Another achievement of this network was adding dropouts during training which reduced overfitting and also using data augmentation (rotations, translations and color variations) which increased robustness. Alex Krizhevsky finally introduced the use of ReLU for the nonlinearity functions decreasing training times as ReLus are several times faster than Tanh functions. Due to this particular advances (depth, GPUs and dropout) the paper is widely regarded as one of the most influential publications in the field and it was the winner of the 2012 ILSVRC.

8.4.2 Alexnet weights

In this case the Caffe model used was the one published in 2015 with the name [FSL15] Multi-view Face Detection Using Deep Convolutional Neural Networks. This work proposed a Deep Dense Face Detector (DDFD), a method that does not require face landmark annotation and is able to detect faces in a wide range of orientations using a single model based on deep convolutional neural networks. It successfully solved a face recognition problem

8.4.3 VGG16

Built by Karen Simonyan and Andrew Zisserman in the Visual Geometry Group [SZ14] from the Department of Engineering Science, University of Oxford it was their contribution to the study of how increasing depth of the Convolutional networks affects their performance. They achieve it by reducing the filters to 3x3 the minimum size possible (in Alexnet the filter were 11x11) with stride and pad of 1, along with 2x2 maxpooling layers with stride 2. It was a success in 2014 securing the first and second places in ILSVRC-2014 competition. Furthermore they improve the models and finally release two models of 16 and 19 layers using the framework Caffe. VGG16 is very popular in large-scale image (224 pixels) and video recognition as it achieved better accuracy addressing an important feature of ConvNets the depth. Another important achievement (apart of becoming the state of the art in 2015) was that this network is successfully applicable to other image recognition datasets because it has excellent generalization capabilities and works well in both tasks: classification and localization. The major drawbacks with VGG16 is that, due to its depth and number of fully-connected nodes, the trained network is over 533MB making the deployment a more computationally demanding task and impossible to integrate in low memory devices like mobile phones. A graphic explanation of the architecture can be seen in the figure 8.1

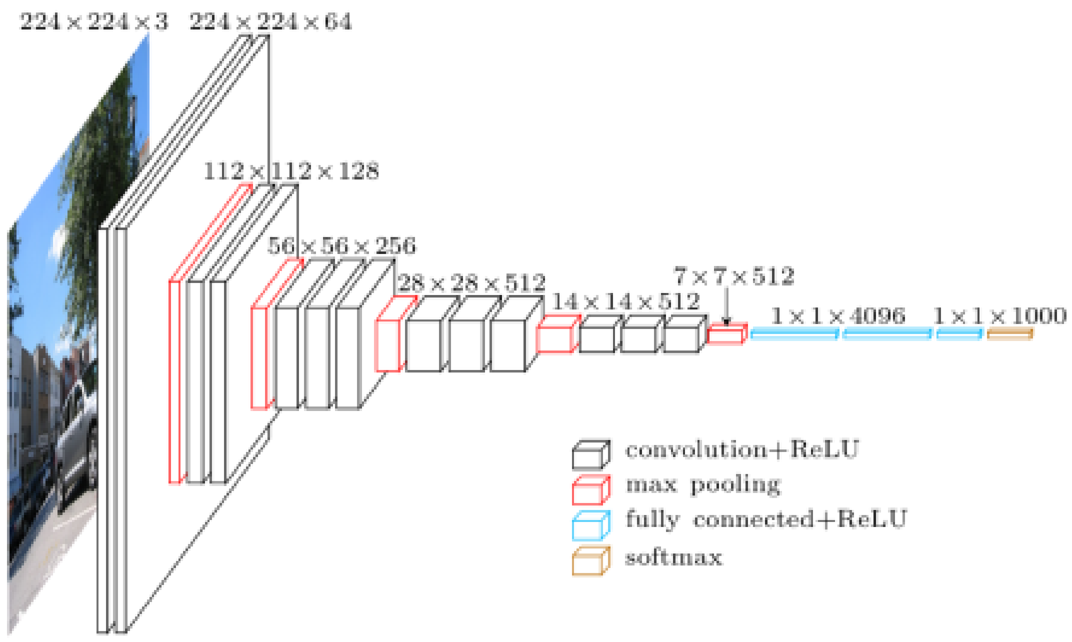


Figure 8.1: VGG16 architecture image by Davi Frossard.

8.4.4 VGG16 weights

The Computer vision center of Barcelona fine tuned a VGG16 network (trained in Imagenet) with 8000 face images using Caffe framework for this purpose. Those weights are imported to the VGG16 model.

8.4.5 Microsoft ResNet

Unlike traditional sequential network architectures such as AlexNet and VGG, ResNet is instead a different architecture that relies on micro-architecture modules or networks-in-network. The term micro-architecture refers to the set of building blocks (introduced by GoogleNet Inception modules) used to construct the network with the idea that layers didn't always have to be stacked up sequentially. ResNet won ILSVRC 2015 with an error rate of 3.6% where humans generally hover around a 5-10% error rate. A collection of micro-architecture building blocks leads to the macro-architecture of more than a hundred of layers. First introduced by He et al. in their 2015 paper [HZRS16], Deep Residual Learning for Image Recognition, the ResNet architecture has demonstrated that extremely deep networks can be trained using standard SGD (and a reasonable initialization function) through the use of residual modules and reducing the model size to 102MB for ResNet50. Resnet solved the notorious vanishing gradient problem (the gradient back-propagated to earlier layers may take values infinitively small) making use of shortcut connections where the gradients can flow to any other earlier layer. The authors, demonstrated with experiments that they can now train a 1001-layer deep ResNet to outperform its shallower counterparts

8.4.6 Resnet weights: ResFace101

ResFace101 is a publicly available ResNet network trained with human faces and fined-tuned on CASIA images following the augmentation process from the paper by I. Masi "Do We Really Need to Collect Million of Faces for Effective Face Recognition?" [MTH$^+$16] a face recognition project in 3d head data augmentation and pose estimation face rendering.

8.4.7 ImageNet dataset

Most of this Caffe Zoo models have been pre-trained on the ImageNet dataset. In the context of deep learning it is used the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) for comparing networks performances and they provide a reduced version of the ImageNet project. For the purpose of this image classification challenge there is a subset of ImageNet of 1000 object categories and the models avilable from the Caffe Zoo repository have been trained on 1.2 million training images with another 50000 images for validation and 100000 images for testing. These 1000 image categories represent object classes that we encounter in our day-to-day lives, such as species of dogs, cats, various household objects, vehicle types, and much more. In this work networks will be also trained in Imagenet weights for experimenting the influence of weights in fine tuning networks.

8.5 Methodology for training

For purposes of performance comparison the three networks have been created using the same procedures and trained and tested using the same subsets of the previously created dataset. A representative subset of 15% of the total images was created and isolated from the rest of the infrastructure for testing purposes and the rest of the set is used for training using a 10% part for validation. The same image size 224x224 is used for input layers and the same mean image is subtracted for the three models. The mean image can be seen in the figure 8.2.Using the downloaded pre-trained models weights, the following steps are used for building a new model:

- A Network is fine-tuned with the training subset using the weights of downloaded trained network. The weights are transfered to all the layers except the last convolutional layer and all the subsequent fully connected layers that are retrained in the case of Alexnet and VGG and only the last FC in the case of Resnet. The last FC layer will output just the 7 classes that we have.
- The same network architecture is fine-tuned with the augmented subset using the same previously downloaded weights.
- The networks are retrained , once again with the augmented dataset, but this time the weights used will be from networks trained with Imagenet dataset (not faces).
- The inference of the trained networks is computed with the isolated test set.

8.6 Computing resources

As it was mentioned the Caffe framework has been choose with the software Nvidia DIGITS as a GUI user interface. The software installations have been done using Nvidia Cloud Dockers containers that contains all the Digits, Caffe, CUDA and CUdnn software already packed. It is needed to have a host machine with the GPU Driver and a Docker plugin called nvidia-docker for launching the containers. Using Dockers allowed to work with three different GPU

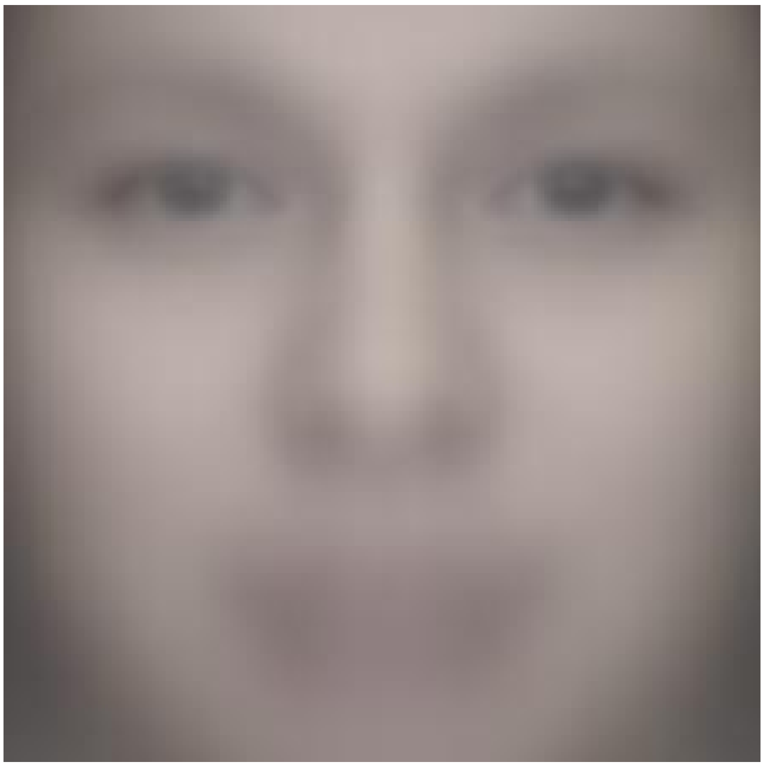


Figure 8.2: Mean image of the built dataset

computers without compromising software versions and paths. Three different machines have been used:

- A personal laptop with geForce 950M GPU for modeling learning and configure
- An Ubuntu virtual machine host with GPU Tesla K80 machine in Amazon Web Services cloud providers.
- An Ubuntu host with GPU Tesla P100 machine in Google Cloud services.

Using spot or preemptible instances can lower the costs enormously. The time differences training with two different GPUs from Nvidia -Tesla K80 and Tesla P100- are very notable and the Tesla P100 can be 4.5x times faster than others GPUs as it can be seen in the table 8.1. But, it clearly increases the financial costs being -in April 2018- the cost per hour of a Tesla k80 0,45$ and 1,60$ for a Tesla P100 in Google Cloud.

Network	Dataset	P100	K80
VGG	Original	0h 7m	30m
Alexnet	Original	0h 41m	3h 10m
Resnet	Original	3h 45m	13h 10m

Table 8.1: Differences training with two different Nvidia Tesla GPUs

Cloud provider	Instance type	Ubuntu	Nvidia driver
Google Cloud	n1-standard-4 (4 vCPUs, 15 GB)	16.04 xenial	384.111
AWS	p2.xlarge 4 vCPUs, 61 GB	16.04 xenial	384.111

Table 8.2: Virtual machines used for training

DIGITS	Digits docker version	CuDNN	CUDA	Caffe
6.1.0	18.02	7.11	9.0.176	0.16

Table 8.3: Software used for training

8.7 Software resources

Using Dockers containers allowed to sue the same software versions in different environments. It has been deployed the same software infrastructure (that can be seen in table 8.3 in three machines types that can be seen in table 8.2. Mainly, it has been used:

- Two cloud providers: Amazon Web Services and Google Cloud.
- A laptop with GPU.
- Ubuntu 16.
- Two virtual machines with 4vCPUs.
- Two Nvidia Tesla GPUs: k80 and p100.
- Dockers containers.
- Nvidia Docker version with CUDA, CuDnn, Caffe and Digits.

8.8 Fine-tuning details

In the case of VGG16 and Alexnet the last Conv layer will be retrained and the rest of the FC will also be recomputed. Fine-tuning is performed by minimizing the soft-max loss optimized using Stochastic Gradient Descent (SGD) with standard L2 norm over the learned weights. The non frozen layers were initialized with parameters drawn from a Gaussian distribution with zero mean and standard deviation 0.01. Bias is initialized with zero. On the case of VGG16 (just because it has been the first network used for this work) 15 successful experiments have been done for learning by doing the principle details of fine-tuning CNN networks that are:

- The importance of different learning rates finding 0,001 the better initial one
- The importance of initial weights trying to use those that were computed from similar datasets
- The importance of the size of the dataset as VGG16 has been trained using different size sets (while it was collected the whole data)
- The differences between learning just the last FC layer or recompute the last Conv layer being better to retrain the last Conv layer.

The most critical hyperparameter for an stable learning has been the learning rate. The overall learning rate for the entire CNN is set to 0.001 and the network converges quickly from the first epochs. It can be seen in the figure 8.4. When the validation accuracy remains steady and the network seems not to learn anymore downgrading the learning rate to 0,0001 increase the accuracy. So, we decrease learning rate by an order of magnitude once validation accuracy for the fine tuned network saturates. The learning curves can be seen in the figure 8.3. In general for all the other parameter settings we use the same values as originally are used in the original paper of the authors of the network. In the case of the three networks the same hyperparameters and methodologies have been used training all layers for 30 epochs with SGD with the parameters showed in table 8.4· Those networks converge similarly from the first epochs and the differences are in time training been Alexnet much more faster. 11 minutes in

Hyperparameter	Value
Initial LR	0.001
LR policy	step
Gamma	0.1
Momentum	0.9
Stepsize	33%
Solver type	SGD
Batch size	60

Table 8.4: Same parameters values used for training the three networks

Network	Time training	Iterations for 30 epochs	Learned parameters	Solver perfomance
Alexnet	7 mins	6570	56,896,903	1681 img/sec
VGG16	41 mins	27960	134,289,223	356.6 img/sec
Resnet	3h 45 mins	41940	42,619,847	68.68 img/sec
Alexnet Aug	11 mins	8400	56,896,903	1681 img/sec
VGG16 Aug	1 h 32 mins	33552	134,289,223	356.6 img/sec
Resnet Aug	4h 45 mins	53640	42,619,847	68.68 img/sec

Table 8.5: Comparison of the models time training with original and augmented dataset in a Tesla P100 GPU of 12 Gigas.

the augmented dataset for an hour training in the case of VGG16. This is clearly due to a much more deep architecture that needs more computations and memory per layers. As it can be seen in the table 8.5 the memory of the GPU (Tesla P100 has 12 gigas) can process much less samples in each second of training. The solver function of the Caffe framework can process 1681 images per second in a Alexnet architecture and 68 in a Resnet101.

8.9 Image inference

Obtaining a sentiment from the categorical description model using the human face is a computer vision and machine learning classification problem. Based on the use case of this work a model that will analyze the sentiments of a car driver we can have:

- Inference in a mobile phone
- Inference in portable computer like a RaspberryPI or a laptop computer
- Inference done in a REST web service

For a mobile phone the size is very important as it will be downloaded by the user and mobiles have memory limitations. In this case, there are some optimization tricks like layers reduction, convolutional layers cumulatively contain about 90-95% of the computation but only about 5% of the parameters, and have large representations. Fully-connected layers contains about 5-10% of the computation but about 95% of the parameters, and have small representations. Caffe is a framework chosen for its speed in inference but recently Google has been making great achievements with it's TensorFlow framework specially in Android mobile phones.

Using Caffe framework a python code has been built for having an inception model in a REST web service. Two tests had been done for computing the time differences between using GPU or not. In a laptop computer with the simple Nvidia 950M GPU and testing the inference with 742 images the same Caffe code had the following results:

- With GPU real time was 1 minute and 6 seconds
- With CPU real time was 20 minutes

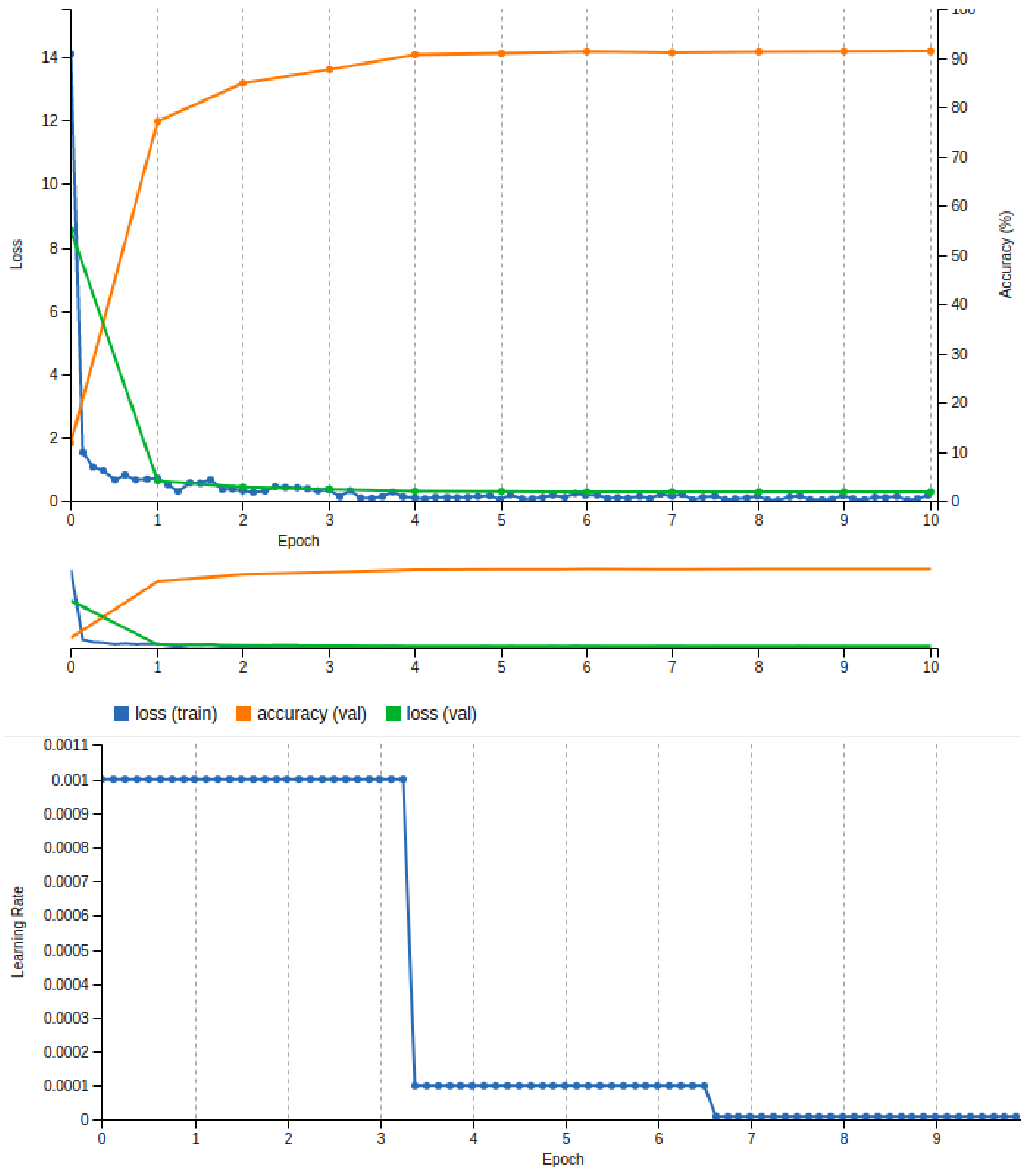


Figure 8.3: VGG16 fine-tuning and LR decreasing.

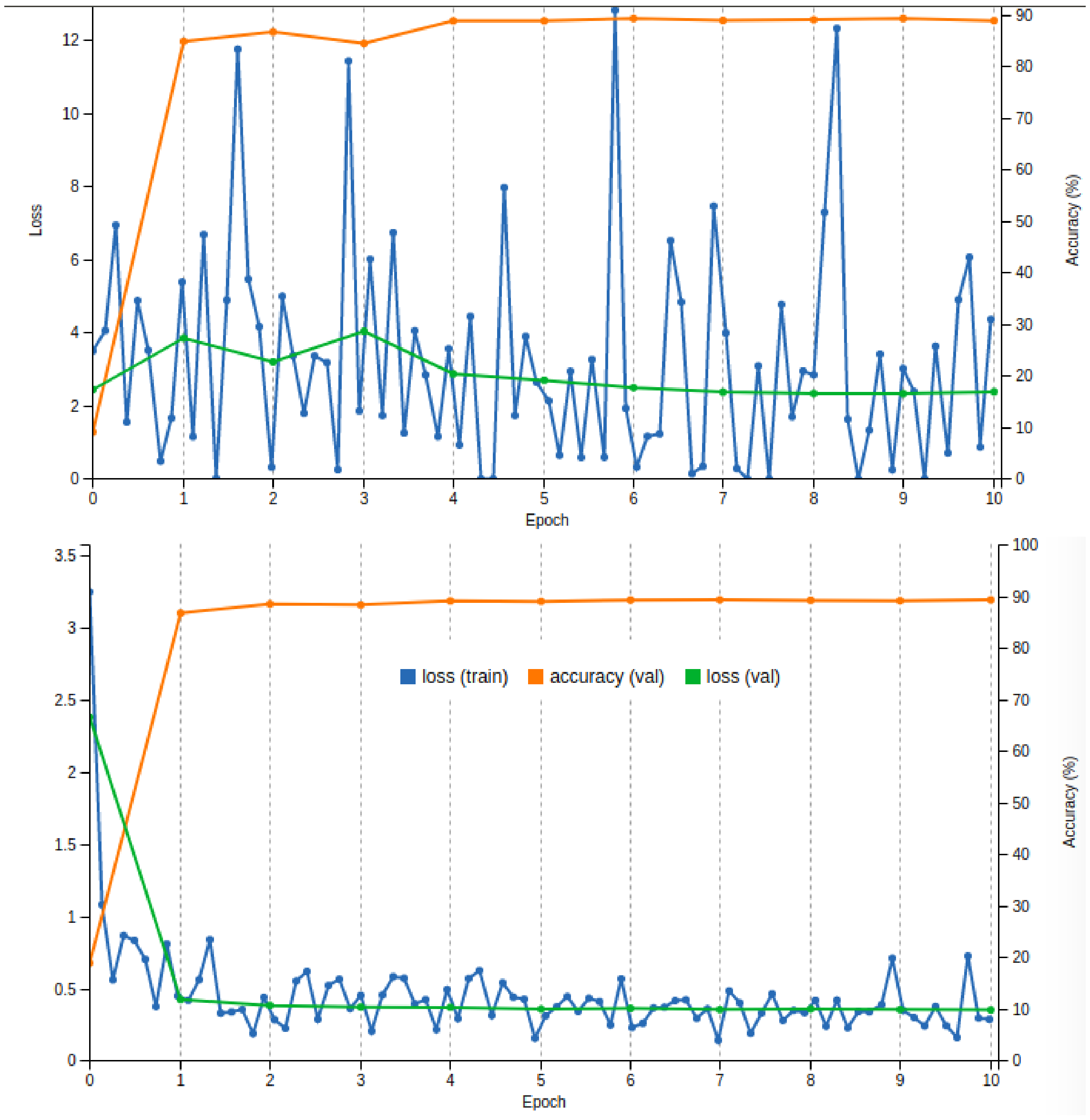


Figure 8.4: Loss curve stability differences between different LR (0,01 vs 0.001 in the bottom)

Chapter 9

Experimental methodology

With the objective of testing different deeplearning models for solving our face analysis problem diverse experiments were performed to assess the accuracy of the different algorithms. It has also been measured their execution and training time for exploring their different advantages. Firstly, it is going to be explained the data distribution for training and testing. Secondly, the parametrization of the algorithms is going to be explored and the final parameters are going to be explained. Finally, a qualitative analysis is going to be performed where overfitting is going to be deeply explored.

9.1 Experimental settings

As it has already been explained three ConvNet architectures were chosen: VGG16, Resnet101 and Alexnet. Every network is going to be trained and tested three times:

- Trained with the build dataset
- Trained with the augmented dataset
- Trained with the augmented dataset and Imagenet weights

All the networks have been trained in the same computing resources: n1-standard-8 instance from the google cloud: 8vCPUs and 30 GB memory. The instances have an Nvidia Tesla P100 attached and the Caffe frameworks and Digits software is installed using Docker container provided by Nvidia Cloud services.

9.2 Data distribution

9.2.1 Original dataset

The final dataset has 37848 images A 10% of the training set is going to be used for validation and a set with 15% of this images is going to be isolated for testing purposes.

9.2.2 Augmented dataset

The augmented dataset built from the final dataset (minus the 15% test partition) has 35741 images with a more balanced classes proportion. A 10% of the training set is going to be used for validation

9.2.3 Test set

Test set respects the classes distribution of the original dataset (unbalanced dataset) and it is isolated and never used for training neither in the augmented training. So, none of the six trained networks have never seen the images that will be tested. As it is a subset of the original dataset the images where preprocessed and proceed from the same datasets collected. The final numbers can be seen in the table 9.1

Set	% used	Images
Original training		28954
Original validation	10%	3217
Augmented training		35747
Augmented validation	10%	3972
Test	15%	5677

Table 9.1: Final number of the dataset distributions.

9.2.4 Overfitting Test set

Based in the Face Place that was created by the Tarrlab at Brown University, Carnegie Mellon it includes multiple images for over 200 individuals and some of them disguised (some participants returned for a second session several weeks later with a haircut, or a new beard) thus, more variety intra-classes. This data images weren't included in the training dataset because it was mostly neutral poses. Some examples were collected from google images with different emotions, hand labeled and added to the set. Some problems recognizing and tagging some expressions make labeling a difficult task. Some samples can be seen in the figure 9.1

The new dataset has the following characteristics:

- 742 images.
- Unbalanced with 544 neutral poses and the rest distributed in 6 classes.
- More racial diversity specially black people.

9.3 Evaluation metrics

The computed metrics are:

- Softmax with Loss computed with the multinomial logistic loss of the softmax of its inputs.It's conceptually identical to a softmax layer followed by a multinomial logistic loss layer, but provides a more numerically stable gradient providing the probability distribution.

- Accuracy computed counting as correct by comparing the true label to the top scoring classes (argmax)
- Computing time
- Model size

9.4 Method details

9.4.1 Parameters

A set of common parameters have been choose as the more adequate after some experiments, specially remarkable is the learning rate (LR) parameter as it is crucial for the learning speed and the final accuracy. Learning rate have been set to 0.001 as it is the value that performs a more stable way to learn and the network learns faster and steady.

AlexNet For a Nvidia Tesla 100 with 16GB of memory the maximum batch size as been 60 image The time training has been 11 minutes for the augmented that means 8400 iterations and this architecture achieved around 95% accuracy in the test set.
Vgg16 For a Nvidia Tesla 100 the maximum batch size as been 40 images and it tooks an hour for training the last convolutional and the FC layers during 23840 iterations. It achieved the lowest accuracy of the experiments with 91% for the test set.
Resnet For a Nvidia Tesla 100 with 16GB of memory the maximum batch size achievable as been 20 images. The network took 4 hours 45 minutes to retrain the last FC layer in 53640 iterations but it achieved 97,17% accuracy.

9.5 Analysis

The accuracy computed for five of the six experiments were greater than 95% each one of the model can be usable for a production system. The results for the six experiments and their

Model	Dataset	Validation Acc	Time training	Model size	Test Acc
ALEXNET					
	Original	95.24	00h07	228MB	95.48
	Augmented	96.025	00h11	228MB	95.91
RESNET					
	Original	97.211	03h45	171MB	97.17
	Augmented	97.315	04h45	171MB	96.88
VGG					
	Original	91.73	00h41	537MB	93.97
	Augmented	93.29	01h02	537MB	95.84

Table 9.2: Results of the six experiments

trade-off can be seen in the table 9.2 we can compare size of the models and time training that can be take into account depending of the inference needs. Resnet has the great accuracy without the needs of training it with an augmented dataset but the model is very slow in training and this can make him unusable if constant retraining is needed.

From the point of view of the test set the three networks have an excellence performance more than 0.90 accuracy is more than an acceptable result. Apart from performance if we include the trade off of training and inference and taking into account *the Parsimony principle* widely known as Occam's razor. It tells us to choose the simplest scientific explanation that fits the evidence **Alexnet became the chosen model**. It trains at great speed due to its lower complexity and an acceptable size for lots of devices so it can be implemented in many hardware devices and retraining the model with new data can be done in a very acceptable time. Testing time inference in the three models will be a definitive argument for the less complex model: Alexnet.

The three models have been tested with the build code for deploying a REST webservice and 742 images and had the following 9.3 results (in a Tesla p100 machine). In the table it can be seen the speed of each model but also the differences between GPU and CPU. Resnet is the more precise model but the difference of time for training and make inference makes Alexnet unbeatable. At this point let's move to explore one of the most well known dangers existing in fine tuning networks with not a great quantity of data: overfitting. In principle, in this work

Network	P100	CPU
VGG	0m14s	7m40s
Alexnet	0m7s	5m49s
Resnet	1m14s	12m04s

Table 9.3: Inference of 742 images time comparison using NvidiaP100 GPU or CPU

all the recommendations for not having overfitting had been applied:

- Dropout between the FC layers.
- Data augmentation.
- Transfer learning.
- Isolated Test Set.

In the next chapter, we explore more in depth the issue of overfitting by carrying out new experiments.

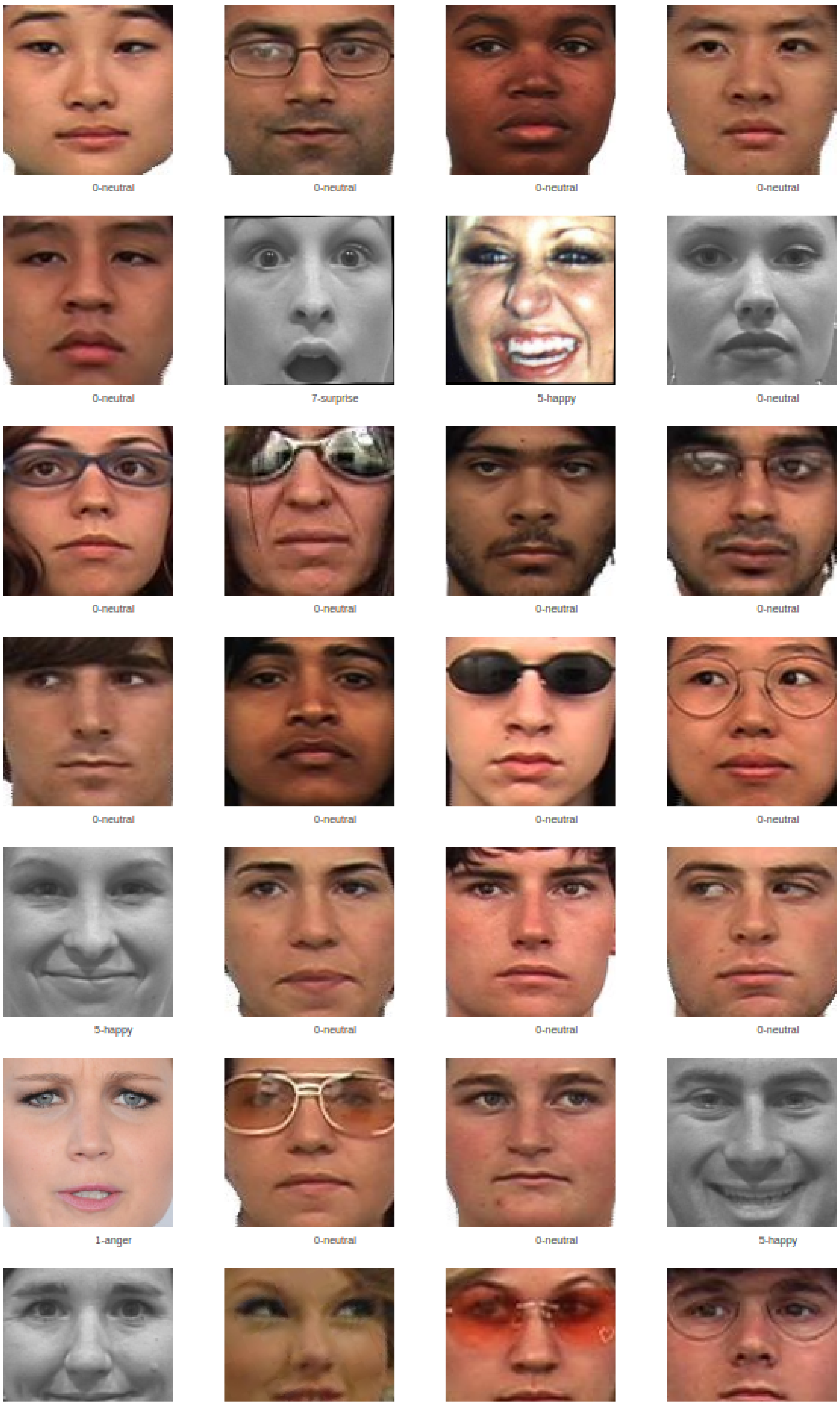


Figure 9.1: Some of the variety examples of the new dataset that were selected to trick the models

Chapter 10

Dealing with overfitting

10.1 New experiments

A simple definition for overfitting is the capacity of a model to generalize outside of the training set. Despite the regularization techniques and successful results deep artificial neural networks can exhibit a remarkably lack of generalization as it will be shown. Our models, exhibit no remarkably signs of small generalization error but complex models like Resnet and small datasets like ours (less samples than network parameters) can easily come up with natural model architectures that generalize poorly. For exploring this it has been decided to retrain the networks with weights not trained with faces but trained with ImageNet. In the following table it will be shown experiments done with the same methodology but the imported weights are different. In addition a new dataset was created to complement this experiments, the *Overfitting test set*. In this case, this dataset was a less meticulous work and the thoroughness was put in search human faces that could cheat our models. This new dataset has human faces that were not present in the training set that was mostly composed of Caucasian actors and scientists (mostly Asian) some of them wearing sunglasses or long beards.

So, for experimenting with the possible consequences of overfitting new experiments have been proposed:

Accuracy	Validation	Test Set	Overfitting Set
ALEXNET			
Face	0.95	0.96	0.79
ImageNet	0.96	0.96	0.40
VGG			
Face	0.93	0.96	0.80
ImageNet	0.92	0.95	0.50
RESNET			
Face	0.97	0.97	0.82
ImageNet	0.96	0.97	0.65

Table 10.1: Comparison of the accuracies of the three models using different weights for training

- Fine-tune the networks with ImageNet weights.
- Test networks with images that could cheat the models.

10.2 New test results

The same inference conditions applied in the experiments methodology have been reproduced for testing the new test set but the results are surprisingly different as it can be seen in the table 10.1. This new experiment confirms the danger of overfitting. If networks are trained with ImageNet weights they gave the same results in validation and test accuracies, times training and models sizes but they completely lacks of generalization capacities. The networks are not capable of classifying face expressions that have some subtle differences with the training sets. We are just talking of different skin colors or some occlusions like sunglasses. So, a model trained with ImageNet can give an excellent result in a laboratory but it is not clearly recommendable for a production systems. Thus, training weights remains the most important point in fine-tuning. Having the two models we can compare the the two loss curves being the more stable the curve of the model trained with faces weights. So, loss curves can be a good indicator for an overfitting. The curves can be seen in the figure 10.1.

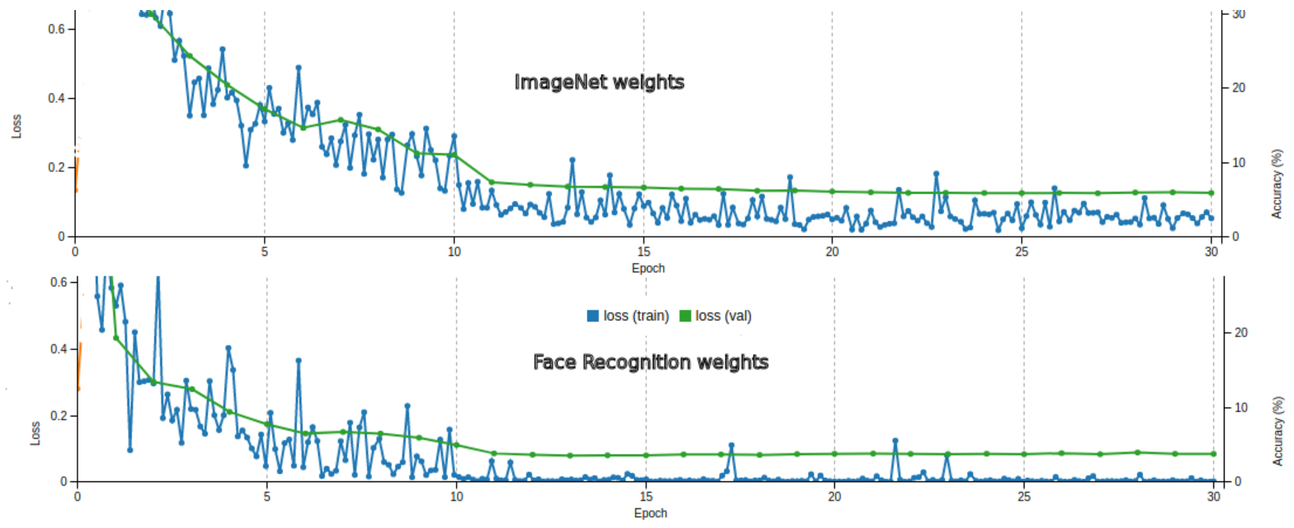


Figure 10.1: Loss curves differences training Alexnet with different weights and same architecture, dataset and LR

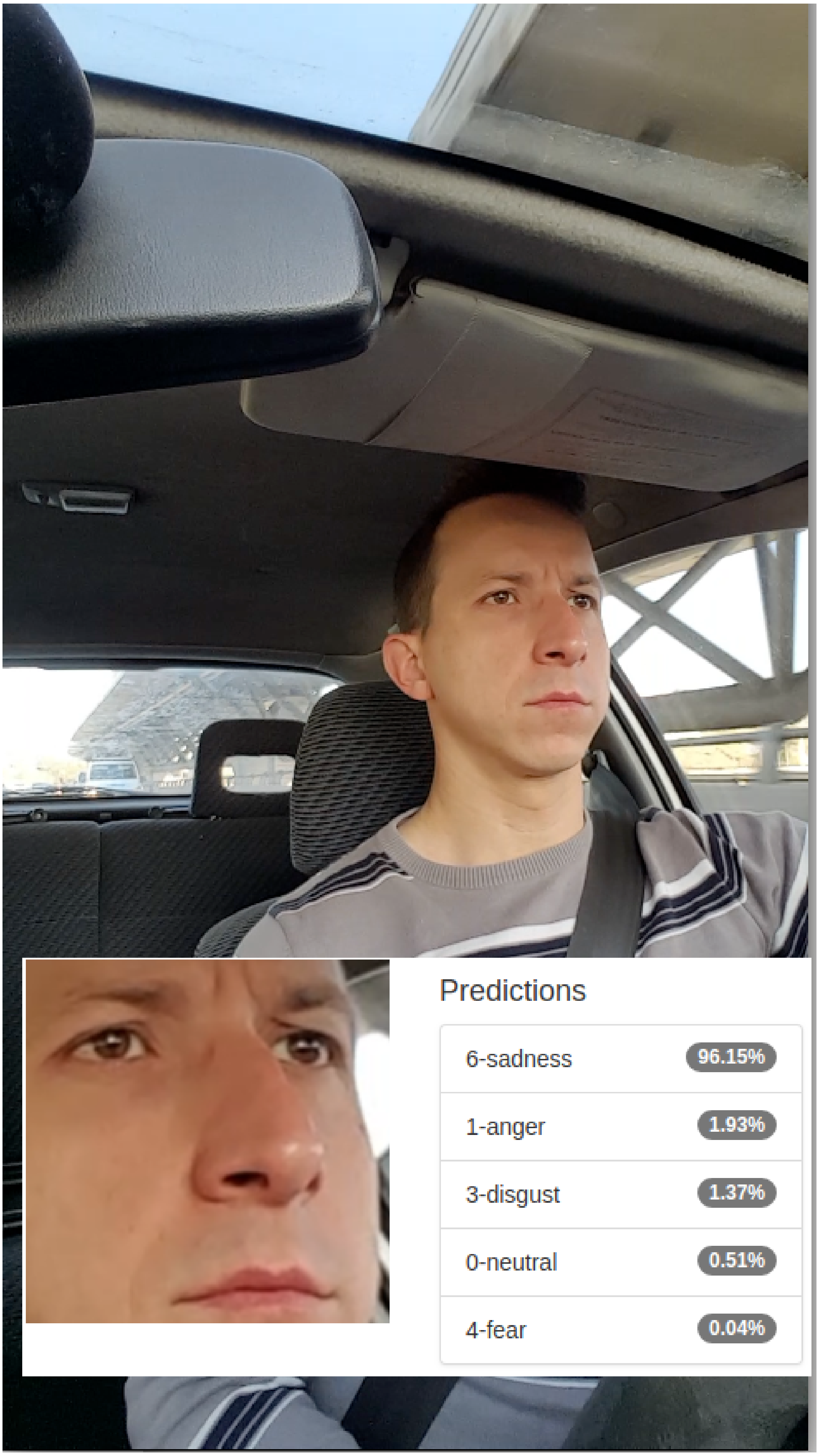
Chapter 11

Testing inference in real drivers

As it has been mentioned this methodology, workflow and model will be used in a company that merges computer vision and cognitive perception to raise awareness and promote good driving practices. Accidents by distraction are a tragedy not only in the human aspect, they also suppose great losses for a professional fleet. They want to prevent accidents and promote good driving habits with AI software. Three different drivers have been recorded in different lighting and environment conditions, the cameras have also been placed in different angles. Those angles were very different than the used for training in the proposed workflow. Some extreme poses cannot be excellently preprocessed because they are very difficult to align the eyers. Despite this, the model has had successful results specially when the conditions where more acceptable. If the company needs more classification of spontaneous poses (on the wild) more tagged data will need to be provided for retraining the networks. In general the model had good results specially because it has to deal with neutral and happy poses and this are very successful classes but, we are going to see some badly classified examples and try to explain why. And also some good results.

In the figures it can be seen the results of class misunderstanding in the figure 11.1 where the pose was tagged as anger but it is tagged as sad. On this case, the human representation of the expression can be discussed. But, as it was already commented in this figure 6.2 there are levels of intensity in the expressions representation. In the case of the image 11.2 the model can not find a winning class as the best result needs more than 50% of confidence in the prediction for being accepted. In the figure 11.3 we have an image with an extreme angle of the camera (there were not training samples like this) and a night scene. The pose was tagged as neutral and the network predicted sad with 72% degree of confidence. Certainly the shadows of the illumination and the beard of the human face can trick the network around the mouth. A similar camera angle (from the bottom) with good illumination conditions also gave a bad prediction as the image in 11.4 was tagged as happy and the network predicted neutral. The angle and the mustache can be the cause of a bad prediction. Because, as it can be seen in figure 11.6 network does not have problems with intense facial representations. It can be objected that the happiness of the face was not very intense. As it can be seen in the image 11.5 a centered angle of the view helps to the network to have very good predictions despite the low intensity of the expressions in the pose.

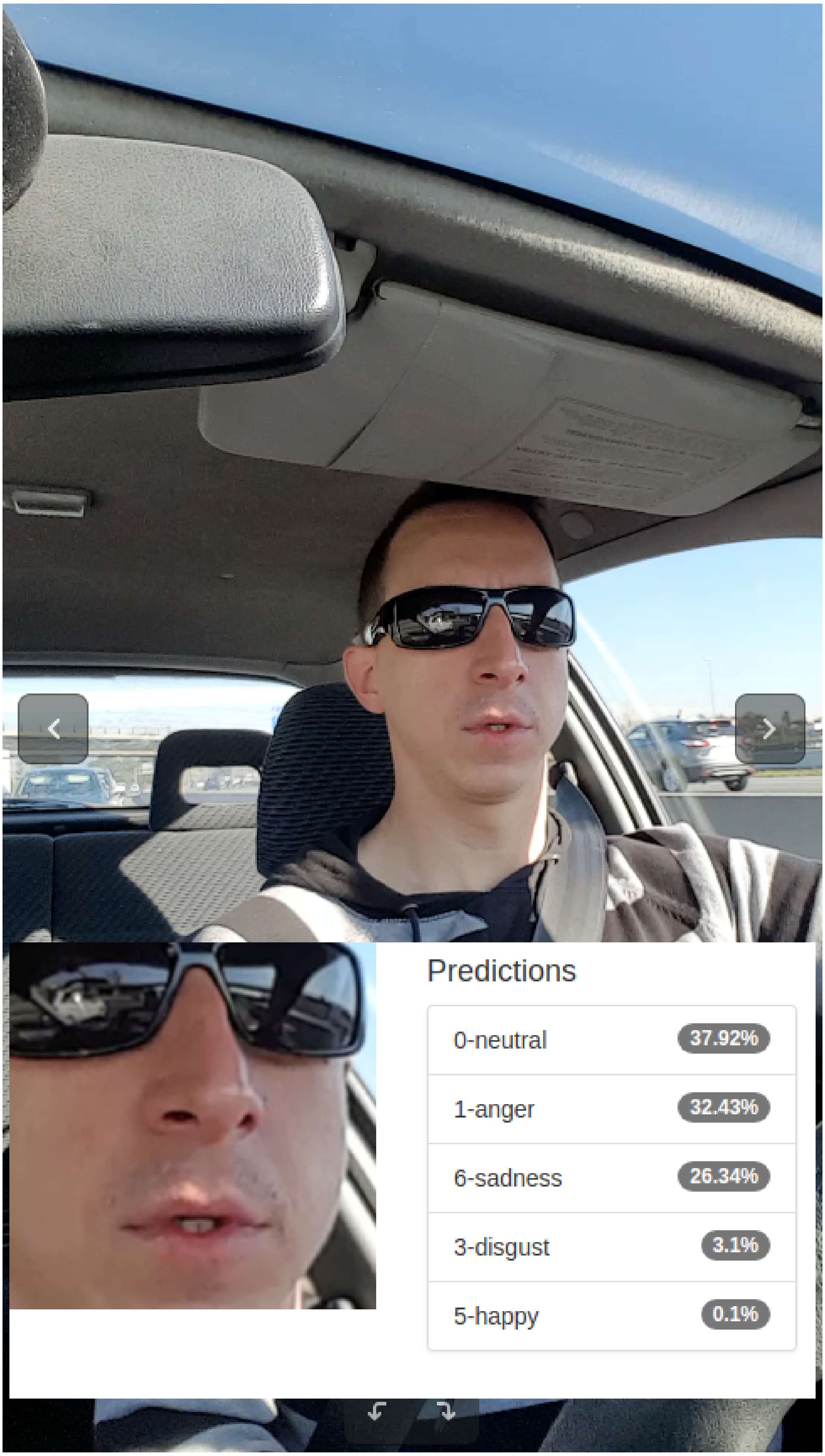
So due to the preprocessing it can not be guaranteed that the network give good results with some extreme poses but the results are surprising as there are some angle of view that the network has never been trained with.



Predictions

6-sadness	96.15%
1-anger	1.93%
3-disgust	1.37%
0-neutral	0.51%
4-fear	0.04%

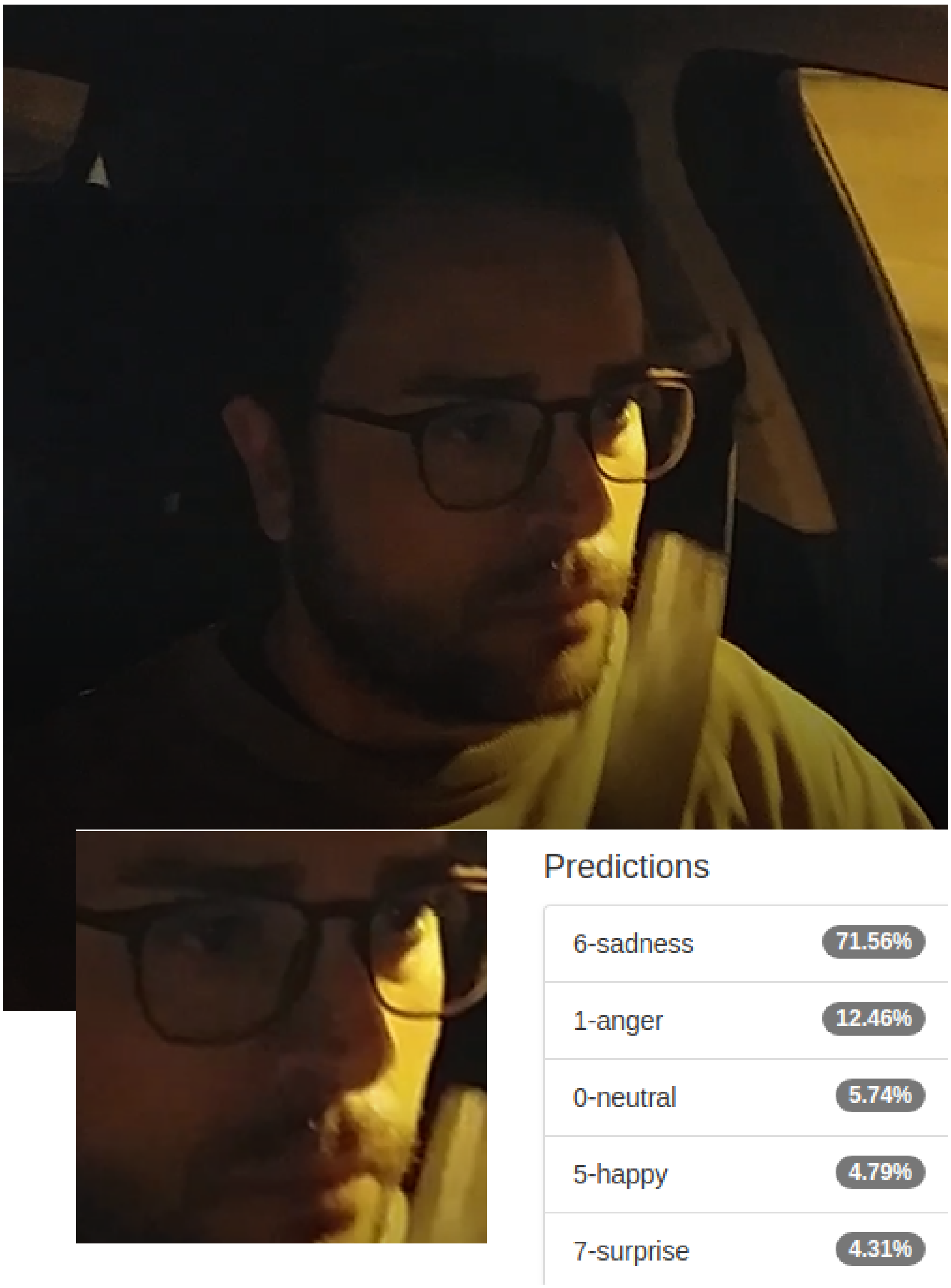
Figure 11.1: Bad classification: the pose was tagged as anger but classified as sad



Predictions

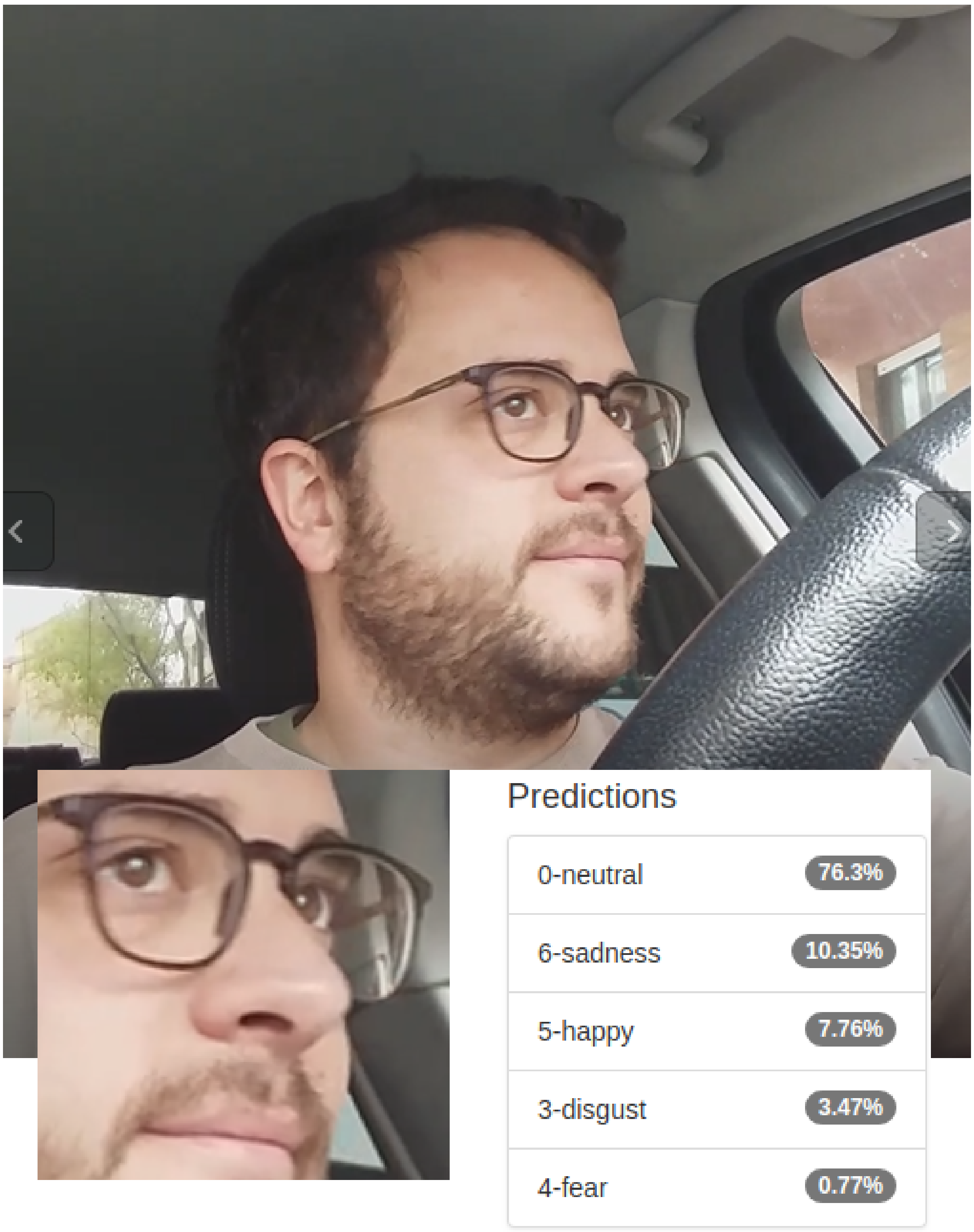
0-neutral	37.92%
1-anger	32.43%
6-sadness	26.34%
3-disgust	3.1%
5-happy	0.1%

Figure 11.2: No winning class in this predictions


Predictions

6-sadness	71.56%
1-anger	12.46%
0-neutral	5.74%
5-happy	4.79%
7-surprise	4.31%

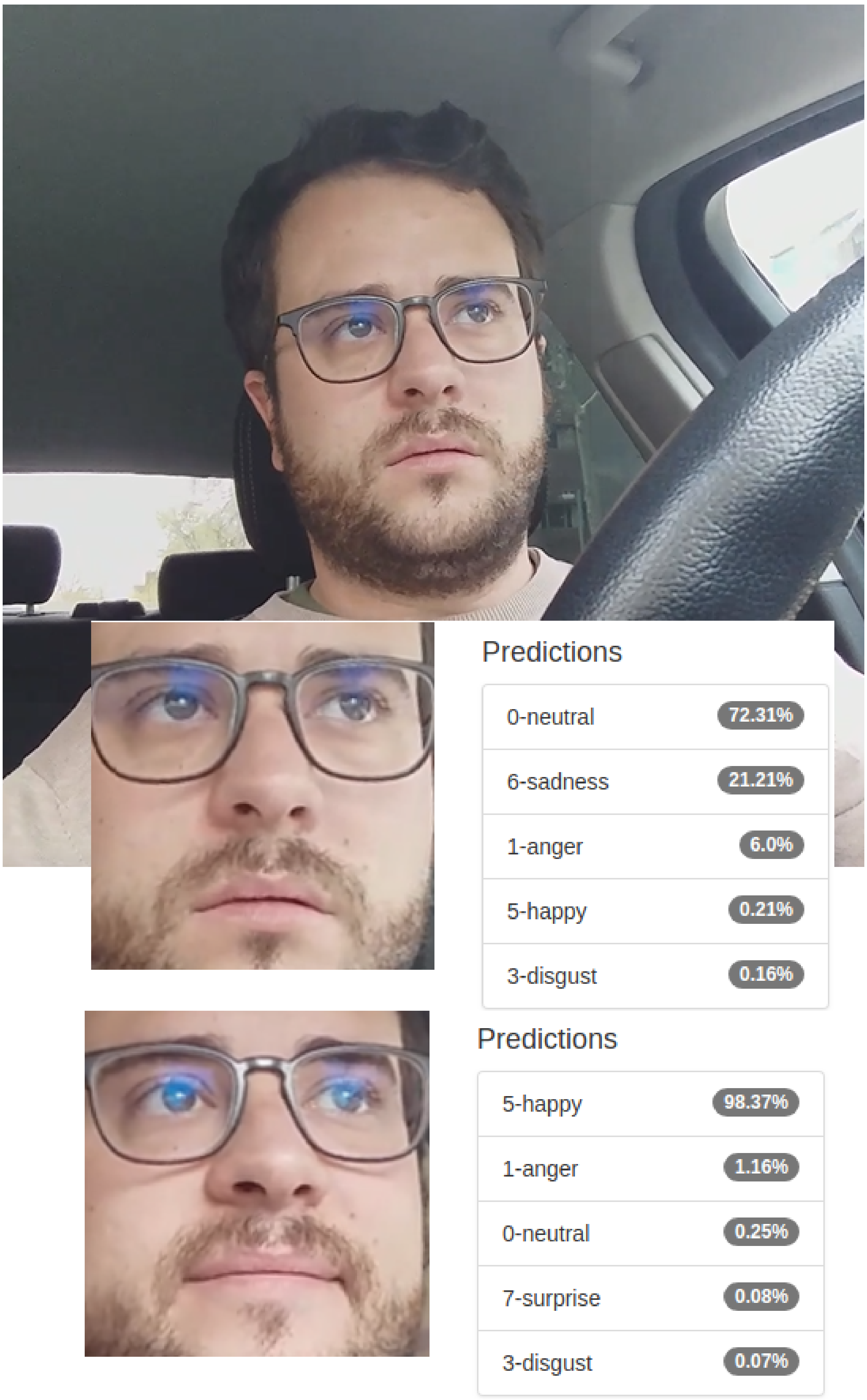
Figure 11.3: Extreme angle. Image was tagged as neutral



Predictions

0-neutral	76.3%
6-sadness	10.35%
5-happy	7.76%
3-disgust	3.47%
4-fear	0.77%

Figure 11.4: Image tagged as happy.



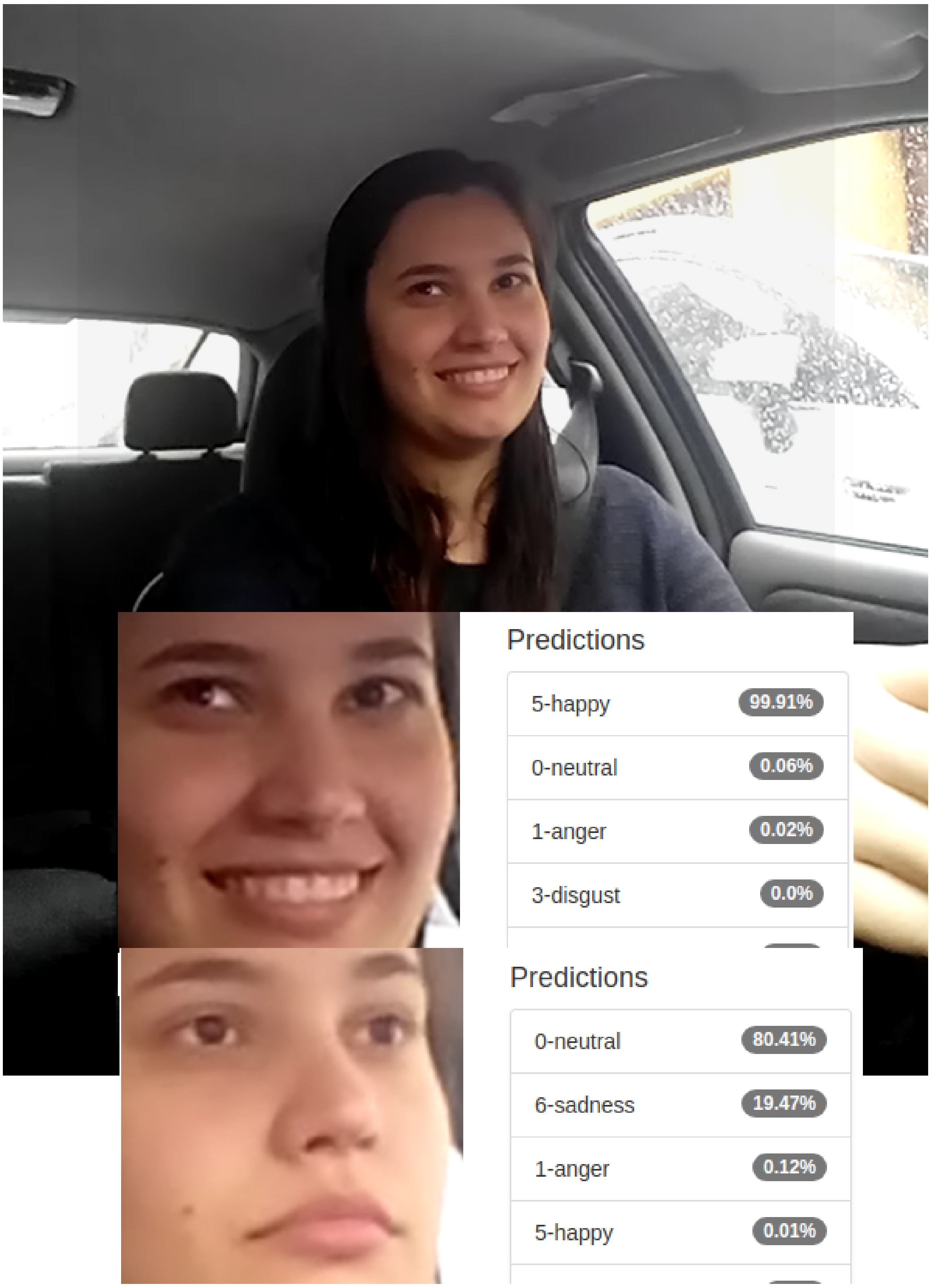
Predictions

0-neutral	72.31%
6-sadness	21.21%
1-anger	6.0%
5-happy	0.21%
3-disgust	0.16%

Predictions

5-happy	98.37%
1-anger	1.16%
0-neutral	0.25%
7-surprise	0.08%
3-disgust	0.07%

Figure 11.5: Two good predictions.



Predictions

5-happy	99.91%
0-neutral	0.06%
1-anger	0.02%
3-disgust	0.0%

Predictions

0-neutral	80.41%
6-sadness	19.47%
1-anger	0.12%
5-happy	0.01%

Figure 11.6: Good predictions: happy and neutral classes were tagged

Chapter 12

Conclusion

12.1 Summary of Thesis Achievements

This work makes several appointments to a methodology that can be used in a variety of computer vision use cases different than face analysis. The workflow only has a specificity related to faces (in the preprocessing of face images with HOG detector) but in the rest of the workflows it can be retrained an reused in a variety of object classification cases. The work goes from zero, building a dataset and training a variety of networks to and inferring model for image classification. It provides reference timings in training and inference. And helps for choosing hardware architectures adaptable to different necessities like inference in portable computers (CPU) to webservices with GPU servers. The deeplearning models have been developed with software packaged in Dockers and this is definitely a future recommendation as it allows to move between different environments. ConvNet were trained in preembtible resources of cloud computing providers those are virtual machines that costs a much lower price than normal instances because they make use of excess compute engine capacity of the providers (spot instances in case of AWS).

Apart from the provided pipeline, one of the biggest freely face emotions dataset has been built for research purposes. It also has been analyzed the advantages of make data augmentation in this type of datasets. Some recommendations have been proposed for fine-tuning deep learning models specially what it was found as the key points: the learning rate and the imported weights. And a final relevant point has been emphasized. It has been showed how easy it can be to train and deploy a deeplearning model that did not generalize well. For this purpose it has been trained models with Imagenet that had excellent validation and testing accuracy but did not know how to classify images that were choose to trick them. In this case, it was showed the importance of fine-tuning the networks with imported weights from very similar datasets trained networks.

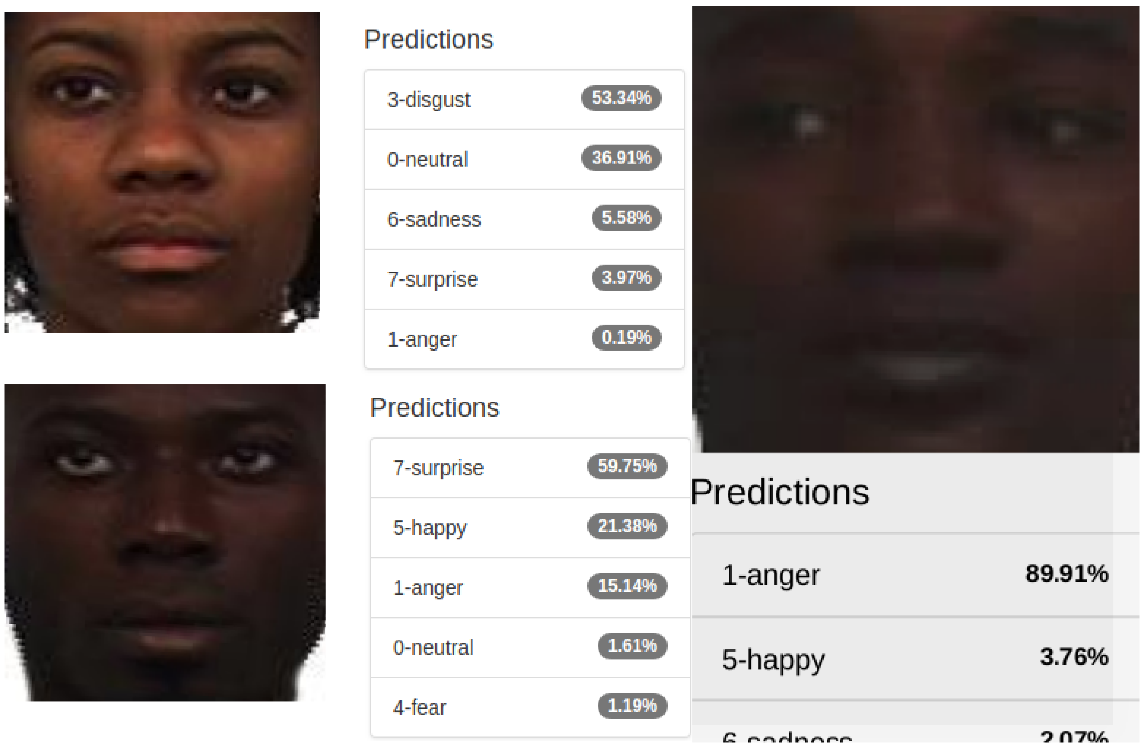
12.2 Biased ConvNets

Testing the inference of the models the networks have been presented with some examples of black race individuals and the disturbing implications of algorithmic bias have arose (see figured 12.1). This machine bias danger is constantly in the news but it's difficult for a researcher to realize it until it is experimented. It is not just a problem of training -as some other races not present in the dataset have been successfully classified by the models- it is a problem of awareness. There is non intentioned bias in the algorithms and this is a malfunctioning software that has to be taken into account. As we have security by design in software architecture discipline why not to have a ethical by design principle?

12.3 Future Work

Derived from the direct and real application of the constructed model there are two clear lines of improvement:

- Improve the algorithm for aligning of extreme poses (spontaneous and naturalistic ones)
- Work with temporal information combining classification models with recursive neural networks for improving the detection of transitions during change of face emotions in videos.



Predictions

3-disgust	53.34%
0-neutral	36.91%
6-sadness	5.58%
7-surprise	3.97%
1-anger	0.19%

Predictions

7-surprise	59.75%
5-happy	21.38%
1-anger	15.14%
0-neutral	1.61%
4-fear	1.19%

Predictions

1-anger	89.91%
5-happy	3.76%
6-sadness	2.07%

Figure 12.1: Wrong classifications in biased classifiers.

And of course, as for all the deeplearning works more tagged data is needed. Face analysis is not a very popular topic, it is not easy to find data so, why not to generate them? As we commented in the introduction some techniques of synthetically generate human face expressions are having successful results and it can be a great generator of training data.

Bibliography

[CCK$^+$17] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint arXiv:1711.09020*, 2017.

[CL08] Manuel G Calvo and Daniel Lundqvist. Facial expressions of emotion (kdef): Identification under different display-duration conditions. *Behavior research methods*, 40(1):109–115, 2008.

[CMD04] Pamela M Cole, Sarah E Martin, and Tracy A Dennis. Emotion regulation as a scientific construct: Methodological challenges and directions for child development research. *Child development*, 75(2):317–333, 2004.

[DCR01] Matthew N. Dailey, Garrison W. Cottrell, and Judith Reilly. CAlifornia Facial Expressions (CAFE), 2001.

[DGD13] Kirsten A Dalrymple, Jesse Gomez, and Brad Duchaine. The dartmouth database of children's faces: acquisition and validation of a new face stimulus set. *PloS one*, 8(11):e79131, 2013.

[EF71] Paul Ekman and Wallace V Friesen. Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124, 1971.

[FSL15] Sachin Sudhakar Farfade, Mohammad J Saberian, and Li-Jia Li. Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650. ACM, 2015.

[HPN$^+$16] Song Han, Jeff Pool, Sharan Narang, Huizi Mao, Shijian Tang, Erich Elsen, Bryan Catanzaro, John Tran, and William J. Dally. DSD: regularizing deep neural networks with dense-sparse-dense training flow. *CoRR*, abs/1607.04381, 2016.

[HRBLM] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[JGS14] Rachael E Jack, Oliver GB Garrod, and Philippe G Schyns. Dynamic facial expressions of emotion transmit an evolving hierarchy of signals over time. *Current biology*, 24(2):187–192, 2014.

[Kin09] Davis E King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LAK$^+$98] Michael J Lyons, Shigeru Akamatsu, Miyuki Kamachi, Jiro Gyoba, and Julien Budynek. The japanese female facial expression (jaffe) database. In *Proceedings of third international conference on automatic face and gesture recognition*, pages 14–16, 1998.

[LCK$^+$10] Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 94–101. IEEE, 2010.

[LDB$^+$10] Oliver Langner, Ron Dotsch, Gijsbert Bijlstra, Daniel HJ Wigboldus, Skyler T Hawk, and AD Van Knippenberg. Presentation and validation of the radboud faces database. *Cognition and emotion*, 24(8):1377–1388, 2010.

[MTH$^+$16] Iacopo Masi, Anh Tran, Tal Hassner, Jatuporn Toy Leksut, and Gérard Medioni. Do We Really Need to Collect Millions of Faces for Effective Face Recognition? In *European Conference on Computer Vision*, 2016.

[PDW07] Amy S Pollick and Frans BM De Waal. Ape gestures and language evolution. *Proceedings of the National Academy of Sciences*, 104(19):8184–8189, 2007.

[Psy99] School of Natural Sciences University of Stirling Psychology. Pain faces database, 1999.

[Ros17] Adrian Rosebrock. Face alignment with opencv and python, 2017.

[SZ14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[TZS$^+$16] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 2387–2395. IEEE, 2016.

[WP17] Jason Wang and Luis Perez. The effectiveness of data augmentation in image classification using deep learning. Technical report, Technical report, 2017.

[WZ16] Xiaolin Wu and Xi Zhang. Automated inference on criminality using face images. *arXiv preprint arXiv:1611.04135*, 2016.

[ZHT$^+$11] Guoying Zhao, Xiaohua Huang, Matti Taini, Stan Z Li, and Matti Pietikäinen. Facial expression recognition from near-infrared videos. *Image and Vision Computing*, 29(9):607–619, 2011.

[ZZLQ16] 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.