EMOTION RECOGNITION BASED ON FACIAL EXPRESSIONS USING CONVOLUTIONAL NEURAL NETWORK

A THESIS SUBMITTED TO
THE FACULTY OF ARCHITECTURE AND ENGINEERING
OF
EPOKA UNIVERSITY

BY

SABRINA BEGAJ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JUNE, 2020

ABSTRACT

EMOTION RECOGNITION BASED ON FACIAL EXPRESSIONS USING CONVOLUTIONAL NEURAL NETWORK

Sabrina Begaj
M.Sc., Department of Computer Engineering
Supervisor: Dr. Ali Osman Topal

Over the last years, there is a large number of studies focused in automatic facial expression analysis because of its practical importance in many human-computer interaction Systems. With the transition of facial expression recognition (FER) from laboratory-controlled to challenging in-the-wild conditions and the recent success of deep learning techniques in various fields, deep neural networks have increasingly been leveraged to learn discriminative representations for automatic FER. In this thesis, we study the challenges of Emotion Recognition Datasets and try different parameters and architectures of the Conventional Neural Networks. The dataset we have used is iCV MEFED, a relatively new, interesting and very challenging.

Keywords: Deep Learning, Facial Expression Recognition, Data Preprocessing, Convolutional Neural Network, Image Recognition

ABSTRAKT

NJOHJA E EMOCIONEVE BAZUAR NË SHPREHITË E FYTYRËS DUKE PËRDORUR RRJETAT NEURALE KONVOLUCIONALE

Sabrina Begaj

Master Shkencor, Departamenti i Inxhinierisë Kompjuterike

Udhëheqësi: Dr. Ali Osman Topal

Gjatë viteve të fundit, është kryer një numër i madh studimesh të cilat janë fokusuar në analizimin e shprehjeve të fytyrës në mënyrë automatike.

Kjo ka ndodhur për shkak të rëndësisë praktike në rritje në jetën e përditshme të marrdhënies njeri-kompjuter. Tranzicioni i njohjes së shprehjeve të fytyrës nga testimet e kontrolluara në laborator në kushte të jetës reale, është fushë me shumë interes në "deep learning" dhe rrjetat neurale po trajnohen që gjithnjë e më shumë të mësojnë përfaqësime dalluese për njohjen automatike të tipareve të fytyrës.

Në këtë punim diplome, ne do të studiojmë sfidat aktuale të Dataseteve të Njohjeve së Emocioneve, duke testuar parametra të ndryshëm të Rrjetave Neurale Konvolucionale. Dataseti i përdorur nga ne në këtë punim është iCV MEFED, një dataset interesant, relativisht i ri dhe shumë sfidues.

Fjalët kyçe: Rrjetat Neurale, Njohja Automatike e Fytyrës, Njohja Automatike e Emocioneve, Deep Learning, Convolutional Neural Network

ACKNOWLEDGEMENTS

I would like to express my special thanks to my supervisor Prof. Dr. Ali Osman Topal for his continuous guidance, encouragement, motivation and support during all the stages of my thesis. I sincerely appreciate the time and effort he has spent to improve my experience during my graduate years.

I am also deeply thankful to Prof. Dr. Arban Uka for his help during my Master Studies and his great contribution of explaining the topics of Image Processing in a very interesting and engaging way. In the same time, I would like to thank also Prof. Dr. Betim Çiço for his insight and invaluable opinions which have affected me over the course of these studies

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

FER	Facial Emotion Recognition
CNN	Convolution Neural Network
NN	Neural Network
AU	Action Units
FACS	Facial action coding system
FL	Facial Landmarks
BEs	Basic Emotions
CEs	Compound Emotions
VM	Virtual Machine
RDP	Remote Desktop Connection
DBN	Deep Belief Network
DAE	Deep AutoEncoder
RNN	Recurrent Neural Network
GAN	Generative Adversarial Network Collagen

CHAPTER 1

INTRODUCTION

Facial expressions are one of the most powerful, natural and universal signals for human beings to transmit and communicate their emotional states and intentions to others [1] [2]. Many studies [3] show that 55% of messages which are related to feelings and attitudes are shown in facial expressions, 7% of which is in the words that are spoken, the rest of which are paralinguistic (the way that the words are said). Facial expression has proven to play a vital role in the entire information exchange process.

1.1. Facial Emotion Recognition Impact

There is a large number of studies which have been conducted on automatic facial expression analysis because of its practical importance in many human-computer interaction Systems. It has gained popularity recently because of its potential applications in the process of diagnosis and national security. These systems can also be useful in many fields in sociable robotics, medical treatment, driver fatigue surveillance, HR specialists, journalists during interviews, education (on students while having a lecture), in investigations, customer services and so many others.

However, micro-expression is considered so fleeting that it is almost undetectable and thus difficult for human beings to detect [4]. As early as the 20th century, Ekman and Friesen [5] defined six basic emotions based on cross-culture study [6], which indicated that humans perceive certain basic emotions in the same way regardless of culture. These prototypical facial expressions are anger, disgust, fear, happiness, sadness, and surprise.

Since the affect model based on basic emotions is limited in the ability to represent the complexity and subtlety of our daily affective displays and other emotion description models, such as the Facial Action Coding System [7] and the continuous model using affect dimensions [8], are considered to represent a wider range of emotions.

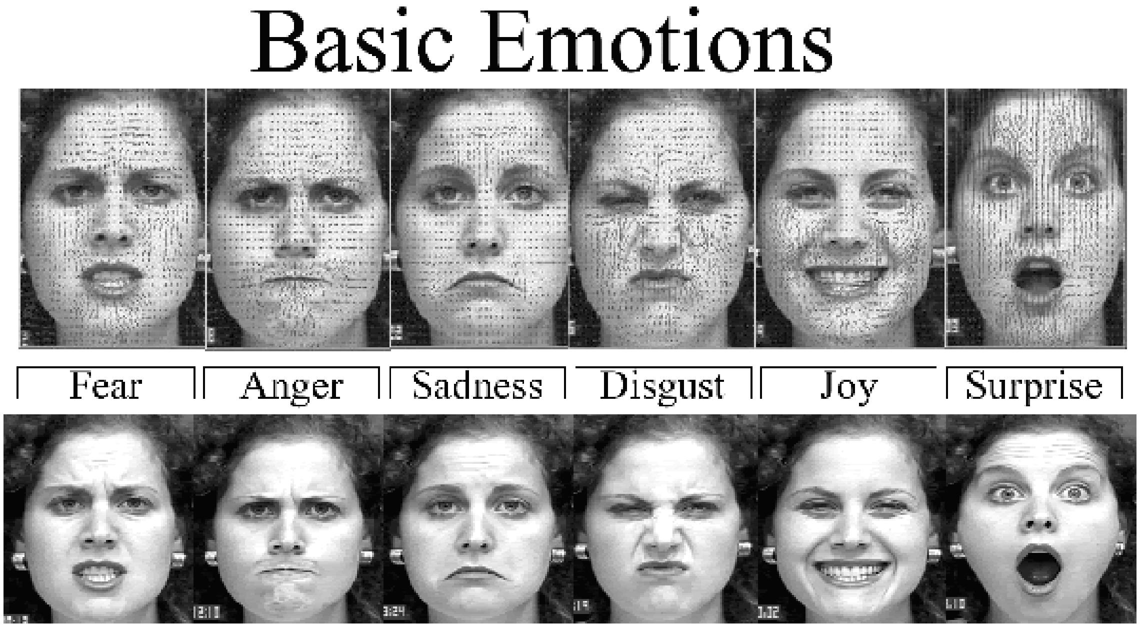


Figure 1. Basic Facial Expressions

1.2. Structure of the Thesis

This thesis is organized in 6 Chapters. In the first Chapter we have a general introduction about the Importance of Facial Emotion Recognition, we brifly showcase the general problems and explain the organization of the thesis.

In Chapter 2 we give an overview of “what is happening” in the “world of Emotion Recognition”. We have described the most used datasets dhe the difference between them. In the other sections we showcase the literature review by categorizing them in two Approaches in this field which are “Hand-Crafted” and Deep Learning.

Our Thesis is based on Deep Learning, that’s why we have a dedicated chapter (Chapter 3) to explain what is Image Classification, how many types of learning do we have, what are ANN, how is CNN organized and lastly, what is Haar Cascade.

In Chapter 4 we describe our methodology, the approach we have regarding this project. We give a description of the dataset that we have chosen, including an explanation for each emotion having in focus the difference between them regarding the face expressions.

In Chapter 5 we showcase the eperiments that we have done regarding dataset and the Convolutional Neural Networks.

Our last Chapter is Chapter six where we give the conculsion of our work and future recommandations.

CHAPTER 2

LITERATURE REVIEW

There are different methods that are used for Face Expression Recognition in the last 20 years but usually they are separated into two main methods: Conventional FER Approach and Deep learning-based approach. This Chapter will give an overview of what these methods consist of, how they are applied in Face Expression Recognition and what are the most used datasets in this area.

2.1. Datasets

In Facial Emotion Recognition are used many kinds of datasets and each of them have their own characteristics. In the majority of studies are used two dimentional static images while in some others are used sequences from 2D videos since they show expressions in many dimentions.

The degree to which FER is successful is highly affected by some certain circumstances such as the light, the background, posture, ect. For this matter they are separated in two main groups which are: “images in the lab” and “images in the wild”. There is no “best group”, which one to choose depends on the purpose of a certain study and the methods used.

In the figure below, are shown two images with two different emotions from the most used datasets such as CK+, MMI, MPI, JAFFE, KDEF. We can easily spot the difference between them. Some are black and white images while some are RGB images. Some of them are focused on the face while blurrying the background some aren’t.

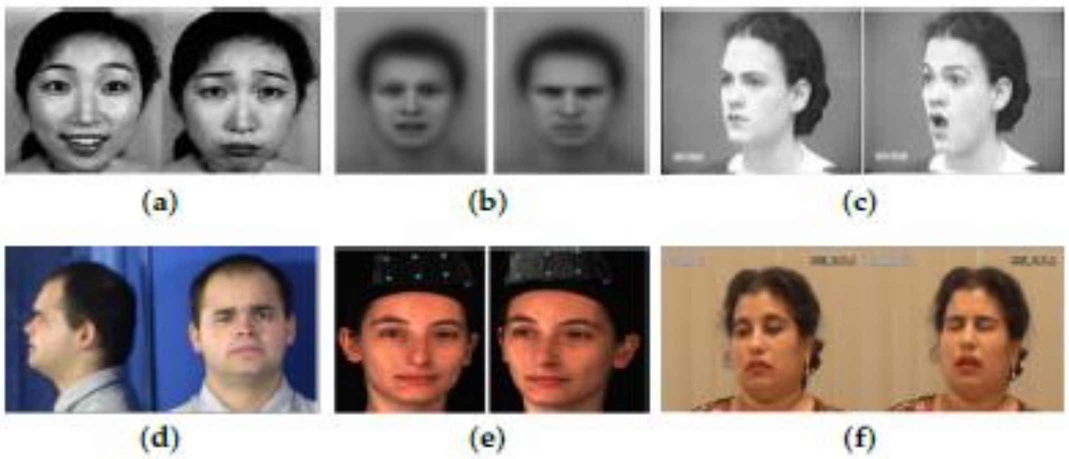


Figure 2. List of Datasets (a) JAFFE; (b) KDEF; (c) CK+; (d) MMI; (e) MPI; (f) UNBC.

Table 1 is a collection of the most used datasets in Emotion Recognition, including characteristics of each dataset such as number of subjects, conditions, ect.

In the first column is the name of the dataset (including the reference paper in brackets), followed by the second column "Samples" which shows how many images or instances it has. For all pictures, we have the number of people / faces who are photographed which are noted as "Subject". Also, in the table is stated that not all the photos are taken in the same conditions. Some of them are taken in the laboratory, some are taken from movie frames, while some others are taken from the web. Usually the ones taken in the lab are photos taken dedicated only to the specified datasets while from the web and movies are considered as images "in the wild" since they are very different from each other (different people, background, ect).

The election method changes from one dataset/image to another. The difference between P and S is that P means Pose, while S means spontaneous. Pose pictures have been taken in the laboratory to specialists who give their guidance how to properly express a certain emotion. Spontaneous pictures have been taken of people who express their emotions at the moment in a spontaneous way (usually the one in the wild).

Description of the most known datasets:

CK+ [9]: The Extended CohnKanade (CK+) database is the most used of all laboratory-controlled databases for FER systems evaluation. This dataset consists of 593 video sequences taken from 123 people or as it is called subjects. These videos are shooted in Laboratory. The “Elicitation method” is that some photos are taken in a spontaneous way, while the others are taken by taking poses. In total there are labeled seven basic expressions (anger, contempt, disgust, fear, happiness, sadness and surprise) based on FACS. The algorithms which have been performed in this database are not uniform because the data is not separated in training, test and validation.

MMI [10], [11]: This database is also laboratory-controlled and it includes 740 images and 2,900 videos. All these are taken in the lab to specialists who give their guidance properly to express an emotion. What differentiate it form CK+, is that in this database, we do not have spontaneous pictures and the sequence begins with a neutral expression and then reaches a peak near the middle before returning to the neutral expression again. The images here are in challenging conditions since the people wear accessories (glasses, mustache, hat, ect).

TFD [12]: The Toronto Face Database (TFD) is a combination of different facial expressions, so it has a large number of samples. Some of them are annotated and are taken in lab conditions. In common with two other datasets above is the elicitation method and six basic expressions & the neutral one. Characteristics of TFD that we have not noticed yet in the above datasets, the face in these images is already detected and normalized in such way that the size of every image is 48*48.

JAFFE [13]: The Japanese Female Facial Expression (JAFFE) includes 213 images of posed expressions of 10 Japanese females. For each person are taken 3-4 images for each six basic expressions and one neutral expression. The disadvantage of this database is that it contains few examples for expression.

Database	**Samples**	**Subject**	**Condit.**	**Elicit**	**Expression distribution**
CK+ [9]	593 image sequences	123	Lab	P & S	6 basic expressions plus contempt and neutral
MMI [11]	740 images and 2,900 videos	25	Lab	P	6 basic expressions plus neutral
JAFFE [13]	213 images	10	Lab	P	6 basic expressions plus neutral
TFD [12]	112,234 images	N/A	Lab	P	6 basic expressions plus neutral
FER-2013 [14]	35,887 images	N/A	Web	P & S	6 basic expressions plus neutral
AFEW 7.0 [15]	1,809 videos	N/A	Movie	P & S	6 basic expressions plus neutral
SFEW 2.0 [16]	1,766 images	N/A	Movie	P & S	6 basic expressions plus neutral
Multi-PIE [17]	755,370 images	337	Lab	P	Smile, surprised, squint, disgust, scream and neutral
BU-3DFE [18]	2,500 images	100	Lab	P	6 basic expressions plus neutral
Oulu-CASIA [19]	2,880 image sequences	80	Lab	P	6 basic expressions
RaFD [20]	1,608 images	67	Lab	P	6 basic expressions plus contempt and neutral
KDEF [21]	4,900 images	70	Lab	P	6 basic expressions plus neutral
EmotioNet [22]	1,000,000 images	N/A	Web	P & S	23 basic expressions or compound expressions

RAF-DB [23]	29672 images	N/A	Web	P & S	6 basic expressions plus neutral and 12 compound expressions
AffectNet [24]	450,000 images (labeled)	N/A	Web	P & S	6 basic expressions plus neutral
iCV – MEFED [25]	31,250 images (labeled)	125	Lab	P	50 basic and compound emotions

Table 1. List of Datasets

To have better results inot only in FER, but in image recognition in general, we apply data augumentation. This method is applied in training data by adding some modifications to the image such as cropping images in different coordinates, rotating in different degrees, scaling in random numbers, shearing, etc. It is important to find the most appropriate modifications to the dataset so it will be in the similar form as it can be in reality. Radom cropping (when the image has a considerable background) and mirroring are the most used types in data augumentation in FER because the show the spatial invariance of a human face and this also helps a lot in expanding the dataset. An example of data augmentation of human facial images using mirroring and cropping is shown in Figure 3.

One technique used for analyzing the data is the usage of Action Units. They are base in FACS systems which classifies the moevements of the muscles of the human face while showing a certain expression. The entire face is separated in different AU based on anatomical features and the combinations of them gives as output a certain expression. In figure 4 is illustrated an example of the combinations of these different Action Units to represent an expression (emotion).

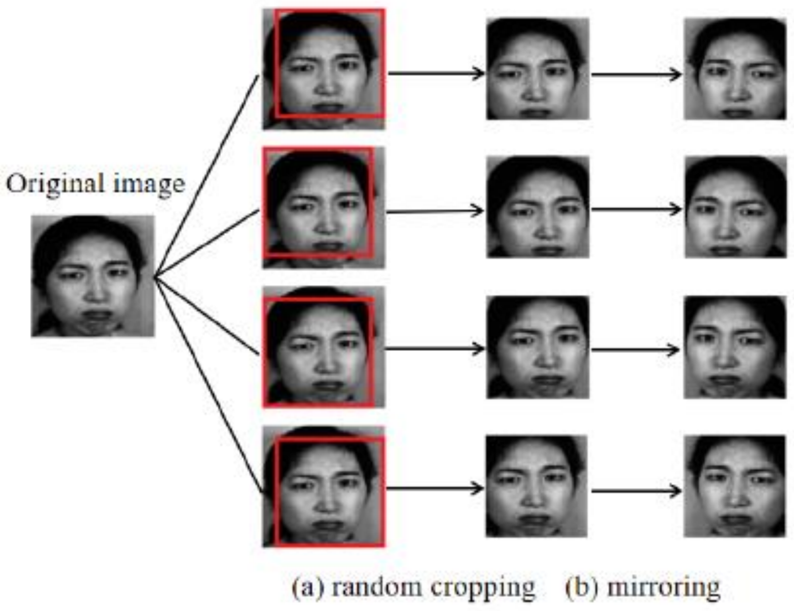


Figure 3. Example of Data Augmentation

In total, there are 46 facial action units encode the basic movements of individual or groups of muscles that are typically observed when a facial expression produces a particular emotion.

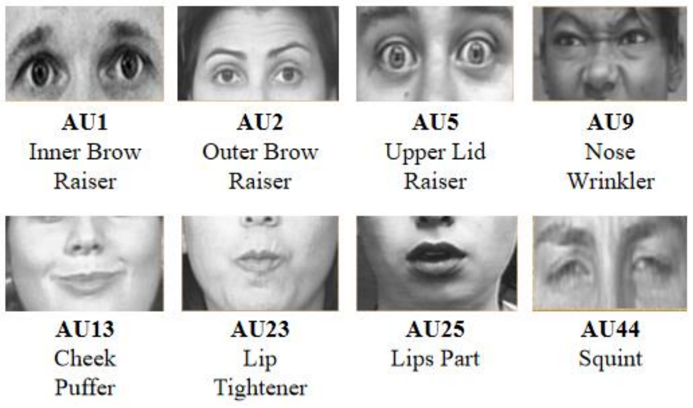


Figure 4. Examples of facial muscle movement

Category	AUs	Category	AUs
Happy	12,25	Sadly disgusted	4,10
Sad	4,15	Fearfully angry	4,20,25
Fearful	1,4,20,25	Fearfully surprised	1,2,5,20,25
Angry	4,7,24	Fearfully disgusted	1,4,10,20,25
Surprised	1,2,25,26	Angrily disgusted	4,25,26
Disgusted	9,10,17	Disgusted surprised	1,2,5,10
Happily sad	4,6,12,25	Happily fearfully	1,2,12,25,26
Happily surprised	1,2,12,25	Angrily disgusted	4,10,17
Happily disgusted	10,12,25	Awed	1,2,5,25
Sadly fearful	1,4,15,25	Appalled	4,9,10
Sadly angry	4,7,15	Hatred	4,7,10
Sadly surprised	1,4,25,26		

Table 2. Basic and Compound Emotion Categories

2.2. "Hand-Crafted" Approach

"Conventional Methods" or "Hand-Crafted" methods are highly dependent on manual feature engineering. The researchers need at first to process the image and to find the most appropriate feature extraction and classification method for the target dataset. This approach can be divided in three main steps which are: image preprocessing, feature extraction and expression/emotion classification, as shown in Figure 1. The next sections explain the process for each step which are done separately but depend on each other.

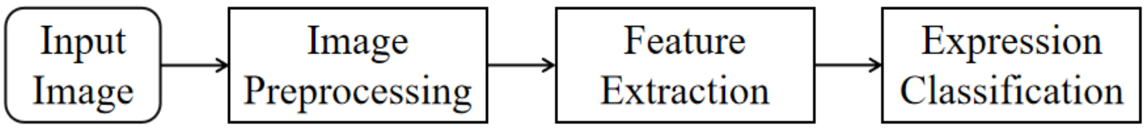


Figure 5. Three main steps used in "Conventional Approach"

2.2.1. Image Preprocessing

The Image Processing step has a huge impact in the performance of the algorithms and the results. It directly affects the extraction of features, consequently the performance of expression / emotion classification. The main goal is to eliminate the irrelevant information of the input images and enhance the detection ability of the important or relevant characteristics / information.

There are different problems that may occur with the images dataset e.g., in image background: occlusion, noise, light intensity and other. Shooting images with different equipment and in different conditions can cause data diversity. All these factors need to be preprocessed before starting the classification.

Main methods used in Preprocessing are:

Noise reduction is usually the first step of preprocessing. The most used filters to remove noise in images are: Adaptive Median Filter (AMF), Average Filter (AF), Gaussian Filter (GF) and Bilateral Filter (BF).

Histogram equalisation is applied to improve the quality of the image using the histogram of that image. It gives better quality of images in the same time without lossing any information.

Face Detection has developed as an independent field. This step is very important because it finds "the location" of the face in the image and extracts the face region. Most used face detector, also considered as state-of-the-art is Viola-Jones Algorithm

Normalisation of the scale and grayscale is use to normalise the color and the size of the of the images. This preprocessing method is used in order to reduce the complexity of the image in order to let the algorithm focus only on the most important features of the face.

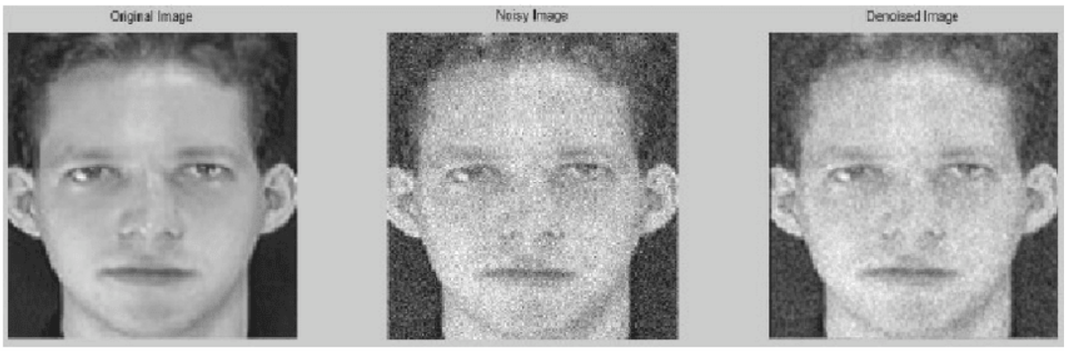


Figure 6. Noise reduction using Gaussian Filter

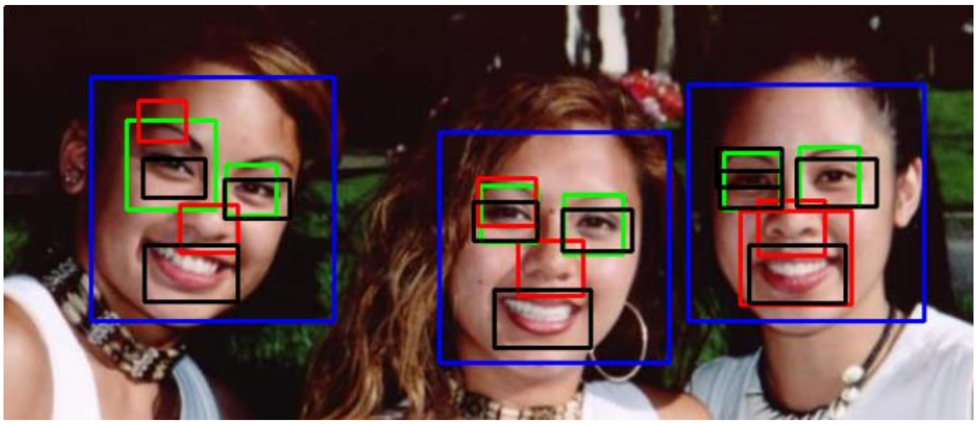


Figure 7. Viola-Jones Algorithm for Face Detection

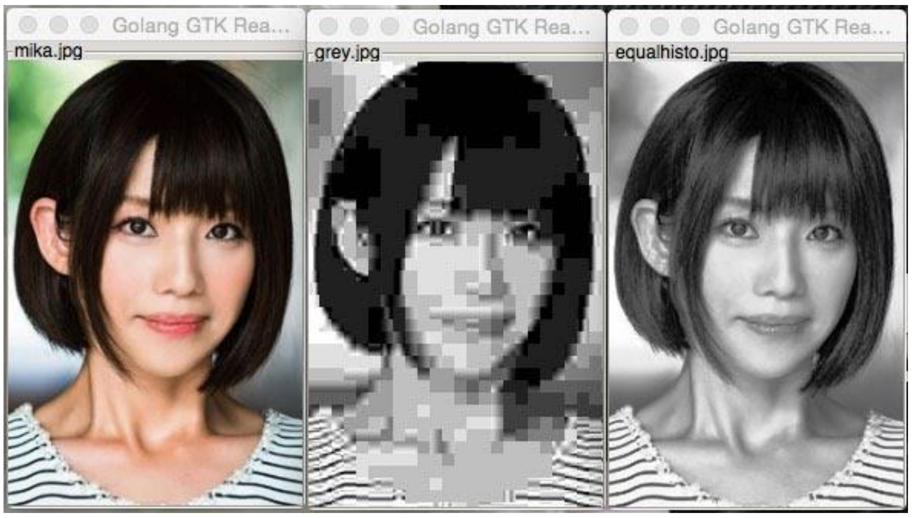


Figure 8. Histogram Equalization

2.2.2. Feature Extraction

Feature extraction is used as a process to get some information or data from the images which are useful. These informations could be symbols, vectors, ect. which actually are not “a real image.” but they “embody” the features of the image. The most used feature extraction methods in FER systems mostly include Local Binary Pattern (LBP), Gabor feature extraction, optical flow method, Haar-like feature extraction, feature point tracking, etc.

Worth mentioning here is Local Binary Pattern, which calculates the brightness relationship between each pixel contained in the image and its local neighbourhood. The binary sequence is then encoded to form a local binary pattern. Finally, it uses a multi-region histogram as a feature description of the image, as shown in Figure 9.

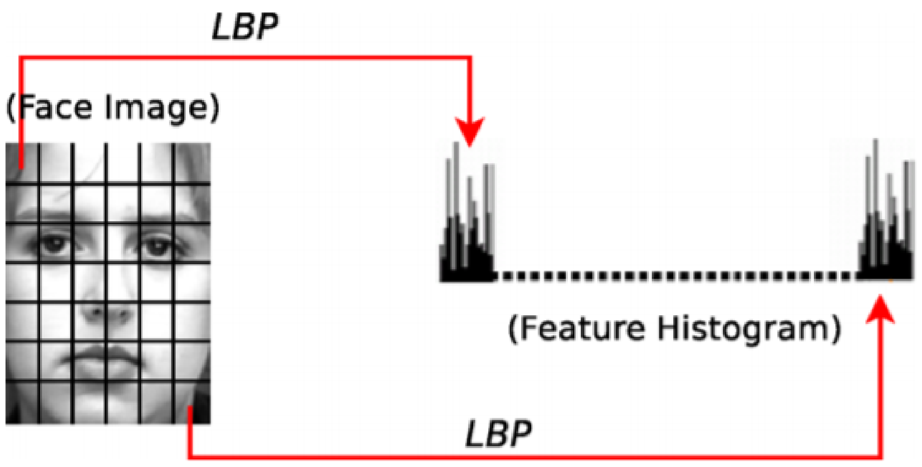


Figure 9. Example of Local Binary Pattern

Feature Extraction is usually the most crucial part of the whole process of “Conventional Approach” since it is directly connected to the performance of the algorithm.

Basically, the results of the entire process are based on what we “feed” to the algorithm and if the features are not extracted properly, even if we used the best classification algorithm, the accuracy would be low.

2.2.3. Expression Classification

The last step of this approach is to find and apply the most appropriate classifier which can correctly predict the emotion in the human face. Most used classifier in Facial Emotion Detection are SVM (Support Vector Machine), kNN (k-Nearest Neighbors), Adaboost (Adaptive Boosting), Bayesian, SRC (Sparse Representation-based Classifier), and PNN (Probabilistic Neural Network). Most of them need small amounts of data and are fast at processing.

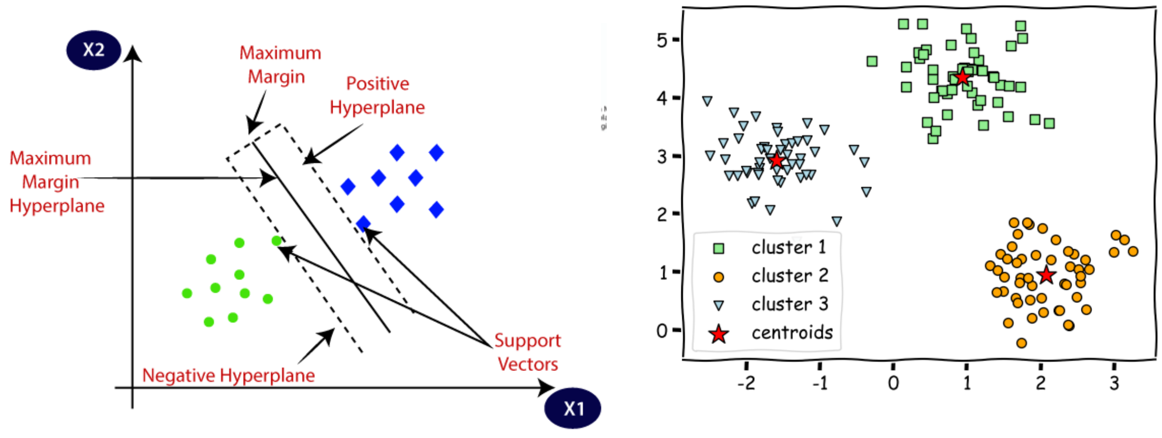


Figure 10. Support Vector Machine (left) and k-Nearest Neighbour (right)

Based on studies done until now, we can come to the conclusion that:

Conventional Approach when compared to Deep Learning Approaches is less dependent on the hardware and the type of data.

In the Conventional Approach feature extraction and classification are treated as separated steps and have to be applied manually, so it is not possible to make them simultaneously. While in Deep Learning Approach can be done together.

2.3. Deep Learning Approach

In Deep Facial Expression Recognition, are three main steps we have to follow:

1. Pre-processing
2. Deep Feature Learning
3. Deep Feature Classification

Since we live in the Age of Technology, it is becoming increasingly necessary for everyday life to have intelligent monitoring. For example, cameras and assistive robots need to understand what is going on with human emotions. Expression recognition is very easy to guess for humans, but extremely difficult task for smartest AI technologies. Automatic recognition of emotions has a lot of problems, starting with the categories of emotions and up to a greater study from psychologists and their collaboration with scientists.

Scientists need to use big datasets in order to carefully research automatic recognition of facial expressions. The datasets have been created by them, but they should be supervised by psychologists who understand emotion recognition better and have to give their input. Unfortunately, the datasets are limited but the scientific community has been trying passionately to create new and useful ones these last years.

2.3.1. Image Preprocessing

As we said, the first step is the pre-processing of visual information. This means that we have to be careful choosing the relevant facial expressions in order to train the neural network. For example, face alignment should be considered carefully, because the first step is always to detect the face and remove non-face areas.

This will reduce the errors that might occur in the future. A very important task here is to augment the data received from alignment in order to make the neural network understand better the face and its positions. The last step here is the Illumination normalization and Pose normalization which has been proposed by Hassner.

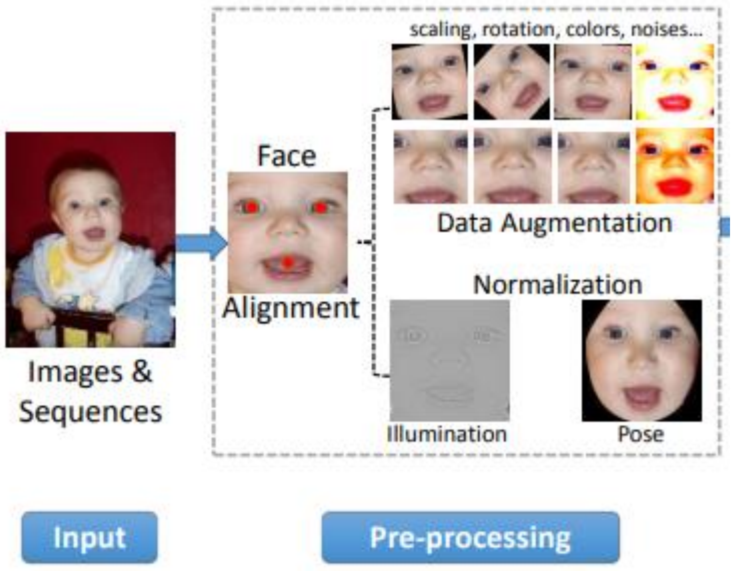


Figure 11. Input, Pre-processing and its steps

2.3.2. Deep Features Learning and Classification

A number of techniques like CNN (Convolutional Neural Network), DBN (Deep Belief Network), DAE (Deep AutoEncoder), RNN (Recurrent Neural Network) and GAN (Generative Adversarial Network) have been created and researched on.

Deep Learning is one of the most researched fields in the world because there have been attempts to capture high-level abstractions for different types of applications performed in industry.

I will briefly explain each of the aforementioned techniques starting with Convolutional Neural Networks. CNNs are very similar to Neural Networks. They are made up of neurons that have learnable weights and biases. These last years, CNNs have been used extensively in Facial Emotion Recognition

They are made of three types of heterogeneous layers:

- Convolutional layers
- Pooling layers
- Fully connected layers

The first layer has learnable filters to convolve through the whole input image and produce various specific types of activation feature maps. The three main benefits are:

- local connectivity
- weight sharing
- shift invariance

Local connectivity makes sure to learn correlations between pixels that are next to each other. So each neuron is getting data from a small group of pixels in the local area (hence the name local connectivity). A local connectivity can be depicted as:

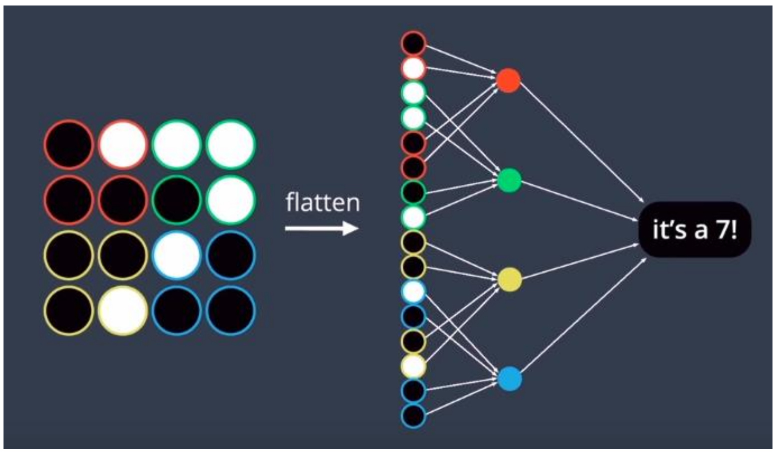


Figure 12. Local Connectivity in CNNs

For weight sharing in the same feature map, we might say that this reduces the numbers of parameters needed to learn. This basically means that the same weight is used for two layers in the model.

The pooling layers are used to reduce the spatial size of the feature maps and the computational cost of the network. Pool layer performs downsampling operation thus reduces the amount of computation performed in the network.

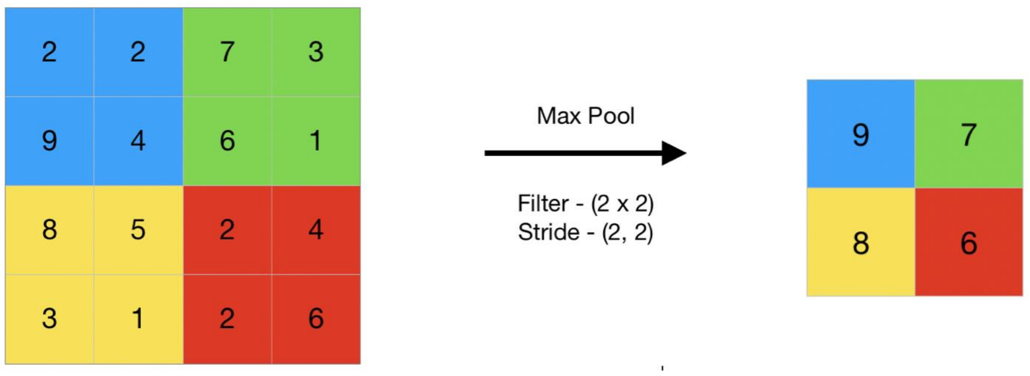


Figure 13. Pool Layers visualized in CNNs

The most used down-sampling strategies for translation invariance are average pooling and max pooling. The fully connected layer is included at the end of the to ensure that all neurons in the layer are fully connected to activations.

	AlexNet [26]	VGGNet [27]	GoogleNet [28]	ResNet [29]
Year	2012	2014	2014	2015
No. of Layers	5+3	13/16 + 3	21+1	151+1
DA	Yes	Yes	Yes	Yes
Dropout	Yes	Yes	Yes	Yes
Inception	No	No	Yes	No
BN	No	No	No	Yes
Used in	[30]	[31], [32]	[33], [34]	[35], [36]

Table 3. Comparison of CNN model achievements

Some of the configurations in well known Convolutional Neural Networks are:

- Region based CNNs (R-CNN) [37]
- Faster R-CNNs [38]
- 3D CNNs [39]

The last type of CNNs mentioned above, can greatly contribute to image identification even in medical fields with the scans of cancerous tissue.

In Table above we can see a comparison of CNN models and their achievements Another technique I mentioned before is Deep Belief Network (DBN) which is proposed by Hinton [40]. Usually the DBN is built with a stack of Restricted Boltzmann Machines which is a generative stochastic artificial neural network that accepts continuous input. In a DBN, the units in higher layers are trained to learn the conditional dependencies among the units in the adjacent lower layers, except the top two layers, which have undirected connections.

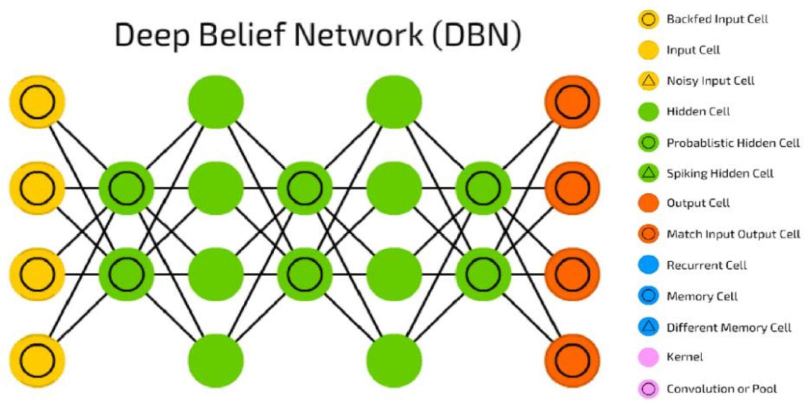


Figure 14. An easy representation of DBN

Different from DBNs are DAEs or differently called as Deep Autoencoders, which were introduced at [41] in order to have dimensionality reduction. There are some different types of DAE like the one mentioned in “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” which is a Denoising Encoder, or the Contractive Autoencoder mentioned in “Contractive auto-encoders: Explicit invariance during feature extraction,”.

At the root of any Autoencoder is the unsupervised learning algorithm which uses back propagation to generate values very similar to the input value.

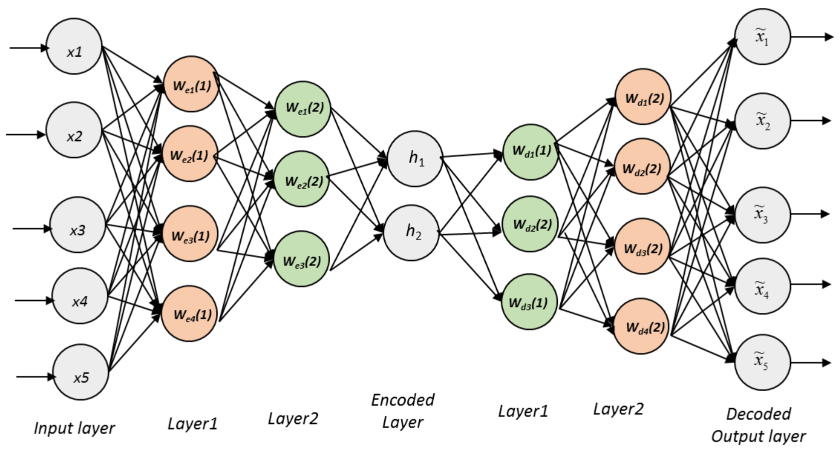


Figure 15. The architecture of Stacked Autoencoder

I also mentioned Generative Adversarial Network (GAN) which was introduced in 2014 by Goodfellow et al [42] which is very interesting because it has been widely seen in social media over the last years with constructing for example an image which represents how a person will look when he gets older. This is the most basic usage but we can see it more clearly in action in Photoshop Content Aware tool.

2.4. Evaluation of the Approaches

There are differences in the performance between conventional FER approaches and deep learning-based FER approaches. The performances of these two approaches are discussed separately as below:

The conventional FER approaches are based on manual feature extraction and less dependent on data and hardware, which has advantages in small data sample analysis. Can achieve promising results on 2D datasets collected under unique conditions.

All the mentioned conventional FER approaches obtain a high accuracy of over 90% on the JAFFE and CK+ datasets, which are similar to deep learning-based FER approaches.

Differences appear when FER approaches are applied to datasets that have significant interpersonal variations. Take the performance on the MMI dataset as an example, the average accuracy of the conventional FER approaches is about 60.94%, while the deep learning-based approaches is about 73.28%.

For more challenging task like FER under wild environmental conditions, conventional approaches are rarely applied since feature extraction for complex datasets is still an obstacle.

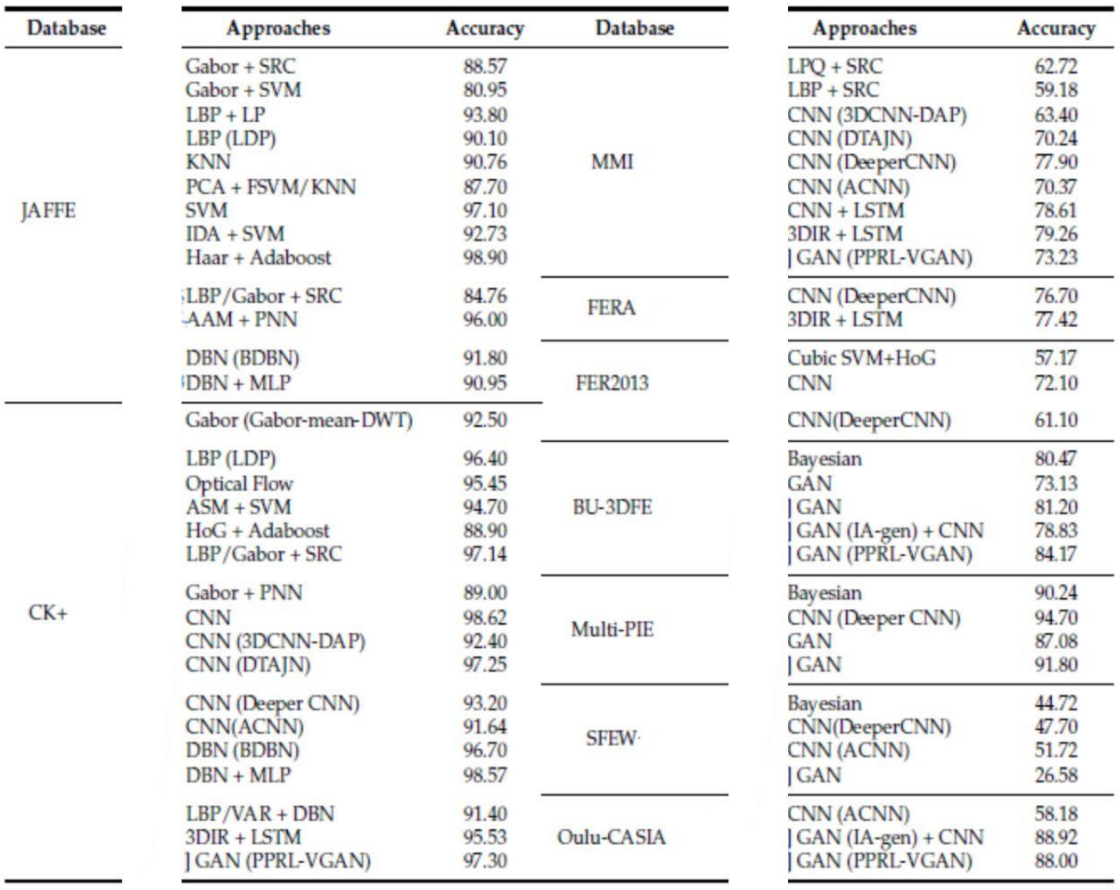
Deep learning-based FER approaches highly reduce the reliance (dependencies) on image preprocessing and feature extraction and are more robust to environments with different elements.

• As shown in CNN framework can be applied to almost all the FER datasets and achieve stable accuracy. The characteristics of CNN (e.g., local connectivity and weight sharing) make it advantageous in image processing.

• DBN is able to capture general appearance changes and to train on large but sparsely labelled datasets. By combining with other methods, DBN can achieve promising results for the FER on JAFFE and CK+ datasets. However, performance on other listed datasets currently cannot reveal its advantages well. Perhaps more attempts can be made for FER in wild environmental conditions in the future.

• Performances on MMI and CK+ datasets indicate that LSTM-based FER approaches have an advantage on video sequences. As a type of RNN that based on long-range context modelling, LSTM network is well suited for the temporal feature extraction of consecutive frames.

• Significant progress on BU-3DFE, Multi-PIE and Oulu-CASIA is made by using GAN-based FER approaches. GAN model, composed of a generator and a discriminator, has been successfully applied in image synthesis to generate facial images, videos and other impressively realistic images. Hence, GAN-based models conducive for pose-invariant and identity-invariant expression recognition.



Database	Approaches	Accuracy
JAFFE	Gabor + SRC	88.57
	Gabor + SVM	80.95
	LBP + LP	93.80
	LBP (LDP)	90.10
	KNN	90.76
	PCA + FSVM/KNN	87.70
	SVM	97.10
	IDA + SVM	92.73
	Haar + Adaboost	98.90
	LBP/Gabor + SRC	84.76
	AAM + PNN	96.00
	DBN (BDBN)	91.80
	DBN + MLP	90.95
CK+	Gabor (Gabor-mean-DWT)	92.50
	LBP (LDP)	96.40
	Optical Flow	95.45
	ASM + SVM	94.70
	HoG + Adaboost	88.90
	LBP/Gabor + SRC	97.14
	Gabor + PNN	89.00
	CNN	98.62
	CNN (3DCNN-DAP)	92.40
	CNN (DTAJN)	97.25
	CNN (Deeper CNN)	93.20
	CNN(ACNN)	91.64
	DBN (BDBN)	96.70
	DBN + MLP	98.57
	LBP/VAR + DBN	91.40
	3DIR + LSTM	95.53
	] GAN (PPRL-VGAN)	97.30

Database	Approaches	Accuracy
MMI	LPQ + SRC	62.72
	LBP + SRC	59.18
	CNN (3DCNN-DAP)	63.40
	CNN (DTAJN)	70.24
	CNN (DeeperCNN)	77.90
	CNN (ACNN)	70.37
	CNN + LSTM	78.61
	3DIR + LSTM	79.26
	| GAN (PPRL-VGAN)	73.23
FERA	CNN (DeeperCNN)	76.70
	3DIR + LSTM	77.42
FER2013	Cubic SVM+HoG	57.17
	CNN	72.10
	CNN(DeeperCNN)	61.10
BU-3DFE	Bayesian	80.47
	GAN	73.13
	| GAN	81.20
	| GAN (IA-gen) + CNN	78.83
	| GAN (PPRL-VGAN)	84.17
Multi-PIE	Bayesian	90.24
	CNN (Deeper CNN)	94.70
	GAN	87.08
	| GAN	91.80
SFEW	Bayesian	44.72
	CNN(DeeperCNN)	47.70
	CNN (ACNN)	51.72
	| GAN	26.58
Oulu-CASIA	CNN (ACNN)	58.18
	| GAN (IA-gen) + CNN	88.92
	| GAN (PPRL-VGAN)	88.00

Table 4. List of Databases, Approaches and their Accuracy

Based on these results, we decided to choose for our study the Deep Learning Approach

CHAPTER 3

DEEP LEARNING CNN AND OBJECT DETECTION

This chapter gives an overview of Deep Learning. It explains briefly what Image Classification is and its uses. It lists the three types of Learning and the difference between them, mostly focusing on Supervised Learning. Since one of the most important parts of this thesis is the implementation of Convolutional Neural Networks, it is necessary to understand where they "came from" and how they were inspired. That is why we give a brief explanation of them in this Chapter.

In this thesis we work with face images datasets and they come in many different shapes and sizes. To prepare the data for our model we first should detect the face and then crop it. Haar Cascade algorithm is used for face detection.

3.1. Image Classification

Classification includes a broad range of decision-theoretic approaches to the identification of images. All classification algorithms are based on the assumption that the image in question depicts one or more features and that each of these features belongs to one of several distinct and exclusive classes. Image classification analyzes the numerical properties of various image features and organizes data into categories. Classification algorithms typically employ two phases of processing: training and testing.

$$f(x): x \to \Delta; \quad x \in R^n, \quad \Delta = \{c1 , c2 , \ldots, cL \}$$

Number of bands = n; Number of classes = L

f(.) is a function assigning a pixel vector x to a single class in the set of classes Δ

In simple words, Image classification mean - assigning pixels in the image to categories or classes of interest. Is a process of mapping numbers with symbols.

3.2. Types of Learning

In Deep Learning there are three types of learning: Unsupervised Learning, Supervised learning and Semi-Supervised Learning.

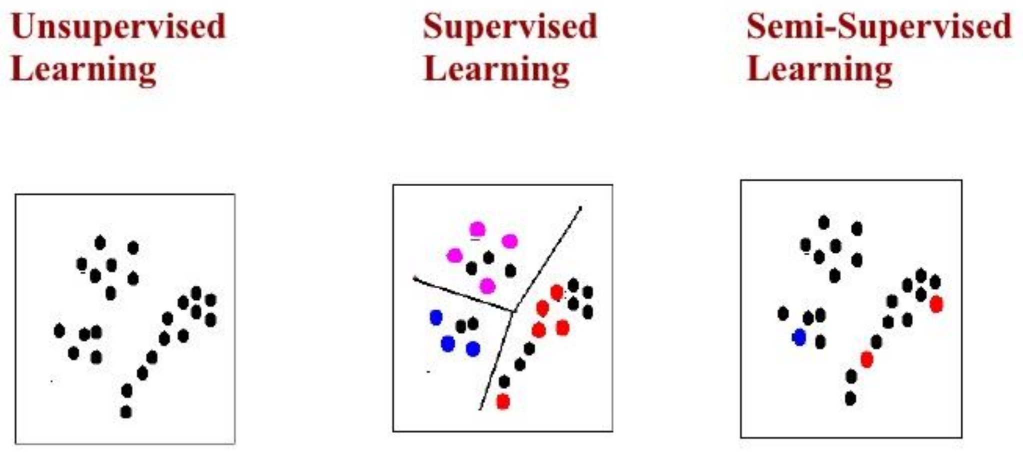


Figure 16. Three types of Learning

Unsupervised learning is used mainly to discover patterns and detect outliers in data. Unsupervised learning is the training of machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unsupervised Learning has two main categories of algorithms: Clustering and Association.

Semi-supervised machine learning is a combination of supervised and unsupervised machine learning methods. In this type of learning, the algorithm is trained upon a combination of labeled and unlabeled data. Typically, this combination will contain a very small amount of labeled data and a very large amount of unlabeled data. The basic procedure involved is that first, the programmer will cluster similar data using an unsupervised learning algorithm and then use the existing labeled data to label the rest of the unlabeled data.

3.2.1. Supervized Learning

In computer Vision image classification is applied in order to ‘understand’ the contents of an image. The aim of a classification system is that based on predefined categories, predict and add labels to the inputs. In our case since the dataset is made of images, the goal would be that based on predefined categories such as emotions, by classifying the images, we add labels to them based on the categories we have. Classifying a certain image as expressing sadness, happiness, ect.

Supervised Learning consist of target or outcome variable (dependent variable). That target variable is to be predicted from given independent variables (Predictors). With the help of these variables, we generate a function that maps to desired outputs. We need to train this process until we get the desired level of accuracy on training data. The algorithm modifies according to the pattern it perceives in the input and output received. Supervised Learning has two categories of algorithms: Classification and Regression.

Supervised learning always has a clear objective and can be easily measured for accuracy. The training of the machine is also tightly controlled, which leads to very specific behavioral outcomes.

On the downside, it is often very labor-intensive, as all data needs to be labeled before the model is trained, which can take hundreds of hours of specialized human effort. The costs can become astronomical. This creates an overall slower training process and may also limit the data that it can work with.

3.1. Convolutional Neural Network

Differently from traditional feedforward Neural Networks, in CNNs, there are not used Fully Connected layers until the very last layer(s) in the network. We can thus define a CNN as a neural network that swaps in a specialized “convolutional” layer in place of a “fully-connected” layer for at least one of the layers in the network.

A nonlinear activation function, such as ReLU, is then applied to the output of these convolutions and the process of convolution continues (along with a mixture of other layer types to help reduce the width and height of the input volume and help reduce overfitting) until the end of the network is reached and apply one or two Fully Connected layers where we can obtain our final output classifications.

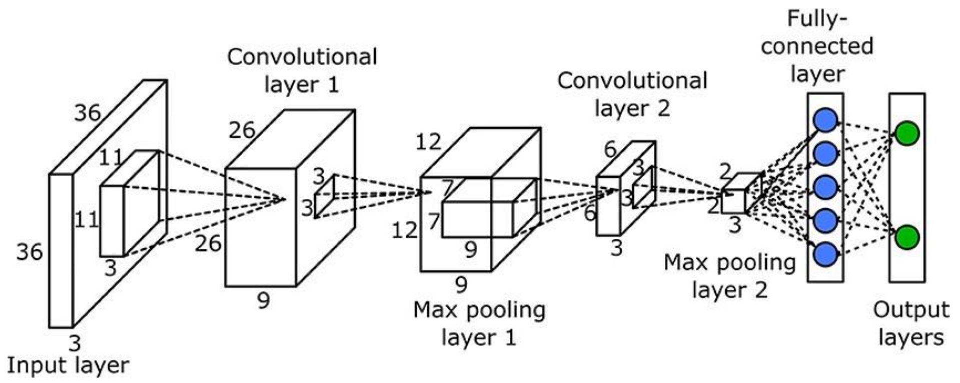


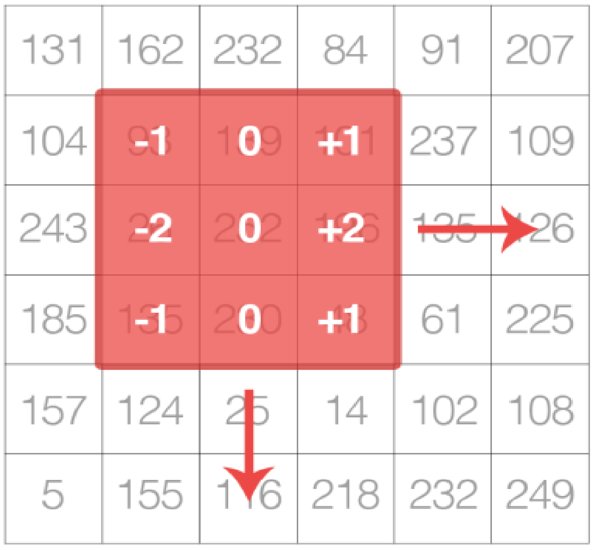
Figure 17. Overview of Input, Output Layers and Pooling

Each layer in a CNN applies a different set of filters, typically hundreds or thousands of them, and combines the results, feeding the output into the next layer in the network and from those layers CNN learns the values automatically.

In terms of deep learning, an (image) convolution is an element-wise multiplication of two matrices followed by a sum.

A kernel is a matrix, which is slid across the image and multiplied with the input such that the output is enhanced in a certain desirable manner. It can be visualized as a small matrix that slides across, from left-to-right and top-to-bottom, of a larger image.

At each pixel in the input image, the neighborhood of the image is convolved with the kernel and the output stored. As shown in the picture below:



$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

n_{in}:	number of input features
n_{out}:	number of output features
k:	convolution kernel size
p:	convolution padding size
s:	convolution stride size

Figure 18. Kernel and the input image plus the Formula for the number of output features

Convolutional Neural Network Layers are:

- Convolutional Layers
- Activation Layers
- Pooling Layers
- Fully-connected Layers
- Batch Normalization
- Dropout

More specifically they are explained in 4th Chapter in Architecture of Convolutional eural Network.

3.2. Object Detection using Haar Cascade

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features” in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

The algorithm has four stages:

- Haar Feature Selection - First step is to collect the Haar Features. A Haar feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.
- Creating Integral Images – are used to make the prevous process very fast.
- Adaboost Training - selects the best features and trains the classifiers that use them. This algorithm constructs a “strong” classifier as a linear combination of weighted simple “weak” classifiers. The process is as follows.
- Cascading Classifiers - During the detection phase, a window of the target size is moved over the input image, and for each subsection of the image and Haar features are calculated.

In this algorithm, each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

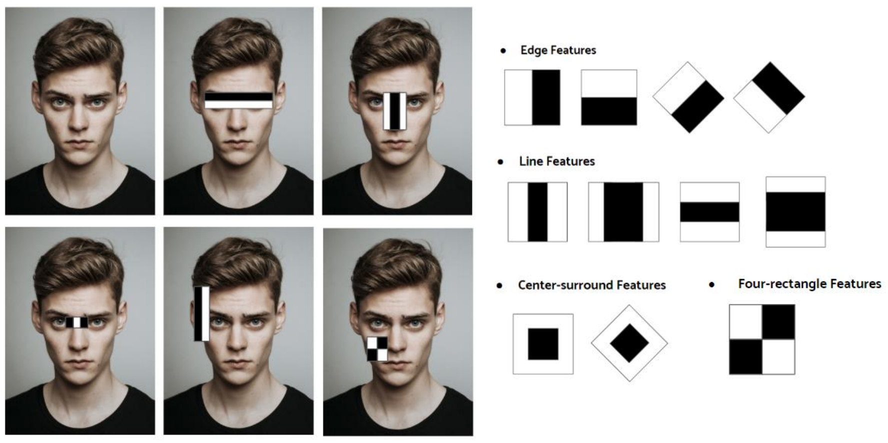


Figure 19. Figure Features of Haar Cascade Algorithm

CHAPTER 4

MATERIALS AND METHODS

4.1. Methodology

In this thesis we use two metodologis the Qualitative Method because we have a wide literature review, explanations and then Quantitative Method since we are going to apply different algorithms and provide calculations while analyzing the performance.

Qualitative Approach

Qualitative Research is a very needed part of our research. It helped us to gain a deep understanding of the method used. This was based on the research papers that are mentioned in the literature review about the latest trends in Emotion Recognition and the stare-of-art algorithms. This approach is done in order to provide a rich description of the method used.

Quantitative Approach

Qualitative Research is the core of the thesis since it has an experimental approach. There have been different approaches to preprocess the challenging dataset, to find most fitted algorithms and to adapt the right parameters in Convolutional Neural Networks.

4.2. Dataset Description and Challenges

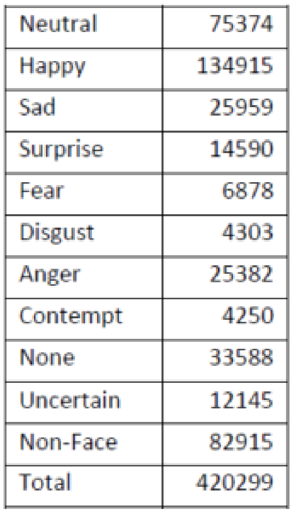
We had a very difficult time choosing the right dataset to work with. Not because there are limited datasets but because the work, I can do with the datasets is limited. we intended for this thesis to have a complete study of the matter from raw photo quality up to training and testing the neural network. After proper research we had to choose between:

FER2013 Dataset

This dataset is relatively old and has been used many times. But the main issue is that the accuracy has not been good. Cubic SVM + HoG has 57.17% accuracy; CNN has 72.1% accuracy while Deep CNN 61.1%. So, dealing with this particular dataset could be worthwhile.

AffectNet Dataset

AffectNet is a new and promising dataset which claims to have more than 1 million images collected from three search engines using emotion related keywords in six different languages. This dataset is excellent to train Neural Networks but on the down side, the great number of images in this dataset need a lot of processing power and it would be very helpful to have a GPU also. Since my resources are limited and AffectNet is quite big it was a difficult choice. We decided to not have my research based on it.



Neutral	75374
Happy	134915
Sad	25959
Surprise	14590
Fear	6878
Disgust	4303
Anger	25382
Contempt	4250
None	33588
Uncertain	12145
Non-Face	82915
Total	420299

Figure 20. The number of manually annotated images for each expression in AffectNet Dataset

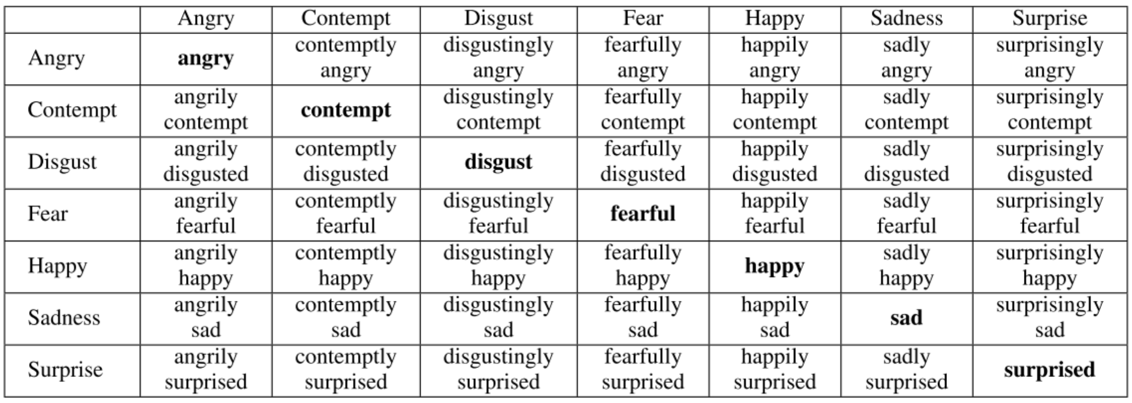
iCV MEFED Database (Multi Emotion Facial Expression Dataset)

This dataset is created by iCV Research Lab in Estonia and it consists of 125 different subjects who act 50 emotions each in front of a Canon 60D camera. The image sample is 5184x3456 pixel and each of them is under the same lighting condition in a green background. Subjects have been trained by psychologists to express their emotions effectively. There are seven basic emotional states plus Neutral: 1-Anger, 2-Contempt, 3-Disgust, 4-Fear, 5-Happiness, 6-Sad, 7-Surprise, N-Neutral. However, the creators of this dataset intended to have mixed emotions for example: angrily surprised.

This is a very interesting dataset because it is very new and hasn't been tested much. Also, the image samples are very big so we have more freedom to choose the cropping and resizing technique. [32] Another reason to choose this dataset is the supervision of psychologists over subjects. FER2013 and AffectNet have images taken from the internet which are not necessarily well taken and are not dedicated to the matter.

On the other hand, this dataset is not very big and dealing with high image samples is time consuming. However, I decided to chose this dataset because I can have a full study of Convolutional Neural Network [33] starting from raw image data, cropping it in two different ways (manually and automatically through face detection algorithm Viola-Jones), creating labels manually for seven basic emotions by choosing the most dominant emotion in the other 42 mixed ones.

It is very appealing to me to understand the differences between the manually cropped images and the automatically cropped ones. The difference in accuracy between man and machine would be at test and also we would understand how this affects the experiment. Since the subjects have been trained specifically to have a dominant emotion, I could be part of manually labeling the images myself. I chose to do the labelling because in the [34] "Dominant and Complementary Emotion Recognition from Still Images of Faces" paper published in April 2018 from the creators of this dataset, it is specifically said in the Abstract that: "Experiments indicate that pairs of compound emotion (e.g., surprisingly-happy vs happily-surprised) are more difficult to be recognized if compared with the seven basic emotions.".



	Angry	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Angry	**angry**	contemptly angry	disgustingly angry	fearfully angry	happily angry	sadly angry	surprisingly angry
Contempt	angrily contempt	**contempt**	disgustingly contempt	fearfully contempt	happily contempt	sadly contempt	surprisingly contempt
Disgust	angrily disgusted	contemptly disgusted	**disgust**	fearfully disgusted	happily disgusted	sadly disgusted	surprisingly disgusted
Fear	angrily fearful	contemptly fearful	disgustingly fearful	**fearful**	happily fearful	sadly fearful	surprisingly fearful
Happy	angrily happy	contemptly happy	disgustingly happy	fearfully happy	**happy**	sadly happy	surprisingly happy
Sadness	angrily sad	contemptly sad	disgustingly sad	fearfully sad	happily sad	**sad**	surprisingly sad
Surprise	angrily surprised	contemptly surprised	disgustingly surprised	fearfully surprised	happily surprised	sadly surprised	**surprised**

Table 5. The compound number of emotions in iCV MEFED Database

That made it easier for me to decide that in fact I really should do another labelling even though it requires more time and effort.

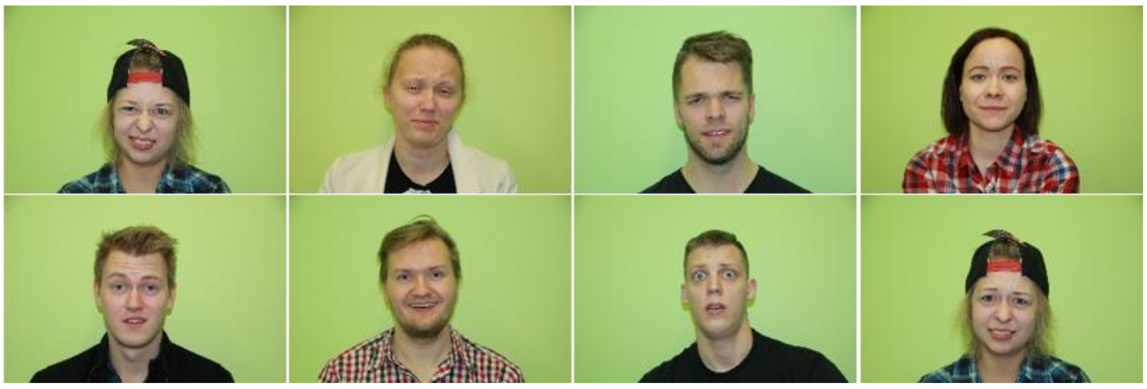


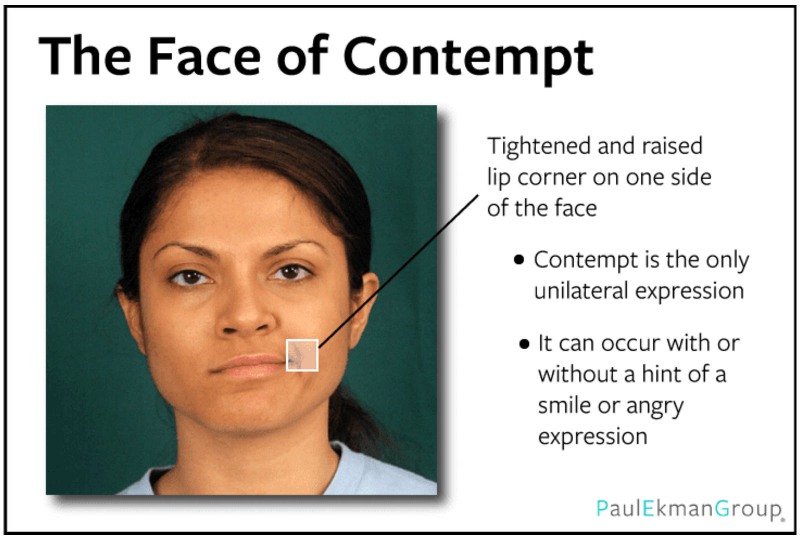
Figure 21. A preview of iCV MEFED Dataset

4.3. Emotions Explanations

In this thesis we analyze 7 main emotions which are based on the selection on Paul Erkam psychologist who has studied for many years the human emotions and not onlyfrom face expressions. Below we are going to explain shortly 7 emotions. The main idea of this section is to give an understanding of what we are actually predicting and how do they change from each other regarding the face movements.

Contempt is described as a feeling of dislike for and superiority (usually morally) over another person, group of people, and/or their actions. It has been accepted by many emotions experts to be a universal emotion; however, some emotions scientists still don’t distinguish contempt as a distinct emotion.

Regarding the Facial Expression the main distinguishing feature is the upper lip only on one side. This expression is not symmetrical and often accompanied by anger.



The Face of Contempt

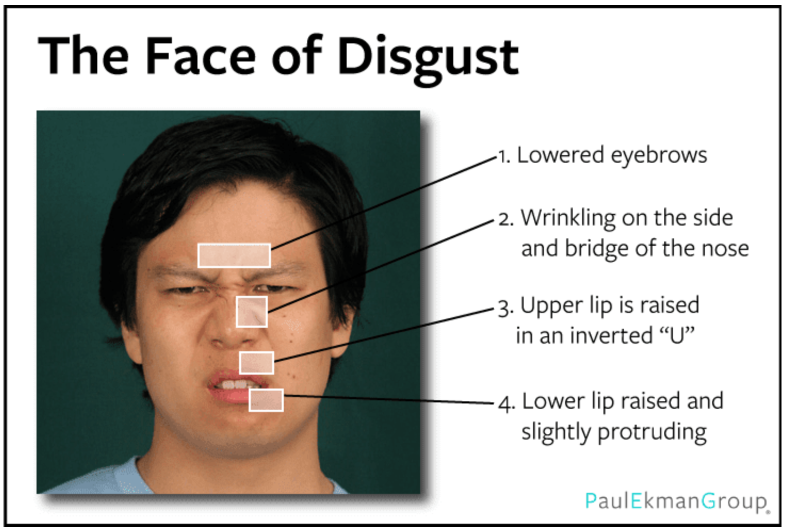
Tightened and raised lip corner on one side of the face

- Contempt is the only unilateral expression
- It can occur with or without a hint of a smile or angry expression

PaulEkmanGroup

Figure 22. Distinguishing Face Expressions for Contempt

Disgust arises as a feeling of aversion towards something offensive. We can feel disgusted by something we perceive with our physical senses (sight, smell, touch, sound, taste), by the actions or appearances of people, and even by ideas. One of the main distinguishing features in disgust is the wrinkling of the nose combined with lower lip raised.



The Face of Disgust

1. Lowered eyebrows
2. Wrinkling on the side and bridge of the nose
3. Upper lip is raised in an inverted "U"
4. Lower lip raised and slightly protruding

PaulEkmanGroup

Figure 23. Distinguishing Face Expressions for Disgust

Fear arises with the threat of harm, either physical, emotional, or psychological, real or imagined. While traditionally considered a “negative” emotion, fear actually serves an important role in keeping us safe as it mobilizes us to cope with potential danger. This emotion is usually confused with “surprise” not only by the algorithm but by human judgment too. The eyebrows are raised and pulled together, while eyelids are tense.

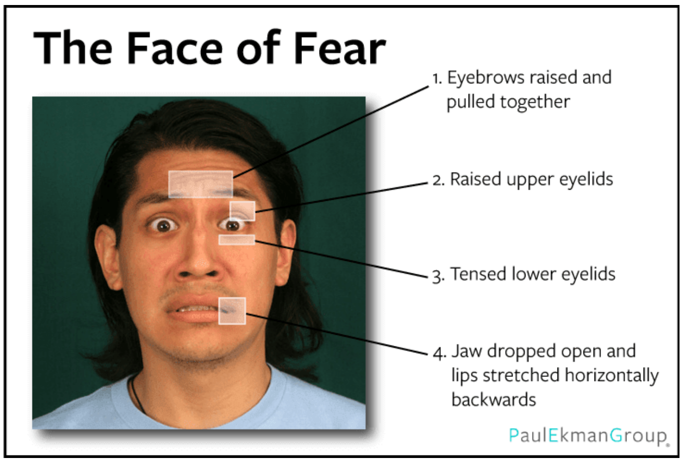

The Face of Fear
1. Eyebrows raised and pulled together
2. Raised upper eyelids
3. Tensed lower eyelids
4. Jaw dropped open and lips stretched horizontally backwards
PaulEkmanGroup

Figure 24. Distinguishing Face Expressions for Fear

Happiness is one of the seven emotions which is easily distinguishable from other six emotions regarding human judgement and algorithms performance. People use the word happiness to refer to their overall sense of well-being or evaluation of their lives rather than a particular enjoyment emotion. Smiling is the universal indicator of happiness where lips are pulled back and usually the teeth are exposed.

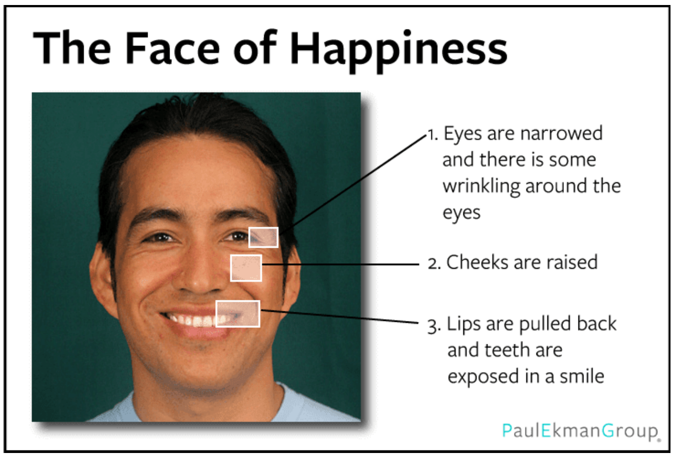

The Face of Happiness
1. Eyes are narrowed and there is some wrinkling around the eyes
2. Cheeks are raised
3. Lips are pulled back and teeth are exposed in a smile
PaulEkmanGroup

Figure 25. Distinguishing Face Expressions for Happiness

Sadness is one of the seven universal emotions experienced by everyone around the world resulting from the loss of someone or something important. What causes us sadness varies greatly based on personal and cultural notions of loss. Sadness can also be experienced along with other emotions, such as: anger, joy, fear. One very strong and reliable sign of sadness is the angling-up of the inner corners of the eyebrows

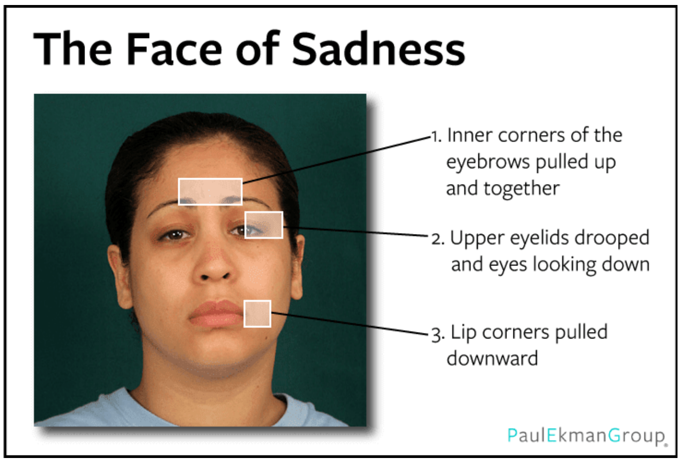

The Face of Sadness
1. Inner corners of the eyebrows pulled up and together
2. Upper eyelids drooped and eyes looking down
3. Lip corners pulled downward
PaulEkmanGroup

Figure 26. Distinguishing Face Expressions for Sadness

Surprise arises when we encounter sudden and unexpected sounds or movements. As the briefest of the universal emotions, its function is to focus our attention on determining what is happening and whether or not it is dangerous.

Surprise is the briefest of all the emotions, lasting a few seconds at most. Even though this expression is usually confused with fear, in surprise, the eyebrows are raised but show more curve than seen in fear. The upper eyelids and jaws are also more relaxed.

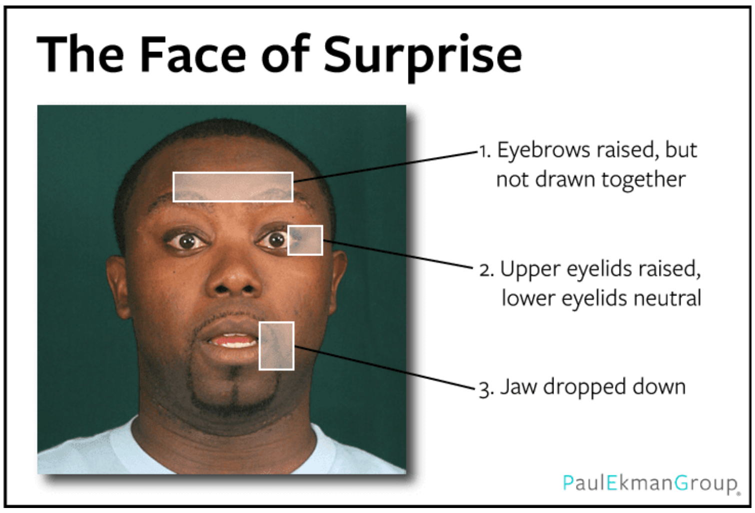

The Face of Surprise
1. Eyebrows raised, but not drawn together
2. Upper eyelids raised, lower eyelids neutral
3. Jaw dropped down
PaulEkmanGroup

Figure 27. Distinguishing Face Expressions for Surprise

Anger is one of the seven universal emotions which arises when we are blocked from pursuing a goal and/or treated unfairly. In anger the eyebrows come down and together, the eyes glare, and there is a narrowing of the lip corners. During conscious suppression or unconscious repression of anger, the expression may be less obvious, though the person may show signs of their anger in a split-second micro expression.

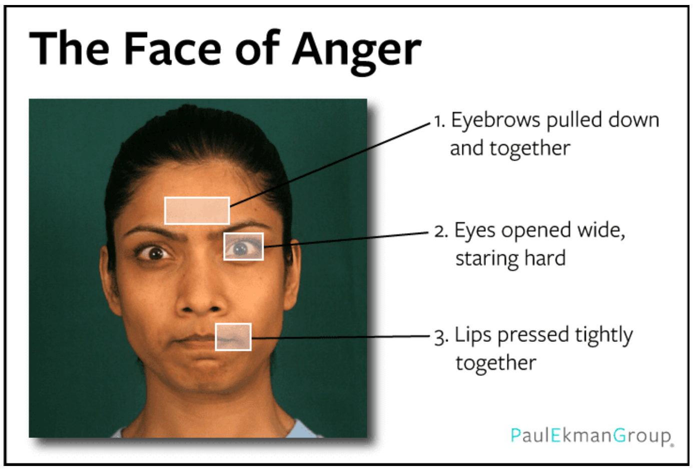

The Face of Anger
1. Eyebrows pulled down and together
2. Eyes opened wide, staring hard
3. Lips pressed tightly together
PaulEkmanGroup

Figure 28. Distinguishing Face Expressions for Anger

4.3.1. Libraries and Tools

The experiments in this thesis are done using Python Scripting Language because of its benefits regarding AI. All the experiments can be done in pure Python Language but since It provides many packages libraries which make the code short, understandable and organized, we have chosen the second method.

Below are listed the main packages used and a short explanation for the main packages used in this project:

Numpy - When we work with Images, we always convert them into multidimensional arrays, in 1 or 2 dimensional arrays. In order to do this conversion and to work with arrays we use the Numpy package.

Pandas - We use this package for data manipulation and analysis. It is very helpful when our data is in .csv format.

Tensorflow/Keras - Keras is a Neural Network library which is built on top of Tensorflow and is easier to use when we compare it with TF. All the layers of our Network are composed by Keras API.

Matplotlib - This library helps us with image visualizations. All the graphics and visualizations in this thesis are created using Matplotlib.

CV2 (Open-CV) - As its name states (Open Source Computer Vision Library) is a very helpful library when we work with images starting from converting them to grayscale. It has more than 2500 optimized algorithms which can be used in images, like face detection, image recognition, etc.

Since we have done many experiments and we have tried to choose the best system and working environment for this project we have use three different IDE:

PyCharm is a very good choice when the code is organized in different files/modules and directories.

Jupyter Notebook which works at the local level is not the best choice when working with different modules but you can run your code separated in blocks which helps a lot in debugging and better understanding your code.

Google Colaboratory (Colab) is an online notebook (Very similar to Jupyter) which is specifically designed for writing and executing python code and especially for Machine Learning since it has all the necessary packages and it gives you the opportunity to use GPUs.

4.4. Architecture of Convolutional Neural Networks (CNN)

In this section we are going to show the basic architecture of our network. In our model there are going to be many layers and different parameters but the general structure of the layers is as follows.

Convolution Layer

We start with the Convolutional Layer which is used to recognize patterns of the image. We are going to use a 2D convolution layer since our images are converted in grayscale.

The parameters needed here are the number of filters/kernels, the size of the kernel, the padding to decide if we should add or not padding depending on the dataset (how important are the patterns in the borders and how we want the dimensions of the image) and input shape which are the dimensions of our images.

model.add(Conv2D(filters, (kernel_size), padding="same", input_shape = inputShape)

Activation Layer

After the Convolution Layer we continue with the Activation Layer by applying a nonlinear activation function. Even though this is technically not considered as a layer since it doesn't have weights or parameters, it is part of the architecture of the CNN.

model.add(Activation('relu'))

Batch Normalization

After Activation we add Batch Normalization in order to normalize the activations of the volume of the inputs that were given before going to the upcoming layer at the network.

model.add(BatchNormalization())

Pooling Layer

In order to reduce the size of the input image, except Convolution Layer, we use a Pooling Layer. By reducing the size, we reduce the computations and the number of parameters. We use MaxPooling2D since our images are 2 dimensional.

model.add(MaxPooling2D(pool_size=(2, 2)))

Dropout Layer

We can consider Dropout as a method to regularize which helps in preventing overfitting. As a parameter we add the frequency of the rate.

model.add(Dropout(0.2))

Before adding the last layer we have to convert our 2D features maps to 1D feature vectors.

model.add(Flatten())

Fully Connected Layers

The last layer of the CNN is the fully connected Layer. As parameters we add the units and the activation function.

model.add(Dense(units = 512, activation='relu'))

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1. Dataset

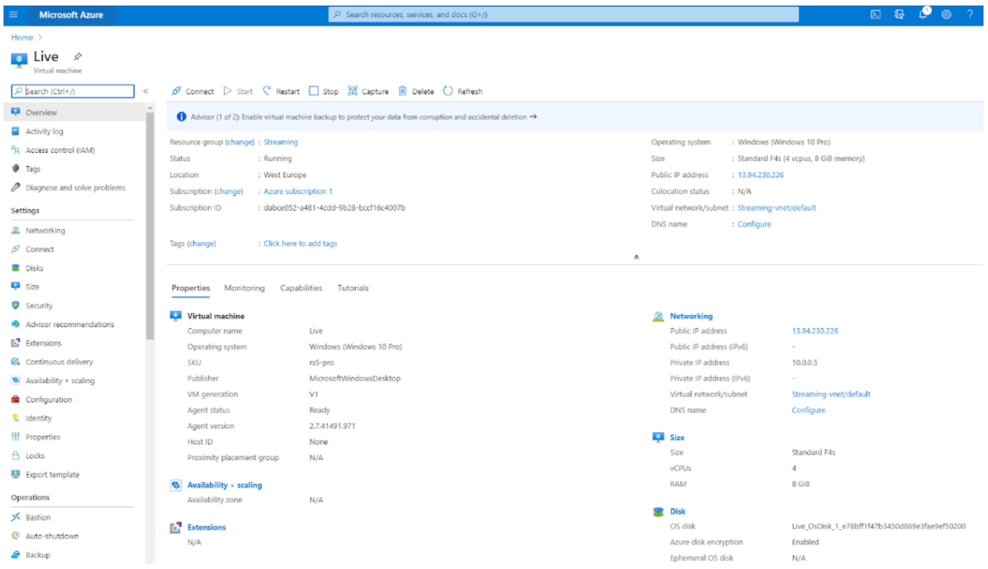
Renting a Virtual Machine to do the heavy network processes.

Dealing with a high volume of data is very difficult. The main problem lies with the internet speed. My actual internet speed is 15 Mbps in download and about 4 Mbps in upload. That means that to download the 74 GB of data I need about 49300 seconds which is equivalent to 14 hours.

Still this is only the beginning because to actually use the data in Google Colab for example, I would be needing about 40 minutes per gigabyte. Just to use an online batch image cropping software it would need at least ten hours per 15 GB without counting the download and processing and would take weeks to finish all the cropping.

In terms of time this is unacceptable and I needed to find a viable solution. Also, I needed to use a second computer in order to let it run certain tasks. The solution I found was to use a Virtual Machine offered from Microsoft Azure Portal. I signed up for a Virtual Machine which used about 0.7 dollars per hour. Even though it is a bit expensive because I chose Windows 10 Pro as the main Operating System for it, I needed more control over the VM and that’s why that was a compromise I had to make.

I created the instance and I got assigned a public IP address which I used to connect through RDP (Remote Desktop Protocol).



Advisor (1 of 2): Enable virtual machine backup to protect your data from corruption and accidental deletion →

Resource group (change)	: Streaming
Status	: Running
Location	: West Europe
Subscription (change)	: Azure subscription 1
Subscription ID	: dabce052-a481-4cdd-9b28-bccf16c4007b
Tags (change)	: Click here to add tags

Operating system	: Windows (Windows 10 Pro)
Size	: Standard F4s (4 vcpus, 8 GiB memory)
Public IP address	: 13.94.230.226
Colocation status	: N/A
Virtual network/subnet	: Streaming-vnet/default
DNS name	: Configure

Virtual machine

Computer name	Live
Operating system	Windows (Windows 10 Pro)
SKU	rs5-pro
Publisher	MicrosoftWindowsDesktop
VM generation	V1
Agent status	Ready
Agent version	2.7.41491.971
Host ID	None
Proximity placement group	N/A

Availability + scaling

Availability zone	N/A

Extensions

N/A

Networking

Public IP address	13.94.230.226
Public IP address (IPv6)	-
Private IP address	10.0.0.5
Private IP address (IPv6)	-
Virtual network/subnet	Streaming-vnet/default
DNS name	Configure

Size

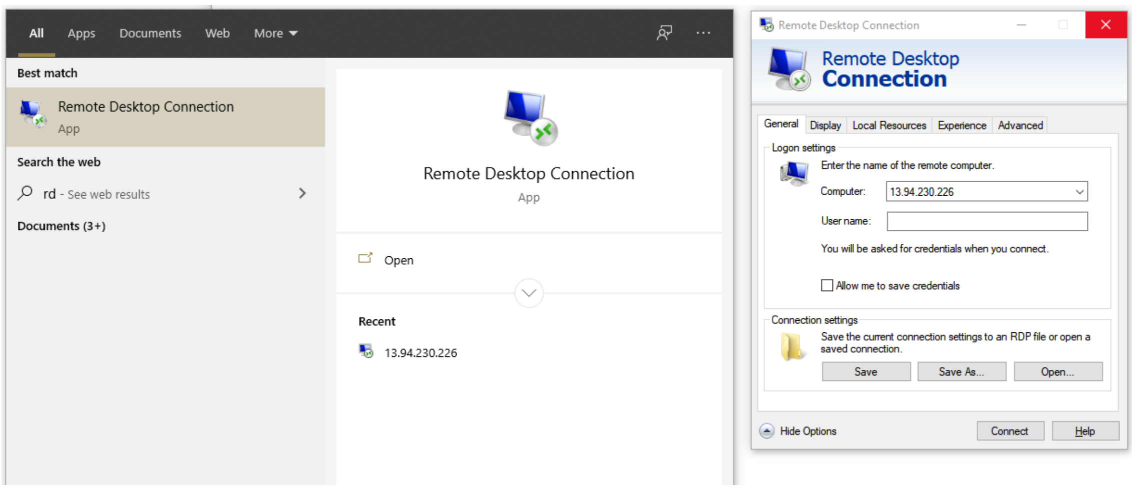
Size	Standard F4s
vCPUs	4
RAM	8 GiB

Disk

OS disk	Live_OsDisk_1_e78bff1f47b3450d869e3fae9ef50200
Azure disk encryption	Enabled
Ephemeral OS disk	N/A

Figure 29. Microsoft Azure Portal Virtual Machine Information

That's where I did most of my downloading/uploading and processing certain elements of my work. Connecting through RDP is relatively easy and very convenient:



Enter the name of the remote computer.

Computer: 13.94.230.226

User name:

You will be asked for credentials when you connect.

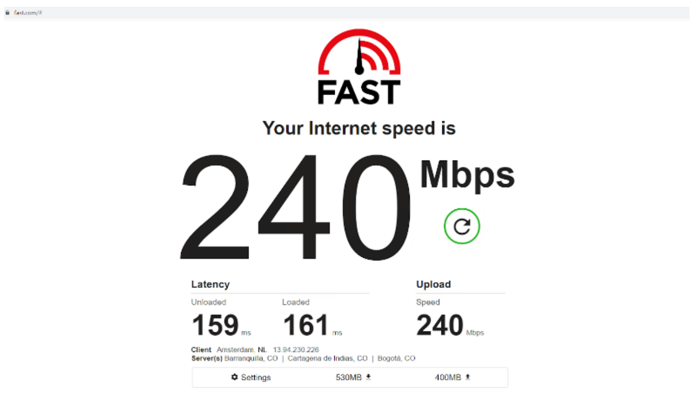
☐ Allow me to save credentials

Connection settings

Save the current connection settings to an RDP file or open a saved connection.

Figure 30. Microsoft Azure Portal Virtual Machine Information

The download and upload speed are quite good and the VM instance proved itself to be very effective since the download/upload is symmetrical.


FAST

Your Internet speed is

240 Mbps

Latency

Unloaded 159 ms

Loaded 161 ms

Upload

Speed 240 Mbps

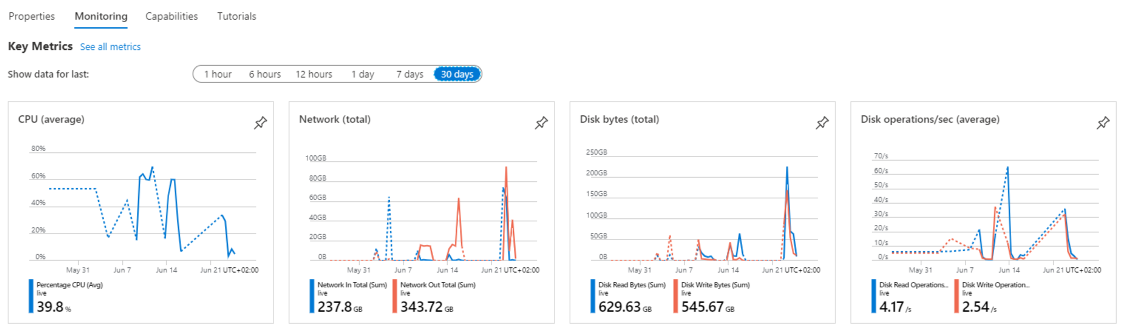
Client Amsterdam, NL 13.94.230.226

Server(s) Barranquilla, CO | Cartagena de Indias, CO | Bogotá, CO

Settings 530MB 400MB

Figure 31. VM instance Download/Upload Speed

Below you will find a screenshot of the Network Bandwidth used in the last 30 days. I have used about 238GB in Download and 344GB in Upload. In terms of efficiency a VM instance is very helpful.

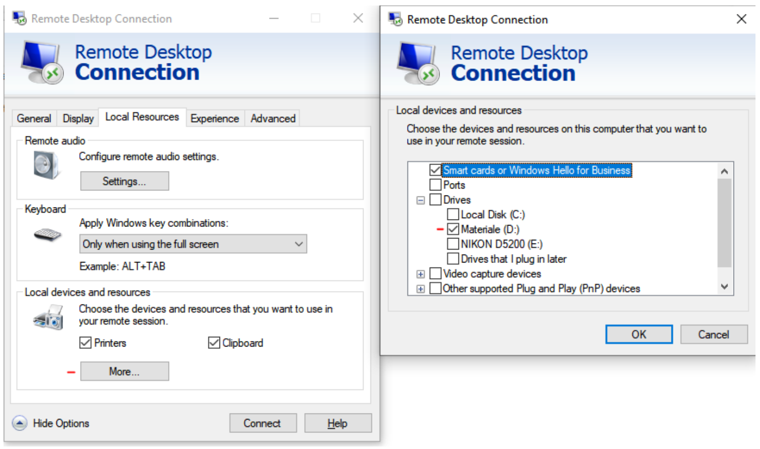


Show data for last: 1 hour 6 hours 12 hours 1 day 7 days [✓ 30 days]

Percentage CPU (Avg) live	39.8 %
Network In Total (Sum) live	237.8 GB
Network Out Total (Sum) live	343.72 GB
Disk Read Bytes (Sum) live	629.63 GB
Disk Write Bytes (Sum) live	545.67 GB
Disk Read Operations... live	4.17 /s
Disk Write Operation... live	2.54 /s

Figure 32. Monitoring Virtual Machine Data Network and Disk Operations

The difficulty stands between moving the data back and forth from my actual computer to the Virtual Machine. In fact, the Remote Desktop Protocol is very helpful in that matter because it allows you to mount a local drive into the VM. Then the process is basic copy pasting. However, this is not a very fast solution because RDP is not actually designed to transfer files, but to be more efficient I transfered .zip or .rar files (so I wouldn't be limited by the files' individual size) through the night.


Remote Desktop Connection

General | Display | Local Resources | Experience | Advanced

Remote audio
Configure remote audio settings.
Settings...

Keyboard
Apply Windows key combinations:
Only when using the full screen
Example: ALT+TAB

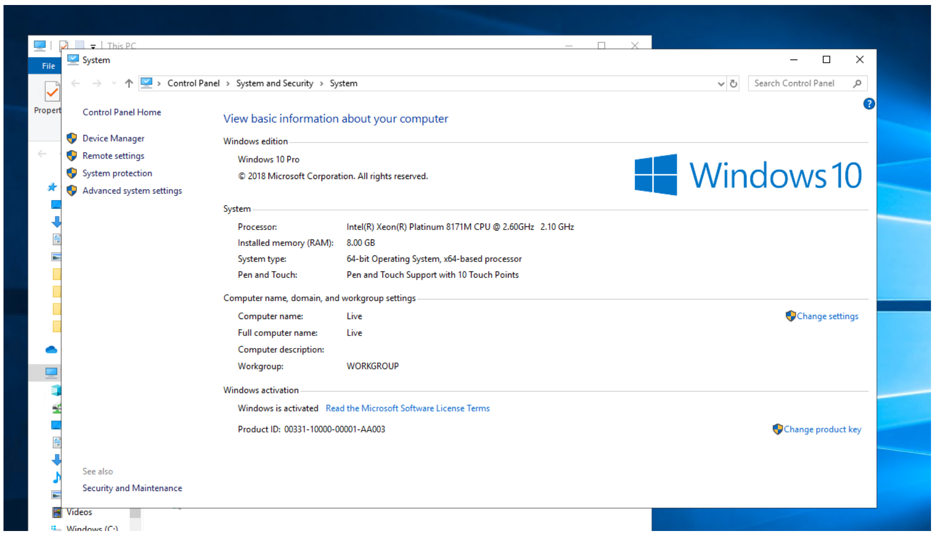
Local devices and resources
Choose the devices and resources that you want to use in your remote session.
☑ Printers ☑ Clipboard
More...

Hide Options | Connect | Help

Local devices and resources
Choose the devices and resources on this computer that you want to use in your remote session.
☑ Smart cards or Windows Hello for Business
☐ Ports
☐ Drives
☐ Local Disk (C:)
☑ Materiale (D:)
☐ NIKON D5200 (E:)
☐ Drives that I plug in later
☐ Video capture devices
☐ Other supported Plug and Play (PnP) devices
OK | Cancel

Figure 33. Connecting the VM with the Laptop Local Drive through RDP

I understood from this process that the machine I have “rented” online, is not one of the best machines on the market, since I don’t have a GPU, the virtual processors are limited, the RAM is limited and the storage is an extra cost we have to deal with. Still, it was the best solution I could find because it involved very fast and reliable internet speed without interruption. This came in handy when I cropped the images manually one by one, but didn’t help me when I tried to load the heavy python processing. Below is a screenshot of the Virtual Machine specs.


Control Panel > System and Security > System

View basic information about your computer

Windows edition
Windows 10 Pro
© 2018 Microsoft Corporation. All rights reserved.

System
Processor: Intel(R) Xeon(R) Platinum 8171M CPU @ 2.60GHz 2.10 GHz
Installed memory (RAM): 8.00 GB
System type: 64-bit Operating System, x64-based processor
Pen and Touch: Pen and Touch Support with 10 Touch Points

Computer name, domain, and workgroup settings
Computer name: Live
Full computer name: Live
Computer description:
Workgroup: WORKGROUP

Windows activation
Windows is activated Read the Microsoft Software License Terms
Product ID: 00331-10000-00001-AA003

Figure 34. Virtual Machine specifications

Dealing with image extracting, cropping and resizing the images.

One of the most challenging tasks of the work I intended to do was to unzip the files in the Virtual Machine, copying a part of it into my laptop and practically working simultaneously in two different devices.

First, I tried to unzip the big 74GB files I downloaded from the iCV Research Lab. The unzipping process took a significant amount of time in the Virtual Machine which proved once again that it is not wise to work heavy processing stuff on it since the CPU is not one of the best on the market. After that, I had to send a copy to my computer through RDP. That also was very time consuming. During this process I had constant freezing and stalling of both of the devices.

After having the files copied on my computer and on my virtual machine, I could start the first step into processing the images, cropping them.

To crop the images into the faces subject was difficult to do manually since I had to find the right software which would do batch processing. After completing my research, I decided to upload the files in an online web application which let me decide the image dimensions and the parts to include or exclude from the image.

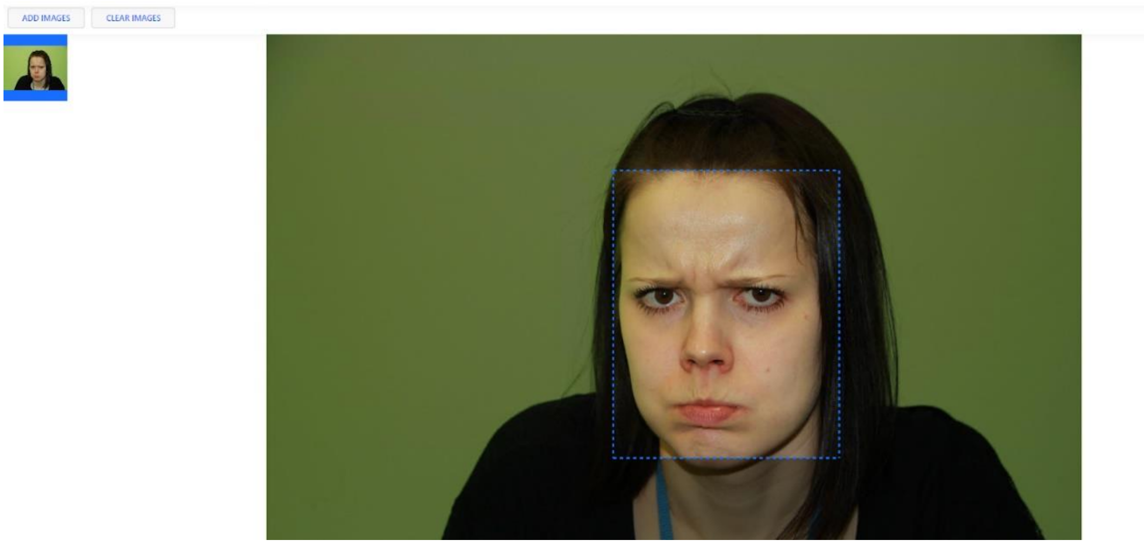
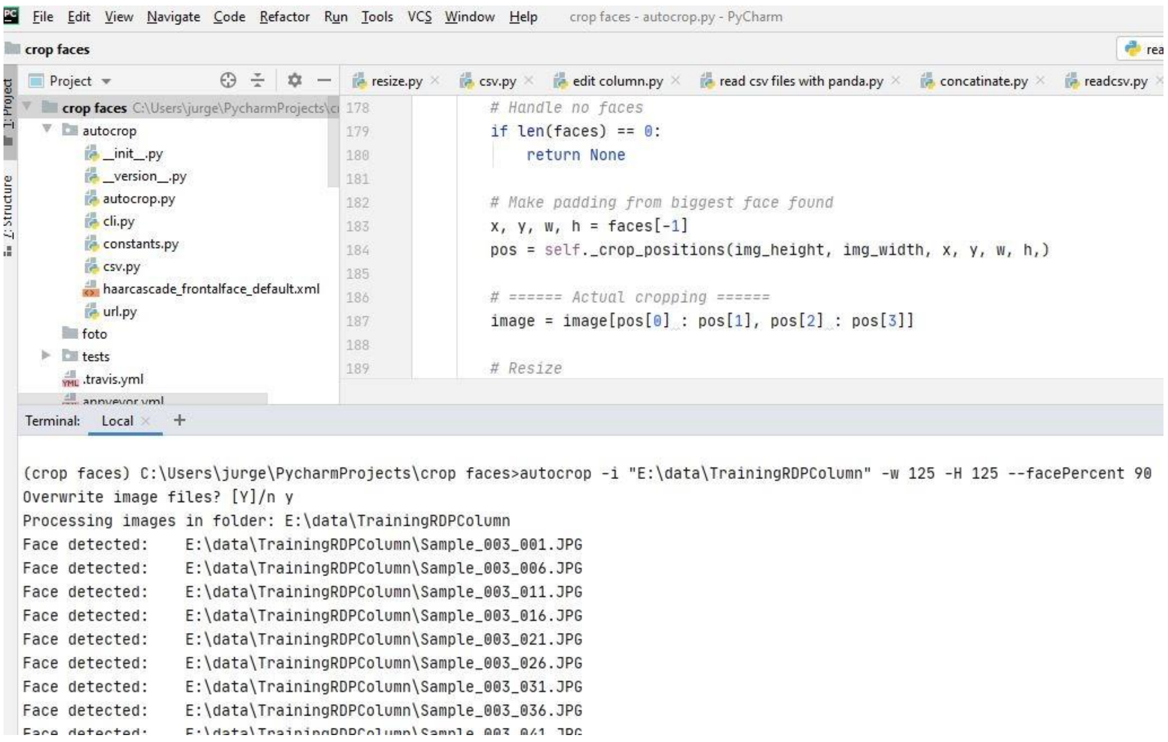


Figure 35. Process of Manually Cropping the images

This process was done completely on Virtual Machine and it was the most consuming workload in time perspective. After finishing the manually cropped images, I was thinking of automatically cropped ones in order to make the difference between the two. To batch crop all the images in the same size I had to map the image and find out where were the coordinates I had to work with. The problem here was that not every subject has posed in the same position in front of the camera. Some of the subjects have their heads turned, some others are too close to the camera and some others are not in the center of the image. That meant that the process was becoming more of a manually job than an automatic one.

I decided to do proper research on face recognition and I was convinced to use the Viola-Jones face recognition algorithm in order to detect and crop any of the subject faces by inputting the arguments of face percentage in photo and the same width and height I used on the manually cropped images. Below I am showing a piece of the code I used to complete this process


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crop faces - autocrop.py - PyCharm
crop faces
Project
resize.py
csv.py
edit column.py
read csv files with panda.py
concatinate.py
readcsv.py
crop faces C:\Users\jurge\PycharmProjects\c
autocrop
init.py
version.py
autocrop.py
cli.py
constants.py
csv.py
haarcascade_frontalface_default.xml
url.py
foto
tests
.travis.yml
appveyor.yml
Terminal: Local

```
        # Handle no faces
        if len(faces) == 0:
            return None

        # Make padding from biggest face found
        x, y, w, h = faces[-1]
        pos = self._crop_positions(img_height, img_width, x, y, w, h,)

        # ====== Actual cropping ======
        image = image[pos[0] : pos[1], pos[2] : pos[3]]

        # Resize
```

```
(crop faces) C:\Users\jurge\PycharmProjects\crop faces>autocrop -i "E:\data\TrainingRDPColumn" -w 125 -H 125 --facePercent 90
Overwrite image files? [Y]/n y
Processing images in folder: E:\data\TrainingRDPColumn
Face detected:    E:\data\TrainingRDPColumn\Sample_003_001.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_006.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_011.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_016.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_021.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_026.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_031.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_036.JPG
Face detected:    E:\data\TrainingRDPColumn\Sample_003_041.JPG
```

Figure 36. Process of Automatically Cropping the images

The algorithm was very quick to detect and crop the image faces.

However, here I had another problem too. First, every original image used by the Viola Jones Algorithm was directly manipulated which led me to a more time-consuming task to transfer again the photos from the Virtual Machine since I had to do many tests. This algorithm couldn’t create new images but instead cropped the original ones. Secondly, after manually reviewing the algorithm work on the subjects faces,

We found out that some of the cropped images had problems with the face detection. Mostly this problem resulted from the subjects with a noticeable Adam's apple which the algorithm mistook for a face. Since I was batch processing 3196 photos at the same time, I decided it would be wise to try one more time just to automatically detect and crop these images again. Unsurprisingly, the result was the same. That means that this algorithm has a huge flaw.

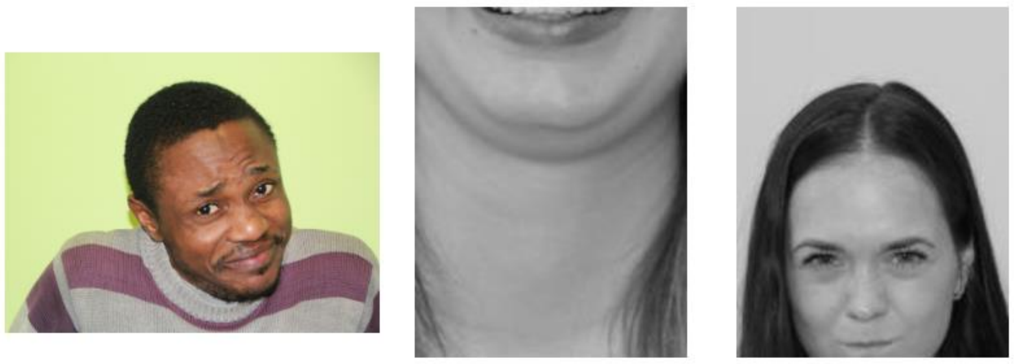


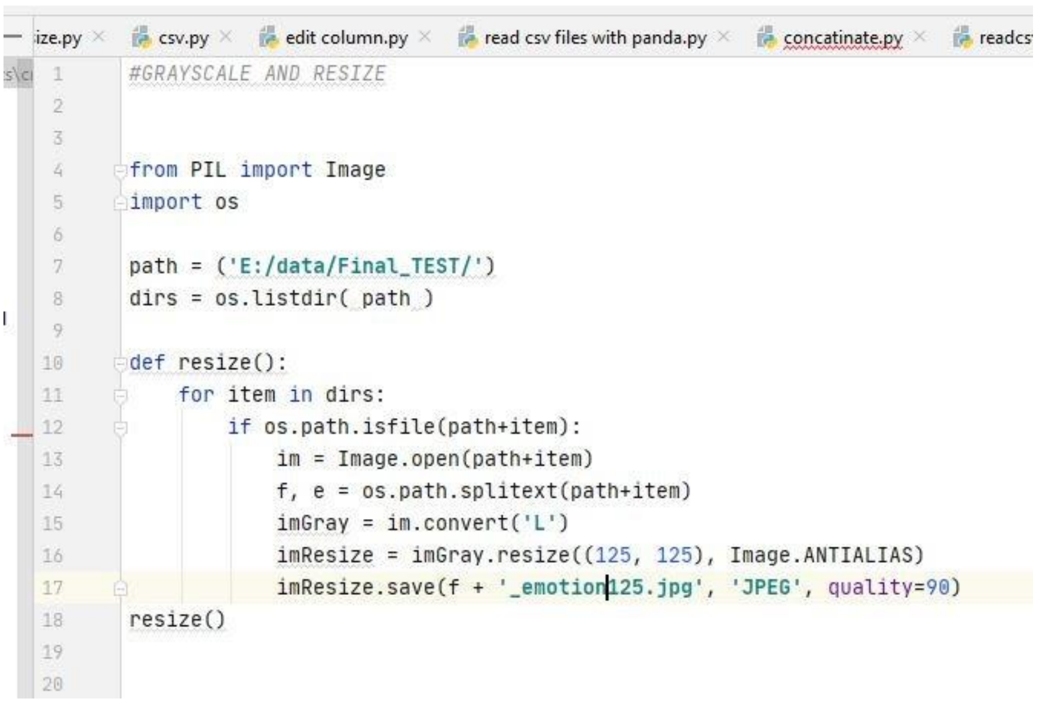
Figure 37. Problems with the face detection and cropping using Viola-Jones

We had to choose between fixing or not the badly cropped images.

We decided not to fix them, because if the batch cropping process would be at least ten to a hundred times bigger than 3196 images, it would be impossible to manually review the results. That means that the datasets which have used this algorithm have flaws in it. By calculating the flawed images, we could see that the percentage of this error is about 0.2% of the total number of images.

The real difference between manually cropped and automatically cropped ones, is exactly the process of errors made while cropping by both sides. We will try to see if the flaws in automatic face detection have a real impact in the results or not.

The next step in line was to resize the images. I could do this process automatically with few lines of code in Python.



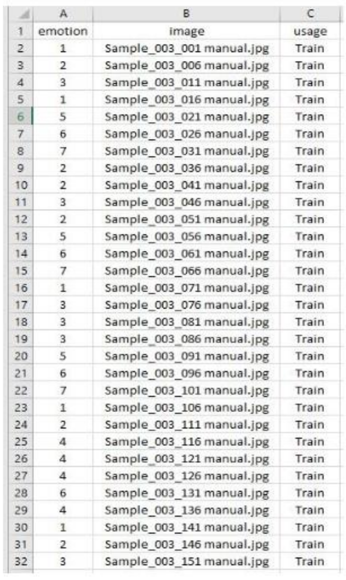
```
#GRAYSCALE AND RESIZE


from PIL import Image
import os

path = ('E:/data/Final_TEST/')
dirs = os.listdir( path )

def resize():
    for item in dirs:
        if os.path.isfile(path+item):
            im = Image.open(path+item)
            f, e = os.path.splitext(path+item)
            imGray = im.convert('L')
            imResize = imGray.resize((125, 125), Image.ANTIALIAS)
            imResize.save(f + '_emotion125.jpg', 'JPEG', quality=90)
resize()
```

Figure 38. Resizing and converting images to grayscale (code sample)

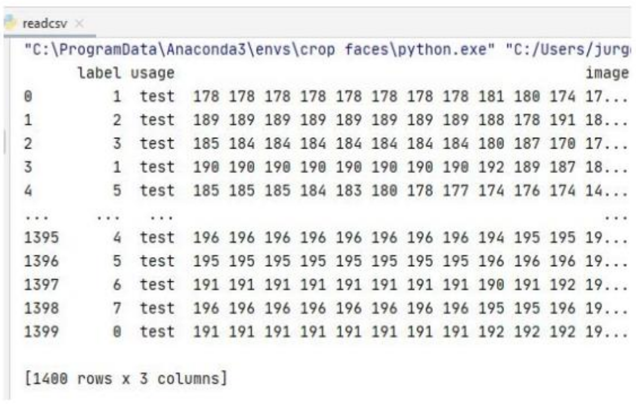
I researched the best image size to train the neural network and I decided to use these dimensions: 140px x 180px. I chose not to have the usual square image because the face's height is usually bigger than the face width, even though that isn't the best aspect ratio to train the algorithm.

Creating .csv files of manually labelled images

One of the key parts of the thesis, was to label the seven basic emotions of the subjects images manually. By default, this dataset had labels of two mixed emotions, for example, sadly surprised or surprisingly sad. The whole process for me was to select the dominating emotion of the faces of the subjects. This was one of the most challenging tasks because I had to carefully review the images in order to train the algorithm in the same way I view them. Also it was a great way to understand the background of training the neural network by finishing all the parts of preparing it by myself.



	A	B	C
1	emotion	image	usage
2	1	Sample_003_001 manual.jpg	Train
3	2	Sample_003_006 manual.jpg	Train
4	3	Sample_003_011 manual.jpg	Train
5	1	Sample_003_016 manual.jpg	Train
6	5	Sample_003_021 manual.jpg	Train
7	6	Sample_003_026 manual.jpg	Train
8	7	Sample_003_031 manual.jpg	Train
9	2	Sample_003_036 manual.jpg	Train
10	2	Sample_003_041 manual.jpg	Train
11	3	Sample_003_046 manual.jpg	Train
12	2	Sample_003_051 manual.jpg	Train
13	5	Sample_003_056 manual.jpg	Train
14	6	Sample_003_061 manual.jpg	Train
15	7	Sample_003_066 manual.jpg	Train
16	1	Sample_003_071 manual.jpg	Train
17	3	Sample_003_076 manual.jpg	Train
18	3	Sample_003_081 manual.jpg	Train
19	3	Sample_003_086 manual.jpg	Train
20	5	Sample_003_091 manual.jpg	Train
21	6	Sample_003_096 manual.jpg	Train
22	7	Sample_003_101 manual.jpg	Train
23	1	Sample_003_106 manual.jpg	Train
24	2	Sample_003_111 manual.jpg	Train
25	4	Sample_003_116 manual.jpg	Train
26	4	Sample_003_121 manual.jpg	Train
27	4	Sample_003_126 manual.jpg	Train
28	6	Sample_003_131 manual.jpg	Train
29	4	Sample_003_136 manual.jpg	Train
30	1	Sample_003_141 manual.jpg	Train
31	2	Sample_003_146 manual.jpg	Train
32	3	Sample_003_151 manual.jpg	Train



```
"C:\ProgramData\Anaconda3\envs\crop faces\python.exe" "C:/Users/jurg
      label usage                                              image
0         1  test  178 178 178 178 178 178 178 178 181 180 174 17...
1         2  test  189 189 189 189 189 189 189 189 188 178 191 18...
2         3  test  185 184 184 184 184 184 184 184 180 187 170 17...
3         1  test  190 190 190 190 190 190 190 190 192 189 187 18...
4         5  test  185 185 185 184 183 180 178 177 174 176 174 14...
...     ...   ...                                                ...
1395      4  test  196 196 196 196 196 196 196 196 194 195 195 19...
1396      5  test  195 195 195 195 195 195 195 195 196 196 196 19...
1397      6  test  191 191 191 191 191 191 191 191 190 191 192 19...
1398      7  test  196 196 196 196 196 196 196 196 195 195 196 19...
1399      0  test  191 191 191 191 191 191 191 191 192 192 192 19...

[1400 rows x 3 columns]
```

Figure 39. Labelled images in .csv format in two methods

After labelling the images in a separate .csv file, I had to create another .csv with images data on it. Compiling the file was not difficult, but dealing with it was more challenging than the work I had done so far.

The generated .csv was a hefty 300MB file with 25200 columns which was extremely difficult to manipulate through excel. The problem was that I had to put the labels in the first column. Excel takes lots of time just to open the file and lags while viewing portions of it.

Another problem is that excel can only view up to 16384, while the file generated had 25200 columns. So excel actually truncated 28 million values. I tried different approaches to edit the .csv through different softwares like EmEditor, TadEditor or CSVed but each one of them was unsuccessful. During each run, the file size was a great problem.

I decided to redo the whole process of image resizing one more time to a more reasonable and acceptable size. The new size I chose is 125x125. At this point I still had difficulties managing the data with Excel, since every .csv is stored with columns and they cannot be merged. The solution was to manually do everything. At first, we saved the .csv file from Excel as Text file with \t.

Then, I tried to open the .txt file in Python using pandas with the following code:

```
import pandas

df = pandas.read_csv(r, ‘source/file.txt/’, index=None)

print (df)
```

I could see that the numbers were in one column but in the middle of them was the tab escape sequence e.g: *“320\t21\t31\t”*.

We added a line of code to convert .txt to .csv plus adding a header using:

```
df.to_csv(‘./file.csv/, header=[‘images’])
```

After that we tried to find and replace every escape sequence with a space between. By using only Python capability and no libraries:

```
text = open("input.csv", "r")

text = ''.join([i for i in text]) \

    .replace("\t", " ")

x = open("output.csv","w")

x.writelines(text)

x.close()
```

Now we had a full .csv file in a row as I needed it and a header. Earlier I created the labels in another .csv file in Excel with a header and the process now was to concatenate the two .csv files together. I used the simple concat function in pandas:

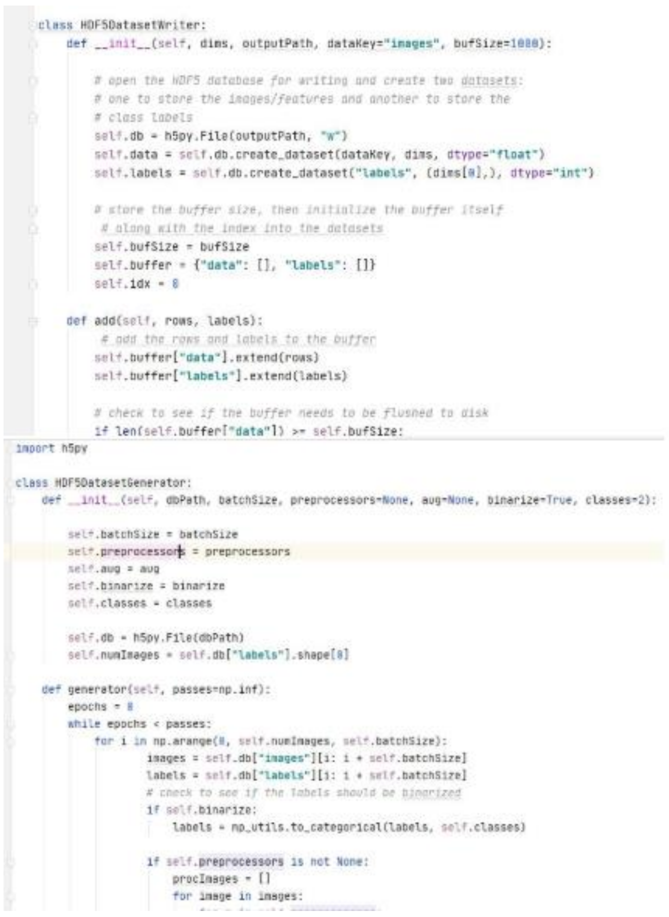
```
result = pd.concat([labels, images], axis=1, sort=False)
```

Right now I had all the files of the training ready to go. The problem lies with the testing images which should be done the same way and should have the same headers as the .csv file we created for training.

I repeated the process for the images to be the same size, to be saved as .csv with the header named images. I created the labels header with the same header I did in the training process.

Finally, we have a full .csv file around 270MB which is very big to open and has 3 columns with 3194+1400 rows altogether.

After the labeling step finished and I will be converting the file in hdf5.



```python
class HDF5DatasetWriter:
    def __init__(self, dims, outputPath, dataKey="images", bufSize=1000):

        # open the HDF5 database for writing and create two datasets:
        # one to store the images/features and another to store the
        # class labels
        self.db = h5py.File(outputPath, "w")
        self.data = self.db.create_dataset(dataKey, dims, dtype="float")
        self.labels = self.db.create_dataset("labels", (dims[0],), dtype="int")

        # store the buffer size, then initialize the buffer itself
         # along with the index into the datasets
        self.bufSize = bufSize
        self.buffer = {"data": [], "labels": []}
        self.idx = 0

    def add(self, rows, labels):
         # add the rows and labels to the buffer
        self.buffer["data"].extend(rows)
        self.buffer["labels"].extend(labels)

        # check to see if the buffer needs to be flushed to disk
        if len(self.buffer["data"]) >= self.bufSize:
```

```python
import h5py

class HDF5DatasetGenerator:
    def __init__(self, dbPath, batchSize, preprocessors=None, aug=None, binarize=True, classes=2):

        self.batchSize = batchSize
        self.preprocessors = preprocessors
        self.aug = aug
        self.binarize = binarize
        self.classes = classes

        self.db = h5py.File(dbPath)
        self.numImages = self.db["labels"].shape[0]

    def generator(self, passes=np.inf):
        epochs = 0
        while epochs < passes:
            for i in np.arange(0, self.numImages, self.batchSize):
                    images = self.db["images"][i: i + self.batchSize]
                    labels = self.db["labels"][i: i + self.batchSize]
                    # check to see if the labels should be binarized
                    if self.binarize:
                        labels = np_utils.to_categorical(labels, self.classes)

                    if self.preprocessors is not None:
                        procImages = []
                        for image in images:
```

Figure 40. Part of the code for “class HDF5DatasetWriter” and class HDF5DatasetGenerator

After experimenting on many distributions of the data, since we relabeled images many times, the final distribution of data is equal. We have 250 images per emotion.

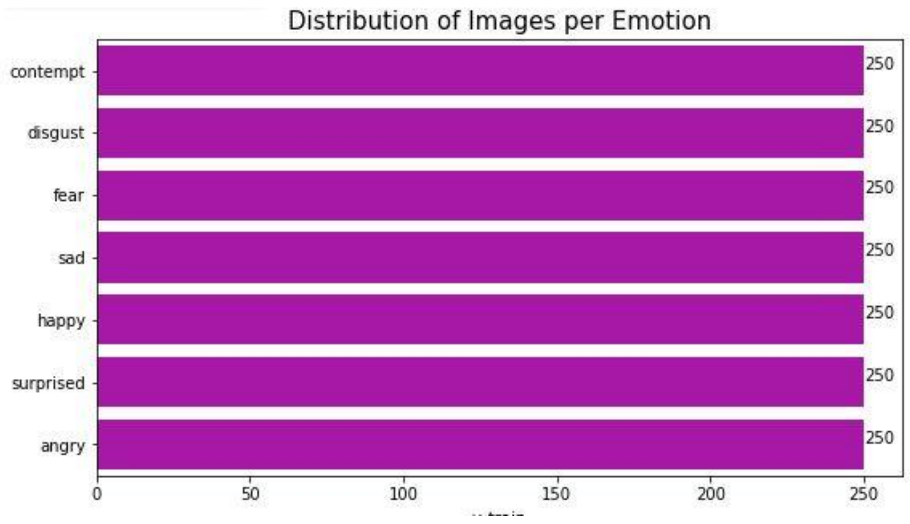
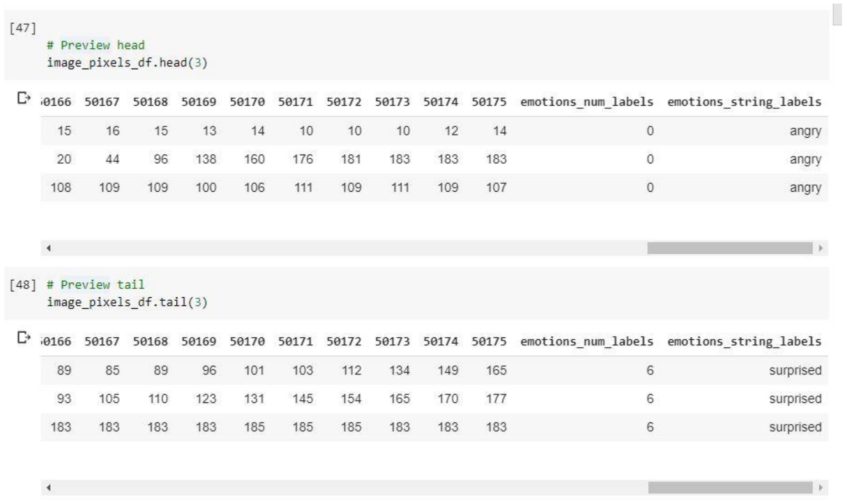


Figure 41. Figure Distribution of Images per Emotion

We have tried many image sizes, but they didn’t have any significant improviment in the data which is worth mantionin. Here we will show the output only for 110x75 and 224 x 224 sizes.



```
[47]
     # Preview head
     image_pixels_df.head(3)
```

0166	50167	50168	50169	50170	50171	50172	50173	50174	50175	emotions_num_labels	emotions_string_labels
15	16	15	13	14	10	10	10	12	14	0	angry
20	44	96	138	160	176	181	183	183	183	0	angry
108	109	109	100	106	111	109	111	109	107	0	angry

```
[48] # Preview tail
     image_pixels_df.tail(3)
```

0166	50167	50168	50169	50170	50171	50172	50173	50174	50175	emotions_num_labels	emotions_string_labels
89	85	89	96	101	103	112	134	149	165	6	surprised
93	105	110	123	131	145	154	165	170	177	6	surprised
183	183	183	183	185	185	185	183	183	183	6	surprised

Figure 42. Figure Represantation of the data

Firstly, we separated our images in folders and then converted to csv as it is shown in the figure above.

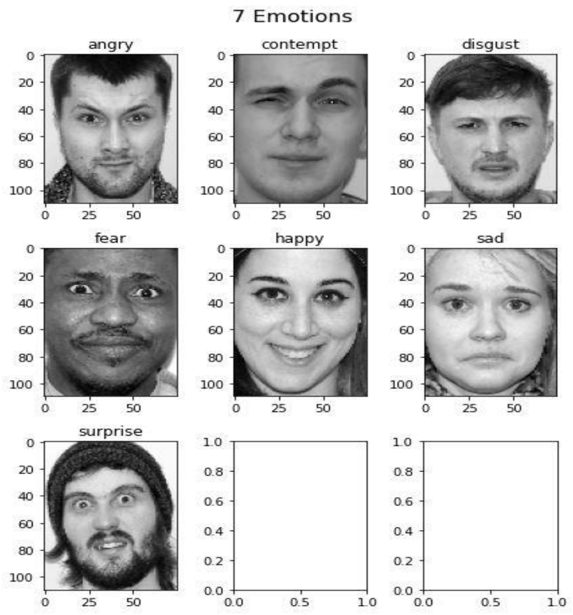
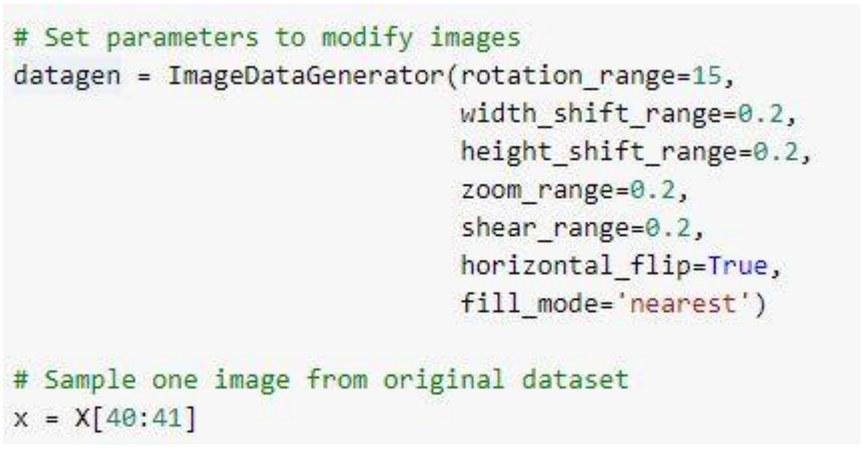


Figure 43. Represantation of Labeled images

When we run the model for the second time, we do data augmentation as a common technique to increase the data, to improve results and avoid overfitting.



```
# Set parameters to modify images
datagen = ImageDataGenerator(rotation_range=15,
                             width_shift_range=0.2,
                             height_shift_range=0.2,
                             zoom_range=0.2,
                             shear_range=0.2,
                             horizontal_flip=True,
                             fill_mode='nearest')

# Sample one image from original dataset
x = X[40:41]
```

Figure 44. Data Augmentation

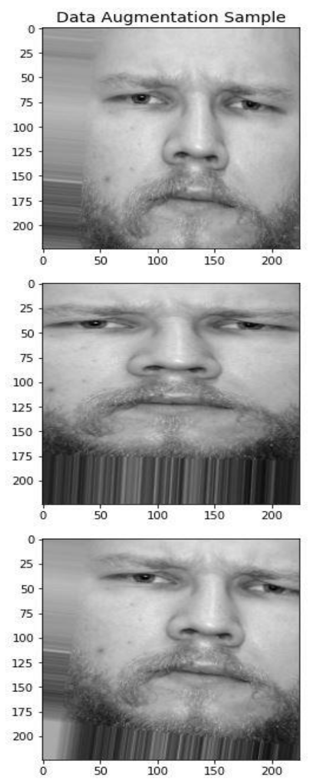


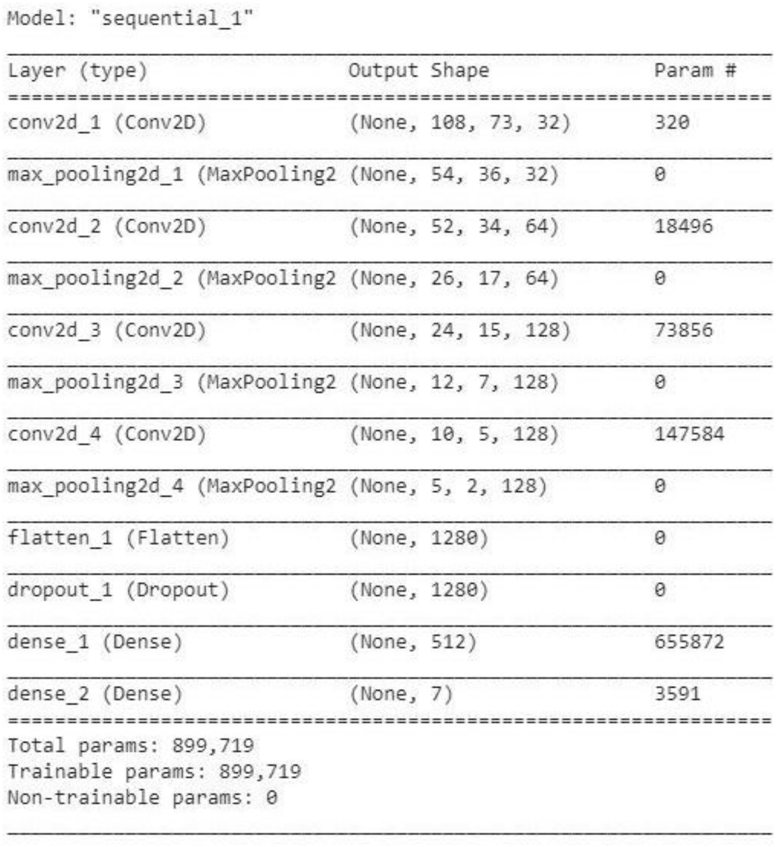
Figure 45. Sample of Images after data augmentation

5.2. CNN Model

First Experiment

After preprocessing our we constructed the model of our CNN. This dataset were very challenging for me, because even after many experiments, we couldn’t get very good results.

After many experiments when we changed parameters and added more layers, these are the results:



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 108, 73, 32)       320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 36, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 52, 34, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 17, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 15, 128)       73856
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 12, 7, 128)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 10, 5, 128)        147584
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 2, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1280)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 1280)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               655872
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 3591
=================================================================
Total params: 899,719
Trainable params: 899,719
Non-trainable params: 0
_________________________________________________________________
```

Figure 46. Results when we added more layers

Our network has 4 Convolutional Layers, 4 MaxPooling, One dropout and 2 Fully Connected Layers. In total the model has 899718 parameters.

So, an image (the input) is being passed through the model, and a filter (Conv2D) passing through the image and taking the dot product of the pixel values and applying relu (Rectified Linear Unit) to get a number between 0 and infinity. The pooling layer (MaxPooling2D) takes the maximum values from the convolution layer and reduces the dimensions of the image (and also helps speed up processing time). The flattened layer (Flatten()) takes the image pixels in the previous layer and flattens it out (5 x 2 x 128 = 1280).

The Dropout layer randomly ignores certain neurons and helps prevent overfitting. The densely connected layer (Dense) is connected to every node in the previous layer and applies ReLu to its output values. Finally, the output layer (also Dense) applies the softmax activation function to show probabilities for all nine emotions, and all of the probabilities will add up to 1.

Below is shown the result of training the model with our dataset with 39 epochs and batch size 64.

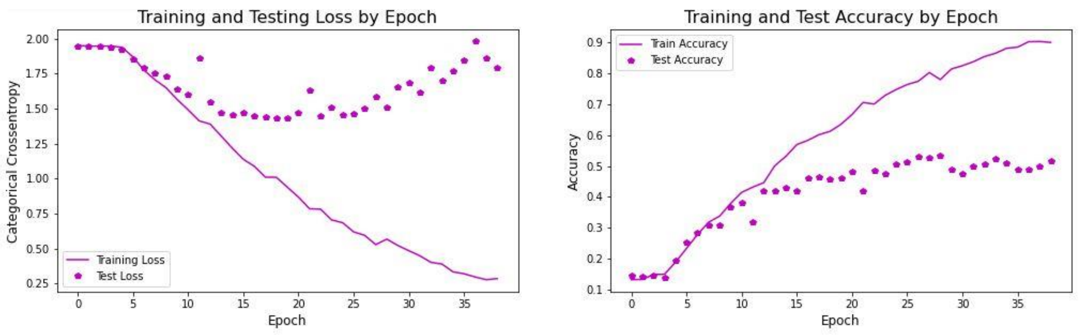


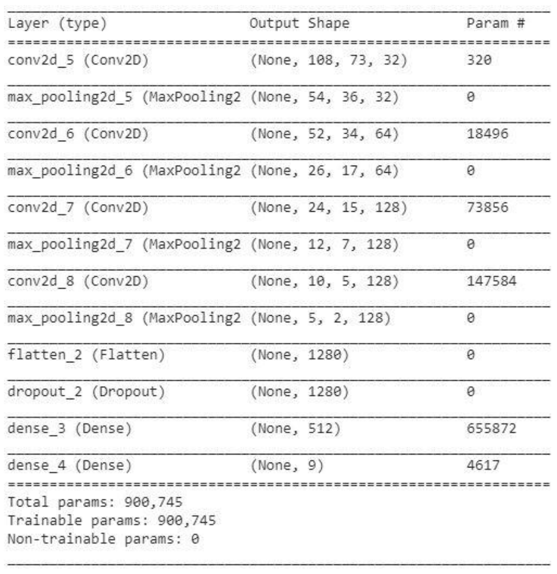
Figure 47. Model 1 Evaluation

CNN Model 1 is overfit, since the training data performs better than the testing data. We notice that after epoch 25 the accuracy score is higher for my training data, meaning my model makes better predictions on my training data than my testing data. And the loss values are lower for my training data, meaning that my model has less errors in my training data than my testing data. However, this all actually makes sense considering that I'm training a CNN model on a small, un-augmented dataset.

What we will do in the next model is augment my data and see if that makes the model more fit. Augmenting my data creates altered versions of the same image and introduces those "new" images to the model, making it seem as if more data has been added to my dataset.

Second Experiment

In the second experiment we use the same model as in the first but here we add data augmentation in order to have more data and to improve accuracy.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 108, 73, 32)       320
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 54, 36, 32)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 52, 34, 64)        18496
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 26, 17, 64)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 24, 15, 128)       73856
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 12, 7, 128)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 10, 5, 128)        147584
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 5, 2, 128)         0
_________________________________________________________________
flatten_2 (Flatten)          (None, 1280)              0
_________________________________________________________________
dropout_2 (Dropout)          (None, 1280)              0
_________________________________________________________________
dense_3 (Dense)              (None, 512)               655872
_________________________________________________________________
dense_4 (Dense)              (None, 9)                 4617
=================================================================
Total params: 900,745
Trainable params: 900,745
Non-trainable params: 0
_________________________________________________________________
```

Figure 48. Second experiment with added data augmentation

Since we have more images the parameters are higher than the first model, here we have 900,745 parameters.

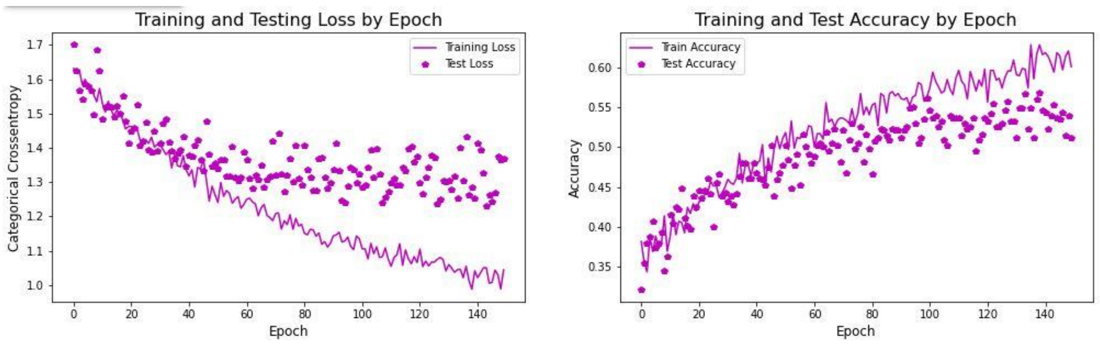


Figure 49. Training and Testing Loss and Accuracy

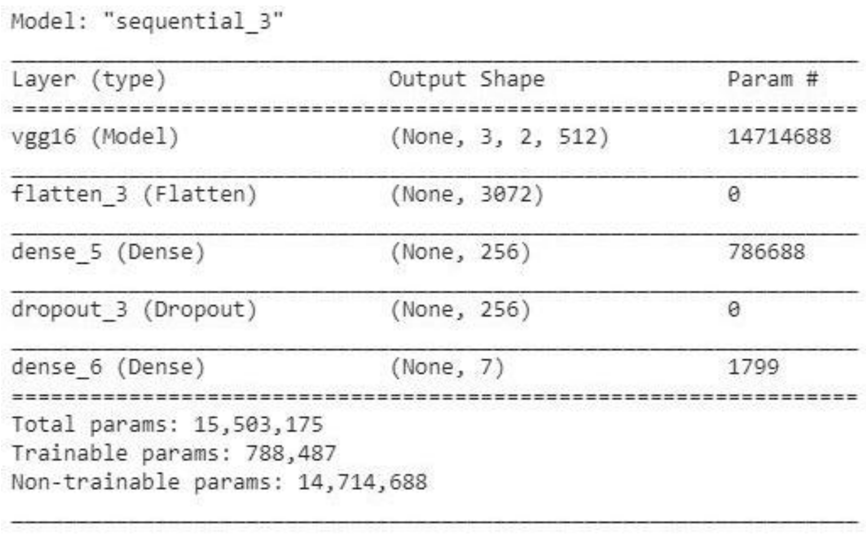
Data augmentation through ImageDataGenerator seems looks like improved the results a little since we do not have that much overfitting even though there is a tendency after epoch 60.

Third Experiment

In the third experiment we used a PreTrained network, which is VGG16. Since our dataset is not big enough, usally using pre trained model is recommended. According to Chollet, "a pretrained network is a saved network that was previously trained on a large dataset, typically on a large-scale image-classification task." We'll be using the VGG16 pre-trained model because it's simple, widely-used, and is also based on image classification.

In the cnn_model_3.summary() above, there are seven neurons in my output layer, dense_6 (Dense). In the Param # column, the first row corresponds to VGG16's model, which has 14.7 million parameters.

Freezing a layer or set of layers means preventing their weights from being updated during training. If we didn’t do this, then the representations that were previously learned by the convolutional base will be modified during training. Because the Dense layers on top are randomly initialized, very large weight updates would be propagated through the network, effectively destroying the representations previously learned. That why we left only 4 trainable wights after freezing.

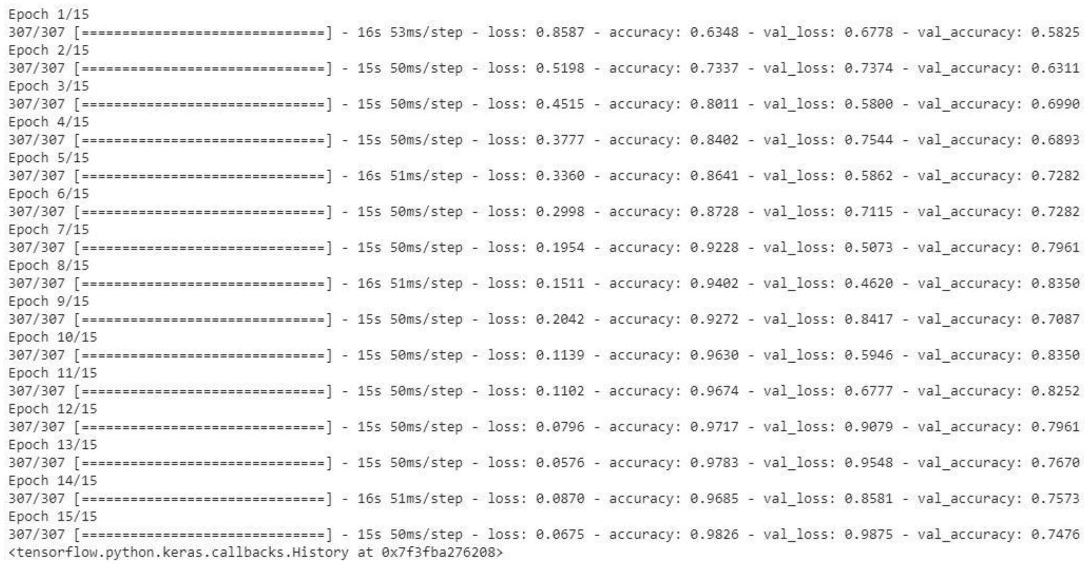


```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 3, 2, 512)         14714688
_________________________________________________________________
flatten_3 (Flatten)          (None, 3072)              0
_________________________________________________________________
dense_5 (Dense)              (None, 256)               786688
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_6 (Dense)              (None, 7)                 1799
=================================================================
Total params: 15,503,175
Trainable params: 788,487
Non-trainable params: 14,714,688
_________________________________________________________________
```

Figure 50. Third Experiment

As it is clear, that method did not perform well in our dataset. The reasons can be that VGG performs better in bigger images and in 3D images. It also needs a bigger number of images.

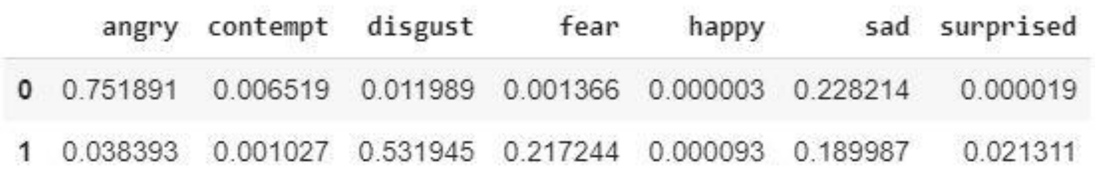
Having a challenging dataset with few images and high number of classes makes the perdiction difficult. We tried the same algorithms for prediting between two emotions only and we got a pretty good accuaray:



```
Epoch 1/15
307/307 [==============================] - 16s 53ms/step - loss: 0.8587 - accuracy: 0.6348 - val_loss: 0.6778 - val_accuracy: 0.5825
Epoch 2/15
307/307 [==============================] - 15s 50ms/step - loss: 0.5198 - accuracy: 0.7337 - val_loss: 0.7374 - val_accuracy: 0.6311
Epoch 3/15
307/307 [==============================] - 15s 50ms/step - loss: 0.4515 - accuracy: 0.8011 - val_loss: 0.5800 - val_accuracy: 0.6990
Epoch 4/15
307/307 [==============================] - 15s 50ms/step - loss: 0.3777 - accuracy: 0.8402 - val_loss: 0.7544 - val_accuracy: 0.6893
Epoch 5/15
307/307 [==============================] - 16s 51ms/step - loss: 0.3360 - accuracy: 0.8641 - val_loss: 0.5862 - val_accuracy: 0.7282
Epoch 6/15
307/307 [==============================] - 15s 50ms/step - loss: 0.2998 - accuracy: 0.8728 - val_loss: 0.7115 - val_accuracy: 0.7282
Epoch 7/15
307/307 [==============================] - 15s 50ms/step - loss: 0.1954 - accuracy: 0.9228 - val_loss: 0.5073 - val_accuracy: 0.7961
Epoch 8/15
307/307 [==============================] - 16s 51ms/step - loss: 0.1511 - accuracy: 0.9402 - val_loss: 0.4620 - val_accuracy: 0.8350
Epoch 9/15
307/307 [==============================] - 15s 50ms/step - loss: 0.2042 - accuracy: 0.9272 - val_loss: 0.8417 - val_accuracy: 0.7087
Epoch 10/15
307/307 [==============================] - 15s 50ms/step - loss: 0.1139 - accuracy: 0.9630 - val_loss: 0.5946 - val_accuracy: 0.8350
Epoch 11/15
307/307 [==============================] - 15s 50ms/step - loss: 0.1102 - accuracy: 0.9674 - val_loss: 0.6777 - val_accuracy: 0.8252
Epoch 12/15
307/307 [==============================] - 15s 50ms/step - loss: 0.0796 - accuracy: 0.9717 - val_loss: 0.9079 - val_accuracy: 0.7961
Epoch 13/15
307/307 [==============================] - 15s 50ms/step - loss: 0.0576 - accuracy: 0.9783 - val_loss: 0.9548 - val_accuracy: 0.7670
Epoch 14/15
307/307 [==============================] - 16s 51ms/step - loss: 0.0870 - accuracy: 0.9685 - val_loss: 0.8581 - val_accuracy: 0.7573
Epoch 15/15
307/307 [==============================] - 15s 50ms/step - loss: 0.0675 - accuracy: 0.9826 - val_loss: 0.9875 - val_accuracy: 0.7476
<tensorflow.python.keras.callbacks.History at 0x7f3fba276208>
```

After applying the models and saving the history, we started to check the ratio of predicted with actual emotions.

Here is how are model is predicting. It shows the wights for every emotion in a specific image.



	angry	contempt	disgust	fear	happy	sad	surprised
0	0.751891	0.006519	0.011989	0.001366	0.000003	0.228214	0.000019
1	0.038393	0.001027	0.531945	0.217244	0.000093	0.189987	0.021311

Figure 51. Representation of Weights for each Emotion in an Image

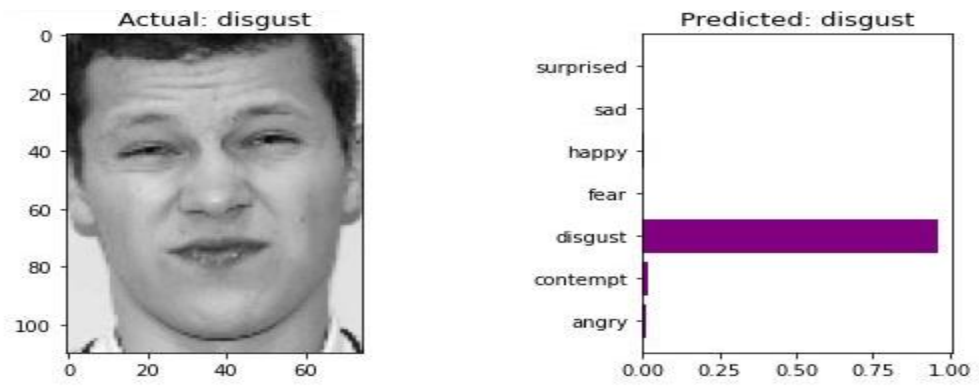


Figure 52. Prediction of actual emotion First Model (disgust): 96.0%

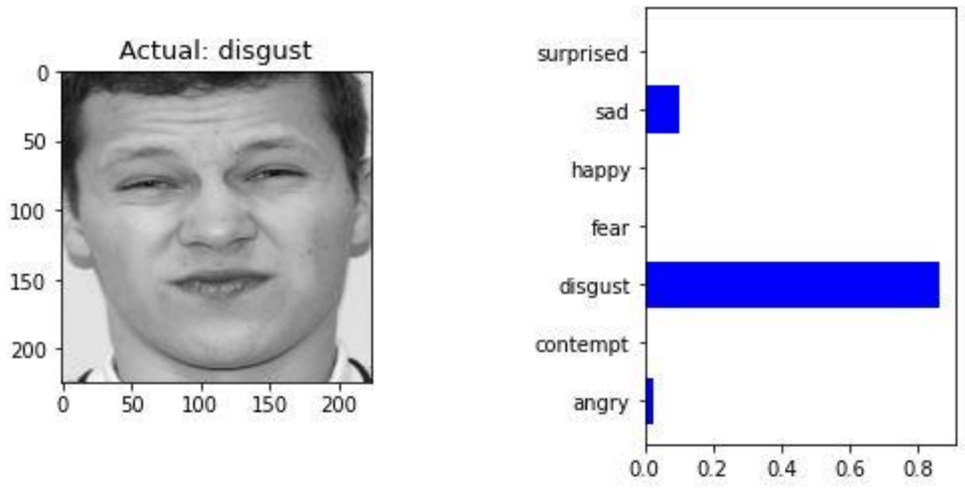


Figure 53. Prediction of actual emotion Second Model (disgust): 86.64%

The face in the first model is expressing the disgust emotion, and the model predicted it is showing the disgust emotion. The model made the highest prediction at 96.0% for the emotion of disgust, followed by 1.58% for for the emotion of contempt.

While in the second model the highest prediction at 86.64% for the emotion of disgust, followed by 10.46% for for the emotion of sad.

Overall looks like our models perform well in “disgust”

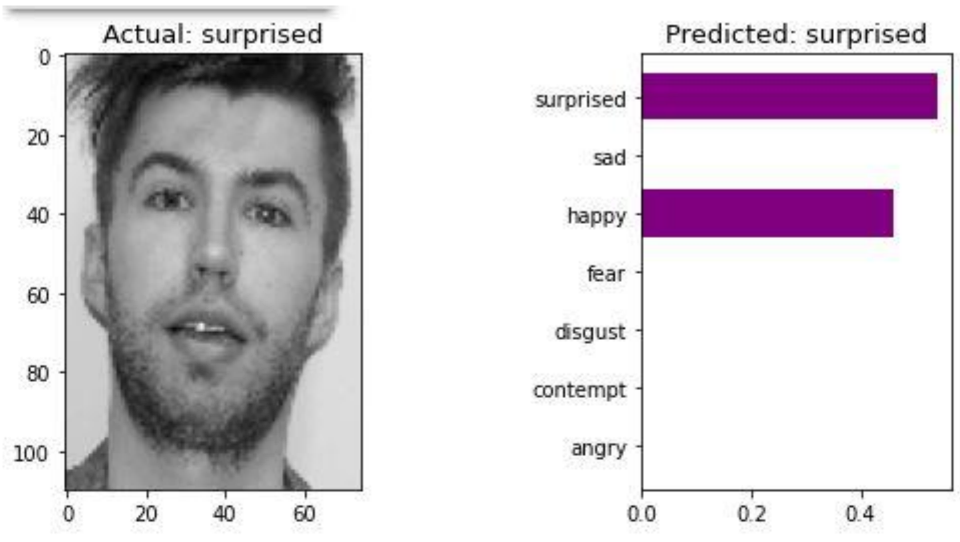


Figure 54. Prediction of actual emotion First Model (surprised): 55.17%

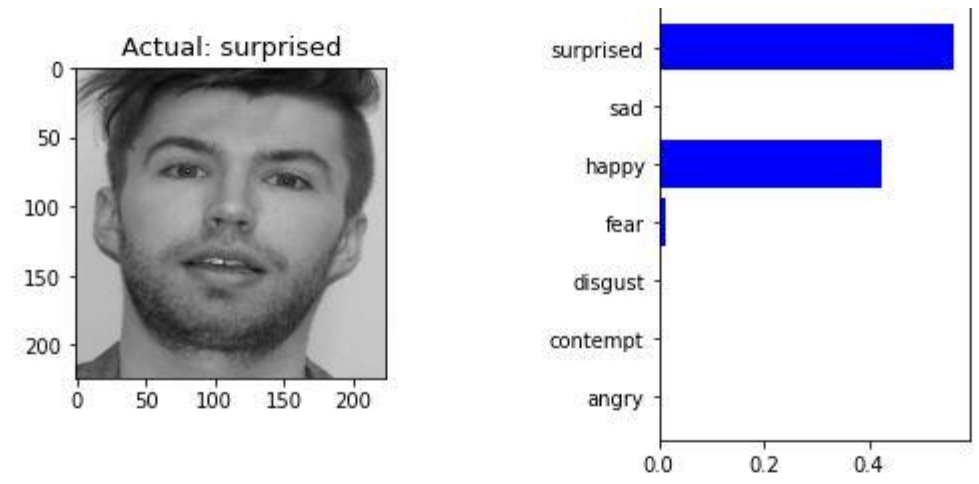


Figure 55 Prediction of actual emotion SecondModel: 56.52%

This face is expressing the surprised emotion, and the model predicted it is showing the surprised emotion. The first model made the highest prediction at 55.17% for the emotion of surprised, followed by 44.52% for for the emotion of happy. Almost the same with the second model.

In the graph below we see the Predicted vs Actual Emotions for the Second Model.

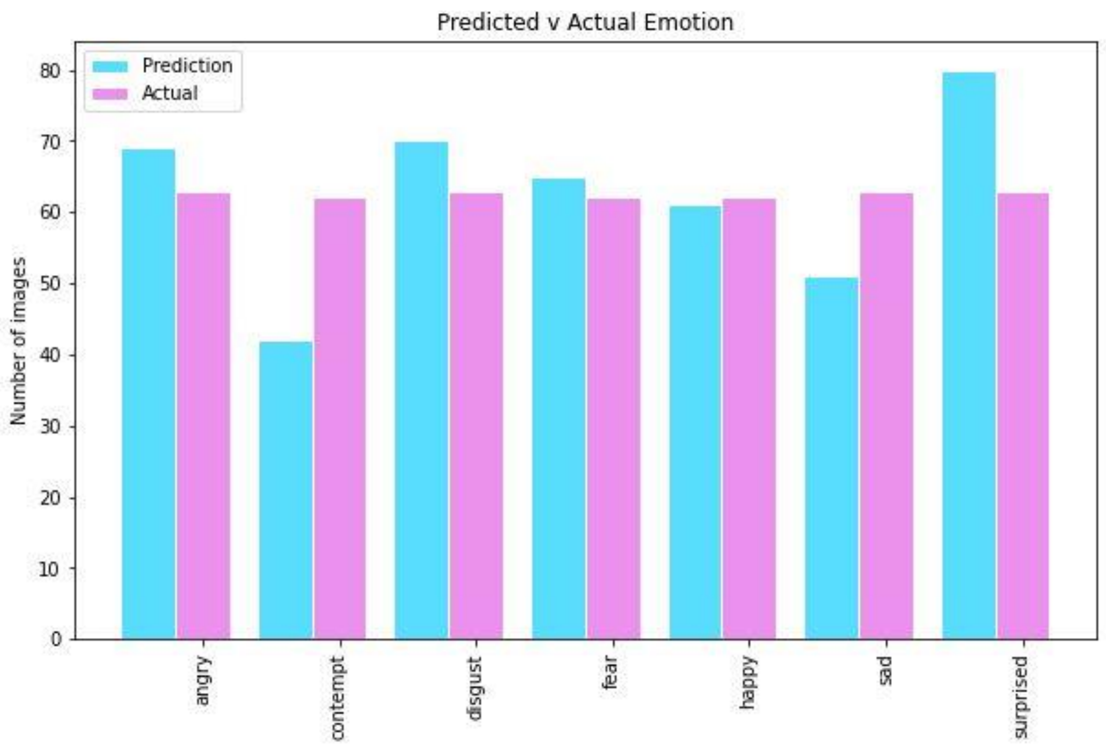


Figure 56 Predicted vs Actual Emotion

We notice that the biggest misspredictions are in emottions surprised, contempt and sad, while fear and happy have very good prediction.

overpredict the emotions: angtry, disgust, fear, surprised

underpredict the emotions: contempt, sad

predict very good emotions: happy

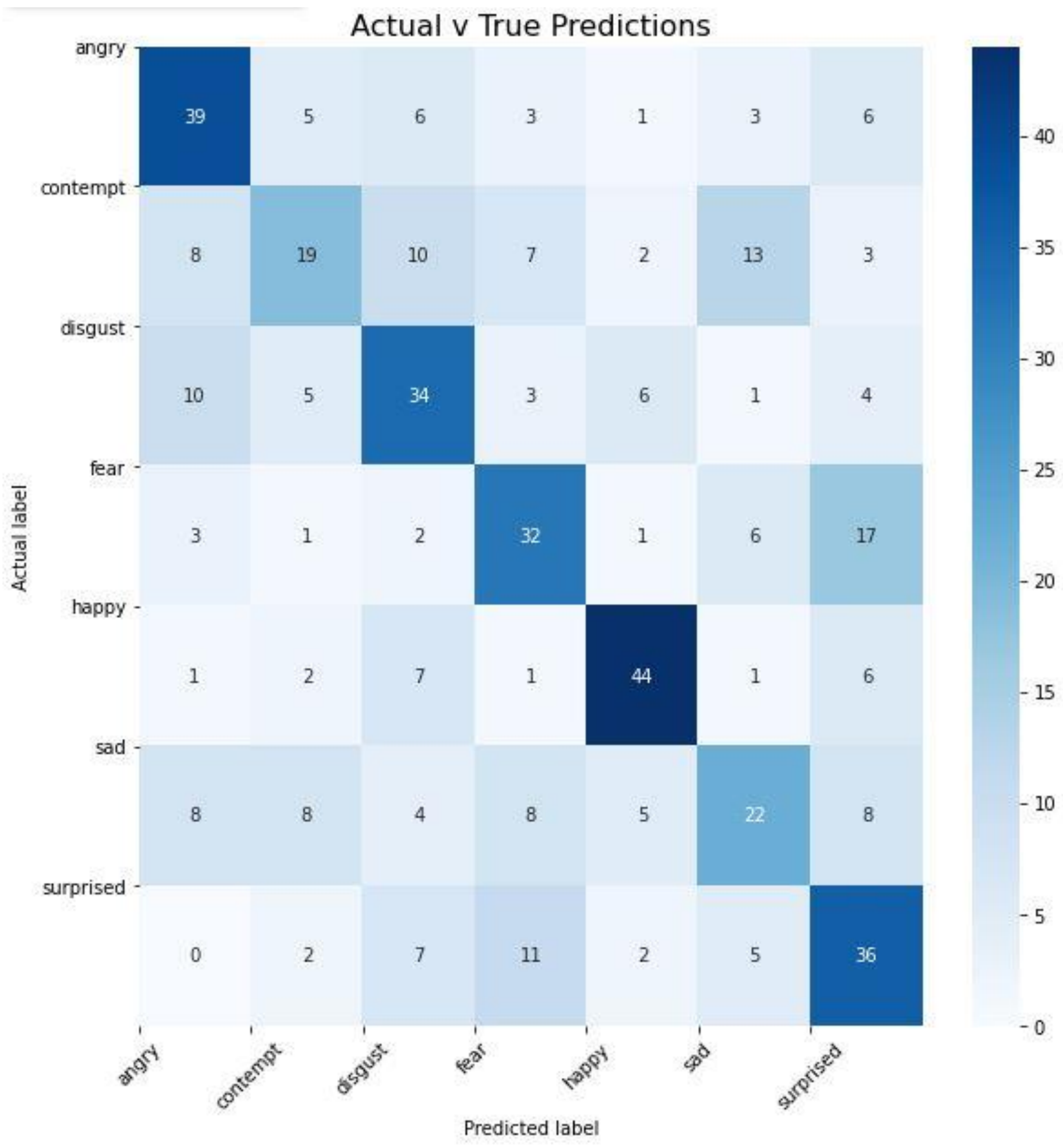


Figure 57 Confusion Matrix

The confusion matrix will help us interpret the predictions more closely, where I'll be able to see which emotions the model tended to confuse with each other, and how well predictions were made for each emotion. As a result, the algorithm has performed best in detecting happy faces and performed worst in detecting contempt. The most problematic case is confusing fear with surprise, which actually we expected this to happen due to their similarity, where in both cases the eyebrows are risen up and the eyes are wide open. The second most problematic is confusing contempt with sad, where again face expression are very similar to each other. We notice that the algorithm performs very well in recognizing the difference between surprised and angry emotions.

CHAPTER 6

CONCLUSION AND FUTURE WORK

My aim in this thesis were to go in detail in every step of Deep Learning, starting from finding the dataset and ending up analyzing the results. I separate the work in two main parts: Dataset and the construction of the CNN.

Choosing a dataset for Emotion Recognition is considered difficult since you have so many datasets available but they very a lot from each other regarding dimentions, conditions, size, ect and is needed to make a deep research to know them, get access (because in most cases you have to request at least 10 days ahead and it is not garanted if you can have them).

Preparing the dataset is quite challenge too. It is very time consuming since it takes weeks to "relable" or to correct possible mistakes in the dataset. And we are takinginto considersation a dataset which is considered small. Regarding the labels, it always brings confusion since the way of expressing emotions are more subjective than objects. Based on the research I have read even psychologist have different ideas. Regarding the data preprocesing hdf5 format is the best choice when the data is in bigger dimentions that 128x128 otherwise csv would be a good choice.

Choosing the algorithm for FER when you have more than 5 classes, is very challenging. Especially when you leave "all the work" to the CNN. In this kind of models, we need to design a big network and with high quality images, which was not possible to be applied in my case from the lack of a computer with good parameters and high perfomrance.

As a future work i would consider using Action Units to detect as features the movement of the muscles of the face and then to feed the CNN

Bibliography

[1] C. Darwin and P. Prodger, The expression of the emotions in man and animals., USA: Oxford University Press, 1998.

[2] Y.-I. Tian, T. Kanade, and J. F. Cohn, "Recognizing action units for facial expression analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23.

[3] Mehrabian, A.; Russell, J.A., An approach to Environmental Psychology, Cambridge: The MIT Press, 1974.

[4] P. Ekman, "“Lie catching and microexpressions"," in *The philosophy of deception*, 2009.

[5] Friesen, P. Ekman nad W. V., "“Constants across cultures in the face and emotion.”," *Journal of personality and social psychology,* vol. 17, 1971.

[6] P. Ekman, "“Strong evidence for universals in facial expressions: a reply to russell’s mistaken critique"," *Psychologixal bulletin,* vol. 115, 1994.

[7] P. Ekman, "“Facial action coding system (facs)”," in *A human face,* 2002.

[8] H. Gunes and B. Schuller, "“Categorical and dimensional affect analysis in continuous input: Current trendsand future durections"," in *Image and Vision Computing*, 2013.

[9] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "“The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression"," in *Computer Vision and Pattern recognition Workshops - IEEE,* 2010.

[10] M. Pantic, M. Valstar, R. Rademaker, and L. Maat, "“Web-based database for facial expression analysis,”," in *ICME, IEEE International Conference on. IEEE, 2005*, 2005.

[11] M. Valstar and M. Pantic, "“Induced disgust, happiness and surprise: an addition to the mmi facial expression database"," in *3rd Intern. Workshop on EMOTION (satellite of LREC): Corpora for Research on Emotion and Affect*, 2010.

[12] J. M. Susskind, A. K. Anderson, and G. E. Hinton, "“The toronto face database,”," Department of Computer Science, University of Toronto, Toronto, ON, Canada, 2010.

[13] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "“Coding facial expressions with gabor wavelets"," in *Automatic Face and Gesture Recognition, Third IEEE International Conference*, 1998.

[14] . J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W., "Challenges in representation learning: A report on three machine learning contests," in *International Conference on Neural Information Processing*, Springer, 2013.

[15] A. Dhall, R. Goecke, S. Ghosh, J. Joshi, J. Hoey, and T. Gedeon, "From individual to group-level emotion recognition: Emotiw 5.0," in *ACM International Conference on Multimodal Interaction*, 2017.

[16] A. Dhall, O. Ramana Murthy, R. Goecke, J. Joshi, and T. Gedeon, "Video and image based emotion recognition challenges in the wild: Emotiw 2015," in *ACM on International Conference on Multimodal Interaction*, 2015.

[17] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," *Image and Computer Vision*, vol. 28, 2010.

[18] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, "A 3d facial expression database for facial behavior research," in *Automatic face and gesture recognition*, 2006.

[19] G. Zhao, X. Huang, M. Taini, S. Z. Li, and M. PietikaInen, "Facial expression recognition from near-infrared videos," *Image and Vision Computing,* vol. 29, 2011.

[20] O. Langner, R. Dotsch, G. Bijlstra, D. H. Wigboldus, S. T. Hawk, and A. van Knippenberg, "Presentation and validation of the radboud faces database," in *Cognition and Emotion*, 2010.

[21] D. Lundqvist, A. Flykt, and A. Ohman, "The karolinska directed, emotional face (kdef)," 1998.

[22] C. F. Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild," in *IEEE International Conference on Computer Vision &* , 2016.

[23] S. Li and W. Deng, "Reliable crowdsourcing and deep localitypreserving learning for unconstrained facial expression recognition," in *IEEE Transactions on Image Processing*, 2018.

[24] B. H. M. H. M. Ali Mollahosseini, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," 2017.

[25] J. Guo, Z. Lei, J. Wan, E. Avots and N. Hajarolasvadi, "Dominant and Complementary Emotion Recognition From Still Images of Faces," in *IEEE*, 2018.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing* , 2012.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014.

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE conference on computer vision and pattern recognition*, 2015.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016.

[30] S. Ouellet, "Real-time emotion recognition for gaming using deep convolutional network features," 2014.

[31] Hassner, G. Levi and T., "Emotion recognition in the wild via convolutional neural networks and mapped binary patterns," in *International conference on multimodal interaction. ACM*, 2015.

[32] H. Ding, S. K. Zhou, and R. Chellappa, "Facenet2expnet: Regularizing a deep face recognition net for expression recognition," in *Automatic Face & Gesture Recognition, 2th IEEE International Conference on. IEEE*, 2017.

[33] X. Zhao, X. Liang, L. Liu, T. Li, Y. Han, N. Vasconcelos, and S. Yan, "Peak-piloted deep network for facial expression recognition," in *European conference on computer vision*, 2016.

[34] C. A. Corneanu, M. O. Simon, J. F. Cohn, and S. E. Guerrero, "Survey on rgb, 3d, thermal, and multimodal approaches for facial expression recognition: History, trends, and affect-related applications," in *IEEE transactions on pattern analysis and machine intelligence*, 2016.

[35] P. Hu, D. Cai, S. Wang, A. Yao, and Y. Chen, "Learning supervised scoring ensemble for emotion recognition in the wild," in *19th ACM International Conference on Multimodal Interaction*, 2017.

[36] Mahoor, B. Hasani and M. H., "Facial expression recognition using enhanced deep 3d convolutional neural networks," in *Computer Vision and Pattern Recognition Workshops (CVPRW) IEEE*, 2017.

[37] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and demantic segmentation," in *IEEE conference on computer vision and pattern recognition*, 2014.

[38] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015.

[39] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," in *IEEE transactions on pattern analysis and machine intelligence*, 2013.

[40] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Networks*, vol. 18, 2006.

[41] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," 2006.

[42] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014.

[43] C. D. a. P. Prodger, The expression of the emotions in man and animals, USA: Oxford University Press, 1998.