André Teixeira Lopes

Facial Expression recognition using Deep Learning - Convolutional Neural Network

Vitória - ES, Brasil

André Teixeira Lopes

Facial Expression recognition using Deep Learning - Convolutional Neural Network

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Informática.

Orientador:
Prof. Dr. Thiago Oliveira dos Santos

Co-orientador:
Prof. Dr. Edilson de Aguiar

Departamento de Informática
Centro Tecnológico
Universidade Federal do Espírito Santo

Vitória - ES, Brasil

Prof. Dr. Thiago Oliveira dos Santos
Orientador

Prof. Dr. Edilson de Aguiar
Co-orientador

Prof. Dr. Alberto Ferreira De Souza
Universidade Federal do Espírito Santo

Prof. Dr. Siome Klein Goldenstein
Universidade Estadual de Campinas

Resumo

O reconhecimento de expressões faciais tem sido uma área de pesquisa ativa nos últimos dez anos, com áreas de aplicação em crescimento, como animação de personagens, neuro-marketing e robôs sociáveis. O reconhecimento de uma expressão facial não é um problema fácil para métodos de aprendizagem de máquina, dado que pessoas diferentes podem variar na forma com que mostram suas expressões. Até imagens da mesma pessoa em uma expressão específica podem variar em brilho, cor de fundo e posição. Portanto, reconhecer expressões faciais ainda é um problema desafiador.

Para resolver esses problemas, nesse trabalho nós propomos um sistema de reconhecimento de expressões faciais que usa redes neurais convolucionais. Geração sintética de dados e diferentes operações de pré-processamento foram estudadas em conjunto com várias arquiteturas de redes neurais convolucionais. A geração sintética de dados e as etapas de pré-processamento foram usadas para ajudar a rede na seleção de características. Experimentos foram executados em três bancos de dados largamente utilizados (Cohn-Kanade, JAFFE, e BU3DFE) e foram feitas validações entre bancos de dados(i.e., treinar em um banco de dados e testar em outro). A abordagem proposta mostrou ser muito efetiva, melhorando os resultados do estado-da-arte na literatura e permitindo o reconhecimento de expressões faciais em tempo real com computadores padrões.

Palavras-Chave: Redes Convolucionais; Visão Computacional; Aprendizagem de Máquina; Características Específicas de Expressões; Aprendizagem Profunda

Abstract

Facial expression recognition has been an active research area in the past ten years, with growing application areas such avatar animation, neuromarketing and sociable robots. The recognition of facial expressions is not an easy problem for machine learning methods, since people can vary significantly in the way that they show their expressions. Even images of the same person in one expression can vary in brightness, background and position. Hence, facial expression recognition is still a challenging problem.

To address these problems, in this work we propose a facial expression recognition system that uses Convolutional Neural Networks. Data augmentation and different pre-processing steps were studied together with various Convolutional Neural Networks architectures. The data augmentation and pre-processing steps were used to help the network on the feature selection. Experiments were carried out with three largely used databases (Cohn-Kanade, JAFFE, and BU3DFE) and cross-database validations (i.e. training in one database and test in another) were also performed. The proposed approach has shown to be very effective, improving the state-of-the-art results in the literature and allowing real time facial expression recognition with standard PC computers.

Keywords: Convolutional Neural Networks; Computer Vision; Machine Learning; Expression Specific Features; Deep Learning

Dedicatory

To my wife, Gislayni, my mother, Zenaide, my father, Flávio my sister, Ana and my brother, Daniel, who are the most important people in my life.

Acknowledgment

Firstly, I would like to thank God.

My wife, the love of my life, who always been with me supporting, helping and loving me.

My parents and brothers, for supporting me throughout the challenges of life, including this one.

I would like to say thank you for my advisor, Dr. Thiago Oliveira dos Santos, and my co-advisor, Dr. Edilson de Aguiar, for guiding, teaching and supporting me to achieve this goal.

All my friends who worked with me in High Performance Computer Laboratory (LCAD) were also very important to this work. I thank you all.

I also would like to Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for scolarship.

Contents

List of Figures

List of Tables

1 Introduction

Facial expression is one of the most significant characteristics of human emotion (WU; LIU; ZHA, 2005). Its scientific study began as late as 1872, with the work of *Charles Darwin* in his book "The Expression of the Emotions in Man and Animals" (DARWIN, 1916). According to *Li an Jain* (LI; JAIN, 2011), it can be defined as the facial changes in response to person's internal emotional state, intentions, or social communication. Nowadays, its automatic analysis has been an active research field, driven by the interesting applications of this area like interactive games, data-driven animation, sociable robotics, online/remote education and many others human-computer interaction systems.

Expression recognition is a task that humans perform daily and effortlessly (LI; JAIN, 2011), but that is not yet easily performed by computers. A lot of research work have tried to make computers reach the same accuracy of humans, and some examples of these works are highlighted here. This problem is still a challenge for computers because it is very hard to separate the expressions feature space. Facial features from one subject in two different expressions may be very close to one another, while facial features from two subjects with the same expression may be very far from one another. Figure 1 shows three subjects with a happy expression. As can be seen in the figure, the images vary a lot one form each other not only in the way that the subjects show their expression, but also in lighting, brightness, position and background.

Figure 1: Three different subjects with the a happy expression. As can be seen the images vary a lot one form each other not only in the way that the subjects show their expression but also in light, brightness, position and background. The images are from the following databases: CK+ database (LUCEY et al., 2010), JAFFE database (LYONS; BUDYNEK; AKAMATSU, 1999) and BU-3DFE database (YIN et al., 2006), in this order.

Facial expression recognition systems can be divided in two main categories, those that work with static images (LIU et al., 2014; SHAN; GONG; MCOWAN, 2009; LIU; SONG; WANG, 2012; LOPES; AGUIAR; SANTOS, 2015) and those that work with dynamic image sequences (BYEON; KWAK, 2014; LIEN et al., 1999). Static-based methods do not use temporal information. Sequence based methods, in the other hand, use temporal information of images to recognize the expression based on one or more frames. Automated systems for facial expression recognition receive the expected input (static image or image sequence) and give as output the code of one of the basic expressions (anger, sad, surprise, happy, disgust and fear, for example). Some systems also recognize the neutral expression. This work present methods based on static images and image sequences and it will consider the six and seven expressions (six basic plus neutral).

Automatic facial expression analysis comprises three steps: face acquisition, facial data extraction and representation, and facial expression recognition (LI; JAIN, 2011). Face acquisition aims to detect and extract a face in a given image. Facial data extraction and representation focus on removing the interesting data for the expression recognition and on representing it in some way. And, the last stage, aims to effectively recognize witch expression is presented in the input image. This work will focus on the these two last stages.

1.1 Motivation

Recently, a lot of work has been employed in the facial expression recognition research field (LIU et al., 2014; SHAN; GONG; MCOWAN, 2009; BARTLETT et al., 2005; LOPES; AGUIAR; SANTOS, 2015). Methods using convolutional neural networks (CNN) for face recognition, like the proposed by *Lawrence et. al* in (LAWRENCE et al., 1997), can also be found in the literature. CNN's have a high computational cost in terms of memory and speed in the learning stage, but can achieve some degree of shift and deformation invariance. Nowadays, this approach became more feasible thanks to the hardware evolution and the capable of using the GPU processors to perform convolutions and the large amount of available data, that allows the learning of all CNN's parameters. This network type has demonstrated being able to achieve high recognition rates in various image recognition tasks like character recognition (LV, 2011), handwritten digit recognition (NIU; SUEN, 2012), object recognition (LECUN; HUANG; BOTTOU, 2004), and facial expression recognition (FASEL, 2002b, 2002a; MATSUGU et al., 2003; BYEON; KWAK, 2014; LOPES; AGUIAR; SANTOS, 2015).

Although there are many methods in the literature, some aspects still deserve attention, for example, accuracy is somewhat low in (ZHAO-YI; ZHI-QIANG; YU, 2009) and (WU; LIU; ZHA, 2005), validation methods could be improved in (LIU; REALE; YIN, 2012), (ZHAO-YI; ZHI-QIANG; YU, 2009) and (FASEL, 2002b), the recognition time could be a little improved to be perform real time evaluations in (LIU et al., 2014) and others limitations in general.

Therefore, given the large application area of automatic facial expression recognition, we were motivated to find an efficient and effective approach to cope with these limitations. Thereby, based on the fact that most part of facial expression features are visual, we want to investigate image-based recognition methods to solve this problem in a way that it could achieve a high accuracy on real environments while keeping a fast recognition evaluation.

1.2 Contributions

In this work, we propose two novel facial expression recognition methods. The methods are based on convolutional neural networks and focuses on emphasizing the features present in the facial expressions. The localization of the face is assumed to be known (face localization is not addressed in this work). The first method employs a deeper convolutional neural network and has just one constraint: the location of both eyes centers. The other method uses prior knowledge of the subject neutral expression and the location of both eyes centers to increase the expression recognition performance in some cases and reduce the complexity of the classification problem. In order to consider such information, the method requires two input images of a subject: one in the neutral expression and one in the expression to be classified. Such inputs could be considered an image sequence with two frames. Although the need for a neutral expression is a constraint, we show in this study that the recognition performance can be increased in applications where this type of neutral expression image is available (e.g. applications including calibration phase). In addition, the proposed method can be easily combined with a neutral expression detector in order to be used in scenarios without calibration. Our experiments showed that both approaches can be performed in real time (0.01 second per image) in standard PC computers and can achieve 98.92% and 99.06% of accuracy for six (angry, disgust, fear, happy, sad and surprise) expressions. To the best of our knowledge, both techniques present the best results in terms of time (training and recognition) and accuracy in the literature. In additional, we have performed an extensive validation including cross-database validations between three facial expressions databases. To show the applicability use of the

presented method with an automatic detector for the neutral expression, we also present the results for a neutral expression detector.

In summary, the main contributions of this work are:

- An approach combining standard methods, like image normalizations, synthetic training samples (i.e. real images with artificial rotations) generation and Convolutional Neural Network, for facial expression recognition, that is able to achieve a very high accuracy rate;
- A novel facial expression recognition method that uses prior knowledge of the subject neutral expression to increase the expression recognition performance in time and accuracy;
- A neutral expression detector in order to be used in scenarios where the neutral expression is not known;
- A study of the behavior of the proposed methods with the databases of the literature.
- The source code of this work is available together with the instructions to replicate the reported results, at:
  `htt://wwww.github.com/andreteixeiralopes/deepfacialexpression`

1.3 Structure

The structure of this dissertation follows the structure below:

- After this introduction, Chapter 2 briefly reviews the Literature about Machine Learning, Deep Learning, Convolutional Neural Networks and Facial Expression Recognition. Furthermore, it shows the solutions proposed by others that achieve the state-of-the-art results and are more closely related to the proposed method.
- Chapter 3 introduces two novel Facial Expression Recognition methods based on Convolutional Neural Networks. One that is based on the intensity normalization and another based on the neutral expression subtraction. In addition, one classifier that aims to detect only the neutral expression is presented.
- Chapter 4 describe the experimental setup, the databases used to the method's evaluation, the accuracy metrics and how the data is separated in the training, validating and testing groups.

- Chapter 5 shows the results achieved by the proposed methodology, describes the study conducted for evaluating the impact in the accuracy of each step of the proposed methods, discusses the results achieved, compares with the methods in the literature, and finally presents the limitations of the proposed methods.

- Chapter 6 summarizes the work, presents the conclusions and point out directions for future developments.

2 Theoretical Background

In this chapter, we describe the theoretical background of this work. It begins with a brief introduction of machine learning, artificial neural networks and deep learning. A succinct description of the presented concepts can be found in (BISHOP, 2006), (HAYKIN, 2008), (ROSENBLATT, 1962), (LECUN; BENGIO; HINTON, 2015) and (BENGIO; GOODFELLOW; COURVILLE, 2015). Section 2.2 presents the main concepts of Convolutional Neural Networks (CNN). Finally, in the last section, the facial expression recognition problem is presented with the latest methods in the literature that tries cope this problem and its limitations.

2.1 Machine Learning and Deep Learning

Machine learning has been an active research field since its creation. The interest in this area is motivated by the many applications of the "intelligent systems" in our everyday lives. Nowadays, most of daily used softwares, like search engines, social networks, personal assistants, text editors and a lot of others, have, in some aspect, an intelligent behavior performed by machine learning algorithms. According to *Kluwer* in (KLUWER, 1998), Machine Learning can be defined as the study and construction of algorithms that can learn from and make predictions on data .

Learning algorithms can be supervised or unsupervised (there are some mixed models already proposed) (BISHOP, 2006). Supervised algorithms learns from a labeled data, i.e. the target value (expected result) of a specific input is know and this value is used during the learning. On the other hand, unsupervised algorithms learns how to separate correlated data (groups of similar examples), in this case the target value is unknown or even do not exist. The tasks performed by machine learning algorithms are mainly categorized in classification, regression and clustering (BISHOP, 2006). The classification task aims to create a mapping function that maps a given input in a specific set of outputs. The regression task aims to estimate an unknown function based on a set of input values

with its respective target values, in order to calculate the output value of an unknown input. Finally, the clustering task is similar to the classification, without the target value. The algorithm needs to correlate the input data into groups (clusters) of similar values, in such way that the more similar objects are in the same group.

One of the models of machine learning is the artificial neural network. Artificial neural networks are an artificial representation of biological brains, proposed by *McCulloch and Pitts* in (MCCULLOCH; PITTS, 1988). The artificial neuron and the artificial neural networks proposed by *McCulloc and Pitss* are shown in Figure 2. The processing inside an artificial neuron (Figure 2 on left) consists of a linear combination of the inputs, $net = w_1x_1 + w_2x_2 + ... + w_jx_j = \sum_{j=1}^{n} x_jw_j$. Each input (x_j) is associated to a weight (w_j), that can be interpreted as the importance of the input (x_j). The result of this linear combination is given to an activation function (ϕ), depending of the *net* value this function returns 0 (deactivate the neuron) or 1 (activate the neuron). Therefore, the artificial neuron is a simple function that calculates an output y based on the inputs x_j, the weights w_j and an activation function, this function is shown on equation 2.1. Despite this simplicity of the neuron model, the potential of an artificial neural network (Figure 2 on the right) is based on possibility of neurons arrangement in layers, each layer with dozens (or even thousands) of neurons. The neurons in first layer receives their inputs, process this information, and propagates its output to the neurons in next layer. This operation is performed until the last neuron of the last layer. The learning (training) on artificial neural networks consist of changing the weights based on a set of training samples (with their target values) and find the best W (weights) set that minimize the difference between the network output values and the target values (HAYKIN, 2008).

$$y = \phi(\sum_{j=1}^{n} x_jw_j) \tag{2.1}$$

After the proposal of this model, several learning algorithms have been proposed to train multi-layered artificial networks. One of the most known training algorithms is the Backpropagation, proposed by *Rumelhart et al* in (RUMELHART; HINTON; WILLIAMS, 1988). This method calculates the gradient of a loss function (generally the difference between the expected value and the network output) with respect to all the weights in the network. Then, this gradient is used to update the weights in order to minimize the loss function.

For some decades, after the proposal of artificial neural networks, neural network

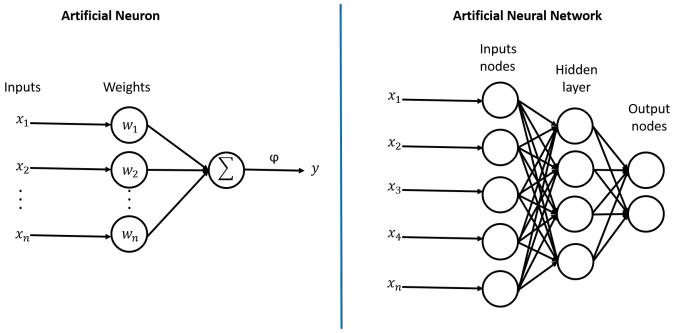

Figure 2: Artificial neuron and Artificial neural network. On the left the artificial neuron model and in the right the Artificial Neural Network , proposed by *McCulloc and Pitss* is shown.

researches were motivated to find a way to train deep multi-layer neural networks (i.e. networks with more than one or two hidden layers) in order to increase their accuracy (BENGIO; LECUN, 2007) (UTGOFF; STRACUZZI, 2002). Until 2006, many new attempts have shown little success, generally training deeper neural networks decreased the accuracy of the learning systems (BENGIO; GOODFELLOW; COURVILLE, 2015). Despite these results, a specific kind of neural network, the Convolutional Neural Network (which is the method studied in this work) proposed in 1998 by *LeCun et al.* (LECUN et al., 1998) has shown to be very effective in learning features with a high level of abstraction when using deeper architectures (i.e, with several layers). Recently, Convolutional Neural Networks have shown very promising results in a large of learning tasks based on visual features (LV, 2011; NIU; SUEN, 2012; LECUN; HUANG; BOTTOU, 2004; BYEON; KWAK, 2014; LOPES; AGUIAR; SANTOS, 2015; TAIGMAN et al., 2014). Therefore, this method was choose to be the object of study in this work and is explained in Section 2.2.

In 2006, *Hinton et al.* proposed the Deep Belief Networks (DBNs) (HINTON; OSINDERO; TEH, 2006) and an algorithm capable of training deeper architectures (i.e with many layers) of this network. This algorithm greedily trains one layer at a time, the learning of each layer is performed in an unsupervised way. The unsupervised learning is performed with Restricted Boltzmann Machines (RBM) (FREUND; HAUSSLER, 1994) (previously, the RBMs has shown successful unsupervised learning only on two layers networks). After that, some algorithms that uses this same approach (training intermediate levels of representation using unsupervised learning) have been proposed (BENGIO; LECUN, 2007), (POULTNEY; CHOPRA; LECUN, 2006), (MOBAHI; COLLOBERT; WESTON, 2009) and (WESTON; RATLE; COLLOBERT, 2008). Since 2006, these networks model has shown very accurate results in a large range of applications, like classification (AHMED et al., 2008; BENGIO; LECUN, 2007; LAROCHELLE et al., 2007), dimensionally reduction

(HINTON; SALAKHUTDINOV, 2006), regression (SALAKHUTDINOV; HINTON, 2007), segmentation (LEVNER, 2008) and many others.

2.2 Convolutional Neural Networks

Despite the difficulties found on training deeper networks in a supervised way, one model has been an exception for this rule: the Convolutional Neural Network. The Convolutional Neural Network model are a specialized kind of neural networks, a biologically-inspired variant of the Multi Layer Perceptron (ROSENBLATT, 1962). Their work was inspired by the previous work of Hubel and Wiesel on the cat's visual cortex (HUBEL; WIESEL, 1968). *Fukushima and Kunihiko* in (FUKUSHIMA, 1980) proposed a model based on local connections between neurons and a hierarchical organization of the layers to obtain a form of translation invariance for pattern recognition in images. It was achieved when neurons with same parameters are applied on patches of the previous layer at different locations. Following this idea, LeCun *et al.* in (LECUN et al., 1989) and in (LECUN et al., 1998), designed and trained Convolutional Neural Networks using the error gradient for several pattern recognition tasks, achieving the state-of-the-art results. Indeed, according to *Serre et al.* in (SERRE et al., 2007), the processing style of Convolutional Neural Networks is consistent with the modern understanding of the visual system physiology.

As discussed in Section 2.1, there were no success on training deeper architectures of neural networks models proposed before 2006. Convolutional Neural Networks are an exception for this rule. Generally, the architecture of these networks are deeper than successful artificial neural networks architectures, Convolutional Neural Networks are typically composed by five, six and seven (or more) layers.

This new approach are neural networks that use convolution in place of general matrix multiplication (BENGIO; GOODFELLOW; COURVILLE, 2015). In the Convolutional Neural Network model proposed by *LeCun et al.*, the architecture is hierarchical and the layers, generally, alternates between convolutional layers and sub-sampling layers and a fully connected layer. In this model a topographic structure is adopted for each layer, each neuron is associated with a specific region of the input image. At each location of the input image there are different neurons associated to it (i.e., there is an overlap between the associated regions of the neurons) (BENGIO; GOODFELLOW; COURVILLE, 2015). Convolutional Neural Networks takes advantage of their input format, images, and construct operations optimized to it. Unlike regular neural networks the neurons are arranged in 2

(for 2D images) or 3 (for 3D images) dimensions. As can be seen in Figure 3, different from general artificial neural networks that receives as input a single vector, Convolutional Neural Networks receives as input a 2D (or 3D) image. The neuron in a Convolutional Neural Network is linked with a specific image region and there is also an overlap between these regions (i.e., a part of one region can be an input of two or more neurons).

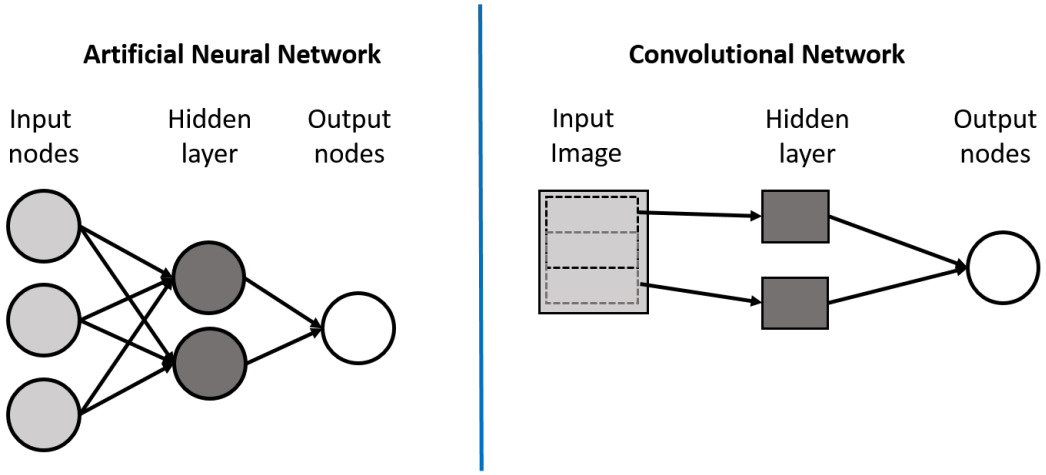

Figure 3: Artificial neural network versus Convolutional Neural Network. Different from general artificial neural networks that receives as input a single vector, Convolutional Neural Networks receives as input a 2D (or 3D) image. The neuron in a Convolutional Neural Network is linked with a specific image region and there is also an overlap between these regions (i.e., a part of one region can be an input of two or more neurons)

One advantage of this kind of network is that its input can be raw images, instead of an already selected set of features. The network is able to learn the set of features that best model the desired classification. The input image is also called map. After the map is given to the network it will perform the specified operations for each layer (convolutions or sub-samplings). The main difference between Convolutional Neural Networks methods is the number of layers and their arrangement.

The convolution layer aims to generate maps representing feature extracted of the input image by a predefined number of kernels. The map is generated in a operation that the kernel is shifted over the valid region of the input image. At the end of the training step, the Convolutional Neural Network will learn which are the best weights associated with these kernels. Convolutional layers are mainly parametrized by the number of generated maps and the kernels size. The learning can use a descendant gradient method, like the one proposed in (LECUN et al., 1998).

The sub-sampling layer aims to increase the position invariance of the network (CIRESAN et al., 2011) by reducing the map size. This layer replaces the output of the previous layer with a summary statistic of a neighborhood in the higher resolution version (BEN-

GIO; GOODFELLOW; COURVILLE, 2015). There are several kinds of sub-sampling layers, the most used are the max-pooling and the average-pooling (BENGIO; GOODFELLOW; COURVILLE, 2015). The max-pooling operation keeps only the maximum pixel value of a specific neighborhood region in the new map, while the average-pooling calculates an average of the neighbors to compose the new generated map.

Generally, the last hidden layer of a Convolutional Neural Network is a fully connected layer. This layer is similar to hidden layers of artificial neural networks. Their input is the maps generated by the convolutions and sub-samplings and the output can be the desired classes or another fully connected layer.

2.3 Facial Expression Recognition

Facial expression recognition is related to systems that aims to automatically analyze the facial movements and facial features changes of visual information to recognize a facial expression (LI; JAIN, 2011). Is important to mention that, facial expression recognition is different from emotion recognition. The emotion recognition requires a higher level of knowledge. Despite the facial expression could indicate an emotion, to the analysis of the emotion informations like context, body gesture, voice, cultural factors are also necessary (CARROLL; RUSSELL, 1996) and (RUSSELL, 1991).

Automatic facial expression analysis usually employs three main stages: face acquisition, facial data extraction and representation, and facial expression recognition (LI; JAIN, 2011), in this order. Face acquisition can be separated in two major steps: face detection (CHEN; WONG; CHIU, 2011; GARCIA; DELAKIS, 2004; ZHANG et al., 2012; BARTLETT et al., 2005) and head pose estimation (LIU; REALE; YIN, 2012; KIM et al., 2011; DEMIRKUS et al., 2014). After the face is located, the facial changes caused by facial expressions need to be extracted. The facial data extraction is a vital step for successful facial expression recognition. An ineffective data extraction causes also an ineffective recognition of the expression. The facial expression representation can be performed in two main ways: using geometric-based features or using appearance-based features (LI; JAIN, 2011). Geometric-based methods, present the shape, location and distances between facial components like mouth, eyes, eyebrow and nose (ZHANG et al., 1998; YANG; LIU; METAXAS, 2007; BARTLETT et al., 2005; JAIN; HU; AGGARWAL,). These features are extracted from the face image to form a feature vector that represents the face geometry. On the other hand, appearance-based methods, deal with the whole face, or specific regions; and the the

feature vectors used by these methods are acquired with image filters applied to the whole face image (LOPES; AGUIAR; SANTOS, 2015; LIU et al., 2014; SHAN; GONG; MCOWAN, 2009; BYEON; KWAK, 2014; LV; FENG; XU, 2014). This work will focus on an appearance-based method (Convolutional Neural Network), in the facial data extract and representation step and in the facial expression recognition.

Once feature vectors related to the facial expression are available, expression recognition can be performed. Usually this step is performed using a machine learning approach, although there are some template matching methods proposed in the literature (AHONEN; HADID; PIETIINEN, 2004; SHAN; GONG; MCOWAN, 2009). Machine learning methods basically performs a three-stage training procedure: feature learning, feature selection and classifier construction (LIU et al., 2014). Feature learning aims to detect the facial changes caused by an expression and discard feature not related to this change. The feature selection aims to reduce the feature set selected in the previous step and select only the most discriminative features for each expression. The selected feature set should minimize the intra-class (same expression) variation of the expression while maximizing the inter-class (different expressions) variation (SHAN; GONG; MCOWAN, 2009). This is one of the main problems of facial expression recognition, because images of different subjects in the same expression are well separated in the pixel space, while two images of the same person in different expressions could be very close to one another in the pixel space.

Facial expression recognition methods have been classified in two main categories, those ones that receive as input just a static image and others that receive as input image sequences (LI; JAIN, 2011). Methods that work with static images use information about just one image and the feature vector do not contain temporal information, only data about the current input (LIU et al., 2014; SHAN; GONG; MCOWAN, 2009; LIU; SONG; WANG, 2012). On the other hand, sequence based methods use information about two or more frames to recognize one expression, and could also include temporal information (BYEON; KWAK, 2014; LIEN et al., 1999). In this works we present one approach that uses static images and another one that uses image sequences.

As described by Li *et. al* in (LI; JAIN, 2011), in most of the cases, facial expression recognition systems receive the expected input (static image or image sequence) and outputs the facial expression, that is usually one of the following: neutral, anger, sad, surprise, happy, disgust and fear.

Several expression recognition approaches were developed in the last decade and a lot of progress has been made in this research area recently. An important part of this recent

progress was achieved thanks to the emergence of Deep Learning methods (LIU et al., 2014) and Convolutional Neural Networks methods (BYEON; KWAK, 2014). These approaches became computationally feasible thanks to the availability of powerful GPU processors, allowing high-performance numerical computation in graphics cards. The remainder of this section presents the last methods of facial expression recognition closely related to the approaches presented in this work and are the state-of-the-art methods. A full survey of the facial expression recognition research area can be found in (LI; JAIN, 2011) and in (CALEANU, 2013)

A deep learning technique for facial expression recognition was proposed by Lv *et al.* (LV; FENG; XU, 2014). They employed a deep belief network (DBN) to establish correlations between images and shapes of facial components (nose, eyes, mouth, eyebrows, etc) with specific expressions. The method extracts as many patches of the face as possible and then selects those that better describes the expressions. The method was trained and tested using the CK+ database (LUCEY et al., 2010) and the JAFFE database (LYONS; BUDYNEK; AKAMATSU, 1999). The training and testing procedures were carried out using a cross-validation technique, and their method achieves an accuracy rate of 90.57% in the JAFFE database and 91.11% in the CK+ database. However, their validation method relies on the possibility of one subject be in the training and in the testing groups at same time. In their methodology, the groups for the cross-validation were random generated using the entire database. The problem in this validation approach is that one cannot guarantee a fair comparison having images of same subject could be present in different groups. In this work, and some others of the literature like (LIU et al., 2014), (LOPES; AGUIAR; SANTOS, 2015) and (SHAN; GONG; MCOWAN, 2009), the authors use a cross-validation method that ensures that, if a subject was used to train the network, this same subject will not be used in the testing step. The training time and the recognition time was not mentioned by Lv *et al.* (LV; FENG; XU, 2014).

Another approach that also uses deep learning is presented by Liu *et al.* (LIU et al., 2014). The authors propose a Boosted Deep Belief Network (BDBN). Their BDBN performs the feature learning, feature selection and classifier construction iteratively in an unified loopy framework. For the six basic expressions (angry, disgust, fear, happy, sad and surprise) recognition, 80 DBNs were used, each one specialized in a specific image patch. The authors performed the experiments using the CK+ database and the JAFFE database. In their experiments, a cross-validation approach was employed without subject overlap (i.e. if images from one subject are in the train data, no images of this subject will be in the test data), they also conducted a cross-database validation (i.e. training

the system with one database, and testing with another). The accuracy was evaluated training and testing in the same database (CK+) and in the cross-database validation, where the training was performed using only the CK+ database and testing in the JAFFE database. To the first case the accuracy was 96.7% and for the second 68.0%. The time required to train the network was about 8 days. The recognition takes about 0.21 second to recognize the expression in each image - the authors use one weak classifier for each expression; therefore, to evaluate the expression in one image they need to present the image to each one of the six classifiers. The experiments were performed in a 6-core 2.4GHz PC using Matlab.

Shan *et al.* conduct a study of different machine learning methods like template matching, Support Vector Machine (SVM), Linear Discriminant Analysis (LDA) and linear programming to the facial expression recognition problem (SHAN; GONG; MCOWAN, 2009). The authors used the Local Binary Pattern (LBP) as a feature extractor to the learning methods. They also conducted an experiment that shows that geometric-based methods are very sensible to the image resolution (i.e. the lower the resolution, the lower the method's accuracy). On the other hand, appearance-based methods like Gabor Wavelets and LBP are not so sensitive to this variation. The experiments were performed using the CK+ and JAFFE database. The training and testing using the CK+ database achieves an accuracy of 95.10% while in a cross-validation with the JAFFE database this accuracy goes down to 41.30%. Both results were achieved using the SVM classifier. The training time and the recognition time were not mentioned by the authors.

3 Facial Expression Recognition System

In this work, two methods for facial expression recognition were studied. The first, presented in Section 3.1, is based on an intensity normalization procedure at the end of the pre-processing step. The second, presented in Section 3.2, is base in the subtraction of the expression image to be recognized from the neutral expression image, also at the end fo the pre-processing step. Both methods perform a pre-processing step to emphasize the features present in the image with the expression to be recognized, that are later classified with a Convolutional Neural Network. The methods are different in both parts, in the pre-processing and in the Convolutional Neural Network architecture.

The first stage of the methods is a pre-processing step that aims to extract the best set of features that describes the facial changes caused by an expression. Once the images are pre-processed they can be either used to train the network or to test it (i.e. recognition step). In the training step, a set of pre-processed images are given to the network with their respective labels so that the best set of network weights for classification can be found. In the testing step, the network is configured with the weight set found during the training and the recognitions are performed. The recognition outputs the confidence level of each expression. The maximum confidence level is used to infer the expression in the image. In order to increase the number of training samples a synthetic image generation method is used during the pre-processing stage (is important to mention that these synthetic image are not used int the test step).

In the following sections, a detailed description of the proposed methods are presented. It starts with the method that applies the intensity normalization in the pre-processing. In this section, a detailed description of the spatial normalization is also presented. Secondly, Section 3.2 presents the method that uses the prior knowledge of the neutral expression. The architectures of the Convolutional Neural Networks for both methods are also presented in their respective sections. Thirdly, a Convolutional Neural Network for Neutral

Expression Detection is presented. The latter is specially used in cases where the neutral expression (a restriction of the second method) is unknown.

3.1 Intensity Normalization Based Method

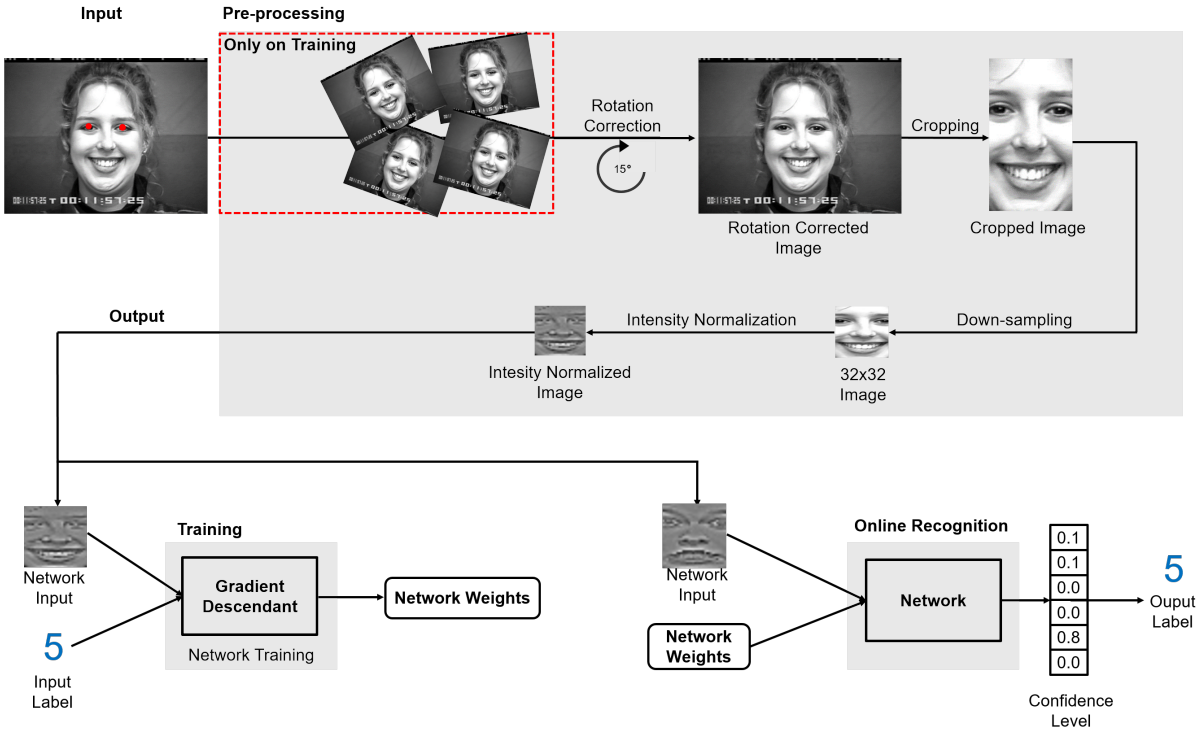

Figure 4: Intensity Normalization Method Overview. The system is divided in two main steps: training and testing. The training step takes as input an image with a face and its eyes locations. Firstly, during training, new images are synthetically generated to increase the database size. After that, a rotation correction is carried out to align the eyes with the horizontal axis. Then, a cropping is done to remove background information keeping only expression specific features. A down-sampling procedure is carried out to get the features in different images in the same location. Thereafter, an intensity normalization is applied to the image. The normalized images are used to train the Convolutional Neural Network. The output of the training step is a set of weights that achieve the best result with the training data. The testing step use the same methodology as the training step: spatial normalization, cropping, down-sampling and intensity normalization. Its output is a single number that represents one of the six basic expressions. The gray parts in the image are the parts of the proposed system.

In this section, an efficient method that performs the three learning stages in just one classifier (CNN) is presented. The only additional step required is a pre-process to normalize the images.

An overview of the method is shown in 4. The training and the testing have slightly

different workflows. For training, the system applies a sequence of: synthetic samples generation, spatial normalization (that comprises rotation correction, image cropping and down-sampling) and intensity normalization. For recognizing an unknown image (i.e. testing phase), the system applies a sequence of: spatial normalization and intensity normalization. The only difference between the image pre-processing steps of training and testing is the synthetic samples generation that is used in training only. The output of the training stage is a set of network weights that better separate the expression classes, while the output of the trained CNN is a single label number that express one of the expressions. The input, for both steps (training and testing), is a single face image with its eyes points.

3.1.1 Synthetic Samples Generation

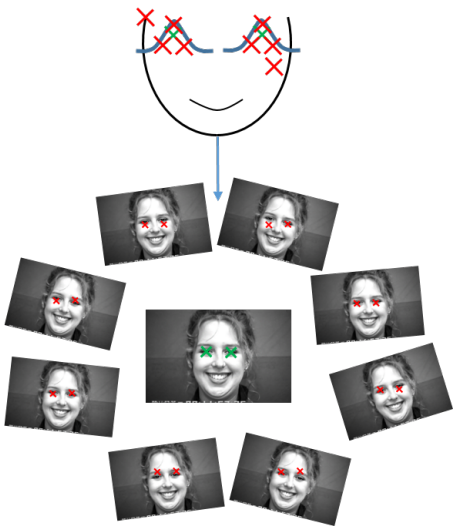

Figure 5: Synthetic Samples Generation and Normalization. In the top, the red points are the calculated eyes, around the original eye, in black. The new eye points are generated using a Gaussian distribution, having the original eye point as the center. For each point, a synthetic image is generated and then normalized, generating a range of rotated and scaled synthetic images, as can be seen in the bottom. This procedure intends to increase the database size and variation.

Even with the spatial normalization step, one can not guarantee that the final image will be exactly aligned with the horizontal axis due to, for example, a problem with the eyes detection. Fortunately, learning algorithms are very good at learning transformation invariance functions in controlled scenarios, but they usually need examples of such variation to be learned. Convolutional network networks are more suitable to work with large amount of data, which allow them to learn better the variation present in

the data. To address these problems while keeping the original database, Simard *et al.* in (SIMARD; STEINKRAUS; PLATT, 2003) propose some data augmentation operations to generate synthetic samples and consequently increase the variation of the database.

Simard *et. al.*, in (SIMARD; STEINKRAUS; PLATT, 2003), show the benefits of applying transformations like rotation and skewing to generate synthetic samples and increase the database size. Based on his work, in this work, a 2D Gaussian was used to include random noise in the location of the center of the eyes generating a modified (translated, rotated and/or scaled) version of the original image. The specific value of the Gaussian standard deviation needs to be carefully chosen, because a very small deviation could cause no variation in the original data and generate a lot of useless equal images, while a big deviation for each eye could introduce too much translation, rotation and/or scale noise in the images making the scenario more complex for the classifier to learn the expression features. In this work the Gaussian standard deviation used was 5 ($\sigma = 5$). For each original sample, 70 new synthetic samples were generated. It is important to note that the synthetic data is only used in the training.

Figure 5 shows the synthetic sample generation procedure. Given the original eyes center, a Gaussian distribution is centred in each eye and a new value is generated for the eyes center. The new eyes center position is therefore equivalent to the original one but disturbed by a Gaussian noise. As the new values given to the normalization procedure are not the real eyes center, the resulting images will be either disturbed by a translation, a rotation and/or a scale, or not disturbed at all.

3.1.2 Spatial Normalization

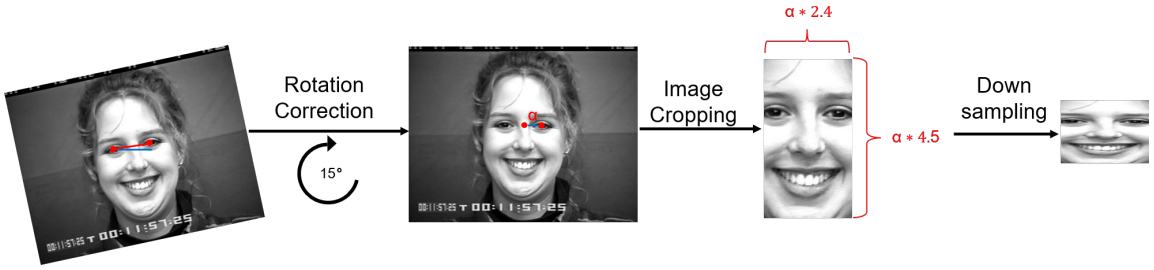

Figure 6: Spatial Normalization. The spatial normalization procedure comprises three steps: rotation correction, image cropping and down-sampling, in this order.

The face images in all the databases used, and even in real environments, vary in rotation, brightness, size, etc. in different images even for the same person. Those features

are independent of expression features and can affect the recognition rate significantly. Convolutional network networks could learn these features, but it would require a much large set of examples that we do not have available.

To address these problems and consequently reduce the complexity of the problem, a spatial normalization procedure is carried out in the images. This procedure helps the facial parts (eyes, mouth, nose and eyebrows) to be in the same pixel space helping the classifier to associate which image parts are related to each expression. This procedure comprises three steps: rotation correction, image cropping and down-sampling. Each step is described below.

Rotation Correction The first step of the normalization procedure is a rotation correction. To perform this correction, two informations are needed, the facial image and the center of both eyes. There are many methods available in the literature to detect eyes center (SARAGIH; LUCEY; COHN, 2010), (LI et al., 2006) and (CHOI et al., 1997). Based on these points, a geometric normalization comprising a rotation and a translation is carried out to align the two eyes center with the horizontal axis and to keep the face centralized in the image. The rotations and translations in the image are not related to the changes caused by an expression and therefore should be removed to avoid negatively affecting the recognition process. Figure 6 exemplifies the processes described here.

Image Cropping The second step of the normalization procedure is a cropping in the rotation corrected image. This step aims to keep the method focused only on expression specific regions, removing all background information and image patches that are not related to the expression (hair, ears, chin and forehead). These features could decrease the recognition accuracy because the classifier will need to handle more information and detect which features are related or not to the expression change. The cropping region is automatically delimited based on the inter-eyes distance, therefore no human intervention is needed. The region is delimited by a vertical factor of 4.5 applied to the distance between the eyes middle point and the right eye center. The horizontal cropping region is delimited by a factor of 2.4, the distance between the eyes middle point and the right eye center. These factor values were determined empirically, the same behavior could be achieved using the distance between the left eye center and the eyes middle point. The result of this step is shown in Figure 6.

Down-sampling The last step of the spatial normalization procedure is a down-sampling. As described earlier, with all facial parts in the same pixel space the classifier will need to handle with less variations. After the cropping step, the images will be of different sizes (this can happens because different subjects have different inter-eyes distance). Therefore, in this step, the images are down-sampled, using a linear interpolation, to 32×32 pixels (empirically defined) in order to remove the variation in face size and keep the facial parts in the same pixel space. The result of this step is shown in Figure 6.

3.1.3 Intensity Normalization

The image brightness and contrast can vary even in images of the same person in the same expression increasing, therefore, the variation in the feature vector. Such variations increase the complexity of the problem that the classifier has to solve for each expression. In order to reduce these issues an intensity normalization was applied. A method adapted from a bio-inspired technique described in (WANDELL, 1995), called contrastive equalization, was used. Basically, the normalization is a two step procedure: firstly a subtractive local contrast normalization is performed; and secondly, a divisive local contrast normalization is applied. In the first step, the value of every pixel is subtracted from a Gaussian-weighted average of its neighbors. In the second step, every pixel is divided by the standard deviation of its neighborhood. The neighborhood for both procedures uses a kernel of 7x7 pixels (empirically chosen). An example of this procedure is illustrated in Fig. 7.

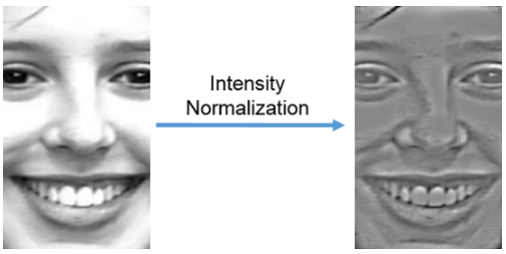

Figure 7: Illustration of the intensity normalization. The figure shows the image with the original intensity (left) and its intensity normalized version (right)

Equation 3.1 shows how each new pixel value is calculated in the intensity normalization procedure:

$$x' = \frac{x - \mu_{nhg\,x}}{\sigma_{nhg\,x}} \tag{3.1}$$

where, x' is the new pixel value, x is the original pixel value, $\mu_{nhg\,x}$ is the Gaussian-weighted average of the neighbors of x, and $\sigma_{nhg\,x}$ is the standard deviation of the neighbors of x.

3.1.4 Convolutional Neural Network for Facial Expression Classification of the Intensity Normalized Image

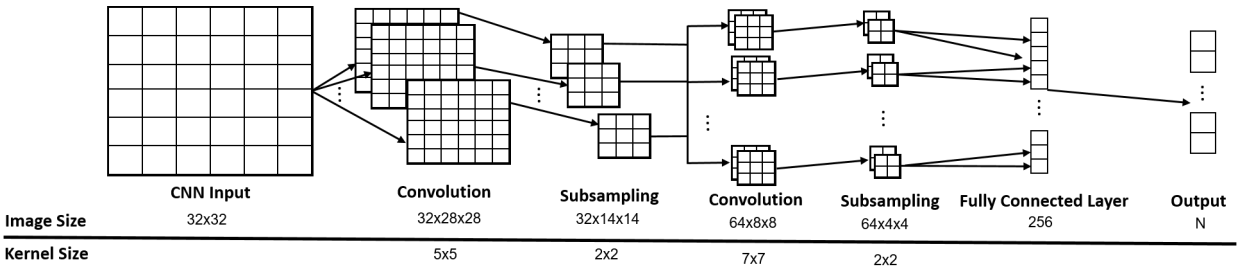

Figure 8: Architecture of the proposed Convolutional Neural Network for the current method. It comprises five layers: the first layer (convolution type) outputs 32 maps; the second layer (subsampling type) reduces the map size by half; the third layer (convolution type) outputs 64 maps for each input; the fourth layer (subsampling type) reduces the map once more by half; the fifth layer (fully connected type) and the final output with N nodes representing each one of the expression are responsible for classifying the facial image.

The architecture of our Convolutional Neural Network is represented in Figure 8. The network receives as input a 32×32 grayscale image and outputs the confidence of each expression. The class with the maximum value is used as the expression in the image. Our CNN architecture comprises 2 convolutional layers, 2 sub-sampling layers and one fully connected layer. The first layer of the CNN is a convolution layer, that applies a convolution kernel of 5x5 and outputs an image of 28x28 pixels. This layer is followed by a sub-sampling layer that uses max-pooling (with kernel size 2x2) to reduce the image to half of its size. Subsequently, a new convolution with a 7x7 kernel is applied to the feature vector and is followed by another sub-sampling, again with a 2x2 kernel. The output is given to a fully connected hidden layer that has 256 neurons. Finally, the network has six or seven output nodes (one for each expression that outputs their confidence level) that are fully connected to the previous layer.

The first layer of the network (a convolution layer) aims to extract elementary visual features, like oriented edges, end-point, corners and shapes in general, like described by *Lecun et al* in (LECUN et al., 1998). In our case the features detected are mainly the

shapes, corners and edges of eyes, eyebrow and libs. Once the features are detected, its exact location is not so important, just its relative position compared to the other features. For example, the absolute position of the eyebrows are not important, but their distances from the eyes are, because a big distance may indicate, for instance, the surprise expression. Not only this precise position is irrelevant but also this value can be a problem, if the system in bound to it, because the position can vary for different subjects. The second layer (a sub-sampling layer) reduces the spatial resolution of the feature map. According to *Lecun et al* in (LECUN et al., 1998), this operation aims to reduce the precision with which the position of the features extracted by the previous layer are encoded in the new map. The next two layers, one convolutional and one sub-sampling, aims to do the same operations that the first ones, but handling features in a lower level, recognizing contextual elements (face elements) instead of simple shapes, edges and corners. The concatenation of sets of convolution and sub-sampling layers achieve a high degree of invariance to geometric transformation of the input. The last hidden layer (a fully connected layer) receives the set of features learned and outputs the confidence level of the given features in each one of the expressions.

This network uses the Stochastic Gradient Descent method to calculate the synapses weights between the neurons, this method was proposed by *Buttou* (BUTTOU, 2012). The initial value of these synapses for the convolutions and for the fully connected layer are generate using the Xavier filler, proposed by *Glorot et al* in (GLOROT; BENGIO, 2010), that automatically determines the scale of initialization based on the number of input and output neurons. The loss is calculated using logistic function of the soft-max output (known as *SoftmaxWithLoss*). The activation function of the neurons is a ReLu (Rectified Linear unit), defined as $f(z) = max(z, 0)$. The ReLu function generally learns much faster in deep architectures (GLOROT; BORDES; BENGIO, 2011).

3.2 Neutral Subtraction Based Method

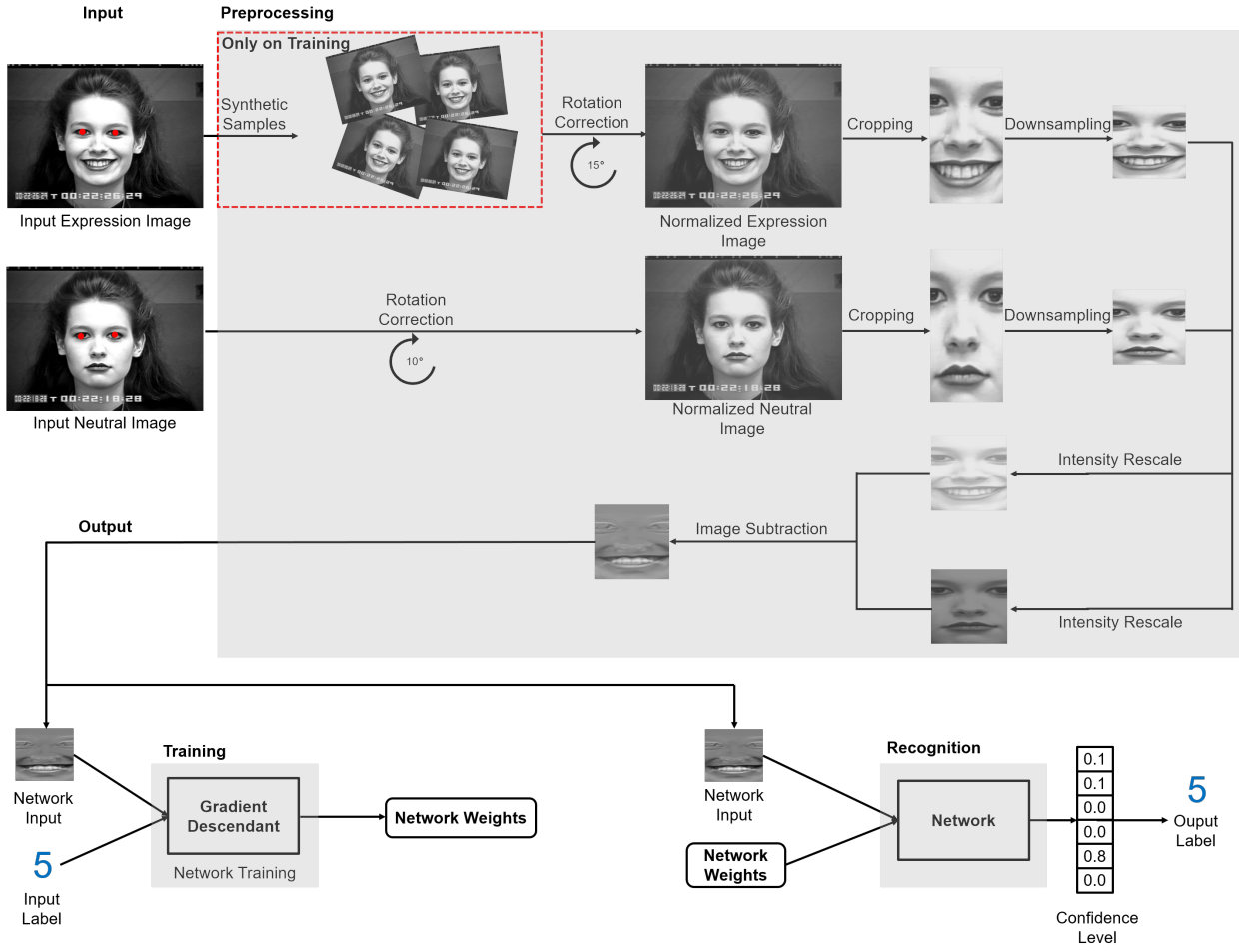

Figure 9: Neutral Subtraction Method Overview. The system receives as input two images of a subject with their respective eye location information, one neutral image and one image to be classified in one of the basic expressions. For training, the database is increased with synthetic samples before the actual preprocessing starts. During the preprocessing, the images are firstly aligned with the horizontal axis (the 15^o and the 10^o are just examples of possible rotations correction). The input images are cropped to focus only on expression specific regions. The cropped images are rescaled in the spatial domain. The rescaled images are then rescaled in the intensity domain to allow the subtraction of the neutral image from expression image. These final images can be either used to train or to test the network. The training receives the processed image and its label and outputs the weights of the network. The testing uses the learned weights to infer the expression of a given image.

In this section, we present a method that also performs the pre-processing step. But, instead of performing an intensity normalization, a neutral subtraction is carried out. The input of the system are two images, one of the subject in the neutral expression and another image of the same subject to be classified in one of the allowed expressions

(i.e. six basic expressions or six basic plus neutral depending on the case), with their respective eye center location. It is important to note that the neutral expression can also be an expression to be recognized and, in this case, both input images would be in the neutral expression. The pre-processing comprises a spatial normalization and a neutral subtraction. The Convolutional Neural Network used in the classification stage (after the pre-processing) comprises one convolutional layer, one sub-sampling layer and one fully connected layer. An overview of the method is illustrated in Figure 9.

The spatial normalization procedure follows the same steps of the method presented in Section 3.1, therefore it will not be succinctly discussed here. The remain of this section shows the neutral subtraction procedure and the architecture of the Convolutional Neural Network to recognize the expression.

3.2.1 Synthetic Sample Generation

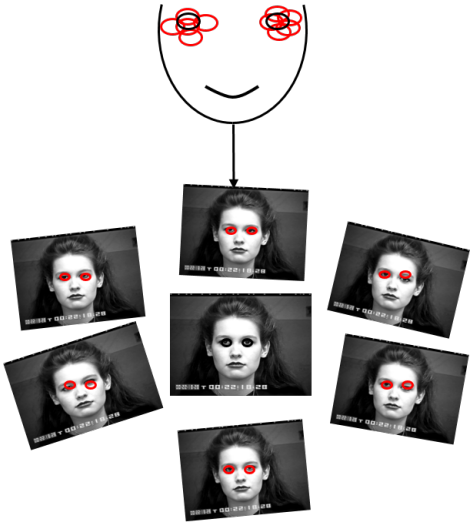

Figure 10: Synthetic Samples Generation and Normalization. In the top, the red circles are the calculated eyes, around the original eye, in black. The new eye points are generated using a Gaussian distribution, having the original eye point as the center. For each point, a synthetic image is generated and then normalized, generating a range of rotated and scaled synthetic images, as can be seen in the bottom. This procedure intends to increase the database size and variation.

In order to increase the database size and variation, synthetic generate samples are used in the training step. This operation is the same as the synthetic samples generation shown in Section 3.1, a succinctly description of this operation can be found there. An illustration of this operation can be seen in Figure 10.

3.2.2 Spatial Normalization

Figure 11: Spatial Normalization. The spatial normalization procedure comprises three steps: rotation correction, image cropping and down-sampling, in this order.

The first step of the pre-processing is a spatial normalization, that comprises rotation correction, image cropping and down-sampling. These operations are performed without human intervention and aims to reduce the problem complexity. A succinctly description of these operations has been already presented in Section 3.1.2. An illustration of this operations can be seen in Figure 11.

3.2.3 Neutral Subtraction

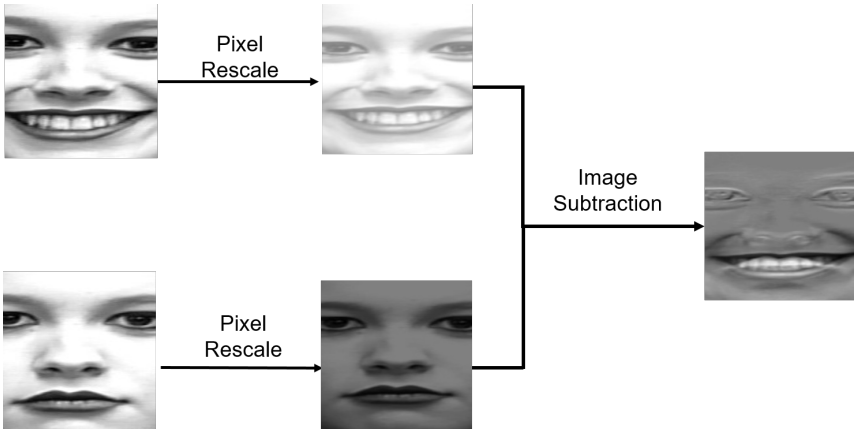

Figure 12: Image Subtraction. The pixels of the expression image are rescaled to be in the [128 256] interval resulting in a whiter image. The neutral image pixels are rescaled to be in the [1 127] interval resulting in a darker image. Once the images are subtracted, pixels with low variation between images will have values closer to 127.

After the spatial normalization procedure, the neutral image is subtracted from the expression image. Both images will be already spatially normalized, and therefore, their facial points will be in the same pixel space. With the subtraction, only the differences

related to the facial changes (i.e. the expression) will be present in the result image. As there is only expression data in the final image, this step simplifies the work of the classifier. It will not need to learn which parts of the image are relevant or not do determine the expression, because only the relevant parts are given to it. In addition, it removes the need for intensity normalization since the images are from the same subject and environmental conditions. An illustration of the subtraction process is shown in Figure 12.

The subtraction of two images could result in underflow, and therefore it is necessary to treat such cases before the actual subtraction. To avoid a possible underflow, the intensity values of the images are firstly rescaled. The pixels of the neutral image were scaled to be in the range [0 127], whereas the pixels of the expression image were scaled to be in the [128 256] range. This rescale operation ensures that the subtraction of the neutral image from the expression image will never result in underflow values. Equation 3.2 shows the complete subtraction operation.

$$Img = (Img_{exp} * 0.5 + 128) - Img_{ntr} * 0.5 \tag{3.2}$$

where, Img is the result image, Img_{exp} is the expression image, Img_{ntr} is the neutral image.

3.2.4 Convolutional Neural Network for Facial Expression Classification of the Neutral Subtracted Image

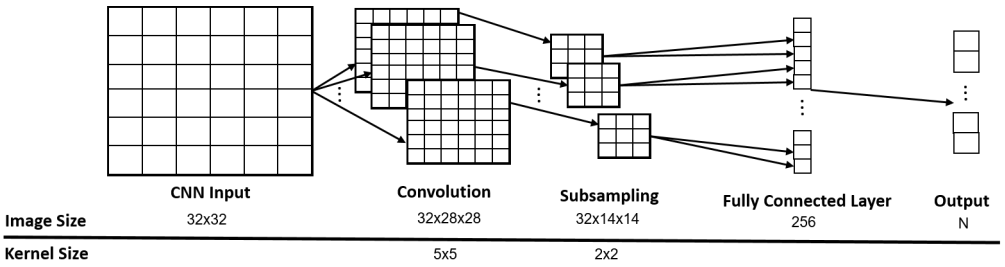

Figure 13: Proposed Convolutional Neural Network. The architecture of the network comprises tree layers: the first is a convolutional layer that produces 32 maps with a 5x5 kernel; the second is a sub-sampling layer that reduces the maps to a half; and, the third layer is a fully connected layer with 256 neurons. The output is composed by N nodes, each one representing one of the basic expressions to be considered.

The Convolutional Neural Network is used in the last stage of the proposed system, to classify the preprocessed image in one of the basic expressions. The architecture of this network was chosen based on extensive studies, shown in chapter 5, and is shown in Figure 13. The network receives as input the neutral difference image, which is a 32×32 pixels grayscale image, and outputs the confidence level of each class (expression). The class with the maximum confidence level is used as the expression in the image. The proposed extraction feature method does a lot of the work for the classifier making the learning task easier. Therefore, a simple network architecture is sufficient to achieve high expression recognition accuracy. The network architecture comprises only one convolutional layer, one sub-sampling layer and one fully connected layer. The first layer is a convolutional layer, that performs 32 convolutions with a 5×5 kernel, resulting in an image of 28×28 pixels. This layer is followed by a sub-sampling layer of 2×2 that reduces the image to a half of the size using a max-pooling approach. The output is given to a fully connected layer that has 256 neurons. The network has one output node for each expression and they are fully connected to the previous layer.

The first layer of the proposed network, a convolution layer, aims to extract visual features like oriented edges, endpoints, corners and shapes in general. These visual features are mainly detected by the convolution layers (LECUN et al., 1998). The second layer of the network, a sub-sampling layer, aims to reduces the spatial resolution of the image. As described by *Lecun et al* in (LECUN et al., 1998) sub-sampling layers increase the position invariance of the extracted features in the previous convolution layer. Indeed, after the detection of the features, its exact location is not so important, but its relative distance to others features is relevant. For example, the absolute position of the lips are not so important, but their distance to each other is relevant, because a big distance between the bottom and the top lips may indicate, for example, the surprise expression.

This network also uses the Stochastic Gradient Descent method to calculate the synapses weights between the neurons. The initial value of these synapses for the convolutions and for the fully connected layer are generate using the Xavier filler, like the network presented in Section 3.1.4.

3.3 Neutral Expression Detection

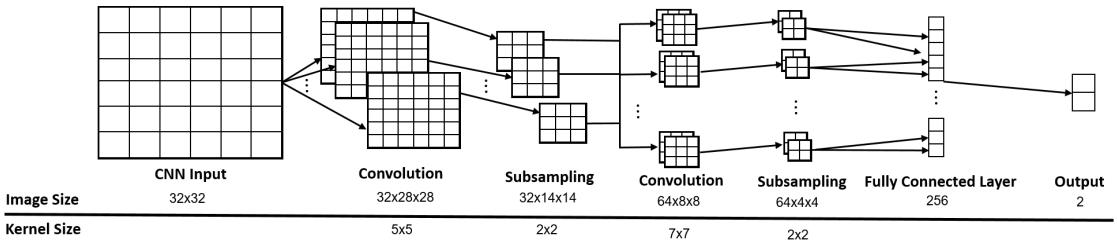

Figure 14: Convolutional Neural Network for Neutral Expression Detection. The architecture of the network comprises five layers: the first is a convolutional layer that produces 32 maps with a 5×5 kernel; the second is a sub-sampling layer that reduces the maps to a half; the third is another convolutional layer that outputs 64 maps with a 7×7 kernel; the fourth is a sub-sampling layer that reduces the image to a half again; and, the fifth layer is a fully connected layer with 256 neurons. The output comprises two nodes, one representing the neutral expression and the other representing the non-neutral expressions.

As described earlier, the system proposed in Section 3.2 was developed to work on environments where the neutral expression is known. But, even without this information our approach can be easily combined with a neutral expression detector that aims to select a neutral expression in a set of expression images. For completeness of the work and to show the viability of this approach, we show an example of a Convolutional Neural Network architecture that can be used to detect whether a given facial image is of a neutral expression image or not.

This network follows a similar approach (pre-processing operations)to the one presented in Section 3.1. The input of the network is a grayscale, 32×32 pixels preprocessed image (with rotation correction, cropping, sub-sampling and contrastive normalization, but without subtraction). The architecture of this network comprises two convolution layers and two sub-sampling layers. The first convolution layer performs 32 convolutions with a 5×5 kernel size and is followed by a sub-sampling layer with a 2×2 kernel that reduces the map to a half of the size using a max-sampling function. The output of the first sub-sampling connects to a new convolution that generates 64 new maps with a 7×7 kernel size and is followed by a sub-sampling layer with a 2×2 kernel that again reduces the map size by half. This layer is followed by a fully connected layer with 256 neurons, that calculates the confidence level of the image in each class (neutral and non-neutral). This architecture is shown in Figure 14.

4 Experimental Methodology

In this chapter, we present the experimental methodology used to evaluate the method proposed in chapter 3. Firstly, we present the three databases used to perform the experiments. Secondly, the methodology of the experiments are described. Finally, the metrics used to compute the accuracy are explained.

4.1 Databases

Three different databases were used to validate the method: the Extended Cohn-Kanade database (CK+) (LUCEY et al., 2010), the Japanese Female Facial Expressions (JAFFE) (LYONS; BUDYNEK; AKAMATSU, 1999) database and the Binghamton University 3D Facial Expression (BU-3DFE) database (YIN et al., 2006).

The Extended Cohn-Kanade (CK+) (LUCEY et al., 2010) database contains 497 sequences of 100 subjects. Each sequence contains about fifteen images and starts with the neutral expression of a subject and proceeds to a peek expression. All images in the dataset are 640 by 480 pixel arrays with 8-bit precision for grayscale values. Each image has a descriptor file with its facial points, these points were used to normalize the facial expression image. The facial points in the database are coded using the Facial Action Coding System (FACS) (EKMAN; FRIESEN, 1978). Active Appearance Models (AAMs) was used to automatically extract these facial points. The database contains images from the following expressions: neutral, angry, contempt, disgust, fear, happy, sad and surprise. To do a fair comparison with the state-of-the-art methods (LOPES; AGUIAR; SANTOS, 2015), (LIU et al., 2014) and (SHAN; GONG; MCOWAN, 2009) the contempt expression was not used. In this database, each sequence contains a lot of images of a subject in an expression, resulting in a lot of images very similar to each other. As done in (LOPES; AGUIAR; SANTOS, 2015), (LIU et al., 2014) and (SHAN; GONG; MCOWAN, 2009) the training and testing database were created by selecting only the last three frames of each sequence for the expression and one frame (usually the first one) of each sequence

for the neutral, resulting in a database with about 2,100 samples (without the synthetic samples) and 147,000 samples (with the synthetic samples). Some examples of the CK+ database images are shown in Figure 15.

Figure 15: Example of the images in the CK+ database. In (1) the subject is in the neutral expression. In (2) the subject is in the surprise expression. In (3) the subject is in the disgust expression. In (4) the subject is in the fear expression.

To verify the generalization of the proposed method some cross-database experiments were also performed. These experiments used the JAFFE database (LYONS; BUDYNEK; AKAMATSU, 1999) and the BU-3DFE database (YIN et al., 2006). The JAFFE database consists of 213 images from 10 Japanese female subjects. All images in the dataset are 256 by 256 pixel arrays with 8-bit precision for grayscale values. In this database there are about 4 images in each one of the six basic expressions and one image of the neutral expression from each subject, resulting in a database with about 213 samples (without the synthetic samples) and 14,910 samples (with the synthetic samples). Some examples of the JAFFE database images are shown in Figure 16.

Figure 16: Example of the images in the JAFFE database. In (1) the subject is in the surprise expression. In (2) the subject is in the happy expression. In (3) the subject is in the sad expression. In (4) the subject is in the sad expression.

The BU-3DFE (YIN et al., 2006) contains about 64 subjects (56% female and 44% male), with age ranging between 18 years to 70 years old, with a variety of ethnic/racial ancestries, including White, Black, East-Asian, Middle-east Asian, Indian and Hispanic

Latino. All images in the dataset are 156 by 209 pixel arrays. The database has about 1344 samples (without the synthetic samples) and 94,080 samples (with the synthetic samples). Some examples of the BU-3DFE database images are shown in Figure 17. These two databases (JAFFE and BU-3DFE) do not have the facial key points like the CK+ database. Therefore, in this work, the marking of the eyes center of the JAFFE was performed manually, whereas the marking of the BU-3DFE was performed with the method proposed in (SARAGIH; LUCEY; COHN, 2010).

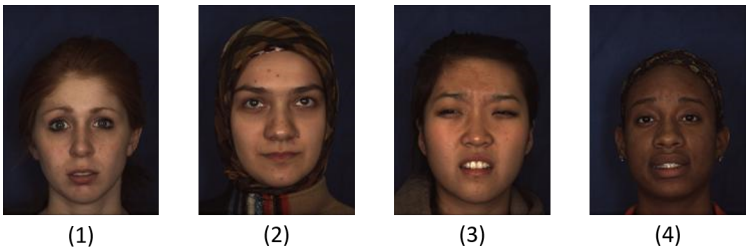

Figure 17: Example of the images in the BU-3DFE database.In (1) the subject is in the fear expression. In (2) the subject is in the neutral expression. In (3) the subject is in the fear expression. In (4) the subject is in the fear expression.

4.2 Evaluation Methodology

As discussed before, in this work, two main classes of experiments were performed: experiments training and testing in the same database (CK+ or JAFFE or BU-3DFE) and experiments training in one database (CK+) and testing in another (JAFFE or BU-3DFE), i.e. cross-database experiment.

To perform the experiments, the databases used in each experiment were divided into three sets: Training set, which is used to train the system; Validation set, which is used to dynamically tune the meta-parameters of the system (e.g. choose the best network weights out of 10 runs with random training samples presentation order); and the test set, which is used to actually measure the accuracy of the system. In all experiments, it is ensured that there is no subject overlap among the three sets. Since our method requires a additional neutral image as input, one of the neutral images of each subject was separated to perform the subtraction, the other images were used as expression image to be recognized for the relevant experiments (i.e. 7-expression experiments).

4.2.1 Same Database Evaluation

This evaluation configuration aims to measure the performance of the system with the CK+ database. As done by *Liu et al.* in (LIU et al., 2014) and by *Lopes et al.* in (LOPES; AGUIAR; SANTOS, 2015) the database was separated in eight groups of non-overlapping subject. Each group has about 12 subjects. In our experiments these groups were separated in three main sets: training set, validation set and test set. The training set is composed by seven groups, whereas the validation set and test set share the eighth group. The eighth group also contains about 12 subjects (as all others), where 11 subjects are used for validation and 1 subject for testing. Our experiments follow a k-fold cross-validation configuration, in which, each time, one group is separated for validation/test and the other seven for training. When a group is selected for validation/test, a leave-one-out procedure is performed within the 12 subjects (having 11 for validation and 1 for test).

For each configuration of validation and test subjects, the training is carried out 10 times, changing the presentation order of the training images. The validation group is used to select the best epoch for each run during training and to select the run with the best presenting order. Based on these informations (best epoch and presenting order), the best network weights are selected and used to compute the accuracy of the test set. With this experimental configuration, the training of the network is performed about 960 times (8 groups $*$ 12 subjects per group $*$ 10 times with different presenting order).

The experiments performed in the BU-3DFE database follows the same approach of the CK+ database, the only difference is that instead of twelve subjects in each group, this database has about eight subjects per group. With this experimental configuration, the training of the network is performed about 640 time (8 groups $*$ 8 subjects per group $*$ 10 times with different presenting order).

In the JAFFE database, a slight different experiment methodology was employed. Because this dataset is much smaller, only ten subjects, each group contains images of just one subject. The data is separated in the following way: eight groups for training, one for validation and one for test. The remainder keeps the same, it uses the k-fold configuration. For each configuration the training is performed ten times, with different presentation orders. With this experimental configuration, the training of the network is performed about 100 times (10 groups $*$ 1 subjects per group $*$ 10 times with different presenting order).

4.2.2 Cross-Database Evaluation

In real environments, the test database may be very different from the train database, varying the background, color intensity, light and others. Thus, a cross-database test could be a more fair method to evaluate the accuracy. Therefore cross-database experiments were performed, using the BU-3DFE database and the JAFFE database. In both experiments, the train database was the CK+, whereas the test database was the BU-3DFE or the JAFFE.

In this evaluation configuration, seven groups of the CK+ database were used to train the network and one was used to be the validation set (to choose the best network weights based on the best epoch presentation order of the training set). The training was done eight times, each one with a different validation set. We run each configuration (training set plus validation set) ten times, each one with a different presenting order in the training samples. The result in the cross-database experiment is computed as an average of the 8 runs (8 groups configuration $*$ 1 best presenting order) showing all the BU-3DFE or JAFFE images in each run.

4.3 Accuracy Metrics

To allow for a fair comparison of the presented method with the literature, the accuracy was computed in two different ways. In the first, one classifier for all basic expression is used. The accuracy is computed simply using the average, C_{nclass}, of the n-classes classifier accuracy per expression, $C_{nclassE}$, i.e. number of hits of an expression per amount of data of that expression, see Eq. (4.1).

$$C_{nclass} = \frac{\sum_1^n C_{nclassE}}{n}, C_{nclassE} = \frac{Hit_E}{T_E} \tag{4.1}$$

where Hit_E is the number of hits in the expression E, T_E is total number of samples of that expression and n is the number of expressions to be considered.

In the second, one binary classifier for each expression performs a one-versus-all classification, as proposed in (LIU et al., 2014). Using this approach, the images are presented to n binary classifiers, where n is the number of expression being classified. Each classifier aims to answer "yes" if the image contains one specific expression, or "no" otherwise. For example, if one image contains the surprise expression, the surprise classifier should answers "yes" and all the others five classifiers should answer "no". The only difference

for this classifier from the architecture presented in chapter 3 is that only two outputs are required for each classifier. The accuracy is computed using the average, C_{bin}, of the binary classifier accuracy per expression, C_{binE}, i.e. the number of hits of an expression plus the number of hits of non expression divided per total amount of data, see Eq. (4.2).

$$C_{bin} = \frac{\sum_1^n C_{binE}}{n}, C_{binE} = \frac{Hit_E + Hit_{NE}}{T} \tag{4.2}$$

where Hit_E is the number of hits in the expression E, i.e. number of times the classifier E responded "yes" and the tested image was of the expression E. Hit_{NE} is the number of times the classifier E responded "no" and the tested image was not the expression E. T is the total number of tested images and n is the number of expressions to be considered.

4.4 Experiments

A complete experiments set was performed for both methods proposed in Sections 3.1 and 3.2. Besides the accuracy evaluation, a set of tunning experiments were also performed. These experiments aims to select the best parameters configuration for the pre-processing steps and for the Convolutional Neural Networks. The tunning experiments uses just a subset of the CK+ database.

To the intensity normalization method, presented in Section 3.1, the tunning experiments comprises experiments evaluating the accuracy increase with each pre-processing step and combinations between them. After that, experiments to evaluate the accuracy in the CK+ database, BU-3DFE database and JAFFE database are shown, with cross-database experiments.

To the neutral subtraction method presented in Section 3.2, the tunning experiments comprises experiments evaluating the parameters of the Convolutional Neural Network (architecture, momentum and learning rate) and the parameters of the pre-processing step (Gaussian standard deviation, amount of synthetic samples, presenting order). Finally, the accuracy in the CK+ database, BU-3DFE database and JAFFE database are shown, with cross-database experiments.

5 Results and Discussion

In this chapter, we present the experiments performed for all three methods: the intensity normalization based method (discussed in Section 3.1), the neutral subtraction based method (discussed in Section 3.2) and to the neutral detection (discussed in Section 3.3). These experiments follows the configurations presented in chapter 4. In additional, each Section presents tunning experiments, that aims to find the best configuration of the: pre-processing operations, amount of synthetic samples, Gaussian standard deviation, network parameters, and others. In the first Section (5.1), we investigate the impact of every pre-processing operation in the intensity normalization method accuracy and present the results for this method according to the evaluation configuration described in chapter 4. In the second Section (5.2), we present the impact of networks parameters (momentum, learning rate, etc), the network architecture and others in the neutral subtraction method and present the results for this method according to the evaluation configuration described in chapter 4. In the third Section 5.3, the results for the neutral expression detector are presented. Finally, the proposed approaches are compared with the state-of-the-art methods in the literature and their limitations are discussed.

The implementation of the pre-processing steps was done in-house using C++ and OpenCV, and we used a GPU based CNN library, also in C++, called Caffe (JIA et al., 2014). All the experiments were carried out using an Intel Core i7 3.4 GHz with a NVIDA GeForce GTX 660 CUDA Capable that has 1.5Gb of memory in the GPU and 960 cores. The environment of the experiments was a Linux Ubuntu 12.04, with the NVIDIA CUDA Framework 6.5 and the cuDNN library installed.

5.1 Intensity Normalization Experiments

In this Section, a study is presented showing the impact of every normalization step in the accuracy of the method presented in Section 3.1 and the results of this method for three database are shown. Firstly, the results of the tuning experiments of the influence

of each pre-processing step is presented. Finally, the results with different databases are shown and discussed in details.

5.1.1 Pre-processing Tunning

As described earlier, the proposed method combines a pre-processing step, that aims to remove non-expression specific features of a facial image and a Convolutional Neural Network to classify this preprocessed image in one of the six (or seven) expression. In this section, we present the impact in the classification accuracy of each operation in the preprocessing step. As these tests aims only to show the impact of the operations, a simplified version of our test methodology (presented in Section 4.2) was employed. Here, we randomly generate the order of presenting of the samples to the network and use a simple k-fold cross validation between the 8 groups of the CK+ database. The database was divided in two set, training (with 7 of the groups) and test (with 1 of the groups). The training was performed 8 times using only 2000 epochs for each of them. The accuracy was computed per expression (using all hits of all runs divided by the number of images of the expression E in the database, $C_{6classE}$) and overall average for all expressions (using all hits of all runs divided by the number of images in the database, C_{6class}).

a) No Pre-processing This first experiment was carried out using the original database, without any intervention or image pre-processing, just a down-sampling to the image be of the same size as the input of the CNN. In this experiment, the average accuracy for all expressions was $C_{6class} = 53.50\%$. The accuracy per expression is shown in Table 1. The accuracy shown is an average of Eq. 4.1 for all runs.

As it can be seen in Table 1, using only the CNN without any image pre-processing, the recognition rate is very low compared to the state-of-the-art methods. It can happen because the variation and amount of samples in the CK+ database could not be so higher to the Convolutional Neural Network learn how to deal with pose, environment and subject variance.

b) Image Cropping Using the raw input (the original image without any pre-processing), an accuracy of 53.50% was achieved. This is a low accuracy rate compared with the state-of-the-art methods. In order to increase this result, as explained in Section 3.1.2, a cropping (without human intervention) is performed in the image to remove non-expression specific regions in the image, in both training and testing steps. The average

accuracy for all expressions was $C_{6class} = 71.60\%$. The accuracy per expression is shown in Table 1. Here the down-sampling is also performed, because the input of the proposed network is a fixed 32×32 pixels image.

Compared with the result shown before, we can note a significantly increase of the recognition rate by adding only the cropping processes. The main reason of the accuracy increase is that with the cropping we remove a lot of information that the classifier will need to handle, and infer that is useless to determine the subject expression (i.e., we make easier the work of the classifier).

c) Rotation Correction Just cropping the image we can note an high accuracy increase, from 53.50% to 71.60%. Despite this increase, the final result still very low compared with the literature. Motivate to increase this result, as explained in Section 3.1.2, a rotation correction (and the down-sampling) is performed in the image to remove rotations that are not related to expression facial changes (that can be pose-specific or caused by a camera movement), in both training and testing steps. The average accuracy for all expressions was $C_{6class} = 61.55\%$. The accuracy per expression is shown in Table 1.

Note that, this result is applying just the rotation correction, but not the cropping. Compared with the result of no pre-processing we can note an increase of the accuracy in about 8.00%. This increase is caused by the lower variation that the network needs to handle. With the rotation correction the facial elements (eyes, mouth, eyebrows) stay mostly in the same pixel space, but still has the influence of the background.

d) Spatial Normalization As seen before, the image cropping and the rotation correction applied separately, increase the classifier accuracy, when compared to no pre-processing, from 53.50% to 71.60% and from 53.50% to 61.55% respectively. This happens because both procedures reduce the problem complexity. Here we discuss the full spatial normalization, composed by the image cropping, rotation correction and down-sampling. Including these both operations (image cropping and rotation correction), in the training and testing steps, the average accuracy for all expressions was $C_{6class} = 87.80\%$. The accuracy per expression is shown in Table 1.

As expected, joining both procedures in the pre-processing step increase the accuracy. Indeed, compared with the raw input this increase was remarkable, $C_{6class} = 87.86\%$ instead of only $C_{6class} = 53.57\%$. This happens because a lot of variation not related to

the expression was removed from the image. Although the Convolutional Neural Network could handle these variations, we need a bigger database (that we do not have) and maybe a more complex architecture.

e) Intensity Normalization The spatial normalization procedure increased a lot the system accuracy, from 53.50% to 87.80%. Now that we have the image normalized in the spatial domain, its normalization in the intensity domain could also increase the system accuracy. The intensity normalization is used to remove brightness variation in the images in both steps, training and testing. This experiment was performed using just the intensity normalization. It uses the same methodology described before. The average accuracy for all expressions was $C_{6class} = 57.00\%$. The accuracy per expression is shown in Table 1.

As it can be seen, just applying the intensity normalization classifier accuracy was also increased. However, compared with the increase achieved by the spatial normalization the accuracy is very low.

f) Spatial and Intensity Normalization As seen before, the spatial normalization and the intensity normalization applied separately, increase the classifier accuracy, when compared to no pre-processing, from 53.50% to 87.80% and from 53.50% to 57.55% respectively. Putting the spatial (rotation correction, cropping and down-sampling) and intensity normalization together, we remove a big part of the variations unrelated to the facial expression and leave just the expression specific variation that is not related to the pose or environment. This experiment was done using the same methodology described before. The average accuracy for all expression was $C_{6class} = 86.67\%$. The accuracy per expression is shown in Table 1.

As it can be seen, the accuracy of applying both normalization procedures is lower than the one that uses only the spatial normalization. The result of the fear expression has a very low accuracy, which reduces the overall recognition average. To verify the need for the intensity normalization, a new experiment using only the spatial normalization procedure and the synthetic sample generation was performed and is presented below.

g) Spatial Normalization and Synthetic Samples The result of the spatial and intensity normalization and only the spatial normalization gives a false impression that the intensity normalization might decrease the accuracy of the method - since the result of

applying only the spatial normalization is better than the result with both normalizations. To verify this suspicion, a new experiment was conducted using only the spatial normalization procedure and the additional synthetic samples for training. This experiment is done using the same methodology described before. For the synthetic sample generation, thirty more samples were generated for each image using a Gaussian standard deviation of 3 pixels ($\theta = 3$). The average accuracy for all expression was $C_{6class} = 87.10\%$. The accuracy per expression is shown in Table 1.

This result increase the accuracy of applying just the spatial normalization (87.86%). But, as it can be seen in the next result, is lower than applying both normalizations and the synthetic samples generation. It show us that the synthetic sample generation procedure indeed increase the robustness of the classifier (motivated by the increase of the samples and its variation).

h) Spatial Normalization, Intensity Normalization and Synthetic Samples

The best result achieved in our method applies the three image pre-pocessing steps: spatial normalization, intensity normalization and synthetic samples generation. This experiment is done using the same methodology described before. The average accuracy for all expression was $C_{6class} = 89.76\%$. The accuracy per expression is shown in Table 1. The accuracy of this experiment shows that joining the three techniques (spatial normalization, intensity normalization and synthetic samples) is better than using only the spatial normalization and the synthetic samples presented previously.

Based on these results, it can be seen that the intensity equalization procedure applied on synthetic samples generation (already spatial normalized) increases the method's accuracy. It happens because the more variation (included by the synthetic samples), the better the classifier learns how to deal with different environmental and pose configurations.

Table 1 shows the mean accuracy for each expression using all the preprocessing steps already discussed. In a) no pre-processing is used, in b) just the cropping is performed, in c) just the rotation correction is employed, in d) the spatial normalization (cropping and rotation correction) is used, in e) only the intensity normalization is performed, in f) both normalizations (spatial and intensity) are applied, g) spatial normalization using the synthetic samples are used and in h) both normalizations and the synthetic samples are used. The accuracy is computed using the six class classifier ($C_{6classE}$).

Table 1: Preprocessing steps accuracy details.

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**	**Average**
a)	28.10%	51.23%	17.91%	70.68%	20.99%	77.52%	53.57%
b)	68.60%	79.01%	23.37%	86.39%	23.46%	87.16%	71.67%
c)	17.17%	79.09%	00.00%	48.92%	05.05%	91.25%	61.55%
d)	81.82%	90.74%	73.13%	95.81%	66.67%	94.50%	87.86%
e)	27.27%	52.94%	08.22%	79.10%	18.29%	85.54%	57.00%
f)	78.51%	93.21%	53.73%	95.29%	75.31	93.12%	86.67%
g)	86.05%	88.30%	69.33%	96.60%	77.11%	95.34%	87.10%
h)	79.34%	94.44%	73.13%	99.48%	72.84%	94.94%	89.76%

In *a*) no pre-processing is used, in *b*) just the cropping is performed, in *c*) just the rotation correction is employed, in *d*) the spatial normalization (cropping and rotation correction) is used, in *e*) only the intensity normalization is performed, in *f*) both normalizations (spatial and intensity) are applied, in *g*) spatial normalization using the synthetic samples are used and in *h*) both normalizations and the synthetic samples are used.

A graphical evolution of the accuracy depending of the pre-processing steps can be seen in Figure 18.

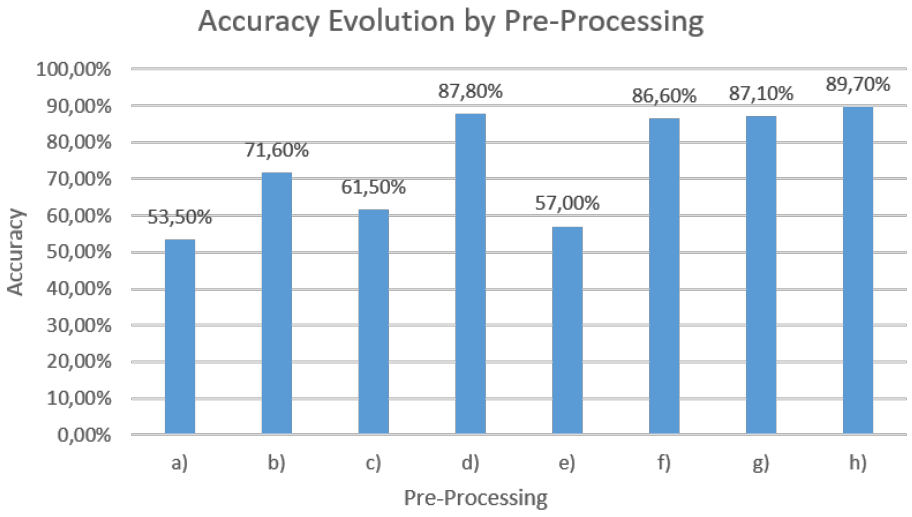

Figure 18: Evolution of the accuracy based on the pre-processing steps. In *a*) no pre-processing is used, in *b*) just the cropping is performed, in *c*) just the rotation correction is employed, in *d*) the spatial normalization (cropping and rotation correction) is used, in *e*) only the intensity normalization is performed, in *f*) both normalizations (spatial and intensity) are applied, *g*) spatial normalization using the synthetic samples are used and in *h*) both normalizations and the synthetic samples are used.

5.1.2 Results

The results of the system accuracy considering the three different databases are shown below. For all databases, the preprocessing step (rotation correction, cropping, down-

sampling and intensity normalization) took only 0.02 second and the network recognition (classification step) took in average 0.01 second per image.

As discussed before, a simplified training/testing methodology was used to evaluate the impact of the pre-processing steps. In contrast to the tunning experiments that used only training and test sets, this section the accuracy is computed using the configurations shown in chapter 4.

Training with CK+ and Test with CK+ Table 2 shows the best result achieved (using both normalizations and the synthetic samples) using both classifiers. As can be seen the binary classifiers approach increases the accuracy. It happens because in this approach the hit can be achieved n times (one for each expression), instead of using just one classifier, where each sample has just one chance to be properly classified. The binary classifier approach was employed to allow a fair comparison with some methods in the literature that just report this results, but we think that the C_{nclass} classifier (C_{6class}) is a more fair evaluation method.

Using the experiment configuration described in 4.2.1 for this database, the training of the network is performed about 960 times (8 groups $*$ 12 subjects per group $*$ 10 times with different presenting order). The time required to train the network each time was about only 2 minutes, resulting in a total training time (using the k-fold configuration) of 32 hours.

Table 2: Accuracy for both classifiers using all processing steps and the synthetic samples for six expression on the CK+ database.

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
$\boldsymbol{C_{6classE}}$	93.33%	100.00%	96.00%	98.55%	84.52%	99.20%
$\boldsymbol{C_{binE}}$	98.27%	99.37%	99.24%	99.68%	98.17%	98.81%
Average of $\boldsymbol{C_{6class}}$: 96.76%						
Average of $\boldsymbol{C_{bin}}$: 98.92%						

The training parameters that achieves the results shown in Table 2 is shown in Table 3. These same parameters are used in the experiments on other databases for this method, shown below.

Table 3: Training Parameters

Parameter	**Value**
Momentum	0.95
Learning Rate	0.01
Epochs	10000
Loss Funtion	Logistic Regression
Gaussian Standard Deviation	3
Synthetic Samples Amount	30

Using the result shown in Table 2 the confusion matrix shown in Table 4 was created for the C_{6class} classifier.

Table 4: Confusion Matrix using both normalizations and synthetic samples for six expressions on the CK+ database.

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
Angry	**126**	6	2	0	1	0
Disgust	0	**177**	0	0	0	0
Fear	0	0	**72**	0	3	0
Happy	3	0	0	**204**	0	0
Sad	3	0	1	0	**71**	9
Surprise	1	0	1	0	0	**247**

Based on the results of the C_{6class} classifier, we can note that the disgust, happy and surprise expressions achieves an accuracy rate higher than 98%. While the angry and fear expression was about 93% and 96% respectively. The sad expression achieves the smallest recognition rate, with only 84.52%. Looking the confusion matrix, the sad expression was confused in the majority of the time with the surprise expression. This shows that the features of these two expression are not well separated in the pixel space, i.e. they are very similar to each other in some cases. The standard deviation aims to measure the amount of variation between data values. The standard deviation in the accuracy, when considering one result per group (discussed in Section 4), for the C_{6class} classifier between the eight groups, was $\sigma = 0.07$. Figure 19 shows some examples of the misclassification.

Figure 19: In (1) the expected expression was sad, but the method returned fear. In (2) the expected expression was angry, but the method returned fear. In (3) the expected expression was sad, but the method returned angry. In (4) the expected expression was angry, but the method returned sad.

Figure 20 shows a illustration of the learned kernels and the generated maps for each convolution layer. In the first convolution layer, the input image is processed by the 32 learned kernels and generates 32 output maps. In the second convolution layer, the 64 learned kernels are used to generate new maps for each one of the 32 maps of the previous layer. The kernels shown in Figure 20 were learned in the training using the CK+ database for the six basic expression.

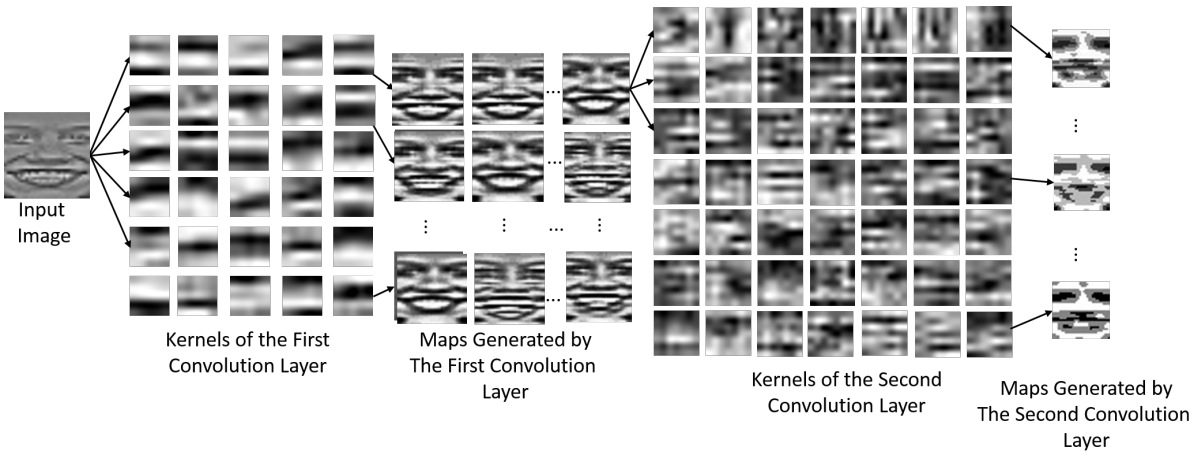

Figure 20: Illustration of the learned kernels and the generated maps for each convolution layer. In the first convolution layer, the input image is processed by the 32 learned kernels and generates 32 output maps. In the second convolution layer, the 64 learned kernels are used to generate new maps for each one of the 32 maps of the previous layer. The sub-sampling layers are not represented in this image. Only a subset of the 32 kernels for the first layer and of the 64 kernels for the second layer are shown. The generated maps were equalized to allow for a better visualization.

Instead of recognizing only six expressions, we can also recognize the neutral expression, resulting in a classifier that recognizes seven expressions. The result of the seven

expression classifier to the CK+ database, using the same methodology os the six expressions is show in Table 5

Table 5: Accuracy for both classifiers using all processing steps and the synthetic samples for seven expression.

	Neutral	**Angry**	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
$C_{7classE}$	95.15%	91.11%	99.44%	92.00%	100.0%	82.14%	98.80%
C_{binE}	97.49%	97.82%	99.76%	99.11%	99.76%	98.79%	98.87%
Average of C_{7class}: 95.79%							
Average of C_{bin}: 98.80%							

As it can be seen, for the binary classifier approach, we have a slight decrease in accuracy, from 98.90% to 98.80%. On the other hand, in the seven-class classifier the decrease was bigger, from 96.7% to 95.7%. It happens because, in the seven-class classifier approach, one more output was included in the network. On the other hand, in the binary-class approach, one new classifier was inserted, keeping the others unchanged. The confusion matrix for the seven expressions is shown in Table 6.

Table 6: Confusion Matrix using both normalizations and synthetic samples for seven expressions on the CK+ database.

	Neutral	**Angry**	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
Neutral	**294**	11	1	1	0	0	2
Angry	8	**123**	1	0	3	0	0
Disgust	0	1	**176**	0	0	0	0
Fear	6	0	0	**69**	0	0	0
Happy	0	0	0	0	**207**	0	0
Sad	0	3	0	3	0	**69**	9
Surprise	2	0	0	1	0	0	**246**

Training with BU-3DFE or CK+ and Tests with the BU-3DFE This experiment follows the same approach as the CK+ database, the only difference is that in the BU-3DFE database the groups have about only eight subjects. The results of this experiment is computed as an average of the 64 runs (8 groups, k-fold $*$ 8 subjects per group, leave-one-out $*$ 1, best configuration of the 10 times with different presenting order). The result for six and seven expression for both classifiers is shown in Table 7.

As it can be seen, the accuracy for the BU-3DFE decreased compared with the CK+ database. One possible reason is that this database has more subjects from different ethnicities and light conditions, and is smaller than the CK+. In addition, we have a

Table 7: BU-3DFE Accuracy using six and seven (six basic plus neutral) expressions.

Classifier	**6-expressions (%)**	**7-expressions (%)**
C_{nclass}	72.89	71.62
C_{bin}	90.96	91.89

increase in the accuracy for the C_{bin} classifier on seven expressions, whereas we have a decrease in accuracy for the C_{nclass} classifier on seven expressions. The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 28 and in Table 29 respectively, in the appendices (B.1).

A more fair evaluation of the proposed method in real environments is the cross-database experiments, i.e. train the method with one database and train with another (in this case the BU-3DFE). It is a more fair evaluation because, in real environments, the network does not know which are environmental conditions and the subjects.

To perform the cross-database experiment, seven groups of the CK+ database were used to train the network and one was used to be the validation set (to choose the best network weights based on the best epoch presentation order of the training set). The BU-3DFE was used to test the network. The training was done eight times, each one with a different validation set. We ran each configuration (training set plus validation set) ten times, each one with a different presenting order in the training samples. The result in the cross-database experiment is computed as an average of the 8 runs (8 groups, k-fold $*$ 1, best configuration of the 10 times with different presenting order) showing all the BU-3DFE images to test the network.

The cross-database training in the CK+ database and testing in the BU-3DFE database is shown in Table 8.

Table 8: BU-3DFE Cross-Database Experiment

Classifier	**Train**	**Test**	**6-expressions (%)**	**7-expressions (%)**
C_{nclass}	CK+	BU-3DFE	45.91	42.25
C_{bin}	CK+	BU-3DFE	81.97	83.50

The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 30 and in Table 31 respectively, in the appendices (B.1).

Training with JAFFE or CK+ and Test with the JAFFE As explained in Section 4.2.1, this experiment follows a slight different approach of the CK+ and BU-

3DFE experiments in regards to the number of groups. The JAFFE database contains images from only ten subjects, therefore, as done in (LIU et al., 2014) and in (SHAN; GONG; MCOWAN, 2009), the images are separated in ten groups, each one with just one subject. The test was carried out using a 10-fold cross validation, the training group contains eight subjects, the validation group one subject and the testing group one subject. The results of this experiment is computed as an average of the 10 runs (10 groups, k-fold $*$ 1 subjects per group, leave-one-out $*$ 1, best configuration of the 10 times with different presenting order). The result for six and seven expression for both classifiers is shown in Table 9.

Table 9: JAFFE Accuracy using six and seven (six basic plus neutral) expressions

Classifier	**6-expressions (%)**	**7-expressions (%)**
C_{nclass}	53.44	53.57
C_{bin}	84.48	86.74

As it can be seen, compared with the CK+ and BU-3DFE results, the accuracy decreased considerable. It also happens in other works, like (LIU et al., 2014) and (SHAN; GONG; MCOWAN, 2009), motivated mainly because the small database. The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 32 and in Table 33 respectively, in the appendices (B.1).

The cross-database experiment was also performed to the JAFFE database. In this experiment the network is trained and validated only on the CK+ database and the tests are carried out on the JAFFE database. This experiment follows the same approach described for the BU-3DFE. The result in the cross-database experiment is computed as an average of the 8 runs (8 groups, k-fold $*$ 1, best configuration of the 10 times with different presenting order) showing all the JAFFE images to test the network. The results for this experiment are shown in Table 10.

Table 10: JAFFE Cross-Database Experiment

Classifier	**Train**	**Test**	**6-expressions (%)**	**7-expressions (%)**
C_{nclass}	CK+	JAFFE	38.80	37.36
C_{bin}	CK+	JAFFE	79.60	82.10

The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 34 and in Table 35 respectively, in the appendices (B.1).

5.2 Neutral Subtraction Experiments

In this Section, a study is carried out showing the impact of the network parameters and meta-parameters and the parameters of the pre-processing steps in the accuracy of the method presented in Section 3.2. The results of this method for three database are also shown. Firstly, the results of the tuning experiments of the influence of each network parameter (architecture, learning rate and momentum) in the accuracy is presented. Secondly, a study presenting the impact in the accuracy of the samples presenting order is carried out. Finally, the results with different databases are shown and discussed in details.

5.2.1 Convolutional Neural Network Tunning

All parameters used to train the Convolutional Neural Network, including its architecture, were either determined using tunning experiments, or replicated from the best known methods for facial expression recognition that use Convolutional Neural Networks (LOPES; AGUIAR; SANTOS, 2015; FASEL, 2002b, 2002a) or replicated from general studies on Convolutional Neural Networks, as presented in (SIMARD; STEINKRAUS; PLATT, 2003). Here, it is shown the accuracy impact of parameter variation (the network architecture, the learning rate and the momentum) in the network. In addition, it is also presented the accuracy impact of others parameters of the method, like the amount of the synthetic samples generated, the Gaussian standard deviation and the order of presenting the samples to the network during training. For these tunning experiments, only a subset of the CK+ database was used. The training was done with 36 subjects and the test with 12 subjects without subject overlap between training and test.

Architecture The main difference between Convolutional Neural Networks is the network architecture. Depending of the input and the expected result, Convolutional Neural Networks have a specific combination of convolutions and sub-sampling layers to achieve a good accuracy rate. To find out the best architecture for our problem, three different architectures were evaluated.

The first, proposed in (FASEL, 2002a), is a network that comprises five layers and receives as input a 64×64 image. This network generates 60 maps with a kernel of 5×5 pixels in the first layer, and apply a subsampling of 2×2 pixels. Subsequently, a new convolution is performed with a 11×11 kernel, and is followed by another subsampling

layer with a 4×4 kernel. At the end, a layer with 100 neurons fully connects with the previous layer. The results of this architecture using the simplified feature space after neutral subtraction as input, for different values of the learning rate and momentum parameters, are shown in Table 11.

Table 11: Parameters impact in the network architecture proposed in (FASEL, 2002a).

L. R. / **Mom.**	**0.1**	**0.01**	**0.001**	**0.0001**	**0.00001**
0.99	30.00%	8.18%	69.09%	83.64%	87.27%
0.95	30.00%	8.18%	85.45%	**88.18%**	78.18%
0.9	8.18%	80.00%	84.55%	82.73%	39.09%
0.7	8.18%	81.82%	88.18%	86.36%	30.91%
0.5	8.18%	**88.18%**	86.36%	71.82%	30.00%

The second, proposed in (LOPES; AGUIAR; SANTOS, 2015) (that is the same shown in Section 3.1), is a network that comprises five layers and receives as input a 32×32 image. The first layer generates 32 maps with a 5×5 kernel, and subsamples these maps with a kernel of 2×2. Subsequently, a new convolution layer generates 64 maps with a 7×7 kernel size, and is followed by a new sub-sampling of 2×2 pixels. At the end, a layer with 256 neurons fully connects with the previous layer. The results of this architecture using the simplified feature space after neutral subtraction as input, for different values of the learning rate and momentum parameters, are shown in Table 12.

Table 12: Parameters impact in the network architecture proposed in (LOPES; AGUIAR; SANTOS, 2015).

L. R. / **Mom.**	**0.1**	**0.01**	**0.001**	**0.0001**	**0.00001**
0.99	8.18%	8.18%	**93.64%**	89.09%	89.09%
0.95	29.09%	92.73%	92.73%	89.09%	82.73%
0.9	29.09%	**93.64%**	90.00%	89.09%	76.36%
0.7	29.09%	91.82%	89.09%	85.45%	42.73%
0.5	29.09%	91.82%	90.91%	85.45%	40.91%

The third, shown in Section 3.2, is a network that comprises three layers and receives as input a 32×32 image. The architecture of this network comprises tree layers: the first

is a convolutional layer that produces 32 maps with a 5×5 kernel; the second is a subsampling layer that reduces the maps to a half; and, the third layer is a fully connected layer with 256 neurons. The results of this architecture using the simplified feature space after neutral subtraction as input, for different values of the learning rate and momentum parameters, are shown in Table 13.

Table 13: Parameters impact in the proposed network architecture

L. R. / **Mom.**	**0.1**	**0.01**	**0.001**	**0.0001**	**0.00001**
0.99	8.18%	8.18%	93.64%	91.82%	89.09%
0.95	8.18%	**95.45%**	92.73%	89.09%	85.45%
0.9	8.18%	**95.45%**	93.64%	90.00%	80.91%
0.7	8.18%	91.82%	90.00%	87.27%	53.64%
0.5	8.18%	92.73%	90.00%	89.09%	51.82%

The best result, an accuracy rate of 95.45%, was achieved using the architecture proposed in 3.2, and was followed by the architecture proposed in (LOPES; AGUIAR; SANTOS, 2015), with an accuracy rate of 93.64%. The worst results, an accuracy rate of 88.18%, were achieved by the architecture proposed in (FASEL, 2002a). Therefore, the architecture proposed in 3.2 was chosen to carry on the experiments with the neutral subtraction based method.

5.2.2 Presentation Order Tunning

The Convolutional Neural Neural network proposed uses a gradient descendant method in the learning process. The gradient descendant method relies in the presentation order of the samples to search for the local minimum (HAYKIN, 2008). Therefore an analysis of the impact in the accuracy of different presenting orders was performed. Table 14 shows the variation in accuracy for different random generated orders of the samples given to the network in the training.

Table 14: Impact of the presentation order in the accuracy

Presentation Order	**Accuracy**
1	95.45%
2	95.45%
3	94.55%
4	**93.64%**
5	**96.36%**
6	94.55%
7	94.55%
8	94.55%
9	94.55%
10	94.55%
Average: 94.82%	
Standard Deviation: 0.74%	

As it can be seen in Table 14, the presentation order has a significant impact in the accuracy. In this example, it increases (or decreases) the accuracy in 3.00% in some cases. Therefore, to avoid the variation of the accuracy based on the presentation order, the final result of our experiments are computed using the network weights of the best run out 10 runs having a validation set for accuracy measurement. Each run has a random presentation order of the training samples. The weights of the best run in the validation set are later used to evaluate the test set and compute the final accuracy.

5.2.3 Results

The results of the system accuracy considering the three different databases are shown below. For all databases the preprocessing step (rotation correction, cropping, down-sampling and intensity normalization) took only 0.01 second and the network recognition (classification step) took in average 0.01 second.

As discussed before, a simplified training/testing methodology was used to evaluate the impact of the pre-processing steps. In contrast to the tunning experiments that used only training and test sets, this section the accuracy is computed using the configurations shown in chapter 4.

Training with CK+ and Test with CK+ Table 15 shows the best result achieved (using both normalizations and the synthetic samples) using both classifiers. As can be seen the binary classifiers approach increases the accuracy. It happens because in this approach the hit can be achieved six times (one for each classifier), instead of using just one classifier, where each samples has just one chance to be properly classified. The binary classifier approach was employed to allow a fair comparison with some methods in the literature that just report this results, but we think that the six-class classifier (C_{6class}) is a more fair evaluation method.

Using the experiment configuration described in Section 4.2.1 for this database, the training of the network is performed about 960 times (8 groups $*$ 12 subjects per group $*$ 10 times with different presenting order). The time required to train the network each time was about only 1 minutes, resulting in a total training time (using the k-fold configuration) of 16 hours.

Table 15: CK+ Accuracy by class using the six basic expressions

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
$\boldsymbol{C_{6classE}}$	97.78%	97.18%	85.14%	100.0%	95.24%	98.80%
$\boldsymbol{C_{binE}}$	98.81%	99.46%	98.81%	99.14%	98.92%	99.24%
Average of $\boldsymbol{C_{6class}}$ (%): 97.19						
Average of $\boldsymbol{C_{bin}}$ (%): 99.06						

The training parameters that achieves the results shown in Table 2 is shown in Table 16. These same parameters are used in the experiments on other databases for this method, shown below.

Table 16: Training Parameters

Parameter	**Value**
Momentum	0.95
Learning Rate	0.01
Epochs	10000
Loss Funtion	Logistic Regression
Gaussian Standard Deviation	5
Synthetic Samples Amount	70

Using the result shown in Table 2 the confusion matrix shown in Table 17 was created for the six-class classifier.

Table 17: Confusion Matrix for the six-class classifier in the CK+ database

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
Angry	**132**	0	0	0	3	0
Disgust	5	**172**	0	0	0	0
Fear	0	0	**63**	5	3	3
Happy	0	0	0	**207**	0	0
Sad	3	0	0	0	**80**	1
Surprise	0	0	0	3	0	**246**

Instead of recognizing only six expressions, we can also recognize the neutral expression, resulting in a classifier that recognizes seven expressions. The result of the seven expression classifier to the CK+ database, using the same methodology of the six-class is show in Table 18

Table 18: CK+ Accuracy by class using seven expressions (six basic plus neutral expression)

	Neutral	**Angry**	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
$C_{7classE}$	85.71%	99.26%	98.87%	91.89%	98.55%	96.43%	98.39%
C_{binE}	97.34%	97.79%	99.82%	99.20%	99.47%	98.58%	99.29%
Average of C_{7class} (%): 95.75							
Average of C_{bin} (%): 98.79							

The confusion matrix for the seven expressions is shown in Table 19.

Table 19: Confusion Matrix for the seven-class classifier in the CK+ database

	Neutral	**Angry**	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
Neutral	**174**	19	0	0	0	9	1
Angry	0	**134**	0	0	0	1	0
Disgust	0	2	**175**	0	0	0	0
Fear	0	0	0	**68**	0	3	3
Happy	0	0	0	3	**204**	0	0
Sad	0	3	0	0	0	**81**	0
Surprise	1	0	0	0	3	0	**245**

As it can be seen in Table 15, a high accuracy was achieved for both classifiers. Comparing the confusion matrices Table 17 and 19 we can note that the accuracy decrease

was mainly motivated by the mistakes on the neutral expression classification. The neutral expression was confused sometimes with the angry and sad expressions. This happens because the facial changes caused by these expressions (angry and sad) are not well separated in the pixel space from the neutral expression. The standard deviation in the accuracy, when considering one result per group (discussed in Section 4), in the accuracy for the C_{6class} classifier between the eight groups, was $\sigma = 0.04$. Figure 21 shows some examples of the misclassification.

Figure 21: In (1) the expected expression was sad, but the method returned angry. In (2) the expected expression was angry, but the method returned sad. In (3) the expected expression was sad, but the method returned angry. In (4) the expected expression was fear, but the method returned sad.

In comparison with the results shown in Section 5.1, we have an increase in the results, from 98.92% to 99.06% for six expressions. Despite the increase be very small (less than 1%), in the neutral subtraction method, the time required to train the network and recognize and expression is 50% of the time required to perform the same tasks in the intensity normalization method.

Figure 22 shows a illustration of the learned kernels and the generated maps for the convolution layer. In the convolution layer, the input image is processed by the 32 learned kernels and generates 32 output maps. The kernels shown in Figure 22 were learned in the training using the CK+ database for the six basic expression.

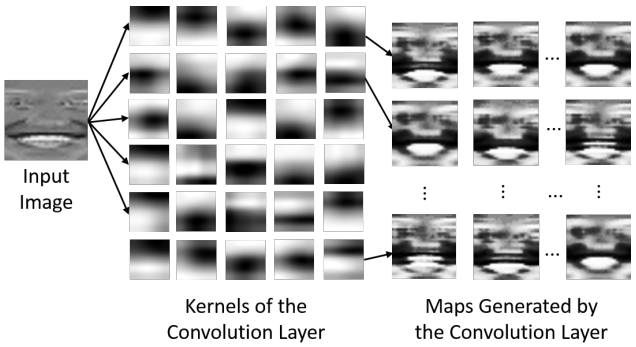

Figure 22: Illustration of the learned kernels and the generated maps for each convolution layer for the neutral subtraction Convolutional Neural Network. In the convolution layer, the input image is processed by the 32 learned kernels and generates 32 output maps. The sub-sampling layer is not represented in this image. Only a subset of the 32 kernels for the first layer is shown. The generated maps were equalized to allow for a better visualization.

Training with BU-3DFE or CK+ and Tests with the BU-3DFE This experiment follows the same approach as the CK+ database, the only difference is that in the BU-3DFE database the groups have about only eight subjects. The results of this experiment is computed as an average of the 64 runs (8 groups, k-fold $*$ 8 subjects per group, leave-one-out $*$ 1, best configuration of the 10 times with different presenting order). The result for six and seven expression for both classifiers is shown in Table 20.

Table 20: BU-3DFE Accuracy using six and seven (six basic plus neutral) expressions.

Classifier	**6-expressions (%)**	**7-expressions (%)**
C_{nclass}	86.82	80.48
C_{bin}	95.61	94.42

As it can be seen, the accuracy for the BU-3DFE decreased compared with the CK+ database. One possible reason is that this database has more subjects from different ethnicities and light conditions, and is smaller than the CK+. The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 36 and in Table 37 respectively, in the appendices (B.2).

The cross-database training in the CK+ database and testing in the BU-3DFE database is shown in Table 21.

Table 21: BU-3DFE Cross-Database tests

Classifier	Train	Test	6-expressions (%)	7-expressions (%)
C_{nclass}	CK+	BU-3DFE	51.22	43.03
C_{bin}	CK+	BU-3DFE	83.74	83.72

The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 38 and in Table 39 respectively, in the appendices (B.2).

Training with JAFFE or CK+ and Test with the JAFFE This experiment aims to measure the performance of the system with the JAFFE database. The methodology is the same as the explained in Section 5.1. The result for six and seven expression for both classifiers is shown in Table 22.

Table 22: JAFFE Accuracy using six and seven (six basic plus neutral) expressions

	6-expressions (%)	7-expressions (%)
C_{nclass}	53.79	43.20
C_{bin}	84.60	83.77

The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 40 and in Table 41 respectively, in the appendices (B.2). The cross-database experiment was also performed to the JAFFE database. The results for this experiment are shown in Table 23.

Table 23: JAFFE Cross-Database tests

Classifier	Train	Test	6-expressions (%)	7-expressions (%)
C_{nclass}	CK+	JAFFE	45.23	36.07
C_{bin}	CK+	JAFFE	81.74	81.73

The confusion matrices for this experiment, on six and seven expressions, can be seen in Table 42 and in Table 43 respectively, in the appendices (B.1).

5.3 Neutral Expression Detection Experiments

As discussed earlier, one constraint of the method proposed in Section 3.2 is the previous knowledge of an image of the subject in the neutral expression. One way to address this problem is to use a neutral classifier. This classifier aims to automatically

detect whether a image of a set of expression images is the neutral expression or not. Here, we show the results of one classifier that achieves a high accuracy rate on a neutral expression classification task.

To train and test the neutral classifier, only the Extended Cohn-Kanade database (CK+) (LUCEY et al., 2010) was used. The methodology was the same as the experiments with CK+. The same preprocessing operations performed in (LOPES; AGUIAR; SANTOS, 2015) were also applied for this test. Table 24 shows the accuracy result for the neutral classifier separated by group (G1 to G8).

Table 24: Neutral Classifier Accuracy

	Accuracy		**Accuracy**
G1	96.62%	**G5**	100.0%
G2	98.03%	**G6**	98.13%
G3	96.79%	**G7**	99.36%
G4	100.0%	**G8**	89.38%
Average of C_{binE}: 97.25%			

This experiment shows that in an environment where the image containing the neutral expression is unknown, this classification can be firstly applied to determine the neutral expression with a high accuracy rate. After this detection, the neutral expression image can be used as input to the facial expression classifier to determine the others expressions.

5.4 Comparisons

The results achieved by the proposed methods are compared with *Zhong et al* in (ZHONG et al., 2012), *Shan et al.* in (SHAN; GONG; MCOWAN, 2009) and *Liu et al* in (LIU et al., 2014). These works use the same experiments methodology and databases and achieve the best accuracy in the literature. Table 25 shows the comparison with the methods in the literature on the CK+ database.

Table 25: Comparison for the CK+ database

Method	Classifier	6-expressions	7-expressions
Ada-Gabor (ZHONG et al., 2012)	C_{nclass}	93.30	-
	C_{bin}	-	-
LBP + SVM (SHAN; GONG; MCOWAN, 2009)	C_{nclass}	95.10	91.40
	C_{bin}	-	-
BDBN (LIU et al., 2014)	C_{nclass}	-	-
	C_{bin}	96.70	-
Intensity Normalization	C_{nclass}	**96.76**	**95.75**
	C_{bin}	**98.92**	**98.80**
Neutral Subtraction	C_{nclass}	**97.19**	**95.75**
	C_{bin}	**99.06**	**98.79**

As can be seen in Table 25 the proposed methods achieves the best results in the CK+ database for all experiments configurations. Besides, the training and recognition time is also much smaller than the others. The whole experimentation time including all the k-fold configurations of the proposed method was 32 hours to the method proposed in Section 3.1 and only 16 hours to the method proposed in Section 3.2, and the recognition is real time (only 0.01 second for each image), almost 100 images per second. In comparison with *Liu et al* (LIU et al., 2014), their training took eight days and the recognition was about 0.21 per image. *Shan et al.* in (SHAN; GONG; MCOWAN, 2009) and *Zhon et al.* in (ZHONG et al., 2012) did not report the training and recognition time.

To verify that the method presented in Section 3.1 needs a deeper network architecture than the presented in Section 3.2, we carried out the experiments in the following way: using the configuration on training and testing in the CK+ database (explained in Section 4.2.1), the network architecture presented in Section 3.2 and the input image of the system proposed in Section 3.1. Table 26 shows the result on recognizing the six basic expressions in the CK+ database.

As it can be seen in Table 26, using a network with just one convolution and one sub-sampling layer (instead of two convolutions and two sub-samplings layers), given as input the feature set shown in Section 3.1, the network achieves a smaller accuracy compared with the result shown in Table 2. It reinforces the fact that the neutral subtraction method improves the feature extraction, allowing the accuracy improvement, even with a simpler Convolutional Network architecture.

Table 26: CK+ Accuracy by class using the intensity normalization method with the network architecture of the neutral subtraction method.

	Angry	**Disgust**	**Fear**	**Happy**	**Sad**	**Surprise**
$C_{6classE}$	80.94%	97.33%	65.70%	99.12%	61.36%	95.98%
C_{binE}	94.60%	98.26%	94.79%	98.80%	95.11%	96.76%
Average of C_{6class} (%): 89.16						
Average of C_{bin} (%): 96.39						

The state-of-the-art methods in the literature (SHAN; GONG; MCOWAN, 2009; LIU et al., 2014) also performed the cross-database experiment in the JAFFE database (training in the CK+ database and testing in the JAFFE). A comparison of this work with these methods, for the cross-database experiment, is show in Table 27. In our literature review, we did not find any work using the BU-3DFE for expression recognition.

Table 27: Comparison for the JAFFE cross-database experiment

Method	**Classifier**	**6-expressions**	**7-expressions**
LBP + SVM (SHAN; GONG; MCOWAN, 2009)	C_{nclass}	-	41.30
	C_{bin}	-	-
BDBN (LIU et al., 2014)	C_{nclass}	-	-
	C_{bin}	-	68.00
Intensity Normalization	C_{nclass}	**38.80**	**37.36**
	C_{bin}	**79.60**	**82.10**
Neutral Subtraction	C_{nclass}	**45.23**	**36.07**
	C_{bin}	**81.74**	**81.73**

Comparing the presented method with *Shan et al.* (SHAN; GONG; MCOWAN, 2009) our accuracy was about 4% smaller in using 7 expressions and the C_{nclass} classifier. They did not report the result of the six basic expressions. On the other hand, compared with *Liu et al.* (LIU et al., 2014), using the binary classifier approach the proposed method significantly increases the accuracy, from 68.0% to 82.0%. *Shan et al.* and *Liu et al.* did not report the results using only the six expressions.

5.5 Limitations

As discussed before, the presented methods needs the locations of each eye for the image pre-processing steps. The eye detection can be easily included to the system adopting the method proposed by *Saragih et al.* in (SARAGIH; LUCEY; COHN, 2010). In addition, as shown in tables 18 and 5, the accuracy of some expressions, like the sad one, was about 84%, while the accuracy of the whole method was about 96%. This suggests that

the variation between these classes are not enough to separate them. One approach to address this problem is to create a specialized classifier for those expressions, to be used as a second classifier. On the other hand, another approach to handle this lower accuracy could be training the system with a bigger database.

In additional, the method presented in Section 3.2 relies on the neutral expression identification for each subject. However, even without this a-priori information, we can still use a neutral expression detector to select the neutral image and give it to the system as input. This classifier was shown in Section 3.3, and it achieves a high accuracy rate. If we use it to first detect the neutral expression and only then classify the expressions based on this information, the overall system accuracy goes down to 96.00% in the CK+ database, which is the neutral classifier accuracy (97.25%) multiplied by the system accuracy (99.06%). To the JAFFE cross-database experiment the accuracy goes down to 79.30%, which is the neutral classifier accuracy (97.25%) multiplied by the system accuracy (81.74%) in this experiment. To the JAFFE cross-database experiment, the result still better than the result shown by Liu *et al* (LIU et al., 2014).

6 Conclusion and Future Works

In this work, two new approaches to extract facial expression features was proposed. These feature extraction methods combined with a simple architecture convolutional neural network achieves better accuracy than the state-of-the-art methods, the intensity normalization procedure and the neutral subtraction procedure achieves, respectively, 98.90% and 99.00% of accuracy in the CK+ database. Furthermore, the time required to train the network was significantly reduced compared with related work in the literature. The time required to train the systems was 16 hours to the intensity normalization method and 8 hours to the neutral subtraction method, works on the literature that report the training time took about 8 days to train the facial expression recognition system. Real time facial expression recognition, in standard PC computers, was also achieved, this same behavior was not reported before by the related works discussed in Section 2.3. Finally, the cross-database experiments show that the proposed approach also works in unknown environments, where the testing images acquisition conditions and subjects vary from the training images.

As explained in Section 2.1, the use of Convolutional Neural Networks aims to decrease the need for hand-coded features. Its input can be raw images, instead of an already selected set of features. It happens because this neural network model is able to learn the set of features that best models the desired classification. To perform such learning, Convolutional Neural Networks need a big amount of data, that we do not have. This is a constraint of deep architectures, motivated by the large amount of parameters that needs adjustment during training. To address this problem (our limited data), the pre-processing operations were applied to the images, in order to decrease the variations between images and in order to select a subset of the features to be learned, reducing the need of a big amount of data. If we had a better set of images, with more variation and a bigger amount of samples (millions), these pre-processing operations could not be necessary to achieve the reported accuracy and even the cross-database validation could be improved.

Preliminary experiments were performed with deeper architectures, trained with a big amount of data. In these experiments, a deep Convolutional Neural Network composed by 38 layers and trained with about 982,800 images from 2,662 subjects, proposed by *Parkhin et al.* in (PARKHI; VEDALDI; ZISSERMAN, 2015) to recognize faces, was briefly studied. The already trained model was used as a pre-trained feature extractor plugged as input of a simple two-layered neural network trained with our own data. In this experiment, no preprocessing operation was applied. Despite the experiment simplicity, the results achieved were promising and even increase the accuracy achieved in the cross-database experiments (reported in Section 5), with the cost of decreasing the accuracy in the same database experiments.

As future work, the application of this feature extraction method will be investigated in others problems, where the features vary over time. In additional, we want to investigate others learning methods in order to increase the method robustness in unknown environments (e.g. with varying light conditions and others). Also, more tests will be performed using the face descriptor proposed by *Parkhin et al.* in (PARKHI; VEDALDI; ZISSERMAN, 2015), using fine adjustment techniques, which aims to train an already trained deep neural network in order to focus on more specific features (in our case, expressions).

References

AHMED, A. et al. Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks. In: . [S.l.: s.n.], 2008. p. 69–82.

AHONEN, T.; HADID, A.; PIETIINEN, M. Face recognition with local binary patterns. In: *Computer Vision - ECCV 2004*. [S.l.]: Springer Berlin Heidelberg, 2004, (Lecture Notes in Computer Science, v. 3021). p. 469–481.

BARTLETT, M. et al. Recognizing facial expression: machine learning and application to spontaneous behavior. In: . [S.l.: s.n.], 2005. v. 2, p. 568–573.

BENGIO, Y.; GOODFELLOW, I. J.; COURVILLE, A. Deep learning. Book in preparation for MIT Press. 2015.

BENGIO, Y.; LECUN, Y. Scaling learning algorithms towards AI. In: *Large-Scale Kernel Machines*. [S.l.]: MIT Press, 2007.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

BUTTOU, L. *Stochastic Gradient Descent Tricks*. [S.l.]: Springer, 2012.

BYEON, Y.-H.; KWAK, K.-C. Facial expression recognition using 3d convolutional neural network. In: . [S.l.: s.n.], 2014. v. 5, n. 12.

CALEANU, C.-D. Face expression recognition: A brief overview of the last decade. In: . [S.l.: s.n.], 2013. p. 157–161.

CARROLL, J. M.; RUSSELL, J. A. Do facial expression signal specific emotions? *Journal of Personality and Social Psychology*, v. 70, n. 70, p. 205–218, 1996.

CHEN, C.-R.; WONG, W.-S.; CHIU, C.-T. A 0.64 mm real-time cascade face detection design based on reduced two-field extraction. *IEEE Transactions on Very Large Scale Integration Systems*, v. 19, p. 1937–1948, 2011.

CHOI, J.-I. et al. Face and eye location algorithms for visual user interface. In: *Proceedings of First Signal Processing Society Workshop on Multimedia Signal Processing*. [S.l.]: Institute of Electrical & Electronics Engineers (IEEE), 1997.

CIRESAN, D. C. et al. Flexible, high performance convolutional neural networks for image classification. In: *Proceedings of the 21th International Joint Conference on Artificial Intelligence*. [S.l.: s.n.], 2011. (IJCAI'11), p. 1237–1242. ISBN 978-1-57735-514-4.

DARWIN, C. *The expression of the emotions in man and animals*. [S.l.]: Smithsonian Institution, 1916.

DEMIRKUS, M. et al. Multi-layer temporal graphical model for head pose estimation in real-world videos. In: . [S.l.: s.n.], 2014. p. 3392–3396.

EKMAN, P.; FRIESEN, W. V. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Palo Alto: Consulting Psychologists Press, 1978.

FASEL, B. Head-pose invariant facial expression recognition using convolutional neural networks. In: . [S.l.: s.n.], 2002. p. 529–534.

FASEL, B. Robust face analysis using convolutional neural networks. In: . [S.l.: s.n.], 2002. v. 2, p. 40–43.

FREUND, Y.; HAUSSLER, D. *Unsupervised Learning of Distributions on Binary Vectors Using Two Layer Networks.* Santa Cruz, CA, USA, 1994.

FUKUSHIMA, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, p. 193–202, 1980.

GARCIA, C.; DELAKIS, M. Convolutional face finder: a neural architecture for fast and robust face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 26, n. 11, p. 1408–1423, 2004.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: *International Conference on Artificial Intelligence and Statistics.* [S.l.: s.n.], 2010.

GLOROT, X.; BORDES, A.; BENGIO, Y. Deep sparse rectifier neural networks. In: GORDON, G. J.; DUNSON, D. B. (Ed.). *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS-11).* [S.l.: s.n.], 2011. v. 15, p. 315–323.

HAYKIN, S. O. *Neural Networks and Learning Machines (3rd Edition).* 3. ed. [S.l.]: Prentice Hall, 2008.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 18, p. 1527–1554, 2006.

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. *Science*, v. 313, p. 504–507, 2006.

HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, v. 195, n. 1, p. 215–243, 1968.

JAIN, S.; HU, C.; AGGARWAL, J. K. Facial expression recognition with temporal modeling of shapes. In: . [S.l.: s.n.]. p. 1642–1649.

JIA, Y. et al. Caffe: Convolutional architecture for fast feature embedding. *Conference on Computer Vision and Pattern Recognition*, 2014.

KIM, W. W. et al. Automatic head pose estimation from a single camera using projective geometry. In: . [S.l.: s.n.], 2011. p. 1–5.

KLUWER, A. P. Glossary of terms. *Machine Learning Journal*, v. 30, n. 2, p. 271–274, 1998. ISSN 0885-6125.

LAROCHELLE, H. et al. An empirical evaluation of deep architectures on problems with many factors of variation. In: *24th International Conference on Machine Learning*. New York, NY, USA: ACM, 2007. p. 473–480.

LAWRENCE, S. et al. Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks*, v. 8, p. 98–113, 1997.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Publishing Group, v. 521, p. 436–444, 2015.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, v. 1, p. 541–551, 1989.

LECUN, Y. et al. Gradient-based learning applied to document recognition. v. 86, p. 2278–2324, 1998.

LECUN, Y.; HUANG, F. J.; BOTTOU, L. Learning methods for generic object recognition with invariance to pose and lighting. In: . [S.l.: s.n.], 2004. v. 2, p. 97–104.

LEVNER, I. *Data Driven Object Segmentation*. Tese (Doutorado) — Department of Computer Science, University of Alberta, 2008.

LI, G. et al. An efficient face normalization algorithm based on eyes detection. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.]: Institute of Electrical & Electronics Engineers (IEEE), 2006.

LI, S. Z.; JAIN, A. K. *Handbook of Face Recognition*. [S.l.]: Springer Science & Business Media, 2011. ISBN 9780857299321.

LIEN, J.-J. J. et al. Detection, tracking, and classification of action units in facial expression. *Journal of Robotics and Autonomous Systems*, 1999.

LIU, P. et al. Facial expression recognition via a boosted deep belief network. In: . [S.l.: s.n.], 2014. p. 1805–1812.

LIU, P.; REALE, M.; YIN, L. 3d head pose estimation based on scene flow and generic head model. In: . [S.l.: s.n.], 2012. p. 794–799.

LIU, W.; SONG, C.; WANG, Y. Facial expression recognition based on discriminative dictionary learning. In: . [S.l.: s.n.], 2012. p. 1839–1842.

LOPES, A. T.; AGUIAR, E. de; SANTOS, T. O. A facial expression recognition system using convolutional networks. In: *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*. [S.l.: s.n.], 2015.

LUCEY, P. et al. The extended cohn-kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression. In: . [S.l.: s.n.], 2010. p. 94–101.

LV, G. Recognition of multi-fontstyle characters based on convolutional neural network. In: . [S.l.: s.n.], 2011. v. 2, p. 223–225.

LV, Y.; FENG, Z.; XU, C. Facial expression recognition via deep learning. In: *2014 International Conference on Smart Computing*. [S.l.]: Institute of Electrical & Electronics Engineers (IEEE), 2014.

LYONS, M. J.; BUDYNEK, J.; AKAMATSU, S. Automatic classification of single facial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 21, p. 1357–1362, 1999.

MATSUGU, M. et al. Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks: The Official Journal of the International Neural Network Society*, v. 16, p. 555–559, 2003.

MCCULLOCH, W. S.; PITTS, W. Neurocomputing: Foundations of research. In: ANDERSON, J. A.; ROSENFELD, E. (Ed.). Cambridge, MA, USA: MIT Press, 1988. cap. A Logical Calculus of the Ideas Immanent in Nervous Activity, p. 15–27.

MOBAHI, H.; COLLOBERT, R.; WESTON, J. Deep learning from temporal coherence in video. In: *26th Annual International Conference on Machine Learning*. New York, NY, USA: ACM, 2009. p. 737–744.

NIU, X.-X.; SUEN, C. Y. A novel hybrid cnn-svm classifier for recognizing handwritten digits. *Pattern Recognition*, v. 45, p. 1318–1325, 2012.

PARKHI, O. M.; VEDALDI, A.; ZISSERMAN, A. Deep face recognition. In: *British Machine Vision Conference*. [S.l.: s.n.], 2015.

POULTNEY, C.; CHOPRA, S.; LECUN, Y. Efficient learning of sparse representations with an energy-based model. In: *Advances in Neural Information Processing Systems (NIPS 2006*. [S.l.]: MIT Press, 2006.

ROSENBLATT, F. *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*. [S.l.]: Spartan Books, 1962. (Report (Cornell Aeronautical Laboratory)).

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Neurocomputing: Foundations of research. In: ANDERSON, J. A.; ROSENFELD, E. (Ed.). Cambridge, MA, USA: MIT Press, 1988. cap. Learning Representations by Back-propagating Errors, p. 696–699.

RUSSELL, J. A. Is there universal recognition of emotion from facial expression? a review of the crosscultural studies. *Psychological Bulletin*, v. 70, n. 115, p. 102–141, 1991.

SALAKHUTDINOV, R.; HINTON, G. E. Using deep belief nets to learn covariance kernels for gaussian processes. In: *Neural Information Processing Systems*. [S.l.]: Curran Associates, Inc., 2007. p. 1249–1256.

SARAGIH, J. M.; LUCEY, S.; COHN, J. F. Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, v. 91, p. 200–215, 2010.

SERRE, T. et al. A quantitative theory of immediate visual recognition. *Progress in Brain Research*, p. 33–56, 2007.

SHAN, C.; GONG, S.; MCOWAN, P. W. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, v. 27, n. 6, p. 803–816, 2009.

SIMARD, P. Y.; STEINKRAUS, D.; PLATT, J. C. Best practices for convolutional neural networks applied to visual document analysis. In: . [S.l.: s.n.], 2003. p. 958–963.

TAIGMAN, Y. et al. Deepface: Closing the gap to human-level performance in face verification. In: *Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2014.

UTGOFF, P. E.; STRACUZZI, D. J. Many-layered learning. *Neural Computation*, v. 14, p. 2497–2529, 2002.

WANDELL, B. A. *Foundations of Vision*. Sunderland, Mass: Sinauer Associates Inc, 1995.

WESTON, J.; RATLE, F.; COLLOBERT, R. Deep learning via semi-supervised embedding. In: *25th International Conference on Machine Learning*. New York, NY, USA: ACM, 2008. p. 1168–1175.

WU, Y.; LIU, H.; ZHA, H. Modeling facial expression space for recognition. In: . [S.l.: s.n.], 2005. p. 1968–1973.

YANG, P.; LIU, Q.; METAXAS, D. N. Boosting coded dynamic features for facial action units and facial expression recognition. In: . [S.l.: s.n.], 2007. p. 1–6.

YIN, L. et al. A 3d facial expression database for facial behavior research. In: *7th International Conference on Automatic Face and Gesture Recognition*. [S.l.]: Institute of Electrical & Electronics Engineers (IEEE), 2006.

ZHANG, Z. et al. Comparison between geometry-based and gabor-wavelets-based facial expression recognition using multi-layer perceptron. In: . [S.l.: s.n.], 1998. p. 454–459.

ZHANG, Z. et al. Regularized transfer boosting for face detection across spectrum. *IEEE Signal Processing Letters*, v. 19, p. 131–134, 2012.

ZHAO-YI, P.; ZHI-QIANG, W.; YU, Z. Application of mean shift algorithm in real-time facial expression recognition. In: . [S.l.: s.n.], 2009. p. 1–4.

ZHONG, L. et al. Learning active facial patches for expression analysis. In: . [S.l.: s.n.], 2012. p. 2562–2569.

APPENDIX A – Publications

- **Lopes, A. T.**; Aguiar, E.; Oliveira-Santos, T.; **A Facial Expression Recognition System Using Convolutional Networks** in: 2015 28th SIBGRAPI Conference on Graphics, Patterns and Images, Institute of Electrical & Electronics Engineers (IEEE), 2015.

- **Lopes, A. T.**; Aguiar, E.; De Souza, A. F.;Oliveira-Santos, T.; **Facial Expression Recognition Using Deep Learning - Convolutional Neural Networks**. Pattern Recognition. ISSN 0031-3203, 2016. **(Invited and Submitted)**

APPENDIX B – Confusion Matrices

B.1 Intensity Normalization Based Method

Table 28: Confusion Matrix for six expressions on the BU-3DFE database in the same database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	109	26	7	2	10	3
Disgust	15	12	14	11	0	7
Fear	12	21	61	10	9	19
Happy	0	6	6	159	3	0
Sad	17	3	14	5	124	5
Surprise	7	6	6	5	11	134

Table 29: Confusion Matrix for seven expressions on the BU-3DFE database in the same database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	191	11	4	5	1	14	6
Angry	22	93	24	3	0	15	0
Disgust	0	9	117	20	6	0	7
Fear	21	3	23	47	6	10	22
Happy	5	0	6	2	160	0	1
Sad	28	10	0	8	3	113	6
Surprise	13	0	6	7	6	5	132

Table 30: Confusion Matrix for six expressions on the BU-3DFE database in the cross-database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	55	30	0	0	23	49
Disgust	12	57	9	6	5	71
Fear	4	3	13	12	24	76
Happy	6	8	5	111	22	22
Sad	6	0	3	3	51	105
Surprise	5	0	0	3	7	154

Table 31: Confusion Matrix for seven expressions on the BU-3DFE database in the cross-database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	116	1	0	4	0	24	87
Angry	77	28	25	0	0	5	22
Disgust	23	8	62	3	9	10	44
Fear	29	0	3	7	11	25	57
Happy	36	3	8	2	100	14	11
Sad	21	7	0	3	3	54	80
Surprise	1	3	2	0	3	5	155

Table 32: Confusion Matrix for six expressions on the JAFFE database in the same database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	168	32	26	10	32	2
Disgust	79	129	21	1	30	1
Fear	38	17	98	6	72	57
Happy	9	6	6	224	19	24
Sad	53	26	42	14	123	21
Surprise	7	1	61	17	21	163

Table 33: Confusion Matrix for seven expressions on the JAFFE database in the same database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	156	33	3	4	22	8	35
Angry	6	175	17	28	2	39	3
Disgust	5	84	118	12	0	41	1
Fear	24	30	0	103	0	80	43
Happy	46	12	1	3	210	6	10
Sad	8	55	27	42	12	116	19
Surprise	66	2	0	36	11	6	149

Table 34: Confusion Matrix for six expressions on the JAFFE database in the cross-database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	2	2	2	2	11	11
Disgust	1	3	0	0	7	18
Fear	1	0	2	0	9	20
Happy	1	0	2	18	11	0
Sad	0	2	0	1	18	10
Surprise	1	0	0	1	3	25

Table 35: Confusion Matrix for seven expressions on the JAFFE database in the cross-database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	4	0	0	0	0	20	5
Angry	5	4	2	0	3	10	6
Disgust	3	2	4	0	1	7	12
Fear	0	0	0	3	0	9	20
Happy	2	2	0	2	17	8	1
Sad	1	2	2	0	1	18	7
Surprise	0	1	0	0	1	2	26

B.2 Neutral Subtraction Based Method

Table 36: Confusion Matrix for six expressions on the BU-3DFE database in the same database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	115	12	0	0	28	0
Disgust	7	141	8	3	0	0
Fear	2	10	76	9	21	14
Happy	0	0	3	171	0	0
Sad	0	0	0	0	166	2
Surprise	0	0	0	6	1	161

Table 37: Confusion Matrix for seven expressions on the BU-3DFE database in the same database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	140	1	1	0	34	0	0
Angry	11	239	21	0	43	0	0
Disgust	8	9	79	0	22	14	0
Fear	9	0	19	146	0	0	0
Happy	11	0	0	0	155	2	0
Sad	10	0	5	1	0	152	0
Surprise	0	0	0	0	0	0	0

Table 38: Confusion Matrix for six expressions on the BU-3DFE database in the cross-database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	138	8	12	0	4	0
Disgust	61	56	12	9	14	7
Fear	65	3	45	5	5	9
Happy	37	0	8	129	0	0
Sad	140	0	0	0	28	0
Surprise	39	0	0	3	7	119

Table 39: Confusion Matrix for seven expressions on the BU-3DFE database in the cross-database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	168	2	0	0	0	6	0
Angry	76	60	12	3	0	4	0
Disgust	44	13	73	15	0	11	3
Fear	37	23	3	41	3	17	8
Happy	25	26	3	9	108	3	0
Sad	110	20	0	0	0	38	0
Surprise	53	1	0	3	0	41	70

Table 40: Confusion Matrix for six expressions on the JAFFE database in the same database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	88	20	13	24	35	0
Disgust	33	59	29	18	32	0
Fear	8	8	123	12	41	6
Happy	21	12	9	98	34	15
Sad	33	28	27	30	7	0
Surprise	11	27	10	13	0	200

Table 41: Confusion Matrix for seven expressions on the JAFFE database in the same database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	57	44	9	13	12	45	0
Angry	35	62	23	17	14	29	0
Disgust	33	33	35	21	9	40	0
Fear	39	8	4	109	4	24	10
Happy	26	16	7	12	84	28	16
Sad	54	37	19	16	12	51	0
Surprise	0	10	33	10	13	2	193

Table 42: Confusion Matrix for six expressions on the JAFFE database in the cross-database experiment

	Angry	Disgust	Fear	Happy	Sad	Surprise
Angry	16	0	2	1	1	0
Disgust	15	0	0	1	1	2
Fear	15	0	1	0	3	3
Happy	15	0	1	2	1	2
Sad	18	0	1	0	2	0
Surprise	5	1	0	1	2	20

Table 43: Confusion Matrix for seven expressions on the JAFFE database in the cross-database experiment

	Neutral	Angry	Disgust	Fear	Happy	Sad	Surprise
Neutral	18	2	0	0	0	0	0
Angry	15	2	0	0	1	2	0
Disgust	13	4	0	0	1	1	0
Fear	19	2	0	0	0	0	1
Happy	13	3	1	3	0	0	1
Sad	17	2	1	0	0	1	0
Surprise	3	2	1	0	1	4	18