ERASMUS UNIVERSITY ROTTERDAM

Erasmus School of Economics

Master Thesis Econometrics and Management Science

A Facial Expression Recognition System using Histogram of Oriented Gradients, Gabor Filters, PCA and LDA, and Probabilistic Neural Networks

Author: Anna-Liisa Distefano (422980ad)

Supervisor: Dr. Kathrin Gruber

DATE OF FINAL VERSION

11 October 2021

Abstract

This paper presents a Facial Expression Recognition (FER) system based on a specific image processing framework using Histogram of Oriented Gradients (HOG) for face detection, combined with Gabor filter extracted features, a two phase dimensionality reduction method using Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), and classification using Probabilistic Neural Networks (PNN). The thesis of this paper is that human emotions are intrinsic in uncertainty both in their expression and recognition and that hence, a suitable FER system is one that is able to capture uncertainty. The Bayes decision strategy allowed by the PNN is deemed to be suitable for this. The proposed system is compared to a end-to-end Convolutional Neural Network (CNN), and is trained on the Extended Cohn-Kanade (CK+) database. It is found that the proposed FER system achieves an average accuracy score of 94.25% (person independent) over the 7 emotions expressed in the database.

Contents

1 Introduction

Facial expressions are largely responsible for the way that humans identify and communicate with each other. The incredible peculiarity of each individuals face and the way that they express emotion are the fruit of an infinitesimal chance of a specific combination of genes, cultural factors and uncontrollable external events. Facial expressions provide the foundation for effective and reliable human communication, to the extent that the misinterpretation or inability to express these are classified as psychological or developmental disorders. Facial expressions are directly linked to emotion expressions, and can be said to be one of the most important channels of human emotion communication. Mehrabian and Russell (1974) found that visual interpretation of a communicated message constitutes 55% of the understanding of the message, while 38% percent is linguistic and only is 7% is verbal (the actual words being said).

According to Ekman and Friesen (1971), there are six universal emotions that can be found in all cultures, namely happiness, surprise, sadness, fear, anger, and disgust. While the distinction of different emotions has been studied qualitatively since the dawn of civilisation, attempts at quantitative emotion analysis only started to arise in the 20th century with models such as the Valence-Arousal space model Russell (1980), which suggests that emotions can be described by a two-dimensional circular space with arousal and valence on the axes. Later, in the beginning of the 21st century, the Facial Action Coding System (FACS) Ekman (2002), also called Action Units (AUs) model, was developed and became a benchmark for quantification of facial movements and hence expressions. Indeed, the FACS model attempts at describing the human face and its expression by a set of well-defined coordinates. Automatic emotion recognition systems on the other hand have only been possible starting from the late 1990s, with the rise of advanced computing systems, Artificial Intelligence (AI) methods and increased availability in computer processing power. Hence, the ambition of teaching machines how to perform the very complex task of emotion recognition only started to be come a reality in the 21st century.

Today, the demand for these systems keeps on increasing across numerous fields: in secu-

rity for surveillance systems, in marketing for consumer preference analysis, in psychology for analysis of human response, in robotics and Human-Machine Interaction (HMI) systems for systems such as self-driving cars. For most humans, emotion recognition is an intuitive, unconscious task, yet a very complex one that involves stimulation of numerous senses simultaneously. Given the full sensory context, humans are able to rapidly and effectively classify whether a human face is displaying anger, sadness, happiness or disgust, and to react accordingly. Machines however, still struggle to achieve reliable results, especially in uncontrolled environments. This is largely due to the fact that cultural, societal and physiological differences in humans make it so that there exists great variability in the way that emotions are expressed and recognized. In the case of digital images, the diverse composition in terms of light, angles and size injects additional variability in facial emotion representation. This means that for automatic systems to be useful, they need to be able to handle this variability in order to classify emotion representations correctly.

There exists a large variety of Facial Expression Recognition (FER) systems, that range from analysis of static images to analysis of real-time videos. FER systems require the ability to recognise facial features and to keep track of (micro) changes in their position across the face. This has led FER systems to become one of the most active applications of computer vision techniques such as pattern recognition, edge detection and image analysis. These technologies have greatly evolved and today it can be said that two main classes of systems exist. A traditional FER system consists of three main steps: face detection, feature extraction and expression classification, each of which has developed into an independent field of research. In these, some commonly used classifiers include k-Nearest Neighbours (KNN), Support Vector Machine (SVM), Adaptive Boosting (Adaboost) or Bayesian Classifiers (Huang et al., 2019). In recent years however, with the emergence of advanced computing methods like deep-learning, deep neural networks such as the Convolutional Neural Network (CNN) have become the amongst the most popular FER classifiers approaches as they allow the learning to occur within the pipeline directly from the input images. This enables the FER system to not have to heavily rely on the manual feature engineering of the images like the traditional systems do.

While being widely popular, the CNNs however have drawbacks, the major ones being that they are computationally expensive, don't provide a probabilistic output, don't perform well with small data sets and most importantly aren't able capture uncertainty. Since emotion expression carries a lot of uncertainty, this deficiency is particularly important in the case of FER systems. Conversely, another type of neural networks, namely the Probabilistic Neural Networks (PNNs) (Specht, 1990), are based on statistical principles hence are able to capture uncertainty. Additionally, they work better on small data sets and are computationally significantly more efficient. Using PNN as a classifier in the FER system implies that image processing and feature engineering prior to classification are needed. For these steps, a variety of specific methods and algorithms is used.

The purpose of this research is to build a FER System based on a series of specific image processing steps, Gabor Filters, a dual phase feature reduction step and a neural network based on the Bayes decision strategy, namely the PNN. The first phase of the system consists of image processing, including the facial detection step for which the Histogram of Oriented Gradients (HOG) algorithm is used; in the second phase, the Gabor filter bank is used to extract the feature vectors describing the images to be classified; subsequently, Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are deployed to reduce the dimensionality of the feature space, and lastly, the PNN classification algorithm is applied. The system is trained using the Extended Cohn Kanade (CK+) (Lucey et al., 2010) dataset, which includes seven expression classes, namely sadness, surprise, happiness, fear, anger, contempt and disgust, and is evaluated on classification accuracy, computational effort, and compared to an end-to-end Convolutional Neural Network (CNN) classifier.

The structure of the paper is as follows: first, the most important literature in terms of the relevant algorithms and methods is discussed, secondly, the theory of the methods are described in detail, after which the algorithm structure is laid out. In the last sections of the paper, the main results are displayed and discussed followed by a conclusion. The code used for this research and additional results can be found in the Appendix).

2 Literature

With the exponential development of Human-Computer interaction systems and Artificial Intelligence methods, vast availability of data and improved computing power, the study of FER systems has developed into an independent field over the course of the last decades. The ambition of teaching machines how to understand facial emotions has in fact become reality, and studies to improve the speed, efficiency and accuracy of these systems continue to be conducted internationally. Today, FER systems can be distinguished in two main classes, namely in the traditional and the deep learning ones (Huang et al., 2019). Typically, a traditional FER system is comprised of four steps, namely image pre-processing, facial recognition, feature extraction and expression classification. Each of these steps can be said to be an independent field of research, as advanced methods continue to be developed across them. On the other hand, deep-learning-based FER systems make use of an end-to-end approach, meaning that they aim to eliminate the feature reduction step and to reduce the pre-processing steps required by the traditional systems. For either of the two approaches, Neural Networks (NNs) have become amongst the most common classifiers due to their accuracy and flexibility. However, different types of NNs tend to be better suited for either of the two approaches. Furthermore, FER systems can be applied to either static images or dynamic image sequences (videos), or either on digital images or in real-time systems, each of which requires special techniques. In the remaining part of this section of the paper, first the history and literature of different types of NNs will be discussed, followed by previous research on image processing techniques, facial detection algorithms and feature extraction methods in the context of FER systems. Lastly, the topic of this research is discussed in relation to previous work.

Artificial Neural Networks (ANN), or simply Neural Networks (NN), is a type of computing system which construction is inspired by the way the human brain analyzes and processes information. Many AI systems today are NN based, as they are capable of solving highly complex problems by overcoming the limitations of human reasoning and of statistical methods. NNs started to be developed in the mid 40s, but their development could not accelerate until significant advancement were made in terms of computing power, which came later

in the 80s. In 1974, the backpropagation algorithm by Werbos (1994) contributed to the reviving of NNs research by enabling a practical training of multi-layer networks. ANN as they are known and used today, are computing systems made of so-called artificial neurons and layers, and are used for a wide variety of tasks for computer vision, natural language processing and data mining. Since their inception, NNs have evolved into a broad family of algorithms, generally differing in the number of units, weights, layers and topology. A first attempt at using NNs for automatic facial expression recognition was introduced by Stonham (1986), who attempts to classify smiles from frowns in a single perceptron network. Later work by Lisetti and Rumelhart (1998), advanced the potential of neural networks for facial expression recognition by proposing a three layer network evaluated with backpropagation. Across the different test configurations, Lisetti and Rumelhart (1998) achieve low error rates, which can however be attributed to the small size of their dataset and to bieng subject to pre-processing the images by hand. Prior to Lisetti and Rumelhart (1998), other connectionist algorithms were proposed for emotion recognition from images (Cottrell and Metcalfe, 1990; Rosenblum et al., 1996), which helped lay the ground for the spread of NNs for automatic emotion recognition.

In the late 1990s, a new type of NN was being developed that would later become a widely used and highly accurate classification algorithm for FER systems today. Convolutional Neural Networks (CNN) are a class of NNs that perform particularly well in image processing or any other 2D type of data. CNNs as they are most commonly used today are evaluated using backpropagation, and were introduced between 1990 and 1998 by LeCun et al. (1998). Indeed, LeCun et al. (1998) wanted to improve Neural Networks in the way that they deal with variability in images, and to create a single network that would perform the entire recognition operation of an image in one (that is with minor pre-processing needed). CNNs have been widely applied to FER, and have been shown to obtain high accuracy scores across numerous facial expression data sets (Huang et al., 2019). Fasel (2002) introduces the use of CNNs for emotion detection from images, which Matsugu et al. (2003) then improved by making it subject independent and achieving a recognition rate of 97.6% for smile detection. Later work by Breuer and Kimmel (2017) report 98.62% accuracy using CNN on the Ex-

tended Cohn-Kanade (CK+) dataset (Lucey et al., 2010) using CNNs, outperforming other previously studies methods on the same dataset (Huang et al., 2019). While achieving high accuracy scores and being widely used for both research and commercial purposes, CNNs are evaluated using backpropagation, which is a computationally expensive method which cannot guarantee to find the global minimum in all cases. This can be especially problematic in regions with limited amount of data, as the CNN can lead to overconfident decisions (Shridhar et al., 2019). Furthermore, CNNs are evaluated based on discrete estimates of the network weights, meaning that they don't take any uncertainty into account.

Another type of NN is given by the Probabilistic Neural Network (PNN), which was first introduced by Specht in 1990 (Specht, 1990), around the same time as CNNs were starting to be developed. PNNs were first introduced as an improvement to neural networks being learned by heuristic approaches such as backpropagation. Specht (1990) sought in fact to create a neural network based on *established statistical principles* and Bayes decision strategy; with a feed-forward neural network structure, that can be used for classification and estimation of a-posteriori probabilities. The main advantages of using PNNs over conventional back-propagation neural networks for classification are that: they are faster than back-propagation, they yield a probabilistic output and can be used on smaller sets of training data. The first application of PNN for facial expression classification is reported by Fazli et al. (2009), who propose to use Gabor filter bank and Linear Discriminant Analysis (LDA) in conjunction with PNN on the images after the pre-processing stage. They use the Cohn-Kanade (CK) database (Kanade et al., 2000), and report to achieve a performance of about 89%. A year later, Neggaz et al. (2010) use an improved Active Appearance Model (AAM) for feature extraction followed by a PNN. They use the JAFFE database (Lyons et al., 1998), and report an average recognition accuracy of 96%. On the other hand, Ouyang et al. (2020) propose a system similar to Fazli et al. (2009) however for facial recognition, combining and improved kernel LDA with PNN, and show that this method saves computing time, improves computing efficiency and precision. They use the ORL, YALE and AR data sets, and report an average recognition accuracy of 97.22%, 83.8% and 99.12%, across the three data sets respectively. Furthermore, in the field of FER, PNNs have also been applied to

mouth tracking, specifically to lip shape extraction and lip motion (Seyedarabi et al., 2006). The main advantages of PNNs over NNs evaluated by backpropagation are that they are computationally more efficient and hence significantly faster in training, they reliably find the global minimum as they are based on precise statistical and probabilistic scores and the exact posteriori likelihood can be derived. However, PNNs occupy more memory to store the model than CNNs do (Mohebali et al., 2020).

Before expression classification, a traditional FER system expects an image pre-processing step, a facial detection step and a feature extraction step. The accuracy of a FER system can largely depend on pre-processing as images are often contaminated by signals from various sources such as harsh lighting, noisy backgrounds, camera quality or other interfering factors. Lopes et al. (2015) demonstrate this by proposing an advanced multiple-step pre-processing method composed of spatial normalisation, synthetic sample generation, image cropping, downsampling and intensity normalisation of the dataset, and show that these indeed improve the classification score of the CNN classifier from 61.70% to up to 93.74%. In fact, the aim of image pre-processing techniques is to eliminate irrelevant information of input images and reduce the size and complexity of the images. Typically, this step comprises of scale and grayscale normalisation, noise reduction and Histogram Equalisation (HE) for image enhancement (Huang et al., 2019). Noise reduction is often needed as edge detection and pattern recognition algorithms are generally sensitive to noise, therefore methods such as Average Filter (AF), Gaussian Filter (GF), Median Filter (MF) are used to reduce noise in input images and improve performance. The Gaussian Filter can be said to have been introduced by Marr and Hildreth (1980), and is often regarded as optimal and is widely used in FER systems. On the other hand, Histogram Equalisation (HE) is the most widely used to regulate contrast of an image through its histogram (which describes the distribution of the tones of the image). Specifically, this is useful in FER systems as it allows to better differentiate the facial features and hence the facial expressions (Zhao et al., 2010). Classic histogram equalisation works with the global contrast of the image which may lead to regions that are too dark or too light, which in turn can be especially bad for images with large intensity variations such as facial images. Contrast Limited Adaptive Histogram Equalization

(CLAHE) (Pizer et al., 1987) overcomes the limitations of classical HE by applying local equalisation within a specific contrast limit, making it highly attractive to FER systems (Bendjillali et al., 2019a; Munir et al., 2018; Cornejo et al., 2015).

The accuracy of a traditional FER system greatly depends also on the accuracy of the facial detection step. Facial detection algorithms, also known as automated facial recognition systems, are a form of pattern recognition algorithms that first started to be conceptualised 1960s. The first publication came with Sakai et al. (1972), who had developed a simple face matching system based on heuristics and anatomical measures. This paper laid the foundation for what would be published as the first book about facial recognition technologies a few years later (Kanade et al., 1977). However, it wasn't until the 1990s that the first truly automatic face detection systems started to be developed, as neural networks and feature extraction methods were improving in parallel. Turk and Pentland (1991) used Principal Component Analysis (PCA) for face detection for the first time, and coined the so-called *Eigenface method* for face detection. Later in the 1990s, texture analysis methods based on techniques such as Gabor Filters started to overtake the PCA and the Linear Discriminant Analysis (LDA) methods, which became known as Bochum systems (Wiskott et al., 1997). A few years later, Viola and Jones (2001) made real-time facial detection possible for the first time, introducing the algorithm that would become known as the Viola-Jones algorithm and widely used across the world. The Viola-Jones algorithm is a four step recognition process, based on Haar-like features and Adaboost training, which has been widely used in FER systems, as seen in Owusu et al. (2014), Agrawal and Khatri (2015), Bendjillali et al. (2019b). Viola-Jones comes however with important limitations, such as being ineffective in recognising tilted or turned faces and being sensitive to lighting conditions. Subsequently, new systems were developed, and it can be said that the Viola-Jones algorithm has been replaced by the Histogram of Oriented Gradients (HOG) face detection method, which was first introduced by Dalal and Triggs (2005) for pedestrian detection on static images. The HOG is an object detection algorithm based on feature descriptors by local evaluations of edge directions, and is especially suited for human detection. The HOG algorithm is invariant to geometric and photometric transformations, and is faster and more accurate

than the Viola-Jones algorithm. After the publication Dalal and Triggs (2005), the HOG method was almost immediately adopted for FER systems (Kanaujia et al., 2006), and has been widely been used for that purpose since (Hu et al., 2008; Chen et al., 2016; Zeng et al., 2018; Jumani et al., 2019).

In terms of feature extraction models, two main classes can be said to exist: the geometric feature-based ones that are based on the position the mouth, eyes, nose and eyebrows, and the appearance-based which extract the feature vectors based on filters on the entire facial perimeter. The aim of the feature extraction step is to produce a concise description of the image in terms of a vector, called feature vector. Popular feature extraction methods for FER systems include the the Active Appearance Model (AAM) (Edwards et al., 1998) as seen in the work of Lucey et al. (2010), Ashraf et al. (2009), Neggaz et al. (2010); and the Gabor filters (Daugman, 1985,9) as seen in Bartlett et al. (2003), Deng et al. (2005), Fazli et al. (2009), Saabni (2015) and Verma and Khunteta (2017). AAM models are statistical model-based algorithms developed from the Active Shape Model (ASM) (Cootes et al., 1995), which are based on the modeling of the shape and the texture of objects. The AAM has been widely used in FER systems as its a flexible model that provides a unified approach to facial analysis. The AAM however requires extensive labelling of the images prior to training, as numerous facial landmarks need to be manually placed per image, and has been shown to be inefficient in terms of time and memory, and sensitive to changing lighting conditions (Gao et al., 2010). On the other hand, Gabor filters have more attractive properties. Gabor filters are a widely used method in computer vision, more specifically used for texture analysis, edge detection and feature extraction. The Gabor filters consist of a set of linear filters that capture the texture in the image in different directions and frequencies at given localised positions, which make them orientation sensitive and able to capture different shapes, sizes and smoothness levels. A set of multiple Gabor filters is called a Gabor filter bank, which method became popular due to possessing optimal localization properties in both spatial and frequency domain, and the ability to closely emulate the human visual system.

Because a traditional FER system required a feature extraction step, this often implies the need for reduction of the parameter space. Two widely used techniques for FER are Principal

Component Analysis (PCA) and Linear Discriminant Analysis (LDA), which both aim at finding the linear combination that best explains the data (Martinez and Kak, 2001). PCA and LDA are dimensionality reduction tools used in multivariate data analysis to approximate a set of data in terms of smaller sets that capture the essential information of the original dataset. Deploying either of these methods separately or together in a FER system improves model performance, increases computational efficiency and reduces memory usage (Pumlumchiak and Vittayakorn, 2017). While initially PCA (Pearson, 1901) was used as a method for facial detection per se, in recent systems it is indeed most commonly used to reduce the dimension of the feature space prior to classification (Deng et al., 2005; Agrawal and Khatri, 2015; Mohebali et al., 2020). PCA has also been used in the context of FER systems after it had been shown that the first three Principal Components retain unwanted variations, and that hence discarding them can improve recognition accuracy (Belhumeur et al., 1997; Martinez and Kak, 2001; Pumlumchiak and Vittayakorn, 2017). Linear Discriminant Analysis (LDA) on the other hand has often been used as an alternative to PCA, or in conjunction with PCA as seen in Deng et al. (2005); Hu et al. (2008); Fazli et al. (2009); Pumlumchiak and Vittayakorn (2017). The benefit of combining PCA and LDA is to obtain a further reduction of the parameter space while minimising the loss of information, and to improve the accuracy and efficiency of the system. Deng et al. (2005) and Pumlumchiak and Vittayakorn (2017) show in fact that using LDA on PCA improves model performance with respect to using only LDA or PCA alone.

The purpose of this research is to develop a novel FER system based on solid statistical principles using a combination of the above discussed methodologies and algorithms. The system is inspired by Pumlumchiak and Vittayakorn (2017), which combines HOG, Gabor Filters (8 orientations), PCA and LDA, and outperforms previous similar systems. Pumlumchiak and Vittayakorn (2017) however use the the weighted K-Nearest-Neighbours (KNN) approach based on Euclidean distance as a classifier. Similarly, inspiration is taken by Slavković et al. (2013), which combine Gabor Filters (at 5 scales and 8 orientations), PCA and NNs, however opting for a NN that is based on the backpropagation learning algorithm. Furthermore, most similar to the FER system proposed by this paper is given by Fazli et al. (2009), which

combines Gabor Filters with PCA, LDA and use PNN as classifier and apply it to the Lucey et al. (2010) dataset, however opting for an alternative system (Shih and Chuang, 2004) for face detection. Based on the literature and previous research, this paper proposes a system combining a specific image pre-processing procedure with HOG, Gabor Filters, PCA, LDA and PNN, and compares it to and end-to-end CNN. To the best of my knowledge this has not been explored yet, so focus this research is be to develop a novel FER system based on the latter algorithms and methodologies.

3 Methods

In this section every method used in the algorithm is explained in detail. First, the image processing techniques are described, including the face detection one. Then, the feature extraction and dimensionality reduction methods are described, followed by the classication algorithm and the model metrics. The specifics of the algorithms can be found in the next section of the paper.

3.1 Image pre-processing

The aim of the image pre-processing and the facial feature extraction phase is to find an optimal representation of the face images prior to being subjected to the classification algorithm. First, all images are converted to grayscale. Second, a Gaussian Filter (GF) is applied to reduce image noise. Then, faces are detected and cropped using the Histogram of Oriented Gradients (HOG) algorithm. Lastly, histogram equalisation using Contrast Limiting Adaptive Histogram Equalisation (CLAHE) is applied to adjust the contrast level of the images and the scale of the images is normalised.

3.1.1 Gaussian Filter

Facial images are often corrupted by noise, which can affect the performance of the FER system. Gaussian Filters (GF), also known as Gaussian blur, are a type of low-pass filters used in image processing to reduce noise and therefore mask the effect of redundant details, while preserving important information such as edges. A smoothing operation using GF on

a 2D image is described by a convolution process based on the bi-variate Gaussian function described in equation 1 below:

$$\phi(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

where x and y are the distances on both axes of a given pixel from the origin that is assumed to be (0,0), and where the choice of σ dictates the extent of the blur, such that the larger the σ, the higher the blur. GFs are linear filters, which application results in a linear combination of pixel values in the neighbourhood of a given pixel. The convolution operation on image $I(x, y)$ of size mxn, can be described by the following:

$$I(x, y) * \phi(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} I(x, y)\phi(x - s, y - t) \tag{2}$$

where $a = (m - 1)/2$ and $b = (n - 1)/2$. By convolving the Gaussian function over the pixels of the image, a new value - which is a weighted average of the neighbouring pixels - is assigned to each pixel in such a way that the pixel in question gets assigned the largest weight - which corresponds to the maximum of the Gaussian function - while the weights of the neighbouring pixels decrease according to the shape of the Gaussian function. In practice, pixels that are too far away from the center of the filter equate to 0, so a discrete approximation is necessary. Generally, pixels that are further away than 3σ away from the center are already too close to 0.

Additionally, the function in 1 is separable, meaning that $\phi(x, y) = \phi(x)\phi(y)$, which implies that the 2D GF filter can also be implemented as a sequence of two one dimensional convolution filters, which is computationally more efficient.

3.1.2 Histogram of Oriented Gradients

Next, the face perimeter is detected and the images are cropped to remove redundant background information. For this task, the Histogram of Oriented Gradients (HOG) object detector is used. The HOG is based on the notion that high differences in contrast indicate the presence of an edge in the image and hence, most likely an object. The HOG algorithm

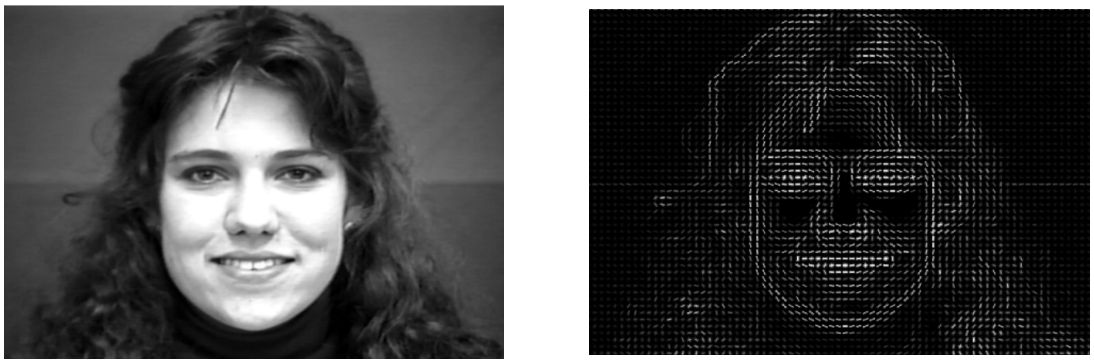
consists of four main steps, namely: gradient computation, orientation binning, block normalisation and object detection. First, the picture is divided into small cells by applying a grid. For each pixel within each cell, the gradient magnitude α and the gradient direction β are computed (gradient computation). The gradient magnitude measures the difference in pixel value along the x and y axes with respect to a given pixel, which is measured by:

$$\alpha = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{3}$$

and the gradient direction is given by:

$$\beta = tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right) \tag{4}$$

where f represents a cell of pixels in the grid. Given these measures, a histogram is compiled in the range of 0-180 degrees with bin width equal to 20 (orientation binning). For each direction, the frequency of magnitudes corresponding to that given direction over the pixels in the cell gives the value of the bin of that direction. Hence, the histogram represents the distribution of gradient magnitudes over the gradient directions. The values of each bin then make up a feature vector of size 9 for a cell in the grid.



(a) Original picture (b) HOG descriptors

Figure 1: Visualisation of HOG descriptors

This operation is then applied on all the cells of the grid, and subsequently, on larger blocks across the image, from which new feature vectors are then extracted. Then, the set of feature vectors, also known as HOG descriptors, are used to train a Linear Support Vector Machine

(SVM) classifier to predict whether a region in an image corresponds to a face.

(a) Original picture

(b) Cropped image

Figure 2: Image of subject 66 before and after face detection and cropping

3.1.3 Histogram equalisation - CLAHE

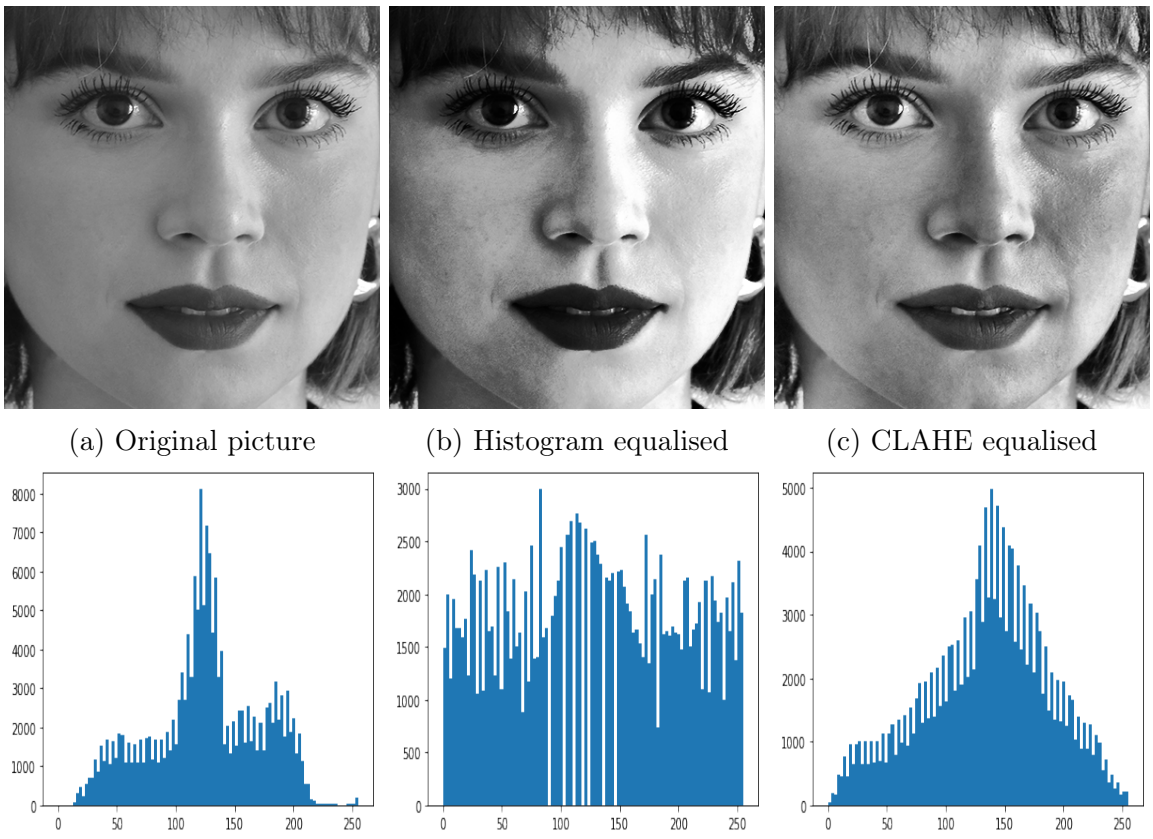
Facial images can have an unsatisfactory level of contrast which can also differ across the images in the data set. If an image contrast is too low or high, the features become more difficult to recognise and extract and therefore the effectiveness of the FER system will be affected. To adjust and even out the constrast, CLAHE histogram equalisation is used. Every digital image can be described by its histogram, that is simply by plotting the frequencies of its RGB or gray-scale values. In the case of gray-scale images, this means that the values of the histogram fall in the range [0,255], where 0 corresponds to black and 255 to white. Let R denote pixel intensity such that $R \in [0, L]$, where $L = 255$. Then, histogram equalisation can be described by the following transformation of R:

$$H(R) = \lfloor (L-1) \sum_{r=0}^{R} p_r \rfloor \tag{5}$$

where p_r is the probability of a pixel having value R, and the floor function rounds down to the nearest integer.

This transformation alone however does not provide optimal results as it can create areas that are either too dark or too bright, as seen in Figure 3. For this reason, the Contrast Limited Adaptive Histogram Equalization (CLAHE) is chosen instead. The principle of CLAHE is to divide the image into grids of a specified size, and apply the transformation

in 5 to each of them while smoothing the edges of the grid using bilinear interpolation. The grid separation alone can increase noise in areas of the image, therefore CLAHE also limits the contrast by constraining the histogram range prior to equalisation. Figure 3 shows the result of applying CLAHE versus simple histogram equalisation on the image alongside their historgrams.



(a) Original picture (b) Histogram equalised (c) CLAHE equalised

Figure 3: Effect of Histogram equalisation and CLAHE

3.2 Feature Extraction - Gabor Filters

Two-dimensional (2D) Gabor filters are linear band pass filters used to analyse textures and extract features from images. Gabor filters are contextual filters, meaning that pixels are analysed based on their contextual information, that is the pixel values in their neighbourhood. The feature extraction method using Gabor filters is described by a number of relations, namely by a complex function describing the filters (equation 6), by the convolution operation (equation 8) and by the choice of the parameter range which dictates the

number of filters in the Gabor filter bank.

The Gabor filter is represented by a complex sinuisoidal impulse response function multiplied by a Gaussian kernel function, also called envelope, as shown by the set of equations below:

$$\begin{aligned} g_{\sigma,\theta,\lambda,\gamma}(x,y) &= exp\left[-\frac{\tilde{x}^2+\gamma^2y^2}{2\sigma^2}\right]exp\left[i\Big(2\pi\frac{\tilde{x}}{\lambda}+\psi\Big)\right] \\ \tilde{x} &= xcos\theta + ysin\theta \\ \tilde{y} &= ycos\theta - xsin\theta \end{aligned} \tag{6}$$

where (x, y) are the pixel coordinates, θ determines the orientation (or angle) of the filter, λ the wavelength (or intensity), σ the standard deviation of the Gaussian kernel, γ the aspect ratio (that is the ellipticity of the filter determining whether its more circular or elliptical), and ψ the phase offset, that is the center or symmetry. Each of these parameters can be tweaked to create a set of Gabor filters, called a Gabor filter bank.

Typically, the value of ψ is set to equal 0, while γ ranges from 0 to 1 so can often be set in the mid range. Typically, a Gabor filter bank is built on the range of its orientation θ and wavelength λ. Inspired by Fazli et al. (2009); Pumlumchiak and Vittayakorn (2017), the orientations are chosen according to:

$$\theta_m = \frac{m\pi}{M}, \quad m = 1, 2, ..., M-1 \tag{7}$$

where M is the number of orientations distributed over the interval [0,π], which is chosen as $M = 8$. To the wavelength λ, a range of values can be assigned to inject further variability in the filter bank, however, a single value is chosen and equal to $\frac{3\pi}{4}$. Technically, each of the parameters can be chosen across a range of values. In fact, how many values are chosen for each parameter determines the number of filters in the Gabor filter bank. Given this configuration, the filter bank is comprised of a total of 8 filters (figure 4. For a detailed summary of the parameters refer to Table 2 in the next section of the paper.

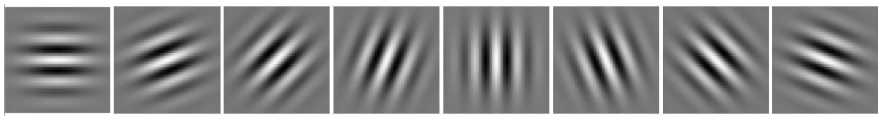


Figure 4: Gabor Filter bank example

To extract the feature vectors, a given image is convoluted with every filter of the filter bank at every pixel coordinate (x, y). Let $C = (c_x, c_y)$ be the center of the kernel of size KxK. The processing of the filter on a given grayscale image $I(x, y)$ is applied according to the following equation:

$$G_{\sigma,\theta,\lambda,\gamma}(x, y) = \sum_{i=0}^{K} g(x_i, y_i) \cdot I\Big(x + x_i - c_x, y + y_i - c_y\Big) \tag{8}$$

The size of the kernel is typically chosen with respect to the size of the input image, so for a 64x64 pixel image a kernel size of 9x9 can be chosen. The output of the Gabor filter bank method is an array of feature vectors describing the respective images.

Figure 5: Original processed image (left) and after convolution of the Gabor kernels

Figure 5 shows an image from the CK+ data set (Lucey et al., 2010) convoluted with filters with $\sigma = 0.75$.

3.3 Dimensionality Reduction

Feature extraction using Gabor Filters increases the dimensionality of the problem significantly, hence reduction of the parameter space is necessary. In this section, Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) for dimensionality reduction are described.

3.3.1 Principal Component Analysis

To reduce the size of the network input vector without significant loss of information, conventional PCA (Pearson, 1901) is used. PCA is defined as an orthogonal linear transformation

that transforms the M dimensional data to a new coordinate system made of P principal components. It uses linear combination of the data to find a set of axes - known as principal components - that best describe the variance in the data. The PCA transformation can be described by the equation:

$$Y = XA \tag{9}$$

where Y is the score matrix, X is the data matrix and A contains the weights a_{mp}, also known as loadings, for the m-th variable and p-th principal component. These are the elements of the eigen vectors of the variance-covariance matrix of X, that describe the variation in the original data. In the new coordinate system, the position of the data points are called scores. The score of the k-th observation on the p-th principal component is described by the linear combination of the single data points and the weights:

$$Y_{kp} = a_{1p}x_{k1} + a_{2p}x_{k2} + a_{Mp}x_{pM} \tag{10}$$

The weights a_{mp} in A are calculated such that the variance of Y_1 - the first principal component - is maximised, and the variance of Y_2 is maximised under the restriction that Y_1 and Y_2 are not correlated. In the same way, the variance of the other components is also maximised such that the correlation between the other components is 0. The resulting system is structured so that the first principal component describes most of the data, followed by the second that describes the second most amount and so on. Typically, one chooses the principal components up to the point where an additional principal component does not provide a large increase in the total variance explained, and such that at least 90% of the variance is explained.

3.3.2 Linear Discriminant Analysis

After PCA, LDA is applied to reduce the dimension space further. LDA is a statistical method used to find the features that linearly separate two or more classes. LDA is commonly used either as a classifier alone, or as a feature reduction reduction step prior to classification. LDA can be used in a supervised setting, that is when the classes of the data are known beforehand. The LDA method aims at maximising the variance/distance between

classes while minimising the variance/distance within each class, and can be described as an optimisation problem of the set of linear combinations below:

$$\sum_{s=1}^{S}\sum_{i=1}^{N_s}(x_i^s - \mu_s)(x_i^s - \mu_s)' \tag{11}$$

$$\sum_{s=1}^{S}(\mu_s - \mu)(\mu_s - \mu)' \tag{12}$$

where x_i^s is image i belonging to class s, μ_s is the mean image per class s, μ is the total mean image, S is the number of classes, N_s is the number of images per class. In other words, LDA finds the optimal allocation of data points per class and reduces the dimension of a vector drastically while mainting accurate information.

3.4 Classification

3.4.1 Probabilistic Neural Network

Lastly, the feature vectors are classified using a Probabilistic Neural Network (PNN). PNNs allow to classify unlabelled patterns of data based on a set of patterns for which the label is known, and consist of four layers, namely an input layer, pattern layer, summation layer and output layer (figure 6). For classification, the probability density functions (PDFs) for each class need to be estimated. To learn the a-posteriori PDF from the input data, the PNN uses a non-parametric density estimation method, namely the Parzen window method[1] (Parzen, 1962), and the Bayes decision rule as discriminant (Mood, 1950) for pattern classification.

The Bayes decision strategy can be interpreted so that it allows to minimise the so-called expected risk. Let the discriminant function of class j be defines as:

$$d_j(x) = p_j l_j f_j(x) \tag{13}$$

where p_j represents the prior probability of class j, l_j represents the loss function and $f_j(x)$ is the PDF of class j evaluated on the training set. The priori probability p_j is equivalent to

[1] more specifically, this type of PNN is called a Parzen PNN (PPNN)

the ratio of the number of patterns belonging to class j in the training set and the number of patters in the training set. The loss l_j allows to incorporate the probability of an incorrect decision in the decision rule of the PNN, but it can be set equal to 1 under the assumption that no incorrect decisions are made. The value of the loss cannot be retrieved by the data and it constitutes a subjective measure set by the researcher based on knowledge about false positives. Therefore it is often set to equal 1. Then the Bayes decision strategy for a given pattern x on which the PNNs are based can be described by the following rule:

$$j(x) = k \quad \text{if} \quad d_k(x) > d_i(x) \quad \forall \quad i \in 1, ..., m, k \neq i \tag{14}$$

where $j(x)$ is the class given pattern x and m is the total number of classes.

Given that there is no information available about the PDFs of the different classes, a non-parametric density estimation method needs to be used. Parzen (1962) shows that the the PDF $f(x)$ of a class j consisting of a set of random variables $X_1, ..., X_n$ can be approximated by the following equation:

$$\hat{f}_j(x) = \frac{1}{nh(n)} \sum_{j=1}^{n} K\Big(\frac{x - X_j}{h(n)}\Big) \tag{15}$$

where $h(n)$ satisfies:

$$\lim_{n \to \infty} h(n) = 0 \tag{16}$$

and $K(\cdot)$ represents a Borel function that satisfied a set of conditions (more details appendix or citation).

The structure of the PNN can be visualised in Figure 6. It can be seen that the input layer consists a set of p neurons - where p is the dimension of the input vector - that distributes the elements of the input vector over the neurons of the pattern layer. The pattern layer is made of as many neurons as the size of the training data, that is nm nodes, where n is the number of patterns. The pattern layer consists of two steps. The first step consists of measuring the distance between the input pattern and the pattern received by a given pattern node. The distance measure originally proposed by Specht (1990) and most commonly used in Parzen

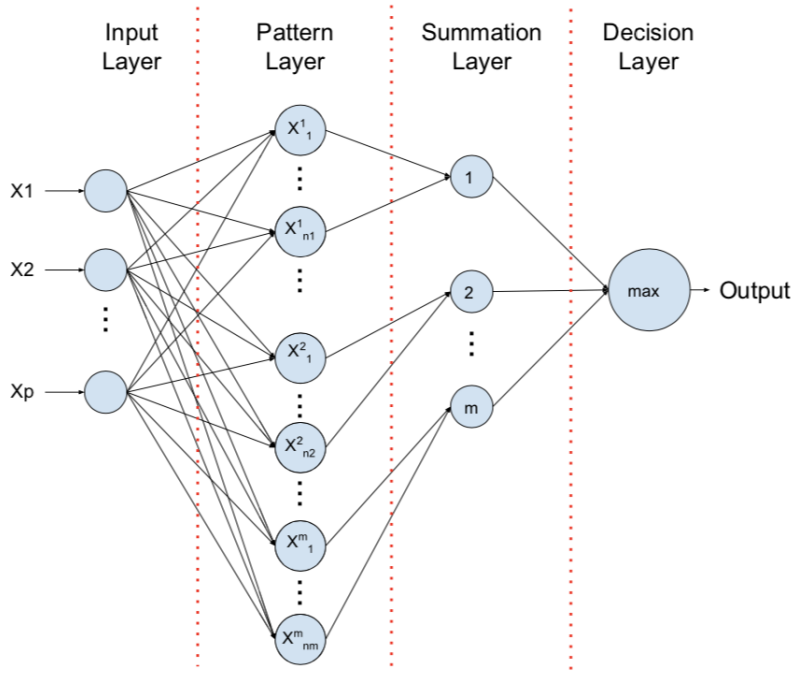


Figure 6: Basic PNN structure (Mohebali et al., 2020)

PNNs, is the dot product[2], defined as follows:

$$D(x, X_{ij}) = (x - X_{ij})'(x - X_{ij}) \tag{17}$$

Given this distance, the pattern layer applies a Gaussian kernel function[3] to this distance, which is given by:

$$K(x, u) = K(D(x, u)) = \frac{1}{\sqrt{2\pi}} exp(-D(x, u)/2\sigma^2) \tag{18}$$

which can also be seen as an exponential neuron activation function. The output of the pattern layer then reaches the summation layer which consists of m nodes. The summation layer sums the outputs of the pattern layer that correspond to each respective class, which, assuming the kernel and distance measures in equations 17 and 18, results in the respective

[2]if instead of a dot-product distance, the Eucledian distance measure was chosen, the kernel function would need to be chosen differently as well.

[3]the kernel function needs to satisfy the following conditions: 1. has its maximum at x=u, 2. $\lim_{D(x,u)\to\infty} K(x, u) = 0$, 3. $K(u, v)$ is continuous on $-\infty < x < \infty$, 4. If $K(x_1, u) = K(x_2, u)$, meaning $D(x_1, u) = D(x_2, u)$, then x_1 and x_2 vectors have the same degree of similarity with respect to u.

estimated PDF of class j, that is:

$$\hat{f}_j(x) = \frac{1}{(2\pi)^{p/2}\sigma^p}\frac{1}{n}\sum_{i=1}^{n} exp\Big[-(x - X_{ij})'(x - X_{ij})/2\sigma^2\Big] \tag{19}$$

Lastly, in case there is no justification to impose differing a-priori probabilities and losses across the classes (equation 13), the output layer simply maximises the output of the summation layer and returns the corresponding class.

The smoothing parameter of the PNN is the σ parameter, which is chosen to be constant over the filters, as Specht (1990) suggests that because σ controls the scale of the activation function, its value should not differ across the patterns. Specht (1990) also argues that small changes in its value don't affect the accuracy score significantly, and that it is relatively straightforward to find an appropriate value.

3.4.2 Model performance

The proposed FER system will be evaluated based on classification accuracy, training time and memory. The classification accuracy is defined as the ratio of the number of correct predictions and the number of total predictions. The confusion table reports the number of false positives, false negatives, true positives, and true negatives.

4 Data

To train and test the proposed FER system, the Extended Cohn-Kanade (CK+) dataset (Lucey et al., 2010) is used, which is an extended version of the Cohn-Kanade dataset (Kanade et al., 2000). The CK+ is an FACS-coded, posed facial expression database comprising of 593 image sequences and still images of seven facial expressions of 123 subjects. In addition to *neutral*, the expressions included in the dataset are: *sadness, surprise, happiness, fear, anger, contempt* and *disgust*. The resolution of the images is either 640x490 or 640x480 pixels with 8-bit gray-scale values for the most part, with a minority being 24-bit color values. Figure 7 shows some examples of the images that can be found in the dataset. Participants were 18 to 50 years of age, 69% female, 81%, Euro-American, 13% Afro-American, and 6%

other groups.

Figure 7: Example images of the CK+ dataset (Lucey et al., 2010)

The database includes a total of 10727 images from 123 subjects, from which only 327 images and their respective labels are selected. These correspond to expression images at peak frame for the 118 subjects for which an emotion was recognised by the FACS coding system at the peak frame. Out of this subset, for each class a varying number of images are found (Table 1). For the experiments, the images are randomised based on the 118 subjects, so to avoid the image of the same subject to be present in both training and test set.

Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
45	18	59	25	69	28	83

Table 1: Images per class

The CK+ dataset is one of the largest posed publicly available facial expression datasets, and is very widely used in the field of FER research. By choosing this dataset, the results of this research can be directly compared with the FER Baseline System as proposed by Lucey et al. (2010), with Fazli et al. (2009), Pumlumchiak and Vittayakorn (2017) and Breuer and Kimmel (2017).

5 Algorithm

In this section, the algorithm is discussed in detail. The pseudocode and the parameter grid can be observed in algorithm 18 and Table 2 respectively.

The proposed system starts with converting all 327 images to gray scale and applying a Gaussian filter with σ=1 to reduce noise. Using these images, the face rectangle is detected using

the HOG method, and the images are cropped according to the coordinates of these rectangles. Then, the CLAHE is applied using grids of size 8x8 pixels. The resulting processed images are then scaled to size 64x64. The image features are then extracted using a Gabor Filter bank of 8 filters, which results in 327 feature vectors of size 64x64x(8+1)=36864, including the original image. Before being concatenated, each convoluted image is normalised by substracting the image mean and diving this difference by the image standard deviation. This corresponds to a 36864 dimensional problem, hence PCA is deployed such that 90% of the variance is explained. This yields to 327 feature vectors of length 148. To reduce the dimensions further, LDA is applied to the 148 principal components, which results in 327 vectors of length 6. Lastly, prior to classification, the image feature vectors and their corresponding labels are split into training and test samples randomising based on subjects and by choosing a training set of size 1/4 of the data. This yields to 240 feature vectors of size 6 for training and 87 vectors of size 6 for testing. Using these, a PNN with $\sigma = 1$ and prior class probabilities $\left\{\frac{32}{240}, \frac{15}{240}, \frac{43}{240}, \frac{17}{240}, \frac{51}{240}, \frac{23}{240}, \frac{59}{240}\right\}$, is trained and evaluated on accuracy score and training time. The pseudo-code of the system can be observed in algorithm 18, and the parameter description is given by table 2. For comparison, two types of CNNs are run to account for the different configurations of the system (for more details refer to section 7, the Appendix of this paper).

Processing		Gabor Filters		PNN	
σ_{GF}	1	σ_G	0.75	σ_{PNN}	1
clip limit	3	θ	$0, \frac{\pi}{8}, ..., \frac{7\pi}{8}$		
grid size	8x8	λ	$\frac{3\pi}{4}$		
image size	64x64	γ	0.5		
		ϕ	0		
		kernel size	9x9		

Table 2: Model parameters

For face detection, `dlib` library's (King, 2009) HOG object detector is used [4], and for Gabor filter generation and convolution, `openCV` library (Bradski, 2000) is used. For training and evaluation of the CNN model, the `TensorFlow` library (Abadi et al., 2015) is used.

[4] `dlib`'s documentation does not give precise details of the detector's parameters and configurations, hence it is assumed that the object detection tool provided library is correct and corresponds exactly to the description of the algorithm described in this paper

The experiments are run on Python's Jupyter notebook's free service Google colab, which provides free access to Google servers and GPUs (graphics processing units).

Algorithm 1: FER using Gabor Filters and PNN

Input: DATA = {IMAGES, LABELS}

Step 1: Initialisation

import libraries
define functions
define parameter values
images, labels, subjects = **CreateDataset**(DATA)

Step 2: Image pre-processing

for *image in images* **do**
    ToGrayScale()
    GaussianFilter(σ_{GF})
    FaceDetectorHOG()
    CLAHE(clip limit, grid size)
    Resize(image size)
end

Step 3: Feature extraction

$X_{features}$ = **GaborFilters**(images, kernel size, $\theta, \sigma_{GF}, \lambda, \gamma, \phi$)

Step 4: Dimensionality reduction

X = **PCA**($X_{features}$, 90%)
X = **LDA**(X, labels)

Step 5: Classification

X_{train}, X_{test}, y_{train}, y_{test} = **TrainTestSplit**(X, labels, 70%, randomise on subjects)
y_{pred} = **PNN**(X_{train}, X_{test}, y_{train}, σ_{PNN})

Step 6: Model evaluation

Prediction accuracy = **Accuracy**(y_{pred}, y_{test})
Confusion table = **ConfusionMatrix**(y_{pred}, y_{test})

Return Prediction accuracy, confusion table, execution time

6 Results

In this section, the findings of the experiment are summarised. First, it is noted that in the image processing phase, `dlib`'s HOG object detector recognises a unique face in 100% of the pictures, and that after feature extraction, the first five principal components explain almost 25% of the variation in the features (as seen in the Appendix of this paper).

In terms of experiments, a PNN with and withour prior (equation 13) is trained on resized original images of size 64x48, on the pre-processed images of size 64x64, and on the fully engineered feature vectors with and without the PCA step. Applying both types of PNN classifiers to each of the intermediate configurations allows to compare to which extent each

step of the system affects the accuracy of the classifier, as well as to gauge the effect of applying a prior probability of each class to the decision discriminant. On the other hand, the CNN is also trained on the raw images of size 640x480, as its computation allows it within the limit of the computing power of the device at hand. The prediction accuracies as well as the computing times obtained by running these experiments are shown in Table 3. The runtime presented in the table does not include the runtime of the image pre-processing and feature extraction steps, which in total account for 1min extra runtime.

	PNN without prior	PNN with prior	CNN
raw images	-	-	39.08% (4m24s)
resizing	28.74% (1m35s)	31.03% (1m34s)	55.17% (11.5s)
processing	45.98% (2m8s)	47.13% (2m4s)	79.31% (8.5s)
processing+Gabor+LDA	91.95% (0.5s)	93.10% (0.5s)	94.25% (6s)
processing+Gabor+PCA+LDA	**94.25%** (0.5s)	**94.25%** (0.5s)	**94.25%** (6s)

Table 3: Accuracy scores per system configuration

Firstly, it is observed that a more refined the system yields to higher prediction accuracies across all models. Particularly, adding the feature extraction and LDA step. Without any pre-handling of the data, the CNN outperforms both PNNs by at least 24 percentage points. The CNN also largely outperforms the PNNs after image pre-processing steps by at least 32 percentage points, while also being significantly faster. However, for the most refined systems the prediction accuracies are comparable, specifically for the system including PCA, the accuracies seem to be equal. The slowest system is given by the PNN applied on raw images, followed by the CNN applied on raw images, while all most refined PNN systems are the fastest, classifying under one second.

Secondly, it is observed that introducing prior probabilities to the PNN discriminant improves PNN classification by about 2 percentage points across all systems aside the full one. Furthermore, adding a prior also does not affect computing time of the FER systems.

	Anger	Contempt	Disgust	Fear	Happiness	Sadness	Surprise
Anger	100	0	0	0	0	0	0
Contempt	0	100	0	0	0	0	0
Disgust	0	0	93.74	6.25	0	0	0
Fear	0	0	0	100	0	0	0
Happiness	0	0	0	0	100	0	0
Sadness	40	0	0	0	0	60	0
Surprise	8.33	0	0	0	0	0	91.67

Table 4: Confusion table of full PNN system with prior (percentages)

Furthermore, Table 10 shows the confusion table normalised over the true population (in percentages) for the full PNN system with prior probabilites (for the confusion tables of the other PNN systems, refer to the Appendix). The table is to be interpreted such that a row represents the true class, whereas a column represents the predicted class. It can be seen that *anger, contempt, fear, happiness* are always correctly classified, while the worst recognised emotion is *sadness* with a recognition rate of only 60%. This is observed across all the PNN systems.

7 Conclusion

The findings of this research reveal that a FER system using image pre-processing steps, HOG for face detection, a Gabor filter bank with 8 orientations, a dual phase feature reduction step using PCA and LDA, and PNN as a classifier achieves an accuracy of 94.25% on the CK+ dataset. This result is higher than the similar works of Fazli et al. (2009) and Pumlumchiak and Vittayakorn (2017) by 5.25 and 3.42 percentage points respectively. It is noticed that *sadness* is particularly badly recognised even when prior class probability is taken into account, so understanding the causes of this exception could be an avenue for further research.

For comparison purposes, this system was compared in terms of prediction accuracy and runtime to the state-of-the arts image classification algorithm, the CNN. Overall, it can be concluded that the traditional FER system proposed by this paper performs similarly to the `Tensorflow` CNN granted the pre-processing, feature extraction and dimensionality

reduction steps. The training time using Google colab on GPU runtime is comparable for both models after dimensionality reduction, however being shorter for the CNNs before dimensionality reduction is performed. As previously mentioned and as shown by the results, the advantages of the PNN are that it discriminates based on a tractable Bayesian decision, such that uncertainty can be injected in the system via prior probabilities on the classes.

Examining the outcomes of the experiments, it is to be noted that the addition of the feature extraction and LDA steps particularly increase the prediction accuracies of the PNN classifiers - that is, by almost 50 percentage points. Similarly, these operations improve the accuracy of the CNN classifiers, though only by approximately 15 percentage points. Identical results after addition of PCA on the feature vector before LDA is further grounds for suspicion that this combination of operations may have a larger effect on the model performance than the choice of the classifier. This leads to question the effect of the LDA and/or PCA algorithm alone versus that of the PNN, suggesting an avenue for further research.

Furthermore, the reader should be aware that the Gabor filter bank parameters and the PNN tuning parameter were found by testing visually the effect of the filters on some selected images of the dataset. To further refine this study, a grid search framework where the PNN model parameters are optimised such that the average prediction score is maximised could be considered. More specifically, the wavelength λ, the aspect ratio γ and the standard deviation of the Gabor kernel σ require special attention as small changes in their values seem to drastically affect the accuracy score of the model. The smoothing parameter of the PNN would preferably also be chosen via optimisation, however Specht (1990) demonstrated that small changes in its value don't affect the accuracy of the model significantly.

Lastly, comparison on other data sets could be an interesting avenue to explore to test the dependency of the model on this specific data set. Non-posed data sets such as FER2013 (Carrier and Courville, 2018) would be interesting to test the model's dependency on the environment that the images are taken in.

Based on these findings, and knowing that human emotion expression and recognition is

intrinsic with uncertainty even for humans themselves, it is suggested to choose the PNN when a real measure of uncertainty is desired. However, if a relatively high accuracy for a less refined model is desired, one could opt for the CNN. In either case, the image pre-processing step is recommended, as it significantly increases the accuracy of the models in exchange of little extra computing time.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Agrawal, S. and Khatri, P. (2015). Facial expression detection techniques: based on viola and jones algorithm and principal component analysis. In *2015 Fifth International Conference on Advanced Computing & Communication Technologies*, pages 108–112. IEEE.

Ashraf, A. B., Lucey, S., Cohn, J. F., Chen, T., Ambadar, Z., Prkachin, K. M., and Solomon, P. E. (2009). The painful face–pain expression recognition using active appearance models. *Image and vision computing*, 27(12):1788–1796.

Bartlett, M. S., Littlewort, G., Fasel, I., and Movellan, J. R. (2003). Real time face detection and facial expression recognition: development and applications to human computer interaction. In *2003 Conference on computer vision and pattern recognition workshop*, volume 5, pages 53–53. IEEE.

Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720.

Bendjillali, R. I., Beladgham, M., Merit, K., and Taleb-Ahmed, A. (2019a). Improved facial expression recognition based on dwt feature for deep cnn. *Electronics*, 8(3):324.

Bendjillali, R. I., Beladgham, M., Merit, K., and Taleb-Ahmed, A. (2019b). Improved facial expression recognition based on dwt feature for deep cnn. *Electronics*, 8(3):324.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Breuer, R. and Kimmel, R. (2017). A deep learning perspective on the origin of facial expressions. *arXiv preprint arXiv:1705.01842*.

Carrier, P.-L. and Courville, A. (2018). The facial expression recognition 2013 (fer-2013) dataset.

Chen, J., Chen, Z., Chi, Z., and Fu, H. (2016). Facial expression recognition in video with multiple feature fusion. *IEEE Transactions on Affective Computing*, 9(1):38–50.

Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59.

Cornejo, J. Y. R., Pedrini, H., and Flórez-Revuelta, F. (2015). Facial expression recognition with occlusions based on geometric representation. In *Iberoamerican Congress on Pattern Recognition*, pages 263–270. Springer.

Cottrell, G. W. and Metcalfe, J. (1990). Empath: Face, emotion, and gender recognition using holons. In *Proceedings of the 3rd International Conference on Neural Information Processing Systems*, pages 564–571.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee.

Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2(7):1160–1169.

Daugman, J. G. (1988). Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on acoustics, speech, and signal processing*, 36(7):1169–1179.

Deng, H.-B., Jin, L.-W., Zhen, L.-X., Huang, J.-C., et al. (2005). A new facial expression recognition method based on local gabor filter bank and pca plus lda. *International Journal of Information Technology*, 11(11):86–96.

Edwards, G. J., Taylor, C. J., and Cootes, T. F. (1998). Interpreting face images using active appearance models. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 300–305. IEEE.

Ekman, P. (2002). Facial action coding system (facs). *A human face*.

Ekman, P. and Friesen, W. V. (1971). Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124.

Fasel, B. (2002). Robust face analysis using convolutional neural networks. In *Object recognition supported by user interaction for service robots*, volume 2, pages 40–43. IEEE.

Fazli, S., Afrouzian, R., and Seyedarabi, H. (2009). High-performance facial expression recognition using gabor filter and probabilistic neural network. In *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, volume 4, pages 93–96. IEEE.

Gao, X., Su, Y., Li, X., and Tao, D. (2010). A review of active appearance models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(2):145–158.

Hu, Y., Zeng, Z., Yin, L., Wei, X., Zhou, X., and Huang, T. S. (2008). Multi-view facial expression recognition. In *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–6. IEEE.

Huang, Y., Chen, F., Lv, S., and Wang, X. (2019). Facial expression recognition: A survey. *Symmetry*, 11(10):1189.

Jumani, S. Z., Ali, F., Guriro, S., Kandhro, I. A., Khan, A., and Zaidi, A. (2019). Facial expression recognition with histogram of oriented gradients using cnn. *Indian Journal of Science and Technology*, 12(24):1–8.

Kanade, T., Cohn, J. F., and Tian, Y. (2000). Comprehensive database for facial expression analysis. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pages 46–53. IEEE.

Kanade, T. et al. (1977). *Computer recognition of human faces*, volume 47. Birkhäuser Basel.

Kanaujia, A., Huang, Y., and Metaxas, D. (2006). Emblem detections by tracking facial features. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pages 108–108. IEEE.

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

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

Lisetti, C. L. and Rumelhart, D. E. (1998). Facial expression recognition using a neural network. In *FLAIRS Conference*, pages 328–332.

Lopes, A. T., De Aguiar, E., and Oliveira-Santos, T. (2015). A facial expression recognition system using convolutional networks. In *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*, pages 273–280. IEEE.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., and Matthews, I. (2010). The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, pages 94–101. IEEE.

Lyons, M., Akamatsu, S., Kamachi, M., and Gyoba, J. (1998). Coding facial expressions with gabor wavelets. In *Proceedings Third IEEE international conference on automatic face and gesture recognition*, pages 200–205. IEEE.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217.

Martinez, A. M. and Kak, A. C. (2001). Pca versus lda. *IEEE transactions on pattern analysis and machine intelligence*, 23(2):228–233.

Matsugu, M., Mori, K., Mitari, Y., and Kaneda, Y. (2003). Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Networks*, 16(5-6):555–559.

Mehrabian, A. and Russell, J. A. (1974). *An approach to environmental psychology.* the MIT Press.

Mohebali, B., Tahmassebi, A., Meyer-Baese, A., and Gandomi, A. H. (2020). Probabilistic neural networks: a brief overview of theory, implementation, and application. *Handbook of Probabilistic Models*, pages 347–367.

Mood, A. M. (1950). Introduction to the theory of statistics.

Munir, A., Hussain, A., Khan, S. A., Nadeem, M., and Arshid, S. (2018). Illumination invariant facial expression recognition using selected merged binary patterns for real world images. *Optik*, 158:1016–1025.

Neggaz, N., Besnassi, M., and Benyettou, A. (2010). Application of improved aam and probabilistic neural network to facial expression recognition. *Journal of Applied Sciences(Faisalabad)*, 10(15):1572–1579.

Ouyang, A., Liu, Y., Pei, S., Peng, X., He, M., and Wang, Q. (2020). A hybrid improved kernel lda and pnn algorithm for efficient face recognition. *Neurocomputing*, 393:214–222.

Owusu, E., Zhan, Y., and Mao, Q. R. (2014). A neural-adaboost based facial expression recognition system. *Expert Systems with Applications*, 41(7):3383–3390.

Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., ter Haar Romeny, B., Zimmerman, J. B., and Zuiderveld, K. (1987). Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368.

Pumlumchiak, T. and Vittayakorn, S. (2017). Facial expression recognition using local gabor filters and pca plus lda. In *2017 9th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 1–6. IEEE.

Rosenblum, M., Yacoob, Y., and Davis, L. S. (1996). Human expression recognition from motion using a radial basis function network architecture. *IEEE transactions on neural networks*, 7(5):1121–1138.

Russell, J. A. (1980). A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161.

Saabni, R. (2015). Facial expression recognition using multi radial bases function networks and 2-d gabor filters. In *2015 Fifth International Conference on Digital Information Processing and Communications (ICDIPC)*, pages 225–230. IEEE.

Sakai, T., Nagao, M., and Kanade, T. (1972). *Computer analysis and classification of photographs of human faces*. Kyoto University.

Seyedarabi, H., Lee, W., and Aghagolzadeh, A. (2006). Automatic lip tracking and action units classification using two-step active contours and probabilistic neural networks. In *2006 Canadian Conference on Electrical and Computer Engineering*, pages 2021–2024. IEEE.

Shih, F. Y. and Chuang, C.-F. (2004). Automatic extraction of head and face boundaries and facial features. *Information Sciences*, 158:117–130.

Shridhar, K., Laumann, F., and Liwicki, M. (2019). A comprehensive guide to bayesian convolutional neural network with variational inference. *arXiv preprint arXiv:1901.02731*.

Slavković, M., Reljin, B., Gavrovska, A., and Milivojević, M. (2013). Face recognition using gabor filters, pca and neural networks. In *2013 20th International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 35–38. IEEE.

Specht, D. F. (1990). Probabilistic neural networks. *Neural networks*, 3(1):109–118.

Stonham, T. (1986). Practical face recognition and verification with wisard. In *Aspects of face processing*, pages 426–441. Springer.

Turk, M. A. and Pentland, A. P. (1991). Face recognition using eigenfaces. In *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*, pages 586–587. IEEE Computer Society.

Verma, K. and Khunteta, A. (2017). Facial expression recognition using gabor filter and multi-layer artificial neural network. In *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, pages 1–5.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pages I–I. IEEE.

Werbos, P. J. (1994). *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*, volume 1. John Wiley & Sons.

Wiskott, L., Krüger, N., Kuiger, N., and Von Der Malsburg, C. (1997). Face recognition by elastic bunch graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):775–779.

Zeng, N., Zhang, H., Song, B., Liu, W., Li, Y., and Dobaie, A. M. (2018). Facial expression recognition via learning deep sparse autoencoders. *Neurocomputing*, 273:643–649.

Zhao, Y., Georganas, N. D., and Petriu, E. M. (2010). Applying contrast-limited adaptive histogram equalization and integral projection for facial feature enhancement and detection. In *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*, pages 861–866. IEEE.

Appendix

In this section, the systems outlined in Table 3 are referred to using the number corresponding to the order in the table. For example, system (1) is corresponds to processing the raw images only.

Google colab notebook code

`https://colab.research.google.com/drive/1nBkBGUY9QeuDeDZboSHTCkp1xa7_V3q4?usp=sharing`

Ratio of explained variance per PCA component on Gabor features after image processing (shown in percentages)

5.84, 5.3 , 5.04, 4.83, 3.59, 2.94, 2.41, 2.16, 2.08, 1.87, 1.8,
1.62, 1.47, 1.35, 1.31, 1.27, 1.2 , 1.17, 1.05, 1.04, 0.98, 0.94,
0.91, 0.9 , 0.86, 0.84, 0.83, 0.78, 0.75, 0.72, 0.7 , 0.69, 0.67,
0.65, 0.64, 0.6 , 0.59, 0.58, 0.56, 0.55, 0.54, 0.52, 0.5 , 0.49,
0.48, 0.47, 0.45, 0.45, 0.44, 0.43, 0.42, 0.41, 0.4 , 0.39, 0.38,
0.37, 0.37, 0.36, 0.36, 0.35, 0.34, 0.34, 0.34, 0.33, 0.32, 0.31,
0.31, 0.31, 0.31, 0.3 , 0.29, 0.29, 0.28, 0.28, 0.27, 0.27, 0.27,
0.26, 0.26, 0.25, 0.25, 0.24, 0.24, 0.24, 0.24, 0.23, 0.22, 0.22,
0.22, 0.22, 0.21, 0.21, 0.21, 0.21, 0.2 , 0.2 , 0.2 , 0.2 , 0.2 ,
0.19, 0.19, 0.19, 0.19, 0.18, 0.18, 0.18, 0.18, 0.18, 0.17, 0.17,
0.17, 0.17, 0.17, 0.16, 0.16, 0.16, 0.16, 0.15, 0.15, 0.15, 0.15,
0.15, 0.15, 0.14, 0.14, 0.14, 0.14, 0.14, 0.14, 0.14, 0.13, 0.13,
0.13, 0.13, 0.13, 0.13, 0.13, 0.13, 0.13, 0.12, 0.12, 0.12, 0.12,
0.12, 0.12, 0.12, 0.12, 0.11

CNN model configurations

The CNNs used in this research are sequential, use the Activation function ReLu, and as optimiser the Adam optimiser. A summary of the models is given below, followed by a detailed summary per system. On systems (1),(2),(3):

1. 2D Convolutional Layer 1
2. Max pooling layer 1
3. 2D Convolutional Layer 2
4. Max pooling layer 2
5. 2D Convolutional Layer 3
6. Dropout Layer
7. Dense Layer 1
8. Dense Layer 2

On systems (4), (5):

1. 1D Convolutional Layer 1
2. 1D Convolutional Layer 2
3. Dropout Layer
4. Max pooling layer 1
5. Dense Layer 1
6. Dense Layer 2

System (1)

```
Model: "sequential_21"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_31 (Conv2D)           (None, 478, 638, 32)      320
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 239, 319, 32)      0
_________________________________________________________________
conv2d_32 (Conv2D)           (None, 237, 317, 64)      18496
_________________________________________________________________
max_pooling2d_21 (MaxPooling (None, 118, 158, 64)      0
_________________________________________________________________
conv2d_33 (Conv2D)           (None, 116, 156, 64)      36928
_________________________________________________________________
flatten_20 (Flatten)         (None, 1158144)           0
_________________________________________________________________
dense_40 (Dense)             (None, 64)                74121280
_________________________________________________________________
dense_41 (Dense)             (None, 10)                650
=================================================================
Total params: 74,177,674
Trainable params: 74,177,674
Non-trainable params: 0
```

System (2)

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_34 (Conv2D)           (None, 46, 62, 32)        320
_________________________________________________________________
max_pooling2d_22 (MaxPooling (None, 23, 31, 32)        0
_________________________________________________________________
conv2d_35 (Conv2D)           (None, 21, 29, 64)        18496
_________________________________________________________________
max_pooling2d_23 (MaxPooling (None, 10, 14, 64)        0
_________________________________________________________________
conv2d_36 (Conv2D)           (None, 8, 12, 64)         36928
_________________________________________________________________
flatten_21 (Flatten)         (None, 6144)              0
_________________________________________________________________
dense_42 (Dense)             (None, 64)                393280
_________________________________________________________________
dense_43 (Dense)             (None, 10)                650
=================================================================
Total params: 449,674
Trainable params: 449,674
Non-trainable params: 0
```

System (3)

Layer (type)	Output Shape	Param #
conv2d_37 (Conv2D)	(None, 62, 62, 32)	320
max_pooling2d_24 (MaxPooling	(None, 31, 31, 32)	0
conv2d_38 (Conv2D)	(None, 29, 29, 64)	18496
max_pooling2d_25 (MaxPooling	(None, 14, 14, 64)	0
conv2d_39 (Conv2D)	(None, 12, 12, 64)	36928
flatten_22 (Flatten)	(None, 9216)	0
dense_44 (Dense)	(None, 64)	589888
dense_45 (Dense)	(None, 10)	650

Total params: 646,282
Trainable params: 646,282
Non-trainable params: 0

System (4)

Layer (type)	Output Shape	Param #
conv1d_22 (Conv1D)	(None, 4, 64)	256
conv1d_23 (Conv1D)	(None, 2, 64)	12352
dropout (Dropout)	(None, 2, 64)	0
max_pooling1d_11 (MaxPooling	(None, 1, 64)	0
flatten_24 (Flatten)	(None, 64)	0
dense_48 (Dense)	(None, 64)	4160
dense_49 (Dense)	(None, 10)	650

Total params: 17,418
Trainable params: 17,418
Non-trainable params: 0

System (5)

Layer (type)	Output Shape	Param #
conv1d_26 (Conv1D)	(None, 4, 64)	256
conv1d_27 (Conv1D)	(None, 2, 64)	12352
dropout_2 (Dropout)	(None, 2, 64)	0
max_pooling1d_13 (MaxPooling	(None, 1, 64)	0
flatten_26 (Flatten)	(None, 64)	0
dense_52 (Dense)	(None, 64)	4160
dense_53 (Dense)	(None, 10)	650

Total params: 17,418
Trainable params: 17,418
Non-trainable params: 0

Confusion matrices of PNN systems with priors (in the order of Table 3)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	15.38	15.38	30.77	0	0	15.38	23.03
Contempt	0	100	0	0	0	0	33.33
Disgust	18.75	6.25	37.50	0	12.50	6.25	18.76
Fear	0	25	12.50	0	15	0	37.50
Happy	5.56	5.56	27.78	11.11	33.33	5.56	11.11
Sadness	40	20	20	0	0	0	20
Surprise	8.33	4.17	25	4.17	0	16.67	41.67

Table 5: Confusion table of PNN system (2)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	38.46	15.38	30.77	0	0	0	15.38
Contempt	0	66.67	0	0	0	0	33.33
Disgust	25	6.25	43.75	0	25	0	0
Fear	0	12.5	25	50	0	0	12.5
Happy	0	11.11	22.22	5.56	55.56	0	5.56
Sadness	60	20	20	0	0	60	0
Surprise	4.17	16.67	0	4.17	8.33	12.5	54.17

Table 6: Confusion table of PNN system (3)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	100	0	0	0	0	0	0
Contempt	0	66.67	0	33.33	0	0	0
Disgust	0	0	100	0	0	0	0
Fear	0	0	12.5	87.5	0	0	0
Happy	0	0	0	0	100	0	0
Sadness	20	20	0	0	0	60	0
Surprise	8.33	0	0	0	0	0	91.67

Table 7: Confusion table of PNN system (4)

Confusion matrices of PNN systems without priors (in the order of Table 3)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	15.38	15.38	30.77	0	0	23.08	15.38
Contempt	0	100	0	0	0	0	0
Disgust	125	12.5	31.25	0	12.50	6.25	12.5
Fear	0	25	12.50	0	25	0	37.50
Happy	5.56	5.56	27.78	11.11	33.33	5.56	11.11
Sadness	40	20	20	0	0	0	20
Surprise	12.5	4.17	25	4.17	0	16.67	37.50

Table 8: Confusion table of PNN system (2)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	38.46	15.38	30.77	0	0	0	15.38
Contempt	0	66.67	0	0	0	0	33.33
Disgust	25	18.75	37.5	0	18.75	0	0
Fear	0	12.5	25	50	0	0	12.5
Happy	0	11.11	22.22	5.56	55.56	0	5.56
Sadness	60	20	20	0	0	0	0
Surprise	4.17	16.67	0	4.17	4.17	16.67	54.17

Table 9: Confusion table of PNN system (3)

	Anger	Contempt	Disgust	Fear	Happy	Sadness	Surprise
Anger	100	0	0	0	0	0	0
Contempt	0	66.67	0	33.33	0	0	0
Disgust	6.25	0	93.75	0	0	0	0
Fear	0	0	12.5	87.5	0	0	0
Happy	0	0	0	0	100	0	0
Sadness	20	20	0	0	0	60	0
Surprise	8.33	0	0	0	0	0	91.67

Table 10: Confusion table of PNN system (4)