ROBUST CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS

A Thesis presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

MUHIND SALIM HMOUD ALRADAD

Dr. Tony Han, Thesis Supervisor

MAY 2015

The undersigned, appointed by the Dean of the Graduate School, have examined the thesis entitled:

Robust Classification with Convolutional Neural Networks

Presented by: Muhind Salim Hmoud Alradad

A candidate for the degree of Master of Science and hereby certify that, in their opinion, it is worthy of acceptance.

Professor Tony Han

Professor Zhihai He

Professor Kannappan Palaniappan

Dedication

Thanks be to God for His guidance and care throughout my time here at the University of Missouri in Columbia as a graduate student working toward attaining the Electrical Engineering degree.

I would like to thank all my lab mates who have made my research experience wonderful and less stressful. Thanks to all my friends for their help and support. I could never have completed my master's research without the support of my family. I offer special thanks to my wife for her support and patience, and I am deeply grateful to my parents for everything.

To all who have been kind enough to stand with me in the good times and bad, thank you.

Acknowledgements

I would like to thank Dr. Tony Han for his wise guidance during my studies and research. This thesis would not have been possible without the intelligent guidance from my adviser. I also would like to thank my committee members, Dr. Zhihai He and Dr. Kannappan Palaniappan, for their time and willingness to offer advice and guide me in my studies.

Table of contents

List of Illustrations

List of Tables

Abstract

For image classification, I achieved the current state-of-the-art performance by using methods based on Convolutional Neural Networks (CNNs). Face image analysis requires both effective feature extraction and classifier systems. This research considers the deep learning algorithm and addresses its working for classification tasks. I shows how to choose between averages and max pooling in CNN architecture. This method improved results by using homogeneous networks that combined average and max pooling together. For gender classification based on facial features, CNNs proved effective for simultaneously extracting relevant features and classifying them. State-of-the-art performance was obtained on two unconstrained datasets: Labeled Faces in the Wild (LFW) and Images of Groups of people dataset. CAS-PEAL-RL dataset was used to test our systems under constrained conditions where all the images were collected inside the lab. For age estimation, I achieved good performance using images of groups of people dataset where the people in the images have been divided into seven groups according to their age.

1. Chapter 1

Introduction

Extracting semantic contents from images are tasks that have attracted a lot of research interest. Although recognizing and category images seems like a simple task, more sophisticated datasets require a robust classifier that can handle the big data demands of 21st century, researchers are still looking for larger, faster and more economical solutions. Designing a system can extract the relevant information from an image efficiently requires computer vision and machine learning.

There are some computer vision systems have successfully closed the gap with human performance, such as handwritten digit recognition [1]. However, today's advanced classification systems encounter problems when there are a large number of objects or high similarity between these objects. For instance, the ImageNet dataset has several thousand categories, and it is very hard to get the right categories if the difference are more subtle in terms of variety or type. Examples of such cases are recognize different car models or different kinds of dogs.

Another difficult classification problem is face recognition, in which we try to classify humans using their facial images. Facial images have a great deal of relevant information compared to other object images. All faces have the same general characteristics such as a nose, a mouth and two eyes. Distinguishing the difference between such characteristics requires complex systems that can automatically do feature extraction and classification.

Designing a robust classifier requires a great deal of training data, automated learning and resources that is justified in terms of the importance of the application. For example, capturing someone's handwritten signature is helpful in some studies. Knowing all the objects in the street will automatically prevent car drivers from an accident. Driverless cars can help us with long journeys. Robust computer vision systems will have a vast impact on blind people and help them act independently in the community. All these applications and others require complex systems that can learn by themselves from the data samples.

Why do we need machine learning? Imagine you want to set some parameters for a computer to observe the difference between dogs and cats. Frist, you set your parameters according to size by saying that cats are smaller than dogs. However, there are some kinds of dogs that are smaller than cats. What if the cat is closer to the camera than the dog making the cat appear bigger. Another setting would depend on shape, but not all cats or dogs have the same shape. It seems that setting the parameters manually would be difficult if not impossible. To solve this problem we need to make sure machines can set the parameters automatically. Teaching computers to learn from data samples is called machine learning.

There are plenty of algorithms that try to assure that the machines make the right decision by setting their parameters automatically. Some systems are trying to find a hidden structure in the data according to the distance between data-points (unsupervised learning). Other systems conclude their functional from a dataset that has labels (supervised learning). In this work, I will focus on the supervised learning part where every image is paired with a label.

Chapter 2 of this thesis will present a literature review about the convolutional neural network. I shall present some techniques that increase the accuracy for Convolutional Neural Networks (CNNs). To test system performance, the Modified NIST or MNIST dataset demonstrated in [1] was chosen.

Classifying facial images according to gender and age will be the tasks discussed in the third chapter. Three face image datasets were used to demonstrate our results. State-of-the-art performance will be shown in the images.

In chapter 4, I will summarize the work accomplished in this thesis and recommend future work.

2. Chapter 2

Background Literature

2.1 Neural Networks

Artificial Neural Networks (ANNs) or short Neural Networks (NNs) have been modeled to handle complex tasks by following patterns that work like the human brain. NNs contain interconnected units with a very simple functions. When combined these simple units can build a complex classification function.

The first known structure was built in 1958 by Rosenblatt [2]. As illustrated in Figure (2.1), his network was very simple. The network consisted of one neural that had multiple inputs and one output. The output is the sum of all the weighted input x_i and the bias b as described in Equation (2.1).

$$f(x) = \varphi(b + \textstyle\sum_{i=1}^{n} x_i \,.\, w_i) \tag{2.1}$$

where φ the Heavyside step function is defined as

$$\varphi(x) = \begin{cases} 1 & if\ x \geq 0 \\ 0 & else \end{cases} \tag{2.2}$$

This network was used to classify two classes: if the result of the summation is greater than or equal to zero, than the network votes for the first class. Otherwise, if the result is less than zero, then the network votes for the other class.

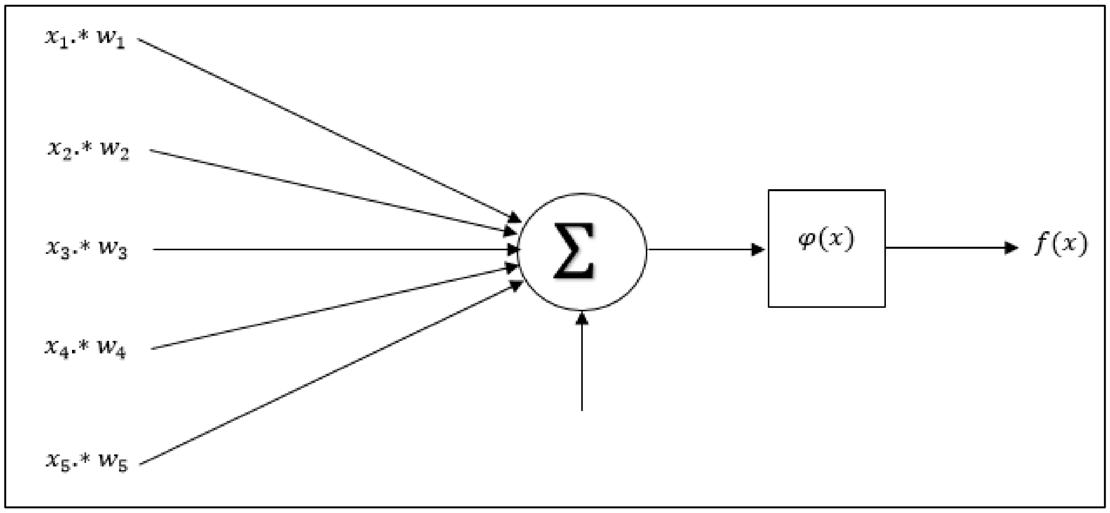

Figure 2.1: The neural network was proposed by Rosenblatt [2]

This simple architecture cannot handle tasks when the data are not linearly separable [3]. This weakness motivates using Multi-Layer Perceptrons.

Multi-Layer Perceptrons (MLPs) are a feedforward artificial neural network architecture that have one or more than one hidden layer. Figure (2.2) illustrates an MLP with one hidden layer containing neurals where each neural in each hidden layer represents one perceptron.

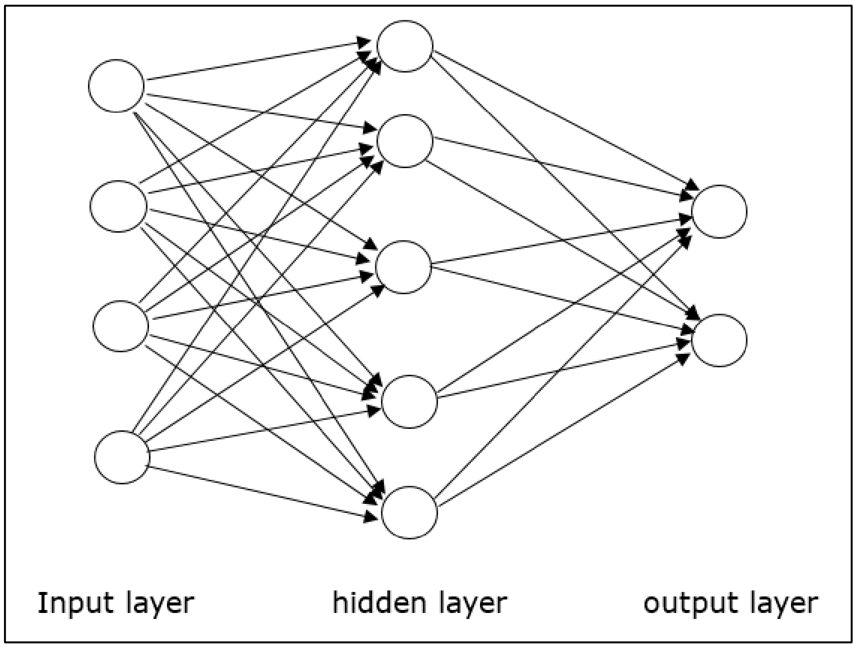

Figure 2.2: Multi-Layer Perceptron with a single hidden layer.

The hidden layer receives data from the input layer and then maps it to the output layer. The weights in the MLP layers are generated from the training data. The most widely used algorithm for training the parameters is backpropagation [4].

2.1.1 The Backpropagation Algorithm

The backpropagation algorithm is a well-known algorithm and widely used due to its simple and efficient structure. It is a supervised learning algorithm that applies the gradient descent technique to minimize an error E [5].

There are two parts in this algorithm

- Forward: The first part occurs when the training example is presented to the neural network layer by layer from input to output.
- Backward: The second part begins at the forward end, which is started by calculating the error and propagates it layer by layer from the output to the input, and updates the weights and biases accordingly.

With one Perceptron neural networks, the Heavyside function in equation (2.2) was applied. However, to accomplish backpropagation, a function that has continued derivatives is required. There are two types of active functions depending on the kind of mapping needed from the input to the output. The first one is a nonlinear function such as sigmoid, which is a monotonically increasing function. The most commonly used sigmoid function is the standard logistic or hyperbolic tangent function shown in Equation (2.3) and (2.4) respectively. The second type of an activation function is the linear function (ReLU) [6] illustrated in Equation (2.5).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

$$f(x) = \tanh(x) \tag{2.4}$$

$$f(x) = \max(0, x) \tag{2.5}$$

Figure (2.3) shows the three activation functions where the x-axis has 100 points between [-10, 10]; the output of the standard logistic is between [0, 1] (Figure (3-A)) the output of the hyperbolic tangent is between [-1, 1] and the output of a rectified linear unit (ReLU) maps linearly to the output where nonpositive points maps to zero.

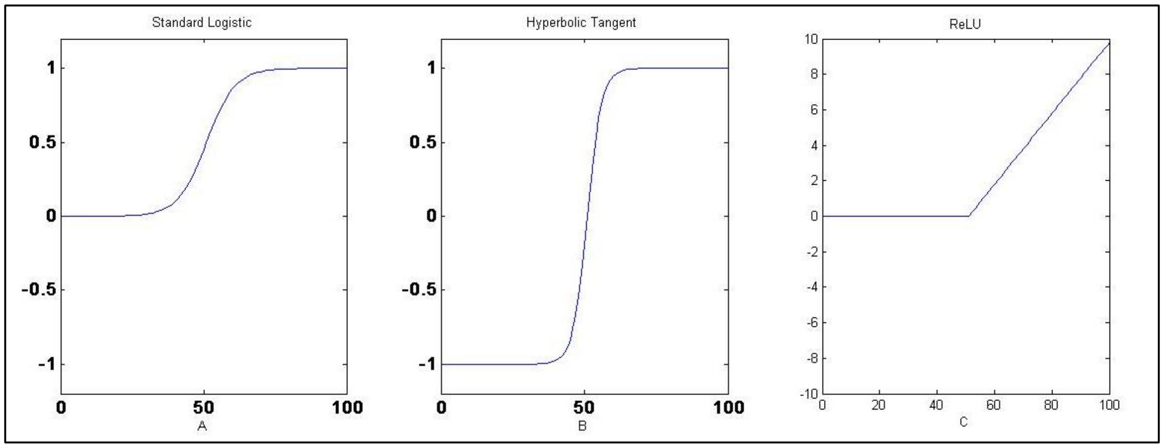

Figure 2.3: Three activation functions where A is the standard logistic function, B is the hyperbolic tangent function, and C is the ReLU function.

During backpropagation assume the dataset has points where inputs are paired with outputs by a mapping function which we want to learn. Let's assume that we have a NN with one hidden layer. For every data point presented to the NN, forward through the network and get the errors then backward pass. Every neural in the hidden layer will work the same as one Perceptron but with a different activation function. By rewriting the first two equations (2.1 & 2.2).

$$Y^l = W^l X^{l-1} \tag{2.6}$$

$$X^l = f(Y^l) \tag{2.7}$$

where W^l is the weight at layer l. At the end of the forward pass, the error squared function is applied.

$$E_n^l = \frac{1}{2}(t_n^l - y_n^l)^2 \tag{2.8}$$

where E_n is the error of the network, t_n is the target value of the data point n, and y_n is the output of the network for point n. The error for the system is the average over the data points.

$$E = \frac{1}{N}\sum_{n=1}^{N} E_n \tag{2.9}$$

During backpropagation, we get the derivative of the error with respect to X^l at every layer using Equation (2.8).

$$\frac{\partial E^l}{\partial X^l} = X_n - t_n \tag{2.10}$$

Calculate the error in the $l-1^{th}$ layer using the chain rule.

$$\frac{\partial E^l}{\partial W^l} = \sum \frac{\partial X^l}{\partial Y^l}\frac{\partial Y^l}{\partial W^l}\frac{\partial E^{l+1}}{\partial X^{l+1}} \tag{2.11}$$

where $\frac{\partial X^l}{\partial Y^l}$ is the derivative of the activation function. For instance, if the hyperbolic tangent is applied:

$$f'(Y) = \tanh'(Y) = 1 - \tanh(Y)^2 \tag{2.12}$$

The second term $\frac{\partial Y^l}{\partial W^l}$ is the derivative of Equation (2.6) with respect to W

$$\frac{\partial Y^l}{\partial W^l} = \frac{\partial}{\partial W^l}\left(W^l\, X^{\mathrm{l}-1}\right) = X^{\mathrm{l}-1} \tag{2.13}$$

The last term is just the error from the previous layer. From Equation (2.10), the error for the first node is the difference between the labels for the target data and the output of the networks. Combing all parts, we have.

$$\frac{\partial E^l}{\partial W^l} = f'(Y^l)\sum X^{l-1}\frac{\partial E^{l+1}}{\partial W^{l+1}} \tag{2.14}$$

Update the weights at time t, the new weights are:

$$W(t) = W(t-1) - \eta\frac{\partial E}{\partial w} \tag{2.15}$$

where η represents the learning rate. In summary the standard backpropagation algorithm is shown below.

Algorithm 1: standard backpropagation for NN

```
-Initialized the weights to small random value
-set the learn rate
-t=1
Repeat
-Do forward pass
   For l=1 to L do % where I the number of the network layers
      $Y^l = W^l\, X^{l-1}$ ,        $X^l = f(Y^l)$
      $l = l + 1$
-Get the error
      $\frac{\partial E^l}{\partial X^l} = X_n - t_n$
   End for
-Do backward pass
   For l=L to 1 do
      $\frac{\partial E^l}{\partial W^l} = f'(Y^l)\sum X^{l-1}\frac{\partial E^{l+1}}{\partial W^{l+1}}$
```

```
-update the weight W
        $W(t) = W(t-1) - \eta \frac{\partial E}{\partial w}$
        $l = l - 1$
    End for
t=t+1
Until E<e where e the smallest error can be reached or t> max t
```

My thesis focuses on the classification task which is required to label every image by indicating whether it belongs to one of the n classes. When we have two classes to choose from, one output of an NN can be used with sample labels; hence, that class label would be one and zero for the other class [7]. To separate such classes, logistic sigmoid function is used as an activation function.

$$f(x) = (1 + e^{-x})^{-1} \quad f'(x) = f(x)(1 - f(x)) \tag{2.16}$$

2.1.2 Online Training Vs Offline Training

In general, there are two main principles used to update the weights in Equation (2.15). When the weights are updated immediately after observing each data point this is known as online training. On the other hand, when the weights are updated after observing the whole data samples, this is known as offline training. Both online training and offline training have advantages and disadvantages.

The online training is faster than the offline one. To show that, let's consider a dataset which has 100 images. The weights are updated 100 times during one epoch on the online training, while the weights are updated once a time in the offline case. When data constantly changes over time, it can easily be detected by online training because the weights will be updated according to every point in the data. In the offline case, the weights are updated according to the average change for all the points in the dataset [7].

The online training usually behaves better due to the noise changing in the dataset. In the practical dataset with nonlinear systems, there are more than one local minimum on the data of different gradients. Finding a local minimum that is closer to the global is the goal. In the online training, the noise in the data can help the weights to cause jumping from one minimum to another until the global rate is reached. On the other hand, keeping jump from attaining local minimum to another may not lead to network convergence. In the offline training, convergence is guaranteed or at least more understandable [5].

Compromise between online and offline training can occur when updating of the weights happens after a small data-batch, which is called mine-batch training. These small-batches would be equal in size and have n number of points, where $1<n<N$ where N is the number of all data points.

2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a kind of feeding forward neural network where every single node can be used to apply filters through overlapping regions. The processing occurs in an alternative fashion between convolution and sub-sampling layers followed by one or more fully connected layers such as standard multilayer perceptron (MLP). This architecture has various benefits compared to the standard Neural Networks.

The NNs have been successfully applied to features that have been extracted from other systems, which means that the performance of NNs depends on matching relevant features that can be obtained. Another way to use NNs is to apply them directly to the raw pixel of the image. However, if the images have high dimensions, more parameters are needed because the hidden layer would be fully connected. To tackle this problem CNNs could be applied. The CNNs depend on sharing the weights, which reduces the numbers of parameters.

The convolution layers apply a local filter to the input image, which leads to a better classification there are correlation in the neighborhood pixels of the same image. In other words, the pixels of the input images can have some correlations with each other. For instance, the nose is always between the eyes and the mouth in face images. When we apply the filter to a subset of the image, we will extract some local features. By combining them subsequently, we will get the same format as the original image but with less dimensional image. These kinds of formats are not found in the fully connected layers.

For the classification task, we would expect that objects in the same dataset have the same structure inside every image. In the face detection task, we look for the same face structures with different locations inside each image, so it is useful to treat all the images with the same local filter that is locally applied to pixels.

Figure (2.4) shows classical CNNs that were presented in [1] and have been successfully applied to MNIST applications.

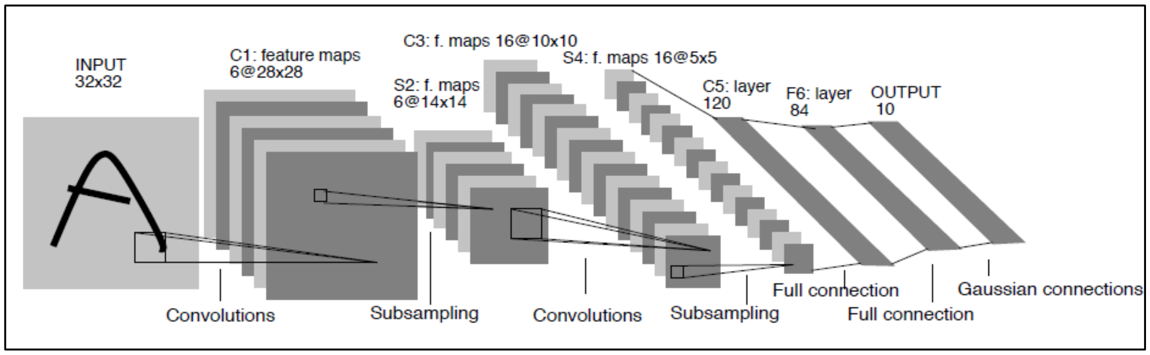

Figure 2.4: Convolutional Neural Network (LeNet5 [1]).

2.2.1. Convolutional Layers

The convolution layers can be mainly divided into two parts: Part 1 is a linear feature mapping that can be done by applying fixed size filters on the output of former layers.

$$Y^l = W^l \otimes X^{l-1} \tag{2.17}$$

where $\otimes$ denotes the convolution operator.

Part 2 involves convolutional layers which is usually a nonlinear mapping as the sigmoid function.

$$X^l = f(Y^l) \tag{2.18}$$

2.2.2. Pooling Layers

Pooling layers usually receive their inputs from convolutional layers. The pooling or the so called subsampling layers is average or max operators over small squared areas. Based on the operator used, these layers are called average pooling or max pooling.

The average pooling can be defined as:

$$s_i = \frac{1}{n} \sum_{i \in R_j}^{n} h_j \tag{2.19}$$

where h is some pixel in the sub-region R_j from the features mapping, and n is number of features in the sub-region where subsampling is required.

The max pooling can be defined as:

$$s_i = \max_{i \in R_j} h_i \tag{2.20}$$

The goals of pooling layers are to reduce the feature mapping sizes and provide a connection to the local neighborhood of the convolutional layer feature by doing their operations on a sequence of mapping features. Both the average and max pooling operators have some drawbacks. The average pooling pays attention to all elements in the pooling region even those that have ≤ zero value, which leads to a reduction in weight magnitudes. On the other hand, the max pooling can easily overfit the network. To avoid these disadvantages, Matthew and Rob Fergus have designed a network with stochastic pooling [8].

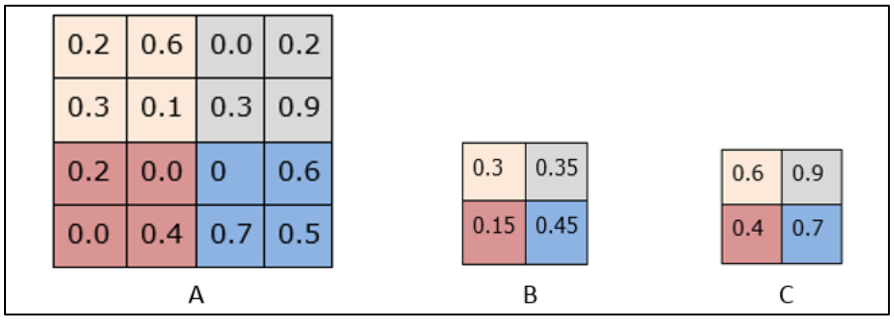

Figure 2.5: The pooling operations: (A) represents feature mapping from a previous layer, (B) represents the features that resulted from Average pooling, and (C) represents features that resulted from max pooling.

2.2.3. Fully Connected Layers

After alternating between the convolutional and the subsampling operations, the features present to the fully connected layers. Normally there is at least one layer in the fully connected part. They work as classifiers in the system and act as Multi Neural Networks.

2.2.4. Backpropagation Algorithm for CNN

There are two passes in the backpropagation algorithm, the forward and backward pass. In both passes, the network acts the same as the NN in the fully connected layer(s). See Section (2.1).

Forward Propagation

To simplify illustration, we assume our CNN has only one convolutional layer, one subsampling layer, and one fully connected layer.

- For convolutional layers, suppose we have an image x that has a size of $m \times m$ and a weight kernel w that has a size of k. So we shall have an output that has a size of $(m-k+1) \times (m-k+1)$ after we convolve the input image with the kernel. The convolutional process is a dot product between the weight and part size of the input that has a size equal to the weight; after that, we sum over all the dot product results.

$$y_{ij}^{l} = \sum_{a=1}^{m} \sum_{b=1}^{m} w_{ab}^{l} x_{(i+a)(j+b)}^{l-1} \qquad (2.21)$$

  where l denote the current layer and *i, j* defines the location of the next pixel in the output of the *l* th layer.

  Every convolutional layer has a normalization part defined as:

$$x_{ij}^{l} = f\left(y_{ij}^{l} + b^{l}\right) \qquad (2.21)$$

  where $f(\cdot)$ is the normalization function and commonly chosen to be the logistic (sigmoid) function and b^{l} is the bias.

- The subsampling layer: As noted in Section (2.2.2), there are two types of these layers. Neither of them have weights nor a normalization part. The output's size from this layer will drop to half if we have a kernel size of (2 x 2).

Backward propagation

The back-propagation process starts from the end layer to the first layer. This process would be similar to Neural Networks on the fully connected layers, which was discussed in Section (2.1).

- Backpropagation for the subsampling layer: The pooling layers need not to have any trainable parameters to be updated. Those kinds of layers can only serve to reduce the size of the features mapping, so, in the backward pass, there is no derivative operation required. In the forward pass, we do subsampling over a square area that is reduced to a single value after the operation. In the backward pass, it is required to return a single value from the error to same size of the squared area. Let's call it dissampling. The dissampling operation depends on the kind of subsampling required. In the case of average pooling, the errors that computed from the layer before the pooling layer distribute on the square area equally. In the max pooling case, the error forwards directly to the place where the feature in the max pooling originated from and the remaining spaces are filled by zeros.
- The backpropagation in the convolutional layers: If we know what errors occurred in the layer before the convolutional layers, say E, then we can find the error in the convolutional layer. Note that we have a square kernel that has a size of $k \times k$. Hence, we need to

perform the chain rule and consequently find the sum over all the regions.

$$\frac{\partial E^l}{\partial W_{ij}^l} = \sum_{i=1}^{m-k}\sum_{j=1}^{m-k} \frac{\partial E^{l+1}}{\partial Y^l}\frac{\partial Y^l}{\partial W_{ij}^l} \tag{2.22}$$

From Equation (2.21) $\frac{\partial Y^l}{\partial W_{ij}^l} = x_{ij}^{l-1}$ and we can get $\frac{\partial E^{l+1}}{\partial Y^l}$ by applying the chain rule again.

$$\frac{\partial E^{l+1}}{\partial Y^l} = \frac{\partial E^{l+1}}{\partial x_{ij}^l}\frac{\partial x_{ij}^l}{\partial y_{ij}^l} \tag{2.23}$$

where $\frac{\partial x}{\partial y}$ from Equation (2.21) equals the derivative of the activation functions. By putting all the terms together, we get.

$$\frac{\partial E^l}{\partial w_{ij}^l} = \sum_{i=1}^{m-k}\sum_{j=1}^{m-k} \frac{\partial E^l}{\partial x_{ij}^l}\, x_{ij}^{l-1}\, f'(y_{ij}^l) \tag{2.24}$$

After that we update the weights accordingly using equation (2.15).

2.3. Handwritten Digits Using MNIST Dataset

To test the system performance, I pick the most classical dataset (MNIST). The MNIST database is a hand digits writing dataset which consists of 70,000 images. These images have been divided by a 6:1 ratio for training and testing, respectively [1]. It is a subset from the

NIST dataset where all the images have been resized to (28 x 28) pixels. Figure (2.6) shows random samples from the dataset.

The NIST dataset used to train the MNIST system is collected from NIST's Special Database 3 (SD-3) and Special Database 1 (SD-1), which contains binary images of handwritten digits. SD-3 uses Census Bureau employees while SD-1 uses high school students. The final training utilizes two sets from SD-3 with 250 writers each, which together creates 60,000 training patterns. However, these 60,000 examples are divided into two subsets of 30,000 each with 250 writers for each set. The final MNIST set comes from the last group of SD-3 250 writers, while the 10,000 MNIST set comes from a subset made up of 5,000 from SD-1 (the high school students) and 5,000 from SD-3 (the Census Bureau employees) [1].

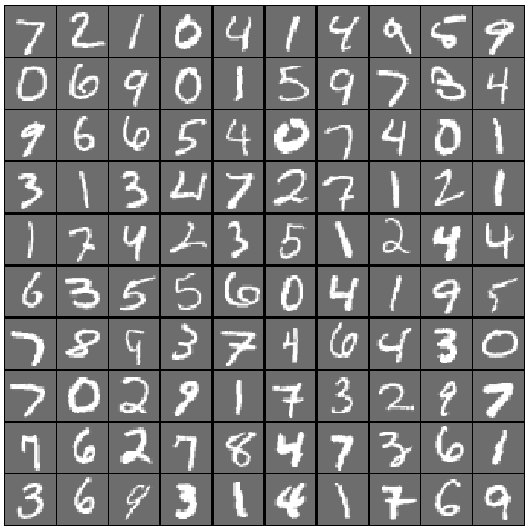

Figure 2.6: Samples of MNIST dataset.

2.3.1. Experiment and Results

I use CNNs to train and test the dataset. In this experiment, I focus on the effects of choosing between average and max pooling layers. All data samples in the dataset are normalized to be between one and zero. The networks consist of a fixed size of layers with two convolutional layers that have kernels sized to (5x5). Every convolutional layer is followed by a pooling layer with active region (2x2) pixels. The final layer is the fully connected layer. Table (2.1) shows the number of feature mapping.

Table 2.1: Size of the proposed CNNs that is used in the MNIST dataset experiences.

Layers	Conv.	pooling	Conv.	Pooling	Fully connected
Number of feature	6	6	12	12	1
Kernel mapping	5x5	2x2	5x5	2x2	---
Feature mapping	24x24	12x12	8x8	4x4	192

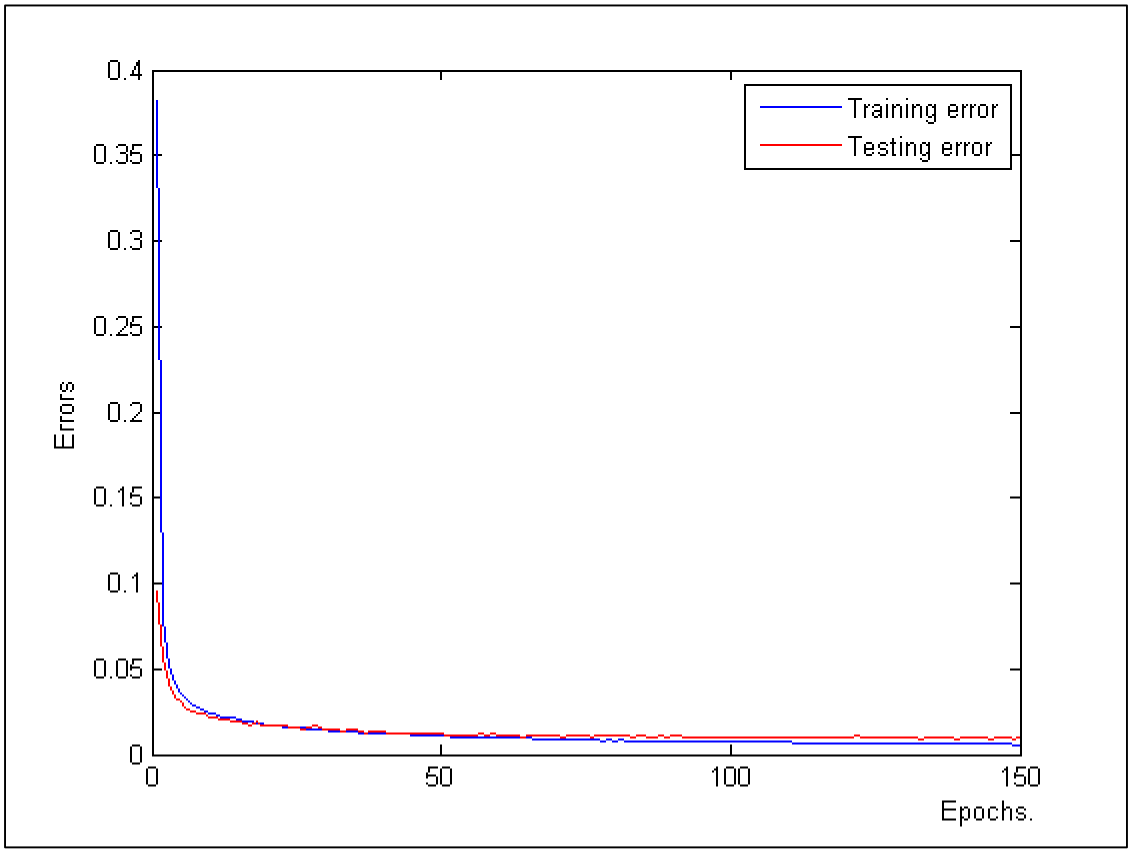

Figure 2.7: MNIST dataset results for both training and testing validity with CNN that have Average pooling for both subsampling layers. The CNN achieved more than 99% accuracy in the test validity.

Figure (2.7) shows the errors of the CNNs for both training and testing data over 150 epochs. The networks in Figure (2.7) apply average pooling for both layers. Testing errors decreases to below 0.01 after 90 epochs and the best accuracy achieved was 99.08%. On the other hand, the results presented in Figure (2.8) belong to the network that has max pooling in both subsampling layers. The error drop very fast, with a reading of 0.01 after 43 epochs. However, network parameters suffer from overfiting for the rest of the training time. The overfiting occurs when the system learns not only the data classification function but also the data noise. The overfiting leads the network to perform good in the training samples and bad in the test samples.

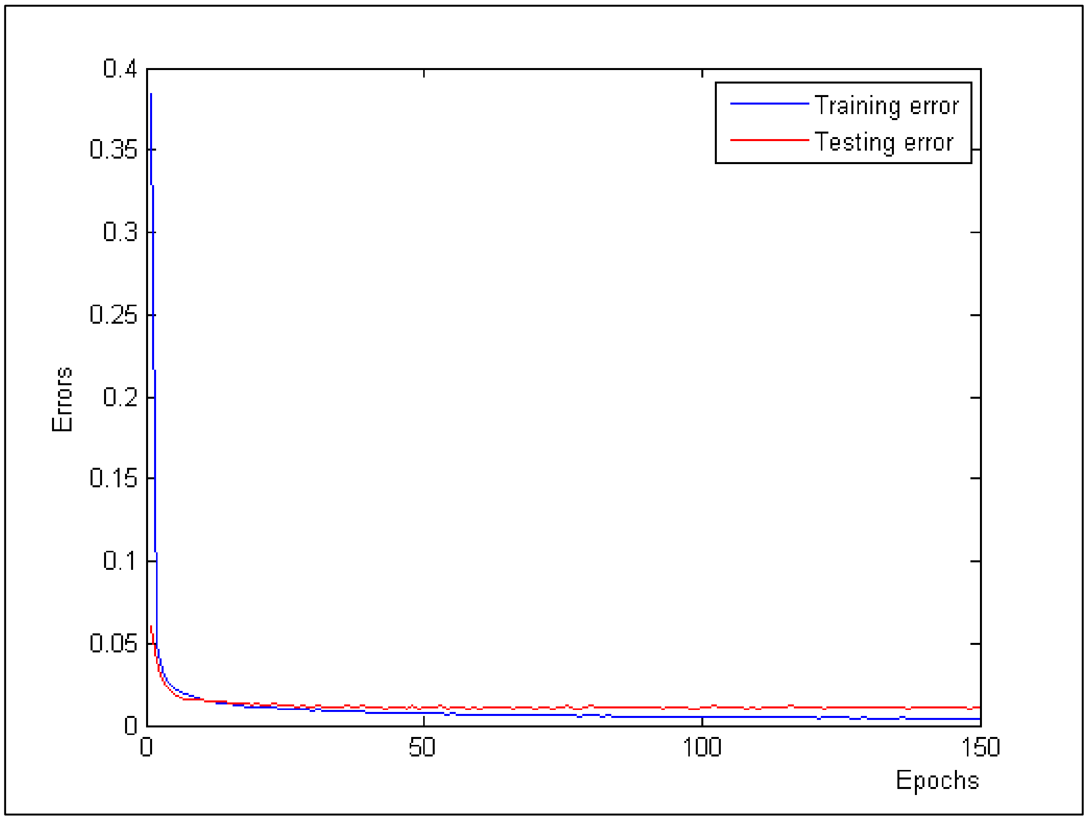

Figure 2.8: MNIST dataset results for both training and testing validity with CNN that has Max pooling for both subsampling layers. The CNN achieved 99% accuracy in the test validity after 42 epochs.

Pooling layers aim to reduce the size of mapping features from the previous convolutional layer by ways that keep a grade of local features translation invariance to following layers. For the first case at which both of the pooling layers are average, neighborhood features are taken into determining the average for them. This operation would have better accuracy, but it is time consuming due to the fact of smoothing out the features, which in turns prevents the network from fast learning. In the case where both pooling layers are max operators, it converges faster because the only high value feature magnitudes are transferred to the next layer, which means the sharp edges of the images will be retained. This kind of learning in which data and noise are both learned are called overfitting.

To avoid the overfiting effect of max pooling layers and the slow convergence of average pooling, we use homogeneous system with both average and max pooling. The next two figures show the performance of networks where the first pooling layer is average and the second one is max and vice versa, respectively.

Figure (2.9A) shows the networks performance when the average pooling is used after the first convolutional layer and max pooling after the second one. The error on the test validity dropped to 0.01 after 41 epochs and to 0.0092 after 140 epochs. The best performance for this size of network, shown in Table (2.1), is when the max pooling layer was used at first, and average pooling was used second as indicated by Figure (2.9B) where the error drops down to 0.0082. The best obtained accuracy occur-s when the numbers of feature mapping increased to 20 and the achieving accuracy is 99.36%. Thus our network missed only 64 samples out of 10,000 test data samples.

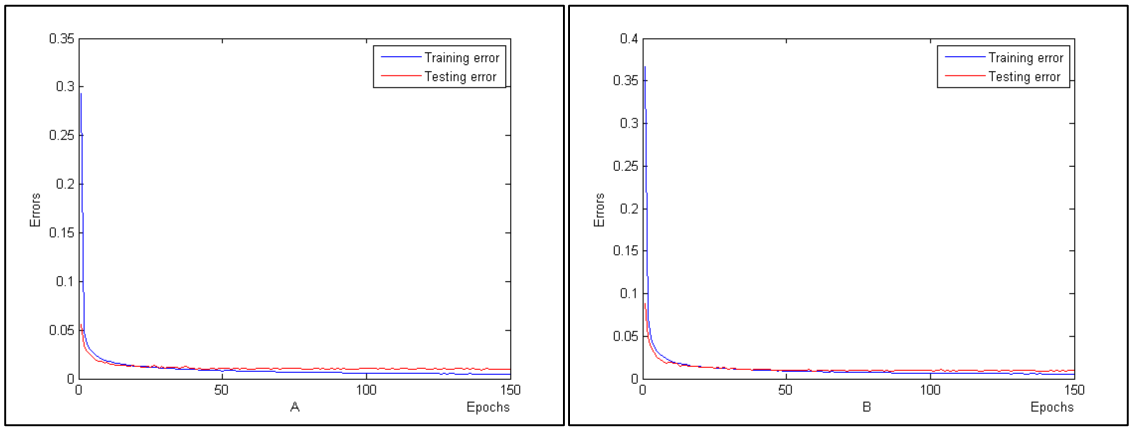

Figure 2.9: MNIST dataset results with homogeneous CNN, (A): average pooling for the first layer and Max pooling for the second one. (B): Max pooling for the first layer and Average pooling for the second one.

In summary, we implement CNNs and obtain results that are based on four kinds of setup with differences in the pooling layers. The best accuracy is achieved when the CNNs has a max pooling followed by an average pooling layer.

3. Chapter 3

Gender and Age Classification

3.1. Introduction

This chapter discusses an automatic gender classification and age estimation by relying on facial images. The goal of this research is to obtain good accuracy by doing some preprocessing of the images. Classifying face images as either a male or female is an important task that has been extensively investigated in the last two decades. Almost every paper in literature addressing this issue has its own method and most of them have used Support Vector Machines (SVMs) [9, 10] and NNs [9, 11, 12] as classifiers. Most previous classification methods in literature depend on global features extraction from whole faces.

Successful gender classification and age estimation require extracting local features. To produce this kind of features, CNN has been applied. We have built our network using the techniques described in chapter 2.

There are variety of applications for gender classification. Knowing the sex and age of customers is required by store managers. Stores that cater to a certain gender have to offer goods that meet their target group's needs. Needless to say, there is a huge difference between things young people purchase and the things that elderly people purchase.

Gender classification should be done prior to other facial image analysis. Under the assumption that we have equal candidates for both genders, this step can save half of the time. It not only reduces the speed of processing, but also increases accuracy by splitting the images into two groups according to gender, which can be distinctly tested in facial expression [13]. Moreover the model used for gender classification can be used for other classification problems such as age estimation.

In our experiment, we use face image dataset collected under unconstrained environments Labeled Faces in the Wild (LFW) [14] and Images of Groups of People [15]. In addition, the CAS-PEAL Face Database is used to test the system performance of the face image dataset that collected inside the lab [16].

3.2. Literature Review and Related Work

Classifying humans according to gender has a 27-year history. Bruce *et al*. [17] 1987 asked 78 undergraduate students to classify face images manually according to sex. During the 1990s, literature on classifying people according to gender and age was prevalent [11], [12].

Cottrell et al. [11] collected 160 face images for 10 males and 10 females. Their work obtained 99% accuracy by applying a multi-layer neural network in an unsupervised way.

Abdi *et al*. in [12] were the first group to use the pixel pass as input to the neural network. They used Eigen decomposition as preprocessor and with a simple perceptron and radial-basis function (RBF) network classifier, thereby they achieved 90% accuracy on their collected data.

Real-time application occurred when Shakhnarovich *et al*. published their paper in 2002 [18] which used an Adaboost classifier and cascaded face detector [19] to automatically detect the face. In the same year, Sun *et al*. wrote a paper entitled "Genetic feature subset selection for gender classification: A comparison study" stating that classifier performance can be improved by good feature selection through the use of a genetic algorithm (GA) that can select the relevant features for six classifiers. In [9] features were extracted from face images using principal component analysis (PCA) other than GA before entering the classifier. On the experiment of [9], four classifiers were used, namely, a Bayes classifier, an NNs classifier, an SVM classifier, and a classifier based on linear discriminant analysis (LDA) and the best performance was achieved with SVM.

In 2004, a subset of the benchmark FERET dataset was used to address the gender classifier in [20], where 500 frontal facial images were cropped and normalized, and LDA was applied as a classifier after extracting features by independent component analysis (ICA).

In 2011, Bing Li *et al*. designed their system by collecting features not only from face images but also from hair and clothing [10]. They trained seven SVM classifiers for five parts of the face: the two eyes, nose, mouth and chin and another two classifiers was trained with features from outside the face. All these classifiers were voting to determine the gender. The authors of [10] have demonstrated that extracting extra features from hair and clothing can improve the performances of the network.

Tapia and Perez in 2013 used mutual information from histograms of local binary patterns (LBPs), intensity, and shape to select features [21]. They depended on selection processes for four mutual information measurement using two kinds of face datasets; the first was collected under controlled conditions and was called a FERET dataset, while the second was a sub-set from the LFW dataset.

3.3. The Difficulties

While the images are being taken, some redundant characteristics accompanied them make getting the relevant information is a difficult task. Under different environments, images are captured to encounter variations in the face belonging to the same person. These problems are evident when the identity of the individual is dependent solely on the facial images. These difficulties can be categorized according to the reason behind them. The following lists some of the barriers to clarity.

- Pose: One of the most difficult issues of face analysis is the variations of the face pose. The angles between the center of the frontal face and the camera can make different parts of the face apparent or hidden in the image.
- Illumination: Changing light source positions or intensities can change pixel values. Typically, the second one (strength of illumination) is easier to deal with because it will affect whole images equally. Projecting strong light changes the circumference of an object and increases the white level of the image. Changing the attitude of direct illumination, however, is more difficult because of its unequal impact on the images. When the point sources of light apply to the face, they lead to shadows in the face by itself which requires a nonlinear system to remove its effect [22]. The effect of illumination is demonstrated in Figure (3.1).
- Facial Expressions: Face analysis can become difficult due to numerous variations among the images that belong to the same person. Changing the shape of the mouth or the eyes affects the performance of some systems.

- Other types of variations: Other type of variations that can affect analysis is when faces are partly covered by sun glasses or a hat. The hair cut or hair style is another difficulty in analyzing face image. Let's consider the gender task as an example where usually males have short hair while females have long hair. However this is not always true.

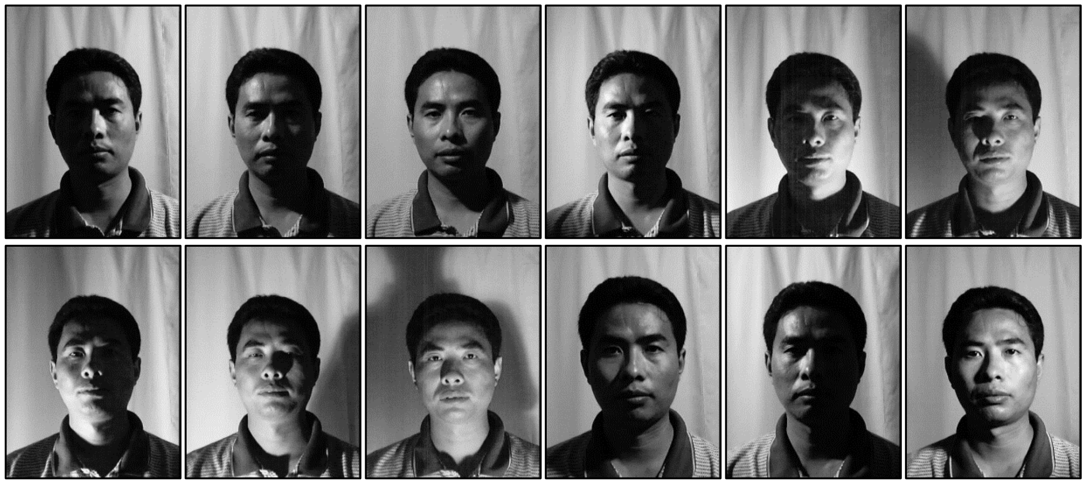
Figure 3.1: The images of one individual from the CAS-PEAL dataset [6] with different degrees of illumination.

3-2. Datasets

I attempt in this research to provide gender classification and age estimation systems that can analyze the face under unconstrained environments. Therefore, two of the largest datasets, LFW and the Image of Groups of People dataset were used along with the CAS-PEAL dataset to test system performance where all images were collected in a lab environment.

3.2.1. Labeled Faces in the Wild Dataset

LFW dataset has more than 13,000 images of faces of which 10,256 belong to 4,264 males and 2,977 images for 1,486 females. All the images are collected from the web by using a Viola-Jones face detector. All images are colored with a resolution of 250 x 250 pixels (Huang et al. 2007) [14]. Figure (3.2) shows some examples from the dataset. The LFW dataset includes images of different head poses, illuminations and facial expressions. Also it contains some images with low quality making face analysis a very difficult task.

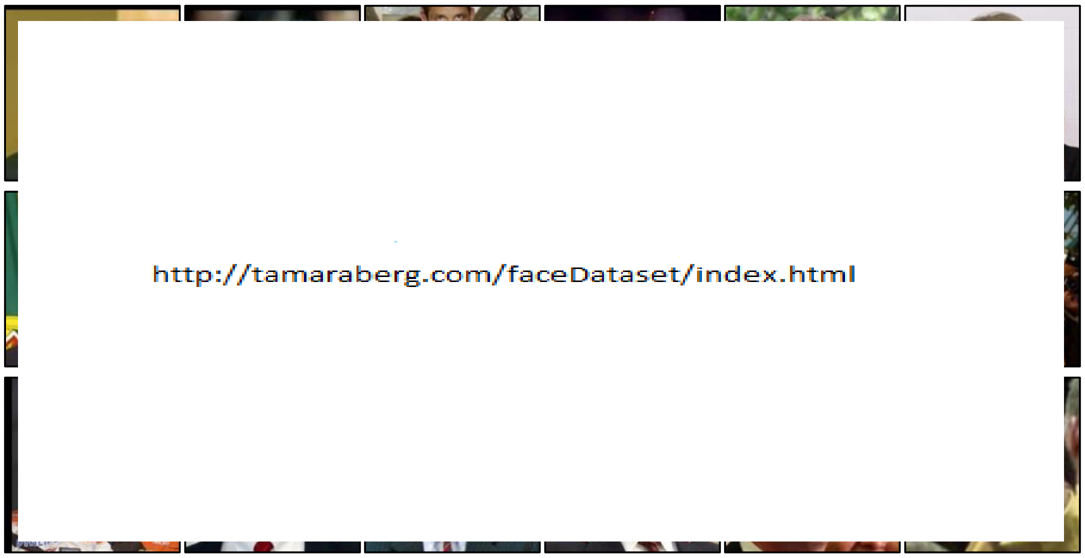

Figure 3.2: LFW dataset image examples, all the images have high resolution (250 x 250) pixels.

Another challenge in this dataset is some of the images can have pattern recognition failure that can be interpreted as either male or female, which makes gender classification difficult. Moreover, some images have been labeled as male although female features are more obvious and vice versa. Figure (3.3) shows this kind of mixing.

In Figure (3.3), the first row of images is labeled as males while the second row of images is labeled as females. Even the same image with a different center is labeled once as male and in another image as female, e.g., the third image in the first and second rows.

Figure 3.3: Samples from LFW dataset, the first row images were labeled as male even though the females have good appearance, and the second row images were labeled as female, even though the males have a good appearance.

Although some images may have more than one individual, the main person that the image labeled with his or her names is always in the middle of the picture. This is a good property in this dataset that can be exploited to select part of the image that has more relevant features. To perform that we simply filter out the edges of the image if it has more than one face according to the result of face detector. Removing redundant or irrelevant features by cropping the centers of an image leads to obtain better results. By doing that, it become easy for the face detector to detect one face only since the other face(s) are distorted. Figure (3.4) shows some filtered images.

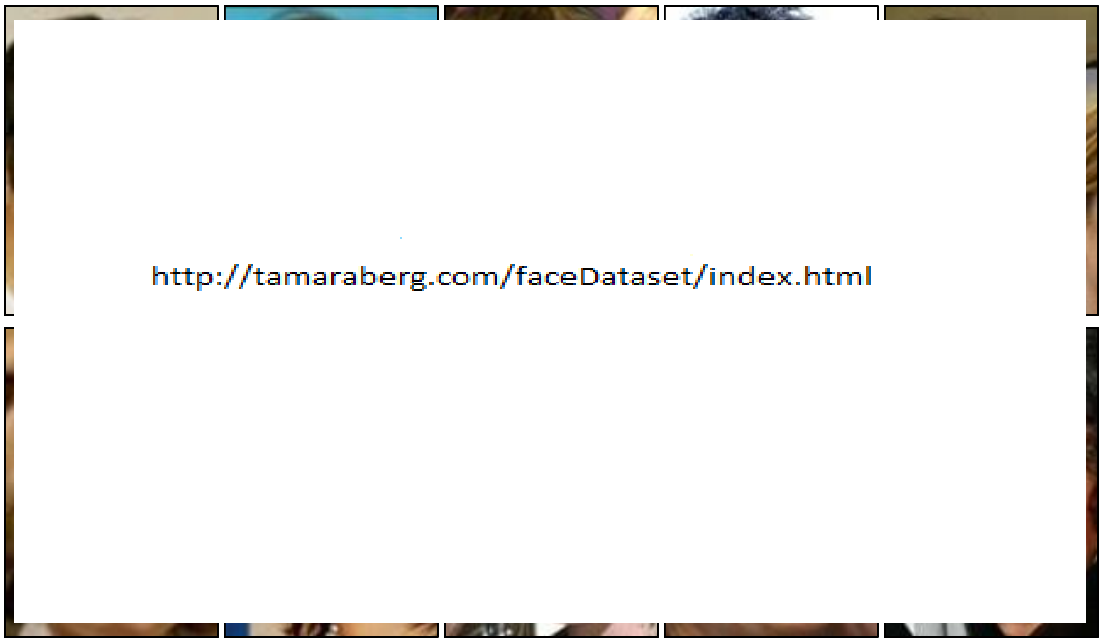
http://tamaraberg.com/faceDataset/index.html

Figure 3.4: Samples from LFW dataset: The first row images were labeled as male, and the second row images were labeled as female after removing the image edges.

3.2.2. Images of Groups of People Dataset

The image of groups of people dataset has been collected by Gallagher and Chen [15]. Their dataset has 5080 individuals whose images are divided into three groups: wedding portraits, group portraits and family portraits. Every image has more than one face labeled according to gender and age based on a total of 28,231 faces as seen in Table (3.1). This dataset is composed of images with different head poses, Illuminations, and face expressions. Some of the individuals wore a hat or glasses. Figure (3.5) shows two images from this dataset.

Due to the high number of faces per image, some faces have low resolution, and about 25% of the faces have less than 13 pixels between the eye centers [3]. Another problem with this dataset is that it has a high percentage of people who are less than 13 years old. Usually it is hard to classify children due to the high similarity between the genders.

Table 3.1: The distribution of the ages and genders of the 28231 face images that present group of people images dataset.

age	0-2	3-7	8-12	13-19	20-36	37-65	66+
Male	439	771	378	956	7767	3604	644
Female	515	824	494	736	7281	3213	609
Total	954	1595	872	1692	15048	6817	1253

http://chenlab.ece.cornell.edu/people/Andy/ImagesOfGroups.html

Figure 3.5: Samples from image of groups of people dataset from family portrait images.

3.2.3. CAS-PEAL-RL Dataset[1]

The CAS-PEAL Large-Scale Chinese Face Database contains 99,594 images for 1040 individuals with different poses, illuminations, and face expressions. All the images were taken in the lab by using nine cameras placed in an arc [6]. Figures (3.6 & 3.7) show different facial expressions and accessories for one individual from that dataset, respectively. The CAS-PEAL-RL is part of the CAS-PEAL dataset that has been made available for researchers. The subset contains 30,863 images and presents all the individuals in the dataset.

The CAS-PEAL dataset has a good percentage of males to females (around 43 percent of the total participants are females), which is one reason behind choosing this dataset in this research.

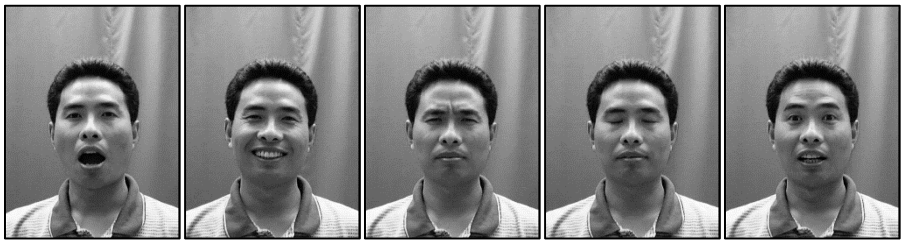

Figure 3.6: The CAS-PEAL dataset with different facial expressions for one individual.

[1] The research in this thesis use the CAS-PEAL-R1 face database collected under the sponsor of the Chinese National Hi-Tech Program and ISVISION Tech. Co. Ltd."

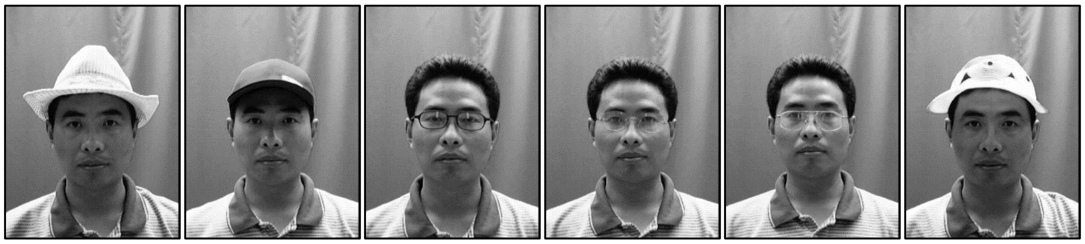

Figure 3.7: The CAS-PEAL dataset with different accessories for one individual.

3.4. Experience and Results

In this thesis, two different sizes of CNNs are applied. The small one has two layers convolutional each followed by pooling layers, and the network has one fully connected layer as shown in Figure (3.8). The big one has three convolutional layers with three pooling layers and also it has one set of fully connected layers. All images are resized before they are forwarded to the input layer. The size of the inputs is (28 x 28 pixels) for the small network and (60 x 60 pixels) for the other. The first pooling layer in both networks is max pooling while the remaining layer(s) are average pooling layers. Tables (3.2 & 3.3) show both sizes of the networks.

Table 3.2: Small size of considered CNN.

Layers	Input	Conv.	pooling	Conv.	Pooling	Fullyco.
Layer size	28x28	24x24	12x12	8x8	4x4	192
Feature mapping	1	6	6	12	12	1

Table 3.3: Large size of considered CNN.

Layers	Input	Conv.	pooling	Conv.	pooling	Conv.	pooling	Fulco.
Layer size	60x60	56x56	28x28	24x24	12x12	8x8	4x4	192
Feature mapping	1	6	6	8	8	16	16	1

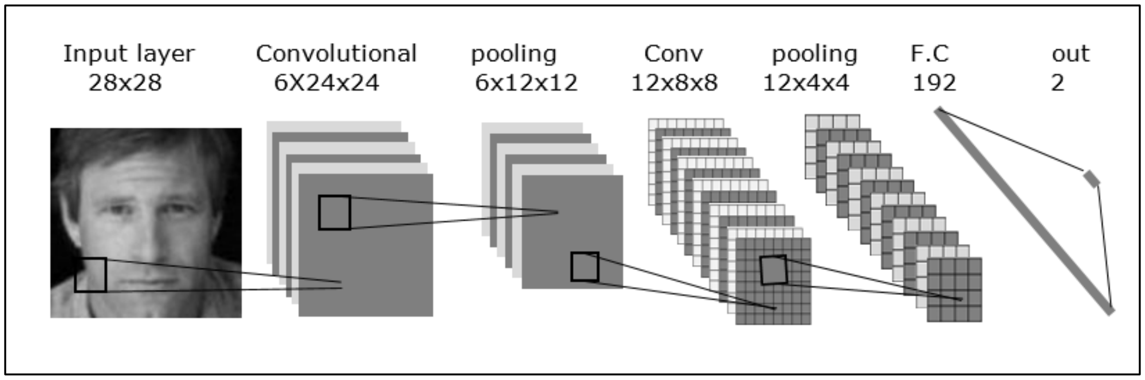

Figure 3.8: Convolutional Neural Network architecture for gender classification with two input size 28x28.

The first experiment is conducted with the LFW dataset where all the images were resized to (28 x 28 and 60 x 60 pixels) to fit our inputs to the small and large network, respectively. Figure (3.9) shows samples from LFW dataset after resizing and transferring them to gray scale. The first line from Figure (3.9) shows the input for our small size (28 x 28) and the second line shows our input for the large network (60 x 60).

Figure 3.9: Samples of LFW dataset after resizing the images. The first line is resized to (28 x 28) pixels to fit our small network input and the second line is resized to (60 x 60) pixels to fit our large network input.

The LFW dataset is not a benchmark dataset for gender classification, so cross-validation is used to evaluate our results. The dataset is split into five folders where the networks are trained each time with 80% of the data and tested with the remaining characters. Thus, for every network setup, the model has already been run five times.

Table 3.4: LFW dataset results for gender classification. LFW1 presents the small network performance. LFW2 presents the big network performance.

LFW	LFW[2]	LFW[17]	Net1	Net2
Results	95.6±0.4	93.60±0.1	95.2±0.3	**95.5±0.4**

Table (3.4) shows our results for the two considered networks compared with two other results from literature. The second column shows the best results from [21]. In [21], the authors manually choose 7,443 face images from the LFW dataset and label them with the ground truth. Even though the number of images in [21] presents part of the data set probably the easy one, we still are able to compare the results. The third column represents the best results from [23] in which the whole dataset is chosen. Net1 and Net2 are our results from the small and large networks, respectively. Figures 3.10A & 3.10B show the performance of our considered networks. As shown, the large network has a slightly better performance in the test validity due to its larger numbers of parameters. However, those numbers represent 17 epochs of training time used to figure out the right parameters compared with only seven epochs in the small network.

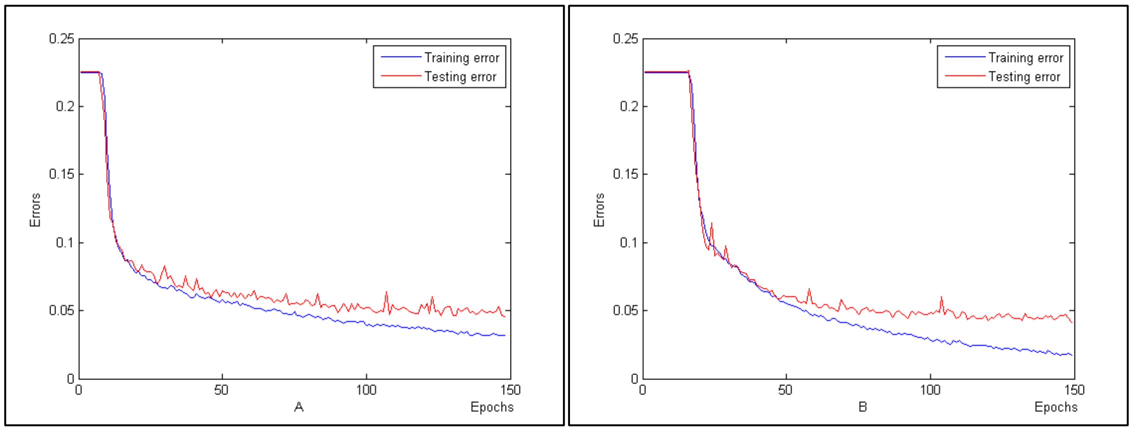

Figure 3.10: LFW dataset performance for both training and testing validity. (A) is the result for the small network, and (B) is the result for the large network.

The Images of Groups of People dataset is not divided into training part and test part; therefore cross-validation setup is used by dividing the data to five folders. This dataset provides the eye coordinate points so it is used to crop an individual face by using the formats in Figure (3.11).

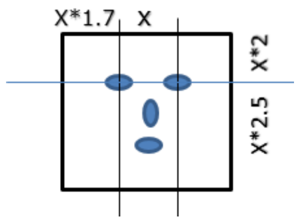

Figure 3.11: Face crop frame. Where x presents the distance between centers of the two eyes.

All the images are cropped, resized to (28x28 & 60x60) pixels and normalized between [0, 1] before they enter the networks. As mentioned before, some individual faces have less than 12 pixels between the eyes, hence we enlarge some images to get the right sizes of setup. This step from preprocessing the images results in bad quality images especially when we needed to resize the images to (60x60) pixels.

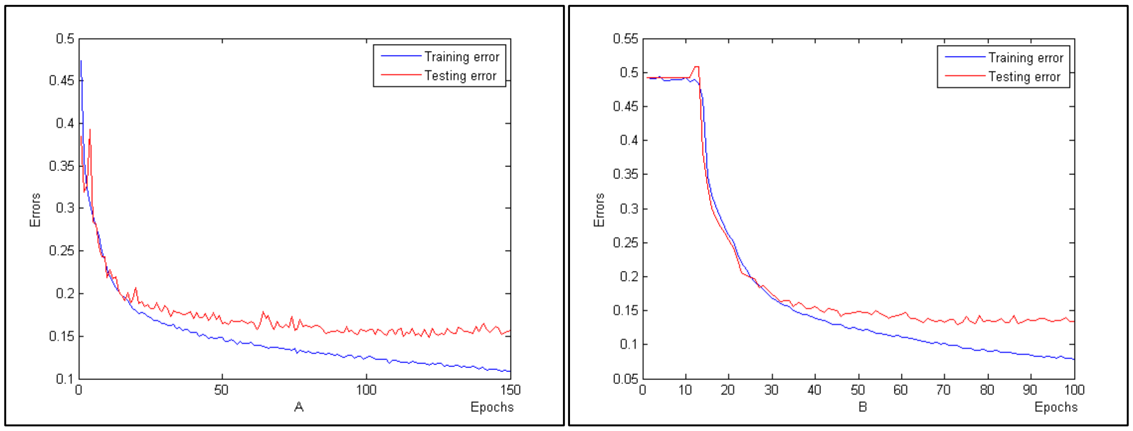

Figure 3.12: The Images of Groups of People data set performance for both training and testing validity. (A) represents the small network performance and (B) represents the large network performance.

The performance of both networks is shown in Figure (3.12). The errors have dropped faster in network1 (the smaller network that presented in Figure 3.8) than network2 which is the larger one. As we can see in Table (3.5) the best accuracy in the test validity is 85.3% for the first setup, and the error is reduced by 7% when we enlarge the network to reach 87.1% at test time. If we compare the improvement in the results when more parameters are used as in the LFW case and The Images of Groups of People dataset case, we see better improvement in the second dataset because of an increase in data samples. In other words, the high number of parameters requires a high number of data samples to prevent the networks from overfiting.

Table 3.5: The Images of Groups of People (IGP) dataset results for gender classification. The second and third columns represent the best results from references [24, 23]. The fourth and fifth columns represent our small and large network, respectively.

IGP	[24]	[25]	Net1	Net2
Results	77.89±0.3	87.24	85.1±0.2	**87±0.1**

Table (3.5) shows our results in The Images of Groups of People data set compared with two recent papers [24, 25]. The third column represents the best result from [25] in which approximately half of the data is removed due to low resolution in order to increase system performance.

The third experiment is with the CAS-PEAL-RL dataset. We just resize the images to fit our input sizes. Figure (3.13) shows our results for this dataset under the two different sized networks. As noted in the previous two datasets, the results of the larger network are better, but the larger network is slower than the smaller network.

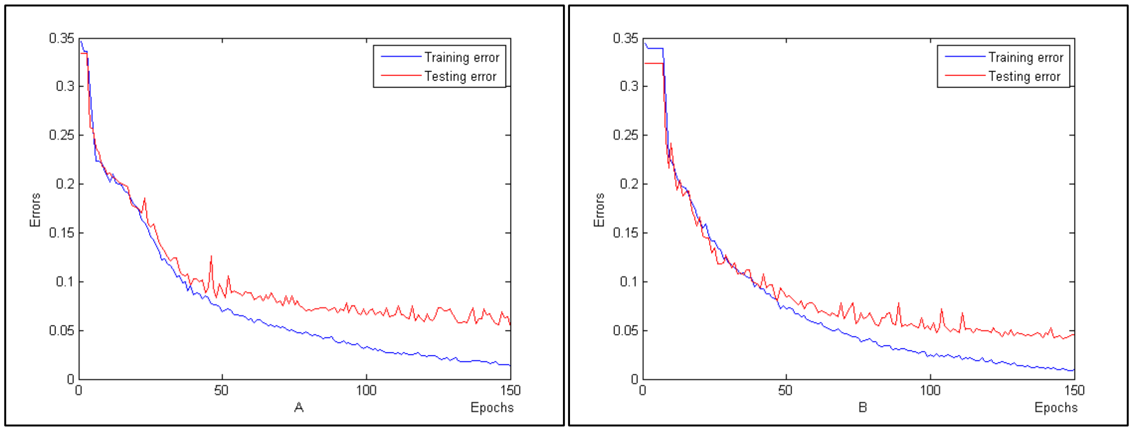

Figure 3.13: CAS-PEAL-RL dataset performance for both training and testing validity, (A): Presents the performance of the small network. (B): Presents the performance of the large network.

Table 3.6: The best accuracy obtained from the three datasets. All results were obtained from the large network

Datasets	LFW	Groups of people	CAS-PEAL-RL
Results	95.5±0.4	87±0.1	96.1±0.1

Table (3.6) represents the best accuracy obtained in the three datasets. All of these results were obtained from the large network with input size of 60x60 pixels and seven layers; hence, a larger number of parameters yielded better results. It was, however, time consuming and required a large number of data samples to prevent the networks from overfiting their parameters.

Table 3.7: Results for three datasets when we do training with one dataset and testing with the other two dataset using our small network. The first column presents training datasets and the first row presents the testing datasets.

Training	Testing	LFW	Group of people	CAS-PEAL-RL
LFW			64%	70%
Group of people		77.4%		76%
CAS-PEAL-RL		77.6%	67%	

Table (3.6) represents the results for our smaller network when whole datasets are used. The first column presents training datasets and the first row presents the testing datasets. In the second row, we use an LFW dataset to train our system and test with the other two datasets (The Images of Groups of People dataset and CAS-PEAL-RL dataset). As can be seen the performance of the network drops due to the noise in the datasets. With a backpropagation algorithm, the networks try to learn the functions that spread the data samples. Because every dataset has its own characteristics, the network will overfit its parameters according to the dataset. Thus, when we use one dataset to train the CNN and another to test it, the CNN performed poorly in test validity.

3.5. Age Estimation

For age estimation we use the image of groups of people dataset. The individuals in this dataset are divided into seven groups according to their age (0-2, 3-7, 8-12 , 13-19 , 20-36 ,37-65, 66+) as indicated in Table (3.1). Table (3.1) clearly shows that the group aged 20-36 has more than half of the data samples, which naturally has a negative effect on the training process. Therefore, we randomly choose 1000 data samples only from this group age and 1000 data samples from group 37-65 for the same reason.

The same data preprocessing and network setups for gender classification are used for age estimation. The only difference between the CNNs in gender classification and age estimation is the number of outputs, where we use only two outputs (male and female), while the age outputs depend on the number of groups that the dataset divided to. In our case, we set the output of our networks to seven.

For the smaller network, we obtain around 65% average accuracy in the test validity and around 69% average accuracy for the larger network. There is a huge overlapping between the group ages due to the similarity between them. For instance, the first and second groups in Table (3.1) have high similarity between each other especially in individual ages between 2 and 3. This similarity lead to overlapping between the data samples which negatively affects network performance.

4. Chapter 4

Conclusions and Future Works

CNNs are a powerful technique that we experiment with in this research. State-of-the-art methodology is used in the gender classification for facial image datasets collected under unconstrained environments. The first dataset is LFW, and all the face images in the LFW dataset are collected from the web. The average performance in this dataset is 95.5% when using our large network. The second dataset is Group of People images dataset. On the Group of People images dataset, the average accuracy obtained from cross validation is 87%.

The performance of CNNs depends on multi setup. In this research, our aim is to find the answer for concerns such as: how to set the right number of layers and how to choose between max pooling and average pooling. The larger network performed better if it has enough data samples to prevent the network parameters from overfiting. The larger network system give better results when compares to the other two datasets. For choosing the right subsampling operator, we find that the homogenous networks, which have both average and max pooling, always give better results.

Several future possibilities are identified for getting better performance with CNNs. It is noted that pooling layers can be changed in such a way to obtain benefits from max pooling without pushing the networks to overfit.

Bibliography

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haner, "Gradient-based learning applied to document recognition," *Proc. IEEE,* vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[2] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and retrieval in the brain," In *Neurocomputing: foundations of research*, 1st ed. MIT Press Cambridge, MA 1988 pp. 89-114.

[3] M. Minsky and S. Papert, *Perceptrons*. MIT Press, Cambridge, MA, 1969.

[4] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representations by error propagation," In *Parallel distributed processing*, 1st ed., vol. 1, MIT Press Cambridge, MA,1986, pp. 318–362.

[5] Y. LeCun, L. Bottou, B. Orr, and K. Robert, "Tricks of the Trade, Efficient Backpropagation.," in *Neural Networks*, Neural Networks: 10.1007/3-540-49430-8_2. Springer, 1998 , pp. 9-50.

[6] V. Nair and G. Hinton, "Rectified linear units improve restricted Boltzmann machines," In *International Conference on Machine Learning*, Isreal, 2010.

[7] N. Schraudolph, F. Cummins and G. Orr, "Neural Networks," Internet: http://www.willamette.edu/~gorr/classes/cs449/intro.html. Full 1999 [Apr. 10, 2015]

[8] M. Zeiler and R. Fergus, "Stochastic Pooling for Regularization of Deep Convolutional Neural Networks," arXiv:1301.3557 [cs.LG].Jan. 2013.

[9] Z. Sun, G. Bebis, X. Yuan, and S. Louis, "Genetic feature subset selection for gender classification: A comparison study," *Proc. IEEE Workshop on Applications of Computer Vision,* Dec. 2002, pp. 165–170.

[10] X. Lian and B. Liang Lu, "Gender classification by combining clothing, hair and facial component classifiers," *Neurocomputing,* vol. 76, pp. 18-27, Jan. 2011.

[11] G. Cottrell and J. Metcalfe, "Face, emotion, and gender recognition using holons," In *Advances in Neural Information Processing Systems,* PP. 564–571, 1990.

[12] H. Abdi, D. Valentin, B. Edelman, and A. Toole, "More about the difference between men and women: Evidence from linear neural

network and principal component approach," In *Neural Computer,* Vol. 24, PP. 1160–1164, 1995.

[13] Y. Saatci and C. Town. "Cascaded classification of gender and facial expression using active appearance models," In *Proc. Int. Conf. Autom. Face Gesture Recog.,* 2006, pp. 393–400.

[14] G. Huang, M. Ramesh, T. Berg, and E. Learned-miller, "Labeled fFaces in the Wild: A database for studying face recognition in unconstrained environments university of Massachusetts," *Amherst, MA, USA,,* pp. 7–49, Oct. 2007.

[15] A. Gallagher and T. Chen, "Understanding groups of images of people" In *Computer Vision and Pattern Recognition*, 2009, PP. 256-263.

[16] W. Gao, B. Cao, S. Shan, X. Chen, D. Zhou, X. Zhang, and D. Zhao, "The CAS-PEAL Large-Scale Chinese Face Database and Baseline Evaluations," In *IEEE Trans. on System Man and Cybernetics,* vol.38, no.1,, pp. 149-161, 2008.

[17] V. Bruce, H. Ellis, F. Gibling, and A. Young, "Parallel processing of the sex and familiarity of faces," In *Canadian Journal of Psychology,* vol. 41, pp. 510-520, Dec. 1987.

[18] G. Shakhnarovich, P. Viola, and B. Moghaddam, "A unified learning framework for real time face detection and classification,"

Proc. Internat. Conf. on Automatic Face and Gesture Recognition IEEE, Feb. 2002, pp. 14–21.

[19] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," In *IEEE Computer Society Conferance on Computer Vision and Pattern Recognition,* 2001, pp. 511–518.

[20] A. Jain and J. Huang, "Integrating independent components and linear discriminant analysis for gender classification," In *Automatic Face and Gesture Recognition,* May 2004, pp. 159–163,.

[21] J. Tapia and C. Perez, "Gender classification based on fusion of different spatial scale features selected by mutual information from histogram of LBP, intensity, and shape," In *IEEE Trans, Inform Forensics And Security, vol.3, no. 8,* PP. 488-499, Mar. 2013.

[22] S. Duffner, "Face Image Analysis with Convolutional Neural Networks," Ph.D thesis, Albert-Ludwigs University, Freiburg Breisgau, 2007.

[23] C. Perez, J. Tapia, P. Estevez, and C. Held, "Gender classification from face images using mutual information and feature fusion," In *International Journal of Optomechatronics,* 2012*,* pp. 92–119.

[24] J. Bekios-Calfa, J. Buenaposada, and L. Baumela, "Robust gender recognition by exploiting facial attributes dependencies," In *Pattern Recognition Letters,* pp. 228–234, 2014.

[25] P. Dago-Casas, D. Gonzalez-Jimenez, and L. Long Yu, "Single- and cross- database benchmarks for gender classification under unconstrained settings," In *IEEE Computer Vision Workshops,* Nov. 2013, PP. 2152 – 2159.