Recognizing Semantic Features in Faces using Deep Learning

Author: Amogh Gudi

Supervisors:
prof. dr. Max Welling
dr. H. Emrah Tasli

A thesis submitted in fulfilment of the requirements for the degree of Master of Science in Artificial Intelligence in the Informatics Institute Graduate Schools of Science

September 2014
[John Connor starts to cry]

THE TERMINATOR. What’s wrong with your eyes?

The human face constantly conveys information, both consciously and subconsciously. However, as basic as it is for humans to visually interpret this information, it is quite a big challenge for machines. Conventional semantic facial feature recognition and analysis techniques are already in use and are based on physiological heuristics, but they suffer from lack of robustness and high computation time. This thesis aims to explore ways for machines to learn to interpret semantic information available in faces in an automated manner without requiring manual design of feature detectors, using the approach of Deep Learning. This thesis provides a study of the effects of various factors and hyper-parameters of deep neural networks in the process of determining an optimal network configuration for the task of semantic facial feature recognition. This thesis explores the effectiveness of the system to recognize the various semantic features (like emotions, age, gender, ethnicity etc.) present in faces. Furthermore, the relation between the effect of high-level concepts on low level features is explored through an analysis of the similarities in low-level descriptors of different semantic features. This thesis also demonstrates a novel idea of using a deep network to generate 3-D Active Appearance Models of faces from real-world 2-D images.
Acknowledgements

I would like to thank Max Welling for the supervision he provided for my thesis. The bi-weekly meetings with him and his research group were very helpful in building this thesis.

I thank VicarVision for providing me the opportunity and resources to carry out this work, along with all the free lunches. I especially thank Emrah Tasli, Marten den Uyl and all the colleagues for their expertise and guidance.

Thanks to Nikolaas Steenbergen and for the most useful everyday discussions about deep learning while sweating it out in the gym.

Finally, special thanks to my family and all my friends for providing me with encouragement and motivation throughout this thesis.
# Contents

**Abstract** iii  

**Acknowledgements** iv  

**Contents** v  

**List of Figures** vii  

**List of Tables** ix  

1 **Introduction** 1  
1.1 Semantic Features in Faces ........................................... 2  
1.2 Application of Semantic Facial Feature Recognition .............. 2  
1.3 Deep Learning .......................................................... 3  
1.4 Thesis Objective and Research Questions .......................... 4  

2 **Related Work** 7  
2.1 Conventional Facial Feature Extraction ............................... 7  
2.2 Deep Learning for Computer Vision .................................. 8  
2.3 Deep Learning for Semantic Facial Feature Recognition ......... 9  

3 **Deep Learning Preliminaries** 11  
3.1 Basics of Neural Network ............................................. 11  
3.1.1 The Artificial Neuron .............................................. 11  
3.2 Deep Neural Networks .................................................. 12  
3.3 Deep Network Architectures ......................................... 14  
3.4 Convolutional Neural Networks ....................................... 16  
3.4.1 Max-Pooling .......................................................... 17  
3.4.2 Local Contrast Normalization .................................... 18  

4 **The Experimental Set-up** 19  
4.1 The Task ................................................................. 19  
4.2 The Datasets ............................................................ 19  
4.2.1 FER-C-2013 Dataset ............................................... 20  
4.2.2 Internal VV Dataset .............................................. 22  
4.3 Pre-processing Steps .................................................. 22  
4.4 Training Method ....................................................... 24
5 Experiments and Results 27
  5.1 Experiments on the FERC-2013 Dataset . . . . . . . . . . . . . . . . . . . 27
    5.1.1 Baseline Classifiers . . . . . . . . . . . . . . . . . . . . . . . . . . 27
    5.1.2 Best Performing Deep Network . . . . . . . . . . . . . . . . . . . . 28
    5.1.3 Experiments with Network Size . . . . . . . . . . . . . . . . . . . . 30
    5.1.4 Experiments with LCN and Pooling . . . . . . . . . . . . . . . . . . 32
    5.1.5 Experiments with Dropout . . . . . . . . . . . . . . . . . . . . . . 34
  5.2 Experiments on the VV Dataset . . . . . . . . . . . . . . . . . . . . . . 35
    5.2.1 Emotion Classification . . . . . . . . . . . . . . . . . . . . . . . . . 35
      5.2.1.1 Experiments with Input Image Resolution . . . . . . . . . 36
      5.2.1.2 Experiments with Pre-processing . . . . . . . . . . . . . . . 37
    5.2.2 Age Classification . . . . . . . . . . . . . . . . . . . . . . . . . . 38
    5.2.3 Gender Classification . . . . . . . . . . . . . . . . . . . . . . . . . 40
    5.2.4 Ethnicity Classification . . . . . . . . . . . . . . . . . . . . . . . . 42
    5.2.5 Detection of Glasses and Facial Hair . . . . . . . . . . . . . . . . . 43
    5.2.6 Active Appearance Modelling of Faces using Deep Learning . . . 44
    5.2.7 Relation between High-level Concepts and Low-level Descriptors . 47
      5.2.7.1 Similarity in First-Layer Weights . . . . . . . . . . . . . . . 47
      5.2.7.2 Joint Classification Experiment . . . . . . . . . . . . . . . 48
  6 Conclusion 51

A Backpropogation in Convolutional Layers 53
  A.1 Stochastic Gradient Descent . . . . . . . . . . . . . . . . . . . . . . . . . 54

Bibliography 55
# List of Figures

2.1 Typical conventional facial feature extraction pipeline ........................................ 7

3.1 The artificial neuron and its activation functions .................................................. 13
3.2 Shallow and deep artificial neural networks ............................................................ 13
3.3 Various types of neural networks and their properties ............................................ 14
3.4 Stacked autoencoders and recurrent neural networks ............................................... 15
3.5 A convolutional layer .............................................................................................. 16
3.6 Convolutional and max-pooling layer .................................................................... 17

4.1 FERC-2013 dataset ................................................................................................. 20
4.2 VV dataset .............................................................................................................. 21
4.3 Pre-processing pipeline ......................................................................................... 22

5.1 Softmax classification on FERC test set ................................................................. 28
5.2 Deep network architecture .................................................................................... 30
5.3 Best performance on the FERC test set ................................................................. 30
5.4 Network training on the FERC dataset .................................................................. 31
5.5 Network performance in depth vs width space ...................................................... 31
5.6 Network performance in pooling vs LCN space .................................................... 32
5.7 Network performance in dropout space .................................................................. 34
5.8 Network training on the VV dataset for emotions ................................................ 35
5.9 Emotion classification on the VV dataset .............................................................. 36
5.10 Experiments with Input image sizes ..................................................................... 37
5.11 Network training on the VV dataset for age ........................................................ 38
5.12 Age classification on the VV dataset .................................................................... 39
5.13 Age estimation: humans vs machines ................................................................... 39
5.14 Network training on the VV dataset for gender ................................................... 40
5.15 Gender classification on the VV dataset .............................................................. 41
5.16 Gender misclassification examples ...................................................................... 41
5.17 Network training on the VV dataset for ethnicity ............................................... 42
5.18 Ethnicity classification on the VV dataset ............................................................ 43
5.19 Network performance for glasses and facial hair detection .................................. 44
5.20 AAM based synthetic faces ................................................................................ 45
5.21 Face models generated by the deep network ....................................................... 46
5.22 Network training on the VV dataset for AAM generation ..................................... 46
5.23 Similarity in learned weights ............................................................................... 49
5.24 Network training on the VV Dataset for joint classification ............................... 49
List of Tables

5.1 AAM test set error rates ................................. 47
5.2 Joint classification results ................................. 50
Chapter 1

Introduction

A picture speaks a thousand words [1], but how many words does the picture of a face speak? As humans, we make a number of conscious and subconscious evaluations of a person just by looking at their faces, some of which are: we identify the person, which can have a defining influence on our conversation with them based on past experiences; we form an estimate of the person’s age, ethnicity, gender, etc., which makes us sensitive to their culture and habits, and based on which we also form opinions (which are often highly prejudiced and wrong); we analyse their facial expressions to gauge their emotional state (e.g. happy, sad), and to identify non-verbal communication messages that they intent to convey (e.g. love, threat). As humans, we use all of this information when interacting with each other. In fact, one of the first things we, as human beings, learn is to recognize faces and interpret some basic emotions from them. It has been argued that neonates, only 36 hours old, are able to interpret some very basic emotions from faces and form preferences [2]. In older humans, this ability is highly developed and forms one of the most important skills for social and professional interactions. Indeed, it is hard to imagine expression of humour, love, appreciation, grief, enjoyment or regret without facial expressions.

On the other hand, one of the strongest ‘stereotypes’ about machines is that interaction with them is always emotion-less. In fact, the term machine-like is used for social interactions that lack emotions (of course, other attributes like being precise, etc. are also associated with machine interactions). The main reason for this is the fact that typically, machines do not use emotions to either convey information to a human user, or to interpret information conveyed by the human user.

In the following sections, an attempt is made to elaborate what, how and why information from faces can be interpreted and utilised by machines to better understand and
serve humans. This can not just include information that humans typically interpret, but also information that is beyond the direct comprehension of human abilities.

1.1 Semantic Features in Faces

When humans look at faces of people, they typically are easily able to interpret characteristics that define the identity and gender of the person, his or her emotional state, an estimate of the person’s age, and also classify the region or ethnic group that the person may attribute his or her origins to. Moreover, based on this information and other factors like the context of the interaction, humans are also able to estimate more complex characteristics about the person (although this may often be inaccurate). Apart from these, faces also convey physiological information about a person like heart-rate and respiration-rate through skin colour [3].

The following semantic features from the human face form the primary set of information that can be directly inferred (or roughly estimated) from faces (without contextual knowledge) by humans:

- Individual identity
- Expressed emotion
- Age
- Gender
- Ethnicity

In addition to these, there are certain other ‘add-on’ features that are not an inherent part of the face (they can be added or removed at will) and that may not necessarily provide any significant additional information by itself, but can still be visually noticed (although humans do tend to form opinions based on these). These are:

- Presence of glasses
- Presence of facial hair (beard, moustache)

1.2 Application of Semantic Facial Feature Recognition

By definition, facial expressions are essentially the physical changes in faces that occur in response to a person’s internal emotional states, intentions, or social communication [4]. As said before, the inability to interpret these expressions has become one of the defining characteristics of machines. However, there are numerous applications where such information can be useful, and this information is usually readily available when physical human-machine interaction is involved (the webcam can see the user’s face when he or she is using the computer).

A primary application is the psychological analysis of subjects in different situations. Often, the size of the data and the special need to analyse ‘micro-expressions’ (expressions lasting a few milliseconds) make such an automated technique quite useful. This
could have applications in medical diagnosis, forensics (lie-detection), studying effectiveness of advertisements, etc. The ability to read facial expressions (and hence, human emotions) adds a new dimension to human-machine interactions (e.g., smile-detector in commercial digital cameras, interactive advertisements, care-giving robots).

Estimation of age from faces is another task that humans do consciously and subconsciously. However, humans are often inaccurate at estimating age, and such estimations are strongly dependant on the prior experiences of each individual. Thus, an automated system which can give a reasonably accurate age classification can be highly useful in speeding up numerous processes and reducing costs. Some examples of this can include age-based area access control, preventing sale of restricted products to minors at supermarkets, etc. Automated age estimation combined with gender estimation can lead to development of highly adaptable human-machine interactions. Also, automated classification of ethnicity, age and gender can provide an efficient and fast way for conducting anthropological study of population distribution at a large scales.

1.3 Deep Learning

Deep Learning refers to the use of Deep Neural Networks, an Artificial Neural Network architecture of multiple non-linear processes, for the task of recognizing patterns in data by training such networks in a supervised or unsupervised way. Particular deep architectures like convolutional neural networks are especially suited for tasks in which conventional approaches suffer from the curse of dimensionality [5].

An important tasks in machine learning is choosing the right features for a given task. For example, when two classes are not separable by a linear plane, it becomes important to determine features which can successfully discriminate between the two classes. Such features-detectors can be hand-crafted/defined by the user, or can be learnt automatically. When these are designed by the user, the solution to the task becomes highly dependant on the user’s domain expert knowledge and understand. Moreover, this can be highly labour intensive and needs to be repeated for every task. This task becomes even more challenging when dealing with high-dimensional data (like images).

The second option, automatic learning of features in data, can overcome both the above mentioned drawbacks and can well be achieved by Deep Learning. In [6], proof has been provided emphasising the advantage of multi-layered deep neural networks: for a given task, a shallow single hidden layered network would need exponentially more number of neurons as a $O(\log n)$ hidden layered network with $O(n)$ neurons per layer (for $n$-bit inputs).
Chapter 1. Introduction

The task of semantic feature recognition from faces has been a popular research topic with many applications already in commercial use. Almost all of these methods follow a face-modelling based approach in solving the problem. Considering the recent success of deep learning techniques [7–10], it is interesting to evaluate the performance of a deep learning based solution to this problem.

The main advantages of a deep learning based approach over a face-modelling based approach are:

- The features are designated by the deep neural network itself, and will not require hand-designing of features/landmarks like in a face-modelling based approach.
- Apart from self-evident features like lip curve, eyebrow position, etc., new and more representative features, that may not be present in a typical face-modelling based approach, can be identified by the deep network.
- Deep networks are inherently parallelizable and they fit well in a parallel computational architecture. Hence, training and executing them can be much faster with the use of platforms like GPUs and multi-thread CPUs, especially when compared to the iterative process of a model based approach.
- The working of a deep learning based solution is intuitive and principle-oriented: higher level concepts can be seen as specific groupings of multiple lower level concepts. Moreover, the human medial temporal lobe seems to have neurons that selectively activate when visual cues representing specific individuals or objects are encountered [11], which is akin to how a Deep Neural Network works [7].

1.4 Thesis Objective and Research Questions

The main objective of this thesis is to build a Deep Learning based solution to extract semantic features from images of faces. This section describes the main questions that will be researched in the course of achieving this objective.

The design of a deep neural network for any particular task involves determining multiple configurations and parameters that can ensure that the network is well suited for the task at hand. Every such combination of hyper-parameters affects the output of the system differently. Therefore, one of the questions that will be researched as part of this thesis is: How could a deep learning technique adapt to the task of semantic facial feature recognition? This question is closely followed by determining how different configurations, hyper-parameters (of the network), scale of the input, and addition of pre-processing steps affect the performance and accuracy of the system.
The task of semantic feature recognition involves the possibility of extracting numerous characteristics about the face (like age, gender, ethnicity, etc.) Considering the variety of such high-level concepts that can be visually interpreted from faces, the question that arises is: *What attributes can a deep learning based system learn to classify from faces, and how well does it perform in the determination of these attributes?*

It is known that classification of high-level concepts by a multi-layered deep neural network results in the lower layers of the network learning to recognize low-level patterns (like edges), while the higher layers combine these low-level information to determine higher-level concepts. With respect to a deep network trained to recognize different semantic features in faces, the question that can be asked is: *How are the high-level semantic descriptions related to their low-level feature descriptors? And how are the low-level descriptors of deep networks, trained to classify different semantic features in faces, related to each other?*

Finally, there are several attributes in human faces whose semantics are not easily defined. For example, the contraction of specific facial muscles, or the locations of certain landmarks on the face may not lead to easily interpretable semantic information. However, such information can be useful for certain in-depth analysis (e.g., psychological research on human subconsciousness, lie-detection, etc.). A good representation of this information can be achieved through a 3-D Active Appearance Model \[12\] of the face. This leads to the following research question: *Is a deep learning based method capable of generating 3-D Active Appearance Models of faces from a 2-D images?*

The rest of this thesis report is organized as follows: In the following Section 2 some relevant and related work in the field of semantic facial feature recognition and deep learning is discussed. Section 3 describes some core concepts about deep neural networks to lay the ground work for the subsequent chapters. Section 4 then describes the experimental set-up, including the datasets, the pre-processing steps, and the training method involved. Section 5 presents the descriptions and results of all the experiments conducted on the datasets for various tasks, and also provides a discussion of the results. Finally, Section 6 concludes the thesis report with a general summary and the mention of some potential future work.
Chapter 2

Related Work

2.1 Conventional Facial Feature Extraction

The typical conventional approach for the task of facial feature recognition essentially follows the pipeline shown in Figure 2.1. Majority of the conventional and commercial facial analysis methods rely on the Facial Action Coding System (FACS) \[13\], which involves identifying various facial muscles that can cause changes in physical facial appearance. \[12\] is one method which uses a model based approach called the Active Appearance Model \[14\] to classify emotion while building a 3-D model of the face that encodes within itself over 500 facial landmarks from which minute muscular movements in the face (Action Units, defined by the FACS) can be derived. The Active Appearance Model is generated by using PCA \[15\] directly on the pre-processed pixels, and is encoded as the deviation of a face from the average face. This model is then used to classify the emotions expressed by the face \[16\] using a single layered neural network.

![Figure 2.1: Typical conventional facial feature extraction pipeline \[17\]: The image is pre-processed making it lighting invariant; the raw pixels are then encoded at a lower level using approaches like SIFT \[18\], HOG \[19\]; The features are transformed to a more compact representation like the AAM \[14\] using methods like PCA \[15\], LDA \[20\] (although in \[12\], the low-level encoding step is skipped); Finally, the higher level representation is obtained which can be human-interpretable.](image-url)
2.2 Deep Learning for Computer Vision

Some of the primary tasks within the field of computer vision are detection, tracking and classification. With the advent of deep learning, the state-of-the-art in all three of these tasks has been improved.

A highly successful demonstration of the capability of Deep Learning for the task of image classification/detection was done by Le et al. in [7]. This work demonstrated that a detector could be learnt in a layer-wise unsupervised way (as described in [21]) without using labelled images. Moreover, their results also showed that the same detector can be sensitive to other non-target high-level categories which it encountered in the dataset (i.e. the unsupervised face detector also shows sensitivity to images of human bodies, cats and other high-level concepts). However, this is one of the few works in this area where successful results were obtained (for image-based tasks) without the use of convolutional neural networks (due to difficulty in training them unsupervised).

However, the important drawbacks of this deep auto-encoder based method is that spatial information (the location of pixels in the image plane) encoded in the image is not used efficiently (due to vectorization of the input image). Also, the weights are not shared like in a convolutional networks, and hence it has high computational resource requirements.

LeCun et al. introduces Convolutional Neural Networks in [22] as a solution to the classification task in computer vision. It shows how the training can be made much more tractable with use of simple tricks like pooling, rectification and contrast normalization. Hinton and Srivastava successfully demonstrate further improvements in training by the use of dropouts in [23, 24].

One of the most successful papers from 2012 showing the application of Deep Learning methods, specifically Deep Convolutional Networks, in image classification is [25] by Krizhevsky et al. Their work focusses on the task of image recognition on the ImageNet dataset [26]. They use a very large and deep network of 650,000 neurons with 5 convolutional and 3 fully connected layers. Such a network is managed with the use of twin 3GB GPUs for the purpose of training and testing the network within reasonable time.

This work demonstrates the applicability of such Deep Convolutional Networks training in a completely supervised way (without the use of unsupervised pre-training). This work also demonstrates the gain in classification accuracy by use of dropouts (as also demonstrated in [23, 24]) and local contrast normalization. On the same dataset in 2013, work by Sermanet et al. in [27] demonstrated an integrated solution by the use of Deep convolutional neural networks for all the three tasks of detection, localization and classification. This work was also based on using very deep convolutional networks trained
on GPUs (using the framework described in [25]), and attained the state-of-the-art in the Classification+Localization task. Baccouche et al. in [10] demonstrated the use of 3-Dimensional Convolutional Neural Networks in combination with Recurrent Neural Networks for human-action classification in videos, thus making use of spatial as well as temporal information in the frame sequences to generate state-of-the-art classification among deep learning based approaches.

2.3 Deep Learning for Semantic Facial Feature Recognition

In the context of this thesis, the sub-task in computer vision that is focussed upon is the task of facial feature recognition. There is a large body of research dedicated to this set of problems, and deep learning has emerged as a highly promising approach in solving this task.

Recent work for the facial recognition task is done by Taigman et al. in [28]. This work has shown near-human performance by deep networks in the task of recognising the identity of a person from faces. Thanks to the use of preprocessing steps like face alignment and frontalization, and the use of a very large dataset, a robust and invariant classifier is produced that sets the state-of-the-art in the Labelled Faces in the Wild dataset [29]. This work utilises a modified version of deep convolutional networks, with certain convolutional layers using unshared weights (while regular convolutional layers share weights).

In the task of emotion recognition from faces, Tang’s [8] sets the state-of-the-art on the Facial Expression Recognition Challenge-2013 (FERC-2013) dataset. This is achieved by implementing a two stage network: a convolutional networks trained in a supervised way on the first stage, and a Support Vector Machine as the second stage, which is training on the output of the first stage (although using a Softmax layer instead of the SVM stage degrades performance by only 0.1%). This classifier also performs a global contrast normalization step, which is the main inspiration for the preprocessing steps used in this thesis as explained in Section 4.3.

The preprocessing algorithms and network architectures implemented in these papers form the main motivation for the methods used in this thesis.
Chapter 3

Deep Learning Preliminaries

This section describes some core concepts that can help better understand the workings of deep learning. First, the basic structure of artificial neurons and neural networks is provided. Next, deep neural networks are introduced, and an explanation is given about its primary concepts along with various architectures in use. Finally, a more detailed description of convolutional neural networks is provided, which is extensively used in the experiments of this thesis.\footnote{Readers who are already familiar with the concepts of deep learning and convolutional neural networks may skip this chapter.}

3.1 Basics of Neural Network

A biological neural network essentially is composed of a collection of neurons as the elementary building blocks, interconnected with each other via axon terminals such that the activation of the neurons form a linear path for the signal. Similarly, an artificial neural network is composed of a system of interconnected artificial neurons, which can compute their activations based on the weights of their connections and the activations of their neighbours.

3.1.1 The Artificial Neuron

With respect to Figure 3.1a, the output of such an artificial neuron can be expressed as:

$$y = f\left(\sum_{n=0}^{N} W_n x_n + \theta\right)$$  \hspace{1cm} (3.1)

$$y = f\left(\sum_{n=0}^{N} W_n x_n + \theta\right)$$
where \( x_n \) refers to the \( n^{\text{th}} \) input to the neuron, \( W_n \) refers to the weight corresponding to this input, \( \theta \) refers to the bias/threshold parameter of the neuron, and \( N \) is the number of inputs to the neuron. Also, \( f(\bullet) \) refers to the activation or transfer function of the neuron, which is described in the next section. In a network of such neurons, each neuron transmits its activation to all the subsequent neurons it is connected to. This neuron then receives this input through one of its weighted input connections, where the activation gets multiplied by the weight associated with that connection. The resulting signal is combined together with signals received from other neighbour neurons from the previous layer by passing them through the activation function, which outputs the activation of this neuron.

**The Activation Function:** Some of the commonly used activation functions are depicted in Figure 3.1b. Activation functions are chosen based on their properties to simplify or enhance the neural network. Considering the fact that sum of linear function would also be a linear function, a multi-layered artificial neural network using linear activation functions without weight sharing would still be equivalent to using a single-layered network. Hence, the use of a non-linear activation function is needed to take advantage of the multiple layers of the network. However, such functions introduce the problem of the vanishing gradient: the gradient of the function vanishes with very high or very low inputs. Rectifier functions provide a good alternative to conventional non-linear activation functions like the Sigmoid of Hyperbolic Tangent. The primary advantages of using this activation function are [30]:

- Flow of gradient is good through the active paths in the network because there is no vanishing gradient problem unlike the Sigmoid or the Hyperbolic Tangent function (as for every input larger than 0, the gradient is set to 1).
- Sparsity is introduced in the network inherently (e.g. random initialization results in half of the neurons being activated due to positive inputs). The advantages of sparsity itself are mentioned in [30]. Some of them are: Information disentanglement (dense representations are highly entangled as one change in input affects many values in the representation); Higher Linear Separability (because of the sparse representation in a high-dimension space).
- The computational costs are lower as it is free from computation of the exponential function.

### 3.2 Deep Neural Networks

A general definition of a Deep Neural Network can be expressed as an Artificial Neural Network with more than one hidden layer between the input and output layers of the
Chapter 3. *Deep Learning Preliminaries*

### Figure 3.1: The artificial neuron and its activation functions

- **(a)** The Artificial Neuron
- **(b)** Commonly used activation function responses: Sigmoid \( \frac{1}{1+e^{-x}} \), Hyperbolic Tangent: \( \tanh(x) \) and ReLU Rectifier: \( \max(0,x) \)

### Figure 3.2: Shallow and deep artificial neural networks

- **(a)** A shallow artificial neural network with no hidden layers
- **(b)** A ‘deep’ neural network with two hidden layers

A shallow artificial neural network with no hidden layers and multi-layered networks had been in use before being associated with deep neural networks. This addition of hidden layers of neurons to the network can make it a universal approximator \[31\], i.e. given the right parameters, multi-layered neural networks have the ability to approximate any continuous function on a more compact subset due to its multi-layered feedforward architecture (irrespective of the choice of the activation function \[32\]).

Of course, the deeper the network, the higher is its ability to learn more complex functions: each layer computes a non-linear transformation of the previous layer, giving it a much higher representational power as compared to a shallower network. As mentioned before, it has been shown that an \( n \)-bit parity function that can be represented by a deep network with \( O(\log n) \) number of layers would require an exponentially higher number of neurons \( O(2^n) \) to be represented by a shallow network with no hidden layers \[6\].

However, a fundamental problem with training deeper neural network was the requirement of high computational resources, which until recently was not available. The recent
availability of powerful Graphical Processing Units (GPUs) has been an important factor in the advancement of research in this field. With GPUs, training time for such large networks has been brought down by many orders of magnitude (For a given network, our experiments have shown up to 82 times faster training on a 5000-core 6 GB GPU in comparison with a 16-core 12 GB CPU).

### 3.3 Deep Network Architectures

In this section, some of the Deep Neural Network architectures in use are explained briefly, with their primary characteristics. Figure 3.3 shows these different types of networks (and other machine learning models) categories based on their training and deepness properties.

**Stacked Autoencoders**: Autoencoders are artificial neural networks that are useful for learning effective representations from ‘raw’ data. For a given set of inputs, an autoencoder tries to learn a transformation to a compressed and distributed representation. Hence, autoencoders can be thought of as performing dimensionality reduction (or lossy compression). A single autoencoder essentially consists of an input layer corresponding to the raw input representation, and a hidden layer which forms the encoding. A Stacked Autoencoder is a neural network made up of multiple layers of autoencoders, such that the outputs of each layer is wired to the inputs of the successive layer.

In an autoencoder, the learning of the encoding is essentially done in a feedback manner where the network attempts to minimize a dissimilarity measure (the reconstruction error) between the input and a reconstruction of the input from the hidden layer (using inverse weights). This is one of the most useful advantages of stacked autoencoders: their
ability to train in a layer-by-layer manner [21], because of which they are able to learn the encoding from unsupervised data.

Deep Belief Networks: Similar to Stacked Autoencoders, these networks are essentially stacks of simpler learning modules called Restricted Boltzmann Machines [38], which is itself stacks of two layers: a visible representing the input data, and a hidden layer which learns feature that capture higher-order correlation in the data. The essential defining property/advantage of Deep Belief Networks (and RBMs) is that they are energy based probabilistic and generative models [34], and hence can fit in applications where the neural network can be combined within other probabilistic models, and where the neural network can be used to generate data based on labels. These networks can also learn in a greedy layer-wise fashion on unsupervised data (often with a final supervised fine-tuning step). However, performing direct gradient descent is intractable, and hence approximating algorithms like contrast divergence [39] are needed.

Recurrent Neural Networks: Recurrent Neural Networks (like the Long Short Term Memory [35]) are networks that are designed with the idea of taking into account the temporal information in data to generate encodings. This is achieved via the way in which data is stored inside the network over time: the activation functions of the neurons are dependant on the recent history of input data and hence acts as a short-term memory, while the interconnecting weights between the neurons can be seen as a kind of long-term memory as their values are affected by the inputs encountered by the network until then. Of course, just as with other types of artificial neural networks, these weights change slowly (depending on the learning rate), especially in comparison to the short-term memory of the activations.
3.4 Convolutional Neural Networks

Despite being universal approximators, the network architectures described above are not good at dealing with some types of problems, e.g. information presented visually. Since every neuron in a layer is connected with every neuron in the subsequent layer, the number of weights for the connections grow rapidly with the dimension of the input. This often leads to slower learning. Another important drawback of these network architectures is that they do not utilize the spatial organisation of the input (like in an image).

This calls for a network architecture that needs to exploit the spatial dimension properties of the input and reduce the number of parameters for training, and convolutional neural networks fulfils these requirements. A convolutional neural network typically consists of a set of filters (defined below) that are applied over all the locations of a spatially organised input to produce a map of output activations (with similar spatial organisation). This is depicted in Figure 3.5 and explained below.

When inputs to the network are defined in a 2-D matrix structure (like images), the convolution operation is defined as applying a ‘filter’ at multiple locations of an input matrix. A filter (also called kernel) itself is defined as a layer of connecting weights between a patch of the 2-D input and a single output neuron. Such filters are applied repeatedly over multiple locations of the input image (they are allowed to overlap), thereby producing outputs as a 2-D matrix of activations. Because of this, the network is now aware of the spatial organisation of the input data. We can formally define the operation of convolutional layers as follows:
Chapter 3. Deep Learning Preliminaries

0.3 0.6 0.7 0.9
-1.2 1.0 -1.2 -0.4
-0.7 0.8 1.4 -0.9
0.9 -1.0 -0.5 -0.1
7 9 15 16
5 10 8 11
3 12 14 2
13 1 4 6

(a) The Local contrast normalization operation with a kernel sized \(2 \times 2\) over a \(4 \times 4\) input: The boxes on the output matrix represent outputs produced by LCN kernels over input shown in the same coloured boxes.

(b) The max-pooling operation with a kernel of size \(3 \times 3\) and stride of 1 in along all dimensions of an input of size \(4 \times 4\): The boxes (of each colour) on the inputs on the left represent max-pooling kernels, and the resulting boxes on the right represent the max-pooled outputs.

Figure 3.6: Convolutional and max-pooling layers.

Assume an input arranged in a 2-D matrix of size \(I \times J\), and a convolutional layer with a kernel \(\kappa\) size of \(\hat{I} \times \hat{J}\) translating on the input with a stride of \(\tilde{I} & \tilde{J}\) along the two dimensions of the input space. This layer will produce an output of size \(\frac{I-\hat{I}}{\tilde{I}} \times \frac{J-\hat{J}}{\tilde{J}}\). The neurons of the layer computes its activations based on equation 3.1:

\[
y_{i,j} = f\left(\sum_{i=0}^{\hat{I}-1} \sum_{j=0}^{\hat{J}-1} \kappa_{i,j} \cdot x_{i+i',j+j'}\right) \forall i \in [1, \frac{I-\hat{I}}{\tilde{I}}), j \in [1, \frac{J-\hat{J}}{\tilde{J}}) \tag{3.2}
\]

where \(x_{i,j}\) and \(y_{i,j}\) refer to the input and output located at \(i\)th row and \(j\)th column in the input and output space.

Another important characteristic of convolutional layers is that the weights of the filter applied at different locations over the input are tied together. This results in a major reduction in the number of weights that need to be updated during back-propagation training. This is also depicted in Figure 3.5. The error backpropagation through convolutional layers is derived in Appendix A.

3.4.1 Max-Pooling

The output of a convolutional layer essentially denotes the presence of features defined by a filter at particular locations on the input space. Therefore, if the location of that feature in the input is translated, the activation output of the layer also translates proportionally.
The purpose of pooling is to introduce robustness to minor distortions in the input, and to give the filters a degree of translation invariance. The pooling operation typically consists of reading a 2-D patch of the input, and producing a single output that is computed via application of a function over the input patch. This operation is applied to multiple patches of the input such that the whole input matrix is covered. It is important to note that no weights are involved in a pooling operation, and it simply acts as a sub-sampling step (defined by some function).

Max-pooling is a widely used specific type of pooling, where output is equal to the maximum of its inputs. That is, the function that defines the output of a max-pooling operation over a given 2-D input patch $X$ can be written as:

$$y = \max(X)$$

This can be viewed as evaluating the presence of a feature to an area of the input space, rather than a specific location at the input space. Figure 3.6b shows a max-pooling operation with overlapping kernels.

### 3.4.2 Local Contrast Normalization

Local contrast normalization is a way to enforce competition among neighbouring neuron activations. This is done by normalizing the activations of a local patch of neurons. This normalization is achieved by a combination of subtractive and divisive normalization: remove the mean of the local patch from each value, and divide the result by the standard deviation of the values. This can be expressed as follows:

For a local input image patch of size $I \times J$, contrast normalization can be performed by

$$y_{i,j} = \frac{x_{i,j} - \frac{1}{IJ} \sum_{\hat{i} \in I, \hat{j} \in J} x_{\hat{i},\hat{j}}}{\sqrt{\sum_{\tilde{i} \in I, \tilde{j} \in J} (x_{\tilde{i},\tilde{j}} - \frac{1}{IJ} \sum_{\tilde{i} \in I, \tilde{j} \in J} x_{\tilde{i},\tilde{j}})^2}} \forall i \in I, j \in J$$

(3.4)

where $x_{i,j}$ and $y_{i,j}$ refer to the input and output located at $i$th row and $j$th column in the input and output space.

Figure 3.6a shows the application of such a local contrast normalization operation with non-overlapping kernels.

---

$^2i \in I$ refers to $i$ in range $[0, I)$, and so forth
Chapter 4

The Experimental Set-up

4.1 The Task

The task of recognising semantic features in faces is essentially an umbrella term for deciphering information encoded in faces in general, both apparent and not-so-apparent. Thus, it can be viewed as a task of extracting information from images with the prior knowledge that the images represent human faces. A list of such information encoded in images of human faces is provided in section [1] For the domain of this thesis, the task is to recognise and estimate the following attributes of the human face from images:

- Emotional expressions
- Age
- Gender
- Ethnicity
- Additional attributes like presence of glasses and facial hair
- 3-D Active Appearance Model of the face along with head pose

4.2 The Datasets

With respect to the tasks at hand, a dataset required to train and test the network essentially has the following requirements:

- Data should be in the form of images in which most of the complete face is visible.
- The faces must be mostly front-facing, i.e. the Y and Z axis rotation of faces must not be too high.
- The resolution of images must be sufficiently large: The crop of the face should not be lower than 48x48 pixels.
Chapter 4. *The Experimental Set-up*

4.2 FERC-2013 Dataset

The Facial Expression Recognition Challenge was an open-for-all challenge as part of the ICML 2013 conference workshop, which contained an emotion-annotated dataset of cropped images of faces.

The main features of this dataset are:

- 28,709 data samples/images with annotations in the train set, plus 3,589 more in the test set.
- Each image is a $48 \times 48$ grayscale images, containing a close crop of the face.

Two datasets were used for the training and testing of the network in this thesis: The emotion dataset from the Facial Expression Recognition Challenge 2013\(^1\) and the multi-annotated private dataset from VicarVision\(^2\) (hereby referred to as the VV dataset).


\(^{2}\)http://www.vicarvision.nl/
Chapter 4. The Experimental Set-up

Figure 4.2: The VV dataset

- Annotations are present in the form of class labels for one of the 7 emotion classes: Angry, Disgust, Fear, Happy, Sad, Surprise or Neutral.
- Dataset includes images of faces with small levels of occlusion and rotation (in all three axis).
- The dataset also includes faces from different age groups, gender and ethnicity (however, annotations for these features are not available).
- The primary source of this dataset is public access images on the internet.
Chapter 4. The Experimental Set-up

4.2.2 Internal VV Dataset

This is a richly annotated dataset containing annotation for various features in images of faces. The main features of this dataset are listed below:

- Images have an average size of around $256 \times 256$ pixels, in RGB and grayscale.
- The dataset includes several annotations for the images: these include expressed emotions, age, gender, ethnicity, beard amount, moustache amount, glasses presence.
- The primary sources of images in this dataset are specially conducted video-shoots in a studio, crowd-sourced web-cam images from real-world environment, as well as public access images on the internet.

The statistics of the dataset and example images can be seen in figure 4.2a.

4.3 Pre-processing Steps

The input to the network is expected to be in terms of images of faces, such that the complete face is visible in the image. However, it can be difficult for the deep network to be able to handle high variations in the pose of faces, and in lighting conditions. Thus, it becomes necessary to pre-process the input so as to make the faces more uniform.

The pre-processing step can be divided into two parts as they seek to minimize two distinct properties of the input image: the variation in location and pose of the face, and the variations in lighting conditions affecting contrast. The basic pipeline of these pre-processing steps is illustrated in Figure 4.3 and is as follows:

- **Face Location Normalization**
  1. Find faces in the image using the a face detection algorithm (specifically, the Viola/Jones face detection algorithm [40]) and extract the crop of the face such that the image is centred around the face.
  2. If images are multi-channeled, convert to grayscale images.
3. Perform in-plane rotate so as to remove tile of the faces in the X-Y plane. This is done by enforcing a zero slope on the line connecting the two eyes in faces.

4. Perform background blurring such that sharp edges caused due to rotation are smoothed out.

5. Resize the image such that the approximate scale of the face is constant. This is done by ensuring that the distance between the two eyes in the faces is constant.

- **Global Contrast Normalization**

Let us consider a dataset of images where $D$ is the number of images in the dataset, and each image in the form of a 2-D matrix of size $I \times J$. So, $x_{i,j,d}$ represents a pixel value of an image $d$ at location $i,j$ in the image.

1. Local Mean Removal: For each image, subtract the local mean of all pixel values from the image. That is,

$$x_{i,j,d}^{\text{NEW}} = x_{i,j,d} - \frac{1}{IJ} \sum_{\hat{i} \in I, \hat{j} \in J} x_{\hat{i},\hat{j},d} \quad \forall i \in I, j \in J, d \in D$$  \hspace{1cm} (4.1)

2. Image Norm Setting: Set the image norm to be equal to 100, i.e. set the square root of the sum of the squares of all pixels in an image to 100. That is, we need to set

$$\sqrt{\sum_{i \in I, j \in J} x_{i,j,d}^2} = 100 \quad \forall i \in I, j \in J, d \in D$$  \hspace{1cm} (4.2)

Thus, we must apply the following formula to the image to set the norm to 100.

$$\Rightarrow x_{i,j,d}^{\text{NEW}} = x_{i,j,d} \times \frac{100}{\sqrt{\sum_{i \in I, j \in J} x_{i,j,d}^2}} \quad \forall i \in I, j \in J, d \in D$$  \hspace{1cm} (4.3)

3. Global Mean Removal: For each pixel in each image, subtract the global mean of pixels at that location throughout the dataset (the train set).

$$x_{i,j,d}^{\text{NEW}} = x_{i,j,d} - \frac{1}{D} \sum_{d \in D} x_{i,j,d} \quad \forall i \in I, j \in J, d \in D$$  \hspace{1cm} (4.4)

4. Global Standardisation: For each pixel in each image, divide by the standard deviation of pixel values at that location throughout the dataset (the train set).

In the following equations, $i \in I$ refers to $i$ in range $[0, I)$, and so forth.
Chapter 4. The Experimental Set-up

set).

\[ x_{i,j,d}^{NEW} = \frac{x_{i,j,d}}{\sqrt{\sum_{\hat{d} \in D} (x_{i,j,\hat{d}} - \sum_{\tilde{d} \in D} x_{i,j,\tilde{d}})^2}} \quad \forall i \in I, j \in J, d \in D \tag{4.5} \]

5. Re-normalize the pixel values in each image between 0 and 255:

\[ x_{i,j,d}^{NEW} = \frac{x_{i,j,d} - \min(X_d)}{\max(X_d) - \min(X_d)} \times 255 \quad \forall i \in I, j \in J, d \in D \tag{4.6} \]

where \( X_d \) represents the \( d \)th image in the dataset, and its \( i \)th and \( j \)th pixels are represented by \( x_{i,j,d} \).

In the FERC-2013 Dataset, it needs to be noted that there are many image of faces that are corrupted in certain ways. Such images have at least one of the following properties: low image quality (where the quality of the image is too low to expect correct classification, even by humans), high face occlusion (where portions of the face is occluded enough to expect incorrect classification), degree of face rotation (where the face is primarily facing away from the camera, or the face has a very high degree of in-plane tilt). Using a Viola/Jones based face detection algorithm (as explained in the face location normalization step above), we are able to gauge and filter out such faces. In our tests, it was found that 26% of images in the FERC-2013 dataset get filtered out because the face detector is unable to find faces in the image due to the reasons mentioned above. By performing such a filtering, we are able to ensure that the deep network’s training is not corrupted by bad data and thus simplify the task to a certain extent.

4.4 Training Method

Throughout the experiments mentioned in this thesis, training of the deep neural network is done using stochastic gradient descent with momentum in mini-batch mode (refer Appendix A.1), with batches of 100 data samples. Negative log-likelihood is used as the objective function. Learning rate for the training is initialized to 0.0025, and is linearly decreased to 0.001 over 50 epochs of training.

Training is evaluated using a validation set, which is roughly 10% of the size of the total dataset (train set + valid set + test set). Stopping criteria of the network training is based on the misclassification rate on the validation set: if the validation set misclassification rate has not decreased for the last 30 epochs, stop the network training. Such a stopping criteria is helpful in reducing over-fitting on the training data. Once the
training is complete, the network is tested on a test set which also contains about 10% of the data samples in the dataset.
Chapter 5

Experiments and Results

In this section, a description of all experiments performed is given, and results of the performance of the network on various test sets are provided. All experiments have been performed on the Nvidia GTX 760[1] with 2GB of memory. The *theano* framework [41] based *pylearn2* library [42] has been primarily used for these experiments.

5.1 Experiments on the FERC-2013 Dataset

This section describes the experiments conducted for the emotion recognition task on the FERC-2013 Dataset under different network configurations as well as training parameters. First, the performance of some baseline classifiers are described on this dataset, so as to better gauge the performance benefit of the deep network. Next, a description of the best performing network and the corresponding test results and analysis are given. The configuration and parameters for this network were obtained on the basis of empirical results which are described later. Finally, all the deep network based experiments and their results are described.

5.1.1 Baseline Classifiers

The task of emotional expression classification on the FERC-2013 dataset involves choosing one of the 7 emotion labels for each image in the test set. Before starting experiments using deep networks, some naive ‘baseline’ methods are tested for classification in order to better gauge the performance of the deep network:

Chapter 5. Experiments and Results

5.1 Experiments and Results

5.1.1 Softmax Weights

(a) Softmax weights learnt by the network on the FERC dataset.

(b) Softmax network classification performance on the test set: The total classification accuracy was 28.16%, with an average precision per class of 17.78%.

(c) Softmax network classification Receiver Operating Characteristic curve on the test set (one-vs-all classes).

Figure 5.1: Softmax classification on the FERC test set

- Random Classifier: A completely random classifier, one that randomly assigns one of the 7 classes to a given image with equal probability, produces an overall accuracy of 14.3%.
- Softmax Regression: This method involves representing the input image as a vector and passing it to the input layer of a one layered neural network with no hidden units. The output layer of this network is a softmax layer with 7 units, each representing one of the 7 emotion classes. This network is depicted in figure. This classifier produces an accuracy of 28.16%.

5.1.2 Best Performing Deep Network

The architecture of this network is as follows:
The input image in the form of 48 × 48 grayscale pixels arranged in a 2D matrix is fed to the first hidden layer of the network: A convolutional layer with a kernel size of 5 × 5 having a stride of 1 both dimensions. The number of parallel feature-maps in this layer is 64. The 44 × 44 output image produced by this layer is the passed to a max-pooling layer of kernel size 3 × 3 with a stride of 2 in each dimension. This results
in a sub-sampling factor of $1/2$, and hence the resulting image is of size $22 \times 22$. The second hidden layer is also a 64 feature-map convolutional layer with a kernel size of $5 \times 5$ (and stride 1). The output of this layer is a $18 \times 18$ pixel image, and this feeds directly into the third hidden layer of the network, which is a convolutional layer of 128 feature maps with a kernel size of $4 \times 4$ (and stride 1). Finally, the output of this layer, which is of dimension $15 \times 15$, is fed into the last hidden layer of the network, which is a fully connected linear layer with 3072 neurons. Dropout is applied to this fully connected layer, with a dropout probability of 0.2. The output of this layer is connected to the output layer, which is composed of 7 neurons, each representing one class label. Because this dataset has mutually exclusive emotional expression labels, a softmax operation is performed on the output of these 7 neurons and the class with the highest activation is chosen. All layers in the network are made up of ReLu units/neurons, described in section 3.1.1.

The network is trained using stochastic gradient descent in batch mode, as described in Section 4.4. The network converges to a validation set classification accuracy of 67.43% in 77 epochs.

The performance of this network on the test set can be viewed in Figure 5.3. As can be seen, the network is able to correctly classify 67.12% of the test samples, maintaining an average precision per class of 59.6%. It can be seen that the network has over 50% precision for all classes except fear (which is 49.1%). This result could be expected as the visual appearance of a face expressing fear is quite varied for different people, and it also resembles the expressions of surprise and sadness, which contributes to the misclassification rate. It can be noted from the one-vs-all classes Receiver Operating Characteristic (ROC) plot that disgust, happy and surprised show very good discrimination qualities. If we look at the distribution of classes in this dataset in Figure 4.1a, we see that the number of happy samples is quite high, and that of disgust is very low (compared to the average number of samples per class). Therefore, although we can expect this dataset bias to boost the precision for the happy label, it seems surprising that the network was able to learn the features of disgust quite well. This could be due to a relatively large difference in the visual appearance of disgusted faces as compared to faces expressing other emotions. Hence, the network might find it easy to learn the unique feature combinations that define disgust.

The state-of-the-art results on the complete FERC test set (including the images of faces that the Viola-Jones face detection module could not find) is 67.4% total classification accuracy (as reported by Charlie Tang in [8]). This is achieved by a network with similar architecture (including the global contrast normalization pre-processing), but with the absence of the softmax layer and the addition of a 2nd stage SVM classifier. The training
Figure 5.2: Architecture of the deep network.

Figure 5.3: Best performance on the FERC Test set: The total classification accuracy of the network was 67.12%, with the average precision per class being 59.6%.

of this two stage classifier is done on a train set added with the horizontal mirroring and in-plane rotated duplicates of all training examples. If the face detection pre-processing step is skipped in the deep network set-up described in this thesis, the network produces a classification accuracy of 60.5% on the complete FERC test set (including the faces not found by the Viola-Jones face detector).

5.1.3 Experiments with Network Size

The first parameters of the network architecture to be considered are the dimensions of the network, i.e. the number of neurons per layer (the ‘width’), and the number of layers in the network (the ‘depth’). In this experiment, the width of a convolutional network was altered by changing the number of feature maps in the network (while keeping the size of the convolutional kernel fixed). The depth of the network was altered
Chapter 5. *Experiments and Results*

### Figure 5.4: Network training on the FERC Dataset.

**A** Plot of the misclassification rate for the validation and training sets during training.

**B** Weights of the first convolutional layer feature-maps.

(A) Classification accuracies of networks with varying depth and width. The dimensions of each cell are proportional to the dimensions of the network, and low to high accuracies are represented on a red to green colour scale.

The colour of empty cells are obtained via 8-neighbour interpolation.

(B) Surface plot of classification accuracy in the network depth vs width space. The surface regions corresponding to empty cell in the heatmap are computed via interpolation.

Figure 5.5 shows a heat-map table with accuracies for different depths and widths of the network. As can be seen, the left-top corner corresponding to low depths and widths gives the lowest accuracies, while the right-bottom corner corresponding to higher depths and widths provide the highest accuracies. This suggests the very intuitive fact that larger the network, the better the performance. Closer examination of these results and the surface plot in figure 5.5 also show that the depth of the network has a higher impact by the addition or removal of a convolutional layer in the network, while keeping the fully connected layer always at the last position.
(A) Classification accuracies of networks with varying applications of local contrast normalization and max-pooling. The height and width of each cell are proportional to the number of pooling and LCN applications respectively.

(Figure 5.6: Network performance in terms of accuracy with varying applications of pooling vs local contrast normalization.)

when compared to the width of the network. However, after 3 layers, this impact seems to get smaller. Similar effects can be seen with the width of the network: with only 1 hidden layer, the change in width seems to have a high impact, but with 4 layers, this impact is much smaller.

Selecting the first layer convolutional kernel size: In the experiment above, the width of the convolutional layers was altered by changing the number of feature-maps, and not by changing the convolutional kernel size. The size of the convolutional kernel was kept fixed at $5 \times 5$. This size was selected based on the following heuristics: Because the first layer of the network should be responsible for recognizing low-level features in faces (like eye, mouth, etc.), it would require to have a convolutional kernel size that is roughly able to ‘look at’ the complete feature. By looking at the input images, we can see that a single low-level facial feature does not occupy more than 10% area of the face. Hence, a convolutional kernel size of $5 \times 5$ is well suited (as it covers 10.4% of the input image).

5.1.4 Experiments with LCN and Pooling

This experiment was conducted in order to determine the closest-to-optimal combination of Local Contrast Normalization (LCN) and max-pooling within the neural network layers. Max-pooling essentially results in a non-linear down-sampling step. The intuition behind performing max-pooling is that it provides a certain degree of translation
invariance on the input, and it reduces the computational complexity for the subsequent layers (by reducing the input). Thus, max-pooling can be considered as a good way to perform intermediate dimensionality reduction of the representation within the layers of the network.

In these experiments, a max-pooling kernel size of $3 \times 3$ is chosen, that is translated along both the dimensions of the 2-D input frame in strides of 2. This selection has been made based on previous work in [8, 25]. The produced a down-sampling factor of $1/2$.

Local contrast normalization is another well-used trick in designing deep architectures, which ensures competition among the activations of nearby neurons by normalizing them locally with respect to each other. LCN can further aid pooling, which selects the neuron with the highest activation, by ensuring a fair competition among the inputs to the max-pooling layer.

For the purpose of this experiment, a network similar to the one shown in figure 5.2, without the pooling and LCN layers, is considered as the baseline. Max-pooling and LCN are then applied at three locations within the network: at the outputs of the first, second and third convolutional layers.

The results of this experiment can be seen in Figure 5.6, which shown a heat-table and a surface plot of the classification accuracy of the network with varying configurations of max-pooling and LCN. As can be seen in the figure, max-pooling the outputs of the first convolution layer, after applying LCN to it, gives the best results (in comparison to max-pooling and LCN on the subsequent layers). It can be seen that simply applying local contrast normalization without pooling the outputs does not yield better results, in fact it degrades it. This might be explained by the fact that in the absence of pooling, normalizing the outputs locally might lead to extra emphasis on certain non-informative activations, which otherwise would have continued to have a low activation and would not have been propagated deeper in the network. It can also be observed that applying pooling to the network is more advantageous in the starting layers of the network, and continuing to pool outputs of deeper layers leads to reduction in accuracy. This could be attributed to the fact that activations of deeper layers represent more information than the activations of starting layers, and hence down-sampling these outputs lead to loss of useful information which leads to poor classification performance.
5.1.5 Experiments with Dropout

Dropout essentially means randomly omitting the neurons of a layer by a certain probability. Dropout is an important recent improvement for neural networks. It works equivalent to adding random noise to the representation (randomly setting outputs of neurons to zero), or performing model averaging, and this helps reduce overfitting \[23, 24\].

In this experiment, the neural network described in Figure 5.2 is used, and dropout is applied on its layers during the training phase. For the purpose of this experiment, it is ensured that all the convolutional layers get the same dropout probability, and its magnitude is altered to test the performance of the network trained under various dropout probabilities. The outputs of the last fully connected hidden layer are also dropped out with another varying dropout probability.

The results of this experiment are shown in Figure 5.7 in terms of a fully-connected layer dropout vs convolutional layer dropout surface plot and heat-table. It can be observed that a fully-connected layer dropout probability of 0.3 gives the best performance. It can also be noted that applying any dropout to the convolutional layer only results in reduction of performance. These results support the optimised network architecture used in \[25\] where the best performance was obtained by using dropouts only on fully connected layers. This can be attributed to the fact that fully-connected layers are more prone to overfitting, while the additional noise caused by dropouts could be adversely affecting the convolutional feature detector.
5.2 Experiments on the VV Dataset

In this section, the details and results of the experiments conducted on the VV Dataset are provided. Unlike experiments in the previous section, the experiments that follow are not focussed on network optimisation. The network used for training and testing for recognition of various features in the dataset has the same architecture as defined in Section 5.1.2 which is the most optimised network for the FERC dataset. In all the experiments that follow, the training of the network was done as described in Section 4.4.

5.2.1 Emotion Classification

The task of emotion classification on the VV Dataset is very similar to that on the FERC dataset. However, a key difference between the two datasets is that all the emotion classes are more uniformly distributed, as can be seen in Figure 4.2a. Therefore, this experiments gives us an opportunity to better evaluate our previously optimised network from Section 5.1.2.

The progression of the training can be viewed in Figure 5.8 along with the learnt first layer weights of the convolutional feature-maps. The performance of the trained network on the test set is shown in Figure 5.12.

As can be seen, the performance of the network is quite similar to the one seen for the FERC dataset. The average precision score is closer to the total classification accuracy due to the uniform class distribution in the dataset. It can be seen from the ROC curve...
Chapter 5. Experiments and Results

36

Figure 5.9: Emotion classification on the VV dataset: The total classification accuracy was 66.56%, while the average precision was 65.64%.

that that Happy and Neutral are the best-learned classification categories of the network, although all the other labels also have a decent amount of area under the curve.

5.2.1.1 Experiments with Input Image Resolution

A difference between the FERC dataset and the VV dataset for emotions is also the fact that the input image resolution is higher, 256 × 256 on average, as compared to 48 × 48 on the FERC dataset. This gives us the opportunity to evaluate the performance of the network that is trained and tested on different image sizes.

The network used for this experiment is the same as described in Figure 5.2 with the only difference being the size of input image and first layer convolution kernel size. The results of this experiment can be observed in Figure 5.10. It can be observed that the performance of the network is about the same for image sizes of 60 × 60 and 48 × 48, and reduces smoothly for smaller image sizes. This can be explained by the argument that downsizing from 60 to 48 does not result in loss of important information, while reducing the size further does. This can also be observed by looking at the multi-sized images in the figure: We can easily gauge the facial expression for the 48 × 48 image (and images of greater size), but this task becomes almost impossible for lower sized images. Another observation that can be made here is that the performance of the network drops when image size in increased to 72 × 72 pixels. This observation can be attributed to the fact that we are using a constant 5 × 5 sized convolution kernel in the first layer. For the 48 × 48 and 60 × 60 image sizes, this roughly corresponds to 10% of the input image area.


Figure 5.10: Input image sizes provided to the network for training and testing: (Left to Right) $72 \times 72$, producing a classification accuracy of 66.3%; $60 \times 60$, producing a classification accuracy of 67.3%; $48 \times 48$, producing a classification accuracy of 67.12%; $36 \times 36$, producing a classification accuracy of 65.87%; $24 \times 24$, producing a classification accuracy of 63.17%.

and can be thought of as sufficient to match the scale of facial features (like eyes, nose, mouth, etc.). However, for the larger $72 \times 72$ images, this only corresponds to about 7% of the image area, which could be too small to capture a ‘complete’ feature. Thus, it can be quite possible that the network performs much better with an appropriately sized kernel size. However, due to the limitations of the GPU memory size, it was not possible to conduct this experiment.

5.2.1.2 Experiments with Pre-processing

The global contrast normalization pre-processing step described in Section 4.3 is based on work by Tang in [8]. It essentially involves removing the local mean and setting the image norm to a constant value before proceeding to remove the global mean (in the whole dataset) and standardizing the image. This step results in removing non-informative information from the image. It is found that this pre-processing step gives a performance boost of around 3% to the classification accuracy of the network.

The face alignment pre-processing step (part of the face location normalization pipeline) was partially inspired by the face frontalization step in [28]. The idea behind this step is: if the only major variance in the dataset images is in terms of the features which need to be classified (e.g. facial expressions) and not in terms of additional noise in location and pose of the face, the deep neural network would be able to learn a better noise-free representation. Again, it was found that this step results in an increase of roughly 5% in classification accuracy. Of course, a drawback of such a pre-processing step is that it makes the network itself less robust to such location variations (the network is now heavily dependant on this pre-processing step for future test cases).
5.2.2 Age Classification

The task of age estimation from human faces essentially involves evaluating facial features that can be considered a sign of ageing (like skin wrinkles) and estimating the age of the person based on past experiences. For the deep network, the process pipeline is exactly the same.

In this experiment, the age annotations in the VV dataset are considered, which contain one of 17 exclusive age labels for each image. The age labels represent 5 year age intervals around ages that are multiples of 5 (except for the range [0-2]).

To accommodate this, the final softmax layer of our network architecture is set to have 17 neurons, one for each age class. The network is trained as usual using stochastic gradient descent in batches of 100, with a linearly shrinking learning rate in the range of 0.0025 to 0.001 over 50 epochs. The training progression of the network is depicted in Figure 5.11 along with its learned first layer weights. It can be noted that the plot of the validation set misclassification is quite unsteady. This can be attributed to the relatively smaller size of the age annotated dataset (323 random samples in the validation set), as well as the uneven distribution of age labels in the dataset (data samples for the [23-27] and [28-32] are much higher than the others).

Performance of the network on the test set can be seen in Figure 5.12: the green squares represent correct classification within \( \pm 2.5 \) year resolution, while the orange squares represent correct classification within \( \pm 5 \) year resolution. It can be seen that the distribution of age within this dataset is quite skewed towards the age range of [23-27] (refer Figure 4.2a), and the result of this can be seen in the confusion matrix: there is a bias in the network towards [23-27] age class. Also note that due to the extreme lack of data...
Chapter 5. Experiments and Results

39

Figure 5.12: Age classification on the VV dataset: The total classification accuracy was 53.12%/72.13% for ±2.5 year/±5 year resolution. The average precision for younger than 50 years age group was 33.3%/51.7%, while for the 6 samples in the greater than 50 years age group, none were classified correctly.

Figure 5.13: Plot of the accuracy of age estimation at different resolutions by the deep network and humans. The human performance was obtained from [43].

samples in the above 50 age range, the network performance is severely degraded. This is the main reason for a low average precision while having a high total classification accuracy.

The task of age estimation from faces is something that humans do inherently, both consciously and sub-consciously. It has been estimated that the estimation of age from faces by humans is accurate up to a range of ±4.2 to ±7.4 [43]. Figure 5.13 shows the performance of this deep network approach to automated age estimation for the VV dataset vs the age estimation by humans at various resolutions for the FG-NET
Chapter 5. Experiments and Results

5.2.3 Gender Classification

The primary way of classifying the gender (male and female only) of a person by humans is heavily based on the appearance of the face, and it is intuitive that humans are very good at this task. Thus, this is a relatively easy task for a machine primarily because there are only two classes to assign, and the difference in face structure between males and females is generally quite easily apparent.

The deep network was trained on the gender annotated part of the VV dataset by modifying the final softmax output layer to only contain two neurons. The networks is again trained using stochastic gradient descent with a linearly decreasing learning rate from 0.0025 to 0.001 over 50 epochs, and a momentum of 0.6. The progression of the network training is portrayed in Figure 5.14 in terms of the validation and training set misclassification rate and the first layer weights. The validation set error rate stop reducing after 29 epochs, which is relatively quicker, possibly due to the presence of only two classes and the differences between faces of the two classes being easily recognisable.

The test set performance of the network is plotted in Figure 5.15. As can be seen, the network is able to correctly classify 90.86% of the faces in the test set, and the
Figure 5.15: Gender classification on the VV dataset: The total classification accuracy was 90.68% and the average precision of the two classes was 88.9%.

Figure 5.16: Examples of data samples misclassified by the deep network: M stands for male label, and F stands for female label; the green labels represent the ground truth and the red labels represent the network classification.

ROC curve shows good discrimination characteristics. A examination of the falsely classified faces can be done by observing the examples shown in Figure 5.16. As can be seen, a large portion of the misclassified faces are those of young children. Classifying gender of very young children is also a difficult task for humans, primarily because well-defined masculine or feminine features are not yet developed for them. Another part of the misclassified images include difficult to classify faces with non-prominent or mixed gender features (see third from right and second from left in the figure). Lastly, there also exist a small portion of examples with incorrect ground truth (fourth image from left), and examples where the image does not include the complete face (right most image). Overall, a classification accuracy of above 90% and the presence of such hard-to-classify faces in the test set suggests that the network performs on a near human level on the task of gender classification.
5.2.4 Ethnicity Classification

The classification of ethnicity of a person for humans is a task that heavily depends on his or her knowledge and past experiences. The VV dataset contains annotations for five classes of ethnicity, the fifth one being ‘others’ which includes all other non-listed ethnic groups like Middle-Eastern, Latin-American, etc.

The deep network’s final softmax output layer was again altered to fit the annotation by setting the number of neurons to 5. As usual, the network was training using stochastic gradient descent for batches of 100 samples with a linearly declining learning rate of 0.0025 to 0.001 over 50 epochs. The learning progression of this network can be seen in Figure 5.17 along with its first layer weights. This network required 124 epochs to converge, which suggests that the facial features required to identify a person’s ethnicity are not quite obvious and common.

The test set performance of the network can be seen in Figure 5.18. It is also apparent from Figure 4.2a that there is a high level of uneven distribution of classes, with the Caucasian and East Asian data samples being more than 15 times more abundant than African, South Asian and Others. This leads us to expect the network to primarily only be able to distinguish between Caucasian and East Asian faces. The performance of the network can be seen in confusion matrix and the ROC curve: the network performs very well for Caucasian and East Asian faces (above 95% precision), but the average precision of all classes is only 61.52%.

Another thing to note here is that the network does not make use of the colour information present in the images (the pre-processing step converts all images to grayscale). Ethnicity is one of the facial attributes that exhibits a high variance in the colour of the
Chapter 5. Experiments and Results

43

Classification

Ground Truth

Caucasian African East Asian South Asian Others

43

0.0

0.2

0.4

0.6

0.8

1.0

False Positive Rate

0.0

0.2

0.4

0.6

0.8

1.0

True Positive Rate

Caucasian [AUC: 0.986]

African [AUC: 0.966]

East Asian [AUC: 0.990]

South Asian [AUC: 0.911]

Others [AUC: 0.886]

5.2.5 Detection of Glasses and Facial Hair

Unlike all the previous features on which experiments were conducted, glasses and facial hair are arguably not an inherent part of the face itself: they can be added or removed from the surface of the face. Moreover, it is needless to mention that this is a extremely trivial task for humans. Because glasses and facial hair are both easily observable, it can be expected that the deep network performs good on these tasks.

The network is trained using the same setup as described in the previous experiments, with the final softmax later altered to have only two neurons for the presence of glasses, and 3 neurons for the amount of moustache and amount of beard (none, light, heavy). The performance of the network on the test sets were as follows: 94.52% accuracy for the presence of glasses, and 88.1% for the amount of beard and 89.13% for the amount of moustache. The ROC curves for the classification labels are plotted in Figure 5.19. As can be seen, the network’s precision for detecting the presence of glasses and heavy moustache is very high. However, due to the slightly ambiguous nature of light beard classification, the network does not learn a very precise light beard (and no beard) classifier(s).
5.2.6 Active Appearance Modelling of Faces using Deep Learning

One of the advantages of a face-modelling based approach is its ability to build 3-D models of faces, which captures the minute physical details. These include the general shape of the face, the texture of the skin, and the location of key-landmark points on the face (as well as the pose of the head). Such a model can further be used to infer the movements of facial muscles (Action Units [13]), as well as to classify higher level semantic features in faces (that our deep network directly attempt to classify from the images). The knowledge of these landmarks can be used to sub-segment the face for various applications. Other advantages of having a 3-D face model exist in the area of virtual reality, compute generated imagery, human-computer interaction, etc.

The Active Appearance Model (AAM) of a face is a 3-D model that comprises of the locations of 512 (pre-defined) landmark points on the face (like edges of the eyes, lips, etc). It is represented by a compressed 89-valued float vector that encodes the shape and appearance (including texture) of the face. As briefly mentioned in Section 2.1, the AAM is conventionally produced by applying PCA directly on the pre-processed pixels of the face image. The shape and appearance parameters within it are encoded as the deviation of a face from the average face (the process description of active appearance modelling of faces can be found in [12]).

Apart from this, it also consists of the pose of the face: the angles made by the normals of the face with respect to the X, Y and Z axis (where the X axis is in-plane horizontal axis, Y axis is the in-plane vertical axis, and Z axis is the axis along the out-of-plane third dimension). However, as mentioned before in section 4.3 we perform in-plane rotation of the image to remove the X-Y plane tilt of the face. Thus, only the angles made with the Y and Z axis are relevant for training the network, as the angle made with...
Figure 5.20: Examples of synthetic faces used for training the network (bottom), and their source images (top). The synthetic faces are obtained using AAM, and are placed over real-world background from their source images, resulting in realistic looking synthetic faces.

the X axis is always zero (post pre-processing). Finally, this annotation is expressed to the network in terms of a vector of 91 float values (89 AAM parameters + 2 angles).

The network was trained on a set of pairs of synthetic faces and their corresponding Active Appearance Models generated by the conventional face modelling method described above. Examples of such synthetic faces is shown in Figure 5.20. The reason for using synthetic faces instead of real faces is that because these synthetic faces are generated by their corresponding AAM, the modelling error between the face and the AAM vector is zero. Hence, we can have true ground truth information for synthetic faces (for real faces, the AAM vector will be corrupted by modelling error from the conventional modelling method).

On testing the network on the test set, a cosine similarity score of 0.76788 is obtained with the ground truth when tested on real faces, and a similarity score of 0.86210 when tested on synthetic faces. Moreover, the pose estimation for the test faces produced an average error of $2.92^\circ/1.89^\circ$ in the Y/X axes for real faces, and $2.23^\circ/1.66^\circ$ in the Y/X axes for synthetic faces. These results are also illustrated in Table 5.1. Note that only the dissimilarities on the test set of synthetic faces can be considered as a true error measure because it is free from the modelling error of the conventional technique.

Some examples of Active Appearance Models generated by the network for real faces from the test set are shown in Figure 5.21 in terms of a 3-D wire-mesh rendering. It can be seen in the figure that the network has been able to detect and model the general shape of them face, the position of the lips and mouth, the position of the eyebrows, etc. The head poses of the 3-D model clearly resemble those of the real input images.
Figure 5.21: Face models generated by the deep network

Figure 5.22: Network training on the VV dataset for AAM generation: Training data includes synthetic faces - Active Appearance vector pairs.
Table 5.1: Cosine similarity scores between the Active Appearance Models generated by the deep network and the ground truth generated by conventional face modelling.

<table>
<thead>
<tr>
<th>Test Set</th>
<th>AAM Cosine Similarity</th>
<th>Pose error in Y-axis</th>
<th>Pose error in X-axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic Faces</td>
<td>0.86210</td>
<td>1.88°</td>
<td>1.66°</td>
</tr>
<tr>
<td>Real Faces</td>
<td>0.76788</td>
<td>2.92°</td>
<td>2.23°</td>
</tr>
</tbody>
</table>

From these results, it would appear that the deep network has indeed been able to learn to generate a complete 3-D face model from images of faces, while having been trained on images of synthetic faces. It must be noted in these results that only the cosine similarity on the synthetic faces test set can be considered as a true performance measure, as it is not prone to errors made by the conventional modelling method. However, by visualizing the 3-D models generated by the deep network (see Figure 5.21), it is evident that the network is able to generate fairly accurate 3-D models of images of real faces. Even subtle changes in the pose of the face have been successfully modelled by the network.

5.2.7 Relation between High-level Concepts and Low-level Descriptors

In all the experiments explained above, the deep network was required to be trained on specific annotation-image pairs for the given task. However, it can be argued that high-level features in faces (like age, emotions) are just combinations of certain low-level features in faces (like eye-edges, lip-curl). It can further be argued that many of these low-level features can be common among different high-level feature descriptions, i.e. certain combination of lip-curl and forehead wrinkles can represent a sad emotion, while another combination of forehead wrinkles and eye-edges can be representative of the age of a person.

5.2.7.1 Similarity in First-Layer Weights

To study the above mentioned argument, this section compares the low-level descriptors that combine to form different higher level concepts. A closer look at Figures 5.4b, 5.8b, 5.11b, 5.17b, 5.14b, 5.22b reveals a similarity in the general pattern in the weights of the feature-maps (even over two different dataset). In order to perform a more empirical study, a cosine similarity score can be used. The cosine similarity of these weights are computed by representing the weights of the feature-maps together in the form of a vector (ordered based on their visual appearance) and calculating the cosine of the angle between them using their dot product and magnitude (\(\cos \theta = \frac{A \cdot B}{|A||B|}\)). Its magnitude varies in the range of -1 for exactly opposite, to +1 for exactly similar.
Figure 5.23 illustrates a heat-table of cosine similarity scores between the first layer weights of networks trained for different high level feature recognition (classification task). Along with this, the similarity of these weights has also been compared with randomly generated weights. Certain patterns can be observed in these results: The similarity scores of network weights trained to recognize facial-hair and glasses seem to have weights most dissimilar from those of emotions and ethnicity. This could be explained by the fact that both emotions and ethnicity are invariant to the presence of facial-hair. On the other hand, there appears to be high similarity scores between the weights of facial-hair and gender, and of facial-hair and age: this could be related to the fact that all females and all young children have no facial hair, and hence the presence or absence of facial-hair can be a good indicator of the person’s gender and age. This could also be the reason for a high similarity score between age and gender. A low similarity also exists between each of the weights of age, ethnicity, gender, glasses and facial-hair, when compared with emotions. This can be attributed to the fact that emotions are sensitive to facial expressions, and invariant to other attributes. On the other hand, weights for joint classification (explained later) are similar to the weights of all the other features as joint classification includes the classification of all those features.

In general, it could be observed that the given task seems to have a strong effect on the lower-level descriptors. A higher correlation in the weights for similar visual tasks is also observed, while it is seen that visually dissimilar tasks exhibit a lower correlation. It must be noted that although the training images for these different tasks are very similar (they are from the same sources), they are not exactly the same images. Therefore, more precise observations can be seen if the networks are training on the same images for different tasks. Further, the statistical significance of these observations can only be tested by repeating the training multiple times for weigh comparison.

5.2.7.2 Joint Classification Experiment

In order to exploit the low-level similarity observations from the previous sub-section, a single network needs to be trained to jointly classify multiple non-exclusive facial features. In order to achieve this, the different annotations per image in the VV Dataset are combined into one single set of image – annotation pair, where the annotations are represented by 30 non-exclusive class labels in the form of an array: 7 Emotion labels, 17 Age labels, 5 Ethnicity labels, 1 Gender label, 1 Glasses label, 3 Beard labels and 3 Moustache labels.

To accommodate this, the architecture of the deep network is altered in one small way: The final softmax output layer is now replaced with a fully connected output layer
Figure 5.23: Cosine inter-similarity scores for first-layer weights learned by the network for different features. Comparisons with random weights are also shown.

Figure 5.24: Network training on the VV dataset for joint classification containing 30 linear neurons, each representing one class label from the array mentioned above. Also, the error measure used as the termination criteria is changed to the mean squared error (between the network prediction and annotation ground-truth) on the validation set. The training of this network is depicted in Figure 5.24. The network is trained and tested on a multi-annotated dataset of roughly 3000 data samples (split as 80% training set, 10% validation set and 10% testing set).

The performance of this network on the test set is provided in Table 5.2. As can be seen, the differences in the accuracies of the joint classification network and the individual networks is very small. This suggests that the deep network is capable of learning to

<table>
<thead>
<tr>
<th>Feature</th>
<th>Emotion (FERC)</th>
<th>Emotion (VV)</th>
<th>Age</th>
<th>Ethnicity</th>
<th>Gender</th>
<th>Joint</th>
<th>AAM</th>
<th>Glasses</th>
<th>Moustache</th>
<th>Beard</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emotion (VV)</td>
<td>0.83</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Age</td>
<td>0.75</td>
<td>0.86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethnicity</td>
<td>0.73</td>
<td>0.82</td>
<td>0.87</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gender</td>
<td>0.73</td>
<td>0.83</td>
<td>0.89</td>
<td>0.82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joint</td>
<td>0.78</td>
<td>0.88</td>
<td>0.93</td>
<td>0.86</td>
<td>0.90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AAM</td>
<td>0.73</td>
<td>0.80</td>
<td>0.81</td>
<td>0.76</td>
<td>0.84</td>
<td>0.86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glasses</td>
<td>0.69</td>
<td>0.77</td>
<td>0.84</td>
<td>0.76</td>
<td>0.82</td>
<td>0.85</td>
<td>0.84</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Moustache</td>
<td>0.74</td>
<td>0.84</td>
<td>0.95</td>
<td>0.85</td>
<td>0.90</td>
<td>0.94</td>
<td>0.84</td>
<td>0.85</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Beard</td>
<td>0.71</td>
<td>0.81</td>
<td>0.90</td>
<td>0.79</td>
<td>0.90</td>
<td>0.92</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>0.03</td>
<td>0.02</td>
<td>0.04</td>
<td>0.04</td>
<td>0.03</td>
<td>0.05</td>
<td>0.03</td>
<td>0.01</td>
<td>0.03</td>
<td>0.03</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.2: Classification accuracies for different features by the joint classification network as compared to individually trained networks on the test set.

classify multiple semantic features in faces in a joint manner, while producing accuracies comparable to the individual classification networks.

<table>
<thead>
<tr>
<th>Semantic Feature</th>
<th>Joint Classification Accuracy</th>
<th>Individual Network Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emotions</td>
<td>67.21%</td>
<td>68.82%</td>
</tr>
<tr>
<td>Age</td>
<td>56.68%</td>
<td>57.92%</td>
</tr>
<tr>
<td>Ethnicity</td>
<td>89.97%</td>
<td>92.24%</td>
</tr>
<tr>
<td>Gender</td>
<td>87.05%</td>
<td>91.76%</td>
</tr>
<tr>
<td>Glasses</td>
<td>91.78%</td>
<td>92.82%</td>
</tr>
<tr>
<td>Beard</td>
<td>92.31%</td>
<td>93.22%</td>
</tr>
<tr>
<td>Moustache</td>
<td>91.11%</td>
<td>92.21%</td>
</tr>
</tbody>
</table>
Chapter 6

Conclusion

In this thesis, a deep learning based approach has been demonstrated for the task of semantic facial feature recognition. This approach is primarily based on the use of convolutional neural networks on two dimensional pre-processed and aligned images of faces.

Through this thesis, a study that explores the effects of network hyper-parameters on the classification performance has been conducted. This has lead to estimation of the near-optimal configuration of the network. This study suggests that a deep convolutional network based approach is naturally well suited for the task of facial expression recognition from images of faces, providing sufficiently high classification accuracy. It is shown that addition of deterministic pre-processing and alignment steps for the input data greatly aids in improving the performance.

This thesis also demonstrates that such a deep network can easily be adapted to the tasks of recognising additional semantic features. Experimental results have shown near-human estimates of performances. However, it is observed that the discrimination power of deep networks are highly dependant on the distribution and quality of the training data.

The relation between the high-level semantic features and low-level descriptors has also been studied. Specific intuitive similarities have been observed between the low-level descriptors for different tasks. Use of this commonality among low-level descriptors is demonstrated by training a single network to jointly classify multiple semantic facial features. This has produced performances that are very similar to dedicated individual classification networks.

Finally, a novel scheme for training deep networks to generate complete 3-D Active Appearance Models of faces from 2-D images has been shown. It is observed that the
resulting face models are able to follow the shape of the original faces well, and also infer the pose of the head.

All the networks in this thesis are primarily trained and tested on the Facial Expression Recognition Challenge-2013 dataset, and on an internal face dataset from VicarVision.

The following suggest some potential future work:

- More extensive experimentation with alternative pre-processing techniques could be carried out. Methods like whitening are used in a wide range of machine learning tasks to reduce redundancy, and could be used to aid deep networks as well.
- The networks described in this thesis are heavily dependant on the face localization and alignment pre-processing step. Considering the success of deep learning for object detection, incorporating face localization and alignment stages within the deep network framework could greatly aid the robustness of the system.
- A limiting factor in the conducted experiments is the available computational resources. This calls for more experimentation on larger networks, as the true optimal performance of these network can only be achieved after extending the upper bound restrictions on the network size.
- Another limiting factor is the size of the dataset and the quality of annotations. Since deep learning is a data driven approach, the discriminating power of these networks can be improved by training them on a larger, homogeneously distributed dataset with more precise annotation.
- The true modelling error of the 3-D face models generated by the deep network can be better evaluated on a test set with 3-D scanned faces as the ground truth.
Appendix A

Backpropogation in Convolutional Layers

Considering a two dimensional $I \times J$ neuron layer and a convolutional layer $\ell$ with a $\hat{I} \times \hat{J}$ filter $\kappa$, the pre-nonlinearity input to the neurons of layer $\ell$ and its output can be given by:

$$x_{i,j}^{\ell} = \sum_{i=0}^{\hat{I}-1} \sum_{j=0}^{\hat{J}-1} \kappa_{i,j} \cdot y_{i,j}^{\ell-1} (i+\hat{i})(j+\hat{j})$$  \hspace{1cm} (A.1)

$$\& \hspace{0.2cm} y_{i,j}^{\ell} = f(x_{i,j}^{\ell})$$  \hspace{1cm} (A.2)

where $f$ denotes the neuron’s activation function.

For an error function $E$, the error that needs to be computed for neurons in the convolutional layer $\ell$ from the previous layer can be given by $\frac{\partial E}{\partial y_{i,j}^{\ell}}$. To compute the gradient component for each weight in the convolutional layer, the chain rule can be used:

$$\frac{\partial E}{\partial \kappa_{i,j}} = \sum_{i=0}^{\hat{I}-1} \sum_{j=0}^{\hat{J}-1} \frac{\partial E}{\partial x_{i,j}^{\ell}} \frac{\partial x_{i,j}^{\ell}}{\partial \kappa_{i,j}} = \sum_{i=0}^{\hat{I}-1} \sum_{j=0}^{\hat{J}-1} \frac{\partial E}{\partial x_{i,j}^{\ell}} y_{i,j}^{\ell-1} (i+\hat{i})(j+\hat{j})$$  \hspace{1cm} (A.3)

Here, we sum over all the expressions of $x_{i,j}^{\ell}$ that contains $\kappa_{i,j}$, because of the weight sharing in convolutional neural networks.

The computation of the gradient requires knowing $\frac{\partial E}{\partial x_{i,j}^{\ell}}$, which can also be computed by using the chain rule:

$$\frac{\partial E}{\partial x_{i,j}^{\ell}} = \frac{\partial E}{\partial y_{i,j}^{\ell}} \frac{\partial y_{i,j}^{\ell}}{\partial x_{i,j}^{\ell}} = \frac{\partial E}{\partial y_{i,j}^{\ell}} \frac{\partial}{\partial x_{i,j}^{\ell}} \left( f(x_{i,j}^{\ell}) \right) = \frac{\partial E}{\partial y_{i,j}^{\ell}} f'(x_{i,j}^{\ell})$$  \hspace{1cm} (A.4)
Now, we can derive the backpropagation of the error back to the previous layer, also using chain rule:

\[
\frac{\partial E}{\partial y_{i,j}^{\ell-1}} = \sum_{i=0}^{\hat{i}-1} \sum_{j=0}^{\hat{j}-1} \frac{\partial E}{\partial x_{i-\hat{i},j-\hat{j}}^{\ell}} \frac{\partial x_{i-\hat{i},j-\hat{j}}^{\ell}}{\partial y_{i,j}^{\ell-1}} \approx \sum_{i=0}^{\hat{i}-1} \sum_{j=0}^{\hat{j}-1} \frac{\partial E}{\partial x_{i-\hat{i},j-\hat{j}}^{\ell}} \kappa_{i,j} \tag{A.5}
\]

### A.1 Stochastic Gradient Descent

Simple gradient descent can be looked at as making repeated small steps in ordered to reduce an error term, defined by some loss function. This can be imagined as rolling down an error-surface which is defined by this loss function. This is achieved by computing the gradient of this surface. This is also the basic principle of stochastic gradient descent, with the key difference being that gradient is estimated from a single training example at a time, instead of the whole training set. This results in faster convergence.

A more efficient way of performing stochastic gradient descent is by using mini-batches. In this technique, instead of computing the gradient based on each training example, the gradient is computed based on a small batch of training examples at a time. The advantage of using such a mini-batch mode is that the estimation of the gradient is much smoother (lower variance due to higher number of samples).
Bibliography


[9] Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, et al. Recent advances in deep learning for speech research at microsoft. *ICASSP 2013*, 2013.


