Fully Connected Artificial Neural Network based Approach for Face Detection

Sachin Bansal¹ Shubham Varshney²

¹,²Department of Computer Science & Engineering
¹²Galgotias College of Engineering and Technology, Greater Noida, India

Abstract— In this work, A Neural Network based approach is presented to detect upright frontal faces in an image. The Neural Network used is a fully connected network with 3 layers. This network examines 20x20 sized windows in an image, and decides whether it contains a face or not. For the negative face examples, we use bootstrap method. After an initial small dataset of non-face examples, we add false detections to the training set as the training progresses. Simple heuristics, like merging overlapping and arbitrating between multiple neural networks can further increase the accuracy of the model.

Key words: Face Detection, Pattern Recognition, Computer Vision, Artificial Neural Networks, Machine Learning

I. INTRODUCTION

Face detection is a process being used in a variety of applications that identifies human faces in digital images. It has been proven that the first step in automatic facial recognition – the accurate detection of human faces in scenes, is the most important process involved. When faces can be located exactly in any scene, the recognition step afterwards is quite simple.

In this paper, a face detection system is implemented using Artificial Neural Network.

Face detection in itself is a challenging problem. This is due to the fact that faces are not rigid objects. Face appearance depends on many factors like emotions, lighting conditions and pose. This is why so many methods have been developed in the past years.

The goal is to detect faces in cluttered backgrounds very quickly. This situation can be found in many applications such as surveillance of public places, common Access Control conditions. So far learning-based approaches have been the most effective and have therefore attracted a lot of attention the last years.

Pattern recognition is a modern day machine intelligence problem with numerous applications in a wide field, including Face recognition, Character recognition, Speech recognition as well as other types of object recognition. The field of pattern recognition is still very much in its infancy, although in recent years some of the barriers that hampered such automated pattern recognition systems have been lifted due to advances in computer hardware providing machines capable of faster and more complex computation.

Face detection, although a trivial task for the human brain has proved to be extremely difficult to perform computationally. It is commonly used in applications such as human-machine interfaces and automatic access control systems. Face recognition involves comparing an image with a database of stored faces in order to identify the individual in that input image. The related task of face detection has direct relevance to face recognition because images must be analyzed and faces identified, before they can be recognized.

Detailed description of the system can be found in Section 2. In section 3, the performance of the system is examined. Conclusions and future scope can be found in Section 4.

II. DESCRIPTION OF THE SYSTEM

Our system operates in two stages: it first applies a set of neural network-based filters to an image, and then uses an arbitrator to combine the outputs. The filters examine each location in the image at several scales, looking for locations that might contain a face. The arbitrator then merges detections from individual filters and eliminates overlapping detections.

A. Neural Network Based-Filter

The first component of our system is a filter that receives as input a 20x20 pixel region of the image, and generates an output ranging from 1 to 0, signifying the presence or absence of a face, respectively. To detect faces anywhere in the input, the filter is applied at every location in the image. To detect faces larger than the window size, the input image is repeatedly reduced in size (by subsampling), and the filter is applied at each size. This filter must have some invariance to position and scale. The amount of invariance determines the number of scales and positions at which it must be applied. For the work presented here, we apply the filter at every pixel position in the image, and scale the image down by a factor of 0.2 for each step in the pyramid.

First, a preprocessing step, is applied to a window of the image. The window is then passed through a neural network, which decides whether the window contains a face. The preprocessing first attempts to equalize the intensity values in across the window. We fit a function which varies linearly across the window to the intensity values in an oval region inside the window. Pixels outside the oval may represent the background, so those intensity values are ignored in computing the lighting variation across the face. The linear function will approximate the overall brightness of each part of the window, and can be subtracted from the window to compensate for a variety of lighting conditions. Then histogram equalization is performed, which non-linearly maps the intensity values to expand the range of intensities in the window. The histogram is computed for pixels inside an oval region in the window. This compensates for differences in camera input gains, as well as improving contrast in some cases.

The preprocessed window is then passed through a neural network. The Network is a fully connected network, consisting of 3 layers including Input and Output layer. Input layer consists of 400 units representing each pixel of the 20x20 window. These units are then mapped to each of the 25 units of hidden layer, which are then mapped to a...
single unit of the output layer. The network has a single, real-valued output, which indicates whether or not the window contains a face.

To train the neural network used in stage one to serve as an accurate filter, a large number of face and non-face images are needed. For this purpose, MIT CBCL face dataset is used. The images contained faces of various sizes, orientations, positions, and intensities.

Fifteen face examples are generated for the training set from each original image, by randomly rotating the images (about their center points) up to 10°, scaling between 90% and 110%, translating up to half a pixel, and mirroring. Each 20x20 window in the set is then preprocessed (by applying lighting correction and histogram equalization). The randomization gives the filter invariance to translations of less than a pixel and scaling of 20%. Larger changes in translation and scale are dealt with by applying the filter at every pixel position in an image pyramid, in which the images are scaled by factors of 0.2.

Non-face images are collected before and during training. Before the training, the non-face images were taken from MIT CBCL dataset. The images are collected during training, in the following manner:
1) Train a neural network to produce an output of 1 for the face examples, and 0 for the non-face examples. The training algorithm is standard error backpropagation. On the first iteration of this loop, the network’s weights are initialized randomly. After the first iteration, we use the weights computed by training in the previous iteration as the starting point.
2) Run the system on an image of scenery which contains no faces. Collect subimages in which the network incorrectly identifies a face (an output activation >= 0.5).
3) Select up to 250 of these subimages at random, apply the preprocessing steps, and add them into the training set as negative examples. Go to step 1.

B. Merging Overlapping Detections
The example in Fig. 1 showed that the raw output from a single network will contain a number of false detections. In this section, we present a strategy to improve the reliability of the detector: merging overlapping detections from a single network.

Fig. 1: Multiple Detections

Note that in Fig. 1, most faces are detected at multiple nearby positions or scales, while false detections often occur with less consistency. This observation leads to a heuristic which can eliminate many false detections. For each location and scale, the number of detections within a specified neighborhood of that location can be counted. If the number is above a threshold, then that location is classified as a face. The centroid of the nearby detections defines the location of the detection result, thereby collapsing multiple detections. In the experiments section, this heuristic will be referred to as “thresholding”. If a particular location is correctly identified as a face, then all other detection locations which overlap it are likely to be errors, and can therefore be eliminated. Based on the above heuristic regarding nearby detections, we preserve the location with the higher number of detections in a small neighborhood, and eliminate locations with fewer detections. In the discussion of the experiments, this heuristic is called “overlap elimination”.

The implementation of this heuristic is illustrated in Fig. 2.

Fig. 2: Merging Overlapping Detections

Each detection at a particular location and scale is marked in an image pyramid, labelled the “output” pyramid. Then, each location in the pyramid is replaced by the number of detections in a specified neighborhood. This has the effect of “spreading out” the detections. A threshold is applied to these values, and the centroids (in both position and scale) of all above threshold regions are computed. All detections contributing to a centroid are collapsed down to a single point. Each centroid is then examined in order, starting from the ones which had the highest number of detections within the specified neighborhood. If any other centroid locations represent a face overlapping with the current centroid, they are removed from the output pyramid. All remaining centroid locations constitute the final detection result.

III. EXPERIMENTAL RESULTS
In this section, we present the error rates of the system over a test set. The system was tested on the test set provided with MIT CBCL test set. This set consists of 472 faces and 23,573 non-faces. After the first iteration, the accuracy was
found to be 98.67% for faces and 22% for non-faces. But, as the network was subjected to continuous false-positives training, the accuracy stabilizes to 59.91% for faces and 99.67% non-faces. This is shown in Table 1.

<table>
<thead>
<tr>
<th>Case</th>
<th>Faces Accuracy (%)</th>
<th>Non-Faces Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Data</td>
<td>98.67</td>
<td>22</td>
</tr>
<tr>
<td>FP1</td>
<td>91.67</td>
<td>39.33</td>
</tr>
<tr>
<td>FP2</td>
<td>82.33</td>
<td>61.67</td>
</tr>
<tr>
<td>FP3</td>
<td>75.67</td>
<td>82.67</td>
</tr>
<tr>
<td>FP4</td>
<td>70</td>
<td>94.67</td>
</tr>
<tr>
<td>FP5</td>
<td>68.33</td>
<td>97.33</td>
</tr>
<tr>
<td>FP6</td>
<td>65.27</td>
<td>98.33</td>
</tr>
<tr>
<td>FP7</td>
<td>62.67</td>
<td>98.33</td>
</tr>
<tr>
<td>FP8</td>
<td>61.33</td>
<td>99.67</td>
</tr>
<tr>
<td>FP9</td>
<td>59.91</td>
<td>99.67</td>
</tr>
</tbody>
</table>

Table 1: Test Set Accuracy

IV. CONCLUSIONS AND FUTURE RESEARCH

Our algorithm can detect between 59.91% of faces and 99.67% of non-faces, with an acceptable number of false detections. Depending on the application, the system can be made more or less conservative by varying thresholds used. The system has been tested on a wide variety of images, with many faces and unconstrained backgrounds.

There are a number of directions for future work. The main limitation of the current system is that it only detects upright faces looking at the camera. Separate versions of the system could be trained for each head orientation, and the results could be combined using arbitration methods similar to those presented here. Preliminary work in this area indicates that detecting profiles views of faces is more difficult than detecting frontal views, because they have fewer stable features and because the input window will contain more background pixels.

REFERENCES