CV spring assignment

256504

April 2023

Abstract

This report covers the task of face alignment, and designing a model that can accurately predict the shape of a face in an image. The aim of this report was to propose and criticise a potential solution to this problem. We found that the use of a random regression forest was a flawed but viable option, but ultimately outclassed by linear regression.

1 Introduction

Face alignment is the process of determining the shape and orientation of a (detected) face in an image. Within this report we will go over a random regression forest and its effectiveness at for the task. Regression forests use multiple randomised decision trees, and returns the average prediction from them all [Pedregosa et al. (2011)].

2 Background

Face alignment is a critical step in many digital processes in use today. For example: Facial recognition/Face ID; Face filters in social media; Deepfake etc. It involves pinpointing different key points in an image which together map out the shape and orientation of the face in the image.

Figure 1: This is a visualisation of an aligned face. There are 44 points on the image

The points in Figure 1 have been determined using a face alignment tool and are points that are applicable to the vast majority of faces e.g. the corners of the eyes and mouth; the extremities of the jawline etc. A fully functional face alignment method should be able to locate these points on any suitable image of a face.

3 Methods

There are multiple steps involved in designing a model for face alignment. The general sequence is as follows.

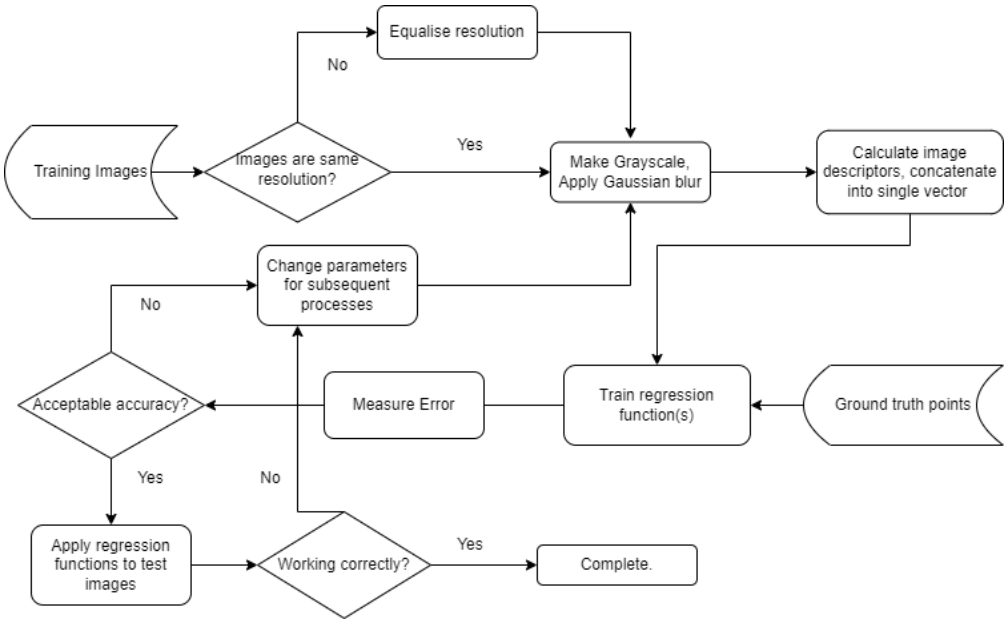

Figure 2: This flow chart depicts the overall process of training a regression model for face alignment

3.1 Pre-processing

The first step in face alignment is processing the images in such a way that makes them easier to read. A good way to start would be uniforming the resolution, size, shape and colour of the data set if not done so already. The data set we worked with during this investigation already had images of exactly the same resolution, and with faces already of roughly (but not exactly) the same size and in the same location. All that was left to maximise uniformity was to make all the images grayscale to eliminate redundant differences in colour.

The next step is to try and remove redundant detail within the image. This can be done using various filters, which due to the findings of Joshi et al. (2021), we chose to apply Gaussian blur to our images in preparation for image description.

Figure 3: An image from the training data set converted to grayscale and with gaussian blur applied

As seen in Figure 3, gaussian blur smooths an image so that the finer details within said image are hidden, leaving only the macro details visible i.e. eye location, face shape etc. This in turn helps to highlight the areas of interest that we are trying to pinpoint during face alignment.

During development, we tested slight variations in gaussian blur to see if they significantly impacted the accuracy of our model.

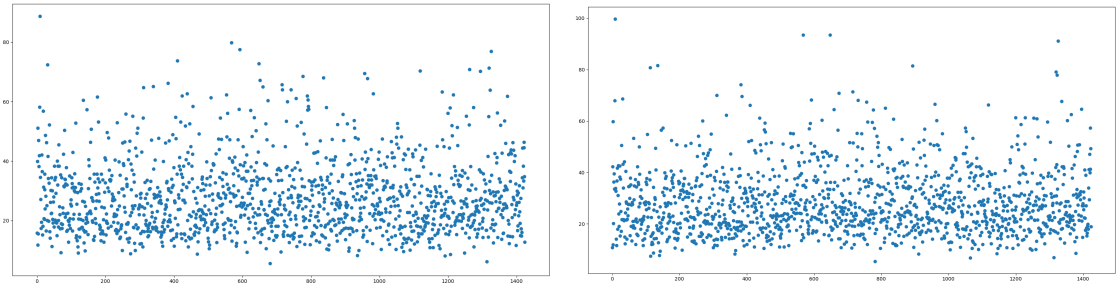

Figure 4: These are scatter plots for the error of models trained using images blurred with a (15,15) and (35,35) kernel respectively

The points in Figure 4 represent the mean squared error of every image in the training data set, with the X axis being the index of the image and the Y axis being the error. Error has been calculated by determining the euclidean distance between a predicted point and its corresponding ground-truth point. As you can see, both kernels give roughly the same accuracy. The right hand kernel of (35,35) seems to reign in some of the outliers, however, so that is what we elected to use.

3.2 Image Description

Now that our images are adequately pre-processed, we need to choose how we are going to describe our images in a way that is sufficient to both train our regressor and predict our key points. We chose SIFT [Wu et al. (2013)] as our method to produce image descriptors. SIFT takes small regions of an image and describes the edge orientations by passing the pixels in that region through a filter, and putting the result in a vector of length 128. Rather than allowing SIFT to detect the key areas it wants to describe, we manually set it do describe the entire image over a regular grid of 5x5.

In figure 5, you can see the placement and size of the SIFT descriptors that will be calculated on every image prior to prediction. Unlike the example above, the descriptors will be calculated on an image that has

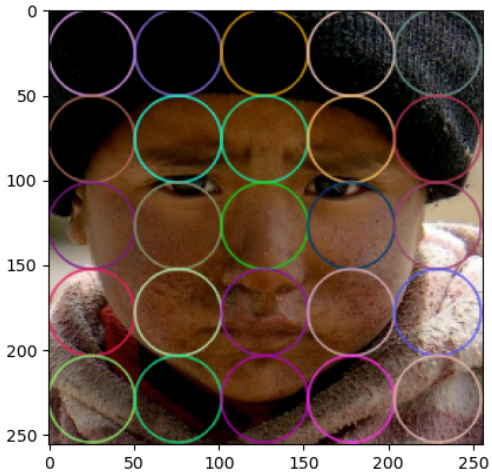

Figure 5: This is a visualisation of the grid of SIFT descriptors calculated on the images.

been through the elected pre-processing methods discussed earlier. The final step to make these calculations ready to be input into the prediction model is to concatenate all 25 descriptors into 1D vector of length 25x128.

3.3 Prediction model

Now we need to decide what model to train and how to train it. As this problem is essentially predicting 88 numbers (44 sets of coordinates), we can treat this as a regression problem as the variables we are trying to calculate can have almost infinite values (unlike a classification problem). For our regressor, we used a module called RandomForestRegressor from Pedregosa et al. (2011): This creates and trains a forest of regression trees based off of the parameters that you give it.

Initially, we tried training one forest to perform the task. This led to mediocre results at best on the training data, and inadequate on the testing data. Therefore, we decided to take a different approach.

3.3.1 Using 2 regression forests

We needed a solution to the problem of the angle of the face, as well as potential overfitting. One solution would be to predict the region in which our predicted points should be first. We therefore elected to split the problem into 2 stages i.e. 2 regression forests. The first forest would predict just the corners of the eyes and mouth, and the tip of the nose. Then, new SIFT descriptors would be calculated around the regions of these coordinates. Then, these descriptors would be fed into another regression forest which would in turn predict the full set of 44 coordinates (overwriting the initial predictions of the first 5 points).

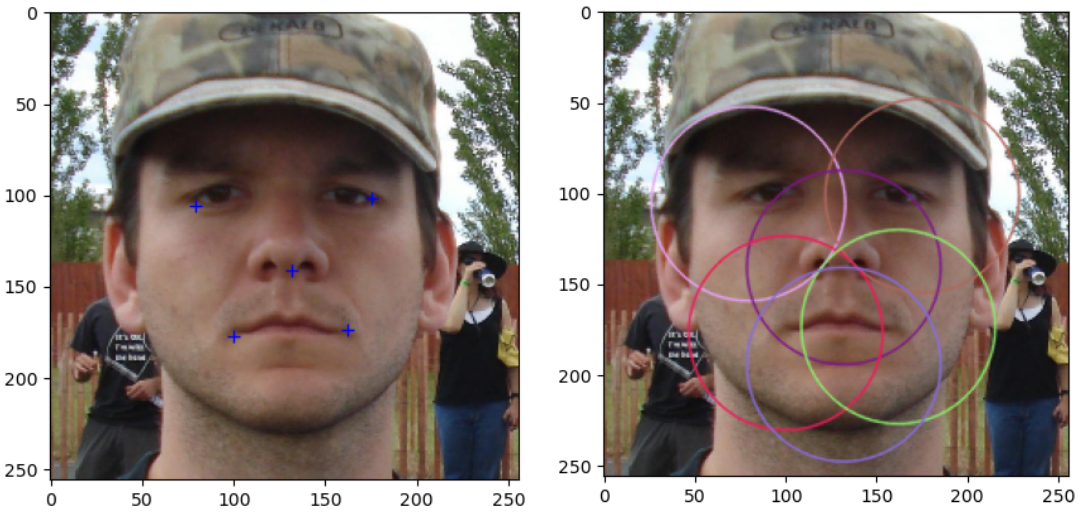

Figure 6: Left shows predicted points, right shows calculated descriptors

As you can see, the descriptors in the right hand image cover the entire face, with minimal extra area being covered. This is useful as it eliminates any redundant information being fed into the second and final regression forest. The diameter of these descriptors is equal to the distance between the corner of the left eye and right corner of the mouth, multiplied by a scalar. You will also notice a sixth descriptor placed around the chin area. This is to make sure the entire face was covered as the initial 5 were not enough to do so. The centre of this descriptor is on the bottom of the circumference of the descriptor on the nose.

4 Results

To get a clearer idea of how our solution would fare against unseen data, we trained our regressors on roughly 80% of the training data. We then used the remaining 20% for a validation set, to measure the effectiveness of the regressors on data that they have not seen.

4.1 Prediction of first 5 points

The first iteration of our face alignment tool seemed to work fantastically when applied to the training data, with most cases having minimal error. Unfortunately, the validation data did not return with the same results

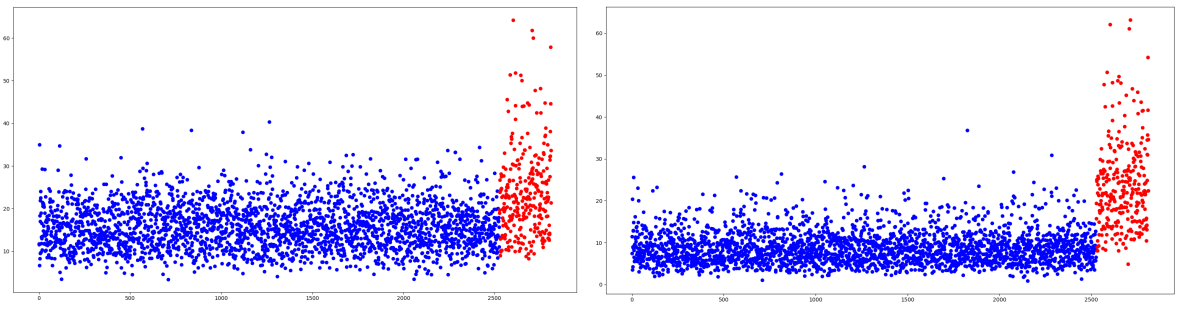

Figure 7: Left shows error for primary regression forest with max tree depth 10; Right shows max tree depth of 25
Blue: Training, Red: Validation

The plots in Figure 7 show that unfortunately neither works as well on the validation data as they do on the training data. However, the regressor with a max tree depth of 10 has a slightly lower average error across the validation images of approximately 23, compared to 24. This difference is only slight, but it is understandable that the deeper tree could be afflicted by issues of overfitting.

If you compare Figure 7 to either of the plots in Figure 3, you will notice straight away that the error is now considerably lessened on the training data set; proving the necessity of having 2 regression trees rather than one.

Figure 8: Left shows success case, right shows failure

We ran our first regression forest on a small set of example images. Figure 8 shows a success and failure case within this set. The failure case indicates that when the eyes are larger and have darkened outlines (eyeliner), the regressor potentially struggles to pinpoint the corner. This in turn could skew the location of the other 3 points.

4.2 Predicting the full set of points

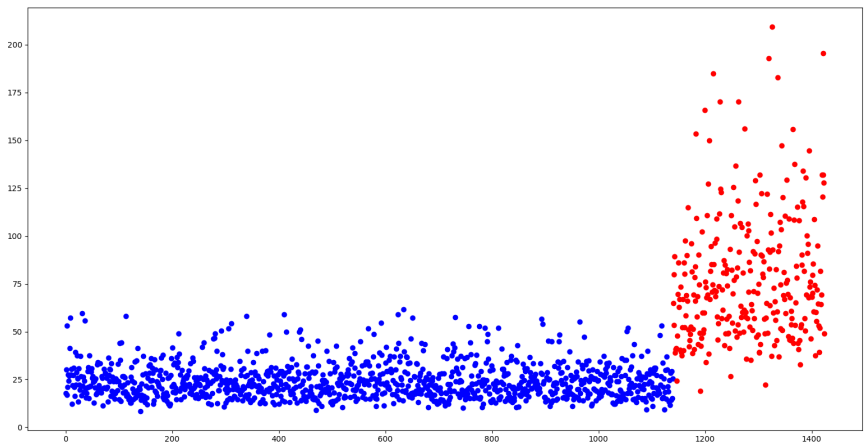

Figure 9: Shows scatter for the secondary regression forest with a max tree depth of 10
Blue: Training data, Red: Validation data

Now that there are 44 points to predict rather than 5, the mean error of the images goes up considerably. Nevertheless the trend is the same before; the prediction on the validation data is much worse than on the training data. Despite this, there are still some clear success cases from the example dataset.

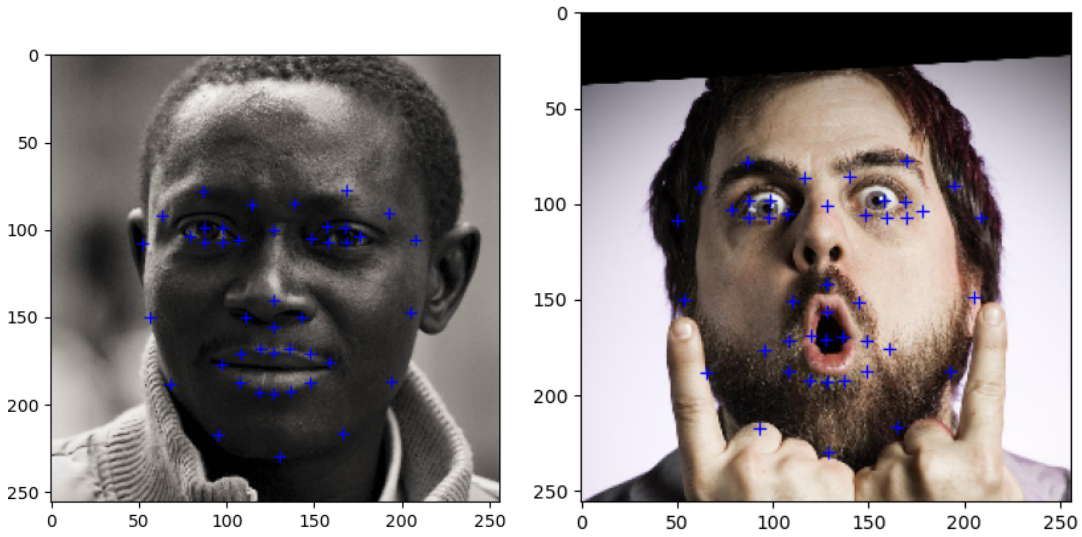

Figure 10: Left shows success case, right shows failure

Now, as demonstrated in the graph and the leftmost example, it is clear that the regressor can predict the points on some images of faces, provided that their angle and expression is somewhat regular. As demonstrated in the failure case on the right, the regressor struggles when the facial expression is irregular.

5 Conclusion

In this report, we found that breaking down the stages of regression helped considerably in the overall accuracy of our results. However, our regressor struggled to perform on the wide variety of facial expressions and angle of capture of the photographs that it will inevitably come across. This could be due to our choice of regression forests to solve this problem. In retrospect, a better choice may have been a series of cascading linear regression models [Lee et al. (2015)]. Regardless, our solution has had some success in predicting the points on images with more regular facial expressions.

References

Joshi, S., Owens, J. A., Shah, S. & Munasinghe, T. (2021), Analysis of preprocessing techniques, keras tuner, and transfer learning on cloud street image data, *in* '2021 IEEE International Conference on Big Data (Big Data)', IEEE, pp. 4165–4168.

Lee, D., Park, H. & Yoo, C. D. (2015), Face alignment using cascade gaussian process regression trees, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', pp. 4204–4212.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Wu, J., Cui, Z., Sheng, V. S., Zhao, P., Su, D. & Gong, S. (2013), 'A comparative study of sift and its variants.', *Measurement science review* **13**(3).