**Computer Vision Report**

It is important to note that the times used in this report are machine dependant, they are included to show the respective change between different instances to show improvements/loss. Also note that all the mean loss calculations in the diagram are for images scaled to 128 x 128, the loss function was not customed to consider resizing of images which would obviously make the points closer together despite potentially not being as accurate but appearing so (I have updated it now).

**A collage of people with red dots

Description automatically generatedPrior research**

Figure 1 Labelled image data used to train and test model.

The train data we have consists of 2811 Images of 256 x 256 pixels with RGB values. This is a LOT of information to process so a good feature extraction method will be essential as we want to use ALL the train data. The labels have 44 (x, y) points for each image which correspond to the face features.

Possible features:

1) **PCA** with Just the raw image data, (Principal component analysis finds the most “influential” components to train our model)

2) **SIFT** (Scale invariant feature transform) can find features that are immune to scaling, rotation, and distortion. (Ning, 2019)

3) **HOG** (Histogram of orientated gradients) uses a sliding window to identify edges detected at certain angles within each square of the image.

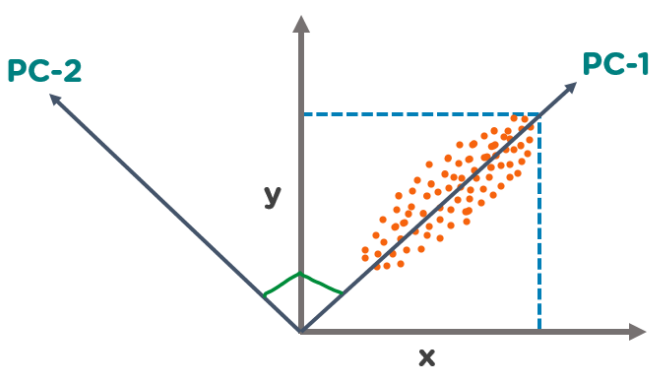


Figure 2 Demonstration of Principle Component Analysis. (Biswal, 2023)

**Pre-processing the image data**

The first pre-processing steps will involve taking all the images and resizing them to a reduced fixed size to reduce the amount of data. Additionally, we can grey scale, blur and normalise the pixel values. I compared processing times and the loss for each combination of these 3 pre-processing methods.

A comparison of a graph

Description automatically generated with medium confidence

Figure 3 Effect on accuracy and processing time of pre-processing techniques.

These results show that grey scaling and normalising the image reduce the processing time and that none of the pre-processing methods influence the accuracy (mean loss) of the linear regression model. Also, gaussian blurring has no real effect (tested with a 5x5 filter).

A close-up of a child's face

Description automatically generated

Augmenting image data involves producing more training data by transforming the current data by rotating, flipping, and repositioning it to “increase its diversity” (Sajid, 2022). For this task I believed that I could create more training data by horizontally flipping each image. (Vertical flipping not needed as the data seems to all be the “right way up”). However, this didn’t work which I will address in fail case section.

A graph of a number of components

Description automatically generated**Image feature selection/extraction**

Principal component analysis (PCA) is a method used to identify the most “significant features” in a dataset (Biswal, 2023). In this case, the train data is pre-processed image data with 128 x 128 pixels, which is treated as a feature in a high-dimensional space. PCA then creates a set of orthogonal axes, known as principal components.

The “elbow” method tells us how many principal components are responsible for making up most of the variance in the data. From the diagrams we can see that at around 32 components we have reached about 75% and at 176 components we have reached 90% of the variance.

Figure 4 shows PCA can significantly decrease processing times around 95% (Dependant on how many components, here I’m using 32). Additionally, it increases the accuracy of the model (not by much but still important). This allows us to simply use the **processed image data** from each image as the training data, whilst being quite a simple approach it proved to be the most effective and efficient(see fail cases).

**A comparison of a number of blue and white bars

Description automatically generated with medium confidence**

Figure 5 The effects of using PCA.

**Predictive model and Loss function**

The Loss function for this kind of data uses a mean Euclidean distance. For a classification model you can measure accuracy by diving the correct predictions by the total predictions, but with a ML task like this where we compare actual points [x, y] to our predicted points [x, y] - we use the “Euclidean distance” which is a measure of the straight-line distance between two points in a Euclidean space. We then take the average of this across all 44 points to find the *mean Euclidean distance* which is an accurate measure of how well our model is performing.

A graph with points and lines

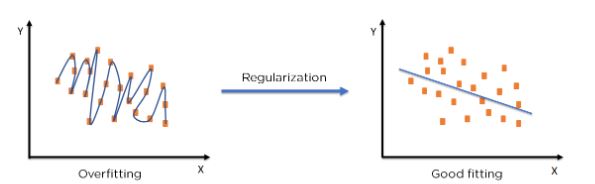
Description automatically generated

Possible models I considered:

* Linear regression model – Good for simple tasks with linear relationships but struggle with complex facial patterns. Super easy to implement.
* Convolutional neural network (CNN) – Ideal for capturing facial spatial dependencies, but computationally expensive and requires a lot of labelled data.
* Long short-term memory networks (LSTM) – Suitable for learning long term dependencies in sequential facial data, but prone to training difficulties due to “vanishing gradient problems”. (Hung, 2023)

Linear regression is a statistical technique for modelling relationships between a dependent variable and one or more independent variables by fitting a linear equation to observed data through a process called Ordinary Least Squares (OLS).

However, Linear regression is prone to overfitting when dealing with high dimensional data (such as image data). Regularization techniques such as “Lasso” (L1) and “Ridge” (L2) are used to address this (Ansari, 2023). These methods introduce a penalty to regression equation, constraining the coefficient of the predictor variables. The alpha value represents the strength of the regularisation applied, helping to prevent overfitting by discouraging complex models.



A diagram of a process flow

Description automatically generated

**A white text on a white background

Description automatically generated**

A graph with a line

Description automatically generated

Figure 6 The effects on validation loss with regularisation strength (alpha)

**Results analysis**

Here are the results of doubling the amount of principle components on time and accuracy. If time isn’t a factor in the efficiency of my model, then to achieve more accurate results (for the predictions.csv) I will use the amount of component which gives the lowest loss – this would be 256 components which also has a lower PCA fit time compared to 128 components with a better accuracy! We can also observe from this data that as the PCA components drastically increases to 1024 components the model’s accuracy worsens due to **overfitting** the data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number of Components** | Mean loss | Train Time (s) | PCA Fit Time (s) | Total Time (s) |
| **8** | 6.07 | 0.088 | 11.35 | 11.438 |
| **16** | 5.80 | 7.60 | 14.40 | 22.00 |
| **32** | 4.82 | 7.62 | 14.93 | 22.55 |
| **64** | 4.02 | 9.13 | 16.87 | 25.00 |
| **128** | 3.90 | 11.32 | 22.38 | 33.70 |
| **176** | 3.87 | 12.98 | 26.02 | 39.00 |
| **256** | 3.74 | 13.42 | 19.26 | 32.68 |
| **512** | 3.79 | 13.64 | 34.69 | 48.33 |
| **1024** | 3.91 | 16.20 | 68.74 | 84.94 |

Here are the best and worst predictions on the test data set. The models final average mean loss was roughly about **3.72**, which means the worst prediction is more of an anomaly compared to the best prediction. It took the model around **17 seconds** to fully be trained with 256 principal components and to make a prediction on every single image in the held-out validation test set (20% of the train data). As you can see the worst prediction image face has the entire right side shadowed out which could explain why it has been predicted poorly.

**A person's face with a couple of images

Description automatically generated**

Figure 7 The models best and worst predictions comparison.

Below I have used the example images to test my model, visually its fairly accurate at finding both the position of the eyes and the general face shape but isn’t entirely accurate. In the 3rd image the mouth shape isn’t correct at all. I believe this is because in the train data there aren’t many faces where the mouth is open like this, the model gets confused.

A collage of men with red dots

Description automatically generatedA collage of men with red dots

Description automatically generated

**Fail cases and bias analysis.**

Augmentation resulted in a deterioration of the model’s performance, show by the increase in loss. From the images we can observe a compression along the X-axis suggesting a problem rooted in the augmentation process (we only flip the image horizontally).

A collage of a person and a child

Description automatically generated

Figure 8 Effect of augmentation on the best and worst predictions.

**A graph of a graph showing a number of blue rectangular objects

Description automatically generated**

The problem with the augmentation lies in learning distinct representations for both original and flipped images, leading to a dilemma when presented with a test image. The model struggles to decide whether to rely on the learned features from the original image or its horizontally flipped counterpart, resulting in an averaging effect. Removing the original image could fix this issue, but this would add no extra training data and slow down the process.

Two other feature detection methods I attempted to use where HOG and SIFT features.I managed to extract both features from the images, but I could not manage to implement the HOG features and the SIFT features gave worse performance than using the image pixel data with PCA.

Figure 9 Model 1: Without Augmentation, Model 2: With Augmentation

**A graph with a line and a number of components

Description automatically generatedA child smiling with many colored sticks

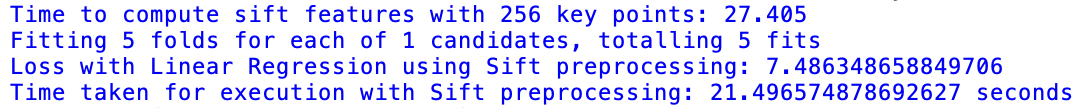
Description automatically generated**

**A close-up of a child

Description automatically generated**

Figure 10 SIFT features.

Figure 11 DOG features.

****

**Eye/Lip colour modification system evaluation**

A close-up of a person's face

Description automatically generated

A diagram of an extract point

Description automatically generated

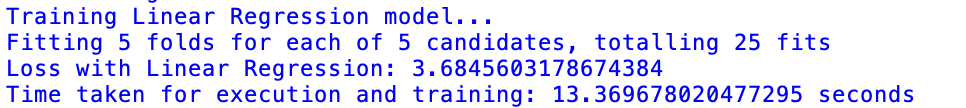
The eye/lip colour modification system works by using the predictive model to get the predictive points from the example images. I managed to locate the points that make up the lip and eye regions is by noticing that all the image label points are in a certain order that represent the feature of the face.

* 20 - 25 represents the left eyes
* 26 - 31 represents the right eye
* 32 - 43 represents the mouth.

I simply then use *matplotlib* patches to fill polygons with random RGB colours using the respective indexes with predicted points to find the feature region. The model will only be as good as the model in the first task as it is responsible for locating the predicted points for the lips/eyes.

**Evaluation and Summary**

I think that my final model is fairly accurate, and I explored many options/methods to increase the accuracy. However, I feel like I could have tried out more complex models instead of using a linear model as the relationships between facial alignment data are complex.



# Works Cited

Ansari, Y., 2023. *Medium.* [Online]   
Available at: https://medium.com/@novus\_afk/regularized-linear-regression-35d5eaaa84d5  
[Accessed 13 5 2024].

Biswal, A., 2023. *Simplilearn.* [Online]   
Available at: https://www.simplilearn.com/tutorials/machine-learning-tutorial/principal-component-analysis  
[Accessed 13 5 2024].

Hung, C.-L., 2023. *Deep learning in biomedical informatics.* [Online]   
Available at: https://www.sciencedirect.com/topics/computer-science/long-short-term-memory-network#:~:text=An%20LSTM%20model%20can%20recall,forget%20gate%2C%20and%20output%20gate.

Jordan, J., 2017. *Jeremy Jordan.* [Online]   
Available at: https://www.jeremyjordan.me/hyperparameter-tuning/  
[Accessed 13 5 2024].

Sajid, H., 2022. *Picsellia.* [Online]   
Available at: https://www.picsellia.com/post/image-data-augmentation  
[Accessed 13 5 2024].