Facial Keypoint Detection

Ashkan Esmaeili
aesmaili@stanford.edu

Khashayar Khosravi
khosravi@stanford.edu

SeyedShabaddin Mirjalili
ssmirjal@stanford.edu

***Abstract*— Facial Keypoint Detection is one of the most challenging and important topics, which is taken into account in realm of Machine Learning and Computer Vision. The importance originates in its applications, the most three important of which are: 1- Face recognition, which is of high importance in identification as an example. 2- Medical purposes and Biomedical applications, Medical surgeries and psychiatric tests like analyzing facial expressions could be carried out on patients using Facial Detection. 3- Tracking Faces in Images and Videos, these days many applications have a built-in feature of facial analysis. In this report, first, we briefly introduce the topic in section I. Then we proceed with the explanation of the methods and algorithms we used in section II including our proposed method PCLWLR. Then in section III, we mathematically formulate the PCLWLR, and finally, we report the achieved RMSE values for different methods in addition to providing plots illustrating keypoints detected on images. We conclude in section V and summarize the results.**

***Keywords*— Keypoint; target variables; edge detection; PCA; PCLWLR; LWLR**

I. Introduction

Face recognition is one of the most famous vision challenges. Many authors have worked on this topic. While deep convolutional networks known to be the state-of-art so far [1], here we focus on machine learning algorithms with less complexity that can also guarantee reasonable performance. The dataset we are going to use is provided for Kaggle Competition [2]. The training data consists of 7049 images of size 96×96 pixels. Each row in data matrix contains the coordinates (x-y positions) for 15 keypoints on the face, and in addition, the image pixels (gray-scale values) are stored in a row-ordered fashion in the same row of data. We could consider each pixel's value as a feature used for prediction. These pixel values are integers in the interval [0 255]. The 15 keypoints are left_eye_center, right_eye_center, left_eye_inner_corner,left_eye_outer_corner,right_eye_inner_corner,right_eye_outer_corner,left_eyebrow_inner_end,left_eyebrow_outer_end,right_eyebrow_inner_end,right_eyebrow_outer_end,nose_tip,mouth_left_corner,mouth_right_corner, mouth_center_top_lip, and mouth_center_bottom_lip respectively. In [3] authors utilized the Mean Patch Algorithm, as introduced in [4], to make accurate predictions about left_eye_center and right_eye_center. In this report, however we focus on prediction of the location of all 15 keypoints on new images of peoples' faces based on what we have learned from the image dataset provided.

It is worth noting that some target variables are missing. Only about 2000 images among 7049 training images have no missing target variables and this makes the matrix completion a necessary step of prediction.

While there are many papers in literature working on matrix completion [5], [6], and [7], it turns out that mean imputation on missing target variables leads to the least RMSE. The first and foremost challenge in image keypoints prediction is that, there is plethora of features (pixels) which makes predictions prone to overfitting. Therefore, feature selection and dimension reduction are necessary. Moreover, there is complexity dealing with this data due to large amount of variation in zooming, translation, and rotation of photos. There are photos which are far from standard caption making the prediction difficult like the following images:

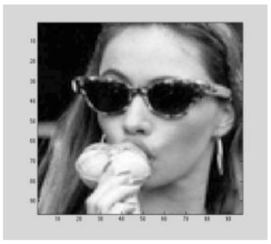
Fig. 1- Sample image

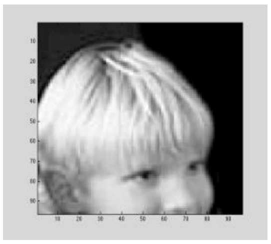
Fig. 2- Sample image

We notice that in Figure 1 the eyes, lips, and mouth information are missing. In Figure 2, lips are hidden and the location of eyes and nose are translated significantly from the standard mean values and further the photo is taken in a rotated fashion.

Therefore, we should look for a method which captures these rotations and translations. We have observed many images in Kaggle dataset. Most of them suffer from such problems. Some are quite blurred and ambiguous for which prediction may be meaningless. Yet, we did not purify the dataset from such troublesome images and worked with the dataset including all training photos.

II. ALGORITHMS

We have implemented several ideas for locating keypoints with better precision. First, some methods are complex in terms of runtime. At the beginning the naïve idea is to consider each pixel in the 96×96 as a feature and run the linear regression on the entire pixels, which is complex and prone to overfitting leading to a high RMSE on test set. By adding a regularization term and applying ridge regression, we can overcome the overfitting issue. Yet, we cannot implement more complex methods according to the large dimension of features. We had two ideas to reduce the dimension of features used to predict. First idea is applying edge detection on the images (edges are assumed to contain the most important information in images) as a feature selection step and we confined predictions to the boxes around keypoints to get rid of additional time consuming calculations. In Figures 3,4, and 5, one can find attempts we have made for the edge detection and ridge regression on the box around the keypoint for prediction. For edge detection purpose, we have applied 'Canny' filter in MATLAB [8]. This performed better in comparison to the previous method. However, there are still some flaws in this approach. The sensitivity of the edge detection algorithm affects the quality of prediction noticeably. In addition, it doesn't capture the rotation, translation, and other characteristics of images.

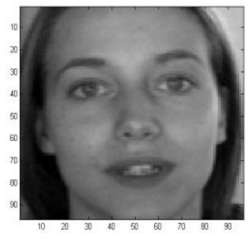
Fig. 3- The original image

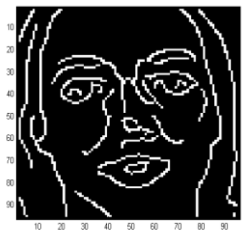
Fig. 4- Edges of the image found using the Canny filter

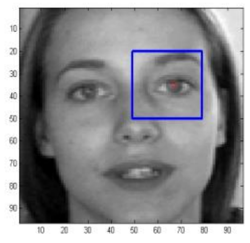
Fig. 5- The box of size 30 plotted around the left eye center which will be used for the prediction. Red dot shows the ground truth.

Secondly, we have come up with the idea of applying PCA to retrieve the first few singular vectors that are assumed to contain the most important characteristics in training set images. To proceed, we project all images onto the new subspace and then we can apply regression methods to obtain better performance. We used the idea of Locally Weighted Linear Regression (LWLR) to take similarities between images into the account of prediction purpose. This helps us to capture the important information in neighbor images. This idea is exactly similar to the logic of K-nearest neighbors (KNN) and LWLR. We will call the proposed method PCLWLR in this paper from now on, which stands for Principal Component Locally Weighted Linear Regression. In the next section, we provide the formulation and mathematical analysis of our algorithm. The RMSE values for each introduced methods will be reported in the summary section.

III. PCLWLR

In this section we explain how PCLWLR works. Let X denote the matrix, which includes all training images as vectors in its rows.
First, we extract the most important information of X via singular value decomposition.

$$X = U\Sigma V^T \tag{1}$$

Figure 6 shows the plot of eigenvalues of X. As it illustrates, the most important information of X is contained within the top 100 initial singular vectors. As a sanity check, we also tried our method for the first 200 singular values and the performance was not changed remarkably. Working with fewer singular vectors is also more desirable in the sense of the computational effort that we make for prediction. Hence, we work with the first 100 singular vectors from now on.

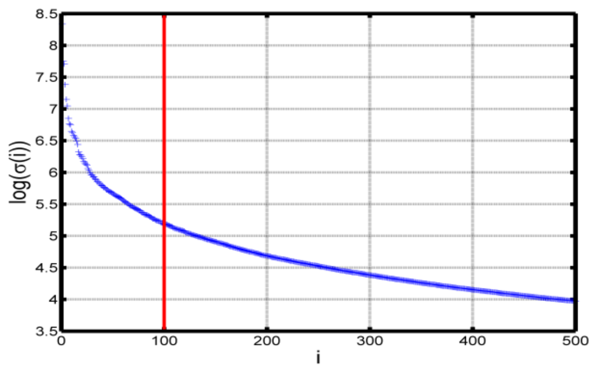

Fig. 6- Logarithm of singular values

Similar to Principal Component Regression (PCR) [9], after loading principal components, we project the data on the singular vectors to find scores $\tilde{X}$ as follows:

$$\widetilde{x_{(i)}} = \begin{bmatrix} v_1^T x^{(i)} \\ v_2^T x^{(i)} \\ \cdot \\ \cdot \\ \cdot \\ v_{100}^T x^{(i)} \end{bmatrix} \epsilon\, R^{100}, \quad \text{for } i = 1, 2, \dots, n. \tag{2}$$

Scores are predictors that we later on use for prediction in the new space. After mapping images to the new space, we define the Euclidean distance between the test score vector and training score vectors as follows:

$$D_i = \left\| \widetilde{x^{(test)}} - \widetilde{x^{(i)}} \right\|^2, \quad \text{for } i = 1, 2, \dots, n \tag{3}$$

Then, we can apply the idea of locally weighted linear regression after defining weights based on these distances. We define the weight matrix W as a diagonal matrix as following:

$$W_{ii} = \exp\left(-\frac{D_i}{2c\max(D)}\right), \quad \text{for } i = 1, 2, \dots, n \tag{4}$$

Here we normalized the distances by the maximum distance value. There is a parameter called c in this definition which is the bandwidth considered for neighbors in assigning the weights. We have tuned for c via hold-out cross-validation as in the following figure:

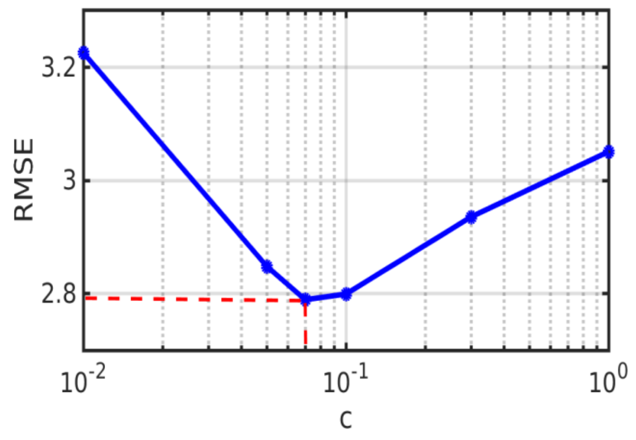

Fig. 7- RMSE values obtained by hold-out cross validation. Here λ is assumed to be 500.

This gives the optimal value of $c = 0.07$. Finally, we find the parameter for prediction θ using the following normal equation

$$\theta = (\tilde{X}^T W \tilde{X} + \lambda I)^{-1} \tilde{X}^T W y \tag{5}$$

Note that we included a regularization term in order to maintain a reasonable trade-off between bias and variance. Tuning for the parameter λ can be done via cross-validation as the following figure shows:

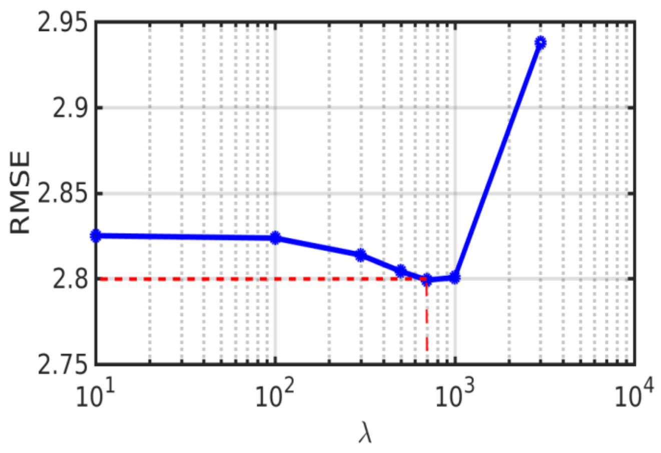

Fig. 8- RMSE values obtained by hold-out cross validation. c is taken to be 0.07.

The optimal value λ is found to be 700.
Note that, in order to find the best pair of c, λ one needs to alternatively optimize the value of c, λ. However after running these iterations we found the values of $c = 0.07, \lambda = 700$ close to optimal.
Finally, in order to make prediction of a target variable y we use the following formula:

$$\widetilde{y^{(test)}} = \theta^T \widetilde{x^{(test)}} \tag{6}$$

IV. Analysis and Simulations

To evaluate our performance, we use the 5-fold cross-validation method on our training data. We divide the training images into five subsets of equal size at random. Then, each time we train our method on 4 subsets and use the fifth subset as the validation set. The provided metric for performance evaluation is RMSE, which is given as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \widetilde{y^{(i)}}\right)^2} \tag{7}$$

where N is the total number of non-missing target variables of the validation set. Note that, since some of targets are missing we do not consider them in the above formula. Finally, we report the mean of RMSE values as our prediction error.
The following figure shows the RMSE values we achieved for each target variable under PCLWLR:

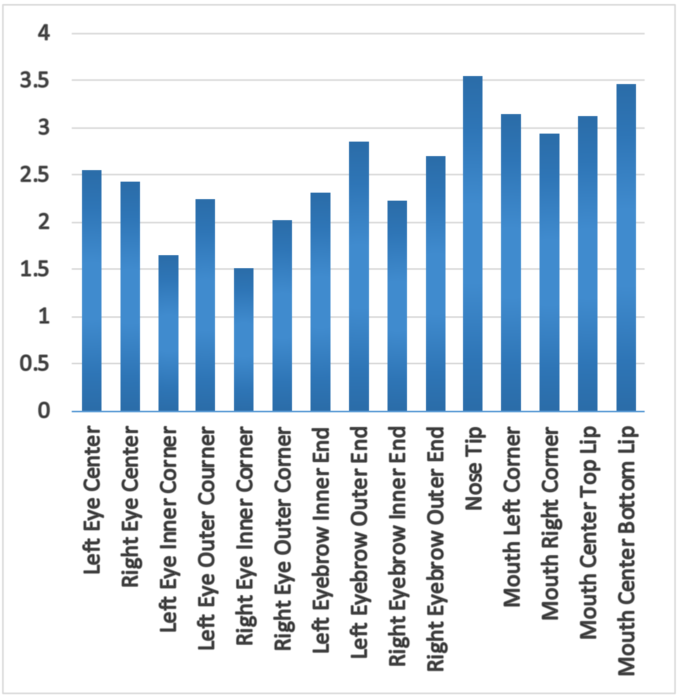

Fig. 9- RMSE values for different keypoints

As mentioned earlier, for better precision the error values are found using 5-fold cross-validation. We also used Bootstrapping to evaluate the confidence intervals for our estimated error values and have found out that the final RMSE of other methods fall outside the confidence interval of PCLWLR.

The following table shows the average RMSE of different methods when applied to all 15 target variables. RMSE of the proposed PCLWLR method on the left eye center is equal to 2.5494, and on the right eye center is equal to 2.4359, which gives us the mean RMSE of $\sqrt{(2.5494^2 + 2.4359^2)/2} = 2.4932$ on the first two keypoints. This shows a great improvement on the value reported in [3], where the authors were able to obtain the error equals to 2.8843 on the validation set.

TABLE I. - RMSE VALUES

METHODS	RMSE
Simple Linear Regression	6.81
Ridge Regression	3.04
Edge Detection + Box Finding + LWLR	2.93
PCLWLR	**2.79**

For better visual understanding, in the following images the keypoints predicted using PCLWLR are shown:

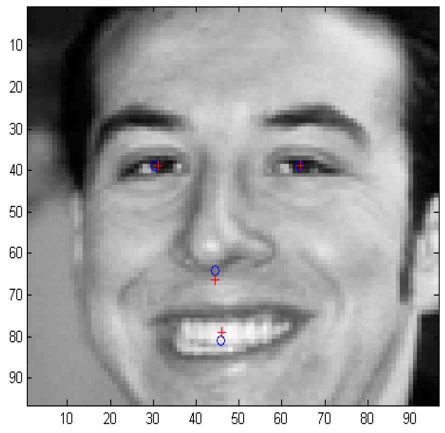

Fig 10- Predicted and true keypoints on a sample image. Red and blue points are ground truth and our predictions respectively.

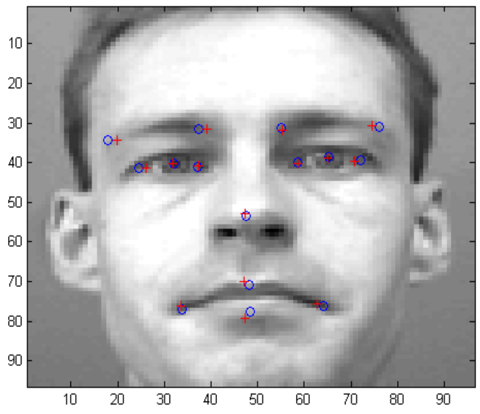

Fig 11- Predicted and true keypoints on a sample image. Red and blue points are ground truth and our prediction respectively.

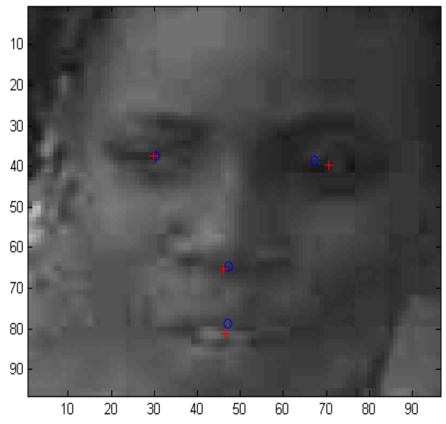

Fig 12- Predicted and true keypoints on a sample image. Red and blue points are ground truth and our prediction respectively.

V. SUMMARY

In summary, we proposed the PCLWLR method for the facial keypoint detection. In this method, after reducing the dimension via PCA, a regularized version of locally weighted linear regression was applied to train the model. Hold-out cross validation was then used for tuning the parameters of the model. Finally, we reported the error of the proposed method using 5-fold cross-validation and compared its performance with other methods. For future work, one idea is coming up with a better idea for finding near neighbors of images, so that the local regression methods achieve higher accuracy. The other method that can be implemented is Support Vector Regression (SVR). Deep Learning methods can also be applied to extract higher level features for the prediction purposes.

ACKNOWLEDGEMENTS

The data set we use here was graciously provided by Dr. Yoshua Bengio of the University of Montreal.

REFERENCES

[1] Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep convolutional network cascade for facial point detection." *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013.

[2] https://www.kaggle.com/c/facial-keypoints-detection

[3] Wang, Yue, and Yang Song. "Facial Keypoints Detection., *Project Report, CS229, Stanford University, 2014*

[4] Jain, Ramesh, Rangachar Kasturi, and Brian G. Schunck. *Machine vision*. Vol. 5. New York: McGraw-Hill, 1995

[5] Cai, Jian-Feng, Emmanuel J. Candès, and Zuowei Shen. "A singular value thresholding algorithm for matrix completion." *SIAM Journal on Optimization*20.4 (2010): 1956-1982.

[6] Keshavan, Raghunandan H., Andrea Montanari, and Sewoong Oh. "Matrix completion from a few entries." *Information Theory, IEEE Transactions on*56.6 (2010): 2980-2998.

[7] Candès, Emmanuel J., and Benjamin Recht. "Exact matrix completion via convex optimization." *Foundations of Computational mathematics* 9.6 (2009): 717-772.

[8] Canny, John. "A computational approach to edge detection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 6 (1986): 679-698.

[9] Zou, Hui, Trevor Hastie, and Robert Tibshirani. "Sparse principal component analysis." *Journal of computational and graphical statistics* 15.2 (2006): 265-286.