VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Artificial Intelligence

ITE2010

FACULTY: PROF. AJIT KUMAR SANTRA

Slot: A2+ TA2

Review- 2

Title : **OPTIMIZED FACIAL KEYPOINT DETECTION TO CREATE SELFIE FILTERS**

Team:

1. Sunil Kumar Sah – 18BCE2470
2. Manan Gupta – 16BIT0322
3. Varsha Agarwal - 17BIT0305

1. ABSTRACT

Detecting facial key-points is a critical element in face recognition, facial filter attachment. However, there is difficulty to catch key-points on the face due to complex influences from original images, and there is no guidance to suitable algorithms. In this paper, we study how dimensionality reduction affects a predictive model to locate key- points. Finally our conclusion after comparison of both algorithms based on the parameters such as Model Metric(mean absolute error) and Inference speed. The assumed outcome should be the increase in inference speed and better fitting of the data to the same model on applying dimensionality reduction i.e. PCA. This facial keypoints detection can be used to create facial filters used in various camera apps.

2. OBJECTIVE

The **objective of this project:**

- To make benchmarks about dimensionality reduction strategies.
- Mainly PCA (Principal Component Analysis) helps in decreasing over- fitting for recognizing facial key-points.
- Principal component analysis (PCA) is a technique used to emphasize variation and bring out strong patterns in a dataset.

3. INTRODUCTION

The **objective of this project** is to make benchmarks about how much dimensionality reduction strategies main PCA (Principal Component Analysis) helps in decreasing over-fitting for recognizing facial key-points. If the idea of deep learning will be used, such as neural network and cascaded neural network then the results of these structures will be significantly better than state of-the-art methods, like feature extraction and dimension reduction algorithms. This method can help to locate the key points in a given image using deep architectures to not only obtain lower loss for the detection task but also accelerate the training and testing process for real-world applications. Two basic neural network structures can be constructed, one hidden layer neural network and other one the

convolution neural network as our baselines. An approach to better locate the coordinates of facial key points with introduced features other than raw input will be proposed. Specifically, a block of pre - trained Inception Model was used to extract Intermediate features and using different deep structures to compute the final output vector. The experimental results will consist of two parts as well. First, the detection accuracy will be calculated by comparing training, validation and testing loss among all the 5 models illustrated in the selected journal. Second, the time complexity can be evaluated of the 5 models by comparing their time consumption in both training and testing phases. The experiments results will show the effectiveness of deep structures for facial key points detection tasks, and using the pre-trained Inception Model has slightly improved the performance of detection compared to baseline Methods. Another method that can be proposed is using PCA, for this we will use a list of 96×96-pixel 8- bit gray level images with their corresponding (x, y) coordinates of 15 facial key points. First, cross validation will be adopted to randomly split the data set into a training set and a test set, so that we can develop our algorithm on the training set and assess its performance on the test set. The algorithm first performed histogram stretching to enhance the image contrast by stretching the range of pixel intensity of each training image. Then, in order for noise reduction, principal components analysis may be applied on the stretched images to
obtain the eigen faces. Using the resultant eigen faces, the mean patch searching algorithm can be implemented with correlation scoring and mutual information scoring to predict the left- and right-eye centers for any query test facial images. We will get to know the minimum hold-out test MSE for predicting left eye center and the minimum holdout test MSE for predicting right eye center. Thus, the best average test MSE for predicting both facial key points can be calculated.

Description of 15 key points present in dataset with (x,y) representing each coordinates.

'left_eye_center_x', 'left_eye_center_y', 'right_eye_center_x', 'right_eye_center_y',

'left_eye_inner_corner_x', 'left_eye_inner_corner_y', 'left_eye_outer_corner_x',

'left_eye_outer_corner_y', 'right_eye_inner_corner_x', 'right_eye_inner_corner_y',

'right_eye_outer_corner_x', 'right_eye_outer_corner_y', 'left_eyebrow_inner_end_x',

'left_eyebrow_inner_end_y', 'left_eyebrow_outer_end_x',

'left_eyebrow_outer_end_y', 'right_eyebrow_inner_end_x', 'right_eyebrow_inner_end_y', 'right_eyebrow_outer_end_x', 'right_eyebrow_outer_end_y', 'nose_tip_x', 'nose_tip_y', 'mouth_left_corner_x', 'mouth_left_corner_y', 'mouth_right_corner_x', 'mouth_right_corner_y', 'mouth_center_top_lip_x', 'mouth_center_top_lip_y', 'mouth_center_bottom_lip_x', 'mouth_center_bottom_lip_y'

4. LITERATURE REVIEW

Ahonen et al. [1] presents a novel and efficient facial image representation based on local binary pattern (LBP) texture features. The face image is divided into several regions from which the LBP feature distributions are extracted and concatenated into an enhanced feature vector to be used as a face descriptor. The performance of the proposed method is assessed in the face recognition problem under different challenges. Other applications and several extensions are also discussed.

Khadharaoui et al. [2] present an approximation of a 3D face model while keeping in mind the robustness required to change facial expressions and optimization of computation time required. The paper presents a complete solution for the extraction of a 3D model of face and its 3D position from a small number of pairs of landmarks 2D-2D and 3D-3D matching. This extraction step is considered as an initialization method for face tracking based on the 3D model. The solution provided is robust, fast and sufficiently descriptive.

Sujata G Bhele [3] Face recognition has been a fast growing, challenging and interesting area in real time applications. A large number of face recognition algorithms have been developed in last decades. In this paper an attempt is made to review a wide range of methods used for face recognition comprehensively. This include PCA, LDA, ICA, SVM, Gabor wavelet soft computing tool like ANN for recognition and various hybrid combination of this techniques. This review investigates all these methods with parameters that challenges face recognition like illumination, pose variation, facial expressions.

Clinton [4] facial gender classification is an area studied in the Face Recognition Vendor Test (FRVT) Still Facial Images Track. While peripheral to automated face recognition, it has become a growing area of research, with potential use in various applications.

Ahmed M Megreya*[5] Face recognition difficulties during childhood are welldocumented. However, mixed results were obtained regarding the precise nature of these difficulties using different experimental tasks. Using recognition memory paradigm (learning a set of faces followed by an old/new recognition test using the previously studies faces mixed with an equal set of distracters.

Wang and Yang [6] present an efficient approach to achieve fast and accurate eye states detection in still gray level images with unconstrained background. The structure of eye region is used as a robust cue to find eye pair candidates. Eyes are located by eye verification using SVMs. The eye contour information is used to detect whether eyes are open or closed after locating eyes. The proposed method is robust to deal with illumination changes, moderate rotations, glasses wearing and partial face occlusions.

Williams et al. [7] extends the use of statistical learning algorithms for object localization. It has been shown that object recognizers using kernel-SVMs can be elegantly adapted to localization by means of spatial perturbation of the SVM. While this SVM applies to each frame of a video independently of other frames, the benefits of temporal fusion of data are well-known. This is addressed here by using fully probabilistic Relevance Vector Machine (RVM) to generate observations with Gaussian distributions that can be fused over time.

Phillips et al. [8] presents two of the most critical requirements in support of producing reliable face-recognition systems. The Face Recognition Technology program has addressed both issues through the FERET database of facial images and the establishment of the FERET tests. The primary objectives of the test was to assess the state of the art. In addition, the aim was also to identify future areas of research and measure algorithm performance.

Bratley et al. [9] decribes two ways to implement the Niederreiter's sequences on a computer and discusses the results obtained in various practical tests on particular integrals. Low-discrepancy sequences are used for numerical integration, in simulation, and in related applications. Techniques for producing such sequences have been proposed by, among others, Halton, Sobol', Faure, and Niederreiter. Niederreiter's sequences have the best theoretical asymptotic properties.

Yang et al. [10] show that images containing faces are essential to intelligent vision-based human computer interaction, and research efforts in face processing include face recognition, face tracking, pose estimation, and expression recognition. However, many reported methods assume that the faces in an image or an image sequence have been identified and localized. To build fully automated systems that analyze the information contained in face images, robust and efficient face detection algorithms are required. Given a single image, the goal of face detection is to identify all image regions which contain a face regardless of its three-dimensional position, orientation, and lighting conditions. Such a problem is challenging because faces are nonrigid and have a high degree of variability in size, shape, color and texture.

Lu et al. [11] present a face recognition system that utilizes three-dimensional shape information to make the system more robust to arbitrary pose and lighting. For each subject, a 3D face model is constructed by integrating several 2.5D face scans which are captured from different views. 2.5D is a simplified 3D (x, y, z) surface representation that contains at most one depth value (z direction) for every point in the (x, y) plane. Two different modalities provided by the facial scan, namely, shape and texture, are utilized and integrated for face matching. The recognition engine consists of two components, surface matching and appearance-based matching.

Besl and McKay [12] describe a general- purpose, respresentation independent method for the accurate and computationally efficient registration of 3D shapes including free form curves and surfaces. The method handles the full six degrees of freedom and is based on the iterative closest point algorithm. The ICP algorithm always converges monotonically to the nearest local minimum of a mean squared distance metric and experience shows that the rate of convergence is rapid during the first few iterations. Experimental results show the capabilities of registration algorithm on point, sets, curves and surfaces.

Kirby and Sirovich [13] describes the exploitation of natural symmetries in a well- defined family of patterns is discussed within the framework of the Karhunen-L&ve expansion. This results in an extension of the data and imposes even and odd symmetry on the eigen functions of the covariance matrix, without increasing the complexity of the calculation. The resulting approximation of faces projected from outside of the data set onto this optimal basis is improved on average.

IARPA Janus Benchmark A ∗ [14] Rapid progress in unconstrained face recognition has resulted in a saturation in recognition accuracy for current benchmark datasets. The development of accurate and scalable unconstrained face recognition algorithms is a long term goal of the biometrics and computer vision communities. The term “unconstrained” implies a system can perform successful identifications regardless of face image capture presentation (illumination, sensor, compression) or subject conditions (facial pose, expression, occlusion).

Fatima Akhmedova [15] One of the most critical decision points in the design of a face recognition system is the choice of an appropriate face representation. Effective feature descriptors are expected to convey sufficient, invariant and non-redundant facial information. In this work we propose a set of Hahn moments as a new approach for feature description. Hahn moments have been widely used in image analysis due to their invariance, no redundancy and the ability to extract features either globally and locally.

Xiangyu Zhu [16] Pose and expression normalization is a crucial step to recover the canonical view of faces under arbitrary conditions, so as to improve the face recognition performance. An ideal normalization method is desired to be automatic, database independent and high-fidelity, where the face appearance should be preserved with little artifact and information loss. However, most normalization methods fail to satisfy one or more of the goals.

Adam NOWOSIELSKI [17] In holistic face recognition systems face is represented by set of features derived directly from brightness of 2D image pixels. To ensure proper representation of face under variability in head rotation, lighting and expression many templates should be provided. Mechanisms for increasing the efficiency of holistic face recognition systems are considered. By expanding the base structure with additional function blocks, without incorporating intricate feature extraction methods, it is possible to increase the effectiveness of the system.

O.Rama Devi [18] Biometric recognition or biometrics is an automatic recognition system, based on physiological and/or behavioral characteristics of an individual. Biometrics makes it possible to confirm, establish an individual's identity based on "who he/she is", instead of "what he/she possesses" (ID card) or "what he/she remembers" (password). A person's biometric characteristics are unique. Such keys are impossible to copy and reproduce exactly. These are ideal keys theoretically. But using biometric identification creates many specific problems.

T Muni Reddy [19] Students attendance in the classroom is very important task and if taken manually wastes a lot of time. There are many automatic methods available for this purpose, i.e., biometric attendance. All these methods also waste time because students have to make a queue to touch their thumb on the scanning device. This work describes the efficient algorithm that automatically marks the attendance without human intervention based on Embedded Linux. This attendance is recorded by using a camera attached in front of classroom that is continuously capturing images of students, detect the faces in images and compare the detected faces with the database and mark the attendance.

B.Ajanta Reddy [20] face recognition is the one of the robust technology compared to other biometric technologies. It has a lot of applications such as remote sensing, access control, surveillance systems etc, but it is difficult task under difficult lighting conditions which is frequently occurred.

5.1. PRINCIPLE COMPONENT ANALYSIS

PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set. And PCA is sensitive to the relative scaling of the original variables.

Implementation of PCA

Step 1: Normalize the data

First step is to normalize the data that we have so that PCA works properly. This is done by subtracting the respective means from the numbers in the respective column. So if we have two dimensions X and Y, all X become $\boldsymbol{x}$- and all Y become $\boldsymbol{y}$-. This produces a dataset whose mean is zero.

Step 2: Calculate the covariance matrix

Since the dataset we took is 2-dimensional, this will result in a 2x2 Covariance matrix.

$$Matrix(Covariance) = \begin{bmatrix} Var[X_1] & Cov[X_1, X_2] \\ Cov[X_2, X_1] & Var[X_2] \end{bmatrix}$$

Please note that $Var[X_1] = Cov[X_1,X_1]$ and $Var[X_2] = Cov[X_2,X_2]$.

Step 3: Calculate the eigenvalues and eigenvectors

Next step is to calculate the eigenvalues and eigenvectors for the covariance matrix. The same is possible because it is a square matrix. $\boldsymbol{\lambda}$ is an eigenvalue for a matrix $\boldsymbol{A}$ if it is a solution of the characteristic equation:

$$\boldsymbol{det(\lambda I - A) = 0}$$

Where, $\boldsymbol{I}$ is the identity matrix of the same dimension as $\boldsymbol{A}$ which is a required condition for the matrix subtraction as well in this case and '***det'*** is the determinant of the matrix. For each eigenvalue $\boldsymbol{\lambda}$, a corresponding eigen-vector $\boldsymbol{v}$, can be found by solving:

$$\boldsymbol{(\lambda I - A)v = 0}$$

Step 4: Choosing components and forming a feature vector:

We order the eigenvalues from largest to smallest so that it gives us the components in order or significance. Here comes the dimensionality reduction part. If we have a dataset with n variables, then we have the corresponding n eigenvalues and eigenvectors. It turns out that the eigenvector corresponding to the highest eigenvalue is the principal component of the dataset and it is our call as to how many eigenvalues we choose to proceed our analysis with. To reduce the dimensions, we choose the first p eigenvalues and ignore the rest. We do lose out some information in the process, but if the eigenvalues are small, we do not lose much.

Next we form a feature vector which is a matrix of vectors, in our case, the eigenvectors. In fact, only those eigenvectors which we want to proceed with. Since we just have 2 dimensions in the running example, we can either choose the one corresponding to the greater eigenvalue or simply take both.

$Feature\ Vector = (eig_1, eig_2)$

Step 5: Forming Principal Components:

This is the final step where we actually form the principal components using all the math we did till here. For the same, we take the transpose of the feature vector and left-multiply it with the transpose of scaled version of original dataset.

$$NewData = FeatureVector^T \times ScaledData^T$$

Here,

NewData is the Matrix consisting of the principal components,
FeatureVector is the matrix we formed using the eigenvectors we chose to keep, and

ScaledData is the scaled version of original dataset

(‘T’ in the superscript denotes transpose of a matrix which is formed by interchanging the rows to columns and vice versa. In particular, a 2x3 matrix has a transpose of size 3x2)

If we go back to the theory of eigenvalues and eigenvectors, we see that, essentially, eigenvectors provide us with information about the patterns in the data. In particular, in the running example of 2-D set, if we plot the eigenvectors on the scatterplot of data, we find that the principal eigenvector (corresponding to the largest eigenvalue) actually fits well with the data. The other one, being perpendicular to it, does not carry much information and hence, we are at not much loss when deprecating it, hence reducing the dimension.

All the eigenvectors of a matrix are perpendicular to each other. So, in PCA, what we do is represent or transform the original dataset using these orthogonal (perpendicular) eigenvectors instead of representing on normal ***x*** and ***y*** axes. We have now classified our data points as a combination of contributions from both ***x*** and ***y.*** The difference lies when we actually disregard one or many eigenvectors, hence, reducing the dimension of the dataset. Otherwise, in case, we take all the eigenvectors in account, we are just transforming the co-ordinates and hence, not serving the purpose.

5.2. Convolution Neural Network Steps of CNN-

- Step 1: Convolution Operation

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.

- Step 1(b): ReLU Layer

The second part of this step will involve the Rectified Linear Unit or ReLU. We will cover ReLU layers and explore how linearity functions in the context of Convolutional Neural Networks.

Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve your skills.

- Step 2: Pooling

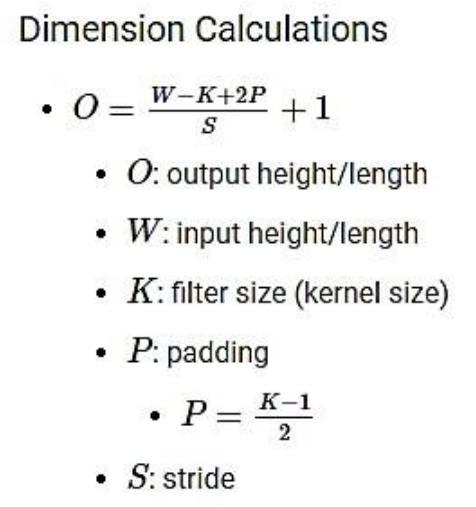
In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

- Step 3: Flattening

This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolutional Neural Networks.

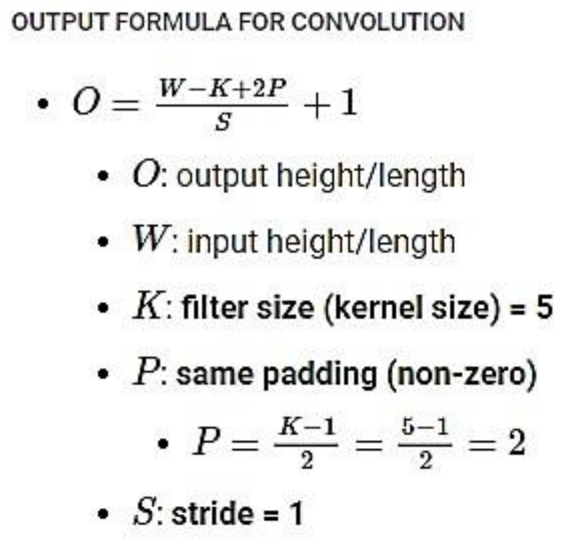
- Step 4: Full Connection

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolutional Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.



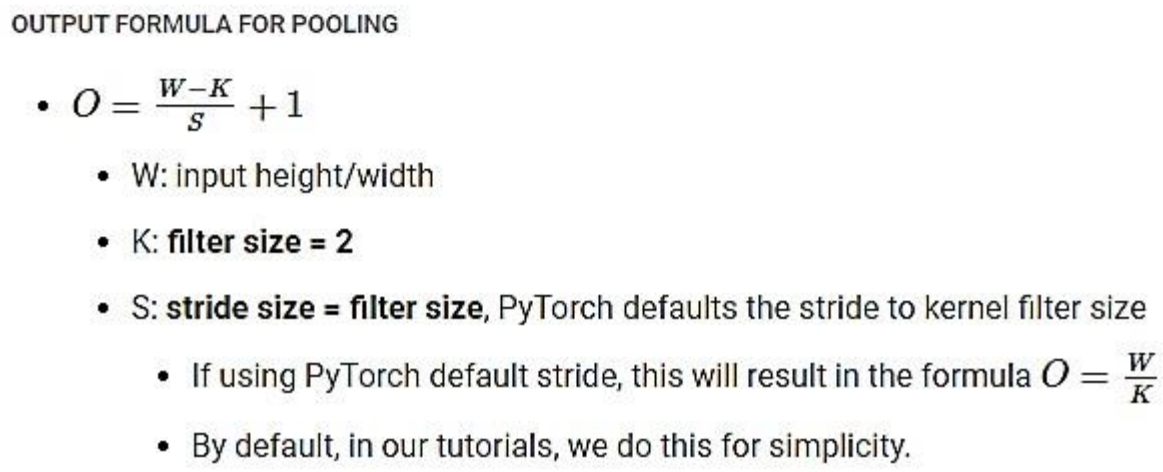
Dimension Calculations

- $O = \frac{W-K+2P}{S} + 1$
  - O: output height/length
  - W: input height/length
  - K: filter size (kernel size)
  - P: padding
    - $P = \frac{K-1}{2}$
  - S: stride



OUTPUT FORMULA FOR CONVOLUTION

- $O = \frac{W-K+2P}{S} + 1$
  - O: output height/length
  - W: input height/length
  - K: **filter size (kernel size) = 5**
  - P: **same padding (non-zero)**
    - $P = \frac{K-1}{2} = \frac{5-1}{2} = 2$
  - S: **stride = 1**



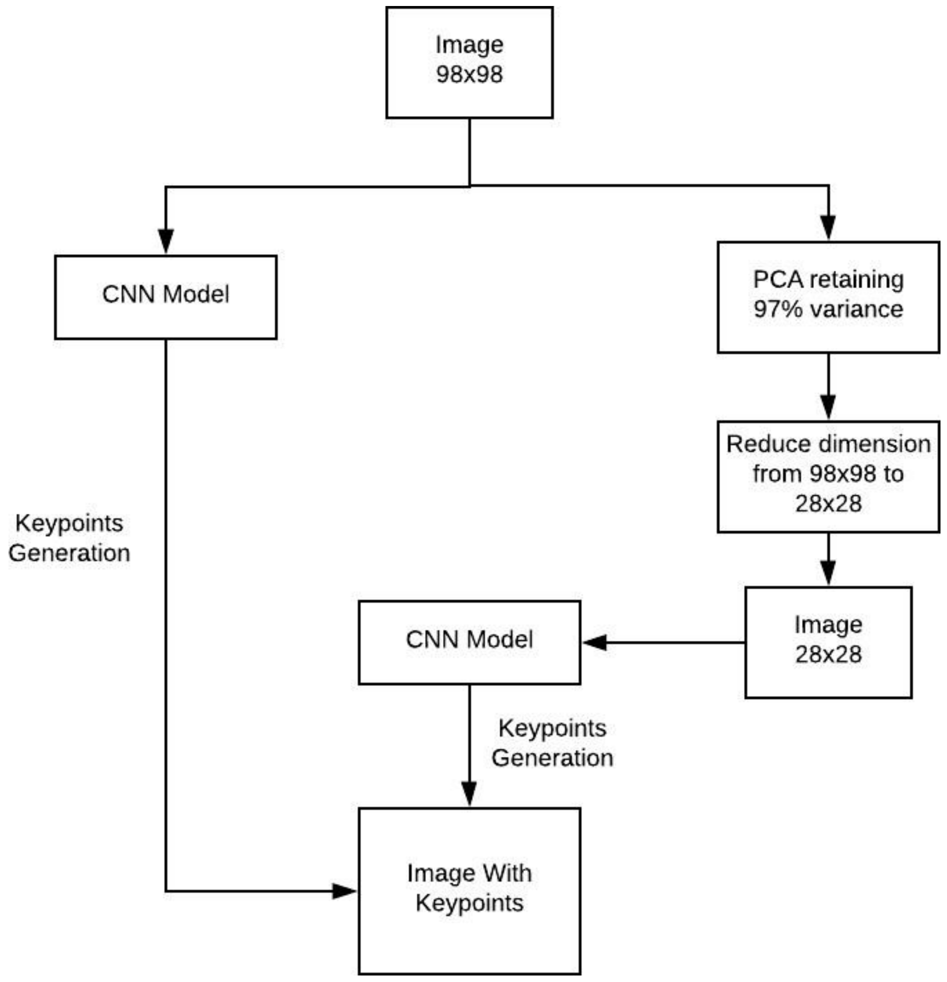
OUTPUT FORMULA FOR POOLING

- $O = \frac{W-K}{S} + 1$
  - W: input height/width
  - K: **filter size = 2**
  - S: **stride size = filter size**, PyTorch defaults the stride to kernel filter size
    - If using PyTorch default stride, this will result in the formula $O = \frac{W}{K}$
    - By default, in our tutorials, we do this for simplicity.

6. PROBLEM STATEMENT

We will begin with convolutional neural network.(CNN) There is no doubt that CNN is the best solution to this problem with such a low RMSE. It has been proved that CNN performs well to image recognition both from theory and practice. The advantages include feature extraction and soft weight sharing, which is the unique properties of CNN. It is these characters that make CNN such a perfect approach. However, the disadvantage still need to be mentioned since this cannot be felt from these paper unless to run our code on computers. Very long time will be taken though the code is run on the server. It raises the problem of tuning because it is time consuming. Thence, we have reasons to believe that better RMSE can be achieved if suitable parameters are applied.

Secondly, Neural network doesn't get the RMSE as well as our expectation. Compared with CNN, neural network itself is not a specific method for image recognition so this would not be surprising. However, we originally expect that neural network may perform better can Regression methods. Now taking a look back, neural network's failure may be attributed to overfitting and tuning. The method of 'Dropout' can avoid overfitting but there is no theoretical guidance to the setting of parameters. Beside its high RMSE compared with CNN, it is still time consuming to train neural network. Overall, we do not recommend neural network in face recognition experiment unless perfect parameters can be found. These perfect parameters can be found using PCA when retaining somewhat from 95-97% variance. So, the main objective of this paper is to check whether PCA is suitable for improving CNN model or not and when we get a optimized model, we using the predicted keypoints on a person's face create a selfie filter something like sunglasses or fake moustache to display the use of the facial keypoints.



Architecture diagram

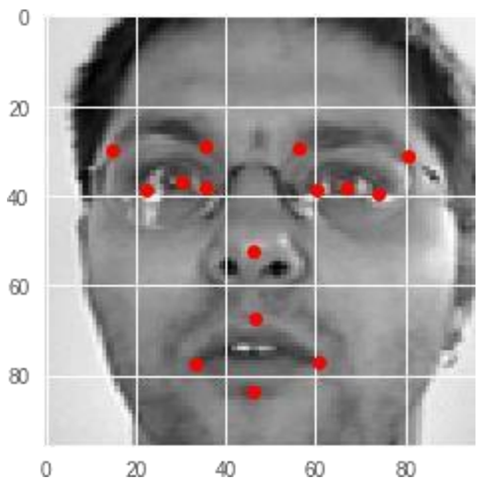
7. PROPOSED METHODOLOGY

The principal focus of this project is to make benchmarks about how much dimensionality reduction strategies main PCA (Principal Component Analysis) helps in decreasing overfitting for recognizing facial key-points.

The method that we propose is using set of trainable image data with their facial keypoints in the form of (x,y) coordinates and preprocess the data using dimensionality reduction techniques mainly PCA(Principal Component Analysis) to reduce the noise of data and decreasing the dimension of each image thereby still retaining maximum possible variance around 95-98% . Then passing these preprocessed images to a Convolution neural network model and check how the training and inference differs in PCA preprocessed images and non pre-processed images.

Possible results could be that the training time of model will decrease considerably and may give better results on test dataset. Since dimensions of an image is decreased , it will also increase the inference speed .

The data has been collected from one of Kaggle's popular dataset on facial keypoints https://www.kaggle.com/c/facial-keypoints-detection/data . The dataset contains around 7000 images each 98x98 dimensions and 15 keypoint for each image in the form of (x,y) coordinates. The sample data is shown below.



8. PLATFORM

In this project, the main language we use is Python. In addition, we use the following packages to achieve our algorithm:

Front-End: Python

Back-End: Databse - https://www.kaggle.com/c/facial-keypoints-detection/data .

Along with that the following convolution neural network models were trained on Google collaboratory which provides Jupyter Notebook like interface and kernels connected in backend to Google cloud Platform's ML engines.

9. CODE

I. Creating Model def
```
create_model_1(inpt=28):
  model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(5, 5), padding='Same',
  activation='relu', input_shape=(inpt,
inpt, 1)))
  model.add(Conv2D(filters=32, kernel_size=(5, 5), padding='Same',
                                activation='relu'))
  model.add(MaxPooling2D(pool_size=(2, 2)))
  model.add(Dropout(0.25))
model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='Same',
  activation='relu'))
  model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='Same',
                                activation='relu'))
  model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
  model.add(Dropout(0.25))
model.add(Flatten())
  model.add(Dense(256, activation="relu"))
  model.add(Dropout(0.5))
  model.add(Dense(30, activation="linear"))
return model
```

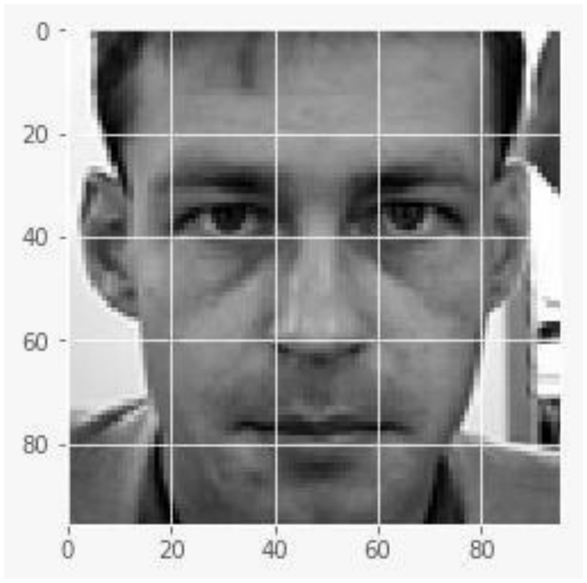
II. Training Model 1 epochs =
```
100
history = model.fit(train_images, train_keypoints,
validation_split=0.2, shuffle=True, epochs=epochs,
batch_size=20)
```
III - Visualization def visualize(history):
```
plt.plot(history.history['mean_absolute_error'])
plt.plot(history.history['val_mean_absolute_error'])
plt.title('model mean absolute error')
plt.ylabel('mae') plt.xlabel('epoch')
  plt.legend(['train', 'test'], loc='upper left') plt.show()
  # summarize history for loss
  plt.plot(history.history['loss'])
  plt.plot(history.history['val_loss'])
  plt.title('model loss')
  plt.ylabel('loss') plt.xlabel('epoch')
  plt.legend(['train', 'test'], loc='upper left') plt.show()
visualize(history)
```

IV - Applying PCA
```
from sklearn.decomposition import PCA h,

w = 28, 28

pca = PCA(n_components=h*w, whiten=True, svd_solver='auto')
```

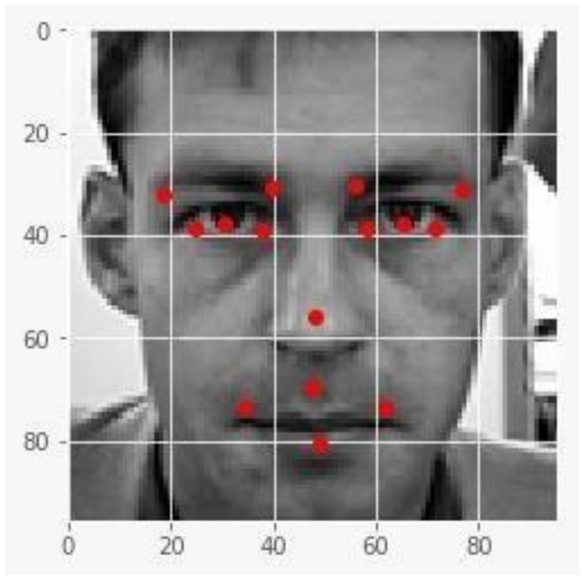
```
%time pca_train_images = pca.fit_transform(reformed_imgs)
CPU times: user 18.2 s, sys: 3.29 s, total: 21.4 s Wall
time: 11.6 s
print("variance/ information
retrieved
",pca.explained_variance_ratio_.cumsum()[-1]*100)4
variance/ information retrieved 99.74417686462402
V - Defining model 2 and
  training model1 =
  create_model(inpt = 16)
  model1.summary()
epochs =
500
history1 = model1.fit(pca_train_images, train_keypoints,
                validation_split=0.2, shuffle=True, epochs=epochs,
                batch_size=64)
```

10. TEST CASES

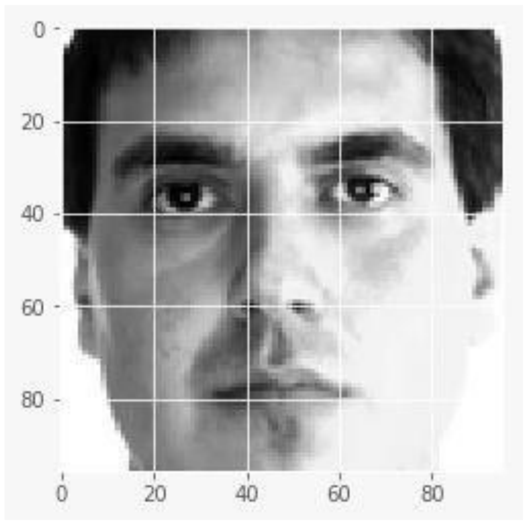
Test case input 1:



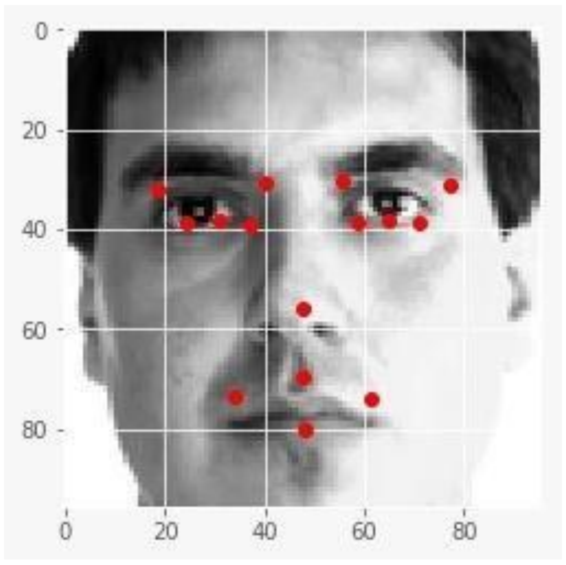
Output :



Test case input 2:



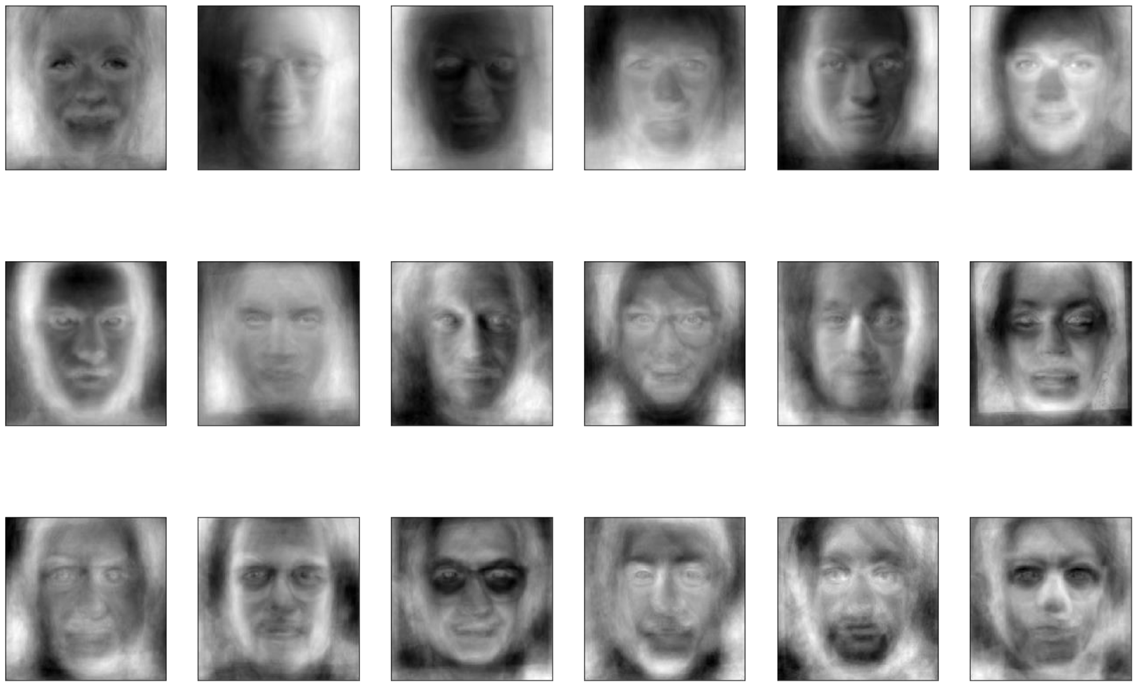
Output:



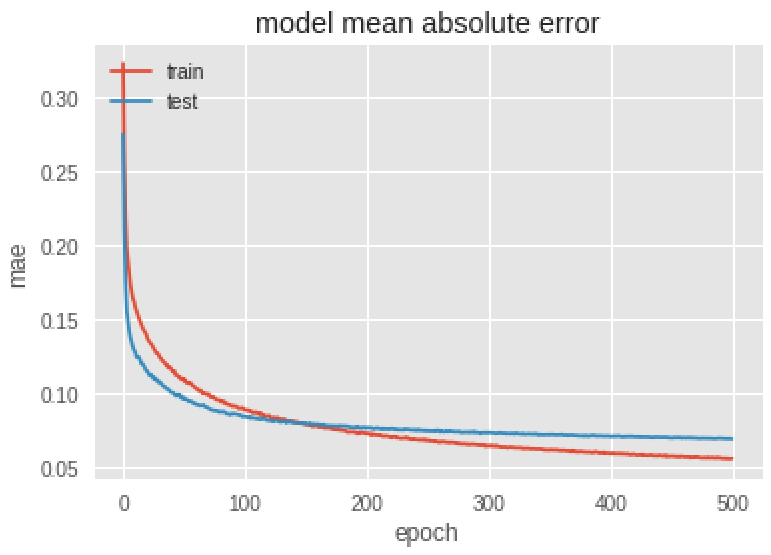
- **RESULT AND DISCUSSION**

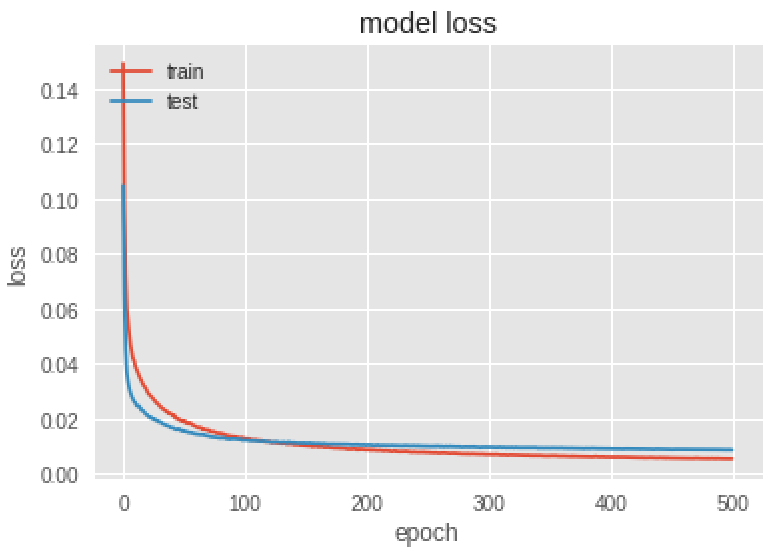
The transformation of PCA is to project the original sample data (n-dimensions) into a k- dimensional(k < n) orthogonal coordinate system, discarding the information in other dimensions. In this way, PCA can extract the main influencing factors, revealing the properties of the data. What's more, in fact, the variables in image data have a quite strong internal correlation. For instance, eyes symmetrically distribute on both sides of the nose. However, in our models, we ignore this correlation subconsciously, which makes the RMSE not very good. But when we adopt PCA, we can remove the correlation among different variables because the project space is an orthogonal

coordinate system. And the running results also strongly support this conclusion. By the way, for PCA is a dimension-reduction algorithm, it will discard part of the information inevitably, but it doesn't hurt the important essentials. Because we retain 97% of the information in practice. We originally didn't expect PCA to behave well, instead, we just want to reduce the time for running our code. Surprisingly, PCA reduces all the
RMSE without exception. After the gun, we may conclude that 'Curse of Dimensionality' is a serious problem in this experiment. PCA has the ability to overcome over fitting and balance bias- Variance. Eigen faces of the dataset looks like below.



But after dimensionality reduction of all the images in dataset and slight change in the model structure to fit the model dimensions , the learning curve shows improvements.





This proves the fact that trying to attain maximum variance using PCA helps the model train faster and better

10. CONCLUSION

Detect facial keypoints is a critical element in face recognition. We have used two methods for facial keypoints detection. From the results, we can say that PCA is a dimensionreduction algorithm, the effect of this method on the data is hard to predict. However, it greatly reduces the dimensions and cut down the consuming time. In this experiment, PCA receives a excellent result. Especially for neural network, this proves our suppose of overfitting, to some degree. Overall, PCA optimized CNN model gives better result than the normal CNN model when used for facial keypoints detection.

11. FUTURE SCOPE

In this paper, we have focused on the task of detecting facial keypoints when given raw facial images. Specifically, for a given 96 ∗ 96 image, we would predict 15 sets of (x, y) coordinates for facial keypoints. Two traditional deep structures, One Hidden Layer Neural Network and Convolutional Neural Network, are implemented as our baselines. We further explored some pre-processing method to reduce computational complexity to fit the requirements for detecting facial keypoints. Experiments which conducted on real- world kaggle dataset have shown the effectiveness of deep structures, especially LexNet Model. As for our future work, we can explore from these few aspects. We have already shown the effectiveness of the pre-processed LexNet Model when used as the pretrained model, but the performance has a chance to improve if we train from the scratch. As we can see from the results, using deep structures can increase time complexity when compared to other start-of-art methods but the results have been shown to improve a lot much. What we can do in the future is to design a deep structure specifically for this task to further improve the performance. Different resolution can greatly affect the results of the facial key points detection, thus what we can do is try to reduce the resolution of our given raw images to see the variance of the performance to further evaluate our mod

12. REFERENCES

[1] Ahonen Timo, Abdenour Hadid and Matti Pietikäinen (2006). Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12) 2037–2041

[2] Khadharaoui T., F. Benzarti and H. Amiri (2013). Approximation of 3D Face Model. *International Review on Computers and Software*, 8(3) 810-815

[3] Wang Qiong and Jingyu Yang (2006). Eye Location and Eye State Detection in Facial Images with Unconstrained Background. *Journal of Information and Computing Science*, 1(5) 284-289

[4] Williams Oliver, Andrew Blake and Roberto Cipolla (2005). Sparse Bayesian Learning for Efficient Visual Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8) 1292-1304

[5] Phillips P. Jonathon, Hyeonjoon Moon, Syed A. Rizvi and Patrick J. Rauss (2000). The FERET Evaluation Methodology for Face- Recognition Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10) 1090-1104

[6] Bratley Paul, Bennett L. Fox and Harald Niederreiter (1992). Implementation and Tests of Low-Discrepancy Sequences. *ACM Transactions on Modeling and Computer Simulation*, 2(3) 195-213

[7] Yang Ming-Hsuan, David J. Kriegman and Narendra Ahuja (2002). Detecting Faces in Images: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1) 34-58

[8] Lu Xiaoguang, Anil K. Jain and Dirk Colbry (2006). Matching 2.5D Face Scans to 3D Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1) 31-43

[9] Besl Paul J. and Neil D. McKay (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2) 239-256

[10] Kirby M. and L. Sirovich (1990). Application of the Karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1) 103-108

[11] 1. Pearson K. 1901. On lines and planes of closest fit to systems of points in space. *Phil. Mag.* 2, 559–572. (10.1080/14786440109462720)

[12] 2. Hotelling H. 1933. Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* 24, 417–441, 498–520 (10.1037/h0071325)

[13] 3. Jackson JE. 1991. *A user's guide to principal components*. New York, NY: Wiley.

[14] 4. Jolliffe IT. 2002. *Principal component analysis*, 2nd edn New York, NY: SpringerVerlag.

[15] 5. Diamantaras KI, Kung SY. 1996. *Principal component neural networks: theory and applications*. New York, NY: Wiley.

[16] 6. Flury B. 1988. *Common principal components and related models*. New York, NY: Wiley.

[17] 7. Horn R, Johnson C. 1985. *Matrix analysis*. Cambridge, UK: Cambridge University Press.

[18] 8. Hudlet R, Johnson RA. 1982. An extension of some optimal properties of principal components. *Ann. Inst. Statist. Math.* 34, 105–110. (10.1007/BF02481011)

[19] 9. Okamoto M. 1969. Optimality of principal components. In *Multivariate analysis II* (ed. PR Krishnaiah), pp. 673–685. New York, NY: Academic Press.

[20] 10. McCabe GP. 1984. Principal variables. *Technometrics* 26, 137–144. (10.1080/00401706.1984.10487939)