IOP Conf. Series: Materials Science and Engineering **308** (2018) 012013 doi:10.1088/1757-899X/308/1/012013

Human face recognition using eigenface in cloud computing environment

S T M Siregar, M F Syahputra, R F Rahmat

Department of Information Technology, Faculty of Computer Science and Information Technology, University of Sumatera Utara, Medan, Indonesia

Email: tr.sintong@students.usu.ac.id |fadlysyah@gmail.com |romi.fadillah@usu.ac.id

Abstract. Doing a face recognition for one single face does not take a long time to process, but if we implement attendance system or security system on companies that have many faces to be recognized, it will take a long time. Cloud computing is a computing service that is done not on a local device, but on an internet connected to a data center infrastructure. The system of cloud computing also provides a scalability solution where cloud computing can increase the resources needed when doing larger data processing. This research is done by applying eigenface while collecting data as training data is also done by using REST concept to provide resource, then server can process the data according to existing stages. After doing research and development of this application, it can be concluded by implementing Eigenface, recognizing face by applying REST concept as endpoint in giving or receiving related information to be used as a resource in doing model formation to do face recognition.

1. Introduction

Face recognition system has become one of the research topics in image processing and computer vision applied in various systems, starting from attendance systems, security systems such as cctv, identity recognition, emotional recognition, granting system permissions and also applied to the robotics field as a way for the robot to recognize someone.

Face detection itself is a pre-processing stage in facial recognition. In doing face recognition of a person, it will take a lot of face samples with various expressions, angle of retrieval and general differences attached to the face like glasses and moustache [1]. To do face recognition of one face does not take a long time, but if we do it on the attendance system or security system on companies that have many faces to be recognized, it will take a long time. In addition, in some cases, there are deficiencies in some of the previous studies, such as face recognition not running in real time, using public datasets in testing, and computing processes that take a long time causing the computer cannot be used during the process.

To solve those problems, several studies have using cloud computing technology. Cloud computing is a computing service that is done not on a local device, but on an internet connected to a data center infrastructure. The system of cloud computing also provides a scalability solution where cloud computing can increase the resources needed when doing larger data processing. Because computing is not done on a local device, then it is done between client communication with cloud server by utilizing web-service communication. Generally using REST-API or SOAP. Because it does not use local computing capabilities, then cloud computing is mostly applied to mobile devices that have the limitation of computing on different devices. In addition, with cloud computing, eliminating the limits

of programming language that must be in doing face recognition. This is because the web service provides a response in the form of XML files or JSON.

In performing face recognition, it needs to perform several stages starting from face detection, feature extraction then doing face recognition. During such stages require adequate and available hardware resources at all times. If you want to do the local device, it will take a long time and besides, the device cannot be used for some time. Therefore, an approach is required to be able to perform face recognition without using the capabilities of local computing devices. This research proposed the implementation of Eigenface algorithm in performing face recognition that will be applied in the cloud computing environment. The contribution is we can perform face recognition in cloud computing requirement, with limited resources, and good image processing strategies.

Some researches have been done in the area of cloud computing which related to face recognition, one of the research is utilizing facial recognition service belongs to one of 'face.com', an online face recognition website and using 'facebook.com' as identity provider by utilizing Graph API owned by facebook.com [2]. A research in the robotics field also takes advantage of cloud computing as a computing medium for the Robot Operating System to be able to recognize humans [3]. Further research is done by developing a real time attendance system by utilizing local infrastructure network. Although it is not applying cloud computing, but research has applied the concept of client-server, in addition in this study only conducted trials on ten subjects only. [4]. In addition, a larger study was also conducted. The study aims to perform face recognition on large-scale face databases using the Extreme Learning method [5]. Furthermore, a research on face recognition by using cloud computing by improving the data security aspect that found in public-cloud [6].

2. Methodology

This research aims to provide face recognition service using API implemented in Cloud Computing environment with Software-as-a-Service (SaaS) model. To perform face recognition Eigenface method will be applied and applying the concept of REST API as a link between client and server.

Several assumptions have been made to limit the scope of the discussion in this study, namely:

1. Using front face only.
2. Using Face with no expression and smile.
3. Using photos of at least 15 people.
4. Not paying attention to lighting conditions or distance taking image.
5. The use of REST API does not have a limit on the number of uses.
6. The image used has a size of 320 x 240 pixels.

The general architecture of the system to be built can be seen in Figure 1.

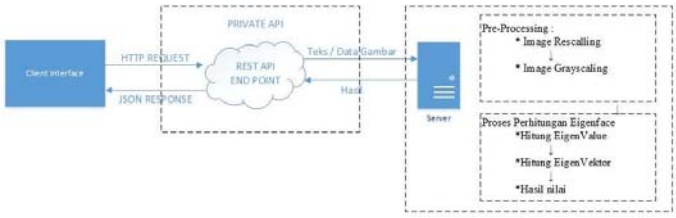

Figure 1. General Architecture

In this study, the data used in the form of taken images using a web camera. Several things to be considered when the data is acquired, they are:

A. Distance of image shooting
B. Image resolution
C. Image format

In doing this research, we use an external web camera that can produce images up to 1920 x 1080 pixels resolution. In addition, the resulting image is expected to have a resolution of 320 x 240 pixels with the format jpg / jpeg / png / gif.

In addition, in this study HTTP Request data and HTTP Response data has been used. Both of the datas play a role as a communication medium that occurs between the client and server. In this research, HTTP Request will be delivered with the following specifications:

A. Using the POST method
B. Multipart / form-data as encode type / content type

For more details on the HTTP Request structure can be seen in Figure 2.

```
POST regis.dev/upload HTTP/1.1
Host: 202.0.107.144
Content-Type: multipart/form-data; boundary=--------------------------
98215567712697562581705625561
Content-Length: 562

----------------------------98215567712697562581705625561
Content-Disposition: form-data; name="file"; filename="101402042_1.jpg "
Content-Type: multipart/form-data
```

Figure 2. HTTP Request Structure

2.1. Haar Feature

Haar feature extraction has take place in the system because the processing time of features is more quickly than image processing per pixel. The search for human face objects is done by searching for some features that have a high degree of differentiator. This is done by evaluating each feature of the training data by using the value of the feature. In this study authors use haarcascade_frontalface_default.xml specially to the detection of the human head from the front view.

2.2. Resizing Process

Resize process is one of the stages in pre-process which is by doing uniformity of each image data that will be used so they have the same size [8]. In this research, the image will be resized to 320 x 240 pixels.

2.3. Grayscale Process

The Grayscale process is performed to obtain similiar image data. Such similiarity is achieved by converting the RGB image into a gray image [7]. Here is a pseudocode to do the Grayscale process while the result of the grayscale process can be seen in Figure 3.

```
FOREACH pixel p IN image :
    luminosity = (p.red + p.green + p.blue) / 3
    p.red   = luminosity
    p.green = luminosity
    p.blue  = luminosity
```

Figure 3. Grayscale process

2.4. Principal Component Analysis

The PCA or Principal Component Analysis process has the following steps:

1. Initialize the face image of all existing face data.
2. Changing into the covariant matrix of any existing face image data.

$$\begin{bmatrix}0&4&3\\1&4&2\\0&0&0\end{bmatrix} \qquad \begin{bmatrix}2&2&1\\3&2&4\\0&0&0\end{bmatrix} \qquad \begin{bmatrix}1&4&2\\2&3&0\\0&0&0\end{bmatrix}$$

Citra A Citra B Citra C

Figure 4. Covariance Matrix of All Faces Image Data

3. Calculate mean value of the covariance matrix.

$$\Psi = \frac{1}{3}\sum_{n=1}^{3}\Gamma_n = \frac{1}{3}\left(\begin{bmatrix}0&4&3\\1&4&2\\0&0&0\end{bmatrix}+\begin{bmatrix}2&2&1\\3&2&4\\0&0&0\end{bmatrix}+\begin{bmatrix}1&4&2\\2&3&0\\0&0&0\end{bmatrix}\right)$$

$$\Psi = \frac{1}{3}\begin{bmatrix}1&3&2\\2&3&2\\0&0&0\end{bmatrix} \tag{1}$$

4. Reduced the covariance matrix against mean to get adjusted data or normal data.

$$\Phi_1 = \Gamma_1 - \Psi = \begin{bmatrix}0&4&3\\1&4&2\\0&0&0\end{bmatrix} - \begin{bmatrix}1&3&2\\2&3&2\\0&0&0\end{bmatrix} = \begin{bmatrix}0&1&1\\0&1&0\\0&0&0\end{bmatrix}$$
$$\Phi_2 = \Gamma_2 - \Psi = \begin{bmatrix}2&2&1\\3&2&4\\0&0&0\end{bmatrix} - \begin{bmatrix}1&3&2\\2&3&2\\0&0&0\end{bmatrix} = \begin{bmatrix}1&0&0\\1&0&1\\0&0&0\end{bmatrix}$$
$$\Phi_3 = \Gamma_3 - \Psi = \begin{bmatrix}1&4&2\\2&3&0\\0&0&0\end{bmatrix} - \begin{bmatrix}1&3&2\\2&3&2\\0&0&0\end{bmatrix} = \begin{bmatrix}0&1&0\\0&0&0\\0&0&0\end{bmatrix} \tag{2}$$

$$C = \frac{1}{M}\sum_{n=1}^{M}\Phi_n\Phi_n^T$$

$$C = AA^T \qquad A = [\Phi_1, \Phi_2, \ldots, \Phi_M]$$

$$L = AA^T, \text{ dimana } L_{m,n} = \Phi_n\Phi_m^T$$

$$L = \begin{pmatrix} \begin{bmatrix}0&1&1&1&0&0&0&1&0\\0&1&0&1&0&1&0&0&0\\0&0&0&0&0&0&0&0&0\end{bmatrix} \begin{bmatrix}0&0&0\\1&1&0\\1&0&0\\1&1&1\\0&0&0\\0&1&0\\0&0&0\\1&0&0\\0&0&0\end{bmatrix} \end{pmatrix} \tag{3}$$

$$L = \begin{bmatrix} 4 & 2 & 1 \\ 2 & 3 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (4)$$

a. After that, the obtained eigenvalue decomposition of data adjust,
b. After that, get eigenvektor of eigen value decomposition.
c. Sorting Eigenvectors.
d. The final data is obtained as the final result of the eigenvector adjusted data.

3. Result and Discussion

In the face image collection testing is done to show that the application can take pictures with web camera and make the process pengunggahan eight photos into the server with the right filename format after the API rules that have been set. Uploaded towards endpoint "/ upload". When the face image collection page is accessed it will look like in Figure 5 To perform a face image capture, users are required to enter a student ID (NIM) in the textbox section as shown in Figure 5.

Figure 5. Application view

In performing face image capture, an image shooting with expression criteria as can be seen in Table 1.

Table 1 Facial Expression List

Step	Expression type
1	Normal
2	Smile
3	Sad
4	Upset / Angry
5	Normal with Glasses
6	Smile with Glasses
7	Sad with Glasses
8	Upset / Angry with Glasses

Every time a user uploads an image, the application will send the image to the server, and before going to the next stage or table. Any image uploaded to the server will be saved in NIM_NoUrutFoto.jpg

format, if the user is using an existing student ID (NIM) then the old image will be replaced with a newer image. Here are some images that have been successfully uploaded into the server that can be seen in Figure 6.

Figure 6. Some of the uploaded images

Because of the process is by utilizing the Cloud Computing concept with the Software-as-a-Service model, the facial recognition process is performed on the server side, where the face recognition application is only an interface application that does not perform the logical process of recognition using the Eigenface method. Therefore, the server as a service provider will provide results from the existing process. The face recognition results are resubmitted by the server in JSON format, where the server will only tell whether the NIM of the user has a match to the face contained in the dataset.

In addition, because the response given by the server is in the JSON format, it is necessary to have a mediation by the face recognition application to read out the response and make it Human-Readable by taking the object from JSON and parsing the information for use in the interface. The Interface Display means is the status bar that will display the information object from JSON and display it into status. The results of facial recognition can be seen in Figure 7.

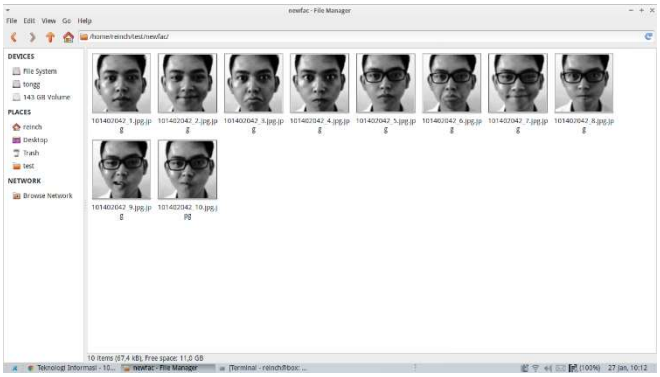

Figure 7. Face Recognition Result

In this study, an eigen image is produced which is a composite of each eigen image of the image used in the introduction. The combined image of the eigen can be seen in figure 8.

Figure 8. Imagery of the collected Eigenface

In this test, the server is hosted on the local domain which is regis.dev:2527, where the domain and port used can be set in the configuration file, so if it used on a server with a domain or Public IP and a particular port, it can be configured easily without having to configure networks on the server.

In this research, there are two endpoints that can be used as route or address in API usage. The first endpoint is "/ upload" and the second endpoint is "/ introduction". API built by using Python programming language with the help of Flask web-framework. As for the interface of facial image collection or facial recognition page using only HTML and Javascript. The impact of using the web-framework Flask resulted in the endpoint in access from the browser, the server will receive a request in the form of "GET" method while in the API development, the request received is only a request with the "POST" method. This also applies to the "/" endpoint, where the server will return the Response by rendering the index page, but since this study does not require an index page, so also if accessing the endpoint of the browser then the server will respond by rendering the index page stored in Directory upload or directory introduction. To overcome this, the endpoint API can only be used if the method used is a POST, where in most cases the POST method is only used in the data transmission of the form, this is because the GET method is the default method sent to be used by the browser. Therefore, testing of the Request received by the API, and also against the Response provided by the API.

4. Conclusion

It can be concluded that by implementing Eigenface we can apply REST concept in face recognition process by giving or receiving related information to be used as resource in forming model to do face recognition. Another conclusion is pre-processing phase must be done to get a similar image source. Furthermore, in this research has the possibility of development by doing uniformity of photo source, whether in the form of background, controlled light source. It can also take advantage of information security or data fields as a development of authentication in API usage.

To improve service performance of facial recognition system, further research can be done by applying scheduling or applying Job Management which can perform thread control that work to do introduction if the introduction process done by several client at once.

References

[1] Zhang, X., & Gao, Y. (2009). Face Recognition Across Pose: a Review. Pattern Recognition, 42(11), 2876-2896.Chandra, K. & Singh, S. 2014. Firefly algorithm to solve two dimensional bin packing problem. *International Journal of Computer Science and Information Technologies* 5(4): 5368-5373.

[2] Sari, R.F. & Indrawan, P. 2013. Cloud Computing Services in Mobile Devices Using Android Face Detector API and Rest Communication, p. 529-537, The Third International Conference on Digital Information Processing and Communications (ICDIPC 2013), Feb 2013.

[3] Bistry, H., & Zhang, J. 2010. A cloud computing approach to complex robot vision tasks using smart camera systems. In Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on (pp. 3195-3200). IEEE.

[4] Putra, I. N. T. A. 2014. Perancangan dan Pengembangan Sistem Absensi Realtime Melalui Metode Pengenalan Wajah. Jurnal Sains dan Teknologi, 3(2).

[5] Vinay, A., Shekhar, V. S., Rituparna, J., Aggrawal, T., Murthy, K. B., & Natarajan, S. 2015. Cloud Based Big Data Analytics Framework for Face Recognition in Social Networks Using Machine Learning. Procedia Computer Science, 50, 623-630.

[6] Haghiat, M., Zonouz, S., & Abdel-Mottaleb, M. 2015. CloudID: Trustworthy cloud-based and cross-enterprise biometric identification. Expert Systems with Applications, 42(21), 7905-7916.

[7] R. F. Rahmat, T. Chairunnisa, D. Gunawan, and O. S. Sitompul, "Skin color segmentation using multi-color space threshold," in *2016 3rd International Conference on Computer and Information Sciences, ICCOINS 2016 - Proceedings*, 2016, pp. 391–396.

[8] Seniman, D. Arisandi, R. F. Rahmat, William and E. B. Nababan, "Chinese chess character

recognition using Direction Feature Extraction and backpropagation," *2016 International Conference on Data and Software Engineering (ICoDSE)*, Denpasar, Indonesia, 2016, pp. 1-6.