ISSN 2356-4407

www.STIKI.ac.id

Bersama memajukan ICT Indonesia

PROCEEDING

IC - ITECHS 2014

The 1st International Conference on Information Technology and Security

Malang, November 27, 2014

Published by:

Lembaga Penelitian dan Pengabdian pada Masyarakat

Sekolah Tinggi Informatika dan Komputer Indonesia

PROCEEDING

The 1st International Conference on Information Technology and Security (IC-ITechs)

November 27, 2014

Editors & Reviewers:

Tri Y. Evelina, SE, MM Daniel
Rudiaman, S.T, M.Kom Jozua
F. Palandi, M.Kom

Layout Editor:

Eka Widya Sari

LEMBAGA PENELITIAN & PENGABDIAN KEPADA MASYARAKAT
Sekolah Tinggi Informatika & Komputer Indonesia (STIKI) – Malang
Website: itechs.stiki.ac.id E-mail: itechs@stiki.ac.id

PROCEEDING

The 1st International Conference on Information Technology and Security (IC-ITechs) November 27, 2014

ISSN 2356 - 4407

viii + 276 hlm; 21 X 29,7 cm

Reviewers & Editors:

Tri Y. Evelina, SE, MM
Daniel Rudiaman, S.T, M.Kom
Jozua F. Palandi, M.Kom

Layout Editor:

Eka Widya Sari

Published by:

LEMBAGA PENELITIAN & PENGABDIAN KEPADA MASYARAKAT

Sekolah Tinggi Informatika & Komputer Indonesia (STIKI) – Malang

Jl. Raya Tidar 100 Malang 65146, Tel. +62-341 560823, Fax. +62-341 562525

Website: itechs.stiki.ac.id E-mail: itechs@stiki.ac.id

GREETINGS

Head of Committee IC-Itechs

For all delegation participants and invited guest, welcome to International Conference on Information Technology and Security (IC-Itechs) 2014 in Malang, Indonesia.

This conference is part of the framework of ICT development and security system that became one of the activities in STIKI and STTAR. this forum resulted in some references on the application of ICT. This activity is related to the movement of ICT development for Indonesia.

IC-Itechs aims to be a forum for communication between researchers, activists, system developers, industrial players and all communications ICT Indonesia and abroad.

The forum is expected to continue to be held continuously and periodically, so we hope this conference give real contribution and direct impact for ICT development.

Finally, we would like to say thanks for all participant and event organizer who involved in the held of the IC-Itechs 2014. We hope all participant and keynote speakers got benefit from this conference.

LIST OF CONTENT

Multiple And Single Haar Classifier For Face Recognition

Go Frendi Gunawan, Subari
Sekolah Tinggi Informatika dan Komputer Indonesia (STIKI)
gofrendiasgard@gmail.com, subari@stiki.ac.id

Abstract

Haar-Classifier is a well-known tool for face detection in an image. Open-CV library already provide a class for Haar-Classifier implementation. However, one significant problem in face detection is tilt-face-pose recognition. In this paper, we will discuss about two different solutions to overcome the problem. The solutions are written in Python which has a very simple syntax. The first solution is using multiple classifier. Each of them is trained to recognize face with different rotation-degrees. The second solution is using single classifier to recognize many face image which have been rotated by different angles.

***Keywords** : haar-classifier, face-recognition, python-opencv*

1. INTRODUCTION

Haar Classifier is a well-known method for face detection. Rather than using complex calculation, Haar-Classifier uses several weak classifiers combined to produce the decision.

Haar Classifier is also implemented as a part of Open-CV library, a popular open source computer-vision library written in C. Open-CV also provide several XML files. Each of them is used to recognize different objects such as face, eyes, and mouth.

Although the XML files are pretty convenient, they don't cover rotated-object recognition. To recognize rotated-object, one need to train their own Haar-Classifier to produce suitable XML file. The process is a bit complicated and takes a long time.

In this paper, we evaluate two different approach to accomplish rotated-object recognition. The first one is by rotating the image into several different angle, and let the original Haar classifier do the recognition task by using the original XML file. The second one is by creating different XML files based on the original one. The new XML files are then used byseveral new classifiers to recognize object in a single image.

2. RESEARCH METHOD

HAAR Classifier

Haar Classifier has been initially proposed by Paul Viola [1] and improved by Rainer Lienhart [2].

The classifier is trained by using a several positive and negative images. Positive images are images contain recognition object, while negative images are images do not contain recognition object.

Haar Classifier consists of of several simpler classifiers that are applied subsequently to a region of interest. The decision whether the area contains recognition object or not is taken by taking conclusion of simpler classifiers's decision. However, if the several first weak-classifier decide that the area doesn't contain recognition object, the area will be passed and not examined further. By using such a mechanism, the computation will be faster since negative areas are briefly ignored.

Each simple-classifier do their job by looking at specific Haar-feature. Haar-feature is a region contains of 2 rectangle areas. The classifier works by comparing the sum of pixel intensity between those areas.

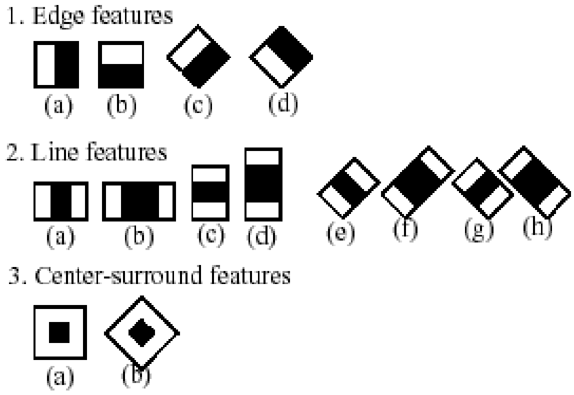

Figure 1. Haar Features

Since calculating the sum of pixel intensity in an area is quiet exhaustive task, We usually use integral image to do the task. Integral image is an imaginary copy of an image which all of the pixel value replaced by the sum of every pixel on the top-left of the pixel. See figure 2 for more explanation. By using integral image we can use a much simpler calculation to obtain the sum of the pixel in the area.

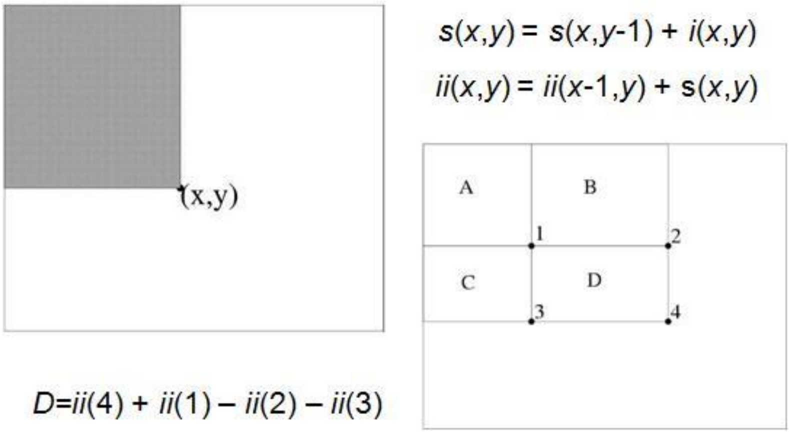

Figure 2. Integral Image

XML File Structure

Open-CV Haar classifier uses a specific XML format. For example, the xml file used for frontal face detection (haarcascade_frontalface_alt.xml) has structure as described in figure 3.

```
<haarcascade_frontalface_alt type_id="opencv-haar-classifier">
  <size>20 20</size>
  <stages>
    <_>
      <!-- stage 0 -->
      <trees>
        <_>
          <!-- tree 0 -->
```

```
    <_>
     <!-- root node -->
     <feature>
       <rects>
         <_>3 7 14 4 -1.</_>
         <_>3 9 14 2 2.</_></rects>
       <tilted>0</tilted></feature>
     <threshold>4.0141958743333817e-003</threshold>
     <left_val>0.0337941907346249</left_val>
     <right_val>0.8378106951713562</right_val></_></_>
  <_>
    <!-- tree 1 -->
    <_>
     <!-- root node -->
     <feature>
       <rects>
         <_>1 2 18 4 -1.</_>
         <_>7 2 6 4 3.</_></rects>
       <tilted>0</tilted></feature>
     <threshold>0.0151513395830989</threshold>
     <left_val>0.1514132022857666</left_val>
     <right_val>0.7488812208175659</right_val></_></_>
  <_>
  …
```


Figure 3. XML Structure of haarcascade_frontalface_alt.xml

Size node represent the size of moving mask where haar-feature located. In figure 3, it is shown that the size of the moving mask is 20 x 20.

Every rect nodes consists of several different numbers. The two first numbers represent left and top position of the Haar-feature relative to the moving mask position, the two next numbers represent width and height of the Haar-feature, while the last number is weight of the Haar-feature.

XML Transformation to Reflect Classifier Transformation

As discussed in the previous chapter, the first proposed solution of rotated-object recognition is by creating more classifiers. In this paper, we conduct a clever way to transform the XML so that the classifier will be able to recognize rotated object.

This transformation is only works for 90^0 and -90^0 of angle. Suppose that the original haar-feature has top-left coordinate (x,y), width (w) and height (h), while the size of moving mask is W and H.

To rotate the feature 90^0 of angle we use these formulas:

new_w = h,
new_h = w,
new_x = y,
new_y = H - x - w

While to rotate the feature -90^0 of angle we use these formulas:

new_w = h,
new_h = w,

new_x = W-y-h,
new_y = x

By using regular expression based replacement, we have created 2 new XML files. Each of them are used by new Haar-Classifiers to recognize images contains 90^0 and -90^0 rotated object.

Image Rotation on Runtime

The second proposed solution is by rotate image on runtime and feed it to original classifier. Image rotation is a very common task to do in image processing. To do image rotation, we can use rotation matrix as described in figure 4.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Figure 4. Transformation by Using Rotation Matrix

In figure 4, x' and y' represents new pixel position after rotation, while x and y represents old pixel position.

By using open-cv, we can simply use *cv2.getRotationMatrix2D* function to obtain the rotation matrix and *cv2.warpAffine* to do the transformation.

Test Case

For testing purpose, we use several images with clockwise and counter-clockwise rotation as shown in figure 5.

Figure 5. Images for Test Case

All of the test-case has many faces in a single image except lena, lena-ccw, and lena-cw.

3. RESULT AND DISCUSSION

We use several to measure the effectiveness and efficiency of both solutions. The parameters are as follow:

- execution time
- true positive

- false negative
- false positive.

Execution times are measured by computer, while other parameters are gathered by observation.

True positive is the count of faces 0^0, $90^{0\ of}$ -90^0 degrees of rotation which are correctly recognized by the classifier.

False negative is the count of faces with 0^0, $90^{0\ of}$ -90^0 degrees of rotation which are not recognized by the classifier.

False positive is the count of non-faces which are misclassified as faces by the classifier.

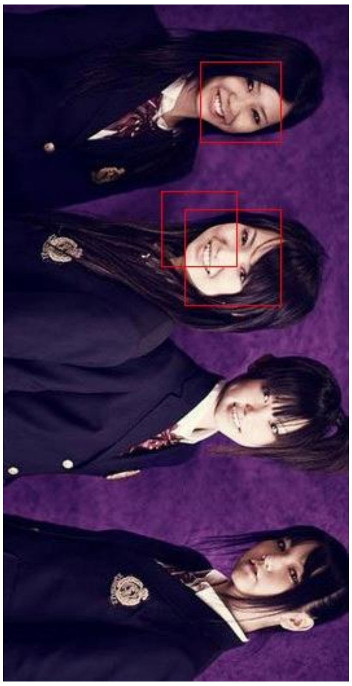

Figure 6. Recognition Result of scandal2-cw.jpg by Using single Haar-Classifier.

In figure 6, there are 3 frontal faces since the most bottom face is not frontal. The system can successfully recognize 2 of 3 faces, therefore the true-positive parameter is 2. The system fails to recognize 1 frontal face, therefore the false-negative is 1. The system also recognize 1 non-face object as face, therefore the false-positive is 1.

The summary result of the experiment is shown on figure 6-9.

Execution Time

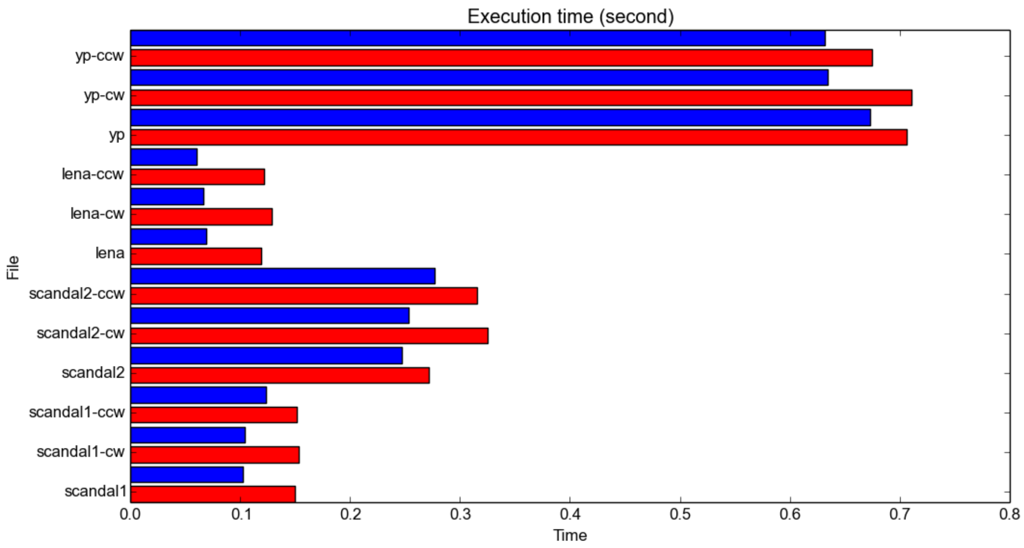

Figure 6. Execution Time

The experiment shows that single classifier (the top one of each pair in figure 6) has better speed compared to multi-classifier. This shows that rotating an image is more efficient that create many classifier.

True Positive

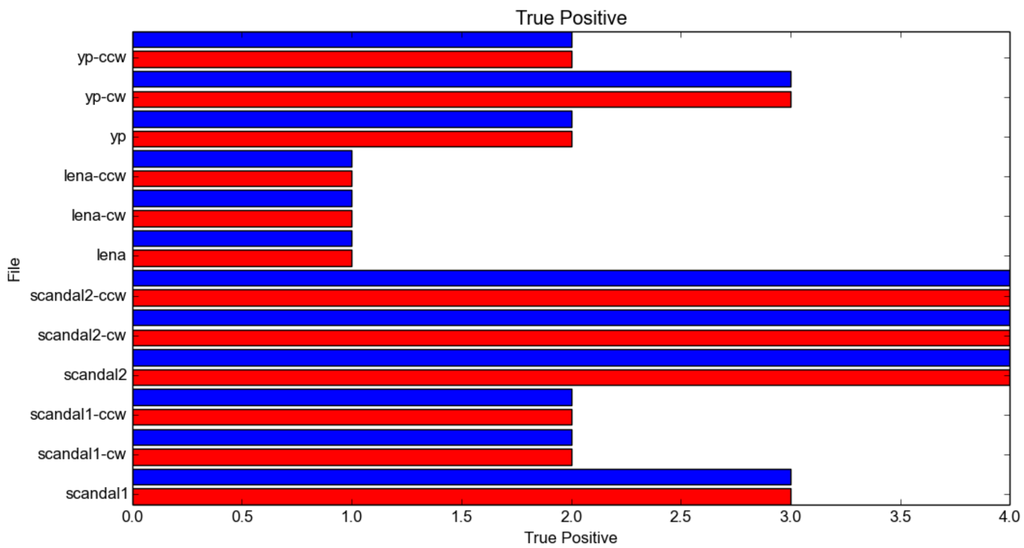

Figure 7. True Positive

As expected true positive value of the two classifiers is match each other. This means that the two classifier can equally recognize true objects.

False Negative

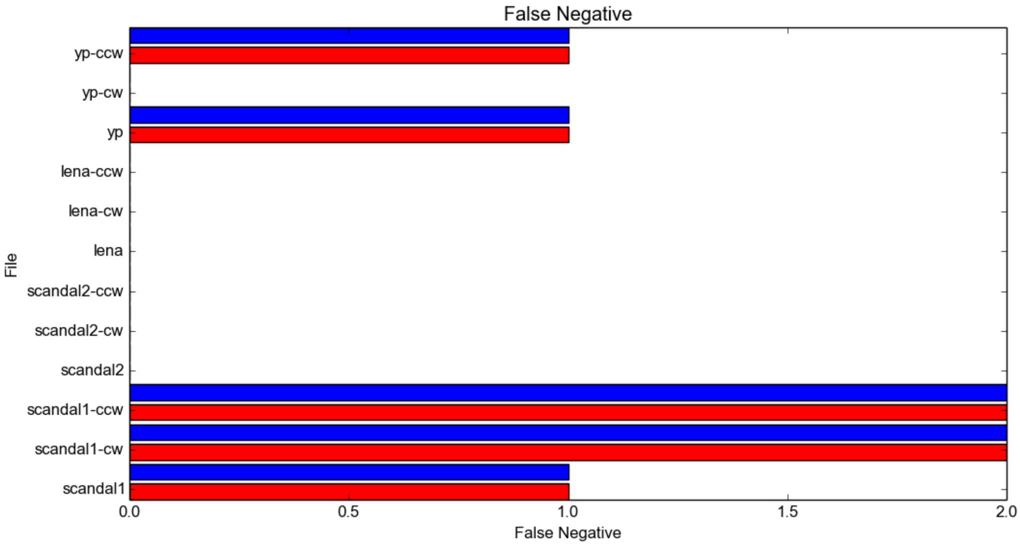

Figure 8. False Negative

As true positive, the two classifiers shows the same value of false negative. This means that there are some faces unrecognized by the classifier.

False Positive

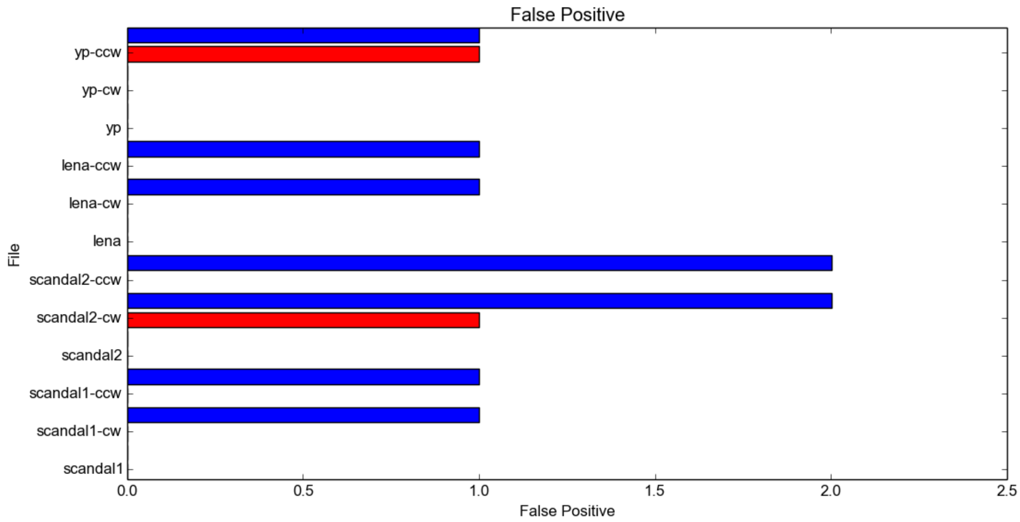

Figure 9. False Positive

The experiment show that single classifier tend to produce more false positive compared to multi classifier. This also shows that muti classifier is more sensitive compared to single classifier.

This result is unexpected. In our expectation, both classifiers should have same recognition quality. We only expect execution time difference.

4. CONCLUSION

From the experiment conducted, we can conclude that single haar-classifier with rotated image has better speed compared to multiple haar-classifier.

However we also find a surprising fact that multiple haar-classifier is more sensitive than single haar-classifier. We will try to find out the reason behind the sensitivity of multiple haar-classifier as a further work.

REFERENCE

[1] Paul Viola and Michael J. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE CVPR, 2001.

[2] Rainer Lienhart and Jochen Maydt. An Extended Set of Haar-like Features for Rapid Object Detection. IEEE ICIP 2002, Vol. 1, pp. 900-903, Sep. 2002.

[3] Opencv developer team, Open-CV documentation http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html, Accessed on April 2014