Towards Privacy-aware Photo Sharing using Mobile Phones

Sadia Shamma and Md. Yusuf Sarwar Uddin
Computer Science and Engineering Department, BUET
shamma167@gmail.com, yusufsarwar@cse.buet.ac.bd

***Abstract*—Mobile phones are used these days to capture photos almost everyday everywhere. When photos are taken and shared among peers through social network sites or photo sharing portals, sometimes they pose threats to personal privacy of persons who appear in those photos, particularly when the person being photographed has reservation about sharing the same. In general, photos containing detectable human faces would be handled with care. In this paper, we propose techniques and develop a mobile app that can be used as an assistant tool in order to make people aware of their photo content. The app, integrated as a part of phone's builtin Camera app, categorically detects faces in an image preview and notify the user about detected faces (if any). The user then uses his/her rightful discretion to share or store the captured picture being respectful that no one's privacy is violated through this capture. Two possible techniques for detecting faces, namely cloud based and in-phone processing based, are proposed and a few experimental results with the app have been reported.**

***Index Terms*—Mobile Phone App, Camera, Privacy, Photo sharing**

I. Introduction

This paper proposes and develops a working solution towards privacy-aware photo sharing using mobile phones. As multimedia phones are very much available now a days, people can capture photos anywhere and everywhere. Photo sharing has become a regular phenomenon these days. Online users share photos through social network websites, such as facebook, or through photo sharing portals, such as Flickr. Due to wide penetration of Internet-enabled smart phones equipped with high resolution camera, people using those phones can easily capture images of people or scenic sites and can upload them immediately to web 2.0 portals either through standard websites or specialized apps, such as Instagram. It has been reported that facebook alone stores nearly 260 billion images and users upload around one billion new photos every week [2]. While photos are intended to capture moments of life very vividly, they also pose significant threats of personal privacy. A person can take a photo of another person where the later one may not necessarily opt in for the take. The photo can be even captured stealthily without letting the subject knowing that his/her photo is taken. Usually face reveals a person's identity quite certainly, which makes the presence of detectable faces in a photo an important issue that deserves to be treated with care. The issue becomes even more sensible in some cultures when the photo contains face of a woman whose consent was not sought while taking the photo.

In this paper, we propose techniques and develop a mobile App that can be used as an assistant tool in order to make people aware of their photo content. The app, integrated as a part of phone's builtin Camera app, categorically detects faces in an image preview and notify the user about detected faces (if any). The app can also detect the gender of the faces, specially whether there is any female face. The user then uses his/her rightful discretion to share or store the captured picture being respectful that no one's privacy is violated through this capture. We propose two possible techniques for detecting faces, one using a cloud service and another one with in-phone processing. Experimental results with the app are reported.

It is note that the app does not really solve the big problem of photo privacy. The key issue is with lack of strong enforcement. The problem is privacy breach (if at all) occurs with the person whose photo is taken, but not with the person who is taking the photo and who is actually holding the camera. This can be called *subject mismatch*. We argue that our proposed solution is mostly an advisory tool that assumes responsible and sensible attitudes to human subject guided with proper judgment. The app helps in avoiding accidental sharing, particularly for automated uploading of photos as they are shoot.

II. Related Work

Many researchers worked in different platform to protect photo privacy in web services and camera devices. Sergej Zerr, Stefan Siersdorfer and Jonathon Hare have developed a search engine for privacy oriented image PicAlert! [7] It automatically identifies potentially sensitive images in the result set and separates them from the remaining pictures. But PicAlert does not support any additional features for pictures taken with mobile phones where we expect a larger proportion of private images. Roberto Yus *et. al.* have worked on Faceblock service [6]. It takes regular pictures taken by smart phones or Google Glass as input and converts them into privacy-aware pictures. Photographer and the person who is being photographed both of them need to be Faceblock user. He Wang *et. al/* proposes techniques on Insight project which recognizes human without face recognition [5].

The closest match with our work is [3] where authors propose a system architecture for privacy-aware photo sharing over social network websites. The system allows users to set privacy preferences in their personal profiles hosted in their respective social network sites. These preferences ascertain certain policies, such as whether other people upload his/her photos without his/her consent as well as whether to notify user when someone's else upload his/her photo in any portal where the user has account in. While the work addresses privacy issues, it does so in an offline manner. In contrast, our proposed scheme is rather online in the sense that we address the privacy issue right on the spot, at the very instance when the users capture photos, not in a later time when they actually

upload them. This happens to be possible because we develop our scheme on mobile phones.

III. Privacy-aware Photo Sharing

We propose a service to prevent people to be photographed by camera unwillingly. If someone attempts to capture a photo, it detects if there is any face in camera preview. If any face is detected, then it notifies the user with a message that he/she is going to capture a photo with a face in it. If it is unintentional, then the user can choose his rightful actions. Face detection is computationally expensive that requires advanced vision algorithms. We use existing available services to detect faces. We apply two approaches for face detection: one is to use a cloud base service for face detection and the other one is to detect face locally in phone using available vision library, such as openCV (www.opencv.org). It is noteworthy that we do not 'recognize' individual person in photos, rather 'detect' the presence of human faces, which is apparently less expensive than recognition in terms of computation, which makes it suitable for in-phone processing. Unlike recognition, it also does not require large training dataset to learn faces.

A. Cloud based Face Detection

One of our approaches is to use cloud base service for face detection. We select a photo and send it to a Web service through REST API to check whether it contains a face. If so, the user is notified with a message. We use REST APIs from SkyBiometry for our demo application.

An advantage of using cloud base face detection is that computational overhead is transferred to cloud. Cloud itself processes the image, analyzes it and provides us the result. Our application notifies user according to that provided result. Accuracy of face detection is very satisfactory in this service as it has large data set, training set and continuous learning algorithm is applied here.

Disadvantage of using cloud base face detection is privacy issue. We need to upload image to cloud server for result. If privacy issue is highly significant, then uploading image to server violates privacy policy. We need to ensure secured transport layer for image uploading and result transferring. Moreover, we need to trust the service that it does not have any image leakage. In case of strong privacy issue user may not like this cloud base face detection system. Another disadvantage of this cloud base service is service latency. It takes some time to process in cloud and show the result according to image size. And last but not bot least, it requires a persistent (and fat) data connection which may not be so viable for all.

B. In-phone processing for Face Detection

Another approach we describe is detecting faces in phone locally. It is recommended for cases where user wants to ensure strong privacy policy. In that, users do not need to upload photo to server as it detects photo locally in phone. We apply a classifier using computer vision algorithm that detects faces locally.

Advantage of in phone local face detection is that it does not hamper privacy issues. As it processes the image in phone, there is no risk of image leaking or privacy violation. Another benefit could be quick processing as it does not involving sending the photo file over Internet and receiving response back. Disadvantage of using local face detector is, however, low accuracy. Since local face detector is tested and trained on small data set, it often fails to detect face accurately. Processing capability of phone varies according to phone configuration. So, processing time and performance may vary according to the configuration of phone.

IV. Implementation

We implemented privacy-aware photo sharing as an Android app. The app uses builtin camera application to capture photos and then analyzes to check whether the photo contains any human faces in it. If so, it prompts the users for his/her explicit consent for the actions to follow. in current implementation, the app runs as a standalone app. That means, the user can easily bypass this app to take photos with phone's builtin camera which does not have, as we know, any privacy enforcement. The app can indeed be wired into camera application so that enforcement can be activated. We do not, however, consider this option in this work.

One important aspect of the app is to detect human faces when a photo is taken by the camera. As we mentioned, for detecting faces we use two techniques. First one is to use a cloud base service and second one is to detect face locally in phone. In the following, we describe these two techniques.

A. Cloud based implementation

In our implementation, we used SkyBiometry REST API for face detection [1]. SkyBiometry is a cloud-based face detection and recognition API. It uses computer vision algorithms for face detection and recognition. This API provides facial attributes like gender, glasses, eyes state, lips state, and mode. Face detection and recognition API currently supports methods, such as faces/detect, faces/recognize, faces/train. The API uses REST interface, which means API methods are called over the Internet, using standard HTTP methods like GET and POST to api.skybiometry.com/fc/. Depending on the HTTP request parameters, server can generate response in either JSON or XML format. It also supports JSONP callbacks to simplify the application development in JavaScript and callbacks for asynchronous invoking of lengthy operations. According to the result provided by service a face detection message is shown. As this service supports gender detection, so female faces can be especially marked for additional attention.

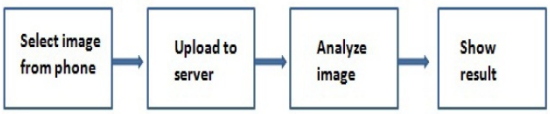

Fig. 1. Steps in cloud based face detection

B. In-phone processing

We use openCV library ported into Android for an in-phone processing of face detection. OpenCV's face detector uses a method that Paul Viola and Michael Jones published in 2001 [4]. This approach of detecting objects in images combines four key concepts: simple rectangular features, called Haar features, an integral image for rapid feature detection, the AdaBoost machine-learning method, and a cascaded classifier

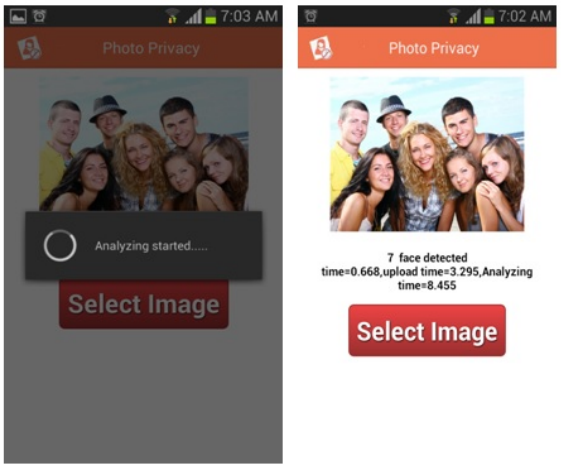

Fig. 2. Detecting face in cloud based service

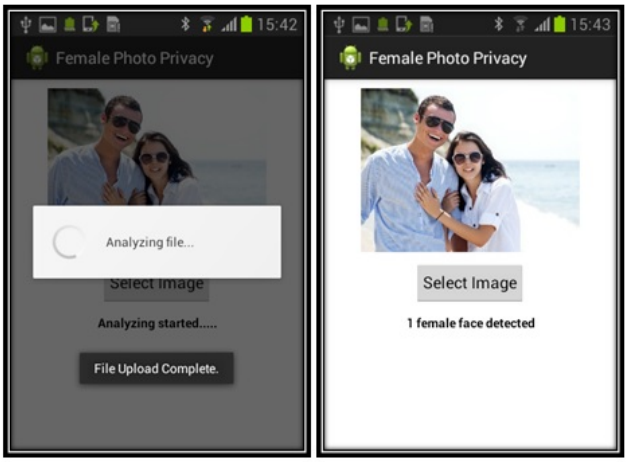

Fig. 3. Detecting female face in the cloud based service

to combine many features efficiently. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. Those XML files are stored in specific directories in openCV installation. We use openCV's face detection algorithm in our app. It detects face in camera preview and notifies user with a message when a face is detected. It then asks for user's permission to capture the photo with face. If the user wants to proceed in capturing photo with face, then it saves the image in phone memory. Otherwise, it discards the photo.

Fig. 4. In phone local face detection

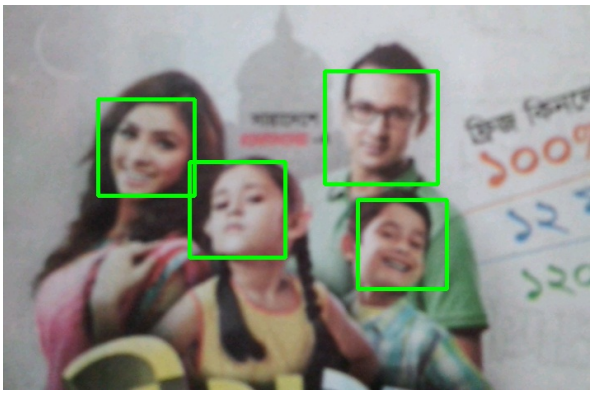

Fig. 5. In phone processing face detection

V. Experiments and Evaluation

We make experiments on our developed app with a small dataset. Firstly, we show results for the cloud based implementation. We measure the processing time for detecting faces for each photo set. We experiment with different resolution of photos as resolution affects processing time significantly. It is shown in Figure 7 that photos with higher resolution need more processing time than lower resolution. Cloud based service usually requires larger processing time as it needs transfer of image data to the server, which may constitute a large portion of processing time. We can in fact express processing time as a sum of its several components, as follows:

$$p(t) = pp(t) + u(t) + a(t)$$

where,
$p(t)$ = Processing time;
$pp(t)$ = Pre-processing time;
$u(t)$ = Data upload time (transfer image data);
$a(t)$ = Analyzing time (detecting faces);

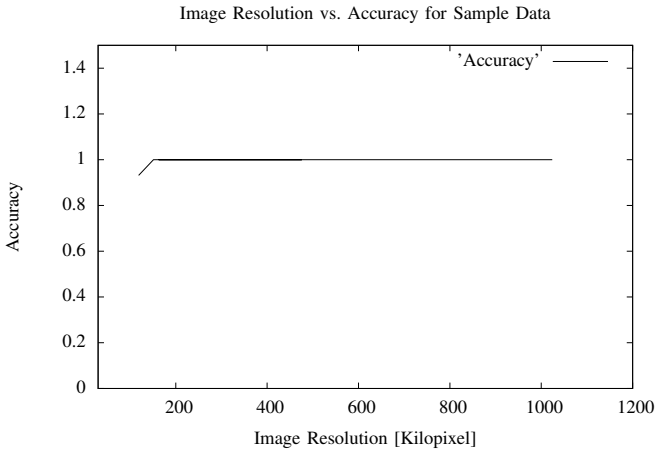

Fig. 6. Accuracy for cloud based face detection

We also measure accuracy of our app in detecting faces. Accuracy is calculated with the following equation:

$$\text{accuracy} = \frac{\text{no of true positive + no of true negative}}{\text{total number of photos}} \quad (1)$$

Where, true position refers to cases when detection technique indeed detect faces, whereas true negative does not find

any face when there is indeed no face in the photo. False cases are those where things happen in contrary: detecting faces when there is actually none or misses a few faces to be detected. We show accuracy results in Figure 6. We see that photos with lower resolution have lower accuracy compared to photos with higher resolution. Since cloud base service uses more complex and efficient algorithm, it has better accuracy than in phone local face detector.

We also show detail accuracy results for openCV based in-phone face detection for a set of captured photos in Table 8. We conclude that accuracy for cloud based service is almost 100 percent, while in-phone processing accuracy is 73 percent.

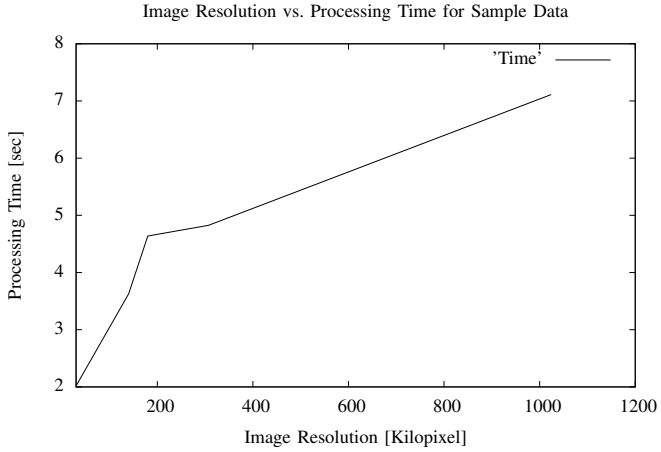

Fig. 7. Processing time for cloud based face detection

Pic_ID	No of Faces	True +ve	True -ve	False +ve	False -ve
114302	1	1			
114345	1	1			
114353	1	1			
114415	1	1		1	
114435	2	2			
114503	1	0			
114513	2	1	1		
114523	5	2	3		
114554	4	3	1		
114617	1	1			
114627	0	0			
114641	1		1		
114701	1	1		1	
114742	2	1	1		
114750	1	1			
114754	1	1			
114802	1	1			
114820	2	2			
115133	1	1			
115137	1	1			
115144	1	1			
115150	3	2	1		
115200	0	0			
115206	0	0			
115223	1	1			
115255	1	1			
115300	4	3	1		
sum	40	30	9	2	

Fig. 8. Detail accuracy calculation of in-phone processing of face detection

VI. Conclusion and Future Work

This problem of privacy breaches arising due to photo capture and sharing using mobile phones is taking much attention in recent years. This paper puts positive efforts towards privacy-aware photo sharing using mobile phones. Particularly, we propose instrumenting phones with a suitable app so that such breaches can be contained and passed to user attention immediately right after they shoot pictures of people. At this stage, the app is an independent standalone application, but it can be wired into built-in camera of devices.

Face recognition and other features can be added as an easy extension of this work. One possible thought can be as follows. Individual user can have some tagged photos of his friends/ relatives, which means these persons can be in photos without any objection. When the user attempts to capture faces other than those, the app detects a privacy breach and asks for confirmation. As a consequence, the user may avoid taking that photo or may take photo in which that unknown face is blurred. As the face area of image is detected, that area can be blurred with an appropriate measure.

Another extended version of this application can be adding features to prevent unwanted photo sharing over Internet. Photos containing unknown faces can be blocked from uploading to network. This can be integrated with gallery application of camera.

Sometime, photos in photos are automatically uploaded in some site where the user account is synced with the device. In that case, user will be asked for confirmation if image contains face before it uploads it to the respective server. If confirmation is unavailable, those photos will be kept in queue. Sometimes, phones are set of party mode, where all photos taken a particular time span are automatically shared in user's online social networks. Unwanted photos can be blocked if our scheme is activated in those cases.

References

[1] https://www.skybiometry.com/.

[2] Doug Beaver, Sanjeev Kumar, Harry C. Li, Jason Sobel, and Peter Vajgel. Finding a needle in haystack: Facebooks photo storage. *Available at https://www.l3s.de/~siersdorfer/sources/2012/picalert.pdf*, 2010.

[3] Thorben Burghardt, Andreas Walter, Erik Buchmann, and Klemens Bohm. Primo–towards privacy aware image sharing. *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2008.

[4] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *IEEE Computer Vision and Pattern Recognition*, 1:I–511 – I–518, 2001.

[5] He Wang, Xuan Bao, Romit Roy Choudhury, and Srihari Nelakuditi. Insight: Recognizing humans without face recognition. *ACM HotMobile*, 2013.

[6] Roberto Yus, Primal Pappachan, Prajit Kumar Das, Anupam Joshi Eduardo Mena, and Tim Finin. Demo: Faceblock: Privacy-aware pictures for google glass. *ACM MobiSys*, 2014.

[7] Sergej Zerr, Stefan Siersdorfer, and Jonathon Hare. Picalert!: A system for privacy-aware image classification and retrieval. *Available at https://www.usenix.org/legacy/event/osdi10/tech/full_papers/Beaver.pdf*, 2012.