sustainability

MDPI

Article

Instant Social Networking with Startup Time Minimization Based on Mobile Cloud Computing

Lien-Wu Chen [1,*], Yu-Fan Ho [1] and Ming-Fong Tsai [2]

[1] Department of Information Engineering and Computer Science, Feng Chia University, Taichung 407, Taiwan; csaaa7@gmail.com

[2] Department of Electronic Engineering, National United University, Miaoli 360, Taiwan; mingfongtsai@gmail.com

* Correspondence: lwuchen@fcu.edu.tw; Tel.: +886-4-24517250 (ext. 3759)

Received: 22 February 2018; Accepted: 12 April 2018; Published: 16 April 2018

Abstract: Mobile communication and handheld devices are currently extremely popular, and provide people with convenient and instant platforms for social networking. However, existing social networking services cannot offer efficient human-machine interfaces or intuitive user experiences. Mobile users must manually input account information and find targets from search results when attempting to add someone to their friend list on social networking sites, such as Facebook and Twitter. Additionally, mobile users may not be able to identify correct targets because some usernames are identical. Typos may occur during the input process due to unfamiliar identifiers, further increasing the total operation time. To encourage social initiation between mobile users, we design an instant social networking framework, called SocialYou, to minimize the startup time based on mobile cloud computing. SocialYou proposes an efficient architecture and innovative human-machine interfaces to alleviate the complexity and difficulty for mobile users using handheld devices. In particular, we implement an Android-based prototype to verify the feasibility and superiority of SocialYou. The experimental results show that SocialYou outperforms the existing methods and saves substantial amounts of operation time for mobile social networking.

Keywords: cloud computing; human-machine interface; image processing; mobile device; social networking

1. Introduction

Through widely available mobile communication methods and handheld devices, social interaction using a specific social networking site (SNS) is no longer limited to use on personal computers or laptops. People can immediately access social networking services on mobile devices, such as smartphones and tablets [1]. Group activity organization [2], repost behavior modeling [3], social opportunity forecasting [4], and incentive crowd sensing [5] have been developed for mobile social networking. Although existing SNSs, such as Facebook and Twitter, can provide opportunities for interaction with strangers, they have certain limitations in providing the best experience for mobile users. For instance, if mobile users intend to add someone to their friend list, they must manually enter the target account or name. Selecting the correct account or name from the search results may require a considerable amount of time [6].

Searching for new friends in social networking processes poses three potential problems. First, mobile users may not know the target account or name. Second, searching for a target account or name may involve complicated text input, engendering the possibility of typos during the input procedure and thereby requiring mobile users to key in the information more than once. Finally, mobile users may not be able to identify the correct target because some usernames are identical [7]. To address these concerns, we propose an instant social networking framework, called SocialYou, which alleviates problems related to unknown targets, manual input, and target selection, thereby minimizing the social startup time.

Social networking services used for randomly making friends have been developed in mobile messenger applications, such as LINE and WeChat. Such developments may cause privacy and safety concerns [8]. To facilitate random social networking, LINE and WeChat designed the "Shake it!" and "Shake" functions, respectively, to facilitate communication with nearby users by shaking smartphones. In addition, WeChat offered the "Drift Bottle" and "Pick" functions to enable one user to send (i.e., throw) a text or audio message without specifying a receiver and other users to receive (i.e., pick up) the thrown message, which can initiate friendship among strangers.

On the other hand, the Twitter user recommender system [9] connects Twitter users to a relevant stranger who contains similar information in recent tweets and has at least one mutual friend between them. In addition, a Facebook Messenger bot, called Chatible [10], can be used to connect Facebook users to total strangers for providing a random conversation between anonymous users. Chatible randomly pairs a user with another user to allow users to chat with someone at random.

The instant friend recommending system [11] has been designed for location-based mobile social networks that allow users to add images and specific information in particular locations. To instantly make friends, the designed system takes the up-to-date positions in the physical space, offline behavior similarity in the real world, and online friendship network information in the virtual community into consideration. Mobile users can discover and interact with the interested people in the vicinity to make friends with the dynamic time, place, and person correlation.

Although aforementioned messenger applications and recommender systems provide rapid methods for making new friends, mobile users cannot immediately obtain detailed information about a specific target that they intend to become acquainted with. For example, when a student meets an attractive girl in the university library, the student intends to make friends with her. The student needs to acquire more detailed information about the girl before directly talking to her to increase the success probability of making friends. In particular, SocialYou can facilitate acquiring such detailed information about a specific target. In SocialYou, face detection and facial recognition techniques are employed to substantially simplify the processes of identity acquisition and input by preventing users from engaging in manual query actions and making typographical errors. Through SocialYou, mobile users can be directly guided to specific target webpages in selected SNSs without inputting the target account and can independently identify search results. SocialYou provides an intuitive social networking method and innovative human-machine interface for mobile users.

On the basis of our previous works [12,13], we design, implement, and evaluate SocialYou through cloud computing to facilitate social networking among mobile users. The SocialYou system consists of the smartphone app, intermediate server, and cloud servers. Face detection, face caching, and face recognition techniques are adopted in the smartphone app, intermediate server, and cloud servers, respectively. Once mobile users register their face photos and associated social identifiers in SocialYou, social networking interaction can be performed easily and quickly among them.

Compared with existing SNSs, SocialYou provides the following features: (1) the users need not know the target beforehand; (2) the users need not manually input information regarding the target; (3) the users can specifically select a target among multiple candidates through the camera preview; and (4) even if the target is not physically present, the users can use the image or video of the target for social networking purposes. The experimental results show that SocialYou outperforms the existing methods and saves substantial amounts of operation time in mobile social networking.

The contributions of our proposed framework are four-fold. First, an innovative gesture-based human-machine interface based on cloud computing is available for instant social networking among mobile users. Second, immediate face detection, intermediate face caching, and remote facial recognition are integrated into the framework to minimize social networking startup time. Third, the computing resources of mobile devices, intermediate servers, and cloud servers are exploited to reduce the overall system response time. Finally, an Android-based social networking system is implemented to verify the feasibility and superiority of the proposed framework.

The remainder of this paper is organized as follows. Section 2 discusses mobile cloud computing and defines the social networking problem. Section 3 describes the design of the instant social networking framework consisting of a remote cloud and a local cache for facial recognition. Section 4 presents the implementation of our system. Experimental results are provided in Section 5. Finally, Section 6 concludes this paper.

2. System Model

Mobile cloud computing has been developed to balance the system load and response time [14,15]. Three different architectures can be employed in mobile cloud computing [16,17]. The first is local mobile cloud computing, which enables mobile devices to cooperatively serve as resource providers of the cloud by forming peer-to-peer networks [18,19]. The second is remote mobile cloud computing, which enables mobile devices to directly utilize the computing power and data storage capacity of cloud servers [20,21]. The third is cloudlet mobile cloud computing, which enables mobile devices to indirectly offload their work load to cloud servers via dedicated cloudlet servers [12,22–25].

Our framework is designed based on cloudlet mobile cloud computing for facial recognition and social networking. Through mobile cloud computing, mobile devices' resources and cloud computing techniques can be integrated to recognize a human face based on its social identifiers within a few seconds. In addition, instead of the entire image, face detection can be performed on mobile devices to upload only the detected face part, thereby minimizing the transmission and recognition times of cloud servers.

In particular, mobile users' privacy can be protected by deploying the SocialYou server between mobile devices and cloud servers. Instead of the local database on mobile devices, SocialYou accounts and face samples of mobile users can be stored in the remote database on the SocialYou server. Furthermore, unauthorized access to cloud servers can be prevented because the authentication information required for accessing facial recognition services is not stored on mobile devices.

Figure 1 shows the system architecture of SocialYou. Mobile users can use the SocialYou App on their devices to log onto the SocialYou server through Wi-Fi or cellular (e.g., LTE/4G) communication capabilities. Through the gesture-based human-machine interface provided by the App, mobile users can focus the camera on a social target, drag an SNS icon to the target face on the touchscreen, and after taking a photo, the target face can be detected and sent to the SocialYou server.

The SocialYou server forwards the received photo to a cloud server for facial recognition. The forwarded face photo is recognized by the cloud server, which subsequently detects the associated SocialYou account. Meanwhile, the target face is recognized by the SocialYou server if the local face cache contains the target information (i.e., the target face and its associated SocialYou account). The recognized SocialYou account is returned to the mobile user for identifier confirmation. The confirmed SocialYou account is then used to find the related SNS identifier through the SocialYou server. The related SNS identifier is sent to the mobile user and the target webpage in the selected SNS is displayed in the related social media App.

Through face detection and facial recognition techniques, our framework eliminates the need for mobile users to manually query target accounts and input social identifiers. The SocialYou App uses augmented reality technology to enable an SNS icon to be dragged to a target face on a touchscreen for social networking. The camera is pointed to a face (i.e., reality object) and receives an identifier (i.e., augmented information) of the face on the screen. In the SocialYou server, the target face is forwarded to a cloud server for facial recognition by the SocialYou App for identifier acquisition. The goal is to minimize the time of social networking operations for mobile users. SocialYou optimizes the social networking process among mobile users by addressing the following concerns:

1. Face Detection: A target face can be properly detected when a mobile user drags an SNS icon to a specific face among multiple candidates on the camera screen.

2. Facial Recognition: A target face can be rapidly transmitted and successfully recognized to obtain the target SocialYou account through cloud facial recognition or local face caching.
3. Account Confirmation: After a target face has been recognized through cloud facial recognition or local face caching, the target SocialYou account can be explicitly confirmed by the mobile user.
4. Social Networking: After a target SocialYou account has been confirmed, the target webpage in the selected SNS can be displayed immediately in the social media App or built-in web browser.

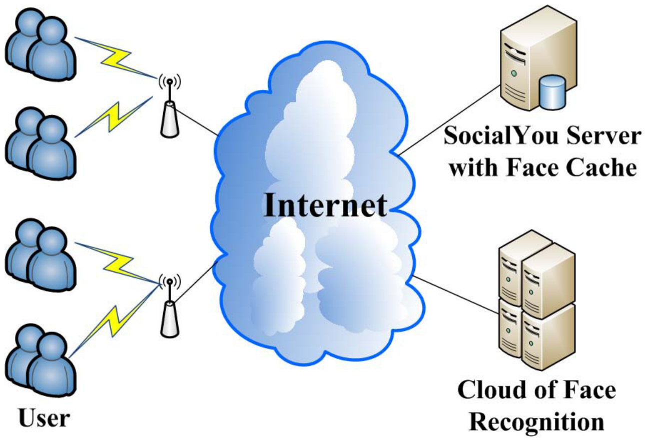


Figure 1. System architecture of SocialYou.

3. Instant Social Networking Framework

The proposed instant social networking framework is composed of the client, cloudlet, and cloud modules. The client module consists of the SNS listing and camera preview interfaces that enable mobile users to drag an SNS icon representing a social networking website on the SNS listing interface to a target face on the camera preview interface. The target face is detected based on where the SNS icon is released on the touchscreen. The face closest to the releasing position is sent to the cloudlet and cloud modules for facial recognition. Based on the registered face photo and associated social identifiers returned from the cloudlet and cloud modules, mobile users can explicitly confirm the identified target information, then the target SNS webpage can be immediately shown in the related social media program.

The cloudlet module possesses face caching and forwarding capabilities to forward the face photo received from the client module to the cloud module for remote facial recognition, thereby mapping the recognized user account to an associated social identifier and storing the recognized face samples in the cache for local facial recognition. The cloud module contains a user database of face samples and facial recognition services to recognize the forwarded face photo based on the database and subsequently return the recognized user account to the cloudlet module. For privacy protection, a SocialYou user can enable the option of instant social networking with permission in advance. Thus, before other people can look for a user's social media profile, the user will be notified through the requesting message sent by the SocialYou server and can decide whether to accept the request of instant social networking from other people.

Compared with existing cloudlet-based architectures [12,22,23], our framework further integrates mobile cloud computing with local face caching in the cloudlet module to reduce the response time of facial recognition. A complete database of face samples is constructed in the cloud module when mobile users register themselves in the cloudlet module, whereas partial face samples are cached in the cloudlet module when mobile users perform facial recognition through the cloud module. Upon receiving a facial recognition request, the cloudlet module simultaneously sends the request to the cloud module and recognizes the target face based on cached face samples. If the face sample of a target has been cached, the cloudlet module immediately returns the facial recognition result; otherwise, the cloudlet module awaits the recognition result of the cloud module and then sends the recognized target information to the users. Thus, the cloudlet module does not need to store a large number of face samples for each registered user; instead, a few face samples from popular targets are cached in the cloudlet module.

Figure 2 shows the flowchart for user registration in SocialYou. When mobile users launch SocialYou for the first time, they must capture some photos of their faces and register themselves on the SocialYou server. Different photos captured from front and side views are snapshotted using SocialYou to construct a facial recognition database. These photos are then sent to the SocialYou server for face detection, and the detected photos are forwarded to a cloud server for feature extraction. Users' facial features are extracted to build the facial recognition database. Finally, the users are required to input their social identifiers, such as Facebook ID and Twitter account, to link the registered SocialYou account to related SNS identifiers.

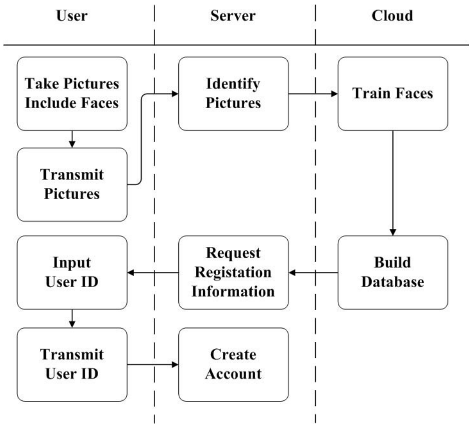


Figure 2. Flowchart of user registration.

Figure 3 shows the flowchart for social networking in SocialYou. Once mobile users successfully create their SocialYou accounts, they can log onto the SocialYou server, use their fingers to click on an SNS icon on the touchscreen, drag it to a target face on the camera preview, and release the dragged SNS icon on the target face. When the SNS icon is released, a picture of the current camera view is captured and scanned to assess whether it contains at least one face. If at least one face is detected, the face closest to the position where the SNS icon is released is transmitted to the SocialYou server, recognized through local face caching, and forwarded to the cloud server for facial recognition. Otherwise, the user must repeat the click-and-drag action on the target face.

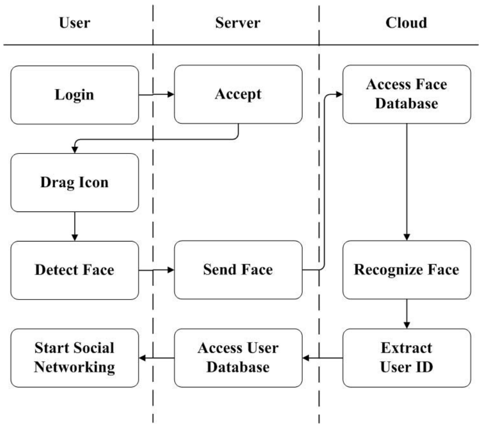


Figure 3. Flowchart of social networking.

When a cloud server receives the forwarded face photo, its associated SocialYou account is identified through facial recognition from the database of registered user faces. If the target face is recognized as one of the registrants in the database, the cloud server returns the recognized result to the SocialYou server. Otherwise, a mobile user must re-manipulate the SNS icon by dragging it to the target face. The identified SocialYou account and registered face photo are transmitted to the mobile user for identifier confirmation. After the mobile user confirms the target SocialYou account based on its registered face photo, the confirmed SocialYou account is used to seek its associated social identifiers from the SocialYou server. The associated social identifier is returned to the mobile user, and the related social media program is automatically executed to show the target webpage of the dragged SNS icon.

Subsequently, the algorithms used for detecting and recognizing target faces are explored. Face detection algorithms can be classified as color-, feature-, learning-, or template-based solutions. Color-based face detection [26] establishes a statistical model for human skin colors that are distributed within a certain range of the color space. By comparing each pixel of the target face with the established statistical model, we can extract the area of skin colors considered as the possible face location. The color-based approach offers rapid detection and easy implementation but may produce inaccurate results if the background colors are too similar to the skin colors. Feature-based face detection [27] utilizes facial features, such as eyes and lips, to detect the location of the face. The feature-based approach offers high detection accuracy for an image with only one face. If the image contains complex background objects, the detection accuracy may be markedly impaired.

Learning-based face detection [28] splits an image into several parts, standardizes each part, trains the parts through the Neural Network, and automatically distinguishes human faces from backgrounds. The learning-based approach offers high detection accuracy if an image contains a face from a front view, but the detection result may be inaccurate for faces captured in profile. Template-based face detection [29] divides facial features into several templates and represents them as standard patterns that can be detected under different face angles and environmental lights. The template-based approach is easy to implement but requires high execution time for considering the face size, facing direction, and rotation angle.

Facial recognition algorithms can be classified as local feature-based solutions that use several facial features from one face for recognition, or global feature-based solutions that treat an entire face as a single feature for recognition. In local feature-based facial recognition [30,31], facial features, such as eyebrows, eyes, noses, and mouths, are characterized as local features to be recognized separately and then integrated in coordination for final results. Although the local feature-based approach has higher recognition success rates than does the global feature-based approach, facial features must be properly localized and recognized, resulting in a high implementation complexity.

In global feature-based facial recognition [32,33], Eigenface, Fisherface, and Local Binary Patterns Histograms (LBPH) methods can be used to identify a single-featured face. Eigenface is obtained and recognized through Principle Components Analysis (PCA [34]). In addition to PCA, Fisherface uses Linear Discriminant Analysis (LDA [35]) to further improve recognition success rates. LBPH provides fast response time for real-time face detection. The design principles and performance comparisons of the aforementioned methods used in our framework are investigated in the following subsections.

3.1. Eigenface

The Eigenface method uses PCA to extract the features of face samples [34,36], where the eigenvalues of face samples must be calculated to find eigenvectors. First, the mean value x of all training samples (i.e., faces) is calculated as follows:

$$x = \frac{1}{k}\sum_{i=1}^{k} x^i, x^i = [x_1^i, x_2^i, \ldots, x_n^i]^T, \tag{1}$$

where k is the number of training samples and n is the dimension of a sample. Second, all training samples subtracted the mean value as

$$\overline{x^i} = x^i - x, \text{for } i = 1, 2, \ldots, k. \tag{2}$$

Third, calculating the covariance matrix

$$C = \sum_{i=1}^{k} \overline{x^i}\,\overline{x^i}^T. \tag{3}$$

Next, deriving the eigenvalues and eigenvector based on the covariance matrix

$$C \times E = \lambda \times E, \tag{4}$$

where eigenvector $E = [e_1, e_2, \ldots, e_n]$, eigenvalue $\lambda = [\lambda_1, \lambda_2, \ldots, \lambda_n]$, and eigenvector e_i corresponds to eigenvalue λ_i. Finally, the original image Y^i is projected to eigenvectors by

$$Y^i = E \times \overline{x^i} = [y_1, y_2, \ldots, y_k]^T. \tag{5}$$

Through the above process, the original image is reduced from N to k dimensions that can significantly lower the computation cost.

Using the Eigenface method, the face samples of a user are trained to find eigenvectors and the features of a human face are classified and projected onto the space of Eigenface. Although the Eigenface method can save a lot of space required by training samples, recognition success rates may decrease due to the variety of face angles and environmental lights.

3.2. Fisherface

In addition to PCA, the Fisherface method further uses LDA to extract the features of face samples [32,35]. The extracted features only contain the differences between human faces, which are not affected by face angles, facial expressions, or environmental lights. Thus, the Fisherface method has higher recognition success rates than the Eigenface method when the face to be recognized has different face angles or facial shadows from training samples in the database. In the Fisherface method, X is the random vector with c classes and each class i has N_i elements (i.e., $X = [X_1, X_2, \ldots, X_c]$ and $X_i = [x_1, x_2, \ldots, x_{N_i}]$). The between-class scatter matrix S_b and within-class scatter matrix S_w are used to represent the distances between images in different categories and the differences among images in the same category, respectively.

$$S_b = \sum_{i=1}^{c} N_i (u_i - u)(u_i - u)^T, \tag{6}$$

$$S_w = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - u_i)(x_j - u_i)^T, \tag{7}$$

where N_i is the number of elements in class i, u is the mean of all classes, and u_i is the mean of class i. The Fisherface method uses LDA to minimize within-class differences and maximize between-class distances for optimizing recognition success rates. Vector W is find to maximize the projection space of S_b/S_w by solving

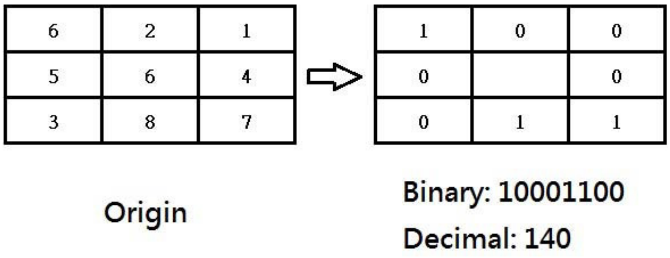
$$W_{LDA} = arg\ max \frac{|W^T S_b W|}{|W_T S_w W|}. \tag{8}$$

To obtain the solution, the generalized eigenvalue decomposition can be used by solving

$$S_b W = \lambda S_w W. \tag{9}$$

3.3. Local Binary Patterns Histograms

The Local Binary Patterns Histograms (LBPH) method compares every pixel with its surrounding pixels in an image [37]. As shown in Figure 4, the center pixel of brightness 6 is picked and its brightness is compared with those of adjacent pixels. If the brightness of an adjacent pixel is larger than or equal to that of the center pixel, the adjacent pixel is marked as 1. Otherwise, it is marked as 0. Each pixel can be represented by binary values, which is called the local binary pattern (LBP). The LBP image is divided into several local regions and a histogram is extracted from each local region. The LBPH method is simple to implement and its computing speed is fast. LBPH is feasible for real-time systems but its recognition success rate may decrease when recognizing the image with the pixels having similar gray values.



6	2	1
5	6	4
3	8	7

1	0	0
0		0
0	1	1

Figure 4. An example of LBPH.

Figure 5 shows the comparisons of recognition success rates using the Eigenface, Fisherface, and LBPH methods under 1, 2, . . . , and 9 face samples per person in the AT&T database containing 50 people [38]. In Figure 5, each simulation is repeated 10 times and the average value is taken. As can be seen, when the number of face samples is larger than 7, all methods have higher and steadier recognition success rates. Through our experimental results, in order to achieve high recognition success rates, it is required that at least 10 face samples for a person must be provided during the user registration in SocialYou.

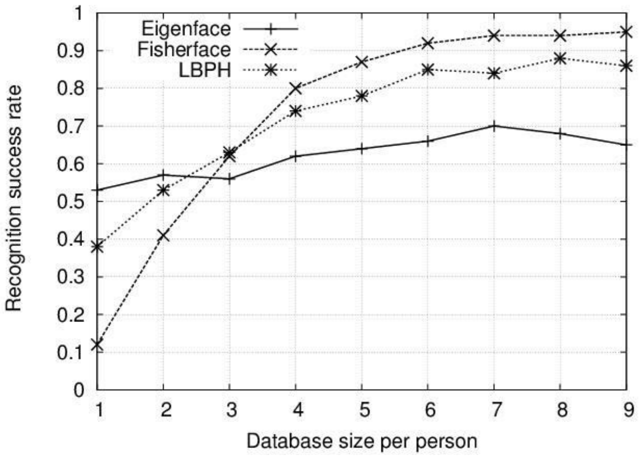
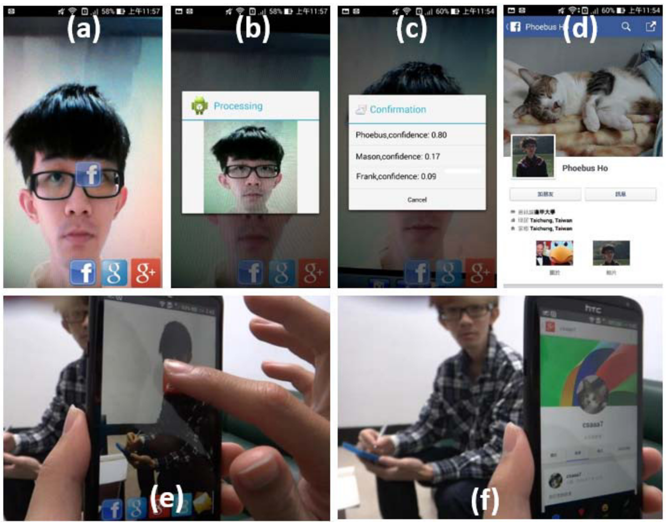


Figure 5. Comparisons of Eigenface, Fisherface, and LBPH.

4. System Implementation

We implemented an Android-based SocialYou prototype [39], as shown in Figure 6, in which the client, cloudlet, and cloud modules are implemented by an Android App, JAVA program, and cloud application programming interfaces (APIs), respectively. Android consists of an operating system, middleware, and user applications for mobile devices. The Eclipse integrated development environment (IDE) is used to develop the client and cloudlet modules of SocialYou. For local facial recognition, the SocialYou server uses the Open Source Computer Vision library [40] to implement the Eigenface, Fisherface, and LBPH methods. For remote facial recognition, the SocialYou server invokes the cloud APIs provided by Lambda Labs [41] and Mashape [42]. Lambda Labs develops and provides facial recognition APIs for developers through Mashape, which consequently distributes, manages, and consumes cloud APIs. Table 1 shows the cloud APIs used for remote facial recognition in the SocialYou server.



Phoebus,confidence: 0.80
Mason,confidence: 0.17
Frank,confidence: 0.09
Cancel

Figure 6. System manipulation of SocialYou: (**a**) click and drag the Facebook icon, (**b**) detect and transmit the target face, (**c**) recognize and confirm the user account, (**d**) search and display the target Facebook webpage, (**e**) click and drag the Google+ icon, and (**f**) search and display the target Google+ webpage.

Table 1. Cloud APIs used in SocialYou.

Name	Usage
createAlbum(String album)	Create a database of face samples.
trainAlbum(String album, String albumkey, String entryid, File files, String urls)	Construct the face database associated with a user account.
viewAlbum(String album, String albumkey)	Obtain the number of face samples in the constructed database.
rebuildAlbum(String album, String albumkey)	Reconstruct the database after uploading face samples required.
recognize(String album, String albumkey, File files, String urls)	Compare with all face samples to recognize a target face.

To minimize the response time (i.e., the sum of face transmission time and facial recognition time) for facial recognition, target face detection is implemented in the SocialYou App to efficiently upload the detected face photo instead of the entire image. The SocialYou App provides two face detection methods: Android software detection and camera hardware detection. Although based on the results of our realistic testing, camera hardware face detection is much faster than Android software face detection, Android software face detection is adopted by default because camera hardware face detection is not supported by all Android devices. If camera hardware face detection is supported by a user's Android device, the user can select it manually.

In our system implementation, the SocialYou App consists of SNS browsing and camera preview interfaces, as shown in Figure 6. Mobile users can use their fingers to click on an SNS icon, drag it to a target face, and release the dragged SNS icon on the target face, as shown in Figure 6a. Target face detection in the SocialYou App then uploads the detected face part, as shown in Figure 6b. By using Mashape cloud APIs, the SocialYou server can send a photo for facial recognition and then receive the three returned user accounts with the most similar faces. Next, the SocialYou server sends the user accounts returned by the Mashape cloud APIs to the SocialYou App, as shown in Figure 6c. Once a returned user account has been confirmed as the target of interest, the target webpage related to the dragged SNS icon can be immediately browsed using the related social media App, as shown in Figure 6d. Figure 6e,f show the system manipulation for Google+. In particular, we provide a demo video online to show how easily and quickly social networking can be done by mobile users through SocialYou (see http://youtu.be/uYhDOlMpLbE).

5. Experimentations

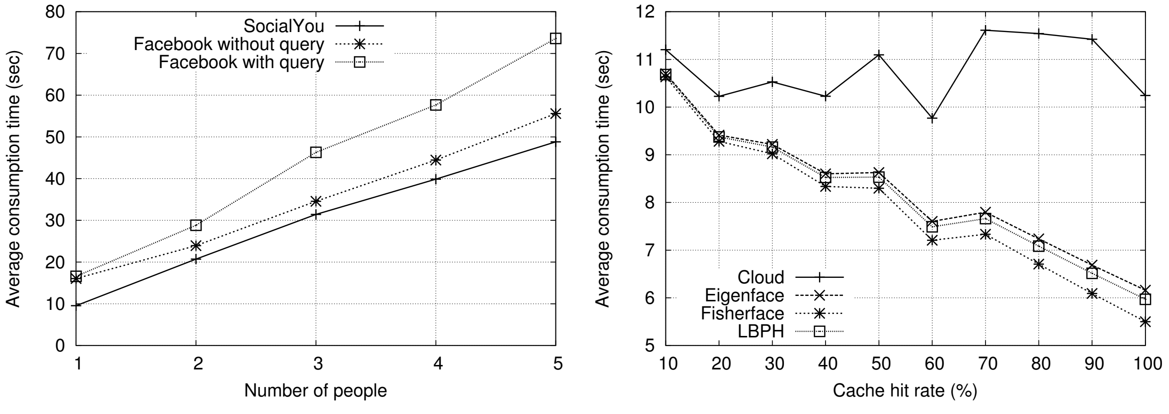
In this section, we compare the total consumption time for social and e-mail services through Facebook, Twitter, G-Mail, and SocialYou. In addition, we compare the performance improved by local face caching using Eigenface, Fisherface, and LBPH methods under different cache hit rates (with the replacement strategy of Least Recently Used). The durations for querying and inputting the target identifier are set to 4 and 8 s, respectively. Note that the total consumption time using the Facebook, Twitter, or G-Mail App is a lower bound without considering the long query duration that can be tens of seconds or even minutes. In our experiments, there are 50 registered users on SocialYou, of whom 10 participated in the comparisons of total consumption time using different social networking methods (i.e., Facebook App, Twitter App, G-Mail App, and SocialYou App). The experimental fields include the classroom, library, stadium, and sports ground in our campus, which show similar performance results in different locations. Each experiment is repeated 10 times by conducting realistic trials, after which we extract average values.

For social startups in Facebook, mobile users must launch the official Facebook App, input the target account or name in the search bar, and find the correct target among the obtained results. Figure 7a shows the comparisons of the total consumption time between Facebook and SocialYou Apps for various numbers of people for social startups. SocialYou has the lowest total consumption time for all numbers of social startup people. SocialYou users are not required to know and input the target identifier beforehand, which can save large amounts of query and operation time. Contrastingly, Facebook users must invest a considerable amount of time to obtain and input target identifiers in advance.

Figure 7b shows total consumption time comparisons for facial recognition by using cloud APIs, Eigenface, Fisherface, and LBPH methods under different cache hit rates for social startups in Facebook. The total consumption time can be markedly reduced when some popular target faces are cached in the SocialYou server. Since popular target faces are supposed to be recognized more often than others, thereby possibly increasing the cache hit rate, a facial recognition result can be quickly returned without waiting for a response from a cloud server, which efficiently reduces the total consumption time of SocialYou.

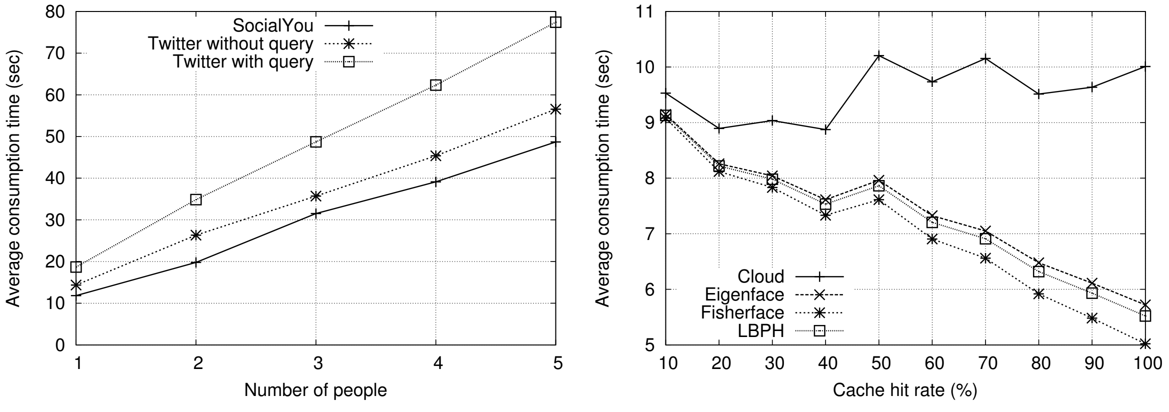
Figure 8a shows total consumption time comparisons between Twitter and SocialYou for various numbers of people for social startups. SocialYou has a much lower total consumption time than does Twitter both with and without query requirements. Manual intervention procedures for obtaining and inputting target identifiers are time consuming; therefore, avoiding them can substantially reduce social startup times. Similar to Figure 7b, Figure 8b shows that local face caching in the SocialYou server can markedly save the total consumption time for cached target faces. Figure 9a,b show that using the G-Mail App yields a similar result to using Facebook or Twitter, both of which require a substantially higher consumption time than SocialYou.

For the scalability of SocialYou, the adopted Cloud APIs are scalable by adding more computing sources for face recognition (i.e., using more virtual machines to recognize faces concurrently) as there are more and more registered users in SocialYou. In particular, the adopted Cloud APIs are flexible to extend computing sources (to recognize target faces in parallel) and data storage (to store training samples of registered faces) for scalability.



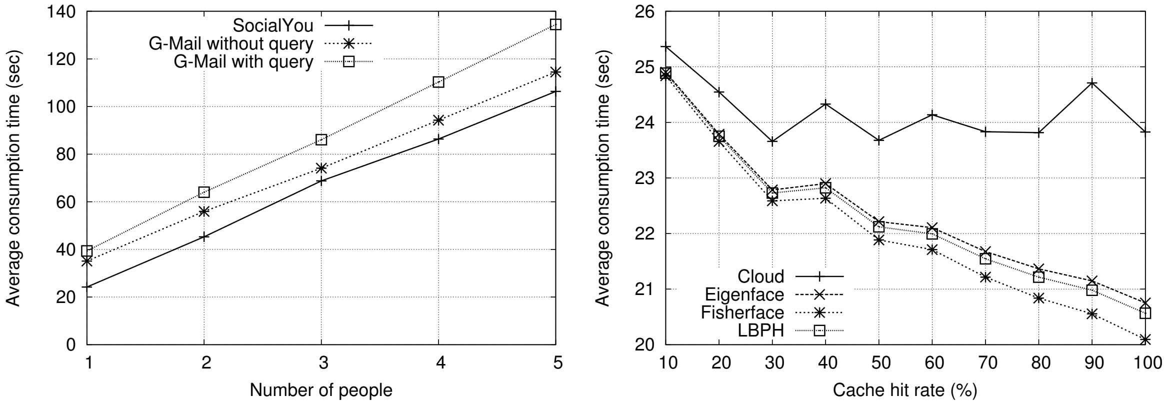
(**a**) (**b**)

Figure 7. Comparisons of total consumption time (**a**) using Facebook App and SocialYou with (**b**) recognizing faces by cloud API, Eigenface, Fisherface, and LBPH methods.



(**a**) (**b**)

Figure 8. Comparisons of total consumption time (**a**) using Twitter App and SocialYou with (**b**) recognizing faces by cloud API, Eigenface, Fisherface, and LBPH methods.



(**a**) (**b**)

Figure 9. Comparisons of total consumption time (**a**) using G-Mail App and SocialYou with (**b**) recognizing faces by cloud API, Eigenface, Fisherface, and LBPH methods.

6. Conclusions

In this study, the instant social networking framework with innovative gesture-based human-machine interfaces were designed and implemented on the basis of mobile cloud computing. Mobile devices, face caching in cloudlet servers, and cloud computing techniques were exploited to reduce overall processing times. In particular, immediate face detection, intermediate face caching, and remote facial recognition were performed on mobile devices, cloudlet servers, and cloud servers, respectively. Through our framework, mobile users are provided with opportunities for instant social networking without knowing target identifiers in advance. The results of realistic tests confirm that the proposed instant social networking system can markedly reduce the total consumption time for social startups and rapidly interact with encountered targets to achieve better social experiences, thereby engendering more efficient social networking for mobile users.

Acknowledgments: This research is supported in part by MOST under Grant No. 106-2221-E-035-019-MY3 and 105-2622-E-035-017-CC3.

Author Contributions: Lien-Wu Chen and Ming-Fong Tsai conceived key ideas and designed the framework; Yu-Fan Ho implemented the framework and performed the experiments; Lien-Wu Chen and Yu-Fan Ho analyzed the experimental results; Lien-Wu Chen wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. King, I. Introduction to Social Computing. In Proceedings of the International Conference on Database System for Advanced Applications, Tsukuba, Japan, 1–4 April 2010; pp. 482–484.
2. Guo, B.; Yu, Z.; Chen, L.; Zhou, X.; Ma, X. MobiGroup: Enabling lifecycle support to social activity organization and suggestion with mobile crowd sensing. *IEEE Trans. Hum.-Mach. Syst.* **2016**, *46*, 390–402.
3. Lu, X.; Yu, Z.; Guo, B.; Zhou, X. Predicting the content dissemination trends by repost behavior modeling in mobile social networks. *J. Netw. Comput. Appl.* **2014**, *42*, 197–207.
4. Yu, Z.; Wang, H.; Guo, B.; Gu, T.; Mei, T. Supporting serendipitous social interaction using human mobility prediction. *IEEE Trans. Hum.-Mach. Syst.* **2015**, *45*, 811–818.
5. Sun, J.; Ma, H. Heterogeneous-belief based incentive schemes for crowd sensing in mobile social networks. *J. Netw. Comput. Appl.* **2014**, *42*, 189–196.
6. Chang, K.T.T.; Chen, W.; Tan, B.C.Y. Advertising effectiveness in social networking sites: Social ties, expertise, and product type. *IEEE Trans. Eng. Manag.* **2012**, *59*, 634–643.
7. Irfan, R.; Bickler, G.; Khan, S.U.; Kolodziej, J.; Li, H.; Chen, D.; Wang, L.; Hayat, K.; Madani, S.A.; Nazir, B.; et al. Survey on social networking services. *IET Netw.* **2013**, *2*, 224–234.
8. Ballesteros, J.; Carbunar, B.; Rahman, M.; Rishe, N.; Iyengar, S.S. Towards safe cities: A mobile and social networking approach. *IEEE Trans. Parallel Distrib. Syst.* **2014**, *25*, 2451–2462.
9. Nidhi, R.H.; Annappa, B. Twitter-User Recommender System Using Tweets: A Content-Based Approach. In Proceedings of the International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 2–3 June 2017.
10. Facebook Messenger Bot—Chatible. Available online: https://www.facebook.com/chatible/ (accessed on 5 April 2018).
11. Xiuquan, Q.; Jianchong, S.; Jinsong, Z.; Wangli, X.; Budan, W.; Sida, X.; Junliang, C. Recommending friends instantly in location-based mobile social networks. *China Commun.* **2014**, *11*, 109–127.
12. Chen, L.-W.; Ho, Y.-F.; Kuo, W.-T., Tsai, M.-F. Intelligent file transfer for smart handheld devices based on mobile cloud computing. *WILEY Int. J. Commun. Syst.* **2017**, *30*, 1–12.
13. Chen, L.-W.; Ho, Y.-F.; Tsai, M.-F.; Chen, H.-M.; Huang, C.-F. Cyber-physical signage interacting with gesture-based human-machine interfaces through mobile cloud computing. *IEEE Access* **2016**, *4*, 3951–3960.
14. Khan, A.R.; Othman, M.; Madani, S.A.; Khan, S.U. A survey of mobile cloud computing application models. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 393–413.
15. Yang, S.; Kwon, D.; Yi, H.; Cho, Y.; Kwon, Y.; Paek, Y. Techniques to minimize state transfer costs for dynamic execution offloading in mobile cloud computing. *IEEE Trans. Mob. Comput.* **2014**, *13*, 2648–2660.

16. Ayad, M.; Taher, M.; Salem, A. Real-time mobile cloud computing: A case study in face recognition. In Proceedings of the International Conference on Advanced Information Networking and Applications Workshops, Victoria, BC, Canada, 13–16 May 2014; pp. 73–78.
17. Ahmed, E.; Gani, A.; Khan, M.K.; Buyya, R.; Khan, S.U. Seamless application execution in mobile cloud computing: Motivation, taxonomy, and open challenges. *J. Netw. Comput. Appl.* **2015**, *52*, 154–172.
18. Marinelli, E. Hyrax: Cloud Computing on Mobile Devices Using MapReduce; CMU-CS-09-164. Master's Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, September 2009.
19. Huerta-Canepa, G.; Lee, D. A Virtual cloud computing providers for mobile devices. In Proceedings of the ACM Mobile Cloud Computing and Services (MCS), San Francisco, CA, USA, 15–18 June 2010.
20. Chun, B.; Maniatis, P. Dynamically partitioning applications between weak devices and clouds. In Proceedings of the ACM Mobile Cloud Computing and Services (MCS), San Francisco, CA, USA, 15–18 June 2010.
21. Kumar, K.; Lu, Y.H. Cloud computing for mobile users: Can offloading computation save energy. *IEEE Comput.* **2010**, *43*, 51–56.
22. Soyata, T.; Muraleedharan, R.; Funai, C.; Kwon, M.; Heinzelman, W. Cloud-Vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Cappadocia, Turkey, 1–4 July 2012; pp. 59–66.
23. Indrawan, P.; Budiyatno, S.; Ridho, N.M.; Sari, R.F. Face recognition for social media with mobile cloud computing. *Int. J. Cloud Comput. Serv. Archit.* **2013**, *3*, 23–35.
24. Amin, A.H.M.; Ahmad, N.M.; Ali, A.M.M. Decentralized face recognition scheme for distributed video surveillance in IoT-cloud infrastructure. In Proceedings of the IEEE Region 10 Symposium (TENSYMP), Bali, Indonesia, 9–11 May 2016; pp. 119–124.
25. Sonmez, C.; Ozgovde, A.; Ersoy, C. Performance evaluation of single-tier and two-tier cloudlet assisted applications. In Proceedings of the IEEE International Conference on Communications Workshops (ICC Workshops), Paris, France, 21–25 May 2017; pp. 302–307.
26. Zhao, L.; Sun, X.; Xu, X. Face detection based on facial features. In Proceedings of the International Conference on Signal Processing, Guilin, China, 16–20 November 2006; pp. 16–20.
27. Hassaballah, M.; Ido, S. Eye detection using intensity and appearance information. In Proceedings of the IAPR Conference on Machine Vision Applications, Yokohama, Japan, 20–22 May 2009; pp. 20–22.
28. Jang, J.-S.; Kim, J.-H. Fast and robust face detection using evolutionary pruning. *IEEE Trans. Evolut. Comput.* **2008**, *12*, 562–571.
29. Hayashi, S.; Hasegawa, O. A detection technique for degraded face images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; pp. 1506–1512.
30. Brunelli, R.; Poggio, T. Face recognition: Feature versus templates. *IEEE Trans. Pattern Anal. Mach. Intell.* **1993**, *15*, 1042–1052.
31. Gritti, T.; Shan, C.; Jeanne, V.; Braspenning, R. Local features based facial expression recognition with face registration errors. In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, Amsterdam, The Netherlands, 17–19 September 2008; pp. 1–8.
32. Belhumeur, P.N.; Hespanha, J.P.; Kriegman, D.J. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1997**, *19*, 711–720.
33. Geng, C.; Jiang, X. Fully automatic face recognition framework based on local and global features. *Mach. Vis. Appl.* **2013**, *24*, 537–549.
34. Ringner, M. What is principal component analysis? *Nat. Biotechnol.* **2008**, *26*, 303–304.
35. Welling, M. *Fisher Linear Discriminant Analysis;* Department of Computer Science, University of Toronto: Toronto, ON, Canada, 2005.
36. Yang, W.-D. A Simple Approach to a Small-Scaled Face Recognition System. Master's Thesis, Department of Computer Science and Information Engineering, National Central University, Taoyuan City, Taiwan, June 2007.
37. Maturana, D.; Mery, D.; Soto, A. Face recognition with local binary patterns, spatial pyramid histograms and naive Bayes nearest neighbor classification. In Proceedings of the IEEE International Conference of the Chilean Computer Science Society, Santiago, Chile, 10–12 November 2009; pp. 125–132.

38. AT&T The Database of Faces. Available online: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html (accessed on 5 April 2018).
39. Chen, L.-W.; Ho, Y.-F.; Li, Y.-E. An augmented reality based social networking system for mobile users using smartphones. In Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), Hangzhou, China, 22–25 June 2015.
40. Open Source Computer Vision (OpenCV). Available online: http://opencv.org (accessed on 5 April 2018).
41. Lambda Labs Face Recognition API. Available online: http://api.lambdal.com/ (accessed on 5 April 2018).
42. Mashape—The Cloud API Hub. Available online: https://www.mashape.com/ (accessed on 5 April 2018).