國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Master Thesis

基於可適性傳輸與情境感知之自動化活動分類

Context-Aware Activity Classification with Adaptive Transmission

林佳志

Chia-Chih Lin

指導教授：陳銘憲 博士

Advisor: Ming-Syan Chen, Ph.D.

中華民國 104 年 9 月

September, 2015

國立臺灣大學碩士學位論文
口試委員會審定書

基於可適性傳輸與情境感知之自動化活動分類
Context-Aware Activity Classification with Adaptive Transmission

本論文係 林佳志 君（R01942038）在國立臺灣大學電信工程學研究所完成之碩士學位論文，於民國 104 年 9 月 18 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

（簽名）

（指導教授）

所　　長　（簽名）

Acknowledgement

I would like to express my great gratitude to Professor Ming-Syan Chen, my advisor, for his constant encouragement and guidance. He always inspires me with his great wisdom and sense of humor. Without his kind and patient guidance, this thesis would not have finished.

I would also like to express my gratitude to Prof. Pu-Jen Cheng, Meng Chang Chen and Dr. Jian-Chih Ou for serving on my committee. They have provided me with valuable suggestions which assist me in improving my thesis.

Moreover, I would like to thank all of the members of Network Database Lab. Thanks to my senior, Chung-Kuang Chou, Hong-Han Shuai, Chih-Ya Shen, Heng-Yu Chi who would always discuss with me and give me suggestions when I met difficulties. Thanks to Yu-Fen Chen, Su-Chen Lin, Yi-Ling, Chun-Chieh Chen, Ya-Wen Teng, Chen-Ying Liu, Jhao-Yin Li, Chi-Hsuan Wu, Li-Yen Kuo, Jung-En Wang, Yu-Lin Chien, Fan-His Wang, Kuo-Feng Hsu, Chia-Lun Wu, Pei-Lun Liao and Han-Ching Ou for sharing the joy, pain and food during this period of time.

Finally, I owe my deepest gratitude to my friends and family for their support and encouragement. Thanks to Chih-Hsiang Cheng who always gives me strength to move forward. Most importantly, thanks to my father who supports me the most. I would like to dedicate the thesis to him.

中文摘要

隨著各類型的穿戴式以及行動裝置不斷推陳出新，應用程式也漸趨複雜化。考量裝置資源有限，大多數現行應用程式採取雲端運算之策略將資料上傳以分攤裝置工作量；然而對於使用者而言，頻寬使用量也是必須斤斤計較的重要資源，特別在資訊量爆發的現代，傳統應用將資料全數上傳的方式將造成使用者龐大負擔而導致使用意願全失。因此，本文擬提出一套考量頻寬使用量之新型態之應用服務架構，讓應用程式得以選擇是否藉由犧牲少部分準確度以大幅降低頻寬使用量。

除此之外，我們將此架構實作於一個需要處理大量影像且長時間運行的應用服務上以驗證想法之可行性。此應用由穿戴式裝置、行動裝置與遠端伺服器組成，旨在將收集使用者日常生活訊息並自動生成一個每日圖文活動摘要。使用者將穿戴攝影裝置週期性拍攝日常生活，接著行動裝置將進行活動辨識以及篩選出具有代表性的生活照，而當行動裝置無法辨別某張照片是否具有代表性時，該照片將上傳至伺服器端進行進階分析。我們雇用 4 名使用者進行為期 14 天的實驗，其實驗結果證實透過可適性傳輸機制，行動端與伺服器端得以在選代表性照片的任務中藉由犧牲少部分精確度達成降低大量頻寬使用量之目的，大幅增加應用服務之彈性。

關鍵字：穿戴式裝置、穿戴式攝影機、系統架構設計、行動服務應用、可適性傳輸

ABSTRACT

With the growth of innovative wearable and mobile devices, smart applications in daily life become more complicated. Most of these applications offload all data from wearable and mobile devices to remote servers to overcome the limitations of device resources. However, offloading all the data, especially multimedia contents, requires a large number of network resources and may result in the dissatisfaction of users who use such applications. To alleviate the problem, we propose a practical system architecture which includes an adaptive transmission mechanism to reduce the network bandwidth usage. We design and implement a multimedia application, which generates a diary-like daily activity summarization, with the proposed system architecture to verify the feasibility. In the experiment with four participants wearing the wearable camera for fourteen days, the results show that over 89% of the overall bandwidth usage can be reduced with sacrificing 11% of the server-side performance via the proposed adaptive transmission mechanism.

Index terms – Wearable device, Wearable camera, Mobile app, Adaptive transmission

CONTENTS

LIST OF FIGURES

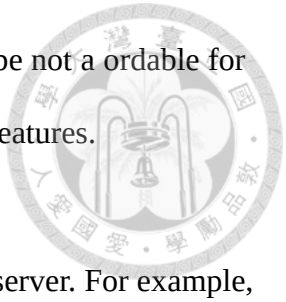
LIST OF TABLES

Chapter 1 Introduction

As the concept of the Internet of things becomes popular, many innovative wearable and mobile devices have been released recently, e.g. Google Glass, Apple Watch. To satisfy various user demands, the complexity of smart applications associated with the devices has increased noticeably. Unfortunately, wearable and mobile devices usually lack hardware resources required for complicated computational task. Recent studies [17, 18] in resource saving issues on mobile devices propose dynamic execution mechanisms. Nevertheless, bandwidth usage saving between mobile devices and servers for a satisfying application is still an open issue. Consuming too much bandwidth may result in the dissatisfaction of using an application since offloading data to servers leads to high network latency and thus user experience may be deteriorated, especially when network connection is low speed or unstable. More bandwidth usage also probably implies that there is more money users need pay to Internet service providers. Therefore, a system architecture design for wearable and mobile applications to reduce the bandwidth usage with the guarantee of satisfactory is necessary. Several ways may be used to save bandwidth usage between wearable/mobile devices and servers for smart applications. We discuss them as follows, assuming that a smart application uses a machine learning based prediction algorithm.

1. **Extracting features on wearable/mobile devices.**

   The features for prediction tasks can be extracted on the devices and the device only upload these features, instead of full raw data, to the server. For example, when a application is dependent on temperature, the device can transmit average temperature in a time window instead of values of every measures. However,



computational costs of some feature extraction techniques may be not a ordable for wearable and mobile devices, especially for images and videos features.

2. **Offoading the raw data periodically.**

The raw data may be periodically collected and offoaded to the server. For example, GPS location data may not be necessary to be updated frequently since a person tends to stay in the same region in a short period of time. Nevertheless, periodically offloading images and videos to the server still costs much bandwidth usage.

3. **Selectively offloading the raw data.**

Suppose that some filter processes are adopted in the wearable and mobile devices and only data of confusing predictions are offloaded to the server, the bandwidth usage will be reduced significantly. For instance, in a navigation task, if a user is still in the same route, the application does not need to ask the server to give a new route.

In this work, we design and implement a multimedia application and further propose a mechanism of adaptive transmission, based on the concept of selectively offloading the raw data, to save network bandwidth usage. The purpose of the application is to automatically generate a diary-like personal daily activity summarization. The hardware of the application consists of a wearable device with a camera, a mobile device and a remote server. The application recognizes a user's daily activities based on context-aware features and identifies several representative images from the daily image collection automatically taken by the wearable camera. Fig 1 illustrates the result of daily activity summarization in our scenario. With meaningful images, a personal activity summarization can help a person easily recall what he or she has experienced in the life. To evaluate the feasibility of the proposed system

architecture, we recruited 4 participants for the experiment. Each participant wore the wearable camera for 14 days, collecting total 26073 images and 888 activities. The results show that more than 89% of the overall bandwidth usage can be saved with only sacrificing 11% of the server side performance via the proposed adaptive transmission.

The rest of the paper is organized as follows. Chapter 2 discusses the related works. In Chapter 3, we introduce our system architecture and the machanism of adaptive transmission. Chapter 4 presents experiment settings and results. Finally, Chapter 5 concludes the work.

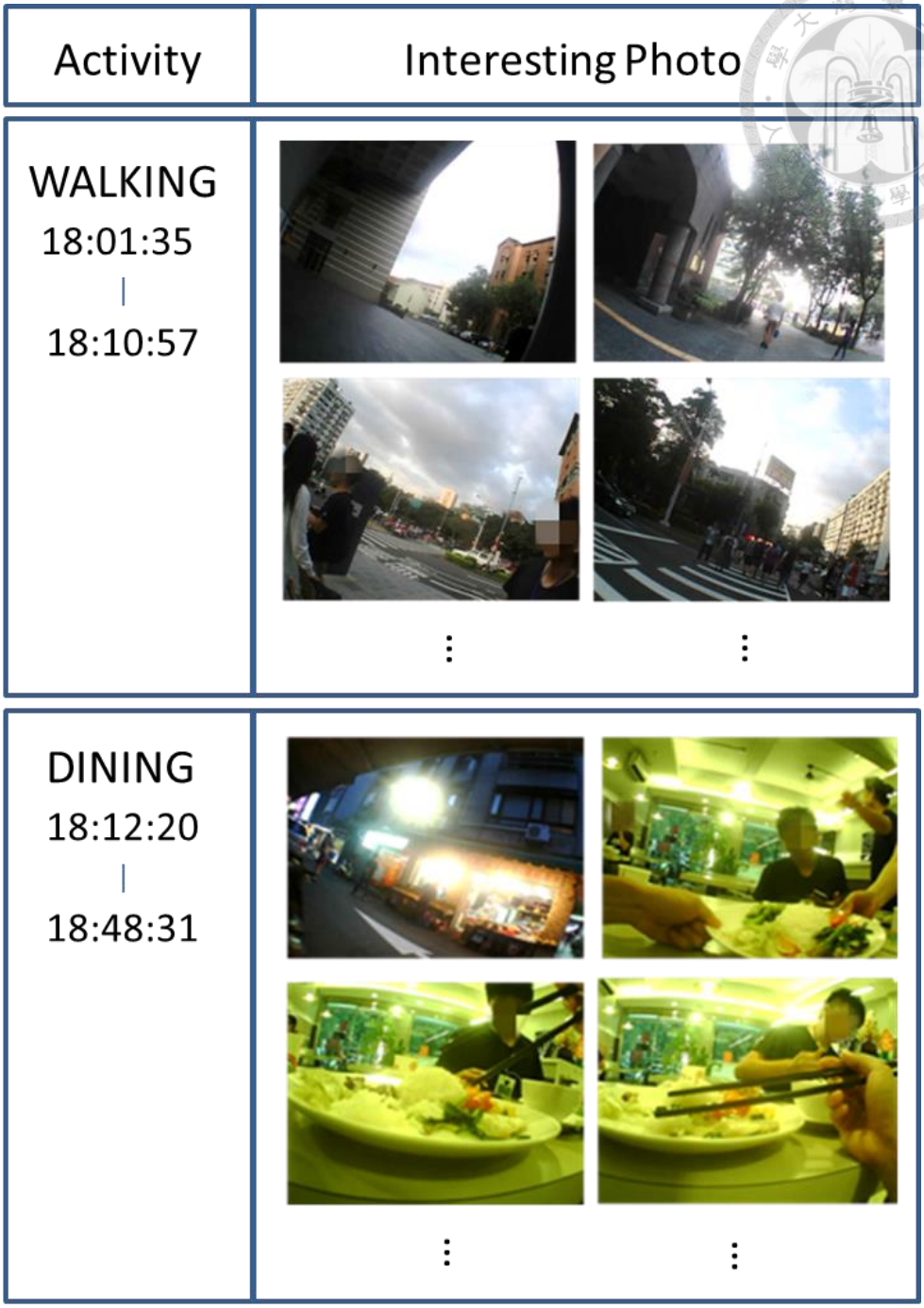


Fig. 1 An example of daily activity summarization

Chapter 2 Related Work

In this chapter, we briefly review works on bandwidth saving, activity recognition and image selection.

2.1 Bandwidth Saving

Several studies have investigated communication costs of offloading data from different points of view. Barbera et al. [16] measured communication costs in terms of energy and bandwidth consumption in the real scenario. Naqvi et al. [23] examined the trade-off between resources utilization and performance for augmented reality application on mobile devices. Tekin et al. [15] focused on solving the latency problem and proposed a method which offloads data to a powerful machine in the proximity of mobile device. Hyytia et al. [14] proposed optimal strategies with multidimensional considerations such as availability of hotspots, expected latency and energy consumption to offload data dynamically. Ha et al. [11] proposed a novel application assisting patients who suffer from serious cognitive decline and implemented it with Google Glass. They used complicated computer vision algorithms and thus offloading data to the backend cloud is necessary. However, the discussion of reducing bandwidth usage between mobile devices and remote servers is still an open issue. In this work, we propose a mechanism of adaptive transmission to reduce the bandwidth usage with acceptable accuracy with the verification by a simulated-approach experiment on real data.

2.2 Activity Recognition and Image Selection

The objective of our application is to recognize the activities and select the meaningful images in daily life. Activity recognition can be classified into three major

categories, visual-based method [1, 2, 3, 4, 6, 7, 12], sensor-based method [13] and hybrid method [5] mixing both the above two. Due to the limitation of the mobile and wearable device, our application recognizes the activity by sensor data. Shoaib et al. [13] gives a comprehensive overview of activity recognition using smart phones. Based on the previous studies, we collect sensor data and extract the context-aware features on smart phones to predict the activity. On the other hand, selecting representative images is the other part in our application and some of the related studies have been focus on it [8, 9, 10]. However, the algorithms in the previous works require a clustering method, which is not suitable for resource-limited mobile or wearable devices. Alternatively, we handle the photo selection problem as an image retrieval problem by considering the past data as an interest set and extract some simple image features from the set to train a classifier. Furthermore, we propose a mechanism of adaptive transmission for image classification to save huge bandwidth usage with only sacrificing some performance.

Chapter 3 Proposed System Architecture

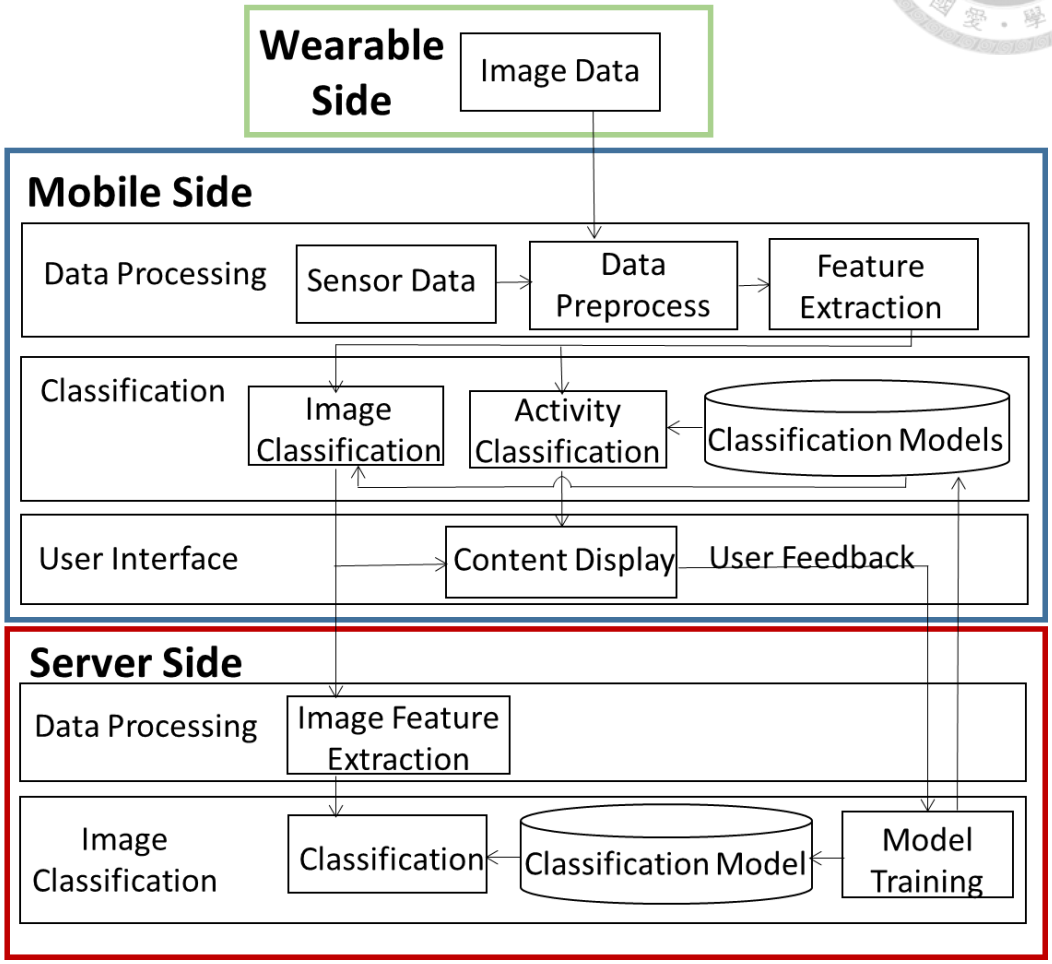


Fig. 2 System architecture

In this chapter, we first introduce the functionalities of the components in the system architecture and then discuss the mechanism of our proposed adaptive transmission.

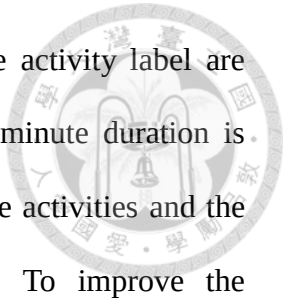
3.1 System Architecture

Our system architecture is shown as Fig 2, which consists of a wearable device, a mobile device and a remote server. The daily activity summarization consists of human activity recognition and representative image classification. The application extracts the

features from images and sensor data. Afterwards, activity recognition is done by an activity classifier in the mobile device and image classification is done by the cooperation of two image classifiers in the mobile device and the server.

The wearable device is only in charge of photo collection due to resource limitations. The system capabilities of a wearable device such as processor performance, memory size and storage capacity are usually not so critical for a user. Instead, a user concerns more about the features, such as small size, lightweight, long battery life, comfort and aesthetics. Therefore, offloading the images to a more powerful and nearby machine, i.e. the mobile device in our application, is a better solution. The wearable device periodically takes an image and offloads the image to the mobile device via Bluetooth for further processing. Before uploading the image to the mobile device, the wearable device stores it in an SD card to prevent connection failures.

The mobile device is mainly responsible for sensor data collection, data processing, activity recognition, image classification and content displaying. We collect various sensor data, such as accelerometer, gyroscope, GPS, and light sensor, to extract context-aware features at a fixed sampling rate. These context-aware features are also considered as metadata of an image in the image classification. Afterwards, the mobile device conducts preprocessing to remove noisy data and extracts simple features from the images and sensor data. Subsequently, the activity recognition and the image classification are conducted based on the extracted features. A time segmentation with a fixed time window, i.e. 5 minutes, is applied to the data. For each 5-minute group, an activity is recognized on sensor features and each image in the 5-minute group is predicted as representative or unrepresentative. Based on the result of the image classifier, some of the images in each 5-minute group are left in the mobile device, some of the images are dropped, and the rest of the images are transmitted to the server for



further analysis (Chapter 3.2). Two adjacent groups with the same activity label are merged to one iteratively, for example, a dining activity with 30-minute duration is merged from 6 adjacent 5-minute dining activities. Finally, all of the activities and the corresponding representative images are displayed to the user. To improve the performance, the mobile device sends the user feedback to the server after the user corrects the contents through the user interface.

The server is in charge of further complicated image analysis and model training. After receiving an image from the mobile device, the server extracts complicated features, e.g. Scale-Invariant Feature Transform (SIFT), from the image in order to make a more precise prediction. A classifier in the server side determines if the image is representative again. Subsequently, the server sends back the result to the mobile device. On the other hand, model training is conducted after receiving the user feedback or in the initial stage. All of the classifiers, i.e. one for image classification in the server, two for image classification and activity recognition in the mobile device, are trained and updated to improve the performance.

3.2 Adaptive Transmission

The mobile device only transmits images adaptively because the network overhead caused by offloading images is very critical. Inspired by the concept of the reject region presented in Fig 3 in decision theory [25], each prediction made by the image classifier in the mobile side comes with a confidence score that is used to determine if the prediction is rejected. The confidence score within a certain threshold interval, i.e. reject region, implies an unreliable prediction, the mobile device then offloads the corresponding image to the server no matter if the prediction is positive or negative. On the other hand, if the confidence score is not in the threshold interval

(confident), and the corresponding prediction is positive, the mobile device displays the image on the user interface directly. If the confidence score is confident and the prediction is negative, the image is dropped.

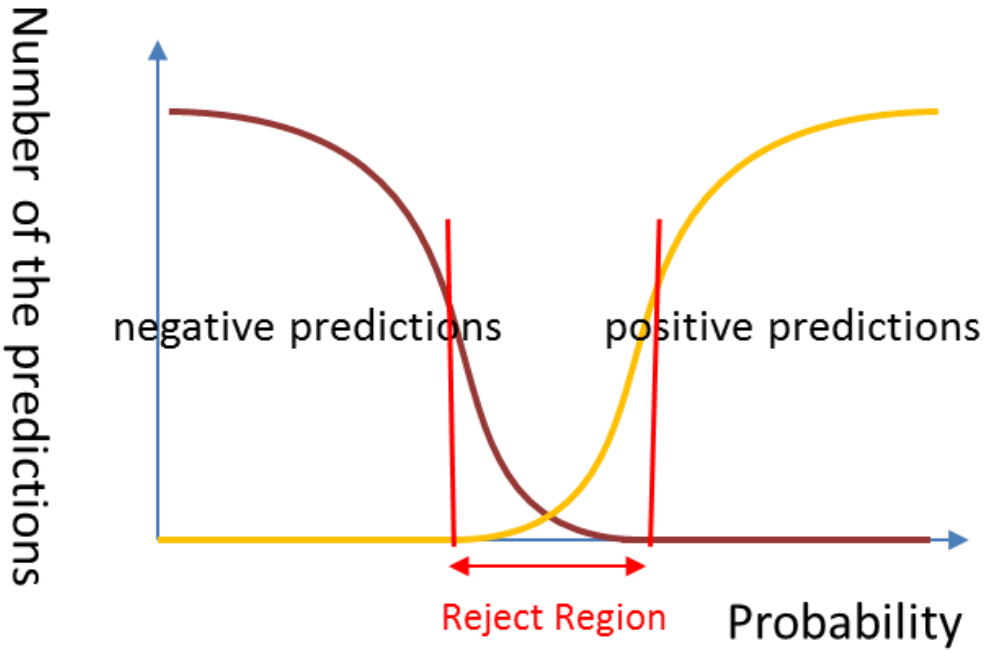


Fig. 3 Concept of the reject region in the two-class classification

Specifically, suppose that the value p returned by the image classifier $f(x)$ is from 0 to 1 for the binary image classification. Note that even if a classifier does not support the probabilistic output, the output can be transformed by Platt Scaling [26]. If the output p is higher than 0.5, the image is regarded as positive; otherwise, the image is regarded as negative. Given a confidence threshold ct, a negative confidence threshold nct is calculated as $1-ct$. A positive prediction for the corresponding p lower than ct or a negative prediction for the corresponding p higher than nct is considered as an unreliable result. That is, the mobile device offloads the image to the server if and only if p is within the interval, i.e. $nct < p < ct$. A summary of the adaptive transmission in the mobile side is provided as Algorithm 1. Note that in general,

the boundary of the threshold value can either be fixed or dynamically adjusted. Adjusting the threshold results in a trade-off between bandwidth usage and prediction accuracy. Assuming that the server makes the best performance, offloading more photos will improve more accuracy in image classification but bandwidth usage and latency are increased relatively as well. The further optimization strategy should consider user acceptance or the scenario of the application itself.

Algorithm 1 Adaptive Transmission algorithm

Require: $0 \leq out, ct \leq 1$

1: $nct \leftarrow 1 - ct$
2: **if** $out > ct$ **then**
3: display the image
4: **else if** $out < nct$ **then**
5: drop the image
6: **else if** $nct \leq out \leq ct$ **then**
7: upload the image to the server
8: **end if**

Algorithm 1. Adaptive transmission algorithm

Potentials of the System Architecture. The proposed architecture can be utilized in different scenarios where imperfect predictions are acceptable. For instance, a personal recommendation service provides information for’’ What do we have for dinner? " or “ Where can I go shopping after 10 minutes? ". In this case, the wearable and mobile devices collect images and context-aware information from the environment and then the mobile device gives a result based on the prediction by the mobile side classifier that considers user preference in the past. If the result is not confident, the mobile device offloads raw data to the server for further processing, which is the same as our proposed approach. Moreover, the proposed system architecture can be extended to the case that

multiple wearable and mobile devices exist where one or several devices can be coordinators and our proposed adaptive transmission is still applicable between them and the server.

Chapter 4 Experiment

To verify the feasibility of the adaptive transmission, we conducted an experiment in the real world. We first describe the settings and then present the results in the chapter.

4.1 Experiment Settings

We first detail our hardware and the data collection, and then describe the model and the features utilized in our application as Table. 1 shown.

4.1.1 Hardware and Data Collection

We use a Linkit ONE board, an Arduino based development board by MediaTek, with a serial camera to build the prototype of the wearable camera as show in Fig 4. The serial camera provides a 640*480 resolution and a UART baud rate from 9600 to 115200 to take a picture in JPEG format. To communicate with the mobile device, Bluetooth 2.1 is implemented in the Linkit ONE board and a 16 GB SD card is utilized as a buffer to prevent the connection failure. Due to the limitation of the battery life of the wearable device, a power bank is used. Besides, we develop an app to control the wearable camera on an android based smart phone and a Windows PC with 2.5GHz Intel core CPU and 8GB RAM is utilized as the server.

Four participants were recruited for the experiment. Each participant wore the wearable camera in front of the chest for 14 days and the camera took pictures at least 8 hours in each day. The wearable device took a picture every 40 seconds on average and the mobile device recorded the sensor data every 10 seconds during the experiment. Seven types of the activity are involved in the experiment, namely working, walking, resting, recreation, dining, commuting and others. All of the participants had to select

several interesting images and labeled the activities that had been experienced after finishing the experiment every day. Note that the participants were allowed to turn off the camera when they had any privacy concerns or felt inconvenient in some situations. In addition, the participants were allowed not to select the images if he or she did not find any interesting image in the daily life. For example, the user might work at home for a whole day that he or she might be unwilling to select any photos.

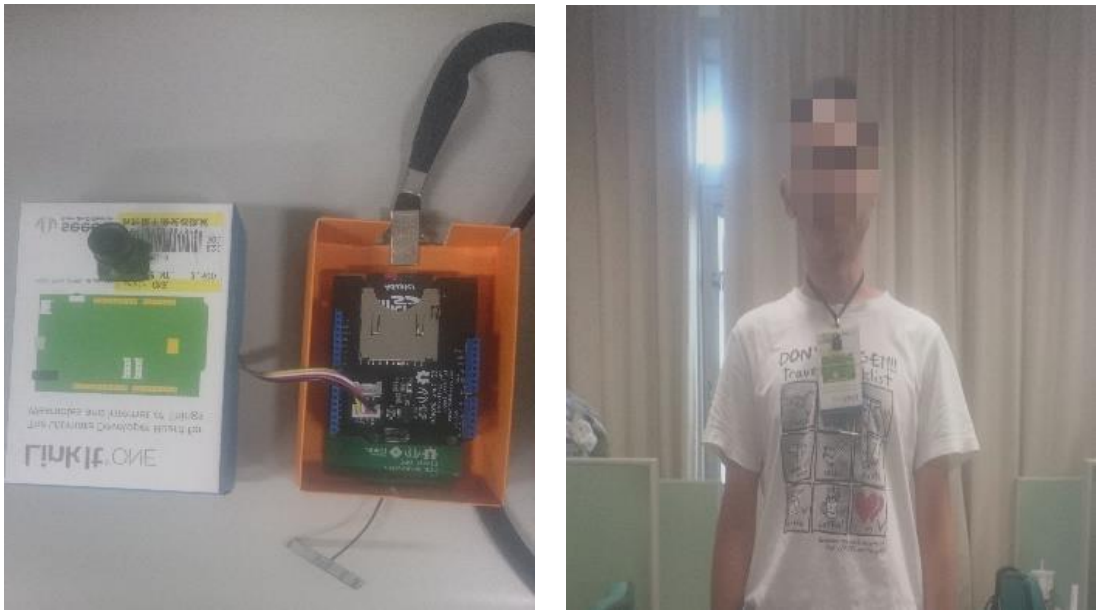


Fig. 4 (a) Wearable device prototype (b) Device wearing demonstration

4.1.2 Mobile Features

Context-aware features are collected by the mobile device, including time-related features, location-related features, and device status features. Time-related and location-related features are useful features due to the spatial-temporal regularity of human activities. For example, some of the people have to work in the office on weekdays. Date, time, day of week, weekdays and the weekend are considered as time-related features and latitude, longitude, altitude, speed, stay points are collected as location-related features. Stay points of each user are extracted based on the GPS

trajectories. We adopted the method proposed by Li et al. [21]. A stay point is defined as a region user stayed for a while and a location indicator is used to identify if a picture was taken within a stay point. Assume a sequence of GPS records $\mathrm{P} = \{P_n P_{n+1}, \ldots P_m\}$ and each P_i consists of a location L_i (longitude and latitude) and a timestamp T_i , if the distance $\mathrm{D} = L_m - L_n \leqq D_{threshold}$ and the timestamp $\mathrm{T} = T_m - T_n \geqq T_{threshold}$, then a stay point S is extracted by average all the location information in P , that is:

$$S_{longitude} = \sum_{i=n}^{m} \frac{L_{i,longitude}}{n+m}$$

$$S_{latitude} = \sum_{i=n}^{m} \frac{L_{i,latiude}}{n+m}$$

We set the distance threshold to 200 m and time threshold to 30 minutes for each of the stay point, representing user is staying or wandering around a stay point within 200 meters over 30 minutes. A location indicator is used to identify if a picture is taken within a stay point. If a picture is taken within a stay point, the location indicator is set to 1, otherwise the location indicator is set to 0. As an on-body device, status of the smart phone reveals the implicit user behavior pattern and the surrounding environment. We consider several binary and numeric features to provide context awareness, including whether the screen of the mobile device is on or off, whether Wi-Fi or 4G is connected, whether the network connection is fast, whether the mobile device is charging, the number of Wi-Fi access points, accelerometer sensor value and ambient brightness.

4.1.3 Image Features

Image features are utilized to select representative images for users. We implement the image feature extraction using Open CV, an open source library for computer vision. To figure out the implicit user preference from the labeled data, low-level features including color, edge and SIFT are extracted from the image. First, RGB and HSV color histograms with total 150 bins are constructed to obtain global color distribution of the image in both mobile and server sides. Besides, to obtain edge distribution of the image, edge histogram descriptor (EHD) standardized in MPEG-7 [22] is extracted in the server. EHD divides the image into 4X4 equal-sized sub-images and categorizes edges into five types for each sub-images, namely horizontal, vertical, 45-degree diagonal, 135-degree diagonal and non-directional edges. Thus, edge histogram with 80 bins (16X5) is generated for each image. Finally, SIFT is extracted on the server side for further analysis. Due to the scale and rotational invariant characteristics, SIFT is a popular algorithm and commonly used to recognize an object, track the motion or compare the similarity between two images. However, unlike color or edge features using a specific feature space to generate histogram, SIFT detects different number of the blob-like local key points for each image and describes each of the key points with a specific corresponding descriptor. Therefore, we adopt the bag-of-features approach (BoF) to deal with the SIFT features [20]. Yuan et al. have been explored the size of codebook from 50 to 250 [19]. However, due to the content of pictures taken in daily life are diverse, the number of code words is set to 5000 in our application.

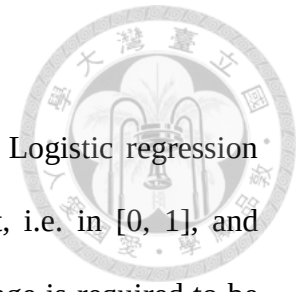
Table 1 Features Table

Category	Attribute Name	Possible Value
Time-related	Month	[1,12]
	Day	[1,31]
	Hour	[0,23]
	Minute	[0,59]
	Second	[0,59]
	is Mon	0,1
	Is Tue	0,1
	Is Wed	0,1
	Is Thr	0,1
	Is Fri	0,1
	Is Sat	0,1
	Is Weekend	0,1

Category	Attribute Name	Possible Value
Location-related	Latitude	[-90,90]
	Longitude	[-180,180]
	Altitude	R
	Speed	R
	Is GPS (Location Provider)	0,1

Category	Attribute Name	Possible Value
Device-related	Accelerometer X	R
	Accelerometer Y	R
	Accelerometer Z	R
	Is Screen On	1,0
	Is Charging	1,0
	Luminance	R
	Connect to Wi-Fi	1,0
	Connect to 3G/4G	1,0
	Is Connecting Fast	1,0
	Number of Wi-Fi AP	R

Category	Attribute Name	Possible Value
Image Features	Color histogram(RGB,75)	R
	Color histogram(HSV,75)	R
	EHD (80)	R
	BoF-SIFT (5000)	R



4.1.4 Classification Model

We use logistic regression [24] as the two image classifiers. Logistic regression provides a probabilistic output value for each classification result, i.e. in [0, 1], and hence a fixed threshold value can be used to decide whether an image is required to be offloaded. On the other hand, a multi-labeled version of logistic regression is applied to activity recognition. Given a set of data after time segmentation, the multi-labeled classifier considers all of the activities and selects the activity with the highest possibility as the prediction result. We build all of the classifiers by Weka, a well-known data mining tool.

4.2 Work Flow of the Application

In this chapter, we show the complete work flow of the application. Since real-time service is not required in our scenario, the application remains the flexibility that user can set a timer to generate the daily summarization at a specific time or generate the summarization periodically. Given a set of raw data pairs consisted of mobile records and images, the application first splits raw data pairs ordered by time into several groups with a 5-minute time window during the time segmentation stage. Next, first-stage feature extraction is applied to the mobile device and an activity is predicted based on the patterns found from each 5-minute groups. Afterward, every image in the 5-minute group of data pairs is determined if it is a representative image through the cooperation of the mobile device and the server. Finally, the predicted activities and images will be ordered by time axis and display to user. If the user gives feedback, the refinement stage is then applied to update all of the required models. We shows the complete algorithm in Algorithm. 2.

Algorithm 2 Daily Activity Summarization

Input(*R*, *w*, *thres*, *t*):

Where ***R*** is a raw data set consists of raw data pairs $r_n = (i_n, l_n)$ ordered by time, time window ***w***, confidence threshold ***thres*** and timer ***t*** where in is an image and l_n is a corresponding mobile log

split *R* into groups $G = \{g_1, g_2, \ldots, g_n\}$ by a 5-minute time window ***w***
, where $g_i = \{r_1, r_2, \ldots, r_p\}$ and $(r_p.time - r_1.time) \leq w$
For each g_i, $i \leftarrow 1$ *to* n
  For each r_i, $i \leftarrow 1$ *to* p
    $m_i \leftarrow$ Mobile_Side_Feature_Extraction ($\boldsymbol{r_i}$)
    , where M is a set of instances and M = $\{m_1, m_2, \ldots, m_p\}$
  End for
  act = Activity_Recognition (*M*)
  activityList.add (*act*)
  For each $m_i \in$ M
    $m_i.append(act.type)$
    $mResult \leftarrow$ Mobile_Side_Image_Classification (m_i)
    If ($mResult > 0.5$ && $mResult <$ *thres*) ||
      ($mResult < 0.5$ && $mResult > 1$ - *thres*)**Then**
      offload r_i to server
      $s_i \leftarrow$ Server_Side_Feature_Extraction (r_i), where s_i is a instance
      $sResult \leftarrow$ Server_Side_Image_Classification (s_i)
      send $sResult$ to mobile device
      If $sResult > 0.5$ **Then**
        ImageList.add (i_i)
      End if
    End if
    Else if $mResult > 0.5$ && $mResult >$ *thres* **Then**
      ImageList.add (i_i)
    End else if
  End for
End for
If *system_time* == *t* **Then**
  For each *act* $\in$ *activityList*
    merge every two adjacent *act* iteratively if they are the same
  End for
  display (imageList, *activityList*)
  Extract list of stay point from GPS trajectories
End if
If *user_feedback* == *true* **Then**
  upload data to server
  train all models
  create new bag-of-feature
  update new model
End if

4.3 Experiment Result

26073 images and 888 activities are collected in the 14 days. The representative images account for about 3% of the total images. We show the proportion of each activity types and the number of total images and interesting images collected from each user in Fig.5 and Table 2 respectively.

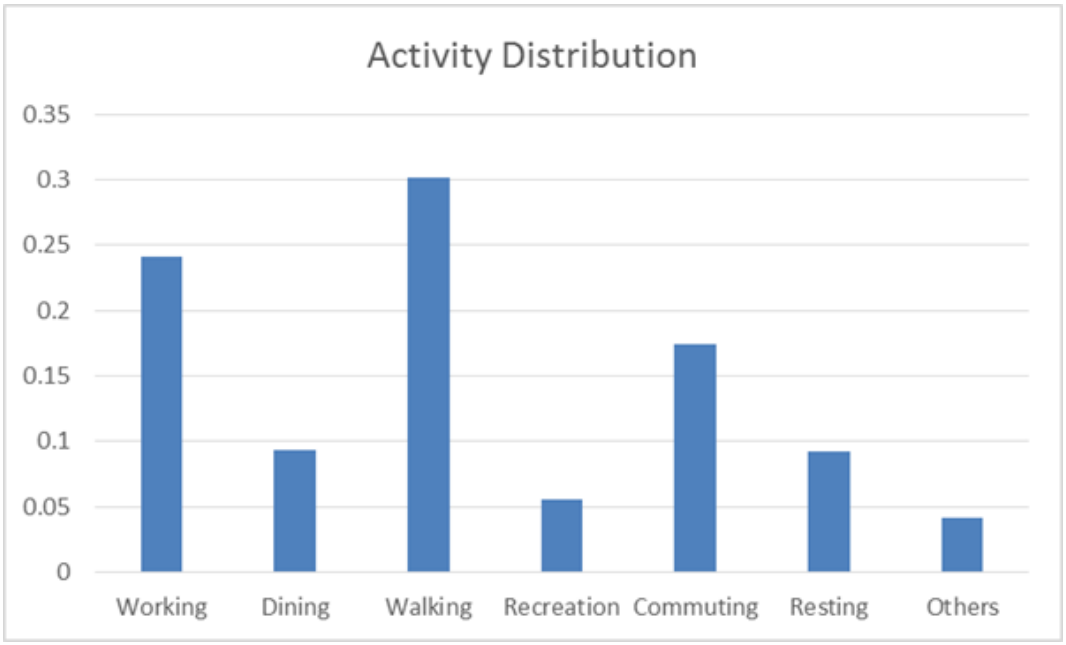


Fig. 5 Activity Distribution

Table 2 Number of Images collected from each user

User No.	1	2	3	4
Total images	8121	8162	5687	4103
Interesting images	292	256	191	141

4.3.1 Performance Metrics

We use accuracy, precision, recall and F1-score as the performance metrics for the activity recognition and the image classification. The Accuracy is calculated as: $Accuracy = \frac{C}{N}$, where C is the number of the instances being correctly classified and N is the total number of the instances. Precision, recall and F1-score are commonly

used to measure the performance of the classification task. Assume the number of true positive instance is T_p, false positive is F_p, and false negative is F_n ,then precision, recall and F1-score are defined as follow:

$$Precision = \frac{Tp}{Tp + Fp}$$

$$Recall = \frac{Tp}{Tp + Fn}$$

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

4.3.2 Activity Recognition

The application recognizes an activity for each of 5-minute time slots in the mobile side. We do not apply the adaptive transmission on activity recognition task because typically the computation cost of processing sensor data is much lower than processing images. Besides, the concern of bandwidth usage is less critical because the network overhead caused by offloading sensor records is much smaller than the overhead caused by offloading images. Therefore, the advantage of the adaptive transmission is not that obvious in our activity recognition task.

The mobile device records sensor data not only every 10 seconds but also as soon as a picture is taken. In other words, we have two set of sensor data with different sampling rate during the experiment. To investigate the performance discrepancy between two sampling strategies, we evaluate activity recognition with the two sets of data as Fig. 6 and Table 3, where the sampling rate of sensor data follows the sampling rate of the images is referred as “sparse”. The result reveals the performance discrepancy between two sampling rates is not significant.

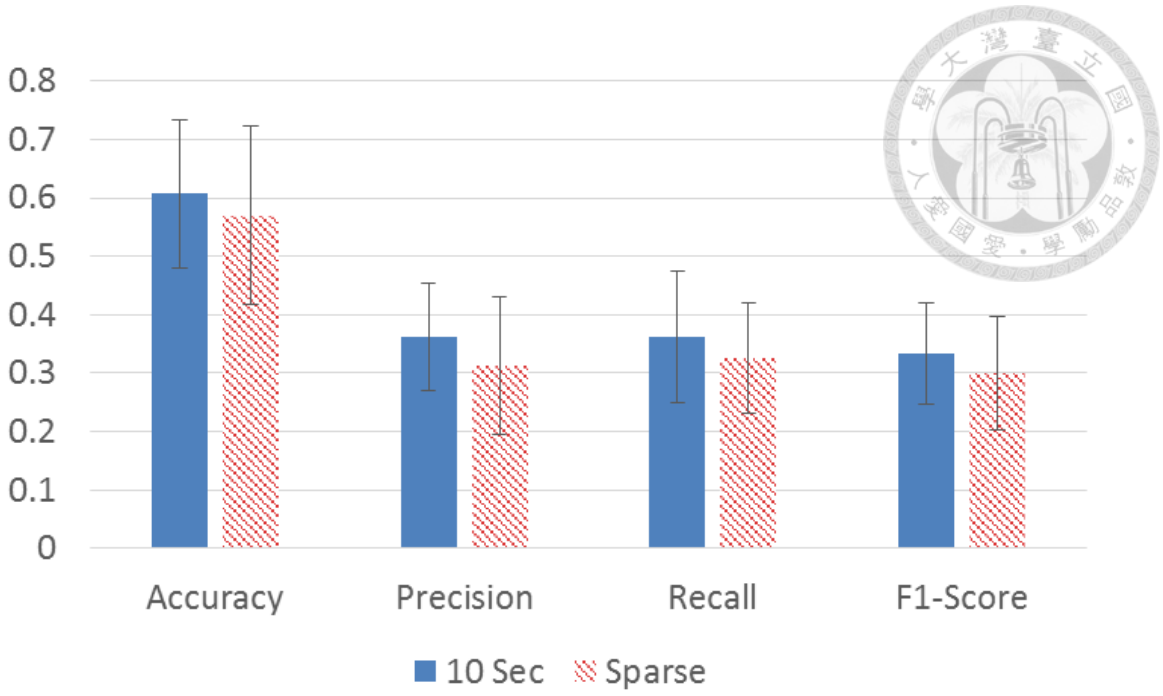


Fig. 6 Performance of activity recognition in different sampling rate

Table 3 Performance of activity recognition for each user

Accuracy(sparse)	0.374	0.7863	0.4954	0.6235
Precision	0.193362	0.500185	0.236526	0.317975
Recall	0.24921	0.472492	0.23861	0.343377
F1-Score	0.201112	0.451686	0.229411	0.315153
Accuracy(10 sec)	0.4542	0.7979	0.5405	0.6349
Precision	0.281005	0.516068	0.302681	0.351612
Recall	0.271717	0.543572	0.26431	0.371275
F1-Score	0.256515	0.473119	0.268626	0.337613

4.3.3 Image classification

As mentioned in Chapter 4, we use color histograms in the mobile side classifier whereas color histograms, EHD and SIFT are used for the server side classifier. Both two classifiers in mobile and server sides consider context-aware features. We show the performance of two classifiers in the interesting images classification task in Fig. 7. Before we applied the adaptive transmission, the F1-score of the mobile side classifier remains approximately 76 % of the F1-score of the server side classifier. The

discrepancy between considering the naïve features and the more complex features is acceptable.

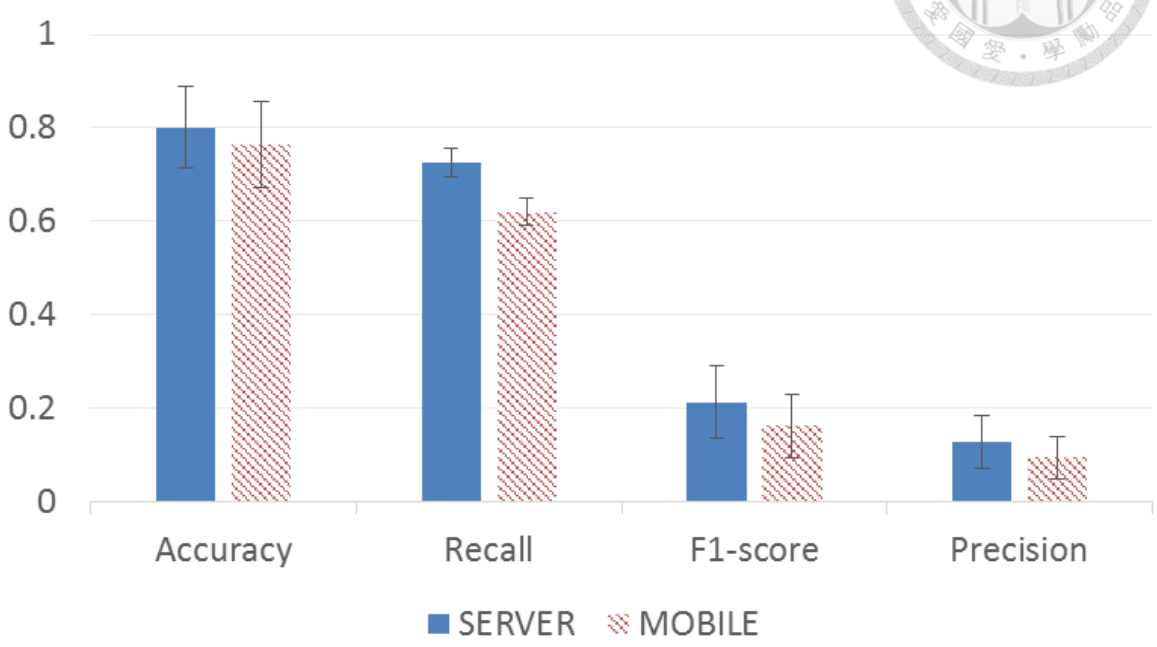


Fig. 7 Performance of image classification in mobile and server sides.

Table 4 Performance of image classification in mobile and server sides.

Performance	Mobile	Server
Accuracy	0.764133	0.801239
Precision	0.09375	0.128
Recall	0.61925	0.72525
F1-score	0.16125	0.21225

Besides, we further investigate two configurations of features used in the server side: (1) removing context-aware features from the server side classifier and (2) adding activity type in server side classifier; the results are shown in Fig. 8 and Fig. 9. To be not confused by incorrect activity prediction, we apply the ground truth of the activity label. 7 binary indicators are used to represent each of the activity. If a record belongs to a certain activity, the corresponding indicator is set to 1. The results in Fig. 8 and Fig. 9 reveal both context-aware features and the activity type are less relevant to the image

classification task.

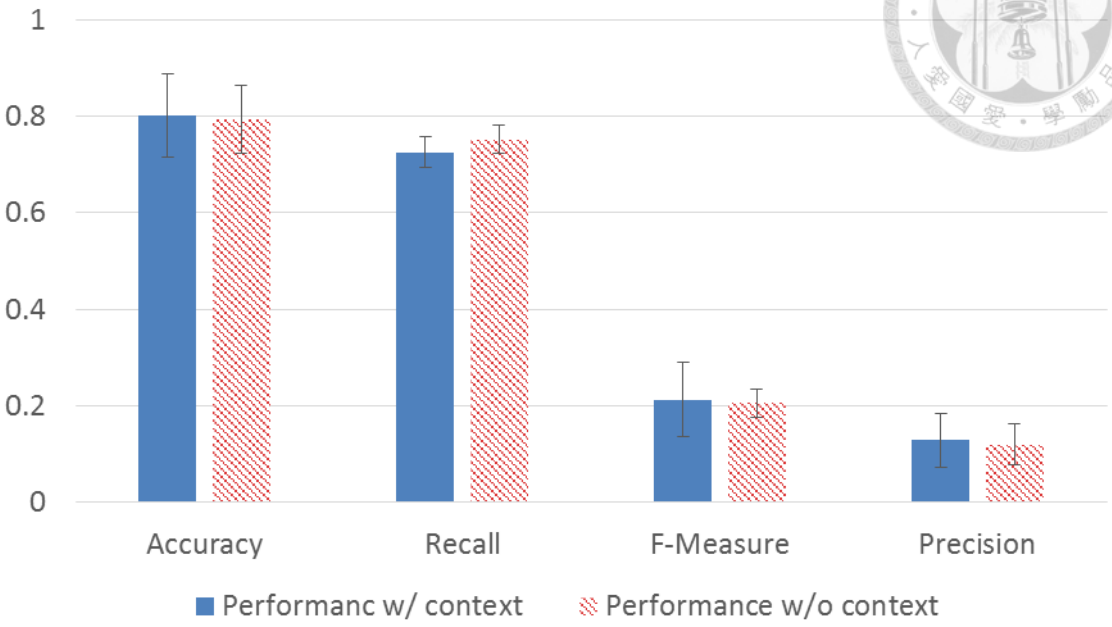


Fig. 8 Performance of image classification with context-aware feature

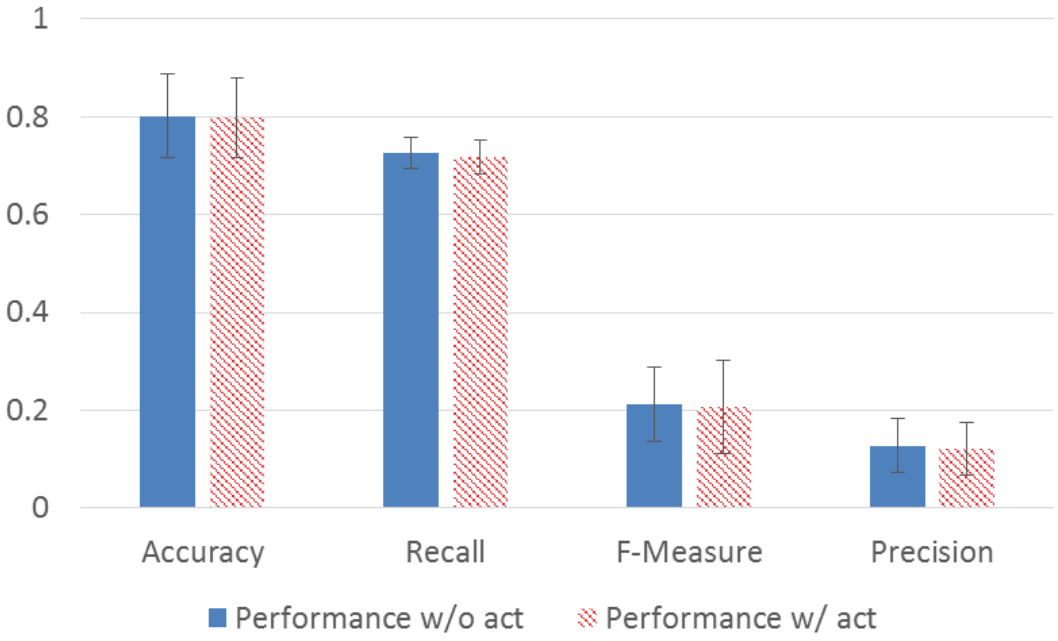


Fig. 9 Performance of image classification with activity type

Table 5 Performance of image classification with context-aware feature

	w/ context	w/o context
Accuracy	0.764	0.801
Precision	0.094	0.128
Recall	0.619	0.725
F1-score	0.161	0.212

Table 6 Performance of image classification with activity

	w/ Activity	w/o Activity
Accuracy	0.798	0.801
Precision	0.121	0.128
Recall	0.718	0.725
F1-score	0.206	0.212

4.3.4 Adaptive Transmission Simulation

To see the relationship between bandwidth usage and corresponding performance in the image classification task, we conducted a simulation using collected real data. In the simulation, different threshold values are set in the mobile side classifier to determine the offloading boundary. Since the output of the image classifiers is from 0 to 1 where 0 represents the image is unimportant and 1 represents the image is interesting, we set the confidence threshold from 0.5 to 1. Fig. 10 and Fig. 11 show the simulation result of adaptive transmission where the performance of each confidence threshold is presented in Fig.10 and the corresponding proportion of the bandwidth usage is shown in Fig.11.

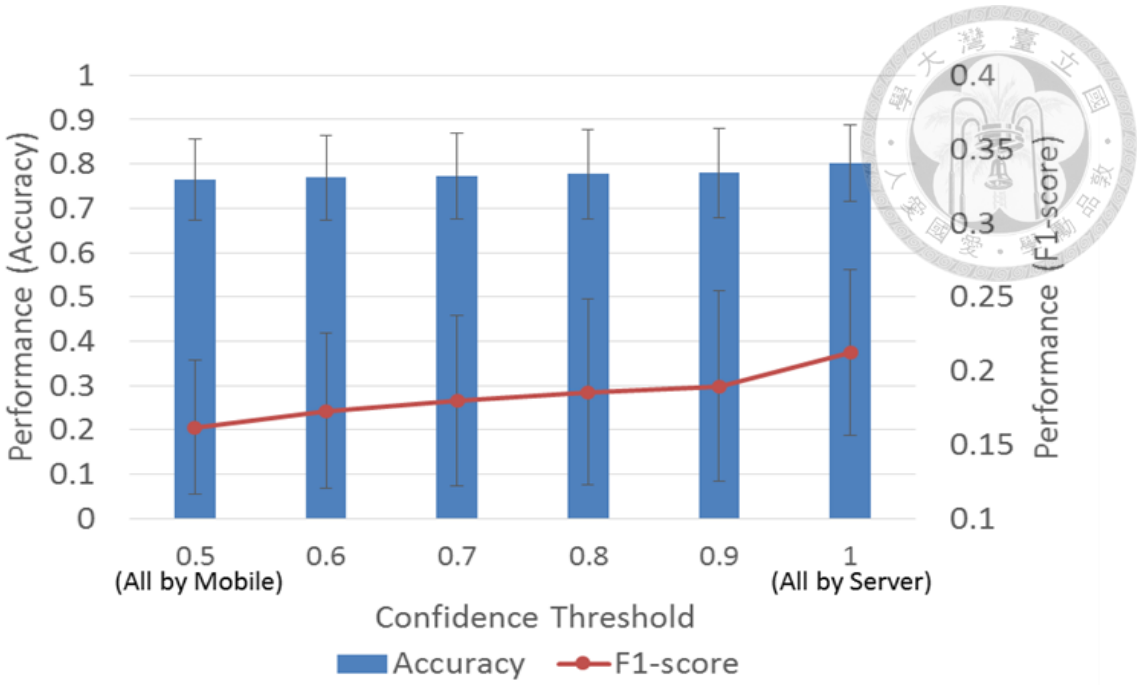


Fig. 10 Performance of adaptive transmission simulation

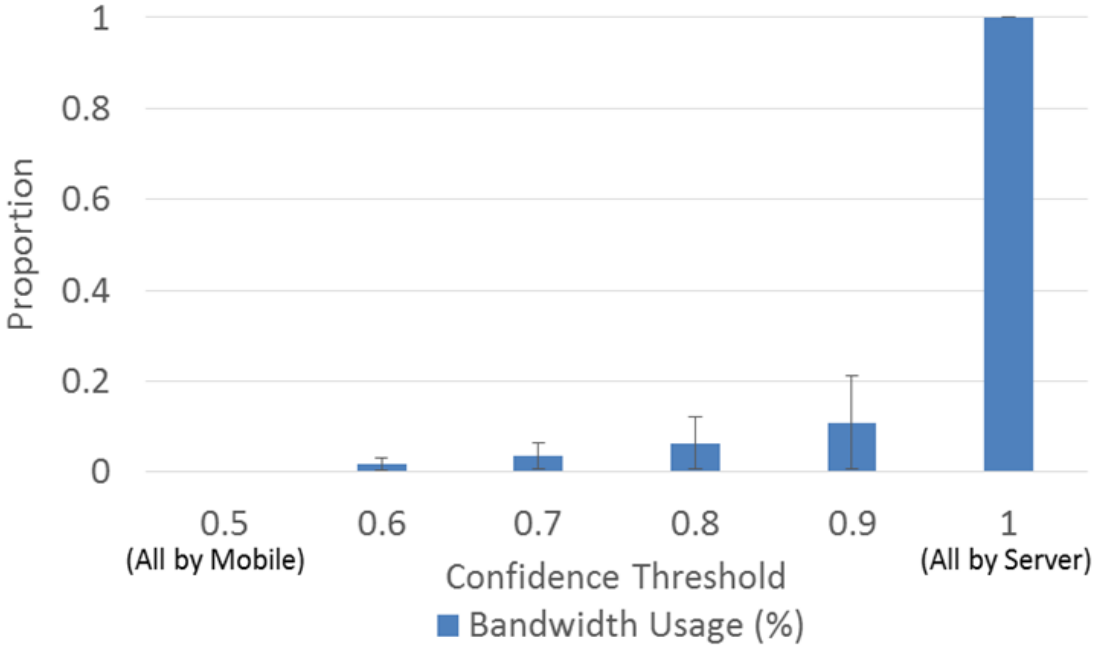


Fig. 11 Proportion of offloading in adaptive transmission simulation

Table 7 Detail of adaptive transmission simulation

Confidence threshold	Accuracy	F-Measure	Classify by Mobile	Classify by Server
0.5	0.764133	0.161602	1	0
0.6	0.768681	0.172745	0.982235077	0.017764923
0.7	0.772645	0.179661	0.964015083	0.035984917
0.8	0.77664	0.185533	0.937343092	0.063477178
0.9	0.77952	0.189486	0.890864455	0.109135545
1	0.801169	0.21205	0	1

Both accuracy and F1-score are improved when the confidence threshold value increases every 0.1. With the growth of the confidence threshold, the overall accuracy and F1-score become closer to the performance of which offloads all the images to the server. When the confidence threshold equals to 0.9, the performance can reach to 87.5 % of server side performance. On the other hand, with the growth of the confidence threshold, the mobile device offloads more instances to the server. When confidence threshold equals to 0.9, the mobile device offloads 11 % of total images to the server. As the expectation, we observe there is a tradeoff between bandwidth usage and performance from Fig. 10 and Fig. 11. However, another question is whether offloading more images to the server gains the same proportion of the performance. We further investigate the performance gain and the corresponding gain of bandwidth usage when the confidence threshold increases every 0.1. Let p_i and b_i be the performance of F1-score and the bandwidth usage of images offloaded to the server at confidence threshold i .The gain ratio is defined as $\frac{p_i - p_{(i-0.1)}}{p_1}$ and the gain ratio of bandwidth usage is $\frac{b_i - b_{(i-0.1)}}{b_1}$. When the confidence threshold moves from 0.7 to 0.9, the performance gain decreases gradually whereas the gain of bandwidth usage increases significantly as shown in Fig 12.

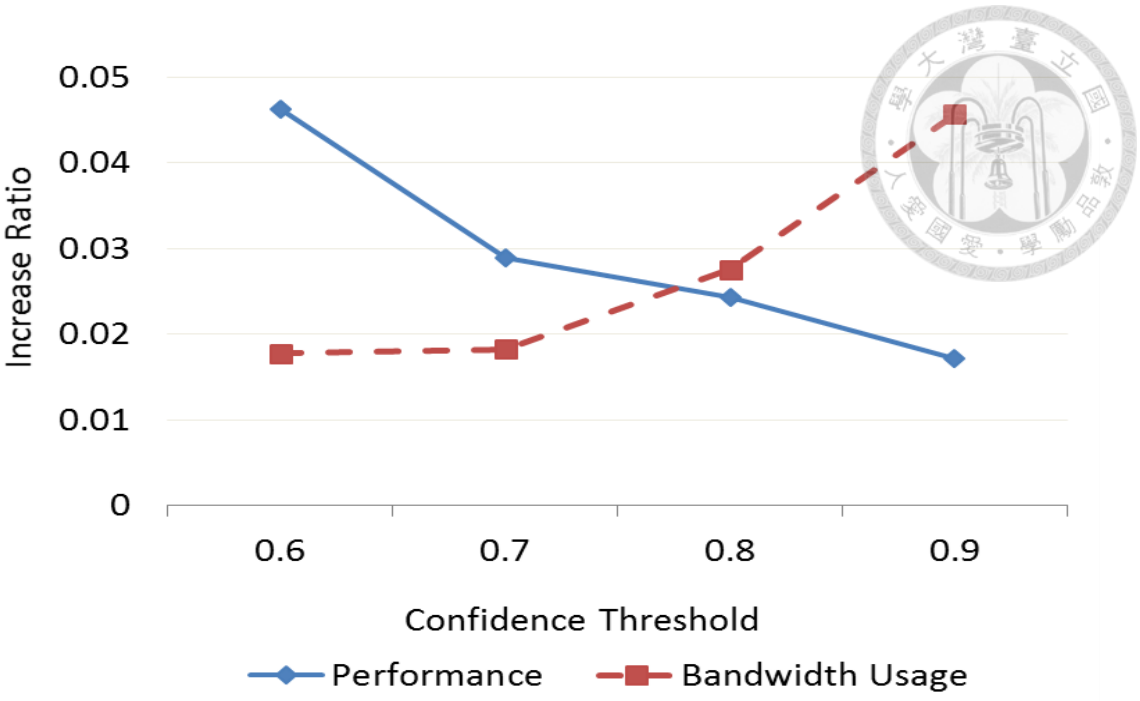


Fig. 12 The ratio of increased performance and corresponding increased bandwidth usage.

Table 8 Ratio of increased performance and corresponding increased bandwidth usage

Confidence threshold	Increased_performance	Increased_bandwidth
0.5	0	0
0.6	0.046188369	0.017764923
0.7	0.028954176	0.018219994
0.8	0.024248388	0.02749226
0.9	0.01716323	0.045658368
1	0.115777133	0.890864455

Chapter 5 Conclusion

In this work, we propose a system architecture which is suitable for developing a cloud-based service with wearable and mobile devices for smart applications. The proposed architecture considers the trade-off between accuracy and bandwidth usage by using an adaptive transmission approach. We further verify the feasibility of the proposed adaptive transmission in an application of diary-like daily activity summarization, which deals with a large number of images. The results of the experiment show that the bandwidth usage can be reduced significantly via adaptive transmission by sacrificing only a small amount of the performance.

Bibliography

[1] Gamhewage C. De Silva , Toshihiko Yamasaki , Kiyoharu Aizawa, An Interactive Multimedia Diary for the Home, Computer, v.40 n.5, p.52-59, May 2007

[2] Aiden R. Doherty , Alan F. Smeaton, Automatically Segmenting LifeLog Data into Events, Proceedings of the 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services, p.20-23, May 07-09, 2008

[3] Deva Ramanan, Detecting activities of daily living in first-person camera views, Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), p.2847-2854, June 16-21, 2012

[4] M. S. Ryoo and L. Matthies. First-person activity recognition: What are they doing to me? In CVPR, 2013.

[5] E. Spriggs, F. D. L. Torre, and M. Hebert. Temporal segmentation and activity classification from first-person sensing. In CVPRW, 2009.

[6] A Fathi, X Ren, and J Rehg. Learning to recognise objects in egocentric activities. In Computer Vision and Pattern Recognition (CVPR), 2011

[7] Y. J. Lee, J. Ghosh, and K. Grauman, “Discovering important people and objects for egocentric video summarization,” in IEEE Conference on Computer Vision and Pattern Recognition, 2012

[8] Stefanie Nowak, Ronny Paduschek, Uwe Kühhirt, “Photo summary: automated selection of representative photos from a digital collection”. ICMR '11 Proceedings of the 1st ACM International Conference on Multimedia Retrieval

[9] Cheng-Hung Li, Chih-Yi Chiu, Chun-Rong Huang, Chu-Song Chen, Lee-Feng Chien “Image Content Clustering and Summarization for Photo Collections” in

Multimedia and Expo, 2006 IEEE International Conference on, july 2006, pp. 1033 –1036.

[10] E. Guldogan, J. Kangas, Gabbouj, M., “Personalized representative image selection for shared photo albums” Computer Applications Technology (ICCAT), 2013

[11] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. “Towards wearable cognitive assistance.” MobiSys '14

[12] Zheng Lu , Kristen Grauman, Story-Driven Summarization for Egocentric Video, Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, p.2714-2721, June 23-28, 2013

[13] Shoaib, M.; Bosch, S.; Incel, O.D.; Scholten, H.; Havinga, P.J. A Survey of Online Activity Recognition Using Mobile Phones. Sensors 2015, 15, 2059-2085.

[14] E Hyytiä, T Spyropoulos, J Ott.,” Optimizing offloading strategies in mobile cloud computing”, 2013

[15] Tekin, C., van der Schaar, M.,” An experts learning approach to mobile service offloading” 2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)

[16] Barbera, M.V. ; Kosta, S. ; Mei, A. ; Stefa, J. “To offload or not to offload? The bandwidth and energy costs of mobile cloud computing” INFOCOM, 2013

[17] Zhixian Yan , Vigneshwaran Subbaraju , Dipanjan Chakraborty , Archan Misra , Karl Aberer, Energy-Efficient Continuous Activity Recognition on Mobile Phones: An Activity-Adaptive Approach, Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC), p.17-24, June 18-22, 2012

[18] Dawud Gordon , Jurgen Czerny , Takashi Miyaki , Michael Beigl, Energy-Efficient

Activity Recognition Using Prediction, Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC), p.29-36, June 18-22, 2012

[19] X. Yuan, J. Yu, Z. Qin and T. Wan, A SIFT-LBP retrieval model based on bag-of features. In ICIP, pp. 1061-1064, 2011.

[20] S. O'Hara and B. A. Draper, "Introduction to the bag of features paradigm for image classification and retrieval," Computing Research Repository, vol. arXiv:1101. 3354v1, 2011.

[21] Quannan Li , Yu Zheng , Xing Xie , Yukun Chen , Wenyu Liu , Wei-Ying Ma, Mining user similarity based on location history, Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems, November 05-07, 2008, Irvine, California

[22] Manjunath, B.S., Salembier, P., Sikora, T.: Introduction to MPEG-7, Wiley (2002)

[23] N. Z. Naqvi, K. Moens, A. K. Ramakrishnan, D. Preuveneers, D. Hughes, and Y. Berbers. To Cloud or Not to Cloud: A Context-aware Deployment Perspective of Augmented Reality Mobile Applications. In SAC, 2015.

[24] S. le Cessie and J. C. van Houwelingen. Ridge Estimators in Logistic Regression. Applied Statistics, v.41 n.1, p.191-201, 1992.

[25] C. M. Bishop. Pattern Recognition and Machine Learning. Springer-Verlag New York, Inc., 2006

[26] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In ICML, 2005.