RESOURCE AWARE SELECTION OF USER GENERATED CONTENT IN CONSTRAINED MOBILE NETWORKS

SESHADRI PADMANABHA VENKATAGIRI
(B.E., Visvesvaraya Technological University)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF SINGAPORE

2014

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Seshadri Padmanabha Venkatagiri
March 31, 2014

ACKNOWLEDGEMENTS

The latin dictum *Vi veri universum vivus vici*, which translates to "By the power of truth, I, while living, have conquered the universe", is an ideal worth pursuing. To seek and learn is the ultimate goal of all mankind.

I would like to thank Almighty God for this opportunity to seek knowledge and advance me towards this noble goal. In this quest for learning my supervisor Prof. Chan Mun Choon, has been the greatest support. He has guided me and corrected me whenever I needed it. All success is due to his support.

I would also like to thank Prof. Anand Akkihebbal, Prof. Ooi Wei Tsang for the timely guidance and help during this long tenure. My family and especially my father has made many a sacrifice so that I may complete my research without worrying about mundane responsibilities. Their blessings have been a protection and moral support to me.

No man is complete without his friends. I have been blessed with the best of friends. Guo XiangFa, Dr. Tan HweeXian, Dr. Choo Faicheong, Girisha, Kartik Shankaran, Mohammad Mobashir, Dr. Prabhu, Dr. Manjunath have all been a source of support and encouragement. I thank them all from the bottom of my heart.

The Bhagavath Gita declares *Nahi Jnanena sadrusham pavitram iha vidyate*, that there is nothing greater than knowledge and only this shall purify mankind. I have strived to walk this path to the best of my abilities. I thank all those who have assisted me in this endeavour.

Contents

Summary

Increasing sophistication of on-board sensors and computing capabilities available in smartphones and their proliferation, have made them an important platform for content generation. Such User Generated Content (UGC) from smartphones, enriched with descriptive meta-information, is gaining significance for its in-situ value in ad-hoc events which have limited infrastructure. User experience is enhanced if the generated content can be shared even as the event is happening. However, smartphone battery and bandwidth constraints makes sharing of UGC challenging.

Existing research in UGC sharing focuses on processing content and enriching its description with the sensor data, but does not take into account network resources required to share the content. Additionally, existing work rely on infrastructure support such as ubiquitous wireless network and GPS localization to select and share UGC.

The contributions of our work includes the design and development of three techniques to share UGC by utilizing subjective (such as user feedback) or objective (sensory data, spatio-temporal conditions) content characteristics under resource constraints. These three techniques are organized in the context of a system called *UGCSelect* and described below:

(i) **Mobile Video Sharing On-Demand (Movisode):** In ad-hoc

events, people generate large quantities of videos. The naive approach of uploading all videos over infrastructure networks becomes impractical when the bandwidth of the network is limited.

Movisode proposes pull-based, on-demand mobile video sharing that accepts user queries for video matching certain spatial and temporal features, and selects a subset of videos matching the query, from available video set using small quantity of metadata uploaded by smartphones. Movisode balances the trade-off between query result visual quality and upload cost of the videos.

(ii) **Automatic Image Linker (AutoLink):** Attendees in ad-hoc events share photos captured in the event venue. Such unstructured photo collections make navigation and content discovery difficult for the user. This problem is compounded by the noise introduced by diverse views, occlusion, and abnormal lighting effects.

AutoLink addresses the problem of organizing such unstructured images by leveraging spatial sensor features and content features to automatically generate content-based links between images efficiently, such that users could navigate between high context images to high detail images.

(iii) **Collaborative Filtering Gel (CoFiGel):** When infrastructure is unavailable, infrastructure-less mobile-to-mobile (m2m) communication can be used to share content. Limited m2m contact capacity has to be managed by sharing only content of potential interest to users. Processing content to infer interest is challenging due to limited processing power and communication bandwidth of smartphones.

CoFiGel proposes a content sharing mechanism which eliminates the need for processing content by leveraging subjective user feedback to predict user interest and push relevant content to users. In distributed m2m networks, user feedback has to be accumulated incrementally. Under bandwidth constraints,

CoFiGel content sharing facilitates faster feedback accumulation so as to improve user interest prediction accuracy, while pushing interesting content to users in a timely manner.

Performance evaluation of the techniques demonstrate significant resource savings while achieving effective UGC sharing. The study also reveals new research avenues to improve the performance of UGC selection.

List of Tables

List of Figures

List of Symbols

Notation	Description
I	Set of UGC items (video/image)
U	Set of users
I_u^+	Set of items that are rated positive by a user $u \in U$
$I_u^?$	Set of unrated items of user $u \in U$
$\mathbb{U}_i^j$	Set of users with unknown ratings for item i in j^{th} round of prediction
$\mathbb{P}_i^j$	Set of users with predicted ratings for item i in j^{th} round of prediction
$\mathbb{K}_i^j$	Set of users with known ratings for item i in j^{th} round of prediction
n	Number of users
$m_{u,i}$	Rating of item i by user u at any given time
$Sim(i,j)$	Cosine-based similarity between items i and j
$RNK_{u,i}$	Rank of item i with respect to user u using cosine-based similarity metric
g_i^+	Number of positive prediction for item i currently
r_i^+	Number of correctly predicted positive ratings for item i currently
Ω_i	Random variable for number of correct positive ratings for item i when all users have rated i
$p_{u,i}$	Probability that the prediction for user u on item i is correct and positive
$\sigma = [\sigma_{i,v}]$	Matrix where each element is the last known position (in bytes) of item i in user v's transfer queue for $v \in H_i$
B	Average device contact capacity
λ	Average device contact rate
H_i	Set of devices with item i
N_i	Set of users that i should be delivered to.
Y_i	Random variable representing the time when i is delivered to *all* users in N_i
Z_i	Random variable representing the time when i is delivered to *any* user in N_i
$M_{i,u}$	Random variable representing the time when user u meets with a user in N_i and transfers i to that user
μ_i	Mean of $E[M_{i,u}]$ over $u \in H_i$
$UTIL_i$	CoFiGel utility
PR_i	Right-hand-side of Equation 5.3
PD_i	Right-hand-side of Equation 5.9
V	The set of all videos segments. $v \in V$ denotes a video segment
G	Directed acyclic graph with $V \cup \{v_s, v_t\}$ vertices and E edges. v_s is *source vertex* and v_t is *sink vertex*.

Notation	Description
t_i	Time of generation of UGC item i
s_v, e_v	Start and end time of video v
$\theta_{v,t}$	Average orientation angle for the t^{th} segment in video v
$\gamma_{v,t}$	Rescaled POI for the t^{th} segment in video v. Scale range is $[0, k_\gamma]$ for a real constant k_γ
θ_q, γ_q	Orientation angle and POI of user query q
$v[t_1, t_2)$	A subset of clip v that falls in interval $[t_1, t_2)$
s_q, e_q	The requested start and end time for query q
$\theta_{q,v,t}$	The angle divergence between video clip v and query q at time t
$\gamma_{q,v,t}$	The difference in POI between video clip v and query q at time t
$\mathbf{T}_{q,v,t}$	Transformation matrix for video clip v and query q at time t
$\mathbf{T}^*_{q,v,t}$	Optimal transformation matrix
$E_{v,l,t}$	Cost to upload video segment $v[t, t+1)$ from smartphone l
(x,y,z)	Point in the cost model
(x',y',z')	Transformed point in cost model
$d_F(\mathbf{T}, \mathbf{T}^*)$	Transformation distance between transformation matrices $(\mathbf{T}, \mathbf{T}^*)$
c_v	Cost of including (effective segment of) v into the query result
$\mathcal{L} = (\mathcal{N}, \mathcal{E})$	Polytree with $\mathcal{N}$ nodes and $\mathcal{E}$ edges
$T = [t_{ij}]$	Time-gap matrix with element t_{ij} representing time-gap between scenes i and j
$S = [s_{ij}]$	Step transition matrix with element s_{ij} representing average number of steps between scenes i and j
t_{gap}	Magnitude of time gap. Maximum threshold for time-gap is Δt_{max}
$Top-N$	Top-N ranking of matching scenes
$Top-M$	Top-M matching images to a bounding box
K	Number of content feature descriptors
$\delta t_{i,j,x}$	Time elapsed difference between capture of UGC item pairs (i,j) and (i,x)
$\delta s_{i,j}$	Step distance difference between capture of UGC item pairs (i,j) and (i,x)
$\mathcal{S}(i,j)$	Estimated step distance between UGC item i and j
a_i	View orientation angle of photo item i
$\delta a_{i,j}$	View orientation angle difference between orientation angles of item i and j
w_r, h_r	Width and height of image/video frame r. $w_{i,r}, h_{i,r}$ is width and height of image i in reference image r
(mx_i, my_i)	Point representing region center of image i
$(mx_{i,r}, my_{i,r})$	Point representing region center of image i within reference image r
$(\Delta mx_{i,r}, \Delta my_{i,r})$	Weighted average of X,Y-offsets of region center of image i within reference image r
$(tx_{i,r}, ty_{i,r})$	Top-left coordinates of bounding box representing region of image i within reference image r
A	Area of region. A_g area of ground truth region and A_p area of predicted region
k_i^j	j^{th} interest point of image i
$XC(k_i^j), YC(k_i^j)$	X,Y-coordinates of interest point k_i^j
$DIST(k_{i_1}^{j_2}, k_{i_2}^{j_2})$	Euclidean distance between interest points $k_{i_1}^{j_2}$ and $k_{i_2}^{j_2}$

Chapter 1

Introduction

1.1 Overview

Smart personal devices such as smartphones, tablets have seen a rapid proliferation in recent times. These devices are also equipped with sensors for localization and tracking, high quality cameras, microphone/speakers and high power CPU and GPU capabilities. These developments have made them an ideal platform for producing, consuming and processing video/image content. Such User Generated Content (UGC) produced using these devices is poised to dominate the Internet of the future [90]. The quantity of content such as high definition images (up to 5MB [43]) and videos (for example, around 100 Hours of video is uploaded every minute to YouTube [9]) produced and consumed is increasingly becoming resource intensive.

1.1.1 Ad-hoc events

An important source of UGC are ad-hoc events. Ad-hoc events are characterized by lack of prior information such as event schedule, venue layout and

lack of infrastructure support. Ad-hoc events can take place indoor, outdoor or semi-outdoor environments. Examples of ad-hoc events can be exhibitions, street performances, mishaps, and other social events.

1.1.2 Applications

Recently, a number of applications, which make use of UGC from ad-hoc events, have been deployed. Some of these applications include crowdsourced surveillance [2], content analytics [1], scene discovery/recommendation [3]. Existing content-sharing platforms (e.g., YouTube) do not suffice for these applications because, these platforms typically require content tagged with keywords. Analyzing content collection using keywords is often insufficient as the content may not be tagged or tagging may be inaccurate and/or at too coarse a granularity. These emerging applications process UGC based on attributes such as location, view orientation, region of scene captured, time of capture, distance of camera from the scene, viewer interest in the content and relationship between content captured by people in the same context.

1.1.3 Challenges

Challenges in sharing UGC in ad-hoc events can be classified into two categories: (1) infrastructure challenges (2) content characteristics.

1.1.3.1 Infrastructure challenges in ad-hoc events

In the context of this thesis, infrastructure support includes: (1) network to share content and (2) localization mechanism to identify the source of content. The challenges arising from the lack of these infrastructures in the context of ad-hoc events are discussed below:

Network infrastructure. Smartphone is the dominant platform for UGC publishing. Mobile networks like 4G/3G/HSDPA and wireless networks (WiFi access points) provide the principle means by which smartphones can share UGC. WiFi is not ubiquitous, but it has higher energy efficiency compared to 3G/4G. On the other hand, 3G/4G networks have higher coverage, but are less energy efficient. Data transfer using mobile and wireless networks on smartphones is constrained by two factors: available bandwidth and battery capacity of smartphone.

1. *Bandwidth bottleneck* in mobile networks arise due to the rapid increase in user population [10, 8]. In many crowded events upload and download of large UGC put a considerable load on mobile network. For instance, the download to upload capacity ratio of the network infrastructure which is usually 9:1, was observed to be 1:1 during the 2013 Super Bowl XLVII. During this event, an 80% increase in mobile broadband data usage compared to previous year's event was reported, with about 388 GB data exchanged [33] at the event venue. Most of the upload traffic included videos and pictures, showing that there is a tremendous user interest in real-time sharing of the event experience.

   In addition to limitation on bandwidth, many mobile service providers impose usage based charging, which reduces the incentive to share content from smartphones using cellular network.

2. *Battery capacity* Wireless communication is the second largest source of power consumption in smartphones. CPU power consumption can also be significant depending on its usage. Power becomes a critical resource situationally. For instance, if one is walking in a park or watching a street show and wishes to upload a video recording, one has to worry about

the remaining battery capacity. The urgency of sharing such video clips in-situ is because, such UGC tend to have more value "at the moment" and lose popularity as time progresses. This reveals a trade-off between energy and the need to share content in a timely manner.

Localization infrastructure. For outdoor localization, Global Positioning System (GPS) is the predominant solution. However, GPS does not work indoors. Even in outdoor environments, the error in GPS is 5 meters or more. This error can result in incorrect labelling of content.

1.1.3.2 UGC characteristics in Ad-hoc events

UGC generated in ad-hoc events have characteristics which affect relevant content selection. They are as follows:

1. Attendees generating the content share the same event context. This results in redundant content
2. Event is captured from diverse views under varying lighting conditions
3. Crowds could cause occlusion (blocking of view) in the content
4. People might capture different moments during the event, but not the entire event itself.
5. There is diversity in the regions of interest captured in the UGC

In order to systematically address the problem of UGC sharing, we classify the UGC characteristics in ad-hoc events into two attribute categories: objective and subjective.

1. *Objective attributes* can be inferred from the content without the viewer feedback, e.g., location, view orientation, etc.

2. *Subjective attributes* are inferred from viewer feedback. For example, viewer's interest (like/dislike) in a particular video.

To summarize, the challenge in achieving UGC sharing in ad-hoc events is to allow the applications to select relevant UGC captured by several event attendees in spite of the limitations and resource bottlenecks in the infrastructure.

1.2 Research goals and contributions

This thesis addresses the following problem:

Given an application query in the form of subjective/objective UGC attributes, how do we "select" matching UGC, while reducing the consumption of resources and infrastructure.

Our solution is based on the conjecture that effective content selection can be achieved if we can make selection algorithms "resource-aware" and "content-aware". Making selection resource-aware ensures retrieval of UGC that results in less resource usage to the smartphone user and the mobile/wireless network. On the other hand, content-awareness ensures that UGC relevant to the application are retrieved.

We broadly divide the design space for the problem into four categories based on the availability of network infrastructure and UGC attribute type as shown in Table 1.1. The design space is organized in the context of a system called *UGCSelect* which facilitates UGC sharing in ad-hoc events. *UGCSelect* operates in two modes. In the first mode, UGC selection is performed assuming availability of network infrastructure. The second mode facilitates UGC selections in infrastructure-less mode. Research contributions of this thesis are

Table 1.1: Design space for UGC selection under resource constraints

UGC attribute category	**With network infrastructure**	**Without network infrastructure**
Objective	**Existing work:** 1. Localization-based selection ([55, 39, 72]): Depends on localization infrastructure. 2. Content-based selection ([41, 16]): Depends on prior information such as training data and localization infrastructure **Contribution #1 and #2:** Multi-source UGC selection. Does not require training data, localization infrastructure: 1. Movisode (for videos) 2. AutoLink (for images)	Future work
Subjective	**Existing work:** Recommendation systems [94] in connected networks which have deployed by large enterprises such as Amazon.	**Existing work:**None **Contribution #3:** CoFiGel, a transmission scheduling mechanism to share content in m2m networks using subjective user feedback

discussed below in the context of these two modes-of-operation of *UGCSelect*.

1.2.1 With network infrastructure

When network infrastructure is available, the straight-forward approach to share UGC is to upload all UGC generated by event attendees. However, this approach is challenging because of the network bandwidth bottleneck. Therefore, mechanisms to select a subset of the available UGC is required. The first approach to selecting content in a infrastructure network would be to use viewer feedback (subjective). This is handled by recommendation systems. There is already a large body of existing work [94] covering this area.

When user feedback is not available, objective content selection is done using location and content based attributes. Localization based solutions use geo-tags [55, 39, 72] to retrieve content. In addition to dependence on localization infrastructure, these schemes do not leverage UGC from multiple sources to balance the trade-off between resource constraints and query result quality. Content based information retrieval methods [41, 16] use features in the image/video to compute the matches to application query. Content based techniques are computation intensive, depend on camera deployments, training data, and require all UGC to be uploaded to server before processing.

We propose two UGC selection mechanisms which address the limitations of localization and content based UGC selection. Both mechanisms do not make assumptions on camera deployment (such as location and movement pattern of the camera) and operate without any training data. They are listed below:

Contribution #1: Mobile Video Sharing On-demand (Movisode). In an event where the attendees capture and share videos of a common scene, uploading all videos consumes resources. Movisode proposes an on-demand mobile video sharing mechanism which avoids uploading user generated videos

unnecessarily.

In response to user application query which specifies the desired spatial and temporal attributes of the required video, Movisode computes the minimum set of matching videos that need to be uploaded under resource constraints, by estimating the view properties of the video relative to the scene, using only a small quantity of metadata obtained from smartphone inertial sensors and video features. Since, absolute location of the video is not required for computing query result, dependency on localization infrastructure is eliminated. This addresses the limitation of localization-based techniques (Table 1.1).

Contribution #2: Automatic Image Content Linker (AutoLink). In an event context with multiple scenes, attendees move from one scene to another, capturing photos of scenes they are interested in. Photos could belong to different scenes, or different regions of the same scene. When image collection is large, organizing images per scene, and within each scene becomes necessary especially for small-factor display devices such as smartphones.

Using the metadata obtained from smartphone inertial sensors and image features, AutoLink clusters the photos for each scene and organizes photos of a scene into an image hierarchy enabling viewers to navigate from images farther from the scene, to images which capture parts of the scene at a closer distance.

This hierarchy also enables images to be retrieved only when the viewer requests more details of the scene, thus achieving an on-demand retrieval similar to Movisode. It also does not localization.

1.2.2 Without network infrastructure

If the network is intermittently connected, or if there is a need to offload traffic load from the mobile/wireless network, then infrastructure-less content sharing becomes necessary. We illustrate the advantage of using infrastructure-less mobile-to-mobile (m2m) links, with a simple measurement of performance during file transfer between a mobile device and a central server using 3G/HSPA network and between two mobile devices directly using WiFi. To quantify the performance of mobile-to-server (m2s) transfer, we upload and then download a 14.3 MB video clip to YouTube using a HSPA network. The HSPA service provides maximum download and upload rate of 7.2 Mbps and 1.9 Mbps respectively. The average download and upload throughputs measured (average of 5 trials) are 1125.2 kbps and 57 kbps respectively. To quantify the performance of m2m transfer, we use two Samsung Nexus S phones that support IEEE 802.11n (link rate is 72.2 Mbps) to exchange the same video file directly over a TCP connection. The measured throughput is 22.6Mbps (average of 5 trials). The large difference between m2s and m2m in measured throughput can be attributed to the differences in link rate and Round Trip Time (RTT) observed (70ms for mobile-to-server and 5.5ms for mobile-to-mobile).

Disruption Tolerant Network (DTN) architecture has been proposed to support content dissemination over infrastructure-less m2m links. In DTN, node mobility and intermittent connectivity between nodes is factored in while disseminating content. By leveraging m2m ad-hoc communication links, information is disseminated over multiple links opportunistically. One typical instance of DTN is smartphone users using WiFi ad-hoc mode, bluetooth or WiFi Direct to communicate with neighbouring devices. When a pair of devices are in proximity, they can exchange information, and link disconnections

occur when the devices move out of communication range.

Selecting UGC using content features in DTN is challenging because of the limitations of processing power of mobile nodes. Additionally, viewer is not aware of all the content available in DTN because of its distributed and intermittent characteristics. This makes selection based on content features difficult. These limitations motivate the use of recommendation systems with subjective attributes to predict user interest in the content, rather than processing the content itself.

For recommendation systems in infrastructure networks, all content is available for the viewer to watch and provide feedback. This feedback is then used to recommend more content to the viewer accurately. In DTN, the viewer is not aware of all the content in network. In order to provide feedback, content has to first reach the viewer over DTN. This results in higher resource consumption. However, if more content is disseminated, more viewer feedback is received, which improves accuracy but consumes more resources. To facilitate recommendation systems to function in DTN context, we propose:

Contribution #3: Collaborative Filtering Gel [86] (CoFiGel), a mechanism that performs content selection and dissemination by balancing the trade-off between recommendation system accuracy and resource usage in mobile-to-mobile (m2m) DTN network.

We leave UGC selection based on objective attributes as future work.

1.2.3 UGCSelect: System overview

This section provides an overview of *UGCSelect*. Chapter 2 discusses the related work and positions *UGCSelect* with respect to existing UGC sharing techniques. Limitations of *UGCSelect* and future directions to improve it are

discussed in Chapter 6.

Figure 1.1 [1] shows how the UGCSelect system works. Participants in an event capture UGC using their smartphones. UGCSelect supports two modes of sharing as mentioned previously. First mode is through infrastructure based Mobile-to-Server (m2s) connection. Second mode is infrastructure-less using Mobile-to-Mobile (m2m) communication links. The decision to select UGC is made by one of the three decision modules: Movisode (Chapter 3), AutoLink (Chapter 4) and CoFiGel (Chapter 5) during the transfer of data.

When content selection has to be performed based without infrastructure, CoFiGel module handles the decision making for selection. When infrastructure is available, and if video UGC has to be retrieved using objective attributes such as spatial and temporal characteristics, then Movisode determines content selection. If UGC is image, then AutoLink is used for content selection based on spatial features of image.

UGCSelect consists of two components: Mobile agent and server. Following sections provide description of these two components.

[1] URLs of icons used in the images. Last retrieved on September 6, 2014.

1. http://theresadelgado.com/wp-content/uploads/2011/11/Compass.png
2. http://www.wpclipart.com/computer/PCs/more_computers/server.png
3. http://tinyurl.com/n8yebuj
4. http://poulingail.edublogs.org/files/2012/09/laptop-kid-18fhs8b.jpg
5. http://gpsmaestro.com/wp-content/uploads/2012/05/smart-phone-icon.jpg
6. http://pndblog.typepad.com/.a/6a00e0099631d083301348623920 0970c-800wi
7. http://tinyurl.com/qdsqvpl
8. http://tinyurl.com/q5g26e5

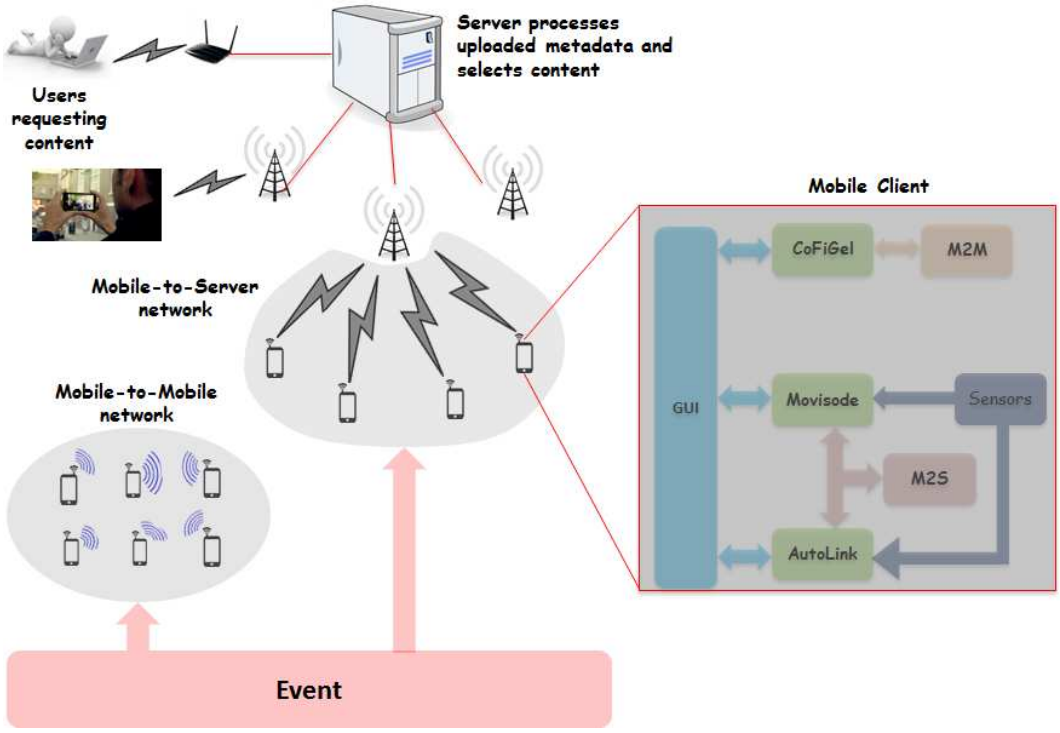


Figure 1.1: Overview of the UGCSelect System

1.2.3.1 Mobile client

There are four submodules which form the mobile client. *GUI*, *sensor module*, *m2s module* and *m2m module*.

GUI. The GUI allows user to capture media through the embedded camera service. Captured content are stored in inbox of media items which allows the user to watch, rate and share content. Users can also tag media as belonging to particular events (from a list downloaded from the server), or register for a particular event, which will let the app automatically tag media as belonging to a particular event if they are taken within a specified time period. As soon as a video/image gets tagged (either manually or automatically), corresponding metadata is uploaded to the server periodically.

Sensor module. The sensor logger runs in the background and records acceleration, location and orientation readings to file along with their timestamps. When the media is created, it is processed based on its type. If it is video, then all sensor data logged within start and end timestamps of the video are extracted and segmented. Average values for sensors for each segment are computed and uploaded to server. If the media is image, sensor logs between two image captures are uploaded to the server. In addition to the sensor data, image/frame features, time of capture are also uploaded.

M2M module. It detects peer mobile devices, schedules and transfers content. It is also responsible for maintaining an history of past device contacts (e.g., time and duration of contact, frequency of contacts).

M2S module. It uploads metadata to server, checks for new application queries and uploads requested video segments and/or images.

1.2.3.2 Server

The server communicates with mobile agent only when m2s connection is available. Movisode and AutoLink have server components which receive the metadata from the mobile device, and for each application request decide the video segments or images to be uploaded. Movisode processes spatio-temporal query and AutoLink retrieves image depending on the user's navigation through the image hierarchy. As user navigates deeper into the image hierarchy, more detailed images for the scene are retrieved.

1.3 Summary

In this chapter, we have introduced UGC as an important source of information which could be exploited by real-world applications. We have identified ad-hoc events as major venues for UGC publication. Further, we have discussed the challenges in sharing UGC in the context of ad-hoc events and presented three contributions of this thesis in the context of system called *UGCSelect* to address these challenges.

Chapter 2

Background

We survey the existing literature associated with UGC sharing in mobile networks and compare and contrast with *UGCSelect*. The related work is classified into three main categories based on the technique used to achieve UGC sharing. They are as follows:

1. Resource management in mobile networks (Section 2.1), which discusses content-agnostic methods of managing network resources

2. Sensor and content based techniques (Section 2.2), which deals with techniques that use objective attributes such as sensor and content features to retrieve content

3. Recommendation techniques (Section 2.3), which elaborates on techniques that use subjective attributes such as user feedback to retrieve content

Figure 2.1 graphically positions *UGCSelect* with respect to the state-of-the-art. A discussion on the novelty of *UGCSelect* with respect to the above techniques is presented in Section 2.4.

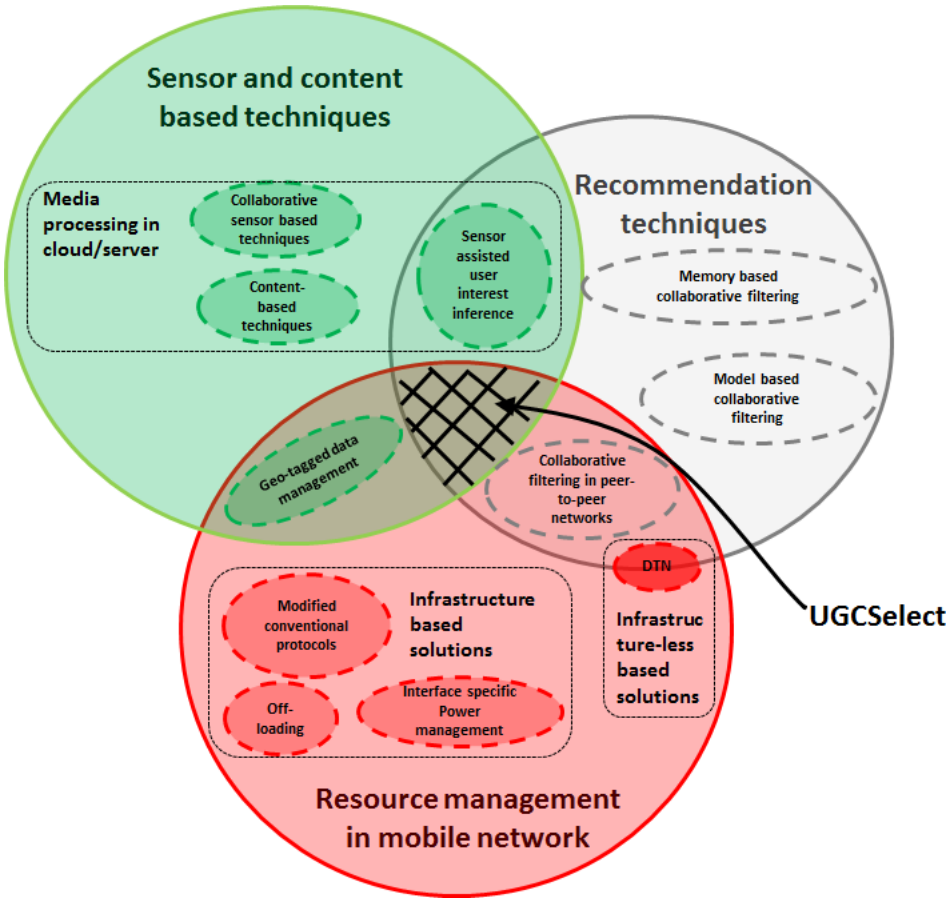


Figure 2.1: Survey of techniques used for UGC content selection and retrieval

2.1 Resource management in mobile networks

Research in resource management for mobile networks could be broadly classified into infrastructure based and infrastructure less. Infrastructure based solutions focus on making the conventional network protocols resource-aware, or distribute the traffic load on different network types, or leverage infrastructure-less means of communication to transfer data. All the techniques discussed here are *not content-aware*. I.e. they do not consider the content attributes to manage resources.

2.1.1 Infrastructure based solutions

2.1.1.1 Modified conventional protocols

A plethora of research effort has been directed towards managing smartphone battery and mobile bandwidth. One direction is to improve existing protocols like TCP, save energy based on traffic patterns and understand the sources of energy consumption. In [45], energy consumption of 3G/4G networks was analyzed and the study indicates a predominance of tail energy (interface is kept on for some time after all data transfer is complete [20]) in these two interfaces and also observes the high energy-per-bit consumption of 3G/4G compared to WiFi through measurements. Energy conservation is also attempted using the screen-off and on states of the smartphone in [46], and by predicting traffic burst patterns and knowledge of the tail energy phase of 3G/4G in [29, 67]. GreenTube adopts a similar method to optimize the download of video-stream while reducing the active periods of the 3G/4G interface. Some proposals attempt to modify and improve the TCP protocol to be more energy efficient. In [78], TCP connection closure using FIN/RST packets is avoided to save

battery power. Smartphone device configuration is optimized for energy efficiency based on the user context and usage using historical usage patterns in [32]. Caching based on viewer behaviour, i.e., how long a video is watched by mobile user, is proposed in [42] to save energy during stream pre-fetching.

2.1.1.2 Off-loading

The second popular approach to augment bandwidth and save energy is Mobile data offloading. It is done in two ways: **(1)** using multiple interfaces **(2)** through collaboration between neighbouring devices. A study in performance measurements and feasibility of multipath TCP over 3G/4G and WiFi is undertaken in [26, 81] for three commercial service providers in USA. CoolTether [88], COMBINE [15] use the neighbouring devices to exchange data with the web. This collaborative approach aims to reduce energy consumption and increase aggregate bandwidth in the process. DozyAP [38] demonstrates that using WiFi tethering for Internet connection sharing allows for putting the interface to sleep to conserve energy. Catnap [31] puts the NIC to sleep between packets to save energy. In [19], a framework called Wiffler is proposed which predicts WiFi connectivity in vehicular networks and uses it for delay-tolerant data exchange. In [59], a measurement study of 97 iPhone users reveals that delay-tolerant use of WiFi for offloading benefits when the data can sustain a delay in the order of tens of minutes. In [99, 83], 3G onloading is proposed to augment the bandwidth of wired network using 3G. BlueStreaming [70] achieves video streaming to mobile devices by using Bluetooth for delay-sensitive control traffic and WiFi for data traffic. Micro-Cast [53] uses mobile-to-mobile 1-hop collaboration to effectively increase video streaming throughput to mobile devices.

2.1.2 Infrastructure-less solutions

2.1.2.1 DTN

When the collaboration of mobile devices extends to multiple hops and allows for delay-tolerance and disruption tolerant links, another class of solutions become available in the form of Disruption Tolerant Network or DTN routing. There are many unicast mobile-to-mobile routing schemes designed to improve point-to-point delivery probability and/or minimize delay [60, 64, 56, 18, 101]. These protocols, however, do not address the issue of information dissemination. When the content is such that it loses value over time (eg. news), content freshness becomes an important criteria. Disseminating fresh content is solved in [44, 47, 82, 14]. These techniques usually involve downloading content from the Internet to a subset of mobile nodes, which then distribute it among themselves with the objective of maximizing content freshness. There are also proposals [47] which deal with caching content in DTN so that it could be retrieved on request. In such schemes, Mobile devices run caching algorithms which refresh/reshuffle their cached content based on a voting process. Pre-defined preference [50] or tags [71] are also used to route content to users. Tag-based routing allows users to declare their interest in the form of a set of keywords. These keywords are disseminated into the network. When a node encounters a user with matching interest keywords, it forwards the information with matching tags to interested users. VStore [63], PhotoNet [98], and CARE [103] address picture dissemination in mobile-to-mobile networks where transmission bandwidth and storage are the main concerns. HORUS [36] use the vehicle trajectory as the camera trajectory and determines most relevant clips that capture a target area and retrieves it over vehicular DTN.

2.2 Sensor and content based techniques

Media content features have long been used for organizing and selecting content. But they require that all content be uploaded to a central server or cloud for processing. Section 2.2.1.1 elaborates on content-based content selection methods.

Sections 2.2.1.2, 2.2.1.3 elaborate on sensor-based and hybrid content-sensor based proposals to select content. Smartphone sensors are used for a variety of applications like localization, augmented reality, tracking, health monitoring. A survey of these mobile sensing applications is available in [54, 58]. These sections focus on the use of sensors for enriching media content. However, discussed techniques are similar to content-based techniques in that they also rely on cloud or a central server for processing sensor-annotated media, and hence are *not resource-aware*.

Section 2.2.2 discusses geo-tagged media management. This area focuses on leveraging GPS and compass sensors to construct a view model of the media. The view model is used to finding matching videos for user query. Geo-tagged media management retrieves media on-demand.

2.2.1 Media processing in cloud/server

2.2.1.1 Content-based techniques

GigaSight [91] performs filtering and indexing videos on a cloud infrastructure. This system tries to identify privacy-sensitive information from crowdsourced video and uses it for retrieval from cloud. MediaScope [49] allows the cloud to retrieve images using image features extracted on mobile phones. Features are used to retrieve most common images, or dissimilar images to maximize

available information to the user. ImageWebs [41, 5] uses Bag-of-visual-words based CBIR to filter images before applying affine co-segmentation which applies affine transform to feature points (filtered by RANSAC) and then computes feature similarity. Brachmann et al [25] uses image webs to build graph between images and uses the connectivity information with known labels to propagate visual words to unknown labels of images.

2.2.1.2 Collaborative sensor based techniques

In recent years, there have been a number of research works that attempt to incorporate collaborative techniques into media capture. iSense [112] applies multi-modality clustering using context information (e.g. location, timestamp) and content information (e.g. color, texture) to search for matching images in mobile phones while CrowdSearch [106] uses crowdsourcing via Amazon Mechanical Turk for image search. On the video side, TelosCAM [95] and VirtualLock [107] use camera sensors to collaboratively provide video surveillance. The capture event is triggered by either users or the proximity sensors. Photo Tourism [93] uses SfM to compare images pairwise to get relative co-ordinates of cameras. Absolute GPS co-ordinates are required to overlay images on maps. Heng et al [66] proposes GPS error correction by combining SfM, GPS and visual vocabulary. VIRaL [51] uses GPS and 3D reconstruction to cluster images for visual search. Gozali et al [35] builds hidden markov model by combining content and time gap features from training data and uses it to predict cluster boundaries for a event photo stream.

Geo-tagged images have been used to identify good-scenic quality routes from community driven geo-tagged images in GPSView [111, 13]. Localization is also achieved by combining images with GPS locations and compass data [76]. Images are clustered based on GPS and compass to identify points-

of-interest in [57, 96]. Geo-location and time based image clustering is proposed in [28]. In [68], social media platform check-in information is combined with GPS tags and records to identify geographical areas of interest for tourists, while [75] uses template matching and compass to correct camera pose for featureless content. Sensor assisted video and image tagging is proposed in SEVA [69] and TagSense [79]. OPS [73] uses sensor annotated images to obtain GPS location of an remote outdoor landmark.

In MoVi [22], social events are captured collaboratively using sensory cues such as ambient sound and light to detect changes in environment. OutListen [7] and Micro-Cool [4] allow concert/event spectators to capture and upload their video clips and use these video clips to reconstruct (either manually or by matching audio content) the concert experience. Zhang and Wang [110] use GPS and compass sensor data to generate a tourist summary video from multiple videos in a video database.

FOCUS [48] uses Structure-from-Motion (SfM) [93] as part of cloud-based architecture for indexing videos. SfM requires all videos to be available for frame-by-frame pairwise comparison of all videos, and camera calibration for all the cameras involved in the video capture. Both systems require all videos from smartphones to be uploaded to the cloud for processing.

2.2.1.3 Sensor assisted user interest inference

Bao et al [21] uses smartphone sensors and acoustic signatures recorded signatures to predict user feedback to movies.

2.2.2 Geo-tagged data management

Video capture, retrieval and classification are addressed by geo-tagging based techniques which utilize a variety of sensors available in smart phones to annotate media. For instance, Ay et al. [17] uses sensor metadata to rank relevant video clips based on the magnitude of area or time interval captured in each video clip. This ranking is presented to queries placed by users. The ranked results are filtered based on occlusion and wiki page information [89]. This effort is further extended [39] to introduce energy conservation by uploading metadata on-demand, and by using WiFi/GSM based location inference to complement GPS.

2.3 Recommendation techniques

Personalization has been the next big jump in technology of the Internet today. Every form of content provider or content search applications today have means to filter and recommend content to suit the needs of a specific user. To achieve this effect a myriad of recommendation systems have been developed and deployed. The prevalent architecture for such personalized content services has been client-server system based on HTTP. The same architecture has been extended to mobile devices also. However, as pointed out before, such a system does not address the resource considerations. The most prominent and popular personalization technique that has seen extensive research and wide deployment is collaborative filtering (CF) [94]. CF techniques can be broadly categorized into memory-based or model-based.

2.3.1 Memory based collaborative filtering (MCF)

Memory-based CF (MCF) utilizes rating history of users to identify neighbourhood patterns among users or items. This pattern facilitates the prediction of ratings for hitherto unrated user-item pairs.

2.3.2 Model based collaborative filtering

Model-based CF uses the user ratings in conjunction with standard statistical models such as Bayesian belief nets and latent semantic model to identify patterns in the ratings of user-item pairs. The resultant model is then used to make predictions for future ratings.

2.3.3 Collaborative filtering in peer-to-peer networks

There exists much research on using CF on peer-to-peer (P2P) systems. PocketLens [74] is a recommender system for portable devices that uses item-item collaborative filtering for making recommendations. It proposes a rating exchange protocol for both distributed P2P architecture and centralized server architecture, where nodes rely on a central server for storing rating information. A probabilistic model-based CF is proposed by Wang et al. [102] for a P2P network.

2.4 Discussion

Table 2.1 shows the comparison of *UGCSelect* with the related work discussed in the previous sections. The comparison is along the following themes:

1. **Network infrastructure resource awareness:** Sensor and content based techniques which rely on the cloud ignore the characteristics of the

network such as available bandwidth and battery capacity of the mobile devices, and are unsuitable to operate under resource constraints. This limitation is also observed in recommendation techniques developed for infrastructure networks. *UGCSelect* includes the network characteristics as part of its selection mechanism and selects only content from mobile devices which have enough battery and bandwidth to share content. Additionally, *UGCSelect* relies on metadata to select content. This prevents the sharing of content if it is not required. Required data is shared on demand.

2. **Content redundancy awareness:** Resource management techniques developed for mobile networks are not *content-aware*. Due this limitation, redundancy in similar content cannot be eliminated. For sensor and content based techniques which use cloud, all content is uploaded to cloud, without regard for eliminating redundancy. This results is inefficient use of available resources. *UGCSelect* eliminates redundancy at the source before sharing by leveraging the metadata from the content and sensors.

3. **Multi-source selection:** UGC from ad-hoc events usually do not capture the event in entirety. This means any single UGC content might not be sufficient to compute the result for application query. Therefore, content from multiple sources have to be combined to get the desired result. This issue is not addressed in geo-tagged media management techniques. *UGCSelect* incorporates multiple-source content to elimintate this limitation. Another use of multiple sources is that it provides diversity of choice for selecting content. When two UGC media are similar, system could choose the UGC which results in minimum retrieval cost. Geo-

tagged media management does not leverage this source diversity. This advantage is incorporated by *UGCSelect*.

4. **Dependence on training data:** Content-based techniques require training data to operate, which might not be available for ad-hoc events. *UGCSelect* incorporates techniques which do not assume the availability of training data.

5. **Dependence on camera infrastructure:** Some collaborative sensor based techniques require camera network deployments. Ad-hoc events might occur in locations where camera network deployments might not be possible or might be rendered useless due to large crowds and occlusion. *UGCSelect* relies on event attendees to capture content. Content redundancy facilitates finding content of a particular view which is not occluded.

6. **Dependence on localization infrastructure:** Ad-hoc events could indoors, outdoors or in semi-outdoor environments. Localization infrastructure like GPS does not work indoors, and is prone to error when the GPS signal is disrupted by canopies and trees. In tightly packed events, where in scenes are in close proximity, GPS error could result in incorrect content selection. Geo-tagged media management and some collaborative techniques relying on GPS, have the above limitation. *UGCSelect* localizes the content using the scene, rather than the absolute location. This eliminates the need for localization infrastructure.

Table 2.1: Comparison of *UGCSelect* with related work

Technique		Network infrastructure resource-aware	Content redundancy aware	Multi-source selection	Dependence on training	Dependence on camera infrastructure	Dependence on localization infrastructure
Resource management in mobile networks	**Infrastructure based solutions**	Yes	No	Not applicable			
	Infrastructure-less solutions	Yes	No	Not applicable			
Sensor and content based techniques	**Media processing in cloud/server**	No	No	Yes	Content-based techniques require training	Some collaborative techniques require camera network	Techniques using GPS require localization
	Geo-tagged data management	Yes	Yes	No	No	No	Yes
Recommendation techniques		No	No	No	Yes	No	No
UGCSelect		Yes	Yes	Yes	No	No	No

Chapter 3

Movisode

3.1 Outline

We describe the details of Movisode, the first of three decision making modules in Figure 1.1. In an event where the attendees capture and share videos of a common scene, uploading all videos consumes network resources and smartphone battery. Alternative to this approach is to select a subset of videos which satisfy the user. The problem is to choose the minimum subset of videos which satisfy the user and also reduce the resource consumption.

Movisode proposes an on-demand mobile video sharing mechanism which avoids uploading user generated videos unnecessarily. As shown in the Figure 1.1, Movisode uses m2s infrastructure network to facilitate user generated video sharing. It consists mobile client component and server component. Movisode mobile client uploads a small amount of metadata information generated on the smartphones to the server initially, instead of uploading the entire video by default. The server will then only fetch relevant videos, in response to user or application queries. By uploading only a small amount

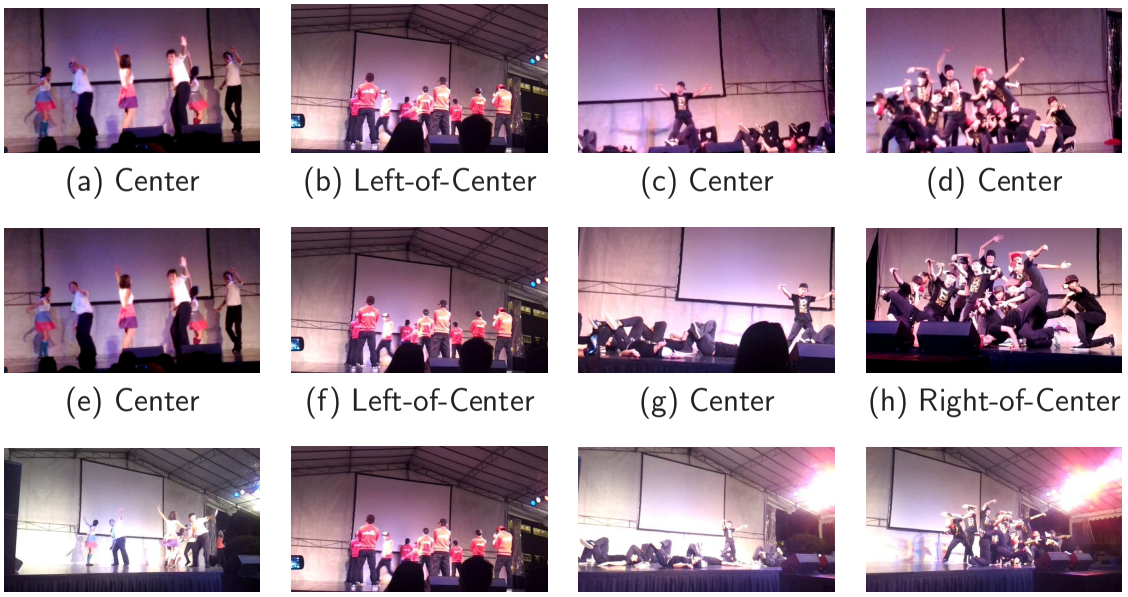


Figure 3.1: Thumbnails for clip segments generated from NAF_230312 event of Jiku Mobile Video dataset using three different algorithms. Row 1: The algorithm finds the best matching angle. The upload cost (as energy) is 110 Joules. Row 2: Our Movisode algorithm. The upload cost is 27 Joules. Row 3: The algorithm minimizes the upload cost. The upload cost is 18.3 Joules. Each column represents a particular time instance of the query interval.

of metadata information to support queries and only upload more data on demand, the network and energy cost on the smartphones are reduced.

Movisode is event-centric as it assumes that mobile videos are grouped according to the event during which the videos are shot. It has several key differences from the conventional video sharing approach (e.g., YouTube): (i) Movisode supports spatio-temporal query by utilizing the objective attributes of the video. Users search for videos taken during a specific period of time, from a specific angle, focusing on a specific point-of-interest (POI) of the scene (we will define the concept of POI more precisely in Section IV-A). (ii) The output video generated by Movisode in response to a user query can be made up of videos uploaded by multiple users. This is particularly useful when no single video available covers the spatial/temporal dimensions of the query. (iii) Movisode does not require localization information, which may not be available (e.g., GPS is not available indoor) and can consume much energy to acquire.

Movisode is driven by two main algorithms. The first is a lightweight metadata extraction algorithm that runs on the smarpthone. The algorithm infers the view angle (from compass sensor on smartphone), POI (from the video content) of available video clips, and uploads these information to the server. The second algorithm runs on the server. It takes in metadata information from the smartphones, upload cost/bandwidth of participating phones, and the results of previous queries. The algorithm then decides which video to retrieve from which smartphones to satisfy a given spatio-temporal query from the user. The objective is to balance the often conflicting goals of meeting user requests and minimizing upload cost, by exploiting the fact that users usually cannot visually detect small deviations in view angles and POI.

As an illustration, consider a request for videos that show the performance

from the center of the stage for a 2-minute interval of a dance event. Figures 3.1(a)- 3.1(d) show thumbnails of the best match videos available, obtained using an algorithm that does not take into account upload cost. Figures 3.1(e)-3.1(h) show thumbnails extracted from the set of clips selected by Movisode to satisfy the request. Movisode is able to select videos that are very similar to the best available videos while incurring significantly less upload cost by saving 83 Joules of energy in this particular case. On the other hand, Figures 3.1(i)-3.1(l) show the thumbnails selected by an approach that considers only upload cost. Although, this approach fetches the clips that results in the lowest upload cost, the view angle, and POI stay mostly on the left of the stage instead of center of the stage as required by the query.

The organization of this chapter is as follows. We present the user interaction model of Movisode in Section 3.2, metadata extraction approach performed by the smartphones in Section 3.3, and video selection algorithm performed by the server in Section 3.4. Results of evaluation are discussed in Section 4.4. Section 4.5 concludes the chapter with a summary.

3.2 Using Movisode

We first illustrate how Movisode works by presenting how a user interacts with Movisode through the Web interface. Organizers of events such as sports matches and stage performances can register their event with Movisode, providing location and time information. Attendees of the event then "check-in" into the event through the Movisode mobile client. After that, the attendees begin to capture videos using their smartphones at the event venue for sharing. Periodically, each smartphone uploads metadata of video that it intends to share.

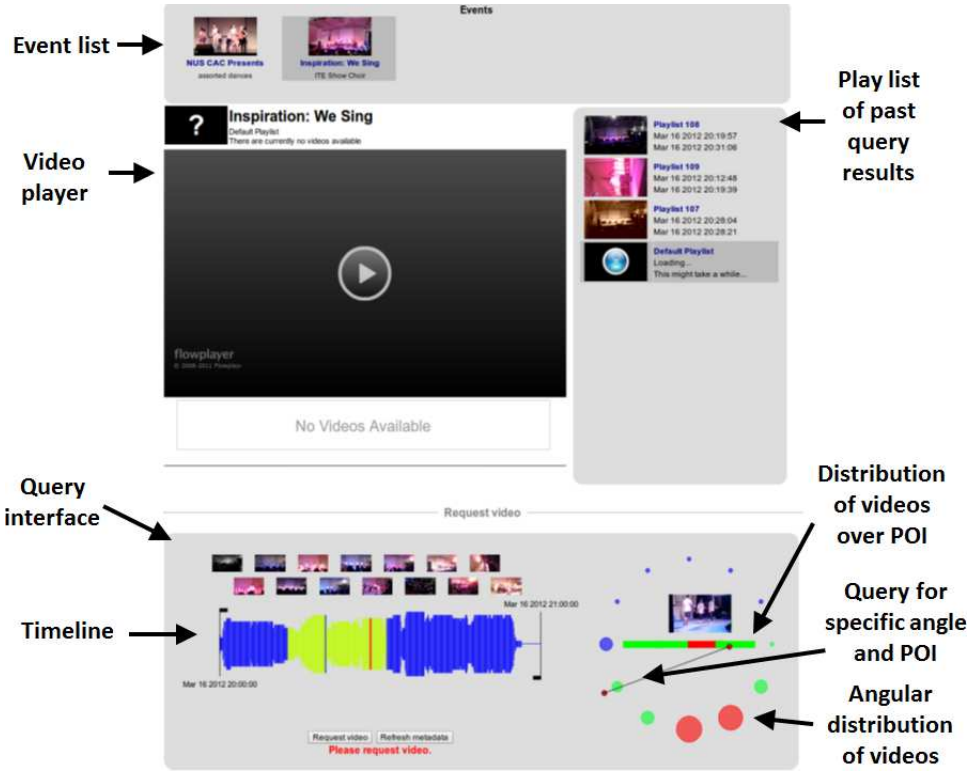


Figure 3.2: Web interface for user query.

A user can place a query in the form of "show me videos of the event from time t_1 to t_2, with cameras recording video from angle θ and pointing at POI γ". Movisode selects the set of clips, based on video metadata previously uploaded by the users, that balances the conflicting objectives of high quality and low cost. Once all the selected clips have been uploaded from the smartphones to the server, its availability is indicated on the web interface.

Figure 3.2 shows the Web interface. The top of the page lists the available events. For a selected event, past query results are shown on the right hand side of the main video window. Below the main video window, the viewing angle, POI, duration, and popularity of the set of video clips are visualized.

3.3 Metadata Extraction

In order to provide metadata about a user captured video on the smartphones, each smartphone runs a light-weight metadata extraction scheme. For each video, the metadata extracted, besides the start and end times, are the viewing angle and point-of-interest (POI). Distance is not included as part of the metadata since we do not assume the availability of location information.

3.3.1 Angle and Point-of-Interest (POI)

We assume the availability of a reference image of the event area in advance, which may be provided either by the participant or the event organizer. For the purpose of computing viewing angle and POI, we view the event from a top-down or bird's eye view. Whatever the shape of the stage or event area may be, it is always projected on to a 2D plane represented by the reference image. If an event contains multiple scenes, each scene will be associated with

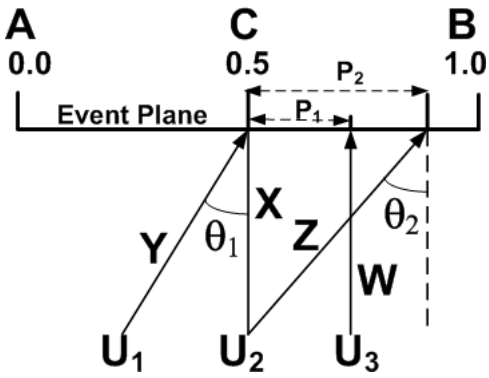


Figure 3.3: Illustration of difference between POI and Angle.

a reference image. The Movisode system processes queries based on reference images.

Figure 3.3 illustrates the model. The event plane, AB, is projected to be a horizontal line and C is the center of the line AB. Without loss of generality, we assume that a line that is perpendicular to the event plane has a view angle of 0°. For example, in Figure 3.3, user U_2 viewing along the line X has a view angle of 0°.

We define the **view angle** of a frame captured in a video clip as the angle between the line representing the view and a line perpendicular to AB. Users U_1 (along line Y) and U_3 (along line Z) have view angles of θ_1 and θ_2 respectively.

We define the **point-of-interest (POI)** of a view captured in a video clip as the position of intersection between the line representing the view and event plane AB. We quantify the POI value as the normalized distance from one end of AB. In Figure 3.3, the POI of Users U_2 (along line X), U_4 (along line W) and U_3 (along line Z) have POI of 0.5, $0.5 + P_1$ and $0.5 + P_2$ respectively.

While computation of a view angle can be performed relatively easily using the compass sensor available on most modern smartphones, computing the POI is more involved and is presented in more details in the next section.

3.3.2 Computing POI

In principle, the POI of an image is computed by finding the horizontal shift in image features between the given image and the reference image. This shift is used to compute the displacement of the video frame's view with respect to the reference image center. The algorithm thus requires a reference image with sufficient features as an input.

The smartphone computes the POI and uploads the average POIs as meta-data to the server periodically. Conceptually, POI is computed as the average relative distance of matching interest points found in the given image and the reference image. We compute the POI for video frames sampled at a pre-defined sample rate.

For each sampled video frame, the frame is resized to reduce the power consumption and delay during computation. The effect of resizing is explored in Section 3.3.3. The interest points of video frame is computed using BRISK [62] feature detector[1]. These interest points are matched with interest points in the reference image.

For each matching interest point pair $(k_r^{j_1}, k_i^{j_2})$, the interest point of video frame (i) is rescaled to the width of reference image (r): $\left(XC(k_i^{j_2}) \times \frac{w_r}{w_i}\right)$, where, w_r is width of reference image and w_i is width of video frame.

The offset distance between the X-coordinate of reference interest point $XC(k_r^{j_1})$ and rescaled $k_i^{j_2}$ is then computed as:

$$\text{Offset } = \left((XC(k_r^{j_1}) - \left(XC(k_i^{j_2}) \times \frac{w_r}{w_i}\right) \right)$$

Finally, the offset is normalized to range $[0, k_\gamma]$ ($\frac{k_\gamma}{2}$ is the normalized center of

[1]available in the OpenCV (http://opencv.org/) library for Android

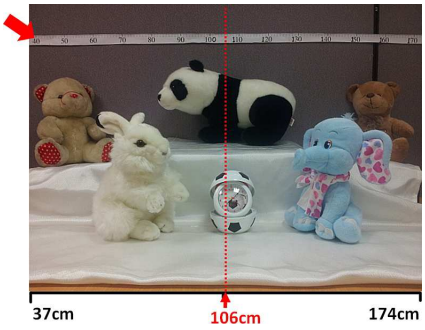


Figure 3.4: Sample reference image for POI computation.

the reference image), with respect to reference image as follows:

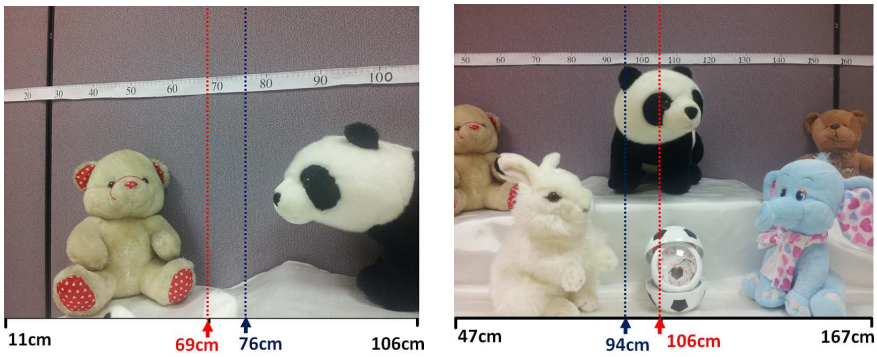
$$\text{Normalized offset } = \frac{(XC(k_r^{j_1}) - \left(XC(k_i^{j_2}) \times \frac{w_r}{w_i}\right)) \times k_\gamma}{w_r}$$

The average of all positive offset distances over all interest points are computed. Similarly, average of all negative offset distances are computed. If the number of points with positive offset is more than negative offset, then positive offset is used to compute POI. Otherwise, negative offset is used. The computed POI is the the normalized offset computed in the previous step plus $\frac{k_\gamma}{2}$. The resulting value is the POI.

Note that since we use interest point feature that is scale invariant to compute POI, our algorithm works even if the reference image and the captured image are of different resolution, zoom-level, and orientation.

3.3.3 Performance

The POI computation runs on the smartphone and needs to be accurate as well as lightweight. Both image resolution and occlusion could affect the accuracy of the algorithm. We evaluate the impact of image resolution in this section. The effect of occlusion on accuracy under real world settings is evaluated using



(a) Left (b) Center

Figure 3.5: Frame samples with different orientations used for POI computation.

real world traces and user study in Section 4.4.

We evaluate POI computation algorithm using three metrics: computation time, energy consumption, and accuracy, for different sub-sampled sizes of the candidate images.

We create a test image as shown in Figure 3.4 as the reference image. Figure 3.4 also shows a ruler (indicated by the red arrow). The reference image captures the ruler segment starting at 37cm and ending at 174cm. The ground truth POI of the reference image is at the mid-point of the image (at 106cm).

We run our experiments on a Galaxy Nexus phone, running the Android operating system. Power is measured with a Monsoon power monitor device [2]. Accuracy is defined as $\big((1 - \frac{|\text{Estimated POI} - \text{Ground truth}|}{\text{Length of ruler segment}}) \cdot 100\big)$.

The experiment is conducted for 12 images and 5 subsampling resize factors. We show two representative images in Figure 3.5 from different angles and POIs. Note that the image shown in Figure 3.5(b) contains scene changes (positions of figures have changed).

The ground truth is indicated by the red dotted line in the image (original

[2]http://www.msoon.com/LabEquipment/PowerMonitor/

image, without subsampling) and the estimated POI is indicated by the blue dotted line. The start and end points of the ruler segment appearing in the image is indicated below the image in centimeter. The ground truth and the estimated POI values are also shown. The summary of the evaluation is shown in Table 3.1 for sub-sampling resize factors varying from 20% of the original image to full sized image.

We find that by subsampling to 60% of the original image size, we could still maintain 91.5% accuracy, i.e., only a 2% reduction in accuracy with respect to the baseline case of using the original image, while achieving a power reduction by approximately 86% and computation time reduction by 86%.

Table 3.1: Performance of POI algorithm for different resizing factors.

Re-size Factor	**Delay(sec)**	**Energy(mJ)**	**Accuracy(%)**
20%	0.098	3.6	75.7
40%	0.311	12.1	82.5
60%	0.899	39.3	91.5
80%	2.558	117.4	93.4
100%	6.494	292.6	93.5

3.4 Video selection Problem

After the metadata of a video is uploaded from a smartphone to the server, it is available to be selected as part of a response to a user query. In this section, we describe how Movisode selects the "right" set of videos given a query.

We illustrate the video selection problem with a simple example. In this example, we have six video clips covering different time intervals. Figure 3.6(a) shows the intervals of this set of video clips over a eight minute duration. In this example, no clip covers the entire eight minutes. The angle and POI of the video clips, labelled $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$, are shown in the Figure 3.6(b).

A set of clips may contain redundancy when we consider temporal overlap. For instance, set $\{v_1, v_6, v_3, v_4\}$ has temporal redundancy. An approach that uploads all available clips would retrieve significant amount of redundant data and is highly inefficient.

A key question that Movisode aims to address is which subset of video clips covering the query duration should be uploaded. To answer this question, Movisode considers the view angle and POI requested and the cost of each video. This decision results in a tradeoff between the accuracy of the clips as required by the orientation and temporal coverage and the power budget of the smartphones.

3.4.1 Video clip and user Query

We now formulate the video selection problem formally. A video clip v is a sequence of temporally consecutive, non-overlapping video segments of unit time duration. v is characterized by the *clip interval* $[s_v, e_v)$, where s_v and e_v are the starting and ending time of v respectively. Each video clip is also associated with two sequences that are characteristics of underlying video segments, namely the sequence of orientation angles ($\langle\theta_{v,t}\rangle$), and POIs ($\langle\gamma_{v,t}\rangle$).

As mentioned previously, a *user query* q is characterized by an interval $[s_q, e_q)$, an orientation angle θ_q, and a POI γ_q.

Let $E_{v,l,t}$ be the energy required to upload segment $v[t, t+1)$ from smartphone l. It is a product of the time to upload the segment and average power consumed by smartphone over its network interface. We obtain the average power from the Android internal API [3]. In cases where the video has already been uploaded and is cached by the server, energy cost is 0.

[3]http://devmaze.wordpress.com/2011/01/18/using-com-android-internal-part-1-introduction/

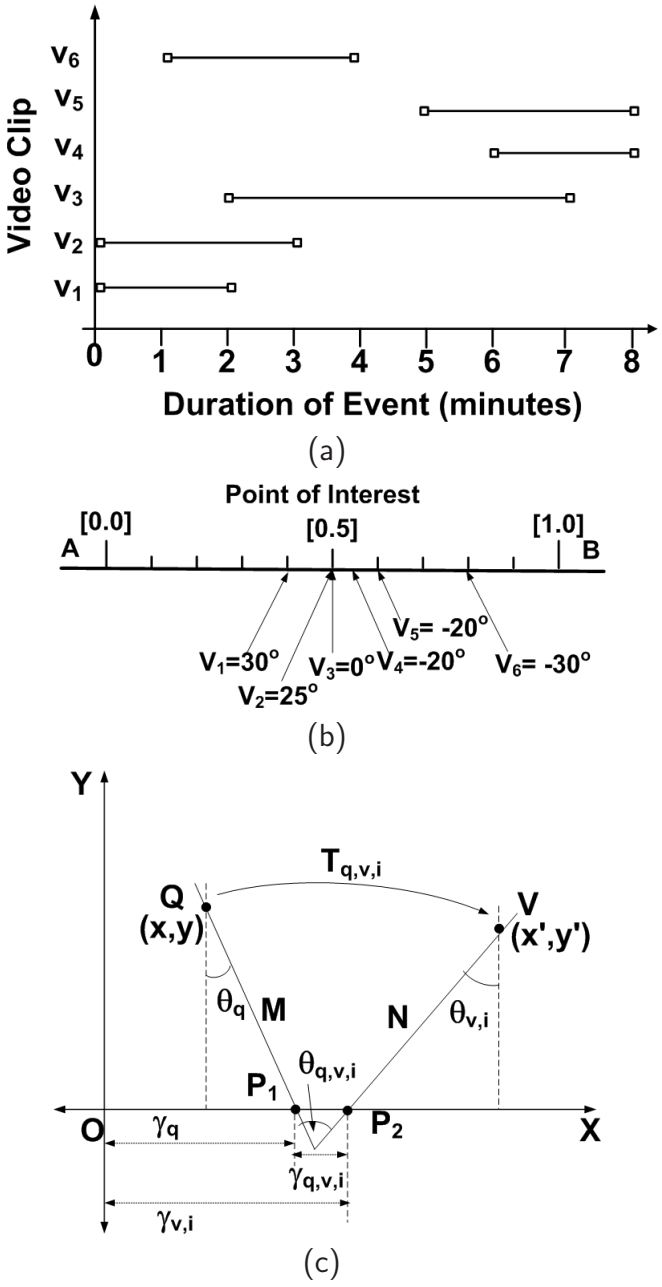


Figure 3.6: Spatial and temporal distribution of videos: (a) Example of clip intervals. (b) The event plane showing videos in Figure 3.6(a) with different view angles and POI values. (c) The event co-ordinate system. The XY-plane represents the visual components of the videos taken at the event. The Z-component is the energy consumed in retrieving the video clip.

Given a set of video clips V, the *temporal coverage* of V, is defined as the union of all clip intervals in V, i.e., $\cup_{v \in V}[s_v, e_v)$. We say that a set of video clips V *temporally covers* a given interval if and only if union of temporal coverage of V and the given interval, is equal to the given interval. Intuitively, every time instance in the given interval is captured by some clip in V. For instance, consider the video clips in Figure 3.6(a). The sets $\{v_1, v_2, v_3, v_4, v_5, v_6\}$, $\{v_1, v_3, v_4\}$, $\{v_2, v_3, v_5\}$, $\{v_1, v_3, v_5, v_6\}$ temporally cover the interval $[0, 8)$.

3.4.2 Cost Model

Three parameters determine the cost: view angle, POI, and the energy consumed. We combine these parameters into a cost model by representing them in a 3D cartesian co-ordinate system in the following way.

First, consider the XY-plane as representing a 2D plane representation of the event plane, as shown in Figure 3.6(c). Let the query be a request for a video q that views the event along the line M that passes through Q and the POI P_1. M makes an angle of θ_q with a line perpendicular to the X-axis. Next, consider a video v with a view along the line N and passes through V and the POI P_2.

We compute the *distance* between the video q and v as the "effort" involved in transforming q to v. Such a transformation can be viewed as two different operations – a rotation of the line M about the point P_1 in a clockwise direction by $\theta_{q,v,t}$, followed by a translation along the X-axis by $\gamma_{q,v,t}$. When the energy cost is included (as Z-axis), the move from (x,y,z) to (x',y',z') can be described

by

$$x' = x \cdot \cos(\theta_{q,v,t}) + y \cdot \sin(\theta_{q,v,t}) + \gamma_{q,v,t} \tag{3.1}$$

$$y' = -x \cdot \sin(\theta_{q,v,t}) + y \cdot \cos(\theta_{q,v,t}) \tag{3.2}$$

$$z' = z + E_{v,l,t}, \tag{3.3}$$

where $\theta_{q,v,t}$ is the angle between the lines Q and V. If the lines fall either side of the normal, then this value is the sum of the individual inclinations of the lines. If they fall on the same side, it is the difference in the inclinations. This value gives the angle by which the every point on the query line has to be rotated in order to align it with the video line.

$$\theta_{q,v,t} = \begin{cases} \theta_q + \theta_{v,t} & \text{if } (\theta_q \geq 0 \text{ and } \theta_{v,t} \leq 0) \text{ or} \\ & (\theta_{q,t} \leq 0 \text{ and } \theta_{v,t} \geq 0) \\ \theta_q - \theta_{v,t} & \text{if } (\theta_q, \theta_{v,t} \geq 0) \text{ or } (\theta_q, \theta_{v,t} \leq 0). \end{cases}$$

Putting in a matrix form, the transformation (clockwise rotation and then translation) matrix to align Q with V is given by:

$$\mathbf{T}_{q,v,t} = \begin{pmatrix} \cos(\theta_{q,v,t}) & \sin(\theta_{q,v,t}) & 0 & \gamma_{q,v,t} \\ -\sin(\theta_{q,v,t}) & \cos(\theta_{q,v,t}) & 0 & 0 \\ 0 & 0 & 1 & E_{v,l,t} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.4}$$

When the query and the video segments are aligned, $\theta_{q,v,t}$ $= 0$, $\gamma_{q,v,t} = 0$, and $E_{v,l,t} = 0$. $\mathbf{T}_{q,v,t}$ reduces to the following best case transformation matrix $\mathbf{T}^*_{q,v,t}$ which is a 4×4 identity matrix.

The best video segment for a query is one that is exactly aligned to the

query and incurs no energy cost. The distance between the $\mathbf{T}_{q,v,t}$ and $\mathbf{T}^*_{q,v,t}$ thus provides a measure of the cost and this *transformation distance* is represented by Frobenius norm (used to quantify geometric transformation error [40]) of the difference between $\mathbf{T}_{q,v,t}$ and $\mathbf{T}^*_{q,v,t}$ as follows: $d_F(\mathbf{T}_{q,v,t}, \mathbf{T}^*_{q,v,t}) = \left|\left|\mathbf{T}_{q,v,t} - \mathbf{T}^*_{q,v,t}\right|\right|_F$.

On simplification, we get the following equation for the transformation distance.

$$d_F(\mathbf{T}_{q,v,t}, \mathbf{T}^*_{q,v,t}) = \sqrt{4 \cdot [1 - \cos(\theta_{q,v,t})] + \gamma^2_{q,v,t} + E^2_{v,j,t}} \tag{3.5}$$

The distance can be computed using the angles from the smartphone compass readings, the POI from the images, the video sizes, the energy level reported by the smartphones, and the available channel rate estimated.

With the cost defined, the problem under consideration can now be described. Given a query q, a collection of video clip V and their availability on a set of smartphones N with their respective uploading cost, we want to find a set of video clips that (i) satisfies q, and (ii) reduces $d_F(\mathbf{T}_{q,v,t}, \mathbf{T}^*_{q,v,t})$.

3.4.3 CCMVA Algorithm

With the formulation above, we now present our solution to the video selection problem. We called our algorithm CCMVA, or Cost- and Coverage-aware Mobile Video Aggregator. CCMVA runs on the server, using the metadata information uploaded by the mobile devices as input.

The input to CCMVA is a set of video clips V, energy consumption to upload a video segment for each device, and the query q. We assume that every video clip either overlaps with the interval $[s_q, e_q)$ or is contained within the interval $[s_q, e_q)$. Otherwise the clip would not be in the query result and

can be omitted. Segments of same interval duration are assumed to be of same size.

We can model the input as a directed acyclic graph $G = (V \cup \{v_s, v_t\}, E)$, where v_s and v_t are the *source vertex* and *sink vertex* respectively. We treat v_s as a video clip with interval $[-\infty, s_q)$ and v_t as a video clip with interval $[e_q, \infty)$.

We construct the edges in G as follows: There is an edge from u to v if either of the following conditions is true: (i) u and v are consecutive, i.e., $e_u = s_v$, (ii) u and v overlaps, i.e., $s_u \leq s_v \leq e_u < e_v$. We add two self loops (v_s, v_s) and (v_t, v_t) in G (for reasons that will be clear later). Any subset of V that temporally covers $[s_q, e_q)$ forms a simple path from v_s to v_t in the graph.

V may not temporally cover the interval $[s_q, e_q)$, in which case there is no path from v_s to v_t, and G does not form a connected component. There is no subset of V that would satisfy the query q in this case. CCMVA, however, still returns a subset of video in V that falls within the interval $[s_q, e_q)$. To handle these cases, we add dummy video clips into G that fill up the gaps – for every consecutive interval $[s, e)$ not temporally covered by video clips in V, we insert a dummy video clip of interval $[s, e)$ into G. These dummy clips ensure that G is connected and a path from v_s to v_t always exist.

If G contains multiple paths from v_s to v_t, then CCMVA algorithm needs to select a path that reduces the transformation distance. Having shorter video clips, increases the flexibility of choosing different video clips with lower transformation distance. In Movisode, we break the videos captured by users into short segments, each of a maximum length L, before we feed the clips into CCMVA algorithm. We call L the *minimum switching interval*. The result of having shorter clips is that, we have more vertices in G and more possible paths to choose from. The effect of introducing the minimum switching interval will

be presented in the evaluation section.

We now consider the cost of including a video clip v in the result of query q. We consider the cost at each time instance t separately. At time t, including video clip v into result of q introduces an transformation distance $d_F(\mathbf{T}_{q,v,t}, \mathbf{T}^*_{q,v,t})$. We combine this cost over time, as c_v:

$$c_v = \sum_{t=s_v}^{e_v-1} d_F(\mathbf{T}_{q,v,t}, \mathbf{T}^*_{q,v,t}) \tag{3.6}$$

To reduce the cost, when two overlapping video clips, v_{j_1} and v_{j_2} ($s_{v_{j_1}} \leq s_{v_{j_2}} \leq e_{v_{j_1}} < e_{v_{j_2}}$), are included in the query result, we only need to upload one of the overlapped segment, either $v_{j_1}[s_{v_{j_2}}, e_{v_{j_1}})$ or $v_{j_2}[s_{v_{j_2}}, e_{v_{j_1}})$. The segment with smaller cost c is preferred. We can remove the segment with the higher cost from its original clip, effectively shortening the clip. We call the remaining shortened segment the *effective segment*. Note that the effective segment of v_{j_2} depends on what other clips are included in the query result. We, however, can pre-compute all possible effective segments for every clip, as shown below.

To consider effective segments in the algorithm, we construct a weighted graph $G' = (E, E')$ as follows. The set of vertices in G' consists of all edges E in G. We add an edge in G' between any two edges (v_{j_1}, v_{j_2}) and (v_{j_2}, v_{j_3}) in G (i.e., for any two hop path through v_{j_2}). Each edge $((v_{j_1}, v_{j_2}), (v_{j_2}, v_{j_3}))$ can now be associated with the effective segment of v_{j_2}, after considering the overlapping with v_{j_1} and v_{j_3}. Note that it is possible for a clip to have an effective segment of length 0, in which case the edge can be omitted from G' for efficiency. The number of edges in G' is $O(|E|^2)$.

Similar to G, the two vertices in G' that correspond to the self-loop (v_s, v_s) and (v_t, v_t) serves as the source and sink. Any path from (v_s, v_s) to (v_t, v_t) gives a set of effective segments that satisfy the query q. In G', however, we

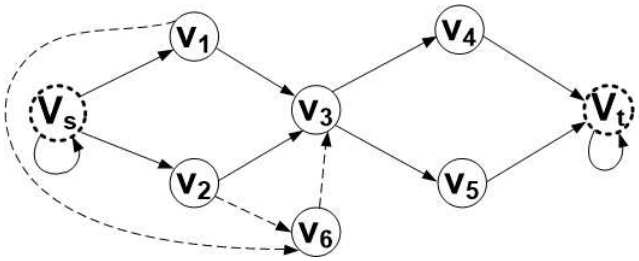


Figure 3.7: Graph G representing the video clips and the temporal relations between the clips.

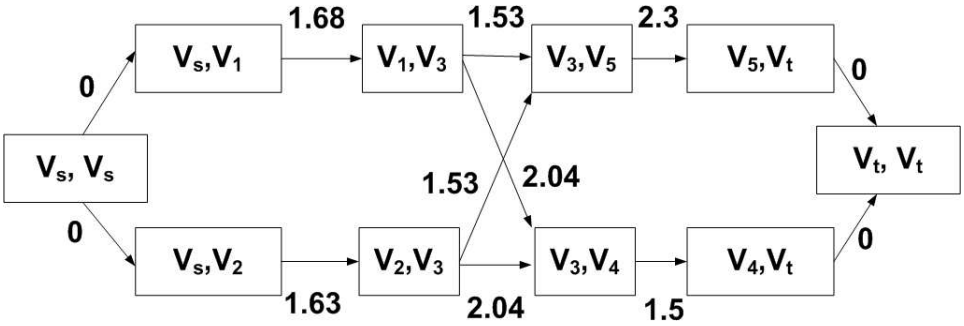


Figure 3.8: Graph G' representing the effective segments and the cost of including each segment into the query result. The cost is computed from Table 3.2. The minimum cost path is for set $\{v_2, v_3, v_4\}$ with a cost of 5.17.

can now label each edge $((v_{j_1}, v_{j_2}), (v_{j_2}, v_{j_3}))$ with $c_{v_{j_2}}$ that corresponds to the cost of including the effective segment v_{j_2} (w.r.t. overlap with v_{j_1} and v_{j_3}) into the results of q. We define $c_{v_{j_2}} = 0$ if v_{j_2} is a dummy vertex.

The CCMVA algorithm is now fairly straight forward – it merely finds the least cost path from (v_s, v_s) to (v_t, v_t) in G'.

3.4.4 Example

To elucidate the algorithm, consider the video clips with properties shown in Figures 3.6(b) and 3.6(a). Consider a query q on interval $[0, 8)$, requiring an view angle of 0° and a POI of 0.5. Upload of clips $\{v_1, v_3, v_4, v_5\}$, $\{v_2\}$ and $\{v_6\}$ are assumed to require normalized energy cost of 0.1, 0.2 and 1.0 respectively.

Table 3.2: Possible effective segments for video clips. Transformation cost is evaluated using Equation 3.6.

Clip	**Effective segment**	$\mathbf{c}_v$	**Edge pair**
v_1	$[0,2] = 2$	1.68	$((v_s, v_1), (v_1, v_3))$
v_2	$[0,2] = 2$	1.63	$((v_s, v_2), (v_2, v_3))$
v_3	$[2,6] = 4$	2.04	$((v_1, v_3), (v_3, v_4))$
v_3	$[2,5] = 3$	1.53	$((v_1, v_3), (v_3, v_5))$
v_3	$[2,6] = 4$	2.04	$((v_2, v_3), (v_3, v_4))$
v_3	$[2,5] = 3$	1.53	$((v_2, v_3), (v_3, v_5))$
v_4	$[6,8] = 2$	1.5	$((v_3, v_4), (v_4, v_e))$
v_5	$[5,8] = 3$	2.3	$((v_3, v_5), (v_5, v_e))$
v_6	No segment	-	$((v_1, v_6), (v_6, v_3))$
v_6	No segment	-	$((v_2, v_6), (v_6, v_3))$

Figure 3.7 shows the graph G for these vertices, with edges capturing the overlaps between video clips. For instance, there is edge from v_2 to v_3 because they overlap, but there is no edge between v_2 and v_4. The weighted graph G' is shown in the Figure 3.8.

Table 3.2 lists all the effective segments in the example. The effective segment is determined by the cost of overlap segment with respect to the candidate videos. For instance, v_2 and v_3 overlaps during $[2, 3]$. v_2 (with view angle 25°) has cost of 0.84 w.r.t q. Similarly, v_3 (with view angle 0°) has cost of 0.5 during this interval. Hence, overlap segment of v_3 is preferred instead of v_2. At minute 2, the effective segment of v_3 starts and the effective segment v_2 ends.

In Figure 3.8, no edge corresponds to v_6 since the effective segment for v_6 w.r.t. v_1, v_2, and v_3 are all empty. $((v_2, v_6)$, (v_1, v_6), (v_6, v_3) are marked as dotted lines in Figure 3.7). The resulting graph is illustrated in Figure 3.8. The effective segment costs (evaluated using Equation 3.6) are labelled in Figure 3.8. The min-cost solution for Figure 3.7 is $\{v_2, v_3, v_4\}$ and the cost is 5.17.

3.5 Evaluation

We evaluate Movisode in three different ways. We conducted **trace-based evaluation** in a realistic setting, we use video and sensor data of two events from the Jiku Mobile Video dataset [85], namely NAF_160312 and NAF_230312. Both events were music and dance performance on stage. We also conduct a phone **test bed evaluation** to understand Movisode performance in real network conditions. Finally, we evaluate the subjective quality of Movisode through an **user study** to verify whether the gains in objective quality metrics translate to subjective quality improvements.

3.5.1 Trace-based evaluation

To capture the factors that affect the phones' energy consumption in our evaluation, we performed video upload using the Android's YouTube app on a HTC Desire smartphone, over 3G/HSPA and WiFi, from various locations in Singapore and measured the energy required using the Android PowerTutor app [108] (PowerTutor is calibrated using Monsoon power monitor device). A 700 KB video clip was used for uploading over 3G/HSPA and a 5 MB video clip was used for uploading over WiFi.

50% of the upload were made using HSPA broadband service. The data plan supports up to 2 Mbps upload speed, but the average upload bandwidth measured was only 0.2 Mbps. The measured average power consumed was 0.96 W. The other uploads are performed with either public WiFi or enterprise WiFi. Public WiFi access tends to provide lower data rate and impose rate limit, while enterprise/campus WiFi provides much higher data rate but are much more restricted in coverage. The average upload speed of public WiFi was 0.3 Mbps, consuming 0.15 W of power. For the Enterprise WiFi, the

numbers are 1 Mbps and 0.44 W respectively.

In our evaluation, each phone is assumed to upload its video metadata to the server every 20 seconds (metadata segment duration). POI traces are extracted using the algorithm described in Section 3.3.1. One frame is sampled for every metadata segment duration and is resized to 60%, to compute the POI values for the dataset videos. The energy consumed by the POI computation is given in Table 3.1.

The upload cost for each phone is randomly assigned from the traces collected during our energy measurements. The cost is normalized to [0,1] using the maximum consumption measured (1.36 W). Queries are generated periodically, with randomly generated requested video length, view angle, and POI value.

We compare CCMVA to three baseline algorithms that differs only in the cost function in G': (i) the EnergyOnly algorithm (energy baseline), where the cost for an effective segment v stored on a smartphone l is $\sum_{t=s_v}^{e_v-1} E_{v,l,t}$, (ii) the MLDOnly (Minimum Link Delay) algorithm (uplink throughput baseline), where the device with the highest upload throughput is always chosen, and (iii) the DivergenceOnly algorithm (angle/POI divergence baseline), where the cost function (energy cost in Equation 3.5 is ignored) for an effective segment v stored on a smartphone l is:

$\sum_{t=s_v}^{e_v-1} \left(\sqrt{4 \cdot [1 - \cos(\theta_{q,v,t})] + \gamma_{q,v,t}^2}\right)$

We evaluate the algorithms using following metrics: (i) average angle divergence, (ii) average POI divergence, (iii) average energy cost incurred by a mobile device, (iv) total energy cost, and (v) average uplink throughput per query. The energy cost includes both transmission and POI computation. In the evaluation, we vary the query rate, length of video requested, and minimum switching interval.

Table 3.3: Movisode: default simulation parameters.

Parameter	**Value**
Execution frequency of algorithm	5 minutes
Minimum switching interval	20 seconds
Length of video requested	15 minutes
Query rate	0.5 query/minute
Metadata segment size	20 seconds

Table 3.3 lists the default simulation parameters. The algorithm executes every 5 minutes and if there are more than one query present, they are executed in a FIFO order.

3.5.1.1 Results

Length of Video Requested. The length of the video requested is varied from 1 to 15 minutes. Increase in video length (Figure 3.9(a)) does not affect the angle divergence of CCMVA and DivergenceOnly, since these two algorithms attempt to choose clips that decrease the angle divergence regardless of video length. In contrast, the angle divergence of EnergyOnly and MLDOnly increases for shorter queries (less then 10 minutes) and then tend to plateau as the query length increases. When longer queries are made, the likelihood of finding clip segments in the cache that match is higher, thus stabilizing the divergence as the query video length exceeds 10 minutes. The behaviour of the candidate algorithms for POI divergence (Figure 3.11(a)) is similar to that of angle divergence. CCMVA sees an increase in average angle divergence of 5.7° compared to DivergenceOnly and has 30.6% higher average POI divergence with respect to DivergenceOnly. The average angle divergence of EnergyOnly and MLDOnly is 64.7° and 79.1° respectively.

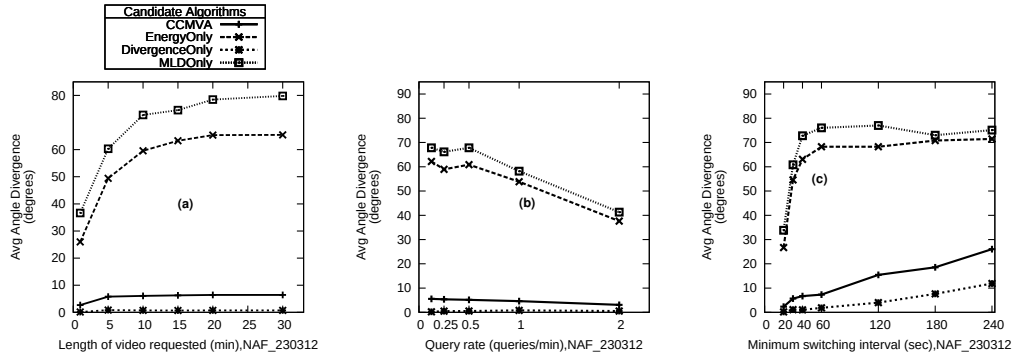


Figure 3.9: Behaviour of angle divergence with: (a) Length of video requested (b) Query rate (c) Minimum switching interval.

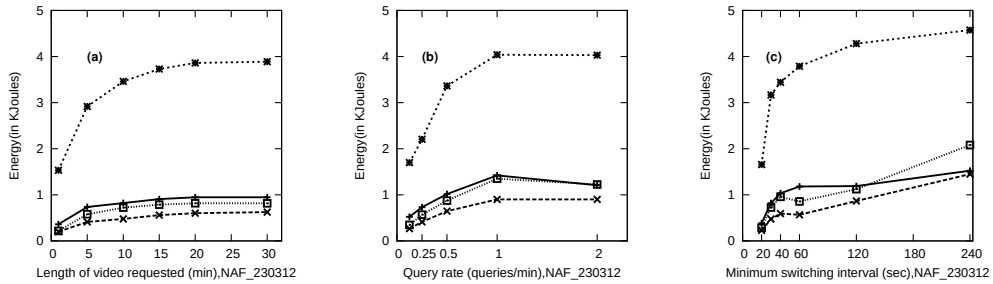


Figure 3.10: Behaviour of energy with: (a) Length of video requested (b) Query rate (c) Minimum switching interval.

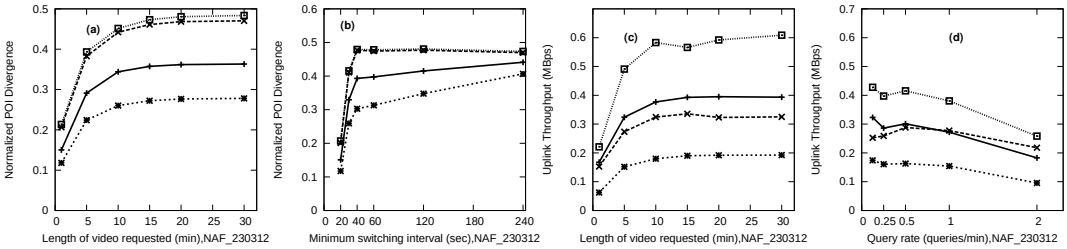


Figure 3.11: Behaviour of POI and uplink throughput: (a) POI Vs length of video requested (b) POI Vs minimum switching interval (c) uplink throughput Vs query rate (d) uplink throughput Vs query rate.

DivergenceOnly always attempts to find the best clip irrespective of the energy cost, leading to the highest energy consumption among the candidate algorithms. CCMVA also attempts to find a clip which lowers the angle divergence, but it does so by considering the energy cost involved in the upload. This approach makes its performance closer to the EnergyOnly algorithm that attempts to decrease energy cost. MLDOnly also has low energy cost, only slightly lower than CCMVA. Figure 3.10(a) shows the results. Phones consume 28.5% more energy than EnergyOnly when CCMVA is used and 4 times more energy than baseline when DivergenceOnly is used.

Figure 3.11(c) shows that uplink throughput increases for shorter video length until the links are saturated for longer video length, at which point the throughput stabilizes. DivergenceOnly shows the lowest throughput performance. The uplink throughput drops by a factor of 3 compared to MLDOnly. CCMVA and EnergyOnly have 36% and 46% lower throughput than baseline.

Query Rate. The query rate is varied from 0.1 to 2 query/
minute. Figure 3.9(b) shows how angle divergence varies with query rate. Angle divergence of CCMVA and DivergenceOnly are not affected by increase in query rate. In contrast, angle divergence of EnergyOnly and MLDOnly decrease with increasing query rate due to the effect of caching. These two algorithms tend to prefer clips found in the cache to save energy or retrieve data faster. As the number of cached video segments increases, the possibility of these algorithms finding video segments with better angular divergence also increases as a side effect. However, the angle divergence of CCMVA remains at a very low value of 4.2° with respect to baseline, while EnergyOnly and MLDOnly show about 41° of angle divergence. Since, POI behaviour is similar to angle divergence, the plots are not shown.

Figure 3.10(b) shows the variation of energy per phone with increasing

query rate. At peak rate, CCMVA consumes 20% more energy than baseline, whereas DivergenceOnly consumes 4 times more.

Uplink throughput per query (Figure 3.11(d)) drops with increasing query rate. At peak load of 2 queries/minute, the thoughput registered by baseline is 0.3 Mbps. CCMVA registers an uplink throughput of 0.2 Mbps. Throughput of DivergenceOnly is half that of CCMVA.

Minimum Switching Interval. We measure the performance of the various algorithms by varying the minimum switching interval from 20 seconds (minimum is metadata segment size) to 4 minutes. Minimum switching interval determines the shortest duration that can be chosen from any clip in generating the query result. The results for angle divergence are shown in Figure 3.9(c). As minimum switching interval increases, angular divergence of CCMVA and DivergenceOnly increase while the result for EnergyOnly stabilize after an initial increase, but the angle divergence can be at least two times CCMVA and eight times that of DivergenceOnly.

Angle divergence for DivergenceOnly and CCMVA increases because, as minimum switching interval increases, these two algorithms are forced to remain on the same clip even if there are better options provided by other clips. Behaviour of the candidate algorithms for POI divergence (Figure 3.11(b)) is similar to that of angle divergence.

The energy cost increases with switching duration for all algorithms (Figure 3.10(c)), for reason similar to that of angle divergence. Energy cost of CCMVA is closer to EnergyOnly than DivergenceOnly. At 4 minutes, CCMVA consumes slightly less energy compared to MLDOnly. This difference occurs when MLDOnly chooses clips from phones connected to Enterprise WiFi which have higher data rate, but also register slightly higher energy consumption compared to Public WiFi.

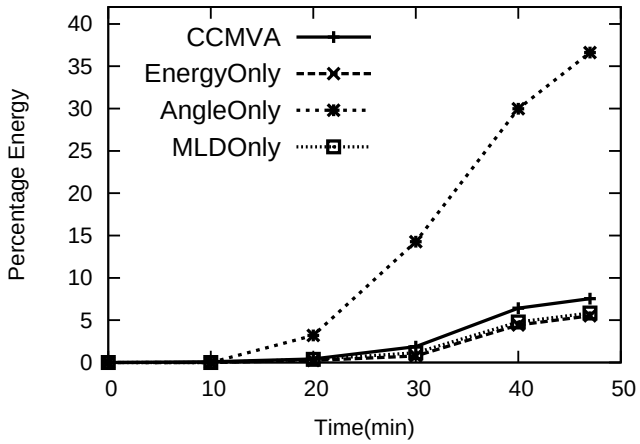


Figure 3.12: CDF of total energy of system over Time.

Different Events. So far, only results for NAF_230312 event is shown. We present a summary of results for both NAF_160312 and NAF_230312 events in Table 3.4 for comparison. The result shows that the trend is similar for both events. CCMVA performs at least 2 times better than other algorithms in terms of divergence, has 1.1 times higher uplink throughput, and consume 10 times fewer energy than DivergenceOnly. In terms of total system energy consumption (Figure 3.12 shows average over both events), CCMVA consumes only 7.5% of the energy that would have been expended if all clips in the phones were to be uploaded, compared to 5.5% consumed EnergyOnly and 36.6% by DivergenceOnly.

Contribution of Caching and Switching. CCMVA's performance is strongly influenced by two techniques used, namely caching and switching. We quantify the impact of these techniques by evaluating variants of CCMVA that use different combination of caching and/or switching. With no switching, once a user clip is selected, it is utilized till the end of the clip before another clip can be selected. Table 4.1 shows the percentage improvement in cost metric for the above schemes with respect to CCMVA. Switching has significant

Table 3.4: Summary of results for NAF_160312 and NAF_230312 events.

Divergence Baseline (DivergenceOnly)		**CCMVA**	**Energy Only**	**MLD Only**
Angle	NAF_160312	5.3°	45.1°	46.9°
	NAF_230312	5.86°	55.1°	63.3°
POI	NAF_160312	20.4%	49.3%	46.8%
	NAF_230312	24.9%	57.8%	60.2%
Uplink Throughput Baseline (MLDOnly)		**CCMVA**	**Energy Only**	**Divergence Only**
NAF_160312		26.7%	34.8%	43.5%
NAF_230312		31.9%	34.3%	65.9%
Energy Baseline (EnergyOnly)		**CCMVA**	**Divergence Only**	**MLD Only**
NAF_160312		2.2%	135.7%	7.1%
NAF_230312		28.5%	373.4%	25%

Table 3.5: Contribution of caching and switching technique measured by percentage improvement.

	NAF_230312		NAF_160312	
Metric	**Cache**	**Cache +Switch**	**Cache**	**Cache +Switch**
Angle Divergence	6.8%	87.8%	7.8%	86.5%
POI Divergence	5.3%	22.7%	3.1%	20.7%
Energy/Phone	38.9%	77.1%	20.5%	35.5%
Uplink Throughput	1.8%	19.9%	29.7%	80.8%

influence on all metrics and this is true for both event traces. The ability to switch to a different clip frequently is crucial in obtaining good solutions.

On the other hand, caching mainly reduces energy consumption of the phones and increases throughput. Caching may not necessarily decrease all

the divergence cost metrics. This is because, one cached video clip could influence the choice of other clips in the solution for the query.

3.5.2 Test bed evaluation

We conducted experimental evaluation using 10 smartphone clients to verify the performance in real settings. Six phones used WiFi to upload and four phones used 3G for communication with the server. A mix of Galaxy S2, Galaxy Nexus, and Nexus S model of phones were used for the experiment. Phones were placed in a semi-circular manner around the scene. Every phone ran the Movisode client app. A run of the experiment lasted for 20 minutes and every data point is the average of three runs.

Each client performed recording for 2 minutes followed by a 10 second idle period. This process was repeated 10 times. Metadata for the client was uploaded every 15 seconds. The average size of uploaded metadata was 150 Bytes. Request from the server consisted of videos of 5 minutes length with angle and POI request generated randomly. Accuracy of the power values provided by Android Internal API was verified with measurements taken using a Monsoon power monitor for 3 different phone models and for both WiFi and 3G in both OFF, ON and transmit state (Table 3.7). We observe that the average error of the API compared to power measurements is 26% for 3G and 20% for WiFi. While the error is not insignificant, we believe it suffice for our evaluation.

Table 3.6 provides a summary of the result. Result is consistent with the simulation evaluation in terms of relative performance comparison among CCMVA, DivergenceOnly, EnergyOnly and MLDOnly. For example, in terms of angle and POI (POI not shown in table) divergence, CCMVA is 4° and 23%

more than DivergenceOnly, while the differences for the other two algorithms are about 70° and 44% respectively. In terms of total energy (includes computation, network and sensor energy), CCMVA is 26% that of DivergenceOnly, while EnergyOnly and MLDOnly consume about 21% and 24% respectively.

Table 3.8 shows the energy expended by each algorithm on 3G and WiFi respectively. DivergenceOnly consumes more energy because it requests 3 times more uploads than EnergyOnly and 47% of it is over 3G. On the other hand, CCMVA uploads only 10% more than EnergyOnly and only 17% of it is over 3G. DivergenceOnly manages to get half the uplink throughput that MLDOnly achieves, while CCMVA is only 17% less than baseline. MLDOnly uploads more than CCMVA because of the difference in the content of the video.

Table 3.9 shows energy per-module breakdown for Movisode client. Network component dominates energy consumption for all algorithms. CCMVA and DivergenceOnly also use compass sensor (consumes 49 mW on average for normal mode of sampling) and POI computation. While CCMVA consumes 17.7% of its energy for sensors, only 0.7% is consumed for POI computation. For Divergence-Only, it is 2.6% and 0.1% of its total energy. The relative drop is because the network module of DivergenceOnly consumes 97% of total energy consumed.

3.5.3 User Study

As angle divergence is an objective measure that may not correlate with quality of user perception, we performed a user study to evaluate whether the selected video content orientation matches the view orientation expected by the user. We conducted a web-based user study with 33 users between the age group of

Table 3.6: Real phone experiment results

Metric	**CCMVA**	**Divergence Only**	**Energy Only**	**MLD Only**
Angle Divergence	29.1°	25.3°	94.3°	92.9°
Total energy (Joules)	453.9	1726.5	366.7	417.6
Uplink throughput (Mbps)	1.3	0.6	1.4	1.5
Uploaded data (MB)	45.7	115	41.5	64.8

Table 3.7: Power measurement using Monsoon power monitor device (in Watts). GN: Galaxy Nexus, S2: Galaxy S2 NS: Nexus S

Model	**Network**	**ON**	**Upload TX**	**OFF**
GN	3G	0.94	1.39	0.63
	WiFi	0.37	1.23	0.43
S2	3G	0.59	0.73	0.36
	WiFi	0.59	1.08	0.42
NS	3G	0.58	0.72	0.21
	WiFi	0.33	0.84	0.26

21-40 years. User anonymity was preserved during the study. All videos used in the study were results obtained by evaluating (Table 3.4) the algorithms on the NAF_160312 and NAF_230312 events in Jiku dataset. For each event, we select videos using the candidate algorithms CCMVA, EnergyOnly and DivergenceOnly for three view orientations (center, left and right of the stage) for a query length of 30 seconds. In total, 18 videos were used for the study.

At the beginning of the user study, participant is shown a thumbnail image (representing the query), a graphical illustration of the view orientation shown in the image and two videos. One video matches the image orientation and the

Table 3.8: Total energy consumption per interface

Interface	**CCMVA**	**Divergence Only**	**Energy Only**	**MLD Only**
3G Energy (KJoules)	0.38	1.64	0.35	0.39
3G Upload(MB)	7.9	54.9	10.28	7.96
WiFi Energy (Joules)	8.9	22.1	9.2	19.8
WiFi Upload(MB)	37.9	60.1	31.3	56.8

Table 3.9: Percentage energy breakup of Movisode client

Algorithm	**Network**	**Sensor**	**Computation**
CCMVA	85.7%	12.9%	1.4%
DivergenceOnly	96.2%	3.4%	0.36%
EnergyOnly/ MLDOnly	100%	-	-

other does not match the orientation. This is the calibration step to illustrate the difference in orientation to the user.

After the calibration step, the user is shown an image, the graphical representation of its orientation, and a video. The user has to rate on Likert scale of 1-5, the following question: "How similar is the video view orientation with that of the query image orientation?". Results of the user study are shown in the Table 3.10. The overall average ratings for DivergenceOnly, CCMVA and EnergyOnly are 3.98, 3.91 and 2.5 respectively.

The performance is observed to vary with requested orientations also. For instance, Movisode performs 97.99% better than EnergyOnly for videos requested from left-of-stage, while it is 36.64% better for videos requested from center-to-stage orientation. This is due to the distribution of available videos.

The number of videos captured from the center-to-stage orientation are more than those captured from left and right of stage. Therefore, the likelihood that EnergyOnly makes a bad choice reduces for center-to-stage videos.

Table 3.10: Summary of user study results for NAF_160312 and NAF_230312 events. Ratings are in Likert scale of range from 1 to 5

	Query	**Divergence Only**	**CCMVA**	**Energy Only**
Overall	Center	4.47	**4.24**	3.12
	Right	3.49	**4.03**	2.68
	Left	4.09	**3.22**	1.63
Average Rating		3.98	**3.91**	2.5
NAF_230312	Center	4.5	**4.19**	3.21
	Right	2.95	**3.47**	2.07
	Left	3.8	**2.46**	1.31
Average Rating		3.69	**3.51**	2.33
NAF_160312	Center	4.43	**4.31**	3
	Right	4.18	**4.58**	3.28
	Left	4.32	**4.2**	1.84
Average Rating		4.3	**4.39**	2.65

3.6 Summary

We have presented Movisode, the module of *UGCSelect* uses sensor cues available in smart phones and content features to provide spatio-temporal coverage for queries while minimizing the content upload cost. We have evaluated CCMVA (algorithm component of Movisode) through trace-based simulation driven by real-world dataset and energy traces, test bed evaluation and user study. Results indicate that CCMVA algorithm which forms part of the Movisode system balances the trade-off between spatio-temporal coverage and energy much better than the other candidate algorithms and provides results

which are close the best available videos. We evaluate Movisode through trace-based simulations on two real-world datasets [85], and also on a test bed of 10 smartphones. The summary of our evaluation is as follows:

1. Our trace-based and test bed evaluation shows that Movisode is able to select clips that are good approximations to the best match video clips (view angle and POI differ by at most 6° and 32% respectively) while incurring not more than 30% higher energy than energy baseline. The approach that considers only angle and POI produces best available matches while incurs 4 times higher cost in uploading. The approach that considers only upload cost, saves energy but selects videos with an high error in view angle and POI of up to 92° and 72% respectively.

2. The subjective quality of video results are consistent with the accuracy reported in terms of view angle and POI, with Movisode receiving 56% better ratings than the approach that considers only upload cost, while differing by only 1.6% in quality compared to the approach that considers only angle and POI.

Chapter 4

AutoLink

4.1 Outline

AutoLink is the second module in the *UGCSelect* system. In ad-hoc events, photos captured by users are unstructured. It is difficult for an user to navigate and search for specific photos in such unstructured photo collections.

AutoLink processes unstructured photo collections to create an image hierarchy. This hierarchy allows users to navigate from images with high contextual information (big picture) to images with high details (specific scene regions).

4.1.1 Difference with Movisode

AutoLink differs from Movisode in the following ways:

1. AutoLink processes images while Movisode operates on videos

2. Movisode operates in a single-scene environment, it relies on only orientation sensors to organize videos. AutoLink operates in multiple-scene

environments. In order to associate a photo to one of the scenes, it requires movement trajectories of event attendees to estimate the location where the photo was taken. Therefore, in addition to the orientation sensor feature, it also leverages the smartphone accelerometer sensors to estimate trajectories.

3. AutoLink estimates the size of the scene region captured by the photo to determine the level of scene detail within the photo.

4.1.2 Challenges

Organizing unstructured photo collection present two challenges:

1. **Lack of location information:** In events with multiple scene, identifying the scene to which photo belongs becomes challenging when localization infrastructure is not available. This challenge is also compounded when scenes located at different locations are similar in content. Figure 4.1 shows two scenes which are similar but located at different places. Image features might not distinguish between the two scenes clearly to identify the matching scene for a photo.

2. **Noise:** Photos captured in ad-hoc events could have occlusions, lighting distortions, view diversity which make the task of identifying matching features in photos difficult. For instance, Figure 4.2 shows two photos belonging to same scene from different angles. Identifying the scene region captured by a photo is challenging due to the distortion introduced by the orientation.

The organization of this chapter is as follows. We present an background for AutoLink in Section 4.2, and algorithm in Section 4.3. Results of evaluation

(a) (b)

Figure 4.1: Similar scenes at different locations

(a) (b)

Figure 4.2: Same scene from different view angles

are discussed in Section 4.4. Section 4.5 concludes the chapter with a summary.

4.2 Background

An event typical consists of multiple scenes, for example, different display booths in a exhibition. Visitors to such events tend to move from one scene to another and capture photos.

4.2.1 Problem

In this multi-scene event context, AutoLink solves the following problem in order to create the image hierarchy: Given a set of scenes identified by the user, for photo captured by the event attendee, find

1. Scene to which the photo belongs
2. Region of the scene the photo captures (describes the level of scene detail in the photo)

The desired result of AutoLink is an image hierarchy as shown in Figure 4.3.

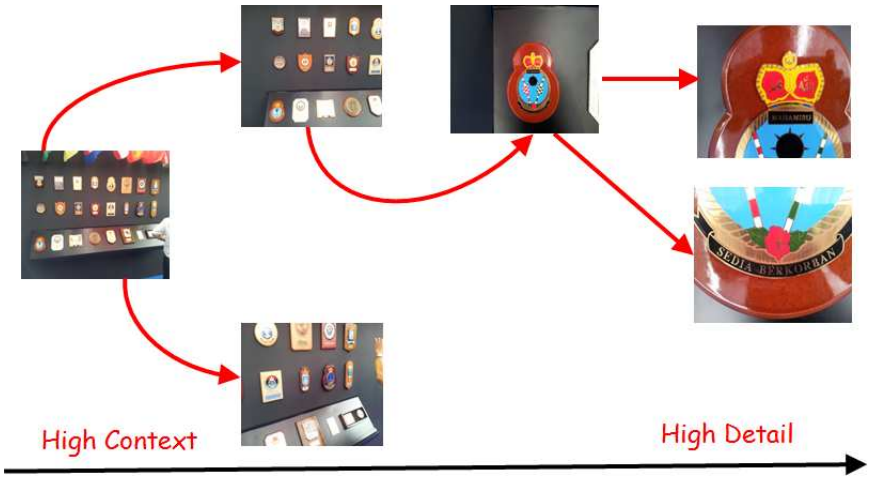


Figure 4.3: Illustration of AutoLink image hierarchy

4.2.2 User interaction

Figure 4.4 gives the general flow of AutoLink. Event attendee uses the smartphone camera to capture the photo. Each photo is timestamped, tagged with orientation sensor readings at the time of photo capture and accelerometer sensor readings between the current and previous photo capture occurrence.

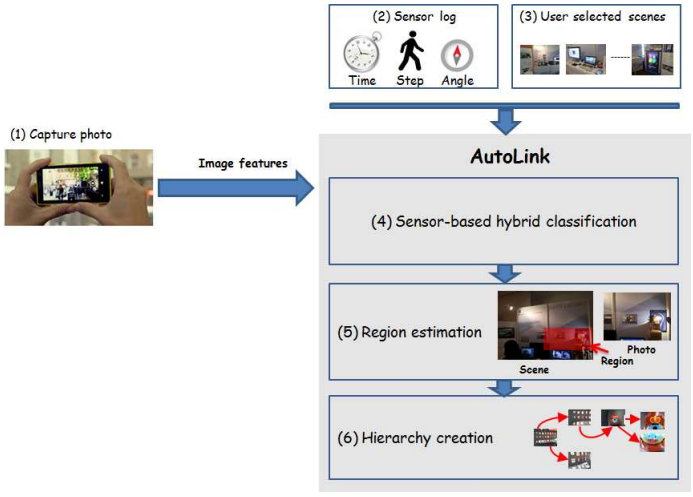


Figure 4.4: Overview of AutoLink

Image content features of the photo, thumbnail and the sensor readings are uploaded to the server. This is represented by steps 1 and 2 in the figure. The mobile client in *UGCSelect* system (Figure 1.1) handles the feature extraction and sensor tagging.

A typical AutoLink user, initiates the interaction by selecting a set of scenes (step 3 in Figure 4.4) from the available photo collection (unstructured) using a web interface.

Using the uploaded metadata and the selected scenes, AutoLink performs a three stage processing (steps 4,5 and 6 in Figure 4.4) on the unstructured photo collection to create image hierarchy. The user can then navigate through the hierarchy using a web browser.

4.3 AutoLink

The processing component of AutoLink is depicted by steps 4, 5, and 6 in Figure 4.4.

Ideally, localization infrastructure can be used to identify the scene for a photo. Due to limitations of localization infrastructure (discussed in Section 2.4), AutoLink proposes a localization infrastructure independent solution by leveraging image content features and inertial sensors to finding matching scene for photos. The *sensor-assisted hybrid classification* component (step 4 in Figure 4.4) performs this function.

As an input, an user selects a subset of images, is called the *reference images*. Once a scene is identified, the reference image for the scene is used to compute the region captured by photo. This function is performed by *Region estimation* component (step 5).

Finally, the photo is added to image hierarchy using the estimated scene and region by *Hierarchy creation* component (step 6).

Each of these stages of AutoLink are elaborated in following sections.

4.3.1 Sensor-assisted hybrid classification

Figure 4.5 illustrates an event with multiple scenes. Identifying matching scene for a given photo in a multiple scene event, is a classification task. In the following sections, we describe the features used by AutoLink and the classification method.

4.3.1.1 Content features

Content features are interest points within an image which capture properties of the image such as color, change in color, edge, region etc. For an image pair, matching interest points indicate similarity in corresponding property. We use four representative features. Each feature reveals a different characteristic of the image. They are as follows:

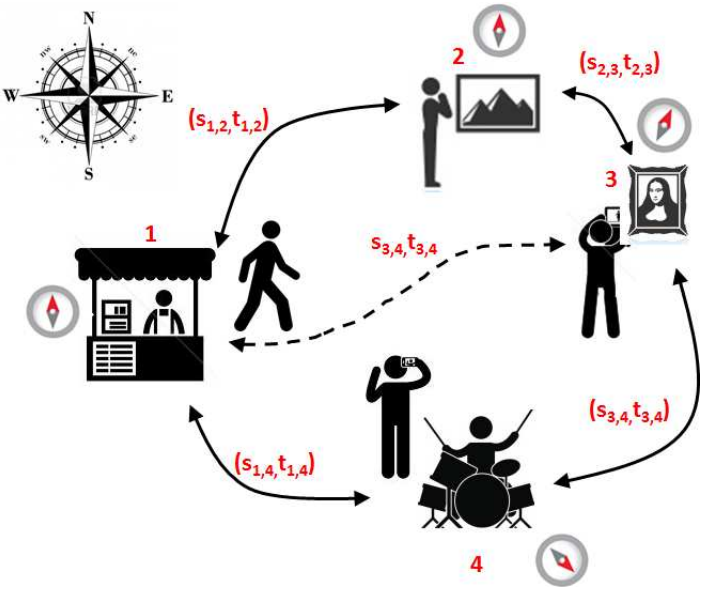


Figure 4.5: Event with multiple scenes

- **Color-SIFT**: captures scale-invariant characteristics which are invariant to color [12]. Anomalies such as ambient lighting could change the color pattern in the image. Due to its color invariance, this feature helps match two images by eliminating the effect of color change.

- **Color histogram**: global color histogram description for the photo. When the color properties of the scene are preserved, color histogram is detects matches with little processing.

- **ORB**: fast binary descriptor which is invariant to rotation and noise resistant [84]. When attendees capture photos from different views and distances, the photos capture rotated versions of the scene. To reduce mismatches due to distortion introduced by view difference, ORB features are used.

- **MSER:** or Maximally Stable Extremal Regions descriptor [34] describes region level features. While, the above three features match images at

pixel level, MSER matches images at region level.

Filtering interest points. When interest points for a image pair are matched, the matching interest points could be spurious. One interest point in the first image, could match multiple interest points in the second image. These spurious matches reduce the accuracy of classification and region estimation.

We apply three methods to filter the interest points. They are as follows:

1. **Super-pixel based filtering:** Image regions surrounding matching interest points should have have similar color properties. If not then the match is incorrect. For instance in Figure 4.6, k_i^5 matches two interest points k_r^2 and k_r^4. Matching pair (k_r^4, k_i^5) represents correct match, while (k_r^2, k_i^5) represents incorrect match. In order to extract image regions, SEEDS [100] super pixel algorithm is used to segment the image.

2. **Distance based filtering:** In Figure 4.7, distance between interest points within r cannot exceed distance between their matches with i. This is because, r is assumed to be the zoomed out version of the scene. Any image belonging to the scene will have higher detail and is captured closer to the scene. In Figure 4.7, matches (k_r^1, k_i^1) and (k_r^2, k_i^2) satisfy the distance constraint, while match (k_r^3, k_i^4) does not satisfy distance constraint with respect to (k_r^5, k_i^3). Following definition formalizes this point: Two interest point matches $(k_r^{j_1}, k_i^{j_2})$ and $(k_r^{j_3}, k_i^{j_4})$ **conform** with each other, if $DIST(k_r^{j_1}, k_r^{j_3}) \leq DIST(k_i^{j_2}), k_i^{j_4})$, where $DIST(.)$ is the euclidean distance between the interest points.

3. **Interest point clustering based filtering:** In Figure 4.6, most of the matches are positioned in the candidate image, such that it preserves

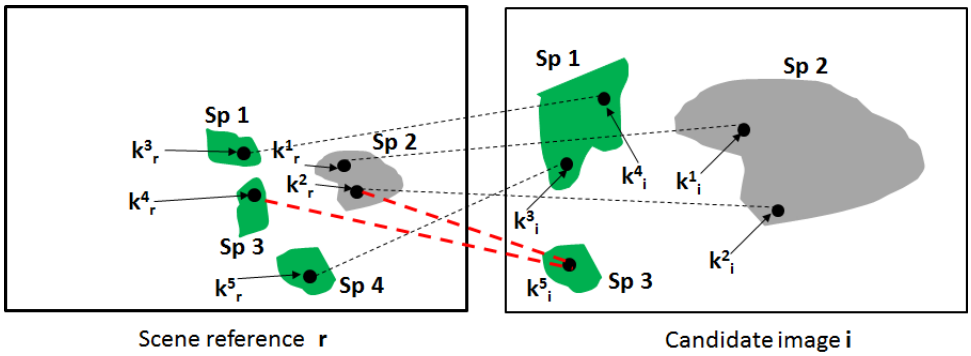


Figure 4.6: Two interest points of the scene matching with same interest point in the candidate image. This causes ambuigity and is resolved by eliminating the unlikely match

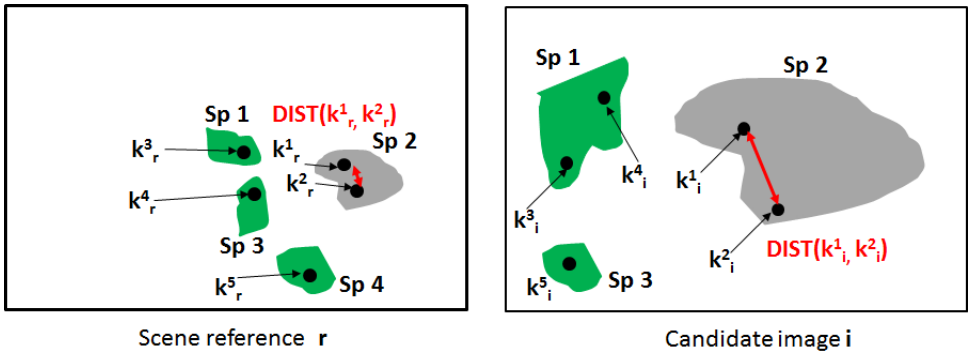


Figure 4.7: Use of distance between interest points to filter out incorrect feature matches

their relative position in the reference image. The two exceptions to this are marked in red dotted lines. In order filter out such matches, we compute the slope of the line joining the matches, cluster the slope values by computing a histogram of slopes and consider only those matches with highest occurrence slope value.

For instance, consider a two-bin histogram for slopes above or equal to zero and below zero. For the image pair in Figure 4.6, cluster size of Bin 1 is three and that of Bin 2 is two. This is because except for matches

connected by the red line, all others are above or equal to zero. Since, Bin 1 has the highest cluster size, only matches connected by black lines are considered for region matching.

4.3.1.2 Sensor-based features

In ad-hoc events, scene content can change dynamically due to human activity. Relying only on content based features will lower scene identification accuracy. Smartphone sensors could be leveraged to improve accuracy when content based features are sufficient for scene identification. Using the accelerometer and compass sensors, distance between scenes and general orientation of the scene could be obtained. We use the pedometer app [1] which computes the number of steps taken by the user, given a accelerometer sensor trace. We apply the pedometer function on the trace from reference set (scenes are manually labelled in the reference set) and create a step transition matrix $S = [s_{i,j}]$ which gives the average number of steps required to move from one scene i to another scene j, for all scene pairs.

The user cannot be expected to hold the mobile device in the hand. Between photo capture events, device could be in his pocket. This makes compass readings reliable mostly around the time when photo is captured. Therefore, only the orientation of the photo when it is captured is used as a feature. From the reference set, average compass orientation(shown in Figure 4.5) of every scene is used for scene identification.

4.3.1.3 Time-based features

In addition to sensor features, time features could also be exploited to reveal photo taking pattern of the event attendees. The time stamps between photo

[1] http://code.google.com/p/pedometer/, retrieved August 20, 2014

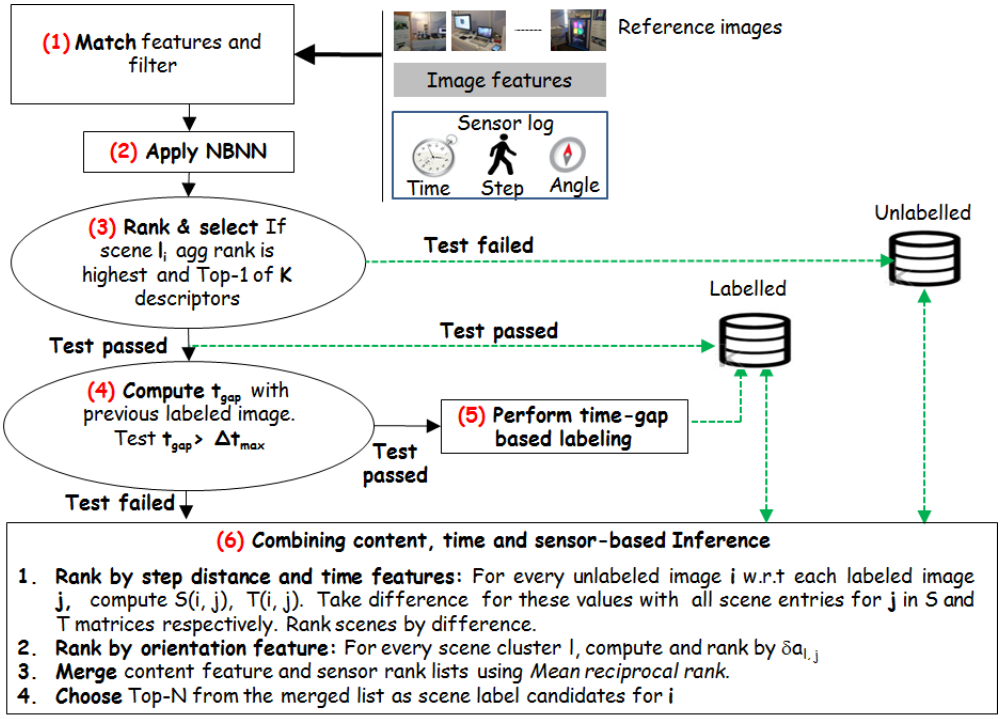


Figure 4.8: Sensor-assited hybrid classification

captures in the reference image set provide information on whether there is a scene change, and if so what is the most likely scene change. We compute a time gap matrix $T = [t_{i,j}]$ which gives the average time taken to traverse the distance between any two scene pairs (i, j). Given a time-gap and T, the most likely scene to which the newly captured photo belongs is predicted. As user trajectory is hard to predict correctly, time-gap information is used to improve the accuracy when content based features do not provide a matching scene.

4.3.1.4 Classification

Figure 4.8 shows the six steps involved in the classification which will be elaborated in this section.

1. **Match.** When a mobile device uploads the image features and sensor

logs, the features are matched with each of the reference images representing the scenes. The matches are then filtered using the method described in Section 4.3.1.1 to remove spurious feature matches.

2. **Apply NBNN.** Naive bayes nearest neighbour(NBNN) [24] classifier is applied on the matches to rank the scenes based on the nearest neighbour distance. This is done for each descriptor separately. NBNN is chosen because it works with less number of training samples. Standard NBNN uses dense features which runs slower. We alleviate the problem by having a higher threshold(set to 1000) for SIFT features. Having a higher threshold produces fewer but better representative interest points. This results in higher accuracy for fewer images. It also reduces the time for interest point matching.

3. **Rank and select.** From the nearest neighbour ranking, we look for the scene which gives the best match for the candidate image, for atleast K descriptors. Having very high K means that the scene has to have a higher matching score to be declared the label for the image. This results in fewer images being labelled, but increases the labelling accuracy. If a scene does not satisfy this criterion, the image is added to the unlabelled list and the algorithm exits. If the criterion is satisfied, then the new image is assigned(or labelled) to the scene. When a new labelled image is added to the collection, the classification attempts to discover labels for hitherto unlabelled images using the updated labelled image collection. This is done using the time and sensor features in the following steps.

4. **Compute** t_{gap}. Figure 4.9(a) shows the time gap(t_{gap}) between two successive images. If the time gap between two images is large enough, that might indicate that the user(who captured the photos) has moved

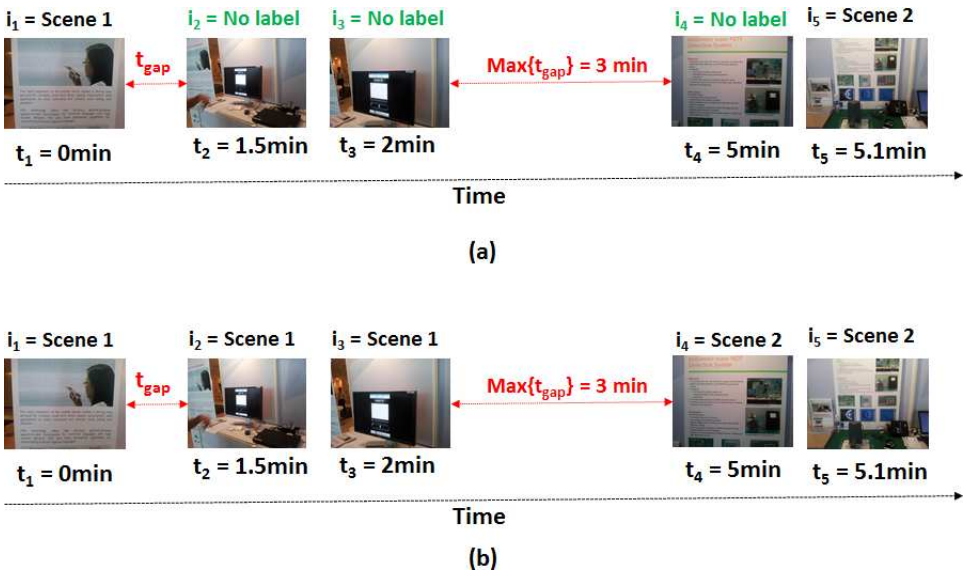


Figure 4.9: Illustration of time-gap

from one scene to another. For instance, i_3 and image i_4 in the sequence belong to two different scenes, and there is a large time-gap between them.

Between two labelled images (i_1 and i_5 in the figure) if there exactly one time-gap which is greater than a threshold Δt_{max}, and this time-gap is the maximum of all the time-gaps, then it indicates only one transition between two these scene(scene 1 and 2 in the figure) has occurred. Here, Δt_{max} is obtained from the average value of all entries of T.

5. **Perform time-gap based labelling.** If a scene transition is detected in the previous step, then the known labels are used to classify the unlabelled images captured between them. All images captured before the maximum time gap are labelled with the first image label because they were captured before the scene transition, and those occurring after the gap are labelled with the second image label because they were captured

after the scene transition. For instance, in Figure 4.9(b), known labels of i_1 and i_5 are used to label the images with unknown labels (i_2, i_3 and i_4). Since, i_2 and i_3 occur before the maximum time-gap, they are labelled with scene 1 , and i_4 is labelled with scene 2.

6. **Combining content and sensor-based inference.** If the time-gap test fails, then sensors are used to discover new labelled images. Steps of sensor-assisted classification are elaborated below:

   (a) **Step distance feature.** Steps estimated using accelerometer traces and pedometer is used to estimate the distance traversed the user two image captures. Consider two image i and j captured at t_i and t_j. Let, $\mathcal{S}(i, j)$ be the estimated step distance between i and j.

   If image i is labelled and j is not labelled, then we look for a scene label which has a similar distance from the known scene (label of image i), by comparing the estimated step distance with all entries of i in S. If i is captured before j i.e. $t_i < t_j$, then the direction of traversal is from i to j. In this case we check all row entries of i in S. Otherwise, direction of traversal is from j to i, and in this case column entries of i have to be checked.

   Based on this intuition, step distance similarity is performed on every unlabelled image j and labelled image i. For each row entry (i, x) and column entry (x, i) in S, $\delta s_{i,j,x} = |\mathcal{S}(i, j) - s_{i,x}|$ is step distance difference. We rank all scenes x in ascending order of $\delta s_{i,x}$.

   (b) **Time elapsed feature.** When event attendees walk between scenes, the time taken to traverse the distance tends to be same. For instance, in Figure 4.5, moving between scenes 2 and 3 usually

takes a shorter time than moving from scene 1 to 3. Although, it is possible that people could take a longer route between two scenes, this feature and step distance could still be used to filter out unlikely scenes in the result.

Consider two image i and j captured at t_i and t_j. Similar to step distance, time elapsed helps identify possible scenes.

For every labelled image i and unlabelled image j, time elapsed is computed as: $t_{i,j} = |t_i - t_j|$. Timestamp order is used for inferring direction of traversal. If $t_i < t_j$, for each row entry (i, x) in T, time elapsed difference $\delta t_{i,j,x} = |t_{i,j} - t_{i,x}|$ is computed. We rank all scenes x in ascending order of $\delta t_{i,j,x}$.

(c) **Orientation feature.** When the attendee captures images of a scene, the image will have similar orientation as the scene. For instance, in Figure 4.5, the person taking a photo in scene 3, will face north, north-east or north-west, but not south. Therefore, difference in orientation of the scene and the image could filter out unlikely possibilities during classification. Orientation difference $\delta a_{x,j} = |a_x - a_j|$ for all scenes x is also used to rank scenes in ascending order.

(d) Finally, all the ranks (content, time and orientation) are combined using mean reciprocal ranking [80]. The scene(s) with top N highest aggregate ranking are considered as label for j. when $N > 1$, the label one with highest rank is used for sensor based estimation in the previous steps.

Figure 4.10: Reference image (r) and candidate image (i)

4.3.2 Region estimation

Once the scene (r in Figure 4.10) of the candidate image (i in Figure 4.10) is identified, the next step in AutoLink is to identify the region (shaded area in Figure 4.10) of the scene which is captured by the image. Region estimation is done in two different ways, and combined for better accuracy.

In the first approach, the region center is estimated using the vertical and horizontal shift in the interest points. The rationale is similar to the POI computation discussed in Section 3.3.2. Finding the region center restricts the search space for the region matching process around the region center making it computationally efficient.

The second approach uses the super pixels to estimate the bounding box. The super pixels segment the image such that regions with similar pixel properties are coalesced into same region. This estimate corrects errors in the size of window based estimation.

Figure 4.11 shows the six steps involved in region estimation. They are

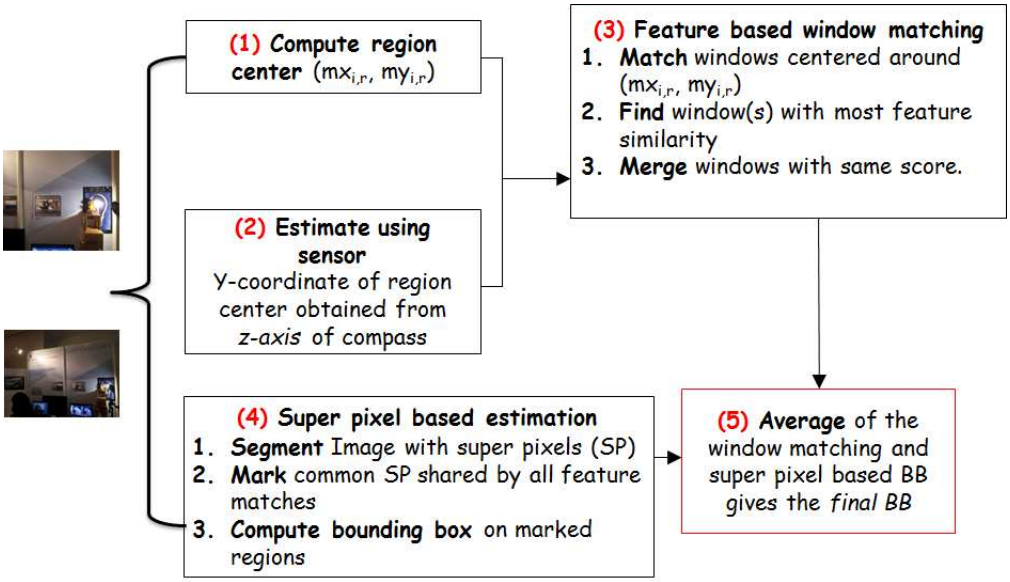


Figure 4.11: Sensor-assited hybrid region matching between reference image for scene r and candidate image i

elaborated below:

1. **Compute region center.** Let, the region center be denoted by $(mx_{i,r}, my_{i,r})$. Filtered matches for each content-based feature are used for computing the region center.

   Given an interest point match $(k_i^{j_1}, k_r^{j_2})$, $k_i^{j_1}$ is rescaled with respect to image r and the horizontal and vertical offset are computed as follows: $\left(XC(k_r^{j_2}) - (XC(k_i^{j_1}) \times \frac{w_r}{w_i})\right)$ and $\left(YC(k_r^{j_2}) - (YC(k_i^{j_1}) \times \frac{h_r}{h_i})\right)$ where w_r, h_r is the image width and height of r and w_i, h_i is the image width and height of i.

   The offset estimates obtained from each match could be positive or negative. The estimates are separated into list of positive and negative values and a weighted average is taken over each list. The weights are obtained from inverse of matching distance of interest points matches.

In order to determine the overall direction of offset, the weight (edge weight) of all positive offsets is compared with edge weight of all negative offsets. The offset direction with higher weight is favoured because it indicates that more similar matches were used to compute the offset. The corresponding weighted average is used as estimate. This is done for both the x-offset and y-offset separately. The weighted average offsets are denoted as $\Delta mx_{i,r}$ and $\Delta my_{i,r}$.

The region center of i is computed as $mx_{i,r} = mx_r + \Delta mx_{i,r}$ and $my_{i,r} = my_r + \Delta my_{i,r}$. Figure 4.12(a) illustrates the region center estimated using the above method.

2. **Estimation using sensor.** The z-axis of the compass sensor provides a measure of forward tilt of the camera which also indicates the vertical shift in the region center. We discretize the angle value and average it with the y-coordinate of the region center estimated using content.

3. **Feature based window matching.** Since, the size of the region is not known before hand, this method applies windowing iteratively (as shown in Figure 4.12(b) and 4.12(c)) by increasing the size of the window iteratively. For each window size, similarity of feature matches contained in the window is computed, until the window with maximum similarity is found. Feature based window matching is elaborated below.

   Let, size of the region around the region center be a bounding box denoted by top-left $(tx_{i,r}, ty_{i,r})$, width $w_{i,r}$ and height $h_{i,r}$.

   (a) Compute aspect ratio of i, $\frac{w_i}{h_i}$. Aspect ratio is required because the scene image and candidate image could be different sizes. This happens in smartphone images when people capture in landscape

or portrait mode.

(b) Initial value of $w_{i,r}$ is set to constant representing minimum allowable width.

(c) Height of region to be estimated is $h_{i,r} = \frac{w_{i,r}}{(\frac{w_i}{h_i})}$. This is done so that aspect ratio of region matches i.

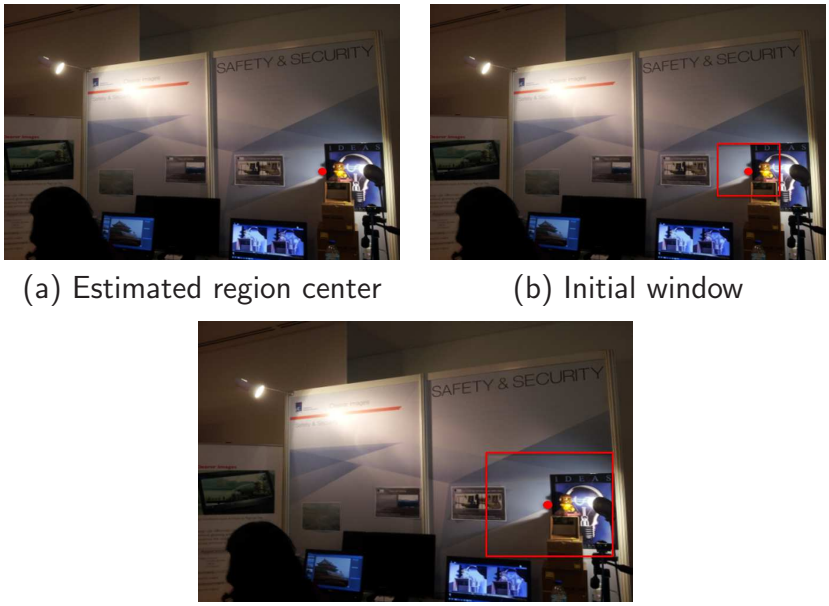
(d) For region $tx_{i,r} = mx_{i,r} - \frac{w}{2}, ty_{i,r} = my_{i,r} - \frac{h}{2}$ with width $w_{i,r}$ and height $h_{i,r}$, identify all interest point matches between r and i which are contained in this region. Average of all matching distances is the cost for this region. Higher cost implies that the region and image i are more different.

(e) $w_{i,r}$ is incremented by a fixed constant, and step (a)-(d) are repeated until $w_{i,r} < w_r$.

(f) Region estimate with minimum cost is chosen as best estimate.

4. **Super-pixel based estimation.** Windowing uses interest points to match bounding boxes. When interest points are located within regions which extend outside the window, windowing underestimates region width and height. Using super pixel boundaries for estimating the bounding box helps in overcoming this limitation. As shown in Figure 4.13, reference (Figure 4.13(a)) and candidate images (Figure 4.13(b)) are segmented. Only super-pixels with matching interest points are used for bounding box estimation (Figure 4.13(c)).

5. **Averaging bounding boxes.** Final estimate of region size is an average of the bounding box estimates obtained from both the methods. Figure 4.14 illustrates the estimated bounding box (in red) and the ground truth bounding box (in blue).



(a) Estimated region center (b) Initial window

(c) Expanded window

Figure 4.12: Feature based window matching for region estimation

4.3.3 Hierarchy creation

Image hierarchy is a poly tree $\mathcal{L} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of images and an edge exists from i to j if:

1. i overlaps with j.
2. Region size of i is greater than j
3. Children of i do not satisfy conditions (1) and (2) with j

The rationale for the above definition is as follows: the hierarchy is a polytree to allow multiple root nodes. Each root node represents a scene. condition (1) ensures image i and j share the same scene. Condition (2) ensures that j captures a sub-region of i in higher detail. Condition (3) guarantees the progressively increasing region detail along the hierarchy.

(a) Super pixel segmentation of reference image

(b) Super pixel segmentation of candidate image

(c) Super pixels in reference image that match the candidate image

Figure 4.13: Illustration of super-pixel based estimation

Figure 4.14: Region estimation: estimated bounding box (in red) and ground truth (in blue)

Once matching scene and region for the candidate image is estimated, the matching scene is used to identify the root node to which the candidate image has to be added. The region bounding box is used to iterate through the polytree until a node which satisfies the above definition is reached. The candidate node is added as a child to that node.

4.4 Evaluation

4.4.1 Methodology

This section describes the details of the datasets, metrics and candidate techniques used for the evaluation of AutoLink.

4.4.1.1 Datasets

Since there are no datasets with accelerometer, gyro and compass sensors, we have collected two real-world datasets which contain sensor annotated images captured using smartphone. Participants were allowed to capture data without

any restriction on their behaviour. Every scene is represented by one reference image determined manually offline. Ground truth labelling was done manually.

Dataset 1. is an exhibition event which involved both indoor and outdoor exhibits spread over a large area. This means that photo captures are separated in time and space and are relatively easier to distinguish. Dataset 1 contains data from 1 participant with 755 photos. We refer to this as the *single user case*. Dataset 1 has 43 scene classes and Dataset 2 has 22 scene classes. Since, there was only one participant in Dataset 1, we have sub-sampled the sensor traces and split the photos to create two virtual users. This dataset is considered as the best case which simulates two people with very similar walking trajectories.

Dataset 2. is captured in an exhibition event held in an indoor environment where scenes involving posters and demos. Scenes were closely located. The scenes also share some common features which makes distinguishing them challenging. Dataset 2 contains 187 images with sensors logs, obtained from 6 participants who volunteered for the evaluation. We refer to this as *multiple user case*. For all results of Dataset 2, sensor data from 5 participants were used for constructing matrices S and T.

4.4.1.2 Metrics and parameters

Metrics of evaluation are described below:

1. **Precision** is defined as the ratio of correctly labelled images to number of labelled images.

2. **Recall** is defined as the ratio of correctly labelled images to number of images.

3. **Running time** of the candidate techniques

4. **Bounding box accuracy** is used to compute the accuracy of region matching [109]. The ground truth bounding box is obtained by manual human labelling of the regions. Let, A_g be the area of ground truth bounding box and A_p be the area of predicted bounding box. The accuracy is given by:

$$\text{Bounding box overlap } (\%) = \frac{(A_g \cap A_p) * 100}{(A_g \cup A_p)} \tag{4.1}$$

There are two parameters that impact the accuracy of AutoLink. They are as follows:

1. **K:** If a candidate scene appears as the best match in the rank list of atleast K feature descriptors types, then it is accepted as label for the image. We have used four descriptor types (discussed in Section 4.3.1.1). The default value for evaluation is $K = 2$. We vary K from 1 to 4 and observe its effects on classification accuracy.

2. **N:** This parameter is the maximum number of candidate scenes that are considered potential labels for an image. Default value for evaluation is $N = 1$.

4.4.1.3 Candidate techniques

Evaluation of AutoLink is done in two phases. For the classification phase, following are the candidate techniques used for evaluation:

1. **NBNN only.** This is the content only version of AutoLink.

2. **NBNN + Time.** This method is AutoLink without sensors. We evaluate this scenario to get the accuracy gain from content and time based methods.

3. **NBNN + Sensor features.** This method is AutoLink without time-gap features. We evaluate this scenario to get the accuracy gain from content and sensor based methods.

4. **NBNN + Time + Sensor.** This is the AutoLink with all features included.

5. **Bag-of-visual-words (BOVW).** OpenCV [6] version of the bag-of-visual words (BOVW) is used as the candidate technique for training based algorithms. BOVW extracts interest points from images and clusters them into patterns called "visual words". A visual dictionary of such words are created using training images. A test image is classified by matching its visual words against the dictionary. This method has been used by ImageWebs [41] for image similarity graph.

6. **Structure-from-Motion (SfM).** SfM extracts interest points from a set of images, uses point correspondence to construct a 3D model of the scene. This 3D model is used to estimate the viewpoint of each photo. This technique is used in PhotoTourism [92]. We have used VisualSfM [105, 104] for our evaluation.

For the region matching phase, the candidate techniques are:

1. **Window matching only.** This technique only estimates the region center and the size based on window matching described previously. The region center is not corrected with sensors.

2. **Window matching + super pixels.** This technique combines the region estimates from window matching and super pixels.

3. **All techniques.** This is the AutoLink region matching, which combines sensor correction with window matching and super pixel estimations.

When only one dataset results are discussed, by default it is the multiple user case (Dataset 2).

4.4.2 Classification

We first discuss the single user case of Dataset 1 where image hierarchy is built using only the personal image collection. We will then elaborate on the case of multiple participants, where image collections of more than one user are used to build the image hierarchy.

Single user case. Table 4.1 summarizes the results for both datasets. In the single user case, the scene images chosen by user and the images to be classified share the same sensor and time log. This is the scenario where classification should produce good accuracy. As expected, AutoLink demonstrates 70% precision and high recall of 78%.

In Dataset 1, the time-gap feature distinguishes scene transitions with reasonable accuracy because of larger area and content difference between scenes is high. Due to this reason, sensors play a less significant role. Without sensors, precision drops by 8% and recall drops by 6%. Without time-gap feature, recall drops by 51%.

The content-only component which is NBNN, does provide a high precision of 78%. Most of the images that could have been labelled using time and sensor features remain unlabelled, resulting in low recall of only 10%.

For BOVW which relies on visual dictionary, the lack of enough training data (only 1 image per scene) results in poor performance. The precision

reduces to 19% and recall to 3%.

Compared to BOVW, SfM based image clustering does better at 53% precision and recall. SfM does not rely on content training. The main source of error for SfM is that it classifies based on "similar" scenes rather than "same" scenes. When scenes are similar in appearance, use of sensors and time features help distinguish them accurately.

Multiple user case. In the case of multiple users, one user chooses the scene images, and images to be classified is received from other users. The sensors and time features are not the same. I.e., each user might capture images in different sequence and move between scenes in different ways.

AutoLink gives a good precision of 71% which is the same as the single user case. The recall is relatively lower at 51%. The difference with the single user case is 37%. This lower recall is because the time features which gave high accuracy in Dataset 1, are not as accurate in the case of Dataset 2. This is because Dataset 2 is a indoor event with closely clustered scenes. The time difference between people spending time in a scene, or moving to the next one is not very different.

When the sensor module is removed, the recall drops by 37% and precision by 9%, demonstrating that the major contributor to the number of correct classifications, in this case, is sensors.

Without the time-gap module, the recall drops by 27%. The drop is less compared to Dataset 1 because sensor module contributes to classification more significantly than time-gap features when scenes are clustered in small spaces.

The NBNN only gives 25% recall which is 51% drop from AutoLink classification. This slight rise of 7% in precision is because AutoLink without time features makes less errors in classification, although it classifies less.

Compared to the single user case, SfM performs very badly at 37% precision and recall. Dataset 2 has higher number of scenes with similar features compared to Dataset 1 which is a mix of indoor and outdoor scenes. AutoLink provides twice the precision with 37% higher recall than SfM.

Finally, BOVW performance is similar to the single user case with 19% precision and recall.

When the number of participants used to construct the transition matrices was reduced to 1, the precision and recall reduced by 5% indicating that one participant's trace is enough to build the transition matrices to get a reasonable accuracy.

Table 4.1: Classification accuracy: precision and recall

	Single user		Multiple users	
Approach	**Precision**	**Recall**	**Precision**	**Recall**
NBNN + Time + Sensors	0.7	0.78	0.71	0.51
NBNN + Sensor	0.76	0.38	0.74	0.37
NBNN + Time	0.64	0.73	0.64	0.32
NBNN only	0.78	0.1	0.78	0.25
SfM	0.53	0.53	0.37	0.37
BOVW	0.19	0.029	0.19	0.19

Variation across phones. Table 4.2 shows the variation of precision and recall across three of the six phones. The standard deviation in precision is 1.9% is very small, indicating that precision of AutoLink is not affected by the number of images captured by the user. But, there is 26% variation in recall. The trend indicates that participants capturing fewer images tend to receive a lower recall. This is because, when the number of images is less, the likelihood of getting good quality scene images to create hierarchy decreases.

Sample results of classification. Figure 4.15 shows classification results for two scenes from Multiple user dataset (Dataset 2). Correct classification event when image captures parts of the scene in higher detail is illustrated

Table 4.2: Variation in precision and recall across phones

Approach	Precision	Recall	Number of photos captured
Participant 1	0.65	0.28	56
Participant 2	0.64	0.46	69
Participant 3	0.67	0.47	118

(a) Scene 1 (b) Higher detail, correct (c) Different view, correct

(d) Scene 2 (e) Occluded image, correct (f) Incorrect

Figure 4.15: Samples of image classification. Correct and incorrectly classified images are shown

in Figure 4.15(b). Classification when image has different view angles is illustrated in Figure 4.15(c). Semi-occluded image classified correctly is shown in Figure 4.15(e). Incorrectly classified image is shown in Figures 4.15(f). Incorrect classification is because of the fewer interest points generated for the image content.

Running time. Table 4.3 gives the running time of candidate techniques. For the multiple user dataset, the running time of AutoLink is at most 298 milliseconds when run on a 4 core 3.4GHz Intel i7 processor. Removing sensor and/or time features results in a negligible decrease in running time. Bag-of-Features has a much lower 35 ms running time, it does not perform well

in terms of accuracy. SfM run on the same machine takes 5.9 seconds/image which is 20 times more than AutoLink. Both AutoLink and SfM use multi-threading.

Table 4.3: AutoLink running time

Approach	**Average running time per image**
NBNN + Time + Sensors	298 ms
NBNN + Time	292 ms
NBNN + Sensors	290 ms
NBNN only	287 ms
SfM	5946.7 ms
Bag-of-Features	35.9 ms

Varying N and K. In Table 4.4 and 4.5 we vary the value of N from 1 to 10 and K from 1 to 4. Increase in N increases the chance of detecting the correct scene match. The variation from 1 to 10 yields a precision and recall improvement of upto 29%. Increasing K does not necessarily increase the precision and recall. While the reason for decrease in recall is because less images are labelled correctly, decrease in precision is because, for higher K, AutoLink has fewer content labels which increases the uncertainty for the sensor-based labelling. Having a very low K value also affects precision because there will be more false content matches. The best precision and recall is achieved when $K = 2$, which is also used for our evaluation.

Table 4.4: Variation of precision with (N,K) parameters

N	**K=1**	**K=2**	**K=3**	**K=4**
1	0.55	0.71	0.66	0.35
3	0.52	0.59	0.54	0.28
5	0.55	0.65	0.59	0.34
10	0.57	0.67	0.61	0.35

Table 4.5: Variation of recall with (N,K) parameters

N	**K=1**	**K=2**	**K=3**	**K=4**
1	0.46	0.51	0.44	0.19
3	0.52	0.58	0.53	0.25
5	0.55	0.64	0.58	0.29
10	0.57	0.66	0.60	0.31

4.4.3 Region estimation.

Figure 4.16 gives the CDF of percentage overlap of the candidate techniques, for the photos that are correctly classified. 58% of photos have atleast 50% overlap with the ground truth bounding box, when all techniques (superpixel, window matching and sensors) are applied. For window matching based region estimation, only 30% of photos have 50% overlap. This is because, window matching relies on interest points and color features only and does not factor in the semantic region boundaries. Lesser overlap is also due to the error of the region center which affects window position. When super pixels are included, 38% of photos have 50% overlap. There is an 8% increase due to considering the region boundaries. Running time per image pair for region matching module of AutoLink is 73 milliseconds.

Figure 4.17 shows the sample estimations for region matching. Blue rectangle denotes the ground truth and red rectangle denotes the estimation. Figure 4.17(c) shows estimation for candidate image which captures changed scene content. Figure 4.17(f) shows estimation for image with higher detail and Figure 4.17(i) shows estimation for occluded image.

4.4.4 Overall accuracy.

User interacts with AutoLink by drawing a bounding box in a scene image. AutoLink lists top-M images that belong to this scene and overlap with the

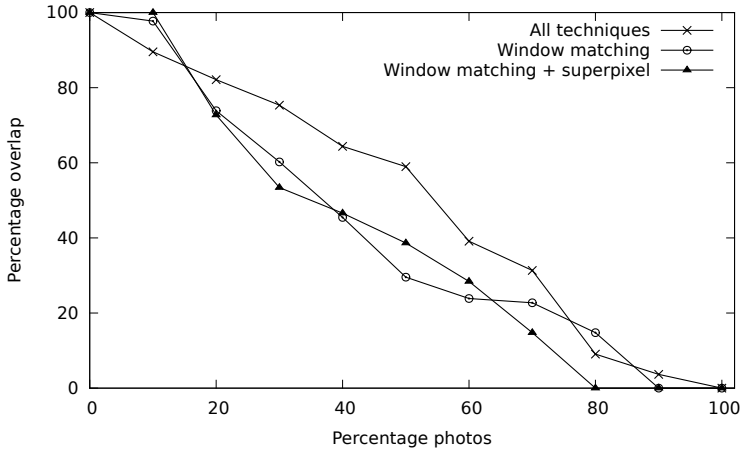


Figure 4.16: Region matching accuracy

bounding box. We measure the accuracy of this list as follows:

In the dataset, we have manually classified and labelled the bounding box for every image. From the set of all possible scenes in the dataset, we randomly select a scene and generate a rectangle bounding box representing the user's request. In all, we generate 10000 user requests. For each request, the best possible match is identified from the ground truth. AutoLink is used to predict a list of candidate images that overlaps with the user request. The list is ordered in decreasing order of bounding box overlap with the user request.

Overall accuracy is measured as follows: if there is atleast one image in the Top-M entries of AutoLink list, which matches the ground truth overlap percentage, then the user request has a match. Otherwise, the user request has no matches. Table 4.6 gives percentage user requests that have a match. We have generated 10000 random user requests on Dataset 2, for this evaluation.

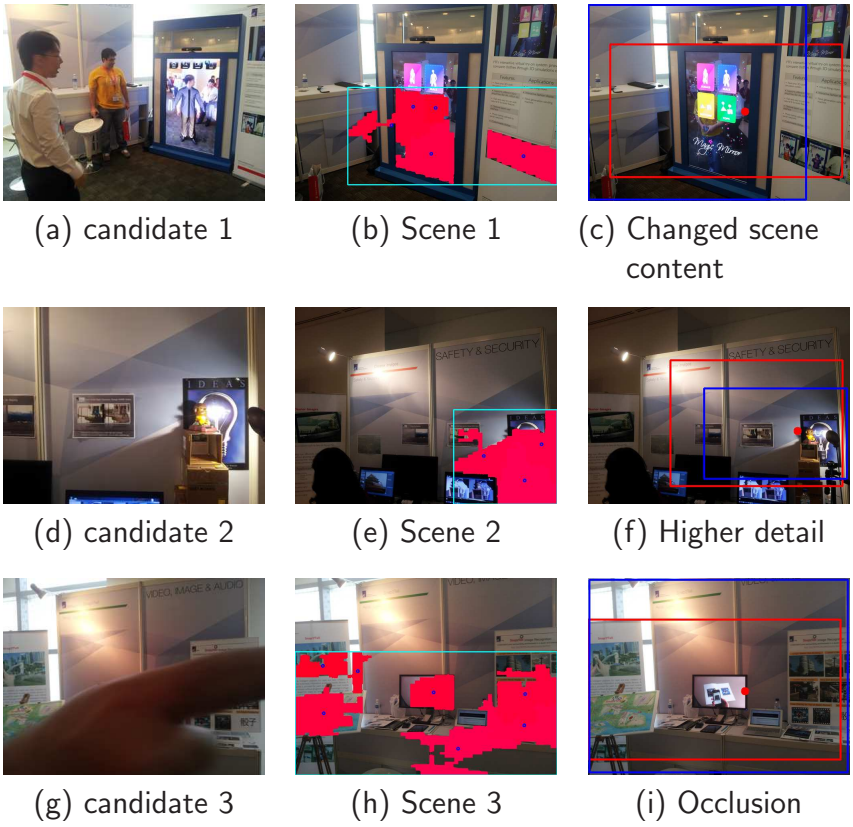
The value of M is varied from 1 to 20. For $M = 1$, accuracy is 56.6%. As expected accuracy increases with more entries in the predicted list, but saturates at $M = 15$. At $M = 2$, 70% of user requests have atleast one matching prediction.

Table 4.6: Overall accuracy of AutoLink

Top-M predictions	Percentage accuracy
1	56.6
2	70.4
3	74.9
5	76.9
10	80.14
15	81.11
20	81.11

4.5 Summary

In summary, this chapter details AutoLink, the sensor- assisted module of *UGCSelect* to automatically generate image content links so that smartphone photos captured in ad-hoc events, could be organized to improve user experience. We have demonstrated its performance with two real world datasets and also suggested potential application scenarios. Evaluation results indicate upto 70% precision, and atleast 50% recall in classification and 20 times faster than state-of-the-art.



(a) candidate 1 (b) Scene 1 (c) Changed scene content

(d) candidate 2 (e) Scene 2 (f) Higher detail

(g) candidate 3 (h) Scene 3 (i) Occlusion

Figure 4.17: Samples of region estimation

Chapter 5

CoFiGel

5.1 Outline

CoFiGel is the last of the three modules of *UGCSelect* (Figure 1.1). When infrastructure is not available, CoFiGel facilitates content sharing using m2m network. CoFiGel uses subjective UGC attributes to select content. Subjective attributes have the following advantages:

1. Content processing (which is CPU intensive) is not required
2. Captures abstract user taste which is not possible in content based selection
3. Requires small quantity of user feedback metadata to describe content and infer user interest. Disseminating small metadata consumes less resources in m2m networks.

CoFiGel uses recommendation systems to process subjective attributes of UGC. In Figure 2.1, we show two main types of recommendation techniques designed for infrastructure networks, that leverage user feedback to predict

user interest. The first technique is model based recommendation which builds a model with training user feedback set, and then uses the model to predict user interest. The second technique is memory based collaborative filtering (MCF). This method of recommendation differs from the model based approach in that it does not build a model which is hard to change when the user set changes. MCF builds a rating matrix from the user feedback for content, and extracts similarity patterns between users to make predictions. This rating matrix can be built incrementally over time. MCF has also seen wide deployment in commercial products [1]. Due to these advantages, MCF is a good candidate for subjective content selection in infrastructure-less m2m networks.

5.1.1 Problem

The performance of m2m networks depend on the following factors:

1. Storage capacity of the mobile device
2. Duration of the m2m contacts
3. Frequency of m2m contacts
4. Number of replicas of the content available in the m2m network

MCF does not consider these factors while making predictions for the user. Using MCF as it is results in wasteful use of resources and poor delivery of items to users in m2m networks.

CoFiGel solves this problem by combining the network characteristics of m2m and the predictions of MCF to share UGC. As part of the solution we propose, a distributed transmission scheduling and storage management

[1] http://www.amazon.com/

algorithm, which balances the trade-off between faster growth of the rating matrix of MCF, while increasing the delivery of interesting items to users in a timely manner.

5.1.2 Differences with Movisode and AutoLink

CoFiGel differs from Movisode and AutoLink in the following ways:

1. CoFiGel is designed for a distributed environment, because it operates in m2m networks. Movisode and AutoLink use m2s links and are designed for a client-server environment.

2. Movisode and AutoLink use objective attributes to share content, while CoFiGel uses subjective attributes of content.

The chapter is organized as follows: Section 5.2 elaborates on MCF. The trade-off involved when combining MCF and m2m is detailed in Section 5.3. System model for CoFiGel is presented in Section 5.4. Utility estimation is discussed in Section 5.5. Section 5.6 discusses the algorithm, Section 5.7 details the evaluation and Section 5.8 concludes the chapter with a summary.

5.2 Memory-Based Collaborative Filtering

We now explain in more detail how MCF works. Typical MCF techniques have the following structure. A training data set is used to build a rating matrix consisting of ratings given for items by users. The rating matrix is used to identify the similarities between users/items and also to predict the ratings of hitherto unrated items by a given user. From this sorted list of predicted ratings, a subset of highly rated items are shown to the user. Feedback from

the user for the item shown is then used to update the rating matrix. The assumption is that *users tend to behave in the same way as they behaved in the past.*

For concreteness, we will use the Cosine-based similarity metric ([65, 52, 74]) in the rest of this paper to illustrate how CoFiGel works. Cosine-based similarity is a popular item-based MCF and has been used in large scale real-world applications such as the recommendation system used by *Amazon.com*. Note that CoFiGel can also work with other MCF algorithms, such as Slope One [61]. Since MCF works for recommendation of any kind of items, we will use the term *items* in the rest of the discussions to refer to content in our application.

In general, ratings can be represented as integer values. For simplicity, we assume that ratings are binary and are expressed as either 1 (positive/like) or 0 (negative/dislike). In computing Cosine-based similarity, unrated items are assigned ratings of 0. After a user has rated an item, the item will not be recommended to the user again.

Let U and I be the set of all users and items respectively and I_u^+ and $I_u^?$ be the set of items that are rated positive and unrated by a user $u \in U$ respectively. Let the rating of an item $i \in I$ for user u at any given time be $m_{u,i}$. Cosine-based similarity metric computes $RNK_{u,i}$, the *rank* of an unrated user-item pair (u, i), in the following way. First, the similarity between two items i and j is computed using

$$Sim(i,j) = \frac{\sum_{u \in U} m_{u,i} \cdot m_{u,j}}{\sqrt{\sum_{u \in U} m_{u,i}^2} \cdot \sqrt{\sum_{u \in U} m_{u,j}^2}} \tag{5.1}$$

Table 5.1: Rating matrix for Cosine-based similarity metric, $\diamond$ denotes ratings that could be predicted and $\star$ denotes unknown ratings

Users	i_1	i_2	i_3	i_4	i_5	i_6
u_1	1	$\diamond$	$\star$	$\star$	$\diamond$	$\diamond$
u_2	1	$\diamond$	$\star$	$\star$	$\diamond$	$\diamond$
u_3	1	1	$\star$	$\diamond$	$\diamond$	1
u_4	$\diamond$	1	$\diamond$	1	1	1
u_5	$\star$	$\diamond$	1	1	$\diamond$	$\diamond$
u_6	1	$\diamond$	$\star$	$\diamond$	1	1
u_7	1	0	$\star$	$\diamond$	0	1

For each unrated item $i \in I_u^?$, $RNK_{u,i}$ is computed as:

$$RNK_{u,i} = \sum_{j \in I_u^+} Sim(i,j) \tag{5.2}$$

Obviously, the rank of item i for user u can be computed only if there is at least one user who has rated both i and another item that user u has rated positively. If the rank cannot be computed, then we say that the particular user-item pair is *unpredictable*. Table 5.1 shows a rating matrix with items that are rated (positively and negatively), predicted and unpredictable.

Typically, the top-k items $i \in I_u^?$ with highest rank are recommended to user u. We say that the prediction of i is *positive* for u if i is among the top-k items in $I_u^?$, and *negative* otherwise. A prediction of i is said to be *correct*, if the predicted rating is consistent with the user rating eventually. Note that the notion of whether a prediction is positive or not changes over time (and thus whether it is correct or not changes over time as well).

The performance of MCF algorithm is measured by several standard metrics [94]. For instance, *precision* and *recall* are used to measure the classification performance of a MCF algorithm. Precision is a measure of recommended items that are relevant to the users, and recall is a measure of the number of

Table 5.2: Predicted rating and coverage for (u_4, i_1) and (u_4, i_3) user-item pairs

User-Item	**Predicted Rating**	**Gain in rated and predictable items**
(u_4, i_1)	1.30	2
(u_4, i_3)	0.71	4

relevant items that are recommended to the users. Another common performance measure used is *prediction coverage*, (or coverage for short), defined as the percentage of *the number of predictable user-item pair*.

5.3 MCF for Mobile-to-Mobile Recommendation

When two mobile devices meet, they need to select which items to be transmitted over the intermittent contacts based on the meta-data information available. As mentioned, since contact capacity is precious, items that are likely to be liked by other users should be transferred and propagated with higher priorities. Running MCF in the context of mobile-to-mobile content sharing, however, leads to another issue: since each user is likely to get a chance to rate only a small subset of all content available, selecting which items for users to rate is also important, to increase the coverage. We illustrate this intricacy in the rest of this section with an example.

Consider the rating matrix shown in Table 5.1. Item i_1 has three common user ratings with items i_2, i_5 and i_6. i_3 has only one common user rating with i_4. Using Equations 5.1 and 5.2, we can compute RNK_{u_4,i_1} as,

$$\begin{aligned} RNK_{u_4,i_1} &= Sim(1,2) + Sim(1,4) + Sim(1,5) + Sim(1,6) \\ &= \frac{1}{\sqrt{5}\sqrt{2}} + 0 + \frac{1}{\sqrt{5}\sqrt{2}} + \frac{3}{\sqrt{5}\sqrt{4}} \approx 1.30 \end{aligned}$$

Similarly,

$$RNK_{u_4,i_3} = Sim(3,2) + Sim(3,4) + Sim(3,5) + Sim(3,6)$$
$$= 0 + \frac{1}{\sqrt{1}\sqrt{2}} + 0 + 0 \approx 0.71$$

The results are listed in Table 5.2. i_1 has a higher rating than i_3 with respect to u_4.

However, the consideration, in terms of coverage, is different. It can be observed from Table 5.1 that all users except u_4 and u_5 have already rated i_1. Knowing the value of m_{u4,i_1}, allows only at most one more rating, RNK_{u_5,i_1}, to be computed. The gain in rated and predictable items is 2. On the other hand, i_3 has been rated only by u_5. Knowing the value of m_{u_4,i_3}, allows the rating of 3 users (u_3, u_6 and u_7) for item i_3 to be computed. The gain in rated and predictable items is 4. Therefore, the rating of i_3 by u_4 has a higher gain in rated and predictable items than rating i_1.

This example illustrates the trade-off between improving user satisfaction and improving coverage when not all data transfer can be completed within a contact. If user satisfaction is more important, then i_1 will be chosen for transfer. If coverage has higher priority, then i_3 should be chosen.

Note that when there is a centralized server with continuous connectivity to users and has access to all rating information and data items, the impact of this trade-off is not significant. However, such a trade-off plays an important role in a resource constraint environment where the contacts between mobile devices are intermittent, contact capacities are limited and only subsets of data items can be stored in the local buffers.

The execution of MCF on mobile devices with intermittent contacts presents a new challenge that is not present in traditional applications of MCF in a cen-

tralized or peer-to-peer environment where connectivities are not intermittent.

5.3.1 Rating Propagation

Consider the case of item i_3 shown in Table 5.1. The only available known user rating for i_3 is from u_5. Using the rating matrix shown, only user u_4's rating can be predicted.

Let, $\mathbb{U}_i^j$ be set of users with unknown ratings for item i in j^{th} round of prediction, $\mathbb{P}_i^j$ be set of users with predicted ratings for item i in j^{th} round of prediction and $\mathbb{K}_i^j$ be set of users with known ratings for item i in j^{th} round of prediction.

The set $\mathbb{U}_3^1$ contains users whose rating cannot be predicted because there is insufficient rating information available. The rating sets for first round are:

- $\mathbb{K}_3^1 = \{u_5\}$
- $\mathbb{P}_3^1 = \{u_4\}$
- $\mathbb{U}_3^1 = \{u_1, u_2, u_3, u_6, u_7\}$

Next, the rating for item u_4 becomes available and round 2 begins. First, u_4 is now in $\mathbb{K}_3^2$ and as more information is available, Cosine-similarity can predict the ratings of u_3, u_6 and u_7. The ratings sets second round are:

- $\mathbb{K}_3^2 = \{u_4, u_5\}$
- $\mathbb{P}_3^2 = \{u_3, u_6, u_7\}$
- $\mathbb{U}_3^2 = \{u_1, u_2\}$

Finally, assume that the ratings for u_3, u_6 and u_7 become known, all user ratings for i_3 become either known or predictable. The ratings sets third round are:

- $\mathbb{K}_3^3 = \{u_3, u_4, u_5, u_6, u_7\}$
- $\mathbb{P}_3^3 = \{u_1, u_2\}$
- $\mathbb{U}_3^3 = \emptyset$

In the presence of intermittent link contact capacity, the proposed approach is to predict the items that has the greatest likelihood of getting the largest amount of positive ratings from users.

5.4 System Model

The MCF algorithm runs locally on each mobile device based on available meta-data information, which consists of the user-item rating matrix and contact history.

The status of $m_{u,i}$ can be either **rated**, **predicted** or **unpredictable**. A rating $m_{u,i}$ is rated if i has be transferred to and rated by u, and the rating can be either 1 or 0. A rating $m_{u,i}$ is predicted if it has not been rated yet, but the rank $RNK_{u,i}$ (see Equation 5.2) can be computed. The predicted rating is 1 if i is among the top-k item according to $RNK_{u,i}$ for user u, and 0 otherwise. A rating $m_{u,i}$ is said to be *correct* if the predicted rating matches the user rating eventually.

Recall that there are two naive methods to pick an item to transfer to another device. The first method, considering only item recall, picks a predicted item that gives the highest rank $RNK_{u,i}$ to maximize the probability that the rating $m_{u,i}$ is correct and positive. The second method considers only the prediction coverage, and picks a predicted item such that *if* the item is rated, then the number of unpredictable items becoming predictable is maximal.

To consider both recall and coverage, we consider the following metric: for an item i, we are interested in the number of correct positive prediction for i eventually, i.e., when i has been rated by all users. Before i is rated by all users, this quantity is considered as a random variable, denoted as Ω_i. At any round t, we know the current number of correct positive rating for i, denoted r_i^+. We also know is the number of positive predictions for item i, g_i^+. Ideally, we would like the following inequality to be true:

$$\Omega_i > r_i^+ + g_i^+,$$

i.e., all the positive predictions for i are correct, and there are additional new postive ratings for i. The key question is thus to estimate the probability that the above condition is true if i is transferred.

In the following, we present approximations on the potential positive ratings for an item and the probability of delivery of items with positive ratings to the users. The goal is to derive approximations that can be used as input to guide and motivate the design of CoFiGel.

5.5 Potential Gain in Positive Ratings

First, we derive an equation to bound $Pr\{\Omega_i > g_i^+ + r_i^+\}$, the probability that the number of correct positive predictions for item i would increase if i is transferred.

$$Pr\{\Omega_i > r_i^+ + g_i^+\} \leq \min\left\{1, e^{\frac{r_i^+ E[\Omega_i]}{n - r_i^+}} \left(1 - \frac{r_i^+}{n}\right)^{r_i^+ + g_i^+}\right\} \tag{5.3}$$

Proof. Let $\Omega_{u,i}$ be the random variable that takes a value of 1 if the predicted

rating for the user u for item i is correct and positive, and a value of 0 otherwise. Let $p_{u,i}$ be the probability that the prediction for the user u on item i is correct and positive, i.e., $p_{u,i} = Pr\{\Omega_{u,i} = 1\}$.

As $\Omega_i = \sum_{u \in U} \Omega_{u,i}$, we can write $Pr\{\Omega_i > \omega\}$ as

$$Pr\{\Omega_i > \omega\} \leq e^{-c \cdot \omega} \prod_{u \in U} \left(p_{u,i} e^c + 1 - p_{u,i}\right)$$

for any $c > 0$ and $\omega > 0$ (Lemma 1 of [30]).

Let $c = \ln\left(\frac{n}{n - r_i^+}\right)$ and $\omega = r_i^+ + g_i^+$, assuming that $0 < r_i^+ < n$. We rewrite the equation as follows:

$$Pr\{\Omega_i > r_i^+ + g_i^+\} \leq$$
$$\left(1 - \frac{r_i^+}{n}\right)^{r_i^+ + g_i^+} \prod_{u \in U} \left(\frac{n p_{u,i}}{n - r_i^+} + 1 - p_{u,i}\right) \tag{5.4}$$

From Taylor series expansion, we know that

$$\frac{n p_{u,i}}{n - r_i^+} + 1 - p_{u,i} = 1 + \frac{r_i^+ p_{u,i}}{n - r_i^+} \leq e^{\frac{r_i^+ p_{u,i}}{n - r_i^+}}$$

Since

$$\prod_{u \in U} e^{\frac{r_i^+ p_{u,i}}{n - r_i^+}} = e^{\frac{r_i^+ \sum^{u \in U} p_{u,i}}{n - r_i^+}} = e^{\frac{r_i^+ E[\Omega^i]}{n - r_i^+}}$$

We have

$$Pr\{\Omega_i > r_i^+ + g_i^+\} \leq e^{\frac{r_i^+ E[\Omega^i]}{n - r_i^+}} \left(1 - \frac{r_i^+}{n}\right)^{r_i^+ + g_i^+}$$

We bound the probability to 1. □

Some points to notes:

- $E[\Omega_i]$ is unknown and needs to be estimated. In the algorithm implementation, we approximate $E[\Omega_i]$ using r_i^+. Once we substitute $E[\Omega_i]$ with r_i^+, the right hand side of Equation 5.3 can be computed using the parameters r_i^+ and g_i^+ which are available locally. Further, to bootstrap the system, we set initialize r_i^+ to a small value (we set it to 1% of all total users in our simulation).

- The concentration bound decreases as g_i^+ increases: as we made more positive predictions for item i, the probability that the predictions are all correct decreases.

- The concentration bound increases for large r_i^+ as r_i^+ increases. If an item has lots of correct positive ratings, then the chances that current positive predictions are correct increases.

5.5.1 Estimation of Delivery Probability

In the previous section, we only consider the effect of transferring an item i on the MCF algorithm, but not how long it takes. For the ratings and items to be useful, the item should reach a user before some time deadline. We estimate the probability it takes to delivery an item i late after the deadline in this section.

Consider a DTN with exponentially distributed inter-contact time and infinite node buffer capacity. Let λ be the average contact rate (number of contacts in unit time) and B be the average contact capacity (duration of contact in time units) of the system. Let H_i be the set of users having item i and N_i be the set of users that i should be delivered to.

Let Y_i be the random variable representing the time when i is delivered to *all* users in N_i and Z_i be the time when i is delivered to *any* user in N_i.

Obviously,

$$Y_i \leq \sum_{u \in N_i} Z_i \tag{5.5}$$

By Markov inequality and Equation 5.5, we can find the probability that a given item i is delivered after a given time t:

$$Pr\{Y_i \geq t\} \leq \frac{E[Y_i]}{t} \leq \frac{E[\sum_{u \in N_i} Z_i]}{t} = \frac{|N_i| E[Z_i]}{t} \tag{5.6}$$

The next step is to approximate $E[Z_i]$. An item i is delivered to a user u only if u comes into contact with a user in H_i for a long enough time for i to be transferred. Furthermore, we consider i as a useful delivery only when this occurs the first time. If we let $M_{i,u}$ be the random variable representing the time when user u meets with a user in N_i and transfers i to that user, then $Z_i = \min_{u \in H_i}\{M_{i,u}\}$.

Let μ_i be the mean of $E[M_{i,u}]$ over $u \in H_i$. i.e.,

$$\mu_i = \frac{1}{|H_i|} \sum_{u \in H_i} E[M_{i,u}]$$

From Bertsimas et al. [23], we know that the expected value of the first order statistic, Z_i, is bounded by the average of the individual means of the random variables of the order statistic:

$$E[Z_i] \leq \mu_i$$

and, therefore, from Equation 5.6:

$$Pr\{Y_i \geq t\} \leq \frac{|N_i| \mu_i}{t} \tag{5.7}$$

We now explain how we can estimate μ_i and bound $Pr\{Y_i \geq t\}$. Upon device contact, each device exchanges a matrix σ, where each element σ_{i,u_1} is the last known position (in bytes) of item i in user u_1's transfer queue for $u_1 \in H_i$. We approximate the system behaviour by assuming that scheduling is FIFO. The time user u_1 meets any user u_2 before transferring item i to u_2 is estimated with $\frac{\sigma_{i,u_1}}{B\lambda}$. μ_i is then approximated as

$$\mu_i = \frac{1}{B\lambda|H_i|} \sum_{u_1 \in H_i} \sigma_{i,u_1} \tag{5.8}$$

Putting the results together, from Equations 5.8 and 5.7, and bounding the probability to 1,

$$Pr\{Y_i \geq t\} > 1 - \min\{1, \frac{|N_i|}{B\lambda t|H_i|} \sum_{u_1 \in H_i} \sigma_{i,u_1}\} \tag{5.9}$$

The values $|N_i|$ and $|H_i|$ can be obtained from the user-item rating matrix being exchanged, and B and λ can be estimated locally by keeping track of contact history.

We note the following in Equation 5.9. The likelihood of not meeting the deadline for an item i increases if (i) $|N_i|/|H_i|$, the ratio of number of nodes not having i to having i increases, (ii) $B\lambda t$, the contact opportunity until time t, decreases, and (iii) σ_{i,u_1}, the waiting time for i in the transfer queue increases.

5.6 Algorithm

Now we can present the workings of CoFiGel based on the results derived from previous sections. At each device, CoFiGel decides which item to transmit by computing a utility $UTIL_i$, which incorporates the number of positive ratings

(rated or predicted) for i, the probability of gain in ratings, and the probability of delivery within the deadline:

$$UTIL_i = (g_i^+ + r_i^+) \cdot PR_i \cdot PD_i \tag{5.10}$$

where PR_i is the right-hand-side of Equation 5.3, and PD_i is the right-hand-side of Equation 5.9.

The utility increases if either (i) the total number of correctly predicted positive ratings we get eventually ($g_i^+ + r_i^+$), increases (ii) the likelihood of the number of correct predictions increases (PR_i), or (iii) the likelihood of delivering an item within the deadline t increases.

Note that since the bounds provided are very loose, we do not expect these computed utilities to reflect the true value of the rating gain. For the purpose of the scheduling, however, only the relatively ordering is important. Items with larger utilities are transferred first. We will shown in the evaluation that the heuristics used is indeed sufficient to provide good results.

5.7 Evaluation

We evaluated CoFiGel using real-world traces and show that substantial improvement can be achieved compared to baseline schemes that do not consider rating or contact history. We use the MovieLens data set [2] as the underlying user ratings. The data set chosen has 100K ratings, 943 users and 1682 items.

For mobility traces we have chosen to use the RollerNet trace ([97]) and the San Francisco taxi trace ([77]) or SanCab trace.

[2] http://movielens.umn.edu, retrieved August 22, 2014

The RollerNet trace is an example of human mobility and the data set consists of about 60 Bluetooth devices carried by groups of roller bladers in a roller tour over three hours. The average contact duration is 22 seconds and the average number of contact per node over 3 hours is 501.

The San Francisco taxi (SanCab) trace is representative of a vehicular DTN environment. The data set consists of GPS coordinates of about 500 taxis in the San Francisco bay area over a one month period. We selected a 6 hour interval for our simulation. By assuming a fixed communication range of 50m, we derive a communication pattern that has average contact duration of 73 seconds and the average number of contact per node over 6 hours is 213.

Size of user generated content such as those found in popular sites like YouTube is 25MB or less (98% of videos are 25MB or less [27]). We choose data size of 11MB for the taxi trace and 15MB for RollerNet. The relatively larger item size for RollerNet is due to the large number of contacts per node. The buffer size and item generation rate are similarly adjusted to ensure sufficient loading in the system.

As some nodes in the trace have very limited contacts with the rest of the trace, we avoid selecting these nodes as the publisher or subscriber nodes (though they can still act as relay nodes). These nodes are identified as nodes which do not have sufficient number of node contacts and contact bandwidth to support meaningful data exchange. After removing these nodes, 22 publisher and 56 subscribers were chosen for the Taxi trace and 10 publishers and 30 subscribers were chosen for the RollerNet trace.

In order to reduce simulation time, we reduce the MovieLens data set selected by randomly choosing 900 items (movies) and 500 users from the original data set. All user-item ratings associated with these chosen user-item pairs from the original dataset are also included.

Finally, as the rating data set and the mobility trace are generated independently, we map the rating data to the mobility trace in the following way:

1. Every item in the reduced data set is randomly assigned to a publisher node in the mobility trace. This node will act as the publisher for the item.

2. Every user in the reduced data set is randomly mapped onto a mobile subscriber node. The actual user-item rating is known only when the item reaches the given mobile node where the user is located.

Table 5.3: CoFiGel: trace based simulation parameters

Parameter	**SanCab**	**RollerNet**
Number of Publisher Nodes	22	10
Number of Subscriber Nodes	56	30
(Item publisher rate)/publisher	20 items/Hr	40 items/Hr
Simulation duration	6 Hrs	Approx.3 Hrs
Item size	11MB	15MB
Default buffer size	2GB	1GB
Default contact bandwidth	3Mbps (375KBps)	3Mbps
Item Lifetime	2 hours	1 hour 15 min
Warmup time	1 Hr	1 Hr
Cool down time	1 Hr	0.5 Hr

The settings in Table 5.3 are used as default unless otherwise specified. Each simulation point is run at least 3 times with different random seeds.

The performance objectives used are prediction coverage, precision, recall

and number of satisfied users and latency, as described in Section 5.3. We compare the performance of CoFiGel with four other algorithms, namely:

1. A scheme that knows the ground-truth of the data. The ground-truth is available from the MovieLens data set. This scheme provides the actual rating coverage and gives an upper bound on the system performance. This scheme is used only in the coverage comparison since ground-truth is not applicable in the user satisfaction evaluation.

2. An epidemic-based algorithm that is similar to CoFiGel except that it does not take into account contact history and time constraints. We called this algorithm NoDeliveryTime. The performance difference between NoDeliveryTime and CoFiGel indicates the improvement provided by exploiting contact history.

3. An algorithm that uses only the rating information available. This is referred to as NoCoverage. The ratings of the items are predicted using the MCF, but the rating update and the potential coverage increase is not considered. By using only limited rating information, NoCoverage is expected to perform the worst.

4. An algorithm which tries to schedule an item so as to acquire prediction coverage of hitherto unrated users and to satisfy as many more users as possible. This is called the NoItemRecall. While this approach also uses contact history, it does not perform multi-round predictions as in the case of CoFiGel. It only acts using the current rating information.

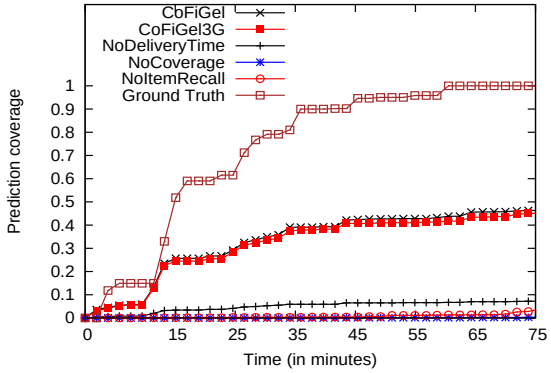
5. CoFiGel3G is a modification of CoFiGel such that it uses the cellular network to upload/download ratings and a central server to run the MCF.

However, the data are still sent over the DTN. By exploiting the cellular network as control channel, ratings information propagate quickly among the nodes and is always up-to-date. However, it is important to note that faster rating propagation does not always translate to higher rating coverage. This is because an actual rating can only be discovered after an user has access to the actual item and provides the rating.

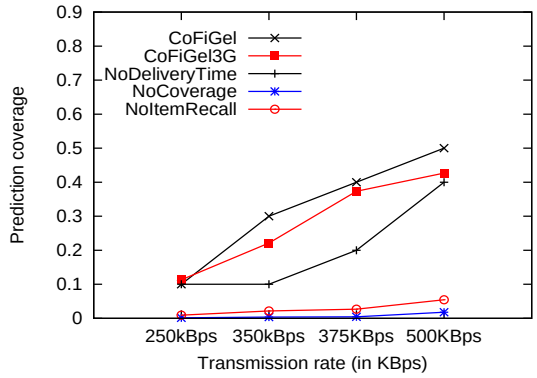
5.7.1 Coverage

In this section, we evaluate the performance of CoFiGel and the other algorithms in terms of prediction coverage, a commonly used metric for MCF. In addition, we also measured the fraction of correctly predicted positive (or FCPP) items which looks at the ratio of correctly predicted positive item to the total of number of positive ratings rather over all ratings. Given that we are simulating a DTN environment, we felt that FCPP provides a better gauge for what is achievable by good algorithms in more challenging environments. Due to space constraints, we will only show the results for prediction coverage for RollerNet (Figure 5.1(c)), but FCPP for both RollerNet (Figure 5.2(c)) and SanCab (Figure 5.3(c)).

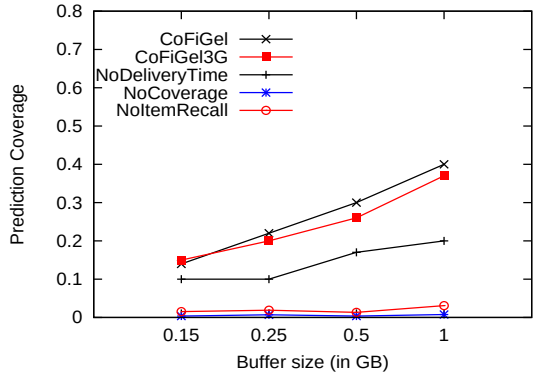
Figures 5.1(a), 5.2(a) and 5.3(a) show how positive ratings increase over time. The actual number of ratings for the items published so far (*Ground-Truth*) are shown to illustrate the best possible outcomes over time. The results show that CoFiGel and CoFiGel3G have the best performance. In the SanCab trace, CoFiGel performs even better than CoFiGel3G. In terms of overall ratings, CoFiGel discovers 45% of the ratings in the RollerNet trace. In terms of FCPP, CoFiGel discovers 84% and 74% of the positive ratings in the RollerNet and SanCab traces respectively. The result is better in the



(a) Prediction coverage over time

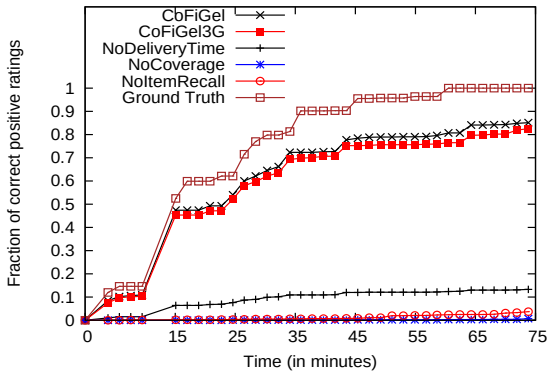


(b) Prediction coverage vs. Transmission Rate

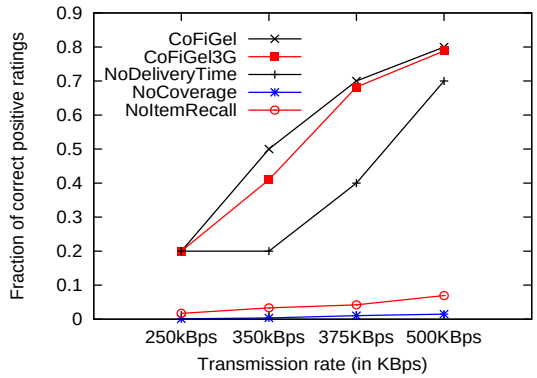


(c) Prediction coverage vs. Buffer

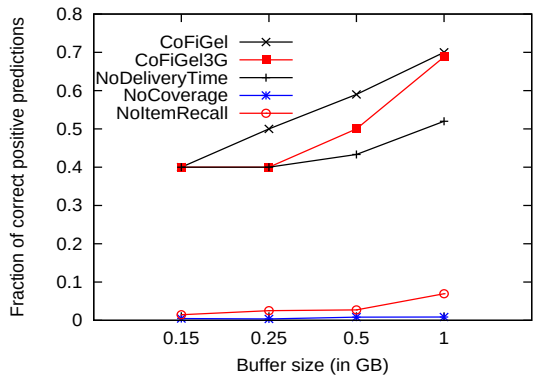
Figure 5.1: RollerNet trace (total ratings = 11536)



(a) Fraction of correct positive predictions over time

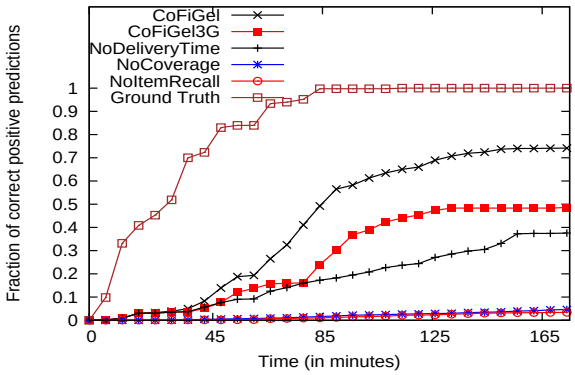


(b) Fraction of correct positive predictions vs. Transmission Rate

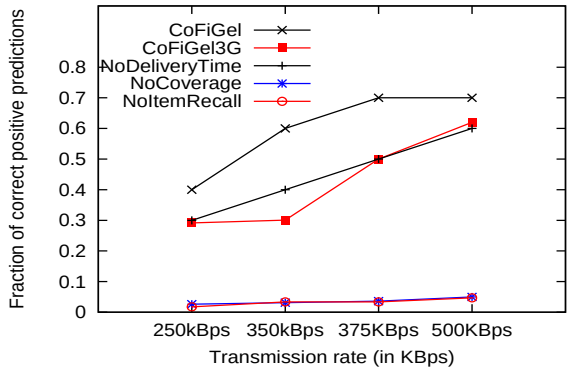


(c) Fraction of correct positive predictions vs. Buffer

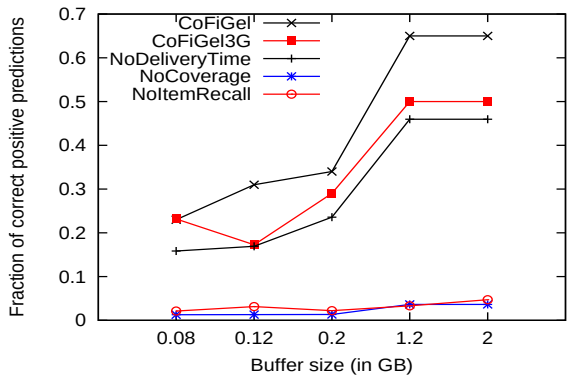
Figure 5.2: RollerNet trace (total positive ratings = 6400)



(a) Fraction of correct positive predictions over time



(b) Fraction of correct positive predictions vs. Transmission Rate



(c) Fraction of correct positive predictions vs. Buffer

Figure 5.3: SanCab trace (total positive ratings = 5400)

RollerNet case due to higher contact rate and capacity. In fact, with RollerNet, the performance of CoFiGel measured using FCPP closely matches the actual ratings in the first 15 minutes and the gap remains small throughout the simulation.

In the results shown, it can be clearly observed that CoFiGel has the best performance, followed by CoFiGel3G. This result can be somewhat surprising since CoFiGel3G uses the same algorithm as CoFiGel but uses the control (3G) channel for centralized rating computation and sharing. We explain the result as follows. Since CoFiGel3G performs centralized rating, the rating matrix gets updated much faster. This fast rating update has the (unintended) consequence that the variable G_i in Equation 5.10 approaches the value of 1.0 much faster than the case for CoFiGel. As the value of G_i gets close to 1 and saturates around this value, this variable becomes useless in term of providing information for relative ranking to decide which data item is more important. However, since propagation of data item lags behind rating data, the loss of this rating information results in CoFiGel3G performing worse than CoFiGel.

For the RollerNet trace simulation with default settings, the higher contact rate and capacity turn out to have adverse effect on NoDeliveryTime, NoCoverage and NoItemRecall, since each algorithm only looks at one aspect of the problem. In terms of FCPP, NoDeliveryTime discovers 13% of the positive ratings, while NoCoverage discovers 0.6% or less of the positive ratings and NoItemRecall discovers around 1%.

For the SanCab trace, NoDeliveryTime's performance is similar to CoFiGel in the early part of the experiment but since it does not take the deadline into account, its performance degrades with time. As expected, as NoCoverage uses only basic rating, it performs very badly. This is also the case with NoItemRecall. Overall, in terms of FCPP, for the SanCab trace, NoDeliveryTime

discovers 38% of the ratings, while NoCoverage and NoItemRecall discover 4% or less.

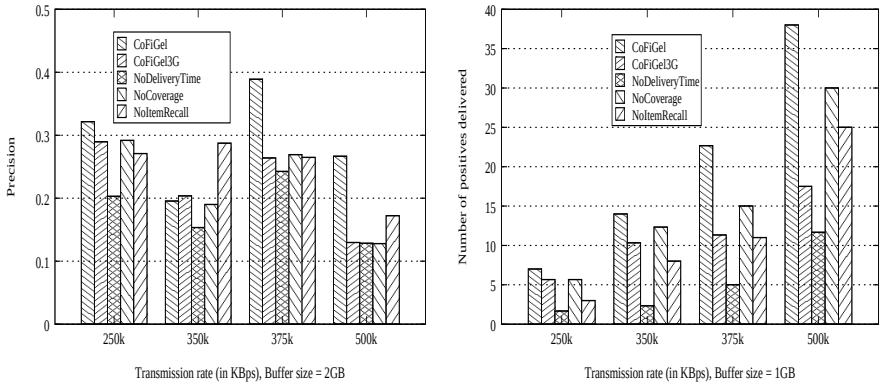
For both traces, the coverage for the NoCoverage is very low, showing that it is important to take into account additional information beyond ratings. Figures 5.1(b), 5.2(b) and 5.3(b) show how coverage varies with transmission rate. While increase in contact capacity results in increased coverage because more items get rated, CoFiGel is able to exploit the increase in transmission rate much better than NoDeliveryTime, NoCoverage and NoItemRecall. For the SanCab trace, in terms of FCPP, relative to CoFiGel, NoDeliveryTime discovers 17% to 46% less ratings while NoCoverage and NoItemRecall discover less than 5% of the ratings consistently.

Similar behaviour is observed in the RollerNet trace. In the results shown, CoFiGel performs better than NoDeliveryTime by up to 105% and discovers at least 50 times more ratings than NoCoverage and NoItemRecall consistently. In general, more improvement comes from taking into account rating coverage gain (from NoCoverage to NoDeliveryTime) than taking into account contact history. The effort by NoItemRecall to increase the number of user ratings is also ineffective due to the absence of rating gain which is capitalized by CoFiGel. Nevertheless, substantial improvement is still observed between NoDeliveryTime and CoFiGel.

The performance with respect to different buffer sizes is shown in Figures 5.1(c), 5.2(c) and 5.3(c). For the SanCab trace, buffer size is varied from $80MB$ to $2GB$. The performance of CoFiGel is better than NoDeliveryTime by 41% to 83%. Improvement of CoFiGel over NoCoverage is about 17 times. For the RollerNet trace, there are two observations. First, for very small buffer size of less than 150MB, very few items make it to the next hop and hence, the FCPP remains same for CoFiGel and NoDeliveryTime. FCPP of CoFiGel

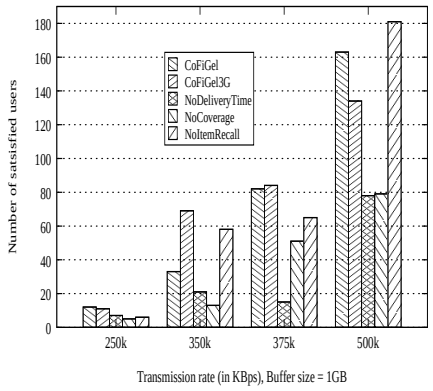
is higher than NoDeliveryTime by up to 36% and for NoCoverage by 50 to 60 times.

5.7.2 User Satisfaction



(a) Transmission Rate Vs Precision

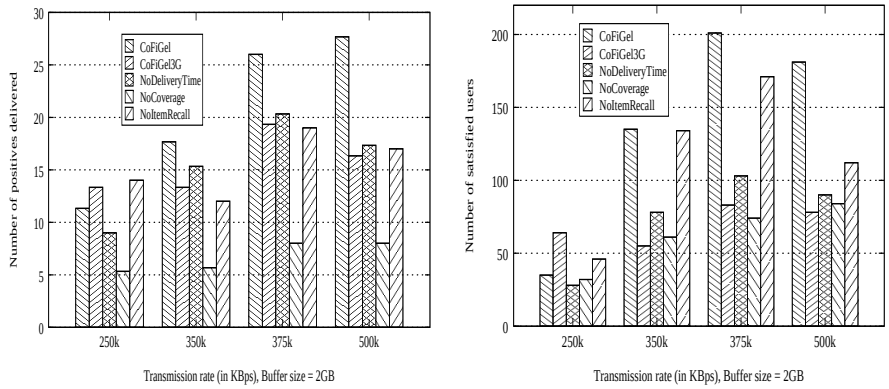
(b) Transmission Rate Vs Delivered positive items



(c) Transmission Rate Vs Users

Figure 5.4: RollerNet trace

While coverage indicates the predictive power of the system, the actual user satisfaction has to be measured by looking at how many items reach users that like them. In order to ensure that the nodes have accumulated enough training data before making the measurement, for the SanCab trace,



(a) Transmission Rate Vs Delivered positive items

(b) Transmission Rate Vs Users

Figure 5.5: SanCab trace

we consider items generated after the first and before the fifth hour. The first hour serves as the training phase, while the last hour is ignored to make sure that items generated later in the trace do not bias the measurement. Similarly, the training phase for RollerNet is 1.5 hours and items generated during last half hour of trace are ignored.

Figure 5.4(a) shows the results for precision of items reaching the users. It is clear that CoFiGel performs very well, except for one case (350K), it has the highest precision. In addition, note that even though NoItemRecall has a higher precision, from the results in the previous section, it has very low coverage. Due to the disconnected nature of DTN and the large number of data items and users available, it is also useful to look user utility in two other ways.

First, we look at the average number of positively rated items that reach any user. The result is shown in Figures 5.4(b) and 5.5(a). CoFiGel clearly outperforms the other two algorithms by a very large margin once the bandwidth exceeds some threshold required for data dissemination. For the SanCab

trace, at the highest transmission rate experimented, CoFiGel delivers up to 59% more useful items than NoDeliveryTime and 245% more useful items than NoCoverage. In the case of the RollerNet trace, the improvements are 117% and 225% respectively.

Another way we measure recall is to look at the number of users who have received at least one useful item. The result is shown in Figures 5.4(c) and 5.5(b). Again, CoFiGel performs well, in particular, at higher bandwidth. At 4Mbps, CoFiGel delivers twice as many useful items to users than NoCoverage and NoDeliveryTime for both traces. At 2Mbps, both CoFiGel3G and NoItemRecall outperform CoFiGel in the SanCab trace. The much more sparse contact interval and lower contact bandwidth may have help to slow down rating matrix propagation enough such that CoFiGel3G is able to fully exploit the benefit of fasting rating propagation without value saturation.

5.8 Summary

We have presented *CoFiGel*, a module of *UGCSelect* that combines collaborative filtering and m2m infrastructure-less network dissemination so as to enable content distribution in a distributed environment with intermittent connectivity. It is designed for sharing of locally stored contents that have *spatial and temporal relationships*. *CoFiGel* has two key components. First, it estimates the potential gain in prediction coverage if an item is scheduled. Second, it estimates the time needed to deliver the ratings. Our analysis allows us to derive approximations that are used as input to the utility computation in *CoFiGel* based on only locally estimated parameters.

We have evaluated CoFiGel through trace-based simulation using DTN mobility traces (RollerNet trace [97] and San Francisco taxi trace [77]) and an

user rating data set from MovieLens [37]. The RollerNet trace is an example of human mobility and the San Francisco taxi trace is an example of vehicular mobility. Our evaluation shows that CoFiGel can provide 80% more prediction coverage in comparison to the baseline algorithms, detecting at least 74% of positive ratings in the process, and delivers at least 59% more positive (liked by user) items in comparison to the baseline algorithms that do not take into account either ratings or contact history.

Chapter 6

Conclusion

This chapter summarizes the key contributions of this work, the insights gained in the process, open issues and future directions for potential research.

6.1 Contributions

We have discussed the importance of UGC authored by smartphone users and their impact on the limited bandwidth and battery characteristics of smartphones. The characteristics of UGC and their in-situ value requires a resource and content aware management.

Existing literature on the resource management, use of sensors and personalization to process content have been discussed in detail. We also go into the details of the limitations of each category of work in Chapter 2. In the same chapter, we also identify the important factors, namely: **(1)** network infrastructure resource awareness **(2)** content redundancy awareness **(3)** multi-source selection **(4)** infrastructure independence. We then present our system *UGCSelect* which uses subjective and objective features of content and the network state to select UGC while conserving the resources of the partic-

ipating nodes. As part of this system, there content selection methods have been enunciated. They are:

1. **Movisode** or **MO**bile **VI**deo **S**haring [87] **O**n-**DE**mand for indexing and on demand retrieval of crowd sourced mobile video

2. **AutoLink** or **Auto**matic image content **Link**ing for Automatic content based linking of sensor-annotated smartphone images

3. **CoFiGel** or **Co**llaborative **Fi**ltering **Gel** for mobile-to-mobile recommendation [86]

6.1.1 Movisode

Movisode provides spatio-temporal coverage while minimizing the upload cost of the system. It uses sensor cues available in smart phones today to achieve the above goal. We have evaluated Movisode through trace-based simulation driven by real-world dataset and energy traces, test bed evaluation and user study. Results indicate that CCMVA algorithm which forms part of the Movisode balances the trade-off between spatio-temporal coverage and energy much better than the other candidate algorithms and provides results which are close the best available videos.

6.1.2 AutoLink

AutoLink organizes unstructured photo collections from ad-hoc events so that users could discover content easily. AutoLink combines image content features and smartphone sensors to classify it based on scenes and estimate the region of the scene captured by the image. This information is used to automatically generate an image hierarchy. The image hierarchy helps users navigate

to a scene, and explore the regions of the scene in higher detail. We have demonstrated its performance with two real world datasets.

6.1.3 CoFiGel

CoFiGel combines collaborative filtering and mobile-to-mobile DTN routing so as to enable content distribution in a distributed environment with intermittent connectivity. It is designed for sharing of locally stored contents that have spatial and temporal relationships. CoFiGel has two key components. First, it estimates the potential gain in prediction coverage if an item is scheduled. Second, it estimates the time needed to deliver the ratings. Our analysis allows us to derive approximations that are used as input to the utility computation in CoFiGel based on only locally estimated parameters.

Simulation results show that CoFiGel performs well. It is able to discover substantially more ratings and is able to deliver much more items that are rated positively by the users than baseline algorithms that do not take into account rating gain or mobility pattern.

6.2 Insights

Following are the insights gained during the design and development of the system.

1. **Factoring in source resources during UGC selection saves energy and bandwidth:** Individual device context varies from one device to another. Each device might experience a different effective bandwidth, might have content with different sizes although belonging to the scene. By factoring in these characteristics, our system chooses

only content with least cost. This contributes to conserving bandwidth and saving energy.

2. **Redundancy management helps:** Redundancy increases the choice for UGC selection, but also contributes to resource cost. Redundancy is also not even. Some regions of the scene and some views could be captured more often than others. Some moments of the event might have more redundant content than others due to popularity of the moment. This uneven distribution of redundancy has to be factored in during UGC selection to ensure availability of content to user.

3. **Combining light-weight content processing with sensor data helps reduce unnecessary content upload:** Smartphones are increasingly becoming powerful computation devices. New smartphone models are equipped with multi-core CPU and GPU units. This could be leveraged to perform some degree of content processing on the phone itself and combine this with sensor cues to index the content. Such indexing helps avoid uploading content unnecessarily. This approach also reduces the need for training based content processing.

4. **Scene relative content organization:** Content organization does not necessarily require absolute positioning of the camera. Organizing and locating the content based on scenes makes the system usable in indoor and outdoor environments without infrastructure.

6.3 Future work

Due to the advent of new sensors on the smartphone platform, mobile vision has presents new areas for exploration. Some of these have been listed in the

context of our work.

- **Improving resource-aware UGC selection.** There are some issues in this work which offer scope for improvement:

    1. **Accuracy of sensors:** Smartphone sensors are susceptible to noise. We do not offer any method to correct the sensor data, but rely on the content to offset any errors in the accuracy. But sensor noise could affect system performance. For instance, compass sensor could be influenced by ambient magnetic field and requires calibration. Accelerometer sensor could trigger false positives due to sudden movements. Sensor fusion has been used to correct these errors. Recent versions of android have introduced sensor fusion as part of the API. Using this could improve the accuracy of Movisode and AutoLink.
    2. **Use of reference images:** Movisode assumed that a reference image for the scene is provided as part of the input. This limitation could be overcome if the reference image is automatically selected from the video collection.

- **Content selection with smartphone depth sensors.** Recently, depth sensors for smartphones have been unveiled for android/ iPhone [11]. Content selection could benefit from depth sensors in many ways. To begin with, Movisode and AutoLink accuracy could be improved by using depth data. Occlusion detection, region of interest identification along depth, selection and streaming of 3D-video from smartphones over limited bandwidth links, use of multiple interfaces on smartphones to load balance 3D content streaming, energy profiling of depth sensor and man-

aging battery life when depth sensor is used are some interesting avenues to explore.

- **Extending video selection for multiple scenes.** Movisode dues POI estimation for one scene, while AutoLink can classify images for multiple scenes. These two functionalities could be merged so that videos segments could be classified based on scenes. This could facilitate automatic linking of video segments based on scene neighboured. Such a mechanism could help user to navigate between video segments from different scenes using these links. Such links could also help annotate scene content.

- **Using AutoLink to generate time-lapse videos and slideshows.** Time-lapse videos and slideshows help browse through and summarize photo collections. They also have a aesthetic value which could make public photo collections popular. Image hierarchy generated using AutoLink could be used to create time-lapse videos and slideshows automatically.

Bibliography

[1] BrightCove. https://www.brightcove.com/en/online-video-platform/analytics.

[2] CrowdOptics. http://www.crowdoptic.com.

[3] Harnessing the Power of Flickr. http://photo.net/column/harolddavis/finding-an-audience-for-your-photography/using-flickr-for-marketing/?

[4] Micro Cool. http://vku.sdo.com/.

[5] Mobile Image Webs. http://www.stanford.edu/ zxwang/miw/.

[6] OpenCV. http://opencv.org/.

[7] Out Listen. http://www.outlisten.com/.

[8] O2 germany admits network meltdown; smartphones blamed. http://www.fiercewireless.com/europe/story/o2-germany-admits-network-meltdown-smartphones-blamed/2011-11-23, 2011.

[9] KPCB 2013 Internet Trends. http://www.kpcb.com/insights/2013-internet-trends, May 2013.

[10] Mobile and wi-fi overloaded by 2020, report claims. http://www.telegraph.co.uk/technology/broadband/10480457/Mobile-and-Wi-Fi-overloaded-by-2020-report-claims.html, 2013.

[11] Structure Sensor. http://structure.io/, 2013.

[12] A. Abdel-Hakim and A. Farag. CSIFT: A SIFT descriptor with color invariant characteristics. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1978–1983, June 2006.

[13] M. Alivand and H. Hochmair. Extracting scenic routes from vgi data sources. In *Proceedings of the Second ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information*, GEOCROWD '13, pages 23–30, 2013.

[14] E. Altman and J.-c. Bermond. Distributed storage management of evolving files in delay tolerant ad hoc networks. In *INFOCOM '09*, pages 1431–1439, Rio de Janeiro, Brazil, 2009.

[15] G. Ananthanarayanan, V. N. Padmanabhan, L. Ravindranath, and C. A. Thekkath. Combine: Leveraging the power of wireless peers through collaborative downloading. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, pages 286–298, 2007.

[16] A. Arpa, L. Ballan, R. Sukthankar, G. Taubin, M. Pollefeys, and R. Raskar. Crowdcam: Instantaneous navigation of crowd images using angled graph. In *3DV (3DIM/3DPVT)*, Seattle, June 2013.

[17] S. A. Ay, R. Zimmermann, and S. Kim. Relevance ranking in georeferenced video search. *Multimedia Systems*, 16:105–125, 2010.

[18] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN routing as a resource allocation problem. In *SIGCOMM '07*, pages 373–384, Kyoto, Japan, 2007.

[19] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 209–222, 2010.

[20] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 280–293, 2009.

[21] X. Bao, S. Fan, A. Varshavsky, K. Li, and R. Roy Choudhury. Your reactions suggest you liked the movie: Automatic content rating via reaction sensing. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 197–206, 2013.

[22] X. Bao and R. Roy Choudhury. MoVi: Mobile Phone Based Video Highlights via Collaborative Sensing. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 357–370, 2010.

[23] D. Bertsimas, K. Natarajan, and C.-P. Teo. Tight bounds on expected order statistics. *Probabability in the Engineering and INformational Sciences*, 20:667–686, October 2006.

[24] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, June 2008.

[25] E. Brachmann, M. Spehr, and S. Gumhold. Feature propagation on image webs for enhanced image retrieval. In *Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval*, ICMR '13, pages 25–32, 2013.

[26] Y.-C. Chen, Y.-s. Lim, R. J. Gibbens, E. M. Nahum, R. Khalili, and D. Towsley. A measurement-based study of multipath tcp performance over wireless networks. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 455–468, 2013.

[27] X. Cheng, C. Dale, and J. Liu. Statistics and social network of youtube videos. In *IWQoS'08*, pages 229–238, 2008.

[28] M. L. Cooper. Clustering geo-tagged photo collections using dynamic programming. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 1025–1028, 2011.

[29] S. Deng and H. Balakrishnan. Traffic-aware techniques to reduce 3g/lte wireless energy consumption. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pages 181–192, New York, NY, USA, 2012. ACM.

[30] J. Diaz, J. Petit, and M. Serna. A guide to concentration bounds. In S. Rajasekaran, P. Pardalos, J. Reif, and J. Rolim, editors, *Handbook on Randomized Computing II*, volume 9 of *Combinatorial Optimization*, pages 457–507. Springer, 2001.

[31] F. R. Dogar, P. Steenkiste, and K. Papagiannaki. Catnap: Exploiting high bandwidth wireless interfaces to save energy for mobile devices. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 107–122, 2010.

[32] B. Donohoo, C. Ohlsen, S. Pasricha, and C. Anderson. Exploiting spatiotemporal and device contexts for energy-efficient mobile embedded systems. In *Proceedings of the 49th Annual Design Automation Conference*, DAC '12, pages 1278–1283, 2012.

[33] J. Erman and K. Ramakrishnan. Understanding the Super-sized Traffic of the Super Bowl. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 353–360, 2013.

[34] P.-E. Forssen. Maximally stable colour regions for recognition and matching. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[35] J. Gozali, M.-Y. Kan, and H. Sundaram. Hidden markov model for event photo stream segmentation. In *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on*, pages 25–30, July 2012.

[36] S. Greenhill and S. Venkatesh. Distributed query processing for mobile surveillance. In *Proceedings of the 15th International Conference on Multimedia*, MM '07, pages 413–422, 2007.

[37] GroupLens Research. Movielens dataset, 2006.

[38] H. Han, Y. Liu, G. Shen, Y. Zhang, and Q. Li. Dozyap: Power-efficient wi-fi tethering. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 421–434, 2012.

[39] J. Hao, S. H. Kim, S. A. Ay, and R. Zimmermann. Energy-efficient Mobile Video Management Using Smartphones. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 11–22, 2011.

[40] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[41] K. Heath, N. Gelfand, M. Ovsjanikov, M. Aanjaneya, and L. Guibas. Image webs: Computing and exploiting connectivity in image collections. CVPR '10, 2010.

[42] M. A. Hoque, M. Siekkinen, and J. K. Nurminen. Using crowd-sourced viewing statistics to save energy in wireless video streaming. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, MobiCom '13, pages 377–388, 2013.

[43] L. Horn. The iPhone 4 is officially the most popular camera on Flickr. published 22 June 2011, retreived 28 July 2011.

[44] L. Hu, J.-Y. Le Boudec, and M. Vojnovic. Optimal channel choice for collaborative ad-hoc dissemination. In *INFOCOM'10*, pages 614–622, San Diego, CA, 2010.

[45] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4g lte networks. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 225–238, 2012.

[46] J. Huang, F. Qian, Z. M. Mao, S. Sen, and O. Spatscheck. Screen-off traffic characterization and optimization in 3g/4g networks. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 357–364, 2012.

[47] S. Ioannidis, A. Chaintreau, and L. MassouliÈ. Optimal and scalable distribution of content updates over a mobile social network. In *INFOCOM '09*, pages 1422–1430, Rio de Janeiro, Brazil, 2009.

[48] P. Jain, J. Manweiler, A. Acharya, and K. Beaty. FOCUS: Clustering Crowdsourced Videos by Line-of-sight. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys '13, pages 8:1–8:14, 2013.

[49] Y. Jiang, X. Xu, P. Terlecky, T. Abdelzaher, A. Bar-Noy, and R. Govindan. MediaScope: Selective On-demand Media Retrieval from Mobile Devices. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks*, IPSN '13, pages 289–300, 2013.

[50] C.-W. C. K. C.-J. Lin and C.-F. Chou. Preference-aware content dissemination in opportunistic mobile social networks. In *IEEE INFOCOM, 2012*, 2012.

[51] Y. Kalantidis, G. Tolias, Y. Avrithis, M. Phinikettos, E. Spyrou, P. Mylonas, and S. Kollias. VIRaL: Visual Image Retrieval and Localization. *Multimedia Tools Appl.*, 51(2):555–592, Jan. 2011.

[52] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM '01*, pages 247–254, Atlanta, GA, 2001.

[53] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, and A. Markopoulou. Microcast: cooperative video streaming on smartphones. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 57–70, 2012.

[54] W. Khan, Y. Xiang, M. Aalsalem, and Q. Arshad. Mobile phone sensing systems: A survey. *Communications Surveys Tutorials, IEEE*, 15(1):402–427, April 2013.

[55] S. H. Kim, Y. Lu, G. Constantinou, C. Shahabi, G. Wang, and R. Zimmermann. MediaQ: Mobile Multimedia Management System. In *ACM Multimedia Systems*, 2014.

[56] A. Krifa, C. Baraka, and T. Spyropoulos. Optimal buffer management policies for delay tolerant networks. In *SECON '08*, pages 260–268, San Francisco, CA, 2008.

[57] Y. A. Lacerda, R. G. F. Feitosa, G. A. R. M. Esmeraldo, C. d. S. Baptista, and L. B. Marinho. Compass clustering: A new clustering method for detection of points of interest using personal collections of georeferenced and oriented photographs. In *Proceedings of the 18th Brazilian Symposium on Multimedia and the Web*, WebMedia '12, pages 281–288, 2012.

[58] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell. A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150, Sept 2010.

[59] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile data offloading: How much can wifi deliver? *IEEE/ACM Trans. Netw.*, 21(2):536–550, Apr. 2013.

[60] K. Lee, Y. Yi, J. Jeong, H. Won, I. Rhee, and S. Chong. Max-contribution: On optimal resource allocation in delay tolerant networks. In *INFOCOM '10*, pages 1136–1144, San Diego, CA, 2010.

[61] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *SDM'05*, Newport Beach, CA, 2005.

[62] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2548–2555, 2011.

[63] X. Li, H. Huang, X. Yu, W. Shu, M. Li, and M.-Y. Wu. A new paradigm for urban surveillance with vehicular sensor networks. *Computer Communications*, 34(10):1159 – 1168, 2011.

[64] Y. Li, M. Qian, D. Jin, L. Su, and L. Zeng. Adaptive optimal buffer management policies for realistic DTN. In *GLOBECOM'09*, pages 1–5, Honolulu, HI, 2009.

[65] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, Jan/Feb 2003.

[66] H. Liu, T. Mei, J. Luo, H. Li, and S. Li. Finding Perfect Rendezvous on the Go: Accurate Mobile Visual Localization and Its Applications to Routing. In *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, pages 9–18, 2012.

[67] H. Liu, Y. Zhang, and Y. Zhou. Tailtheft: Leveraging the wasted time for saving energy in cellular communications. In *Proceedings of the Sixth International Workshop on MobiArch*, MobiArch '11, pages 31–36, 2011.

[68] J. Liu, Z. Huang, L. Chen, H. T. Shen, and Z. Yan. Discovering areas of interest with geo-tagged images and check-ins. In *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, pages 589–598, 2012.

[69] X. Liu, M. Corner, and P. Shenoy. SEVA: Sensor-enhanced video annotation. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(3):24:1–24:26, Aug. 2009.

[70] Y. Liu, F. Li, L. Guo, Y. Guo, and S. Chen. Bluestreaming: Towards power-efficient internet p2p streaming to mobile devices. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 193–202, 2011.

[71] G. Lo Giusto, A. J. Mashhadi, and L. Capra. Folksonomy-based reasoning for content dissemination in mobile settings. In *CHANTS '10*, Chicago, IL, 2010.

[72] J. Luo, D. Joshi, J. Yu, and A. Gallagher. Geotagging in multimedia and computer visiona survey. *Multimedia Tools and Applications*, 51(1):187–211, 2011.

[73] J. G. Manweiler, P. Jain, and R. Roy Choudhury. Satellites in our pockets: An object positioning system using smartphones. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 211–224, 2012.

[74] B. N. Miller, J. A. Konstan, and J. Riedl. PocketLens: Toward a personal recommender system. *ACM Transactions on Information Systems*, 22(3):437–476, July 2004.

[75] R. Miyano, T. Inoue, T. Minagawa, Y. Uematsu, and H. Saito. Camera pose estimation of a smartphone at a field without interest points. In *Computer Vision - ACCV 2012 Workshops*, volume 7729 of *Lecture Notes in Computer Science*, pages 545–555. Springer Berlin Heidelberg, 2013.

[76] J.-S. Park and R. Jain. Identification of scene locations from geotagged images. *ACM Trans. Multimedia Comput. Commun. Appl.*, 9(1):5:1–5:23, Feb. 2013.

[77] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24), Feb. 2009.

[78] F. Qian, S. Sen, and O. Spatscheck. Silent tcp connection closure for cellular networks. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 211–216, 2013.

[79] C. Qin, X. Bao, R. Roy Choudhury, and S. Nelakuditi. TagSense: A Smartphone-based Approach to Automatic Image Tagging. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 1–14, 2011.

[80] D. R. Radev, H. Qi, H. Wu, and W. Fan. Evaluating web-based question answering systems. In *LREC*, 2002.

[81] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley. How hard can it be? designing and implementing a deployable multipath tcp. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 29–29, 2012.

[82] J. Reich and A. Chaintreau. The age of impatience: Optimal replication schemes for opportunistic networks. In *CoNEXT'09*, pages 85–96, Rome, Italy, 2009.

[83] C. Rossi, N. Vallina-Rodriguez, V. Erramilli, Y. Grunenberger, L. Gyarmati, N. Laoutaris, R. Stanojevic, K. Papagiannaki, and P. Rodriguez. 3gol: Power-boosting adsl using 3g onloading. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 187–198, 2013.

[84] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571, Nov 2011.

[85] M. Saini, S. P. Venkatagiri, W. T. Ooi, and M. C. Chan. The Jiku Mobile Video Dataset. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys'13, pages 108–113, 2013.

[86] P. Seshadri, M. Chan, and W. Ooi. Mobile-to-Mobile Video Recommendation. 120:13–24, 2013.

[87] P. Seshadri, M. Chan, W. Ooi, and J. Chiam. On Demand Retrieval of Crowdsourced Mobile Video. *IEEE Sensors Journal*, PP(99), 2014.

[88] A. Sharma, V. Navda, R. Ramjee, V. N. Padmanabhan, and E. M. Belding. Cool-tether: Energy efficient on-the-fly wifi hot-spots using mobile phones. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 109–120, 2009.

[89] Z. Shen, S. Arslan Ay, S. H. Kim, and R. Zimmermann. Automatic Tag Generation and Ranking for Sensor-rich Outdoor Videos. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 93–102, 2011.

[90] M. Siegler. Eric schmidt: Every 2 days we create as much information as we did up to 2003. published Wednesday, August 4th, 2010.

[91] P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, and M. Satyanarayanan. Scalable Crowd-sourcing of Video from Mobile Devices. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 139–152, 2013.

[92] N. Snavely, S. M. Seitz, and R. Szeliski. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. Graph.*, 2006.

[93] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *International Journal of Computer Vision*, 80(2):189–210, Nov. 2008.

[94] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009, January 2009. Article ID 421425, 19 Pages, doi:10.1155/2009/421425.

[95] S. Tang, X.-Y. Li, H. Zhang, J. Han, G. Dai, C. Wang, and X. Shen. TelosCAM: Identifying Burglar Through Networked Sensor-Camera Mates with Privacy Protection. In *Proceedings of the 2011 IEEE 32Nd Real-Time Systems Symposium*, RTSS '11, pages 327–336, 2011.

[96] B. Thomee. Localization of points of interest from georeferenced and oriented photographs. In *Proceedings of the 2Nd ACM International Workshop on Geotagging and Its Applications in Multimedia*, GeoMM '13, pages 19–24, 2013.

[97] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Dias de Amorim, and J. Whitbeck. The Accordion Phenomenon: Analysis, Characterization, and Impact on DTN Routing. In *IEEE INFOCOM 2009*.

[98] M. Y. S. Uddin, H. Wang, F. Saremi, G.-J. Qi, T. Abdelzaher, and T. Huang. PhotoNet: A Similarity-Aware Picture Delivery Service for Situation Awareness. In *Proceedings of the 2011 IEEE 32Nd Real-Time Systems Symposium*, RTSS '11, pages 317–326, 2011.

[99] N. Vallina-Rodriguez, V. Erramilli, Y. Grunenberger, L. Gyarmati, N. Laoutaris, R. Stanojevic, and K. Papagiannaki. When david helps goliath: The case for 3g onloading. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 85–90, 2012.

[100] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. Seeds: superpixels extracted via energy-driven sampling. In *Proceedings of the 12th European conference on Computer Vision - Volume Part VII*, ECCV'12, 2012.

[101] H. Wang, X. Bao, R. R. Choudhury, and S. Nelakuditi. InSight: Recognizing Humans Without Face Recognition. In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, HotMobile '13, pages 7:1–7:6, 2013.

[102] J. Wang, J. Pouwelse, R. L. Lagendijk, and M. J. T. Reinders. Distributed collaborative filtering for peer-to-peer file sharing systems. In *SAC '06*, pages 1026–1030, Dijon, France, 2006.

[103] U. Weinsberg, Q. Li, N. Taft, A. Balachandran, V. Sekar, G. Iannaccone, and S. Seshan. CARE: Content Aware Redundancy Elimination for Challenged Networks. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 127–132, 2012.

[104] C. Wu. Towards linear-time incremental structure from motion. In *3DV-Conference, 2013 International Conference on*, pages 127–134, June 2013.

[105] C. Wu, S. Agarwal, B. Curless, and S. Seitz. Multicore bundle adjustment. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3057–3064, June 2011.

[106] T. Yan, V. Kumar, and D. Ganesan. CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 77–90, 2010.

[107] S. Yoon, H. Oh, D. Lee, and S. Oh. Virtual Lock: A Smartphone Application for Personal Surveillance Using Camera Sensor Networks. In *Proceedings of the IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications*, RTCSA '11, pages 77–82, 2011.

[108] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, CODES/ISSS '10, pages 105–114, 2010.

[109] L. Zhang and L. van der Maaten. Structure preserving object tracking. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 1838–1845, 2013.

[110] Y. Zhang, G. Wang, B. Seo, and R. Zimmermann. Multi-video Summary and Skim Generation of Sensor-rich Videos in Geo-space. In *Proceedings of the 3rd Multimedia Systems Conference*, MMSys '12, pages 53–64, 2012.

[111] Y.-T. Zheng, S. Yan, Z.-J. Zha, Y. Li, X. Zhou, T.-S. Chua, and R. Jain. GPSView: A scenic driving route planner. *ACM Trans. Multimedia Comput. Commun. Appl.*, 9(1):3:1–3:18, Feb. 2013.

[112] C. Zhu, K. Li, Q. Lv, L. Shang, and R. P. Dick. iScope: Personalized Multi-modality Image Search for Mobile Devices. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, MobiSys '09, pages 277–290, 2009.