M.Sc. Engg. Thesis

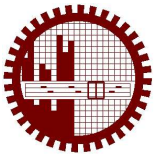
On Demand Media Content Retrieval for Distant Healthcare through mHealth

by
Sadia Shamma

Submitted to

Department of Computer Science and Engineering

in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science and Engineering



Bangladesh University of Engineering and Technology (BUET)
Dhaka 1000

September 2016

The thesis titled "**On Demand Media Content Retrieval for Distant Healthcare through mHealth**," submitted by Sadia Shamma, Roll No. 0413052074P, Session April 2013, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to its style and contents. Examination held on September 20, 2016.

Board of Examiners

1. _____

Dr. Md. Yusuf Sarwar Uddin — Chairman (Supervisor)
Associate Professor
Department of CSE
BUET, Dhaka 1000

2. _____

Dr. M Sohel Rahman — Member (Ex-officio)
Professor & Head
Department of CSE
BUET, Dhaka 1000

3. _____

Dr. Md. Shohrab Hossain — Member
Associate Professor
Department of CSE
BUET, Dhaka 1000

4. _____

Dr. Rifat Shahriyar — Member
Assistant Professor
Department of CSE
BUET, Dhaka 1000

5. _____

Dr. Nova Ahmed — Member (External)
Associate Professor
Dept of Electrical and Computer Engineering
North South University, Dhaka

Candidate's Declaration

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

Sadia Shamma

Candidate

Contents

Acknowledgements **viii**

1 Introduction **1**

1.1 Distant Healthcare System 1

1.2 Healthcare through Mobile Devices 2

1.3 Media Content in Distant Healthcare 2

1.4 Content Retrieval in mHealth Platforms 3

1.5 Contribution of the Thesis 4

1.6 Organization 5

2 Background and Related Work **6**

2.1 Popular mHealth Apps in App-stores 6

2.2 Existing Distant Healthcare Services 10

2.3 Remote Health Monitoring System 11

2.4 Uploading Service and Data Sharing 12

2.5 On-demand Content Retrieval 13

2.6 Prior Upload Content Selection 13

2.7 Scheduling Algorithm for Limited Budget 14

3 On Demand Media Content Retrieval for mHealth 15

3.1 Media Content Collection 15

3.2 Selective Content Upload 17

3.2.1 Formulation of the Upload Scheduling 18

3.2.2 Calculating Utility Values 22

3.2.3 Probability Values Computation 24

4 Implementation of Dursheba 27

4.1 Service Architecture and Components 27

4.1.1 Data Acquisition Module 28

4.1.2 Data Sharing Module 28

4.2 Handling Media Content 29

4.2.1 Content Selection Module for Prior Uploading 30

4.2.2 On Demand Content Retrieval 31

5 Experiments and Evaluation 33

5.1 Platform Setup and Generation of Simulated Data 33

5.2 Performance Metric 36

5.3 Performance Evaluation 38

5.3.1 Accuracy of different classifier 38

5.3.2 Effect of Upload Scheme 39

5.3.3 Effect of *Data Budget* 41

5.3.4 Effect of Penalty Factor or Upload Threshold 43

5.3.5 Effect of Upload Speed 44

5.3.6 Effect of Content Consumption Rate by Doctor 46

5.4 Result Summary 46

5.5 Results from Implementation 48

6 Conclusion and Future Work **51**

List of Figures

1.1 Remote health monitoring system 4

3.1 Selective media content upload 17

3.2 Logistic or sigmoid function 25

4.1 Dursheba App in Google Play 28

4.2 Screenshot from the Developed App 29

4.3 Feedback or treatment 30

4.4 Media data sharing 30

4.5 Flowchart for incremental learning 31

4.6 Device registration and on demand content retrieval. 32

5.1 Fraction of used content and selectively uploaded content 40

5.2 Effect of Upload Scheme 41

5.3 Data Status over Time 42

5.4 Effect of Data Budget 43

5.5 Effect of Uploading threshold 44

5.6 Effect of Data Uploading speed 45

5.7 Effect of Content Consumption rate by doctor 46

5.8 Uploading time vs Media file size 50

List of Tables

2.1 Overview of popular health Application 9

2.2 Overview of Distant Health care Services 10

5.1 Simulation Data 36

5.2 Comparison of classifier 39

5.3 Count 39

5.4 Comparison among upload schemes 47

5.5 Result Summary 47

5.6 Data collected from Android Application 48

5.7 Uploading time 49

Acknowledgments

All praises due to Allah, the most benevolent and merciful.

Foremost, I would like to express my heart-felt gratitude to my supervisor, Dr. Md. Yusuf Sarwar Uddin for his patience, motivation, enthusiasm, constant supervision of this work. He helped me a lot in every aspect of this work and guided me with proper directions. His warm encouragement, thoughtful guidance, critical comments have made this thesis a success.

I would also want to thank the members of my thesis committee for their valuable suggestions. I thank Professor Dr. M Sohel Rahman, Dr. Md. Shohrab Hossain, Dr. Rifat Shahriyar and specially the external member Dr. Nova Ahmed.

I have worked under HEQEP project "Capacity Building for Postgraduate Research in Remote Health Monitoring in Bangladesh" and my thesis was part of this project. I would like to thank Head of this project Prof.Dr. Ashikur Rahman for his supervision on my thesis and research work.

Finally, I would like to mention that I will remain ever grateful to my beloved parents and my brother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. At the end I would like express appreciation to my beloved husband Md. Anwar Parvez who spent sleepless nights with and was always my support in the moments when there was no one to answer my queries.

Abstract

This thesis proposes a patient's health data acquisition system for community health care through mHealth. In this system, patients gather in a community clinic where an attending health worker records patient condition through a mobile app. The collected data can be essentially multimodal. They may be of different formats: numbers (e.g., temperature reading), texts (e.g., description of symptoms), images (specific sign/scar/wound on body parts), audio (recorded interviews, health conditions such as cough/ breathing sound), and video (if required). These contents are partially synced to a backend server. They can be accessed by distant doctor now or later time to assess the patient case. Due to large size of media content and bandwidth limitation of the worker device, uploading all contents to the server is not possible (nor even required); most part of them can be stored locally and later can be delivered to the doctor's device upon request. On-demand delivery delays content upload to save costly upload bandwidth at the cost of slight increase in latency to retrieve content when needed. To this end, we build an mHealth service, called *dursheba*, that entails mobile apps (both for doctor and health worker), a server and a cloud messaging service enabling an effective synergy between patient and doctor in a rich fashion even though they are distant apart.

Chapter 1

Introduction

In this thesis, we propose a health data acquisition system for community health care through mHealth. In this system, patients gather in a community clinic or a service point where a health worker attends the incoming patients, records their health conditions and indicators (such as, vital signs) through a mobile app. The collected data are essentially multimodal and are of different formats: varies from simple numbers and texts to richer images and videos. All these contens are uploaded to a remote server for possible access by a remote doctor for possible assessment and suggestions. We develop a selective content upload scheme for this kind of distant healthcare setting.

1.1 Distant Healthcare System

Distant health care is a form of clinical service where patients and doctors do not remain to be in the *same* place or at the same time. Clinical healthcare is provided at a distance with telecommunications and information technology. It is the future of global healthcare. There are various distant healthcare system like video conferencing, store and forward data, and remote patient monitoring. Video conferencing enables realtime interactions between patients and providers. It decouples patient and doctor in space, but not in time. Store and forward telemedicine means acquiring medical data (like medical images,

biosignals etc.) and then transmitting this data to a doctor or medical specialist at a convenient time for an assessment offline. Remote patient monitoring is a setup of self-monitoring or testing that helps medical professionals to monitor a patient remotely using various technological devices. Distant healthcare system allows patients to maintain independence, prevent complications, and minimize clinical costs.

1.2 Healthcare through Mobile Devices

The emergence of mHealth platforms and services enables patients receiving health services in the form of medical prescriptions, clinical advices and doctoral referrals even though they are distant apart from doctors. These are all achieved through the usage (in some form or other) of handheld mobile devices, such as smart phones and tablets, possessed by average individuals these days. The primary form of mHealth is in the form of tele-doctor (or telemedicine) where a patient calls a distant doctor and have a conversation with the doctor regarding his/her health issues and complaints. In response, the doctor suggests medicines and other advices as deemed appropriate. While this service is simple and easy and has been in deployment for a while, it has several limitations. It requires a doctor to be present to converse to a patient for providing health service. Health-related smartphone apps are available in abundance in recent years, and some estimates predict that nearly 500 million people are using those.

1.3 Media Content in Distant Healthcare

In distant healthcare setup, patients will be examined by a doctor based on these recorded evidence alone. Therefore, patient data should be essentially rich. They must be *multimodal* in the sense that they consist information of different digital formats: from texts and numbers to rich media type. Media contents assist in reproducing patient case with high fidelity and provide good evidence.

A set of possible data formats as we may encounter in such distant healthcare services

are:

- **Numbers** (e.g., temperature reading, blood pressure, blood sugar, pulse, oxygen saturation etc.)
- **Texts** (description of symptoms)
- **Images** (specific sign/scar/wound on body parts)
- **Audio** (recorded interviews, health conditions such as cough sound, breathing sound, etc.)
- **Video** (certain description of a particular complain or patient condition)

1.4 Content Retrieval in mHealth Platforms

In distant healthcare, the patient data once collected are then uploaded to a server so that the data can be accessed by a distant doctor now or later time. While planning to upload contents, a couple of issues immediately come in. Media contents are usually of large sizes and they can be produced in good numbers with great ease (taking pictures do not take much effort, just a couple of tapping on the screen; the same holds for audio-visual contents). Moreover, upload bandwidth is a limited resource, particularly for a health worker who might have budgeted monthly paid data subscription (we are considering a remote setup where free WiFi may not be a viable option). So, it may not be possible to upload *all* contents to the server, at least not immediately right after they are recorded. Interestingly, not all contents are required to be uploaded either, because the doctor may not need all contents in order to provide an advice.

In order to achieve better resource utilization in our system, most part of the patient data are stored *locally* in the device and are later delivered to the doctor's device on doctor's request. This is what we refer to as *on-demand delivery*. On-demand delivery delays content upload only to save costly upload bandwidth at the cost of slight increase in latency.

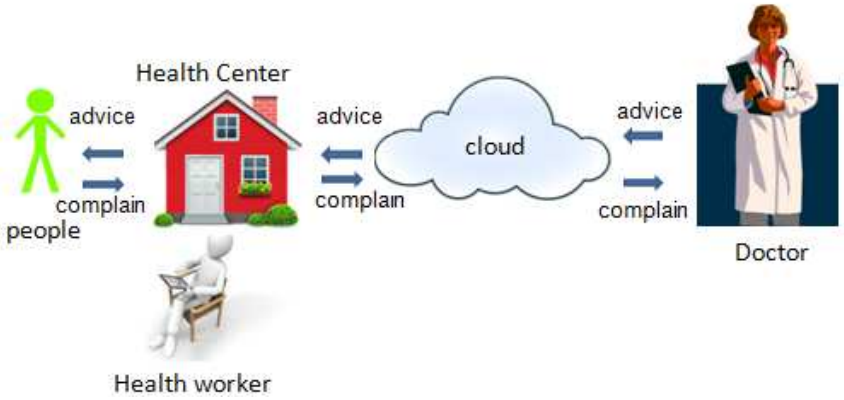


Figure 1.1: Remote health monitoring system

1.5 Contribution of the Thesis

This thesis proposes a patient's health data acquisition system for community health care through mHealth. In this system, patients gather in a community clinic where the attending health worker or nurse record patient condition (that we refer to as *patient data*) through a mobile app, which are then accessed by doctors remotely in later time to make necessary advices. It will be a severe waste of scare upload bandwidth if all contents for a given patient are uploaded but the consulting doctor did actually accessed only some of them. If the system can select useful contents, it can upload those earlier. It saves time and ensure delay free smooth operation.

The contribution of this thesis are:

- A scheduling technique for mHealth applications that can utilize upload bandwidth more effectively by selectively uploading large media contents instead of uploading all.
- This proposed scheme can be used by mobile phone based mHealth applications that handle rich media data.
- A mHealth service, called *dursheba* is developed, that entails a mobile app (both for doctor and health worker), a back-end server and a cloud messaging service. This enables an effective interaction between patient and doctor in a rich fashion even

though they are distant apart from each other.

1.6 Organization

The rest of the chapters are organized as follows. Chapter 2 gives a preliminary description of some distant health care system and mHealth services. Chapter 3, the main chapter of this dissertation, illustrates our proposed mHealth service and problem formulation for selective content upload scheduling. Chapter 4 contains the implementation of our android based mobile application *Dursheba*. Chapter 5 contains experimental results. Chapter 6 draws the conclusion mentioning the key contributions of this thesis followed by some future directions for further research in this field.

Chapter 2

Background and Related Work

In our work, we design and develop a mobile phone assisted distant healthcare service where patient data are collected and uploaded to a collection point for possible checkups by doctors or by trained health professionals. Unlike call center based telemedicine that decouples patients and doctors in space but not in time, we envision to decouple them both in time and space. In our system, patients gather in a health worker station (say, in a community clinic) and let their data to be recorded by the health worker. Data can be multi-modal including texts, numbers, images, audios and videos. Since media contents are large, they should not be uploaded to the cloud as soon as they are generated, but only when they are requested by the doctor. This deferred uploads save lots of upload bandwidth with a slight increase in latency to retrieve contents from the health worker device when needed. We propose interesting tradeoffs in choosing which items should be uploaded upfront and which items should be uploaded only on demand.

2.1 Popular mHealth Apps in App-stores

We made a comprehensive survey on several popular health application from Android platform. Average download count of these app is 50000+ and review score is 4.1. We identify possible usecases in which contests these apps are used. We categorize them

whether they collect patient or health data through the Internet and whether they work offline. A comparative analysis is shown in Table 2.1,

App	Category	Use Case	Connectivity	Size
Drugs.com Medication Guide	Drug Search Engine and pesonal record manager	The easiest way to lookup drug information, identify pills, check interactions and set up your user's personal medication records.	internet required	3.5MB
Diseases Dictionary	medical dictionary	This app contains medical disorders & diseases with detailed definitions, symptoms, causes and treatment information. This medical disease hand book can act as a clinical advisor for self diagnosis and can also be used to look up symptoms, diseases and treatment.	offline	2.9MB
Search, Know, Buy Medicines (HealthKart-Plus)	Generic Drug Search Engine.	App can be used to discover cost effective generic drugs that can substitute prescription medicine. User can explore how prescription medicine works & understand the required precaution and contraindication. Empowers user to compare drugs by prices to choose cost effective generic drug for a given prescription.	internet required	varies

Medscape	medical resource for healthcare professionals	Look up adult and pediatric drug dosing information in seconds, check drug interactions, access medical calculators, and get health plan formulary information to support you with patient care.	internet required	4.6MB
Prognosis : Emergency Medicine	clinical cases and discussions for experienced physicians	Emergency Medicine allows user to test decision making skills in a risk-free environment while sharpening clinical knowledge.	internet required	2.5MB
WebMD for Android	health information and decision-support tools.	The troubling part of the body is selected, symptoms are chosen & user can learn about potential conditions or issues. Find medically reviewed information about causes, treatments, & related symptoms	internet required	8.9MB

Home Remedies (Plus)	Complete guide to Home Remedies and Natural Cures for Common Ailments	Natural home remedies are safe and useful for the common ailments. Most of the non life threatening illness can be treated at home. Learn how one can use natural cures to replace many of the most commonly used over-the-counter drugs in the treatment of common ailments. If user uses home remedies wisely, it will save time and money.	offline database	2.4MB
First Aid	Instructions	First Aid is designed to help you follow the right procedures in a stressful situation or support other people by giving them instructions. It is based on illustrations, videos and short texts that show user to take the necessary action step by step and in the right order.	offline	3.4MB
Medical & Drug Dictionary	medical dictionary	Excellent coverage for general health & biomedical sciences. Provides three features: Annotator, Dictionary and Thesaurus. More than 300'000 medical/biomedical terms	internet required	2.9MB

Table 2.1: Overview of popular health Application

2.2 Existing Distant Healthcare Services

We checked existing distant healthcare services. A comparative analysis is shown in Table 2.2,

Distant Health Care Service	Description
Vigilant Medical [24]	Web applications for images in cloud. Enable physician collaboration to improve patient care.
Nuance PowerShare Medical Image [17]	View, manage, and share medical images and diagnostic reports with healthcare professionals and patients.
CrowdMed [4]	Patient can submit medical cases. Online medical diagnosis experts are ready to help. Community for doctors to solve complex cases collaboratively.
ETIAM [8]	Provides solutions for telemedicine, medical data archiving and diffusion for healthcare facilities.
Tata Cancer Hospital India-online treatment	Upload medical reports and evidence. Get expert suggestion within 3-7 days.

Table 2.2: Overview of Distant Health care Services

Most of these services are web-based. They don't consider budget constraints. But data budget is an important issue for network applications which is also considered in other applications like icloud photo sharing app [16], soundcloud [30], Evernote [9] etc. Certain activities in Google Apps that transfer large amounts of data in a short period of time, like synchronizing a Gmail account to a mobile phone or mail client, can cause an account to reach the bandwidth limit [12]. Google Analytics is used by millions of sites which put limits and quotas on API requests to protect the system from receiving more data than it can handle.

Most of the existing healthcare systems works with only image data. In these services, all collected data are uploaded, no selection method is applied for useful content.

2.3 Remote Health Monitoring System

Mobile based web services have become an attractive platform for delivering distant healthcare services to patients [7]. Some remote patient monitoring systems have been developed for particular diseases in context such as blood pressure, heart diseases and diabetics. A. G. Logan et al. [22] worked on developing a mobile phone-based remote patient monitoring system for the management of hypertension in diabetic patients. In this system, a bluetooth-enabled BP monitor reading is sent to the server via a mobile phone and the physicians report is sent via fax. Prema Sundaram [31] has developed remote patient monitoring system for cardiovascular diseases. In this method, the patients vital signs such as ECG, heart rate, breathing rate, temperature, and SpO_2 are captured, and the values are entered into a database. The readings are then uploaded into a web server as well as are sent to the doctors phone using Android technology. It also enables the doctors to instantly send back their feedback to the nurse station. Remote Health Monitoring using Embedded Systems was proposed by Santosh Kulkarni et al [28]. It is a real time telemedicine system which utilizes GSM/GPRS protocol. ECG signal is generated by simulation software and that signal is fed into a GSM modem to transmit data over the Internet.

In [3], authors presented RHMS for elderly people using smart mobile device. This paper addressed the challenge to provide caregivers an emergency alert system for the elderly based on monitoring of their heart rates, breathing activities, and room temperature measurements. The device was allowed to make an on-demand request for assistance. The remote communication was enabled through the cellular telephone service. Remote health monitoring using online social media systems was proposed by C. Khorakhun [21]. They proposed to use existing infrastructure and develop application for health professionals. Four different access viewpoints were implemented to suit the requirements of

each user in their example scenario to form a network. These viewpoints are from patient, doctor in charge, professional carers, and family members of the patient. Another remote patient monitoring system was developed by Fareed ud din et al. [10]. They proposed a data record system with validation and transmission of health tips to the patients and guardians. It performs patients vital signs monitoring and carries out remedial actions in emergency conditions (e.g., medicine dose tips, precautions alert to patient, alert to hospital staff in emergency conditions etc). Heurtefeux et al. [15] proposed a remote health monitoring system using wireless body area network. This Body Area Network (BAN) is fitted on human subjects to gather various physiological parameters through a set of sensors. Their proposed platform collects, processes and wirelessly transmits medical data from multiple wearable BAN to a medical control center (MCC) through a solar-powered and multi-hop mesh network.

2.4 Uploading Service and Data Sharing

Zhang et al. proposed media uploads with deadline handling [36], and [23] proposed techniques for automatic feature selection and hyper-parameter values using machine learning. Plazzotta et al. [27] proposed multimedia uploading service for multimedia health record. This paper describes the design, development and implementation processes of a service which allows media elements to be uploaded in a patient clinical data repository through an electronic health record by professionals or by patients. Danan et al. [32] proposed a platform for secure monitoring and sharing of generic health data in the Cloud. They combined smartphone, bluetooth and cloud to enable mobile telecare. They demonstrated a telecare application that allows doctors to remotely monitor patients via the Cloud. Kalyani et al. [2] proposed a smart remote healthcare data collection server. This paper presents a method to secure data collection server by protecting and developing backups used for healthcare cloud. This server can collect data and send to a centralized repository in a platform independent format without any network consideration. Girardi et el. [6] proposed a software system, Electronic Multimedia Health Fascicle (EMHF), which is for maintaining large number of electronic health records. Patient's clinical biometric

measurements and biologic parameters can be seen at a time to link any alarming physical status to his recent medical history. Their proposed software was web based and accessible from any mobile device. Anurag et al. [1] proposed health care delivery and data collection integration in eHealth center. An integrated solution eHealth Center was proposed with verifiable cloud-based electronic workflow and records, telemedicine capability and automated online reporting of summarized health data and operational status.

2.5 On-demand Content Retrieval

The idea of on-demand media retrieval was taken from Mediascope [18] which is a crowdsourcing platform allowing collection of image content from user mobile phones. Venkatagiri, S.P. [33] proposed an on-demand mobile video sharing system which allow user to share video captured at an event. On-demand Information Extraction (ODIE) was proposed in [29].

2.6 Prior Upload Content Selection

If we want to provide all health data on-demand, service latency would increase too much. So, we upload a subset of collected data before requesting them. In order to choose which items should be uploaded upfront, we need to consider a set of attributes or features. We got the idea of attribute selection from [20] and [13]. In [20], Kalankesh et el developed a methodology that reduces the dimensionality of primary health data to make data more amenable to perform visualization, mining and clustering. The methodology involved employing a combination of ontology-based semantic similarity and principal component analysis (PCA) to map the data into an appropriate and informative low dimensional space. Gotz et al [13] proposed a methodology for interactive mining and visual analysis of clinical event patterns using electronic health record data. This paper focused on temporal pattern mining, visual query and interactive visualization module.

2.7 Scheduling Algorithm for Limited Budget

In order to handle bandwidth limitation, we borrow some idea from [35]. In [35], author proposed an offline and online scheduler for energy fair mobile cloud computing server. Some of their scheduling algorithms are first-generated-first-scheduled (FGFS), Earliest deadline first (EDF), first-come-first-served (FCFS) etc. In [5], author showed that fair queueing provides several important advantages over the usual first-come-first-serve queuing algorithm. Harchol-Balter et al showed in [14] that, size based scheduling improves web performance. Authors showed that shortest remaining processing time first (SRPT) minimizes queuing time and response time and outperforms fair queuing.

Chapter 3

On Demand Media Content Retrieval for mHealth

In this chapter, we narrate our on-demand media content retrieval technique where a set of *selected* content items are uploaded (from the health worker device) to the remote server before the distant doctor accesses them. The system also retrieves content from the worker device on-demand when needed so. We also describe *Dursheba*, the distant healthcare service developed as a part of our work.

3.1 Media Content Collection

In our system, health workers collect data from patients in different formats such as text (for recording symptoms, readings of vital signs, physical conditions, etc.), images (for recording any specific sign on body parts for scar/mark/wound), audio (for recording interviews, or health conditions such as cough sound, breathing sound, etc.), and as well as video (if required).

Texts are needed for almost all patients. Signs and symptoms of diseases and main complains are stored in text format. Numbers are used to record various health indicators such as body temperature, blood pressure and heart-beat rate. Images can express more

than text. In case of skin diseases, photograph of scar or infected body region help doctor in assessment/treatment. Various pathological reports, for example blood test report, can also captured by taking images if they are needed to be passed as a supporting evidence (thanks to mobile phones and tablets coming with high resolution cameras). Audio recordings are helpful in case of respiratory problems, such as coughing sound and vocal sound document level of infection. Similarly, video data are helpful in many ways. To transfer data from one device to another remote device an intermediate storage is required.

Note that both the doctor and patients can produce media data. Doctor can prescribe patient with audio recordings. He can upload image of prescription or advice.

Patient data is collected and stored in a smart phone device possessed by the health worker. Our system delivers these collected data to a remote server from which serving doctors can access them for assessment. Since these are media contents and are usually large in size, their transfer is a non-trivial matter to handle. We have three possible choices for content transfer.

1. Upload all collected data.
2. Upload only meta data of content. Meta data means basic info, symptom and list of content.
3. Upload partial content by selection.

Upload all: All collected patient data can be uploaded to the cloud immediately after they are generated. Doctors can be provided with a web portal or an app to download and view any data anytime. One disadvantage of this scheme is its high upload bandwidth requirement. The cloud storage also needs to be large enough to store all these data. Unnecessary content may be uploaded in this scheme which perhaps are never used.

Upload on demand: Due to bandwidth and energy constraints, it is often not efficient or desirable to upload all patient data to server for sharing immediately after storing them. Only a set of basic data and a list of other available contents are uploaded to the

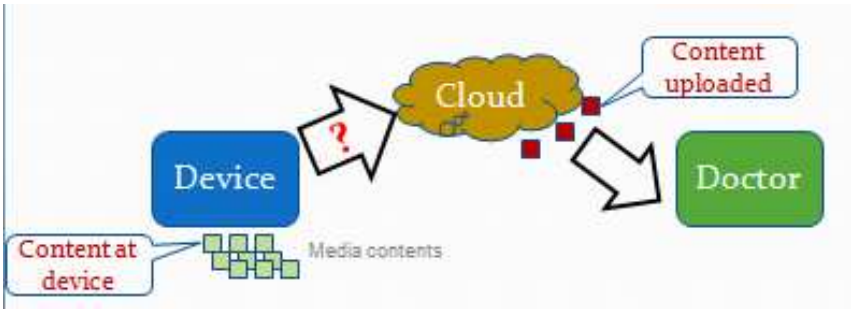


Figure 3.1: Selective media content upload

cloud server. The basic data may consist of patient Id, main sign, symptom of patients diseases and list of media contents accompanied with the patient. When the remote doctor asks for any content from the list, the requested content is uploaded. Advantage of this scheme is that it requires limited upload bandwidth. Besides, unnecessary contents are not uploaded to the cloud. One disadvantage of this scheme is that limited energy of device may hinder uploading when required. So, the requested data content may not be provided to the doctor on time. Latency can be another issue of this scheme. Since all data are uploaded on demand, content retrieval latency can be quite high.

Upload partial content by selection: In this scheme, all collected patient data is stored in the health worker's device. Some selected contents from those data are uploaded to the server before doctor requests them. Useful content selection for prior uploading is done to reduce latency. It optimizes bandwidth usage and latency.

3.2 Selective Content Upload

Dursheba allows contents to be uploaded from the health worker's device to the doctor's device on-demand. For efficiency and timely delivery, the system also chooses a subset of items to be uploaded to the storage server before the doctor even requested them. This is an opportunistic upload scheme that leverages doctor's prior content retrieval pattern in order to upload some part of the patient data in the hope that uploaded contents will be eventually accessed by the doctor. The benefit of uploading upfront is that they may provide reduced latency when the doctor accesses them. The cost of upload is wasted,

however, if doctors do not consume them eventually. The benefit of not uploading upfront rather uploading contents on demand is the opposite. It saves resources, but incurs higher latency when accessed. In the following, we give a scheduling algorithm to decide which items to upload upfront.

In this section, we formulate the upload scheduling problem and then provide a scheduling algorithm to decide which items to upload to the storage server before the doctor even requested them.

3.2.1 Formulation of the Upload Scheduling

We assume that each piece of content whether uploaded apriori or retrieved on-demand (or never used at all) renders differentiated *value* based on where the content reside the doctor needs them. We assign a value to each content piece as a means of measuring how useful they are as they are accessed by the doctors. Contents may arise in three possible categories:

- System upload the content and doctor also accessed them (Category 1),
- Doctor requested the content but system did not upload it a priori (Category 2),
- System uploaded the content but the doctor did not access it (Category 3).

There can be a forth category: contents that were neither uploaded nor requested by the doctors. But this case is trivial, so can be ignored. We argue that the value of a content depends on which category it belongs. For example, contents that have been uploaded as well as consumed (Category 1) have higher value than the items that were requested by a doctor but was not uploaded before and retrieved later (Category 2). This value reduction is to compensate the additional cost it takes in terms of latency to retrieve the item from the worker device. Again, the value of a content that was uploaded upfront but was not later accessed by the doctor does also have a value, but in negative sense because it wasted upload resource unnecessarily. In that, the value rather becomes a *penalty*.

We assume a certain *value or penalty function*, $\phi_k(c)$, exists that gives the value of content c based on its category, $k \in \{1, 2, 3\}$. We expect, in general, the following order holds for a given content c:

$$\phi_1(c) > \phi_2(c)$$

Given a set of patient data that are uploaded and the set of data that have been accessed by the doctors, we can define an overall value derived from all such contents.

Let us assume, we have n number of media contents. For simplicity, we do not consider the patients associated with those contents, rather the system sees a series of content items coming at the ingestion point (i.e., at the health worker device). Let s_i denote size (in bytes) of content c_i. Let p_i denote a binary indicator whether doctor accessed (or requested) content c_i, and x_i denote whether that content is uploaded upfront and available at the server when the doctor requested it.

Given upload matrix $X = \{x_i\}$ and doctor's request matrix $P = \{p_i\}$, we can determine the total value for all contents as follows:

$$V(X, P) = \sum_{i=1}^{n} \big(p_i x_i \phi_1(c_i) + p_i(1 - x_i)\phi_2(c_i) - x_i(1 - p_i)\phi_3(c_i)\big) \tag{3.1}$$

The first summand corresponds to the value derived from Category 1 contents followed by the second summand from the items of Category 2. The last one is due to Category 3 contents, which is in fact a penalty due to unnecessary uploads (hence, the summand is negated). Therefore, the problem of identifying which contents to upload is to find X so as to maximize objective function $V(X, P)$. This is—

$$(\textbf{OPT-UPLOAD})\ \text{Find}\ X^* = \arg\max V(X, P) \tag{3.2}$$

The problem of solving OPT-UPLOAD is that we do not know requests P. For the ease of exposition, let us replace binary p_i with a *probability value* that measures how likely the doctor accesses content c_i. In that, p_i takes value within $[0, 1]$, instead of being binary

$\{0, 1\}$. Hence, P becomes a probability distribution of content accessed by the doctors. These probability values can be estimated from an earlier access history or through a model.

We can shuffle terms in expression (3.1) to have:

$$V(X, P) = \sum_i x_i(p_i\phi_1(c_i) - p_i\phi_2(c_i) - (1 - p_i)\phi_3(c_i)) + \sum_i p_i\phi_2(c_i)$$

Since we are finding X to maximize the above expression, we can eliminate the last term being a constant, which results in:

$$\max \sum_i x_i(p_i\phi_1(c_i) - p_i\phi_2(c_i) - (1 - p_i)\phi_3(c_i)) \tag{3.3}$$

The solution to the above unconstrained optimization is achieved by assigning x_i to 1 only if—

$$p_i\phi_1(c_i) - p_i\phi_2(c_i) - (1 - p_i)\phi_3(c_i) > 0 \tag{3.4}$$

which gives X as follows:

$$x_i = \begin{cases} 1, & \text{if } p_i > \frac{\phi_3(c_i)}{\phi_3(c_i)+\phi_1(c_i)-\phi_2(c_i)} \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

We can say that if probability value p_i is greater than a *threshold*, then the content in question will be uploaded apriori. Threshold value, $P_{th}(i)$, for uploading can be written as:

$$P_{th}(i) = \frac{\phi_3(c_i)}{\phi_3(c_i) + \phi_1(c_i) - \phi_2(c_i)} \tag{3.6}$$

Note that the above formulation is an unconstrained optimization problem. In practice, however, there will be a certain constraint in place. One such constraint can be *data budget*, meaning that the amount of data the device can upload at a certain time. This is a natural constraint because we can safely assume that most health worker would use

devices with metered and limited cellular data connection which might have weekly or monthly cap on how much data volume it can upload within that period (say, for example, 200MB per week). Our selection approach should respect this constraint.

Let ρ be the maximum allowed rate (in bytes) for data upload per hour. So, at hour t, the system is allowed to upload a total of $\rho \times t$ bytes. The upload scheduler keeps track of bytes it has uploaded so far and tries to keep the total upload amount below the allocation. If we assume content upload happens at only hour marks and a total of $U(t)$ bytes have been uploaded so far upto to hour t, the data budget, denoted as B, for the next hour can be computed as, $B = [\rho \times (t+1) - U(t)]^+$ (here, $[A]^+ = \max(A, 0)$). Therefore, the objective of content selection is to choose items within this budget constraint. If for any reason, budget becomes zero, the selected contents items are put in queue, and uploads are delayed so that the system can accumulate data budget.

Hence, we can reformulate Equation 3.3 as follows:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} x_i(p_i\phi_1(c_i) - p_i\phi_2(c_i) - (1-p_i)\phi_3(c_i)) \\ \text{subject to} \quad & \sum_{i=1}^{n} x_i s_i \leq B \end{aligned} \tag{3.7}$$

Let $v_i = p_i\phi_1(c_i) - p_i\phi_2(c_i) - (1-p_i)\phi_3(c_i)$. So, we have the following optimization:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} x_i v_i \\ \text{subject to} \quad & \sum_{i=1}^{n} x_i s_i \leq B \end{aligned} \tag{3.8}$$

which is a Knapsack problem and reportedly NP-Hard. We can adopt the classical greedy heuristic: pick items in decreasing order of v_i/s_i until the sum of sizes exceeds B.

What remains is to show, we discuss how we compute utility functions, $\phi_k(c_i)$ as well as probability values, p_i, for contents, which we describe next.

3.2.2 Calculating Utility Values

Matrix X gives us which contents should be uploaded upfront and which should not. It appears that it depends on how the system sees the value of objects across different categories. Two marginal cases can be analyzed. If the system does not penalize for missed uploads, that is, $\phi_3(c) = 0$, everything should be uploaded (as $p_i > 0$ always holds setting all x's to one). On the other hand, if the system sees no difference between the value of Category 1 and Category 2 contents, that is $\phi_1(c) = \phi_2(c)$, there is no point of uploading any; all x's are set to zero because $p_i > 1$ never holds. In that, all contents will be accessed on demand. Sometimes, the system may not upload all contents as per X, but restricts total uploads within a limit (if it is set forth so).

We observed that, $\phi_k(c)$ should depend on certain attributes of the content. As we have discussed that, utility value of Category 2 content is less than Category 1 for extra delay, so the latency can be attributed to be the main reason for varying utility values. For Category 1 contents, content retrieval time is the time to get any contents from the server to the doctor's device. In case of Category 2 contents, extra time is added to make these contents available to server. So, utility values for Category 1 and Category 2 contents can be defined as follows:

$$\begin{aligned} \phi_1(c_i) &\propto \frac{1}{t_1(i)} \\ \phi_2(c_i) &\propto \frac{1}{t_1(i) + t_2(i)} \end{aligned}$$

Here,

$t_1(i)$ = time needed to transfer content i from server to doctor's device

$t_2(i)$ = time needed to transfer content i from health worker's device to server

Category 3 contents have negative utility as they are not useful but uploaded apriori by the system. Utility value for Category 3 can be defined some multiple of the utility of Category 1 items: so,

$$\phi_3(c_i) = \alpha\phi_1(c_i) = \frac{\alpha}{t_1(i)} \tag{3.9}$$

Here, α is called the *penalty factor*, which indicates how the system weighs the penalty for wrong uploads against the benefit of useful uploads. For resistive operations, we will penalize more for wrong uploads. In that case, the system uploads fewer contents but try to ensure that uploaded contents are all truly useful. Hence, we set $\alpha > 1$. For optimistic operation, however, the value of α can be decreased to increase the number of uploaded contents. The value of α in effect determines the threshold value of probability. Threshold probability value P_{th} (omitting the index for content) can be written as—

$$P_{th} = \frac{\phi_3}{\phi_3 + \phi_1 - \phi_2}$$

$$P_{th} = \frac{\frac{\alpha}{t_1}}{\frac{1}{t_1} - \frac{1}{t_1+t_2} + \frac{\alpha}{t_1}}$$

It can be simplified as:

$$P_{th} = \frac{\alpha(t_1 + t_2)}{t_2 + \alpha(t_1 + t_2)}$$

If we assume that the time needed to transfer data from health worker's device to the server equals to the time needed to transfer data from the server to the doctor's device, that is, $t_1 = t_2$, then the threshold value becomes as follows:

$$P_{th} = \frac{2\alpha}{1 + 2\alpha} \tag{3.10}$$

If we increase the value of penalty factor, uploading threshold increases. High threshold value means fewer contents will be uploaded. If the threshold value decreases, then more content will be uploaded. We observe that, the followings holds: if $\alpha \to 0$, then $P_{th} = 0$ (all contents are uploaded) and when $\alpha \to \infty$, then $P_{th} = 1$ (nothing gets uploaded apriori). So, by tuning α, the system can move from one extreme to the other.

3.2.3 Probability Values Computation

The probability that the doctor will consume a certain content item may depend on many factors, which include the attributes of the content (such as, content type and content size), the attributes of the patient such as patient's gender, age, and complaint type. It may also depend on individual doctors personal preference. The best way to estimate the probability of consuming a certain content is to "learn" the access pattern from historical data and use a probabilistic model to predict probability values.

In our system, we estimate p_i from fitting a logistic regression model on content access patterns. As per regression model, we first need to determine the set of attributes or *features* that influence a doctor's decision to consume a content. A set of possible features can be content type (text, images, etc), diseases type, patient's gender and age, and content size. Once the features are identified, the model tries to compute *weights* for each feature and then generates a probability value as a function of sum-product of feature values and their weights. Let $f_1, f_2, \cdots, f_k$ be k features and $\beta_1, \beta_2, \cdots, \beta_k$ be their corresponding weights. Hence, the sum-product is $\overline{f}.\overline{\beta} = \beta_0 + \sum_{i=1}^{k} f_i \times \beta_i$, where β_0 is called the *bias* or the intercept term. This sum product is then passed to the *logistic* function to obtain the probability value associated with that content. The probability value is given by:

$$p_i = \sigma(\overline{f}.\overline{\beta}) = \frac{1}{1 + e^{-\overline{f}.\overline{\beta}}} \tag{3.11}$$

The logistic function (also known as sigmoid function) is used in this context which takes an input with any value from negative to positive infinity and always generates output values between zero and one, and hence is interpretable as a probability value. Figure 3.2 shows the logistic function. Note that, while the feature values, $\overline{f}$, are given for each content item, the weights, $\overline{\beta}$, are model parameters and are "learned" from prior access history. Given a set of prior dataset, called as the *training data set*, the logistic regression learning refers to determining these weights that best fits the model.

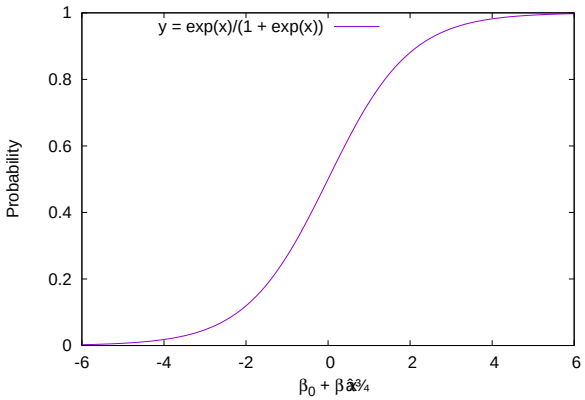


Figure 3.2: Logistic or sigmoid function

Learning weights: Once the model is learned based on the training dataset, we can then predict newer instances whether the content will be consumed by the doctor or not. But, these weighting factors can be changed with time, environment and based on other reasons. So, the learning should be incremental. For incremental learning, in our system we use the basic idea of the *perceptron algorithm* [11]. The algorithm is shown in Algorithm 3.1.

Algorithm 3.1 Algorithm for incremental learning

Get Initial weight vector β_t from classifier model

Receive instance

Predict x_p from weight vector

Receive true value x_t

if $x_p \neq x_t$ **then**

$\beta_{t+1} \leftarrow \beta_t + \eta(x_t - x_p)$

else

$\beta_{t+1} \leftarrow \beta_t$

end if

In the algorithm, x_p is the predicted value that whether the doctor will consume that content or not and x_t is the true value for content consumption. Here in algorithm, η is called the incremental learning rate for the classifier. The value of η can be $0 < \eta < 1$. In

practical cases, value of η is taken between 0.05 to 0.2. Learning rate affects how quickly the system is trained. If the learning rate is too small, it takes much time to learn the threshold and if it is high, the system becomes oscillating and does not converge to a stable condition.

Chapter 4

Implementation of Dursheba

In this chapter, we describe the prototype implementation of our proposed work as an Android based mobile application, named *Dursheba*. Dursheba has been released in Google Play and is available for community people to use. Note that the app is an output of a reseach work and is mostly to demonstrate interaction between doctor and patients as described in the earlier chapters. It is not by any means intended for production use.

4.1 Service Architecture and Components

The app keeps track of patient data in different formats. The app has provisions to enable such recording and store them all as a bundle per patient so that they can be individually accessed. After each visit, recorded data would be uploaded to a server. Both the health worker and the doctor use this app for patient data recording and treatment. Since media contents are big, a probable solution plan for uploading these data is proposed and implemented. This application is designed with on demand data uploading schemes. It can be divided in several modules to accomplish the whole task.

Figure 4.1: Dursheba App in Google Play

4.1.1 Data Acquisition Module

Health worker creates a profile for each patient. All data related to a patient are stored in a chunk together. This chunk data can be divided into two parts.

- **Basic Data and Metadata:** Patient name, sex, weight, temperature, blood pressure and list of media content data. Basic data is common for all patients.
- **Other media data:** Patient data may contain media data of different format. Data can be audio record of cough sound, image of scars etc.

Data of different formats are collected and stored in the local memory of Android devices. We used following Android APIs for collecting data:

- *android.hardware.Camera*: For images and video data
- *android.media.MediaRecorder*: For audio data

4.1.2 Data Sharing Module

An intermediate server is used for sharing data between health worker's device and doctor's device. A simple server based on Node.js is created for this purpose and hosted at OpenShift [26]. In our system we called this server as the RHMS server. Patient data

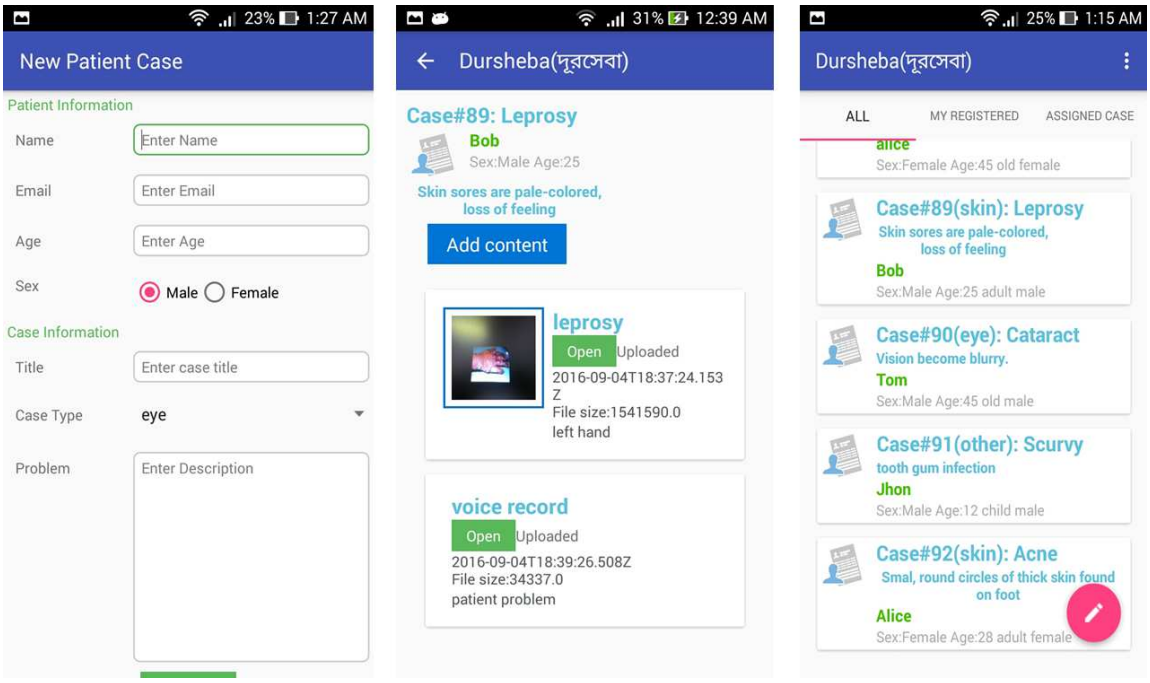


Figure 4.2: Screenshot from the Developed App

is stored in MongoDB [25] database of server. Patients data in health workers devices database is synchronized with patient information in the server database.

Doctor diagnoses disease or problem with the help of media data. Then, he gives feedback or treatment in his device. This feedback is synchronized with the RHMS server. Health worker's device receives this from the server.

4.2 Handling Media Content

As media contents are large in size, sharing these contents with doctor is challenging. All the media contents are stored locally in health worker's device. According to the data budget and other resources, some of the contents are uploaded to server earlier. If some other contents are asked by doctor, those contents are uploaded to server on demand.

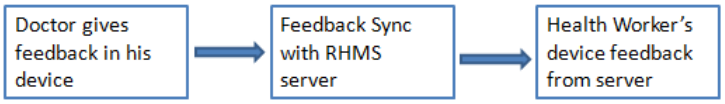


Figure 4.3: Feedback or treatment

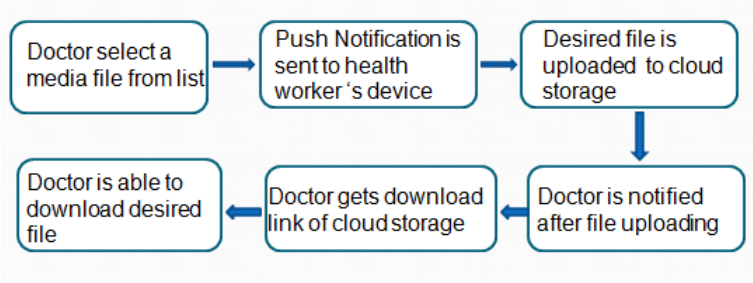


Figure 4.4: Media data sharing

4.2.1 Content Selection Module for Prior Uploading

It is important to understand what characteristics and features of contents influence physician's decision. It has been seen from statistics that the main features that influence doctors are: age, gender, content type, disease type, content size.

Prior uploading content can be selected using machine learning and learning from previous cases. At the very beginning we uploaded all data to server. Among them, we observed that which cases are really consumed by doctor. From this observation we made a classifier model and used it to predict whether doctor will consume that content or not. Classifier model was generated from weka. NaiveBayesUpdateable, logistic regression classifier was build to generate model. From logistic regression classifier model we get weighting factor of different attributes and prediction value which determines whether a content will be selected for prior uploading or not.

We generated classifier model using WEKA [34]. We integrated that model file with android code. This model is written on a file and stored in device memory while installing. According to this model, content is selected for prior uploading.

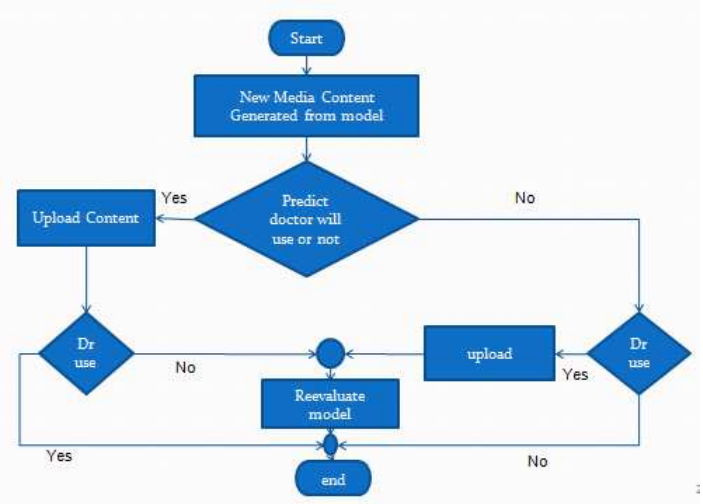
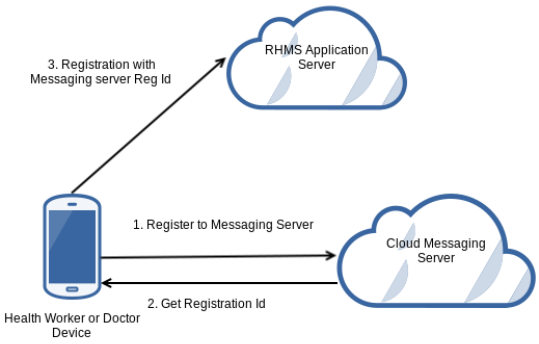


Figure 4.5: Flowchart for incremental learning

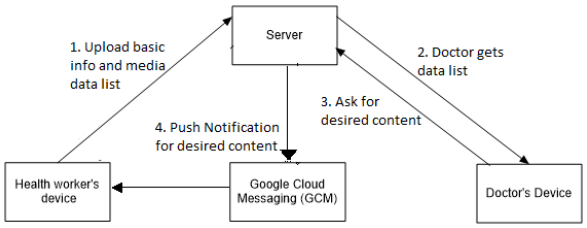
4.2.2 On Demand Content Retrieval

Push notification service like Google Cloud Messaging (GCM) is used for reliable and energy efficient connection between server and Android devices. Both health worker and doctor register their devices in GCM and receive a GCM registration Id. Server identifies each device with corresponding Id. Doctor selects media content from the list of media data. Push notification is sent to the server for corresponding media file with a specific file Id. Then server sends push notification to health workers device with this file Id. Device registration in GCM server is shown in Figure 4.6a and request handling is shown in Figure 4.6b.

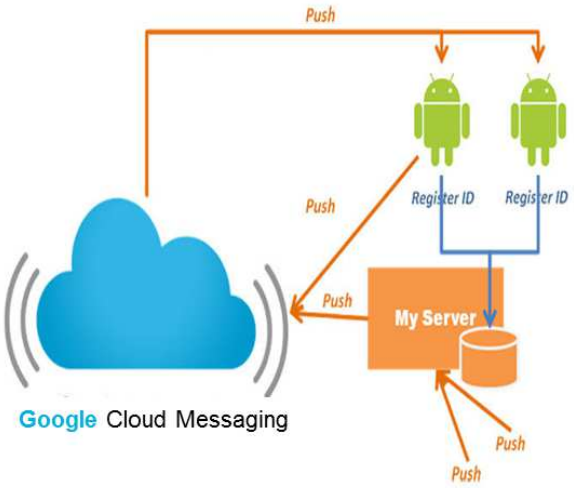
When doctor wants to get a media content, a push notification is sent to health worker's device with corresponding file Id. When health worker's device receives push notification, the desired file is uploaded to the cloud server. When the desired content is fully uploaded, the doctor is notified with a message. Then the doctor will be able to download and check his required content for treatment. Media data sharing module is shown in Figure 4.4.



(a) Device Registration



(b) Request handling module



(c) Push Notification from GCM server

Figure 4.6: Device registration and on demand content retrieval.

Chapter 5

Experiments and Evaluation

In this chapter, we describe the simulation results for our distant healthcare system through mHealth. Through simulation, we study the behavior of our approach and evaluate its performance based on some metrics. We also analyze real time performance of our android application, *Dursheba*.

5.1 Platform Setup and Generation of Simulated Data

We developed a network simulator and a program to mimic whole scenario. Program was written in Java. Synthetic data is used in experiments.

In this experiment,

Number of health worker = 1

Number of doctor = 1

Number of media content = 500

Number of content arrival rate at health worker = 30 per hour

Number of content consumption rate by doctor = 18 per hour

For generating patient content, we followed Poisson distribution function. It expresses the probability of a given number of events occurring in a fixed interval of time and/or

space if these events occur with a known average rate.

$$\delta = -\frac{1}{\lambda} \ln U \tag{5.1}$$

here,

λ = Number of content arrival per hour

U = Random number between 0 to 1

δ = Interval time between 2 content arrival

If we want to get interval time in minute, then

$$\delta = -\frac{60}{\lambda} \ln U \tag{5.2}$$

Similarly doctor checks patient's content following Poison's rule. Equation for content consumption by doctor is

$$\delta = -\frac{60}{\mu} \ln U \tag{5.3}$$

here,

μ = number of content consumption by doctor per hour

We used the logistic regression model to generate a labelled data set. We assume that each content has a set of attributes, namely types of the health complain, content type, content size, gender of the patient, age of the patient). All attributes except content size take nominal/categorical values, so we introduce binary dummy variable to denote the category. For example, age is categorized into three groups: Child (Age <=18), Adult (18 < Age <50) and Old (Age >= 50). Hence, we introduce three variable (Child, Adult, Old) to denote which group the patient belongs. Similar variables are used for other nominal attributes.

For content size, we assume that the size of individual content depends on its content

types, such as text, image, audio and video. We assume that the corresponding average content sizes are (1, 200, 500, 2000) KB (sizes vary 50% more or less around the mean). Since content sizes are large numbers and vary in a wider scale, we take logarithm of sizes. The weight used for log(size) is 0.15.

For given set of weights, we then compute as $w.x$ for each content. The doctor is assumed to consume the content if 5.4 is true, otherwise he does not.

$$\frac{e^{w.x}}{1+e^{w.x}} > P_{th} \tag{5.4}$$

We generate a series of 500 such cases out of which around 60% contents are consumed by the doctor. This ratio, however, can be varied across experiments. Synthetic data are used to meet specific needs or certain conditions that may not be found in the original, real data. This can be useful when designing any type of system because the synthetic data are used as a simulation or as a theoretical value, situation, etc. This allows us to take into account unexpected results and have a basic solution or remedy, if the results prove to be unsatisfactory. Synthetic data are often generated to represent the authentic data and allows a baseline to be set. Another use of synthetic data is to protect privacy and confidentiality of authentic data. Synthetic data is used in testing and creating many different types of systems. As we don't have enough real data and all possible combination may not appear in small data set, we used synthetic data.

At the very beginning, we don't have any information about which contents are really useful. We uploaded first 100 data to the server and observed which contents were really consumed by the doctor. We have true value set that whether doctor has consumed any data or not. From these data, we generated a classifier model for logistic regression. Logistic regression was done using WEKA [34] library. We calculated weighting factor for all cases of all attributes for first 100 data. In Table 5.1, we have shown all categorical variables and associated weights of classifier model. Using this model, predicted value is generated for each content and from this decision contents are prior uploaded. We compared predicted value with real true value for doctor use. If this model predicts wrong, than it is re-evaluated and weighting factors are adjusted with this data. We have

Attribute	Category	Weights
Complaint type	Eye, Gynae, Other, Respiratory, Skin	-96.04, 72.33, 12.07, 82.63, -101.99
Content type	Audio, Image, Text, Video	-130.43, 26.74, 245.49, -159.51
Size	Real Value	42.29
Gender	Male, Female	78.48, 21.62
Age	Child, Adult, Old	-9.76, 10.81, -3.69

Table 5.1: Simulation Data

implemented incremental learning to keep classifier model up to date and accurate. We also used NaiveBayesUpdateable classifier from WEKA for another updateable classifier model.

5.2 Performance Metric

The ***performance metrics*** we consider in our simulation are as follows,

- Total Utility Value
- Accuracy Precision Recall F1 Score
- Time Average Queue length

Parameters are the value, that are changed to see performance metrics. Some parameters are:

- Data Budget (MB per hour)
- Unwanted Upload penalty (α)
- Patient content arrival rate
- Consumption rate by doctor

Total Utility Value : Total utility value is the cumulative sum value of all media contents. It is calculated as follows:

$$V(X,P) = \sum_{i=1}^{n} \big(p_i x_i \phi_1(c_i) + p_i(1-x_i)\phi_2(c_i) - x_i(1-p_i)\phi_3(c_i)\big) \tag{5.5}$$

Accuracy Precision Recall F1 Score: Performance metrics are measured on number of true positive, false positive, false negative and true negative.

In our experiment,

True Positive = System prior uploaded and doctor used

False Positive = System prior uploaded but doctor did not used

False Negative = System did not prior uploaded but doctor requested

True Negative = System did not prior uploaded and doctor did not used

Accuracy refers to how close a measurement agrees with a known value of that measurement. It indicates the correctness of system. Accuracy of classifier means how correctly classifier can identify a media content as useful or not. System accuracy means how correctly system can deliver useful contents to server. System may fail to deliver selected contents to server due to budget constrain though classifier selected contents successfully. When there is no budget constrain or upload speed limitation, system accuracy is same as classifier accuracy.

$$\text{Accuracy} = \frac{\text{No of true positive} + \text{No of true negative}}{\text{Total number of contents}} \tag{5.6}$$

Precision can be thought of as a measure of a classifiers exactness. A low precision can also indicate a large number of False Positives.

$$\text{Precision} = \frac{\text{No of true positive}}{\text{No of true positive} + \text{No of false positive}} \tag{5.7}$$

In binary classification, recall is called sensitivity. Recall can be thought of as a measure of a classifiers completeness. A low recall indicates many False Negatives.

$$\text{Recall} = \frac{\text{No of true positive}}{\text{No of true positive} + \text{No of false negative}} \tag{5.8}$$

In statistical analysis of binary classification, the F1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

$$\text{F1 Score} = 2\frac{\text{Precision* Recall}}{\text{Precision} + \text{Recall}} \tag{5.9}$$

Time Average Queue Length: Queue length indicates average length of queue over time.

$$\text{Time Average Queue Length} = \frac{\sum_{i=1}^{n} T_i L_i}{\sum_{i=1}^{n} T_i} \tag{5.10}$$

5.3 Performance Evaluation

5.3.1 Accuracy of different classifier

We performed our experiment with logistic regression classifier model which learns continuously from new instance. We also used online classifier which does not have initial weight vector. Some other classifier model can be used to predict value whether the doctor will use a content or not. NaiveBayes classifier model can be used to predict data. Naive Bayes fits feature weights independently while logistic regression accounts for correlations amongst features. A comparative study was done between discriminative model logistic regression and generative models Naive Bayes classifier in [19]. Naive Bayes is linear classifier which clasifies using Bayes Theorem and strong independence condition among features.

Some performance metric of classifier used in this system are shown in Table 5.2

Accuracy of classifier depends on what fraction of selectively uploaded data was used

Classifier	η	Accuracy	Precision	Recall	F1-Score
NaiveBayes	N/A	0.92	0.91	0.97	0.939
Logistic Regression	0	0.98	0.99	0.97	0.979
Logistic Regression	0.05	0.982	0.99	0.98	0.985
Logistic Regression	0.1	0.981	0.987	0.985	0.986
Only Online Learning	0.1	0.574	0.677	0.687	0.68
Only Online Learning	0.15	0.598	0.689	0.72	0.704

Table 5.2: Comparison of classifier

Type	True Positive	False Positive	False Negative	True Negative
Count	327	6	3	164

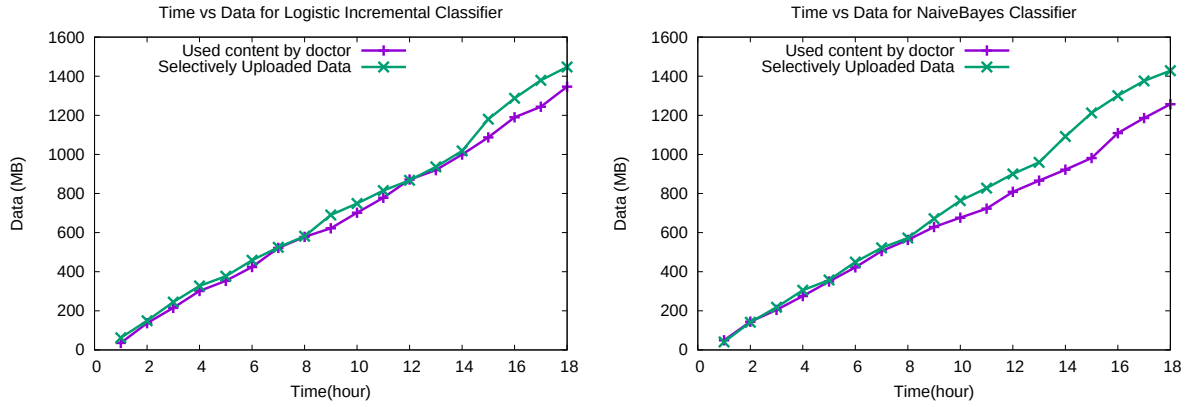
Table 5.3: Count

by doctor. It is shown in Figure 5.1 that, maximum portion of selectively uploaded data were consumed by doctor.

When simulation is performed for 200 MB Data Budget, 1MBps upload speed, penalty factor 1, content arrival rate 30/hour, content consumption rate 18/hour, then number of contents for our system is shown in Table: 5.3

5.3.2 Effect of Upload Scheme

Total *utility value* is calculated for all 3 upload schemes. Our aim is to maximize total value of utility function. Media content is collected with progress of time, and each content adds a value according to the category of content. Three different schemes and their respected utility values are plotted in Figure 5.2a. For "upload all" data scheme, we can see value of utility function increases, but for wrong upload it decreases at some points. For "upload only meta data" scheme, all media contents are delivered to the server on demand. In this case, value function never decreases but it adds less value for each content as delay is added for retrievals. From our experiment, we can see that selective



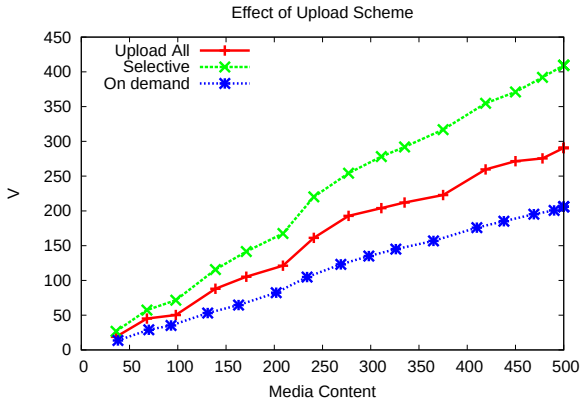
(a) Logistic incremental classifier (b) NaiveBayes Updateable classifier

Figure 5.1: Fraction of used content and selectively uploaded content

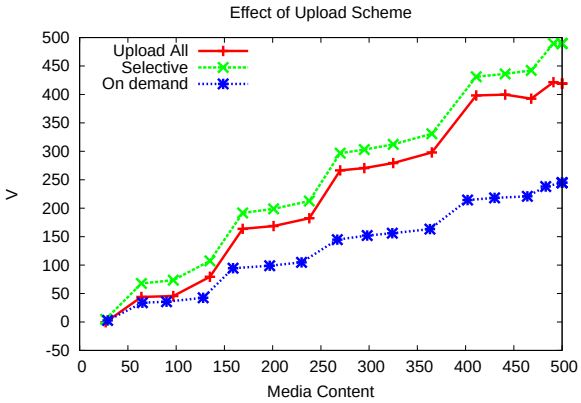
content uploading scheme has maximum value from objective function.

If number of media contents used by doctor dominates in data, then Figure 5.2b occurs. Here we produced a dataset where 80% of total contents were used by doctor. From our experiment, we can see, that upload all data scheme performs better than upload on demand scheme. Large number of contents were used by doctor. As number of wrong upload is less, upload all scheme performs better. When on demand scheme is used much delay accumulates for all contents and performance degrades. But for selective upload scheme, most of contents were uploaded upfront and number of wrong upload was lesser than upload all scheme. So, selective upload scheme outperformed other schemes.

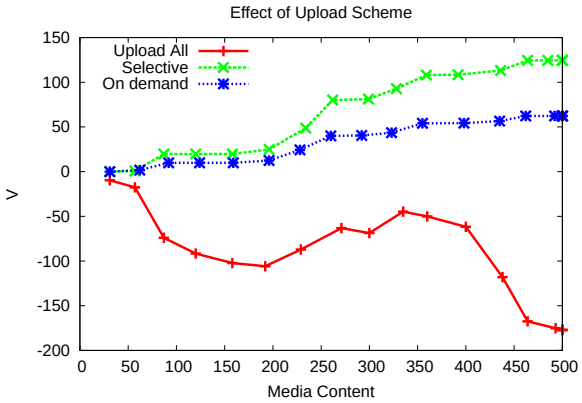
If number of media contents used by doctor is inadequate in data, then Figure 5.2c occurs. Here we can see that, when 20% contents are used in data set, upload on demand scheme performs better than upload all scheme. Large number of contents were unused by doctor. As number of wrong upload was more, upload all scheme performed worse. On demand uploading scheme performed better because it has no wrong upload. But for selective upload scheme, most of required contents were uploaded upfront and number of wrong upload was lesser than upload all scheme. So, selective upload scheme outperformed other schemes.



(a) 60% Contents Used by Doctor



(b) 80% Contents Used by Doctor



(c) 20% Contents Used by Doctor

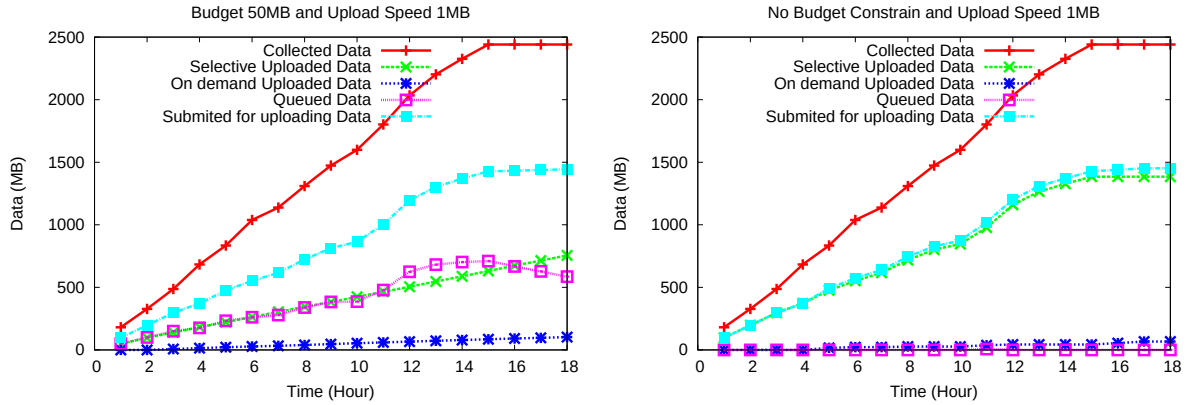
Figure 5.2: Effect of Upload Scheme

5.3.3 Effect of *Data Budget*

Bandwidth budget influences performance of the system.

Effect of *Data Budget* on uploaded data From Figure 5.3a, we can see, that data is collected in every hour and number of collected data increase with time. Size of collected data and uploaded data are also shown here. Among all collected data, a subset of collected data is selected for prior uploading. Selected contents are uploaded by system until it reaches upload bandwidth limitation. If it exceeds limit value, then it is stored in a queue.

For large data limit media contents in waiting queue grows at a smaller rate which is shown in Figure 5.4a. When there is unlimited budget, no contents remain in queue which



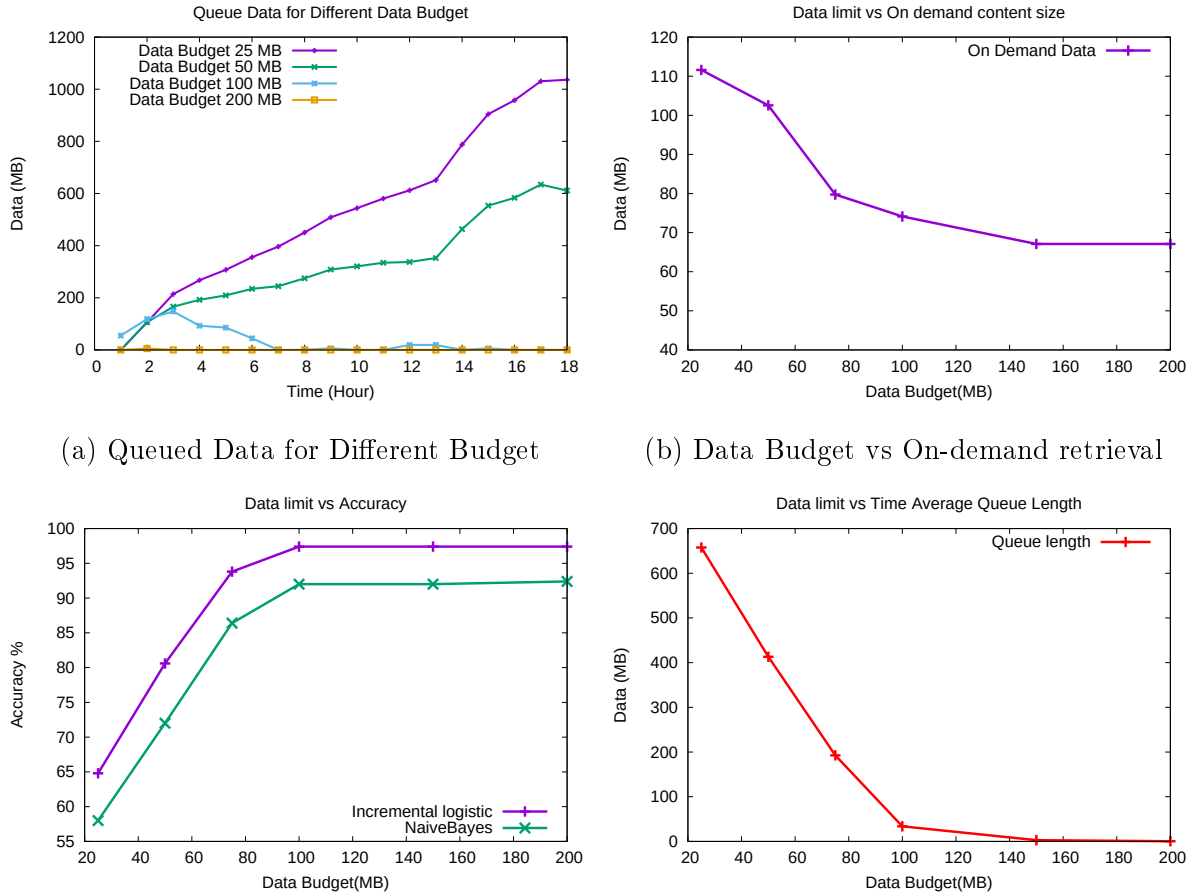
(a) Upload limit 50MB upload rate 1MBps (b) No Budget constrain upload rate 1MBps

Figure 5.3: Data Status over Time

can be observed from Figure 5.3b. Items in queue are sorted according to the value and size of content (value/size). So, contents with large size remain at the end of the queue and uploaded later. It takes long time to finish uploading, when upload limit is set to a small value. Some contents may remain in queue and do not get chance to be uploaded. Uploading finishes earlier when upload limit is large. For limited budget, when content is in queue for prior uploading, it is not available at the server. So, the doctor may ask for on demand content retrieval. As a result, on demand content retrieval increases for limited budget. It is shown in Figure 5.4b

Effect of *Data Budget* on accuracy: Accuracy of system increases with data budget. If data budget is low, media content in queue grows larger. It takes much time to make that content available to doctor. In some cases for budget constrain contents may not be uploaded to server. As a result accuracy falls down for limited budget. It is shown in figure 5.4c.

Effect of *Data Budget* on Time average Queue Length: Time average queue length increases for limited data budget. When data budget is low, selected items are gathered in queue. Queue length gets longer. When data budget is adequate, then contents from queue release quickly. As a result queue length remain small. It is seen from Figure 5.4d, that when data budget is high amount of data in queue is small.



(a) Queued Data for Different Budget

(b) Data Budget vs On-demand retrieval

(c) Data Budget vs Accuracy

(d) Data Budget vs TAQL

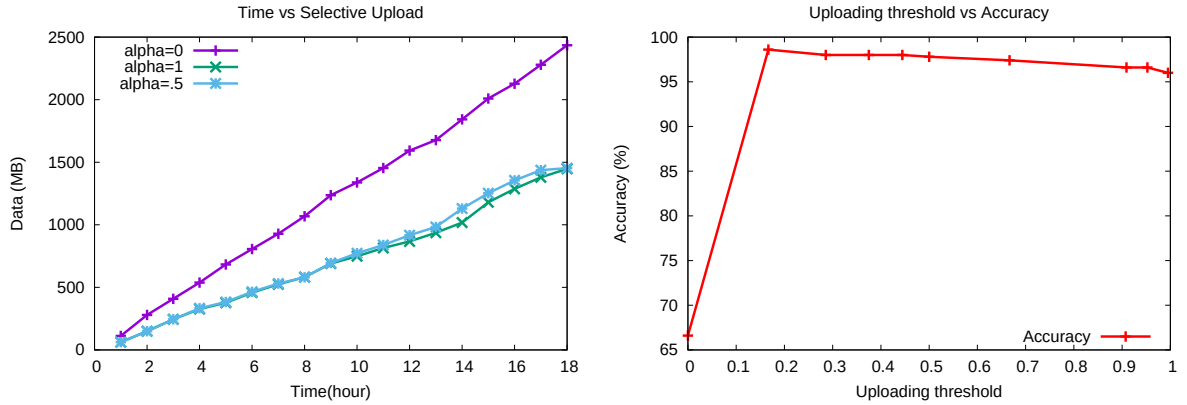
Figure 5.4: Effect of Data Budget

5.3.4 Effect of Penalty Factor or Upload Threshold

Probability threshold value P_{th} for uploading depends on penalty factor. Penalty factor may take any real value. Upload threshold value depends on it and varies from 0 to 1.

Effect of Penalty Factor or Upload Threshold on Prior upload and on Demand Upload: If upload threshold value is low, most of the contents are uploaded apriori. We can see from Figure 5.5a that when upload threshold is zero, all collected contents are prior uploaded. Amount of prior uploaded content decreases with the increase of upload threshold. As a result, the number of on demand uploading content increases for high threshold value.

Effect of Penalty Factor or Upload Threshold on Accuracy: Upload threshold



(a) Selectively uploaded data (b) Upload Threshold vs Accuracy

Figure 5.5: Effect of Uploading threshold

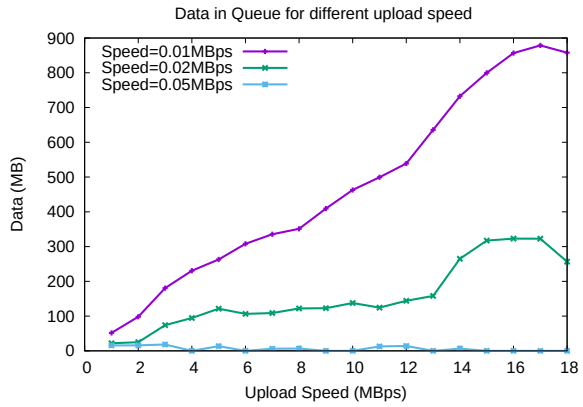
also affects the accuracy of classifier model. Accuracy of classifier model is maximum for a certain threshold value. It is shown in Figure 5.5b. If threshold value is low, the number of prior uploaded content is high. As a result, number of wrong upload increases. Most of the contents are uploaded but many of them are not consumed by doctor. Similarly, when threshold value is low, very few contents are uploaded upfront. As a result, number of on demand contents increased and some required contents are uploaded after doctor requested them. For wrong prediction, accuracy of classifier model decreases.

5.3.5 Effect of Upload Speed

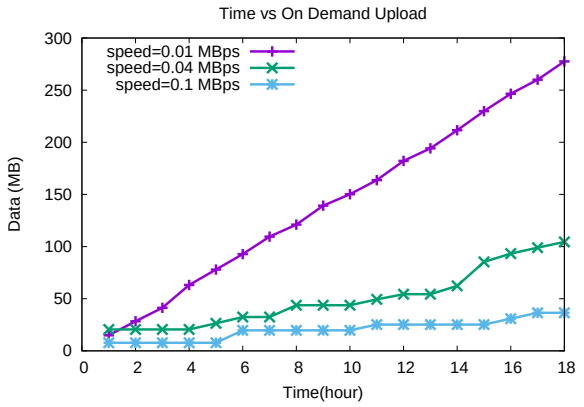
The quality of network facility influences uploading performance. As a result, items in queue grows larger and they take a long time to finish uploading. If we limit uploading hours then all required contents may not be uploaded. As we sorted queue according to the decreased order of probability/size, so large contents are at the bottom of queue. If network facility is available for limited hours, large contents do not get any chance to be uploaded. They remain in queue.

Effect of *Upload speed* on uploaded data: We can see from Figure 5.6b, that if upload limit is high but upload speed is low, system cannot utilizes it's allocated data budget. As a result, uploading does not complete for selected data. Amount of selective

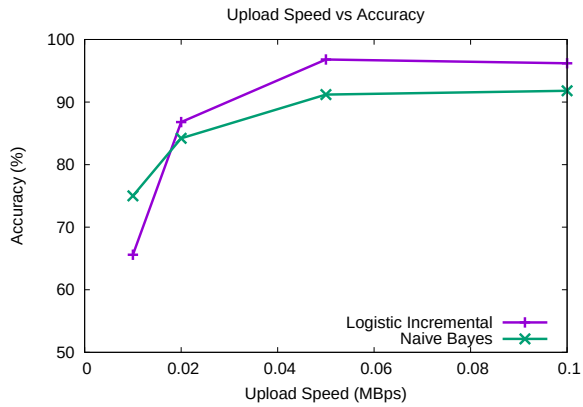
uploaded content is less. As doctor does not avail required contents in queue, he asks for it. So, on demand content retrieval is high for low speed. For sufficient upload speed, selective contents are uploaded timely and on demand media retrieving is less.



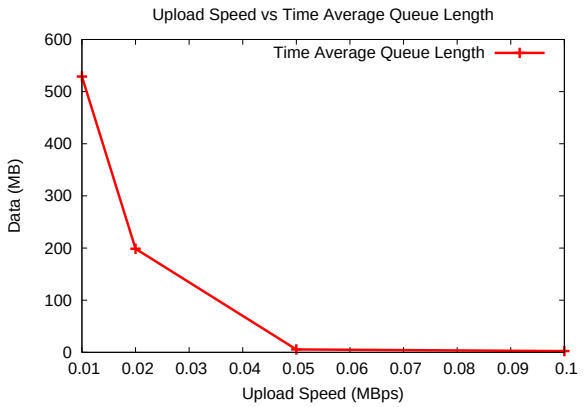
(a) Queued Data varying Upload Speed



(b) On-demand Data varying Upload Speed



(c) Upload Speed vs Accuracy



(d) Upload Speed vs TAQL

Figure 5.6: Effect of Data Uploading speed

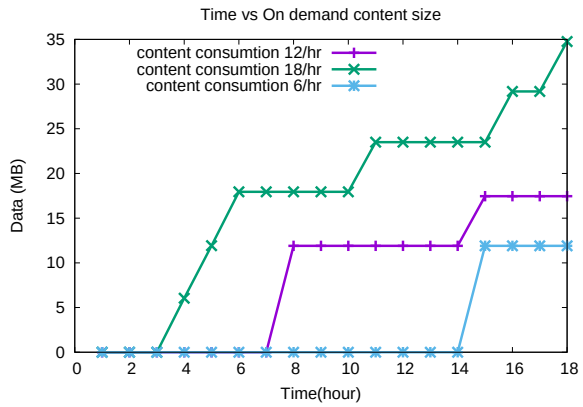
Effect of *Upload speed* on accuracy: Upload speed affects the accuracy of system. If upload speed is very low, selected contents for prior uploading are gathered in queue. As many of them are not uploaded to server and not delivered to doctor, system accuracy falls. For good upload speed, contents are uploaded on time and accuracy is high. It is shown in Figure 5.6c.

Effect of *Upload speed* on time average queue length: Upload speed has effect on time average queue delay. When upload speed is low, selected contents remain in

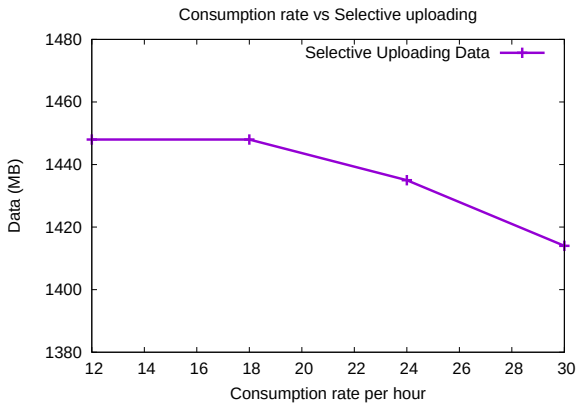
queue. Only a few of them are uploaded to server. As a result, time average queue length is high for low upload speed. It is shown in Figure 5.6d.

5.3.6 Effect of Content Consumption Rate by Doctor

Content consumption rate of doctor affects uploading data. If content consumption rate is high, doctor uses content more frequently. Doctor may not find desired content in server, though that content has been selected for uploading and waiting in uploading queue. As doctor does not get that content in server, he asks for uploading. As a result on demand upload increases. Since total data budget is limited, selective upload decreases to give chance to on demand contents. It is shown in Figure 5.7a and Figure 5.7b.



(a) On Demand Data varying μ



(b) Consumption rate vs Prior uploaded data

Figure 5.7: Effect of Content Consumption rate by doctor

5.4 Result Summary

We have discussed the effects of parameters on performance metric in Section 5.3. As a summary, it is shown in Table 5.5. Comparative analysis among upload schemes are shown in Table 5.4.

Content Set	Scheme	Utility value for 500 contents	Best Performer
60% Media contents were used	Upload all	292.6	Selective
	On Demand	208.4	
	Selective	407.7	
80% Media contents were used	Upload all	427.2	Selective
	On Demand	233.7	
	Selective	482.3	
20% Media contents were used	Upload all	-167.1	Selective
	On Demand	67.9	
	Selective	134.8	

Table 5.4: Comparison among upload schemes

Change in Parameter	Accuracy	TAQL	On demand uploading	Prior uploading
Data Budget increases	increase	decrease	decrease	increase
Data Budget decreases	decrease	increase	increase	decrease
Upload Speed increases	increase	decrease	decrease	increase
Upload Speed decreases	decrease	increase	increase	decrease
Content Consumption Rate increases	–	–	increase	decrease
Content Consumption Rate decreases	–	–	decrease	increase
Penalty Factor increases	increase upto a level and then decrease	–	–	decrease

Table 5.5: Result Summary

5.5 Results from Implementation

Dursheba android application was tested for 3 different schemes. A number of media contents were collected by application. Uploading time was measured for each contents. We tested our application in 4 devices. Those are:

- Samsung Galay S2 LTE - CPU Dual-core 1.5 GHz, RAM 768 MB, internal storage 8 GB, OS jellybean
- Samsung Galaxy J5 - Quad-core 1.2 GHz, RAM 1.5 GB, internal storage 8 GB, OS Marshmallow
- Samsung Galaxy Tab - Dual-core 1.0 GHz, RAM 1 GB, internal storage 8 GB, OS jellybean
- Asus Zenfone 5 - Quad-core 1.8 GHz, RAM 2 GB, internal storage 16 GB, OS jellybean

Size of collected content varied according to the type of contents. Image contents were collected in $2048X1152$ resolution and their average size was 1 MB. Video content was taken in $640X480$ resolution and their average time length was 1 minute and size was 5MB. Audio contents were small in size and their average size was 60KB. Text contents size were very small.

Some collected data from android application is shown in Table 5.6 :

Content no	Clinical Type	Content Type	Age	Gender	Size(MB)
C_1	eye	image	adult	male	0.225
C_2	cough	audio	child	male	0.066
C_3	digestion	text	adult	female	0.006
C_4	skin	image	old	female	0.4866
C_5	eye	video	adult	male	2.23
C_6	skin	image	adult	male	0.551

Table 5.6: Data collected from Android Application

Content no	C_1	C_2	C_3	C_4	C_5	C_6
Uploading Time (ms)	3366	1796	234	4561	7123	5147

(a) Upload all Scheme

Content no	$n_1(t)$(ms)	$u(t)$(ms)	$n_2(t)$(ms)	$l(t)$(ms)
C_1	2135	3620	1340	7095
C_2	2276	1610	1256	5142
C_3	2089	211	1380	3580
C_4	2178	4381	1443	8002
C_5	2565	6744	1389	10698
C_6	2615	4784	1552	8951

(b) On Demand Scheme

Table 5.7: Uploading time

Upload all content scheme was applied first their uploading time was recorded in Table 5.7a

For on-Demand content retrieval scheme, more time is required to bring content to server from health worker's device. It is known as Latency. Latency period is time to make file available to doctor after it is requested. Interaction delay is time that doctor needs to wait before he can see items he requested (when retrieved from worker device). From our experiment, it is shown that uploading time is proportional to file size.

$$l(t) = n_1(t) + u(t) + n_2(t)$$

where,

$l(t)$ = Latency Period;

$n_1(t)$ = Time to notify health worker's device with desired content;

$u(t)$ = Data upload time (transfer image/ audio/ video data);

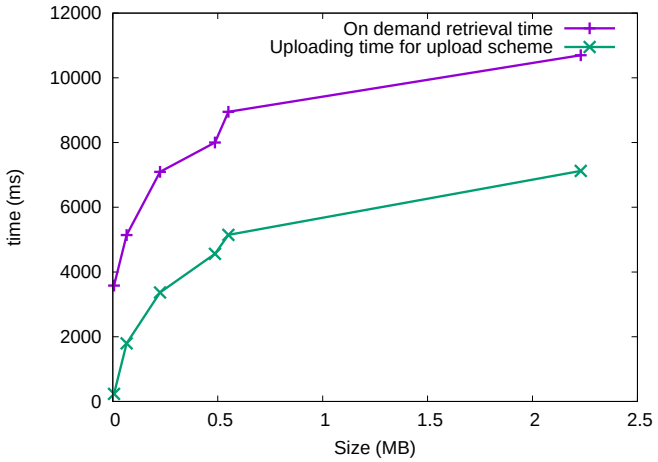


Figure 5.8: Uploading time vs Media file size

$n_2(t)$ = Time to notify doctor's device that content is available at server;

GCM delay was measured in our experiment both for 3G and wifi connection. GCM worked well for both wifi and 3G connection.

Success rate is 100%

We applied "selective content uploading" scheme in android application. We produced some media data, which is similar as our simulation model data. Then we tested with logistic regression classifier model, which we made in simulation section. This classifier successfully predicted whether this content will be used by doctor or not. Our android application did not upload all data apriori. It only uploaded selected contents to the server. We compared our result with actual value and it matched.

Chapter 6

Conclusion and Future Work

We developed a mobile application that keep track of patient data in different formats such as text, image, audio and video. We also developed a scheduling algorithm to decide which items to upload to the storage server before the doctor even requests them. It has been shown in our thesis that, selective content uploading scheme performs better than other schemes. We considered data budget limitation and service latency to evaluate our scheme.

For a remote health monitoring system, a number of health workers and doctors may work simultaneously. Multiple health workers and doctors facilities is provided in our application. Multiple health worker and doctor can log in to system. As several requests may arrive simultaneously, request management queue was implemented to handle these requests. We used priority queue where priority was determined with the value and size of the content.

We understand that proper authentication is required for accessing sensitive medical data of patients. All access to data is restricted through user login (health workers and doctors have user accounts. Necessarily credentials are always checked before performing any operation on behalf of any user. So, the patient data will be stored in the storage with privacy protection. It will not be accessible to others. Sufficient authorization technique is applied to ensure privacy. It only allows the designated user to see patient data. Individual

user id and *password* feature for health worker and doctor exists to ensure privacy.

Our system can be seamlessly interfaced with other devices. Weight machine, temperature sensor, heart rate monitor and other devices can connect with this system. A prototype of Arduino based temperature sensor module was developed that shows ambient temperature of the environment. If health device supports connectivity via Bluetooth, data can be automatically recorded by our system. We believe by providing an effective synergy between patient and doctor even though they are distant apart, our system will be widely used in the area of remote health monitoring.

Bibliography

[1] Anurag Agrawal, Jaijit Bhattacharya, Nishant Baranwal, Sushil Bhatla, Salil Dube, Viren Sardana, Devender R Gaur, Danica Balazova, and Samir K Brahmachari. Integrating health care delivery and data collection in rural india using a rapidly deployable ehealth center. *PLoS Med*, 10(6):e1001468, 2013.

[2] Kalyani Bangale, Karishma Nadhe, Nivedita Gupta, Swati Singh Parihar, and Gunjan Mankar. Smart remote health care data collection server. *International Journal of Computer Science and Mobile Computing*, 3:415–422, 2014.

[3] Matthew Clark, Jongil Lim, Girma Tewolde, and Jaerock Kwon. Affordable remote health monitoring system for the elderly using smart mobile device. *Sensors & Transducers 184.1 (2015): 77*, 2015.

[4] CrowdMed. https://www.crowdmed.com.

[5] Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and simulation of a fair queueing algorithm. In *ACM SIGCOMM Computer Communication Review*, volume 19, pages 1–12. ACM, 1989.

[6] Giovanni Dimauro, Danilo Caivano, Francesco Girardi, and Marco Matteo Ciccone. The patient centered electronic multimedia health fascicle-emhf. In *Biometric Measurements and Systems for Security and Medical Applications (BIOMS) Proceedings, 2014 IEEE Workshop on*, pages 61–66. IEEE, 2014.

[7] Khalid Elgazzar, Muhammad Aboelfotoh, Patrick Martin, and Hossam S Hassanein. Ubiquitous health monitoring using mobile web services. *Procedia Computer Science 10 (2012): 332-339*, 10:332–339, 2012.

[8] ETIAM. http://www.etiam.com.

[9] Evernote. https://help.evernote.com/hc/en-us/articles/209005247-What-are-the-system-limits-of-Evernote-.

[10] Muhammad Ibrahim Fareed ud din, Atta-ur-Rehman Shah. Mobile phone based remote patient's vital signs monitoring and automated alerts. *International Journal of Computer Science and Mobile Applications*, 2013.

[11] Stephen I Gallant. Perceptron-based learning algorithms. *IEEE Transactions on neural networks*, 1(2):179–191, 1990.

[12] Google. https://support.google.com/.

[13] David Gotz, Fei Wang, and Adam Perer. A methodology for interactive mining and visual analysis of clinical event patterns using electronic health record data. *Journal of biomedical informatics*, 48:148–159, 2014.

[14] Mor Harchol-Balter, Bianca Schroeder, Nikhil Bansal, and Mukesh Agrawal. Size-based scheduling to improve web performance. *ACM Transactions on Computer Systems (TOCS)*, 21(2):207–233, 2003.

[15] Karel Heurtefeux, Nasreen Mohsin, and Hamid Menouar. An hybrid platform for remote health monitoring: From concept to deployment. *International Conference on New Technologies, Mobility and Security (NTMS)*, 2015.

[16] iCloud. https://support.apple.com/en-us/HT202299.

[17] Nuance PowerShare Medical Image. http://www.nuance.com.

[18] Yurong Jiang, Xing Xu, Peter Terlecky, Tarek Abdelzaher, Amotz Bar-Noy, and Ramesh Govindan. Mediascope: selective on-demand media retrieval from mobile

devices. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 289–300. ACM, 2013.

[19] A Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14:841, 2002.

[20] Leila Kalankesh, James Weatherall, Thamer Ba-Dhfari, Iain Buchan, and Andy Brass. Taming EHR data: using semantic similarity to reduce dimensionality. In *MedInfo*, pages 52–56, 2013.

[21] Chonlatee Khorakhun and Saleem N Bhatti. Remote health monitoring using online social media systems. *Wireless and Mobile Networking Conference (WMNC), 2013 6th Joint IFIP*, 2013.

[22] Alexander G Logan, Warren J McIsaac, Andras Tisler, M Jane Irvine, Allison Saunders, Andrea Dunai, Carlos A Rizo, Denice S Feig, Melinda Hamill, Mathieu Trudel, et al. Mobile phone based remote patient monitoring system for management of hypertension in diabetic patients. *American journal of hypertension*, 20(9):942–948, 2007.

[23] Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5(1):1–16, 2016.

[24] Vigilant Medical. http://vigilantmedical.net.

[25] MongoDB. https://www.mongodb.org.

[26] OpenShift. https://www.openshift.com.

[27] Fernando Plazzotta, John C Mayan, Fernando D Storani, Juan M Ortiz, Gastón E Lopez, Gastón M Gimenez, and Daniel R Luna. Multimedia health records: user-centered design approach for a multimedia uploading service. *Studies in health technology and informatics*, 210:474, 2015.

[28] S. G. Kulkarni Santosh Kulkarni. Remote health monitoring using embedded systems. *International Journal of Emerging Research in Management and Technology*, 4, 2015.

[29] Satoshi Sekine. On-demand information extraction. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 731–738. Association for Computational Linguistics, 2006.

[30] SoundCloud. http://help.soundcloud.com/.

[31] Prema Sundaram. Patient monitoring system using android technology. *International Journal of Computer Science and Mobile Cumputing (IJCSMC)*, pages 191–201, 2013.

[32] Danan Thilakanathan, Shiping Chen, Surya Nepal, Rafael Calvo, and Leila Alem. A platform for secure monitoring and sharing of generic health data in the cloud. *Future Generation Computer Systems*, 35:102–113, 2014.

[33] Seshadri Padmanabha Venkatagiri, Mun Choon Chan, Wei Tsang Ooi, and Jia Han Chiam. On demand retrieval of crowdsourced mobile video. *Sensors Journal, IEEE*, 15(5):2632–2642, 2015.

[34] Ian H Witten, Eibe Frank, Leonard E Trigg, Mark A Hall, Geoffrey Holmes, and Sally Jo Cunningham. Weka: Practical machine learning tools and techniques with java implementations. *International workshop on Emerging Knowledge Engineering and Connectionist-based information systems*, 1999.

[35] Jianting Yue. *Energy Fair Cloud Server Scheduling in Mobile Computation Offloading.* PhD thesis, McMaster University, Hamilton, Ontario, Canada, 2015.

[36] Mu Zhang, Johnny S Wong, and Wallapak Tavanapong. Design and implementation of a media uploading system, 2005.