

4-2017

Mobile Big Data Analytics in Healthcare

Aramzana Nujum Navaz

Follow this and additional works at: https://scholarworks.uaeu.ac.ae/all_theses

Part of the [Information Security Commons](#)

Recommended Citation

Nujum Navaz, Aramzana, "Mobile Big Data Analytics in Healthcare" (2017). *Theses*. 620.
https://scholarworks.uaeu.ac.ae/all_theses/620

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarworks@UAEU. It has been accepted for inclusion in Theses by an authorized administrator of Scholarworks@UAEU. For more information, please contact fadl.musa@uaeu.ac.ae.



جامعة الإمارات العربية المتحدة
United Arab Emirates University

United Arab Emirates University

College of Information Technology

Department of Information Systems and Security

MOBILE BIG DATA ANALYTICS IN HEALTHCARE

Alramzana Nujum Navaz

This thesis is submitted in partial fulfilment of the requirements for the degree of
Master of Science in Information Technology Management

Under the Supervision of Dr. Mohamed Adel Serhani

April 2017

Declaration of Original Work

I, Alramzana Nujum Navaz, the undersigned, a graduate student at the United Arab Emirates University (UAEU), and the author of this thesis entitled “*Mobile Big Data Analytics in Healthcare*”, hereby, solemnly declare that this thesis is my own original research work that has been done and prepared by me under the supervision of Professor Dr. Mohamed Adel Serhani, in the College of Information Technology at UAEU. This work has not previously been presented or published, or formed the basis for the award of any academic degree, diploma or a similar title at this or any other university. Any materials borrowed from other sources (whether published or unpublished) and relied upon or included in my thesis have been properly cited and acknowledged in accordance with appropriate academic conventions. I further declare that there is no potential conflict of interest with respect to the research, data collection, authorship, presentation and/or publication of this thesis.

Student's Signature: Alramzana

Date: 15/5/2017

Copyright © 2017 Alramzana Nujum Navaz
All Rights Reserved

Advisory Committee

1) Advisor: Dr. Mohamed Adel Serhani

Title: Associate Professor

Department of Information Systems and Security

College of of Information Technology

2) Co-advisor: Dr. Marton Gergely

Title: Assistant Professor

Department of Information Systems and Security

College of of Information Technology

Approval of the Master Thesis

This Master Thesis is approved by the following Examining Committee Members:

- 1) Advisor (Committee Chair): Dr. Mohamed Adel Serhani

Title: Associate Professor

Department of Information Systems and Security

College of Information Technology

Signature 

Date 30/04/2017

- 2) Member: Dr. Mohammad Mehedy Masud

Title: Associate Professor

Department of Information Systems and Security

College of Information Technology

Signature 

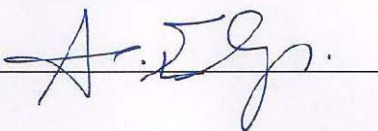
Date 30/04/2017

- 3) Member: Dr. Ashraf M. Elnagar

Title: Professor

Department of Computer Science

Institution: University of Sharjah

Signature 

Date 30/4/2017


This Master Thesis is accepted by:

Dean of the College of Information Technology: Professor Omar El-Gayar

Signature 

Date May 15, 2017

Dean of the College of Graduate Studies: Professor Nagi T. Wakim

Signature 

Date 17/5/2017

Copy 10 of 10

Abstract

Mobile and ubiquitous devices are everywhere around us generating considerable amount of data. The concept of mobile computing and analytics is expanding due to the fact that we are using mobile devices day in and out without even realizing it. These mobile devices use Wi-Fi, Bluetooth or mobile data to be intermittently connected to the world, generating, sending and receiving data on the move. Latest mobile applications incorporating graphics, video and audio are main causes of loading the mobile devices by consuming battery, memory and processing power. Mobile Big data analytics includes for instance, big health data, big location data, big social media data, and big heterogeneous data. Healthcare is undoubtedly one of the most data-intensive industries nowadays and the challenge is not only in acquiring, storing, processing and accessing data, but also in engendering useful insights out of it. These insights generated from health data may reduce health monitoring cost, enrich disease diagnosis, therapy, and care and even lead to human lives saving.

The challenge in mobile data and Big data analytics is how to meet the growing performance demands of these activities while minimizing mobile resource consumption. This thesis proposes a scalable architecture for mobile big data analytics implementing three new algorithms (i.e. Mobile resources optimization, Mobile analytics customization and Mobile offloading), for the effective usage of resources in performing mobile data analytics. Mobile resources optimization algorithm monitors the resources and switches off unused network connections and application services whenever resources are limited. However, analytics customization algorithm attempts to save energy by customizing the analytics process while implementing some data-aware techniques. Finally, mobile offloading algorithm decides on the fly whether to process data locally or delegate it to a Cloud back-end server. The ultimate goal of this research is to provide healthcare decision makers with the advancements in mobile Big data analytics and support them in handling large and heterogeneous health datasets effectively on the move.

Keywords: Mobile Big Data, Analytics, Healthcare, M-Health, Mobile resource efficient algorithms, Smart mobile algorithms, Mobile resources optimization, Analytics customization, Mobile offloading.

Title and Abstract (in Arabic)

تحليل البيانات الكبيرة المتنقلة في الرعاية الصحية

الملخص

الأجهزة النقالة في كل مكان من حولنا تولد كمية كبيرة من البيانات، يعد مفهوم الحوسبة المتنقلة أخذ في التوسع بسبب حقيقة أننا نستخدم الأجهزة النقالة كل يوم داخل وخارج البيت بدون التحقق من ذلك. وتستخدم هذه الأجهزة النقالة تقنية الواي فاي، وتقنية بلوتوث أو البيانات المتنقلة للإتصال، لتوليد وإرسال واستقبال البيانات. تعد أحدث تطبيقات الجهاز المحمول التي تدمج الرسومات والفيديو والصوت من الأسباب الرئيسية لزيادة الضغط على الأجهزة المحمولة، حيث تستهلك البطارية، الذاكرة، ووحدة التشغيل.

يعد التحدي الأكبر في استغلال البيانات المتنقلة وتحليلات البيانات الكبيرة في كيفية تلبية متطلبات الأداء المتزايد والتقليل من استهلاك الموارد. تقترح هذه الأطروحة حلاً قابلاً للتطوير وتقدم ثلاثة خوارزميات (تحسين موارد الجوال، وتخصيص تحليلات الجوال، وتفرغ الهواتف المتحركة)، من أجل الاستخدام الفعال للموارد في إجراء تحليلات البيانات المتنقلة. تقوم خوارزمية تحسين موارد الجوال بمراقبة الموارد وإيقاف تشغيل اتصالات الشبكة غير المستخدمة وخدمات التطبيقات كلما كانت الموارد محدودة، تحول خوارزمية التخصيص تحليلات لتوفير الطاقة عن طريق تخصيص عملية تحليلات أثناء تنفيذ بعض تقنيات علم البيانات. وأخيراً، خوارزمية تحميل المحمول تقرر بحيوية ما إذا كان ممكناً معالجة البيانات محلياً أو تفويضها إلى خادم الخلفية. الهدف النهائي من هذا البحث هو إعطاء صناع القرار حول الرعاية الصحية حلولاً لاسعمال المحمول في تحليلات البيانات الكبيرة ودعم التعامل مع مجموعات البيانات الصحية الكبيرة وغير المتجانسة بشكل فعال.

مفاهيم البحث الرئيسية: موبايل البيانات الكبيرة، تحليلات، الرعاية الصحية، موبايل - الصحة، الخوارزميات الذكية، موبايل الموارد الأمثل، تحليلات التخصيص، موبايل التفريغ.

Acknowledgements

All praise to Allah (S.W.T) who granted me patience, wisdom, memory and knowledge in completing my thesis and studies. My sincere thanks go to Dr. Mohamed Adel Serhani, whose enthusiasm, motivation and immense guidance helped me to achieve this milestone. He introduced me to the stimulating field of research in e-health and their evolving technologies. The idea of further education was instilled by him when I had no clue, and he supported me throughout this journey and words are not enough to thank him.

I would like to thank my committee for their guidance, support, and assistance throughout my preparation of this thesis, especially my co-advisor Dr. Marton Gergely, who has provided me with insightful suggestions during the thesis preparation and review.

I am especially grateful to my professors, Dr. Nabeel Al-Qirim, Dr. Ezedin Barka, Dr. Mamoun Awad, Dr. Nazar Zaki, Dr. Farag Sallabi, and Dr. Elarbi Badidi in the College of IT, United Arab Emirates University, who gave more academic insights to the exciting fields of IT and Management. My special thanks to Mrs. Hadeel El-Kassabi, Dr. Mohammad Mehedy Masud, Mr. Ikbal Taleb, and Dr. Saad Harous from CIT, who have immensely motivated me. In addition, thanks to the CIT administrative staff, specially Mrs. Maryam Mandhari for her support in Arabic translation and document preparations. I am also grateful to Dr. Sujith Mathew of Zayed University, who has motivated and guided me in pursuing a career in research. I would also like to thank the co-authors of my papers, Dr. Abdelghani Benharref and Dr. Mohamed El Menshawy, for their encouragement, support and willingness to help.

Thanks to Arwa and Nayla my classmates, who are a great source of inspiration and motivation to me, and I have enjoyed each moment spent with them. Thanks to Asma Al Falasi for valuable tips in thesis writing. I would also like to thank my sole encouraging Malayali friend in CIT, Nafeen. My heart-felt thanks go to my dearest friend Sabeena, who is more like an elder sister to me, who imparted in my mind the mantra, “I can do it”.

I owe a special debt of gratitude to my mother Safiya Beevi for encouraging and instilling values in me, and my father E.M. Basheer for his sole decision to support my ambitions by letting me pursue my higher education in Information Technology. I owe a special thanks to my brothers, Alsajr and Aljaseer who are my lifelong mentors and well-wishers, and our relationship has withstood the test of time - Alhamdulillah. My sincere gratitude to the family members of Thottathil and Kodikkakom, for the tremendous support and prayers. Thanks to my energizing children, Afzal and Aliya, who believed in me and helped me along the way. I am sure they suspected it was never-ending. Last but not the least, my special and sincere thanks to my life partner, Nujum Navaz who gave me unconditional love, moral support and assistance throughout.

Dedication

To the pillars of my life:

*Almighty Allah, my supporting mother-in-law, my beloved parents, my caring
brothers, my inspiring in-laws, my cheering nieces and nephews, my loving husband,
and our understanding children.*

Table of Contents

Title	i
Declaration of Original Work	ii
Copyright	iii
Advisory Committee	iv
Approval of the Master Thesis	v
Abstract	vii
Title and Abstract (in Arabic)	viii
Acknowledgements	ix
Dedication	xi
Table of Contents	xii
List of Tables.....	xv
List of Figures	xvi
List of Abbreviations.....	xvii
Chapter 1: Introduction and Background.....	1
1.1 What is Big Data?	2
1.2 What is Mobile Big Data?	3
1.3 Big Data in Healthcare	5
1.4 Big Data Analytics	6
1.4.1 Data Acquisition	6
1.4.2 Information Extraction.....	7
1.4.3 Feature Selection.....	8
1.4.4 Predictive Modeling.....	8
1.4.5 Visualization	8
1.5 Problem Statement and Contributions	8
1.5.1 Research Problem and Questions	9
1.5.2 Objectives	9
1.5.3 Contributions	10
1.6 Thesis Outline	12
Chapter 2: Related Work.....	13
2.1 Literature Review.....	13
2.1.1 Mobile Analytics.....	13
2.1.2 Resource / Energy Aware Solutions	15
2.1.3 Mobile Offloading	17
2.2 Key Research Challenges.....	20
2.3 Mobile Big Data 5V's Challenges	23

Chapter 3: Mobile Resources Optimization Architecture	26
3.1 Architectural Requirements	26
3.1.1 Availability	26
3.1.2 On demand Scalability	26
3.1.3 Reliability	27
3.1.4 Concurrency	27
3.1.5 Privacy and Security	27
3.1.6 Maintainability	28
3.1.7 Portability	28
3.1.8 Response Time	28
3.1.9 User Experience	29
3.2 Architecture Overview	29
3.3 Architecture Components Description	30
3.3.1 Data Acquisition and Collection	30
3.3.2 Data Pre-processing	30
3.3.3 Analytics	31
3.3.4 Visualization	32
3.3.5 Smart Algorithms	32
Chapter 4: Smart Energy Saving Approach in Mobile Devices	34
4.1 Optimization Scheme	34
4.2 Mobile Resources Optimization Algorithm	36
4.3 Mobile Analytics Customization Algorithm	39
Chapter 5: Mobile Data Offloading	41
5.1 Offloading Scheme	42
5.2 Offloading Algorithm	45
Chapter 6: Implementation and Experimentation	48
6.1 Technologies	48
6.2 Dataset	48
6.3 Experimental Setup	49
6.4 Scenarios	50
6.4.1 Non-optimized Mobile Device Task Execution	52
6.4.2 Optimized Mobile Device Task Execution	53
6.4.3 Evaluation of Resources Optimization on Battery Drainage and Execution Time	55
6.4.4 Parallel Execution of Tasks for Analytics Optimization	59
6.4.5 Mobile Offloading Evaluation	62
6.5 Results and Discussion	63
Chapter 7: Conclusion and Future Work	65
7.1 Limitations and Future Research Directions	66
7.2 Closing Remarks	67
References	69

List of Publications	75
Appendix	76

List of Tables

Table 1: Experimental Findings before Optimization.....	52
Table 2: Experimental Findings after Optimization	55
Table 3: Execution Time and Energy Savings Achieved.....	58
Table 4: Serial vs Parallel Execution	59

List of Figures

Figure 1: Big Data 5V's	3
Figure 2: Big Data Analytics Platform	6
Figure 3: Popular Medical and Fitness Sensors used for Data Acquisition	7
Figure 4: Architecture for Mobile Big Data Analytics	29
Figure 5: Flowchart Diagram Smart Energy Saving Approach in Mobile Devices ..	35
Figure 6: Mobile Resources Optimization Algorithm	38
Figure 7: Mobile Analytics Customization Algorithm	40
Figure 8: Proposed Mobile Offloading Scheme	43
Figure 9: Mobile Data Offloading Algorithm	46
Figure 10: Experimental Environment.....	50
Figure 11: Application Main Menu.....	51
Figure 12: Experimental Setup to Meter Battery Resources.....	51
Figure 13: User Preferred Optimization Settings.....	53
Figure 14: Background Applications Control Interface.....	54
Figure 15: Control Interface - After Stopping the Non-Preferred Background Apps	54
Figure 16: Optimized vs Normal Battery Discharge Rate	56
Figure 17: Optimized vs Normal Execution Time	57
Figure 18: Serial vs Parallel Execution Time	60
Figure 19: Serial vs Parallel Battery Discharge Rate.....	61
Figure 20: Mobile Offloading Evaluation Phases	62

List of Abbreviations

ACR	Adaptive Clutter Reduction
BYOD	Bring Your Own Device
CAROMM	Context-Aware Real-time Open Mobile Miner
CPU	Central Processing Unit
CVA	Compression Viability Algorithm
EA	Energy Aware
ECG	Electrocardiogram
EEG	Electroencephalogram
E-health	Electronic Health
EMR	Electronic Medical Records
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HDFS	Hadoop Distributed File System
iOS	iPhone Operating System
JSON	JavaScript Object Notation
LTE	Long-Term Evolution
M-commerce	Mobile Commerce
M-health	Mobile Health
QRS	The QRS complex is a name for the combination of three of the graphical deflections seen on a typical electrocardiogram (ECG)
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
WBAN	Wireless Body Area Network

Chapter 1: Introduction and Background

The requirement of current mobile analytics needs, is outstripping the available computing power of hand-held mobile devices for data analytics. Massive amounts of data are now available at your fingertips with mobile embedded multi-sensors with connectivity features, particularly in the field of m-health. This thesis focuses on identifying the need for efficient Big data analytics techniques in mobile devices to improve health monitoring and care of patients. There are a number of challenges and opportunities to investigate techniques for collecting, analyzing and identifying trends in various health conditions such as diabetes and heart diseases in mobile devices considering the small screen size, mobility and the limited battery capacity.

Big data analytics refers to running complex computational algorithms against streams of Big data to infer actionable insights. Mobile Big data analytics refers to the detection of knowledge from raw data collected, which varies from a few dozen gigabytes to many petabytes of data collected from mobile users at the network-level or the application-level (Yazti & Krishnaswamy, 2014). The current solutions of mobile data analytics rely more on pushing the data to the cloud or remote server for analytics. The ever-increasing data which can be qualified as “Big Data” has resulted in significant research interest in Big data mobile analytics.

Results of mobile Big data analytics can provide high-level metrics and summaries using data mining techniques such as clustering, classification and association rules, suitable for decision makers. Mobile applications vendors use location-based and context-based information from the finer data collected at the

application level to provide customized solutions and relevant alerts displayed in dashboards from the analytics.

Mobile analytics can be designed while considering significant reduction of battery consumption of the mobile device being used. Analytics on data can be accomplished locally on mobile device, remotely on cloud server, or both concurrently on the mobile device and on the server. One of the key challenges for mobile device analytics is to decide on where to perform the analytics which may involve preprocessing that might include data cleansing, and selection, in addition to the application of data mining techniques (e.g. classification, clustering, association rules and regression). Mobile analytics may depend on the mobile device's operating system software like iOS (iPhone Operating System), Android and other applications which are running in the background (Zaslavsky, Jayaraman, & Krishnaswamy, 2013).

The combination of Big data and analytics is essential for innovation, especially with the advancement of medicine and its reliance on technology. Our aim is to experience the mobile Big data analytics concept in healthcare and to demonstrate the effectiveness of our proposed algorithms in reducing the mobile resources consumption in real-time analytics. In the following sub-sections, the basic concepts of mobile Big data analytics are detailed.

1.1 What is Big Data?

"Big data is high-volume, high-velocity, and/or high-variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization" (Beyer & Laney, 2012). Big data is a phrase used to mean a massive volume of both structured and unstructured data that is so large that

it is difficult to process using traditional database and software techniques (Beal, 2014). It is the term used to describe huge datasets characterized with the “5 V” definitions: volume, variety, velocity, veracity and value as depicted in Figure 1. Healthcare data (e.g. medical images, electronic medical records (EMR), and biometrics data) is divided into structured and unstructured data in a range of formats and distributed in silos owned by a multitude of stakeholders. This huge data can add value and improve quality of healthcare through innovative analysis as well as improving patient care. This gigantic size of healthcare data will need large amounts of computation which can be done with the help of distributed processing supported through cloud data centers and distributed clusters of Big data.

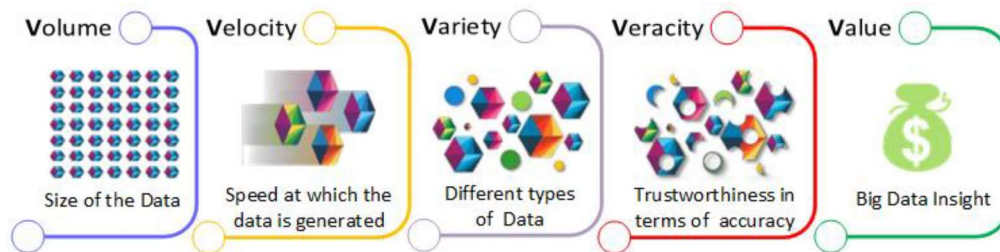


Figure 1: Big Data 5V's

Lately, there are five more additional important characteristics which start with v as well, forming the 10 Vs of Big data (George Firican, 2017). They are Variability, Validity, Vulnerability, Volatility and Visualization.

1.2 What is Mobile Big Data?

Mobile Big data is defined as a large dataset generated by mobile technology (e.g.: devices, applications, and computing) with or without a combination of structured, semi-structured, and unstructured data generated/analyzed in real-time or

offline, which cannot be managed/analyzed by traditional database management methodologies (WorldBank & InfoDev, 2012). Mobile Big data will gain significance in the coming future, as the usage of mobile technology is at a tremendous pace, thus bringing more challenges than answers.

Mobile Big data is huge (volume): Mobile data is the biggest contributor of Big data due to the excessive penetration of mobile devices in our life and is expected to increase in the coming years. Multitude of mobile applications and sensors send and receive data, on a daily basis that gigabytes and terabytes of storage space, get filled in frequently, thus requiring mobile users to often free-up space of their mobile devices.

Mobile Big data is generated proactively and rigorously (velocity): Mobile devices are portable which makes the available time for decision making very short. Hence mobile data analytics should be executed frequently with the collected data samples at the highest speed. This is common in the case of mobile health monitoring using sensors, where ECG or EEG data is sensed at tremendous speed.

Mobile Big data is heterogeneous (variety): The rich variety of data generated on the move includes everything from network geo-location, call records, message records, billing records, application data social media and much more. Also, data comes from various sources, built in and connected sensors in a variety of formats.

Mobile Big data analytics (value): Having received the huge, varied mobile device sensed Big data alone is of no value. We might need to further analyze the Big data and reveal the facts which we didn't know. Data value is mostly about mining knowledge and patterns from mobile Big data which can provide deep insights (Alsheikh, Niyato, Lin, Tan, & Han, 2016).

1.3 Big Data in Healthcare

Electronic medical records (EMR) created from all types of patients primarily from hospitals, clinics, insurance service providers and doctors generate huge data streams. A single patient stay at a multi-specialty hospital generates thousands of metadata like registration details, lab tests, lab reports, medical imaging, diagnoses, medical procedures, medications, medical supplies, insurance details and billing. These metadata need to be verified, processed, and analyzed to retrieve meaningful insights. Growing with all the outpatient and inpatient staying 24/7, 365 days across all the e-health processing systems and combining it globally reveals the scope of the Big health data challenge (Liu & Park, 2014). Other than the healthcare facilities like hospitals, external health data is generated by patients including social media, self-wearable sensors and smartphones. These devices record pulse, temperature, brain activity, sleep patterns, position, ECG and various metadata from biological body parts as well as the GPS locations that are exchanged with associated parties of insurance, physicians and medical technicians.

The data captured and gathered from mobile sensors remain vastly underutilized and thus wasted. Mobile Big data analytics might be helpful for patients and physicians on the move. Customized metrics, reports and summaries can be made available at their finger-tips and even can provide alerts based on the analytics.

Mobile continuous monitoring is evolving with the emergence of smart wearable medical devices and will continue to grow and improve healthcare with better predictive modeling which is termed the 3M strategy for health: Monitor–Measure–Manage (Schatz, 2015).

1.4 Big Data Analytics

Figure 2, inspired from (J. Sun & Reddy, 2013), depicts a comprehensive platform of Big data analytics in healthcare. It consists of a set of activities including data collection, information extraction, feature selection, predictive modeling, and data visualization.

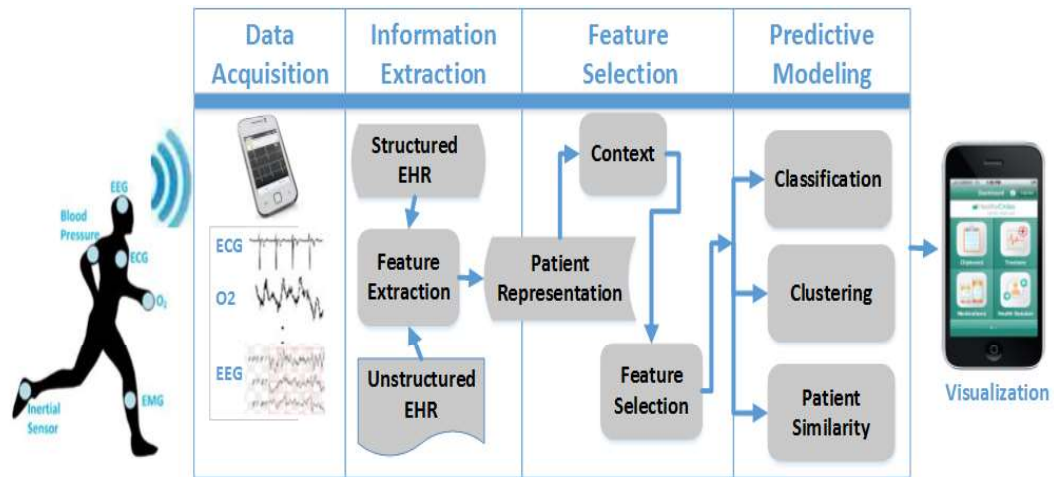


Figure 2: Big Data Analytics Platform

In the following sub-sections, we describe each of these activities and the relationship between them.

1.4.1 Data Acquisition

Health data may pertain to various diseases such as diabetes or cardiovascular diseases, which includes vital signs such as heart rate, blood pressure and physiological signals like ECG and EEG. *Sensors*—accelerometers, location detection, wireless connectivity and cameras, offer a big step towards closing the feedback loop in personalized medicine. There are many providers of body sensors in the market as

depicted in Figure 3 (Bonnie Feldman, 2013). Health signals are continuously captured from on-the-body or in-the-body sensors and thus acquired by the mobile device.

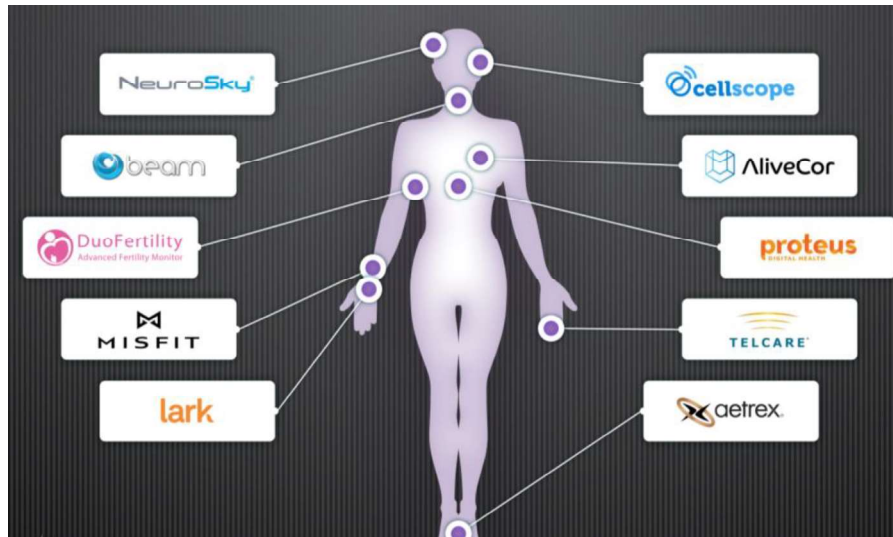


Figure 3: Popular Medical and Fitness Sensors used for Data Acquisition

1.4.2 Information Extraction

The data acquired from sensors might be structured or unstructured, and can be combined with available EHR (Electronic Health Records) data from hospitals. Furthermore, the data collected may be pre-processed to extract relevant information also called feature extraction. This includes preprocessing (i.e. filtering out unwanted data and noise in case of signals) and further processing, for instance feature extraction such as QRS detection in case of ECG. Here the P wave, QRS complex and the T wave represent electrical activity in the heart on an electrocardiogram. The QRS complex follows the P wave and depicts the activation of the right and left ventricles. The detection of QRS complex, is the first step of information extraction to determine cardiovascular diseases.

1.4.3 Feature Selection

The filtered data which identifies the patient is further processed by selecting a subset of relevant features (variables, predictors) with respect to the given healthcare context (e.g. seizure detection in an EEG signal data) to be used to construct the classification/prediction model.

1.4.4 Predictive Modeling

Predictive analysis of healthcare data is carried out using data mining tools and can forecast probabilities, trends or patterns. Each predictive model is made up of several predictors that are variables, likely to influence the prediction. A statistical model is formulated once relevant data is collected.

1.4.5 Visualization

The outcome or insight learned from predictive Big data analytics derives its meaning from visualizations, such as time series charts or dynamic cross-tabs, which are valuable for decision making by healthcare providers.

1.5 Problem Statement and Contributions

Mobile resources are limited and need to be minimized and controlled, which is a vital requirement to help optimize its utilization for mobile data analytics.

1.5.1 Research Problem and Questions

In mobile analytics, one of the most valued and a scarce resource is battery life. In an effort to optimize battery resources for mobile analytics, the following research questions are formulated which needs to be addressed:

- How do we evaluate mobile devices resource requirement for analytics tasks?
- How do we address mobile resource scarcity?
- Where is it best to handle mobile analytics: - the mobile device, server/cloud, or concurrently both on the mobile device and on the server?
- How do you filter out non-useful or redundant data in mobile e-health monitoring where ECG or EEG signals are captured?

1.5.2 Objectives

The objective of the thesis is to evaluate the resource consumptions for each task on mobile devices and to devise activity and context aware intelligent distribution of Big data analytics between mobile and server/cloud. The research aims to provide healthcare decision makers with the advancements in the mobile Big data computing, in effectively handling large and heterogeneous health datasets.

A situation where a patient critically needs to do analytics on mobile device, but is not connected to the internet, there is a necessity to do the analytics locally. If the patient is connected, the data may be transferred to the cloud or server for deep analytics. Similarly, a variety of combination scenarios can be identified, and a few are listed which will be considered in the scope of the study.

Scenario 1: Resource constrained environment.

Objective: Achieve optimization of resource utilization.

Resource aware analytics is to evaluate mobile resources dynamically to decide on the fly the tasks that can be executed with the resources at hand. This could be achieved with smart energy saving algorithms.

Scenario 2: Analytics customization approach on mobile device.

Objective: Context and data-aware analytics.

Context and data-aware analytics is defined to handle the minimum analytics while assuring results with an optimum accuracy. Few tactics will be considered such as data partition, compression, and parallel execution (Masud, Serhani, & Navaz, 2016).

Scenario 3: Decision on offloading to server/cloud

Objective: Evaluate when and what to offload.

Offloading tasks and data to the cloud/server depending on criticality of the process and the available resources.

1.5.3 Contributions

This sub-section briefly describes the main contributions we propose to handle mobile Big data analytics resource optimizations in m-health.

Contribution 1: Develop a smart metering application to analyze and optimize mobile resources.

The mobile device capabilities are hardware, software and network components. Hardware components are comprised of the CPU, sensors, memory, and battery. The software components consist of the Operating System (OS) and applications. Lastly the network components are Wi-Fi, Bluetooth, NFC, and GPS. An Android mobile application is developed to determine the capabilities of the mobile

device, analyze, and meter resource requirements/consumptions of current running applications and services. The measured mobile resources are CPU, battery and memory. Few Big data tasks, mostly in pre-processing, processing and analytics will be executed and the mobile resources will be metered on a continuous basis and energy consumptions will be assessed. The application will serve the resource saving and optimization, and efficiently handle mobile data analytics activities.

Contribution 2: Implement smart energy saving algorithm to handle mobile resources.

The smart energy saving algorithm should dynamically decide (context-aware) on closing/pausing background or unessential applications or services, in accordance with continuous metered resources (resource-aware), and scale to the Big data resource needs. Also, data reduction/sampling (data-aware) techniques will be incorporated to process and diagnose health data, for further reduction in resource consumption.

Contribution 3: Develop mobile offloading algorithm.

The key decision on where to accomplish the analytics which includes the pre-processing, processing and visualization would be our third contribution. Mobile offloading for instance can increase mobile battery/resources life (Loomba et al., 2015). It should occur only when it is essential, (i.e. when the resources are scarce or when Big data is too large and technically impossible to process on the mobile device). A mobile offloading algorithm can help in decision making (i.e.: when to offload processes to a backend server or a cloud data center), and is part of the proposed design. In (Kchaou, Kechaou, & Alimi, 2015), the authors suggest to address the

questions: “what to offload, when to offload and how to offload” in dealing with mobile offloading. In our offloading approach, we have attempted to answer these questions, by proposing an offloading algorithm. The applications may have different performance requirements and mobile devices may have diverse capabilities and energy concerns. Offloading decisions need to be made based on a set of target goals, such as a combination of improving performance, saving energy, managing storage, and reducing the network overhead.

In the following, we summarize the key contributions of this research.

- Propose smart context-, resource-, and data-aware energy saving algorithms for Big data mobile analytics.
- Propose a mobile offloading algorithm for intelligent decision making on local or cloud/server execution.
- Experimental implementation of the proposed algorithms, to be developed and deployed on a mobile device.
- Evaluate the energy savings resulted from the execution of the above algorithms.

1.6 Thesis Outline

The rest of the thesis is organized as follows: In chapter 2, a literature review of relevant research works is presented. Chapter 3 provides the overall mobile analytics optimization architecture. Chapter 4 details the mobile energy saving approach while chapter 5 describes mobile offloading scheme with the algorithm, both of which can be adapted to any organization performing mobile Big data processing. Chapter 6 details the implementation and experimentation on the Android platform. Finally, chapter 7 culminates the thesis with future research directions.

Chapter 2: Related Work

In this chapter, we summarize the existing work related to our study and identify the key research challenges as well as the Big data challenges that arise.

2.1 Literature Review

Mobile data analytics involve a set of tools which enables the processing, analysis and visualization of mobile data collected from various sources like social media. This section surveys the literature on mobile analytics, resource and energy-aware solutions for mobile analytics, and mobile offloading approaches.

2.1.1 Mobile Analytics

The applications which can leverage the usage of mobile analytics can be classified as mobile crowd-sourcing or crowd-sensing. These crowd-sourcing applications depend on multiple sensors integrated within smart phones to collect data from a very large group of mobile users and provide contextual information by giving priority to users' operating requirements. It requires continuous sensing, processing and uploading or transferring sensed/processed data to the cloud or remote servers. These energy expensive functions may in certain cases lead to battery depletion (Zaslavsky et al., 2013).

In 2010, Nokia conducted a data collection campaign wherein Nokia N95 phones were allocated to a heterogeneous sample of nearly 170 participants from the city of Lausanne to be used over a period of one year (Kiukkonen, Blom, Dousse, Gatica-Perez, & Laurila, 2010). The data collection software runs on the background and collects data on various aspects such as social interaction and spatial behavior. It

led to a new kind of research called people-centric sensing where sensors in mobile phones and other wireless devices are used to collect large quantities of continuous measurements about users which tracked their behaviors ranging from intra-personal to inter-personal level. People-centric sensing can lead to a qualitatively new type of approach in studying Big data. It can be further categorized into three: personal, social and public sensing. Personal sensing focuses on personal monitoring. Social sensing emphasizes on sharing data within social and special interest groups. Finally, public sensing focuses on sharing data with everyone promoting the greater public reach (Y. Sun, Song, Jara, & Bie, 2016).

In (Zaslavsky et al., 2013), mobile analytics is classified into push-based independent systems with local sensing and processing, push/pull-based independent/collaborated systems with local sensing and cloud processing/storage and push/pull-based collaborated system with distributed processing and load balancing between cloud and mobile device. Also, they propose an architecture named CAROMM which is an approach to collect and process data streams from a large number of mobile devices with the concept of crowd sensing and using mobile analytics while on the move. The experimental evaluation claims a significant achievement in energy and bandwidth savings.

(Castro et al., 2014) present a collaborative extension to InCense, a mobile phone sensing toolkit to enable behavioral data gathering from populations of mobile phone users during mobile phone sensing campaigns. In collaborative sensing, they incorporated several strategies aimed at optimizing battery, storage, and bandwidth and used in decision making on when to capture audio when many devices are recording a similar audio. Open Mobile Miner (Haghighi et al., 2013), a mining tool

for the mobile that has resource-aware, situation-aware and with hybrid strategies and the adaptation is performed hidden from the user.

A Spark-based framework (Alsheikh et al., 2016) for learning deep models was proposed for mobile data analytics within large-scale mobile systems. Basically, the Spark engine tackles the volume aspect of mobile Big data by parallelizing the learning task into several sub-tasks, but the paper doesn't mention any analytics being done at the mobile side. A backend cloud server is used for analytics as deep model learning in mobile data analytics is sluggish, demanding and resource consuming.

Dated as far as 2002, researchers on mobile data mining have initiated MobiMine (Kargupta et al., 2002). It is a distributed client-server data mining environment that allows smart monitoring of stock market data from mobile devices. The MobiMine server and client apply numerous advanced data mining techniques including clustering, Bayesian nets and decision trees for any time anywhere monitoring.

2.1.2 Resource / Energy Aware Solutions

In mobile data mining the device can play the part of data producer, data analyzer, client of remote data miners or combinations. They can be used as terminals for continuous access to a remote server that runs major data mining services, where the server analyzes data and provides the results of the data mining task to the mobile device for further visualization. Data generated in a mobile context are congregated through mobile devices and transferred to a remote server. Due to memory/storage and energy constraints, heavy data mining tasks on mobile devices may not be performed. However, light and energy efficient data mining algorithms that analyze minor datasets

can be successfully executed on mobile devices. Distributed task allocation strategy (Comito, Falcone, Talia, & Trunfio, 2013) was suggested to deal with a set of independent data mining tasks which needs to be allocated and scheduled over mobile nodes organized into the cluster-based architecture where devices cooperate in a peer-to-peer style to perform data mining tasks. EA (Energy-Aware) scheduler (Comito et al., 2013) needs to know in prior, the estimated memory (EMC) and energy consumption (EEC) of the device candidate to run a task. Given a task and size of the dataset to be analyzed, the EMC and EEC values are obtained based on memory and energy consumption measurements performed on that device.

Big data in cellular networks, distributed across different domains i.e. in space, time, codes, and antennas also require efficient algorithms for solving large-scale mobile cellular data. Through Big data analytics for cellular networks, mobile operators can enhance customers' loyalty, mobile cellular services, reduce expenditure and increase revenue (He et al., 2016).

In (Gaber et al., 2013), real-time visualization for mobile data mining – ACR (Adaptive Clutter Reduction), which is relevant in case where visualization is for a continuous, rapid, and dynamically changing situation such as monitoring heart-patient ECGs is presented. Here DataStream mining algorithms with built in mechanisms for resource-aware strategies for energy management were adapted. These algorithms enabled dynamic adjustment and refreshes the mobile backlight according to the resource levels availability.

System architecture to profile the resources usage in terms of energy consumption for cloud, which can facilitate app developers in creating energy aware software, was suggested in (Alzamil, Djemame, Armstrong, & Kavanagh, 2015). The

current mobile programming platforms also lack factors to support developers in building energy efficient applications, which are inevitable in mobile computing and analytics. To address this, a new concept called Symbolic Execution and Energy Profiles (SEEP) (Hönig, Eibel, Kapitza, & Schröder-preikschat, 2011) which utilizes symbolic execution and platform-specific energy profiles for energy-aware programming was suggested. The major limitation of symbolical execution was that every possible program paths evaluation was tedious when it comes to vast programs spanning several conditions and loops.

2.1.3 Mobile Offloading

Collaborative sensing middleware to aggregate sensed information from multiple mobiles within a particular physical area was suggested in (Castro et al., 2014; Loomba et al., 2015). This offloading of sensed data, which serves multiple applications, can lead to energy savings. The ‘Info-Aggregation’ algorithm, applies pattern mining to determine energy efficiency with respect to the volume of offloaded mobile data.

Cloud offloading is a prevalent energy optimization technique where mobile application’s energy-intensive functionality is transferred to a cloud based server. In (Sarithchandra Magurawalage, Yang, Hu, & Zhang, 2014) explains a model that uses a concept of virtual machine that runs on trusted and resource-rich computer, or a cluster of computers named cloudlet. A cloudlet is a trusted, resource-rich computer or cluster of computers that’s well-connected to the Internet and available for use by nearby mobile devices (Satyanarayanan, Bahl, Cáceres, & Davies, 2009) . According to authors in (Benkhelifa, Welsh, Tawalbeh, Jararweh, & Basalamah, 2015; Quwaider,

Jararweh, Al-Alyyoub, & Duwairi, 2015) by using resource-rich cloudlets, it is possible to bring the cloud closer, rather than depending on the distant cloud. The offloading algorithm in (Sarithchandra Magurawalage et al., 2014), takes into consideration the energy consumption for task execution and the network status for decision making. However, the limitation is that, cloudlets built on VM (virtual machine) technology should be physically present near the mobile device, which restricts mobility. In a characteristic mobile application, Wi-Fi and network communications significantly consumes energy, mostly during remote data transfer.

Cloudlets perform the role of intermediaries between users and the cloud and facilitate communication and offloading. In (Routaib, Badidi, Elmachkour, Sabir, & Elkoutbi, 2014), it is recommended to have a centralized cloudlet-based architecture in which, mobile users are connected to their closest cloudlets through wireless network. However, they suggest that when the number of cloudlets is small and limited to less than 10, mobile users encounter difficulty in delivering requests to their suitable cloudlets due to the time taken for establishing a network connection with cloudlets; the registration of mobile user's information in user's catalog; and the present state of cloudlets.

In (Benkhelifa et al., 2015), a user/application profiling system for mobile devices is proposed for efficient resource augmentation from remote cloud resources. One of the components is a data logger which will log time series data of each application's usage of individual hardware components and therefore its overall energy drain. These individual records are built up to produce a set of usage profiles for each application. One of the challenges is profile creep, whereby a user's habits alter over time, and the models' accuracy in predicting energy usage may become affected.

Elastic HTML5 (Zhang & Jeon, 2012) enables web applications to offload workload using cloud based web workers and cloud-based storage for mobile devices. It is a collection of software components and functions to create and manage web workers in the cloud so as to extend the computation functionality of a browser-based application running on a device.

In Cuckoo (Kemp, Palmer, Kielmann, & Bal, 2012), a complete framework for computation offloading for Android was suggested, which includes a runtime system, a resource manager application and a programming model for developers.

To the best of our knowledge and considering the above-mentioned literature review there is no all-inclusive solutions that consider the following:

- Evaluate mobile Big data analytics requirements.
 - Assess mobile device capabilities to execute Big data analytics.
 - Assess the energy/ CPU requirements of running applications.
- Consider user preferences and profile.
 - Eliminates profile Creep.
 - User can override the settings required for performing the analytics task.
- Address the energy constraints in mobile devices while handling Big data.
 - Develop smart energy saving algorithms to effectively stop/pause dispensable applications processes and services.
 - Develop data reduction techniques to effectively process data.
 - Set dynamic battery threshold event which triggers, prioritizes the current running tasks and save the instance state of the application.
- Establish when to offload mobile data analytics to cloud/server.

- Develop mobile offloading algorithm, which considers analytics segmentation, monitored resources, offloading options and user preferences.
- Evaluate Compression viability before offloading.

2.2 Key Research Challenges

The availability of wireless networks and their data transmission rates vary according to the present location of the mobile device. Because of mobility, device users may lose connectivity with a cloud or a server and might or might not reconnect for a long time. The offloaded data and the processed data should not be lost during provision of a seamless service (Sarathchandra Magurawalage et al., 2014). The following are the research related challenges identified from surveying the literature on the related fields.

How to process Big data in mobile devices

A challenge in analyzing Big data is to obtain a quick inference in real-time from various sources of data (e.g.: mobile surveillance and context-aware navigation). From an implementation perspective, processing Big data is usually linked to parallel programming technologies like MapReduce. However, in a mobile environment there are only few tools available for Big data processing. With the continuous real-time data streams, mobile devices require capabilities for iterative computations, which depend on sophisticated models of data caching and in-memory computation. Machine-learning-based data analytics need specific fine-tuning in learning a classifier over large scale datasets. The learning procedures are comprised of learning a set of typical parameters that need to be found by means of feature selection and

optimization. There is a lack of Big data processing tools with machine learning capabilities in the mobile environment.

How to maintain privacy and security of Big data, when stored and transmitted

Mobile application security, which includes information confidentiality, integrity, and availability, is a major concern in enterprise mobile applications (Unhelkar & Murugesan, 2010). Offloading healthcare organizations' Big data elevates extra challenges in relation to privacy, security and ownership of data. These data may consist of personal and private user data of patients, physicians, employees, customers and stakeholders, which are being generated and transferred at a terrifically fast velocity in the connected world of intelligent devices, sensors, networks, and software / web / cloud applications. Often these datasets are not stored at a centralized location (i.e.: the traditional way), but distributed over various servers and networks, thus facing a risk from data attacks as well as third party data misuse. Evidently, many types of health related sensitive data cannot be stored in the mobile device or cloud without considering these privacy and security implications. Furthermore, communication through wireless and mobile networks, are more vulnerable to third party attacks than in wired networks. Hence, data security needs to be ensured at all points of mobile and analytics offloading scenarios.

How to implement mobile sensing

In healthcare, smartphone-embedded sensors with context-aware and location aware applications and services can be a source of Big data. Many enterprise applications for smart phones are mostly available in the two flavors - iOS and Android platforms. The challenge here is to devise real-time multiple sensing applications which provides high accuracy and low power consumption.

What are the Big data storage options and how are they managed?

A major mobile computing challenge is the cost of storing health data on the internal memory of a smart-phone, and the transmission costs when large amounts of data are involved. Data storage hosted on a single cloud or server may not be advisable in unforeseen circumstances, which might prompt to back up the data with an alternate service provider, which may increase mobile power consumption (Kumar & Lu, 2010).

How reliable is mobile device data with respect to mobility?

Another potential concern with mobile cloud computing is reliability. A mobile user performing computation in the cloud depends on the network connectivity to 3G, 4G, Wi-Fi, Bluetooth or ZigBee. Mobile device users are always on the move, and they are fully dependent on these networks which make computing on the cloud not possible in case of limited connectivity or complete disconnection. Mobile cloud computing is also difficult to maintain in some locations such as the basement of a building, interior of a tunnel, or subway. Dependence on the cloud for important computations could lead to problems during service outages (Kumar & Lu, 2010).

How to formulate an offloading decision with respect to the context?

The main challenge of the offloading decision is context adaptation with respect to the available offloading options: offload to a server, a cloud or to other devices. It is more complicated when the numbers of options are numerous and the nature of surrogate devices are heterogeneous. Hence, the offloading decision needs numerous aspects, most importantly the mobility pattern of the user and the accessibility medium which adopt the bandwidth of the available surrogates (Orsini, Bade, & Lamersdorf, 2016).

Is Mobile Cloud Computing interface friendly?

Interoperability is a critical factor when a mobile device communicates with a server and in particular, a cloud environment. The web interfaces of cloud environments are not designed for mobile interaction and hence the overhead is higher (Debashis De, 2015). Also, compatibility of different mobile OS's, with cloud interfaces is a concern. Standards for interacting between mobile devices and cloud servers would be required.

How to predict and manage mobile/cloud resources

Mobile/Cloud resources should be allocated in an on-demand fashion as mobile analytics progress. As it is difficult to correctly predict the resource needs before execution, unnecessary resource utilization may happen before actual executions happen (Debashis De, 2015) .

These observed challenges motivate us to further study mobile analytics of Big data and propose solutions to cope with theses.

2.3 Mobile Big Data 5V's Challenges

Big data has five characteristics: volume, velocity, variety, veracity and value. Big data analysis requirements are methods used to find meaning and discover unseen relationships in Big data (Chawda & Thakur, 2016) .

Volume: If the Big data analytics requirement needs to store data in mobile devices or transmit to cloud/server, the device and cloud/server requires adequate storage space. Also, intermediary results, models and rules of preprocessing and processing may require the need of being stored. Volume scarcity can be addressed by

data archiving or compression. Also, minimal data storage can be made possible by filtering out unwanted data while extraction.

Velocity: The challenge here is to react on time to handle the velocity of the incoming stream of data. It adds stress to the volume requirement and also the timeliness factor (i.e. to trigger the reaction at the appropriate time).

Variety: It refers to handling data from heterogeneous data sources. Unstructured and semi structured data may need to be enforced with structure to process further. At this point mapping data extracted to high level database schemas may be necessary. Also, integrating many data sources and metadata may be required. Again, handling data variety to extract information implies storage requirement to be handled. To work with data, it is essential to know where it comes from, how it is recorded and in which structure it is stored. Metadata management can handle the uncertainty of the data, as it gives clues to the overview of the data and its semantics.

Veracity: Great insights need quality data. Improving the data quality means correcting the incorrect values and completing the missing ones. There are Big data cleaning techniques such as Talend (“Data Quality Management: Talend Enterprise Data Quality Tools,” 2017) to assure data quality. Standard quality dimensions are timeliness, accuracy, completeness, and consistency. There is a need to evaluate the quality of the data and also the quality of service (process) of data handling, at each stage namely pre-processing, processing, analytics and visualization (Serhani, Kassabi, Taleb, & Nujum, 2016).

Value: Value generated from data mainly gets shaped from the data analytics tasks performed over the data. Visualization techniques can be used to display the value of the data as interactive dashboards and reports. These helps to visualize the

insight or value derived and are very useful or even critical for decision makers. This requires appropriate programming languages and frameworks to support the analytics tasks and visualization.

Handling the volume, velocity, variety, veracity and value all depends on the particular process and scenario of the analytics requirement.

Chapter 3: Mobile Resources Optimization Architecture

In this chapter, we present an architecture developed to support Big data analytics in mobile devices. This architecture provides a comprehensive view of the proposed contributions from a healthcare perspective. Also, we detail the architectural requirements to help us, in understanding the problem, and design and develop the optimization architecture.

3.1 Architectural Requirements

The proposed architecture exhibits some architectural requirements, which are detailed below.

3.1.1 Availability

Big data is accumulated in real-time and at a rapid pace. They are useful only if the insights are available to the people and processes that need them, when they need them. People can visualize and analyze data in a real-world context to see the effect of changes immediately so they can make immediate and smarter decisions.

3.1.2 On demand Scalability

Big data volume storage requirements put pressure on the scalability requirement, as the system needs to be more scalable with increasing demand of storage and processing. Also, continuous sensing, demands scalability as the system needs to regulate with the incoming sensed data while at the same time do the storage and processing of the received data. The system should scale in such a way that it

allows adding additional hardware resources, to keep the analytics continuing as the incoming data grows.

3.1.3 Reliability

If some of the tasks fail due to non-availability of data, network or computing resources the system should know how to continue with the computation, by re-assigning or resuming the incomplete work once it is available. Appropriate logs need to be written, which can help to resume the task. Also, the user needs to be notified of any critical task failures. Different industries, especially the healthcare sector's dependence on the availability and reliability of information can be a matter of life and death.

3.1.4 Concurrency

The analytics tasks should follow one of the principles of functional programming to avoid update lockups. It also should avoid the need to coordinate multiple processes writing to the same file. These should be taken care of by the programmer/developer of the system.

3.1.5 Privacy and Security

Any information stored, especially health data, should have restricted access and preferably defined roles and access levels for the different tasks involved. So, without the right security and encryption solution in place, m-Health Big data can mean big problems. In Big data environments, data is routinely replicated and migrated among a large number of nodes. In addition, sensitive information can be stored in system logs, configuration files, disk caches, error logs, and so on. We can secure Big

data environments with different technologies such as Vormetric (Cloud et al., 2015), which protects data across all these areas, delivering encryption, privileged user access control, and security intelligence.

3.1.6 Maintainability

The huge volumes of mobile Big data, which can be collected and analyzed 24 hours a day, seven days a week, can help users identify looming faults, forecast the optimal time for maintenance and, ultimately, enable them to predict conditions that could become a hurdle in the future and tackle them early. Therefore, regular routine backup and maintenance like freeing up storage space should be taken care before daily services are affected.

3.1.7 Portability

Easier data portability and data administration needs to be made available so that users can have confidence that their data is current and consistent. The system should have the ability to move data among different tasks, computing environments or cloud services.

3.1.8 Response Time

Response time depends on several factors (e.g. network state, processing time, carrier/Wi-Fi latency, locations). In optimal mobile app design, the response time should be kept less than 1000 milliseconds. However, rather than depending on the industry standards, it is advisable to find the response time that is acceptable for the user, possible for the application process and matches the criticality of the service.

3.1.9 User Experience

There are IT measures which focus on how the user is experiencing the app (App Dynamics, 2015). One of the measures is application load per period, which is related to the number of transactions over a period of time. This is to make sure that as the load increases, the application performance and the user experience doesn't degrade. The mobile application should be user friendly and tested to eliminate app crashes, bugs and network errors for a seamless user experience.

3.2 Architecture Overview

Figure 4 illustrates the proposed scalable architecture of Big data analytics in mobile devices. It starts with data collection using various health monitoring sensors, pre-processing, analytics, and visualizing the value realized from the Big data analytics. The architecture aims to improve battery performance with the optimizations performed at the mobile device.



Figure 4: Architecture for Mobile Big Data Analytics

3.3 Architecture Components Description

In this section, we describe the key processes and the role of each of the components of the above architecture.

3.3.1 Data Acquisition and Collection

This is the first phase, where patient data is collected from various data sources. It may include sensed data (i.e.: heart rate, temperature, blood pressure, blood glucose, ECG, EEG, GPS) using health sensors which form a WBAN (Wireless Body Area Network). In addition, to these sensed data, various other patient related data (i.e.: lab reports, radiographic digital images, weight, medicine, diagnosis, demographics, food intake, sleep pattern and historical data) may be collected in the mobile device using mobile communication technologies. Most of these data may be extracted from online databases, using various query parameters and retrieved by a structured means. However, many data (i.e. images, audio, and video) from heterogeneous data-sources may be unstructured or semi-structured. This vast variety of patient data with the relevant metadata, complies with the Big data 5V's requirement, and satisfies one or more of the characteristics (i.e.: Volume, Velocity, Variety, Veracity, and Value).

3.3.2 Data Pre-processing

If humans are involved in any of the data collection process, there is always high chance of some errors or inconsistencies. However, when humans deal with information, a great deal of heterogeneity is comfortably tolerated, while machine

analytics expect homogeneous data, and cannot understand natural language. In consequence, data must be carefully structured prior to data analytics.

Often, the data collected will not be in a format ready for analytics. The original data format might require an extraction process to filter the required information and express it in a structured form suitable for analysis (Alexandros Labrinidis & H. V. Jagadish, 2012). This phase deals with the pre-processing which includes filtering and data cleansing. Data cleansing consists of handling imprecise, unreliable, ambiguous or uncertain data. In the traditional approach, before the next stage (i.e. the analytics), data quality and trust need to be ensured. However, when dealing with Big data it is not often feasible to ensure data accuracy, especially when velocity requirements are present. Also, information extraction and integration from unstructured data with diverse variety and volume, it is difficult to maintain data quality, even after data cleansing. Hence it is necessary to use Big data technologies such as Talend (Swanson, 2016) to manage Big data in context of noise, heterogeneity, and uncertainty.

3.3.3 Analytics

As noted earlier, Big data is mostly noisy, dynamic, heterogeneous and interrelated. It could be made valuable by identifying frequent patterns and correlations and disclose the hidden patterns and knowledge. Methods for querying and mining Big data are profoundly different from traditional analysis on small samples. One of the challenges of interconnected Big data is data redundancy and need techniques like feature selection. Feature selection is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. In the Big data scenario,

analytics consist of mining large amount of data, multivariate statistical analysis and multi-scenario analysis and simulation executed for a long-term period, often continuously and repeatedly.

3.3.4 Visualization

The value of Big data analytics can be completely realized with proper visualization. Knowledge and insight realized can be presented using various views including summary of patient health related findings, monitored results, diagnosis, and recommendations as graphs, patterns, and tables. These dashboards and indicators may help the health-care providers and patients to make timely decisions. However, one of the biggest challenges in mobile device visualization is the limited screen size to present the information/insights in a clear and appropriate manner.

3.3.5 Smart Algorithms

Big data analytics in mobile device is not only about the infrastructure, but also about algorithms and tools on applications that are used to analyze and process the data. We propose three smart algorithms with the aim to facilitate the mobile analytics and minimizing the energy usage of the mobile devices. We intend to incorporate smartness, by automating decisions to reduce energy usage, customize process, and offload data and tasks to the cloud/server. The proposed mobile device smart algorithms consist of the following:

- Resources optimization
- Analytics customization
- Offloading optimization

The resources optimization algorithm applies to the four phases mentioned above namely data collection, pre-processing, analytics and visualization. The analytics customization algorithm is applicable only in the analytics phase. However, the offloading algorithm may be applied in pre-processing and analytics phase depending on the decision, when and where to offload few or whole processing tasks to a cloud/server. The following chapter provides more details on these smart algorithms and energy saving approaches.

Chapter 4: Smart Energy Saving Approach in Mobile Devices

In this chapter, we describe in detail the first two contributions in mobile device energy saving approach, specifically the Resource Optimization and Analytics Customization approach. The optimization scheme and the algorithms are provided in detail.

4.1 Optimization Scheme

A flowchart diagram is illustrated in Figure 5, which combines the resource optimization and analytics customization approach to save mobile device battery life. It starts with the local (i.e. mobile device) execution of tasks in the pre-processing or analytics phases. If mobile device applications which are not important to the process/tasks are running in the background, they are closed or paused. Network connectivity, with data transfers is a major factor in battery consumption. 3G and 4G consume significantly more energy to transfer data of all file sizes in comparison with GSM and Wi-Fi. Wi-Fi is more energy efficient than cellular networks once it is connected to an access point. All the network connectivity's, incur an upkeep energy to keep the interface up (N.Balasubramanian, A.Balasubramanian, & Venkataramani, 2009). If the analytics tasks do not require network connectivity, the non-required connectivity is switched off. Similarly, the location data is limited, if it is no longer required, by switching off the GPS. A sensible way to reduce the energy consumption in mobile devices is to dim the backlight (Lin, Hsiu, & Hsieh, 2014). Dimming the backlight, of the mobile device is one of the last options to save further energy.

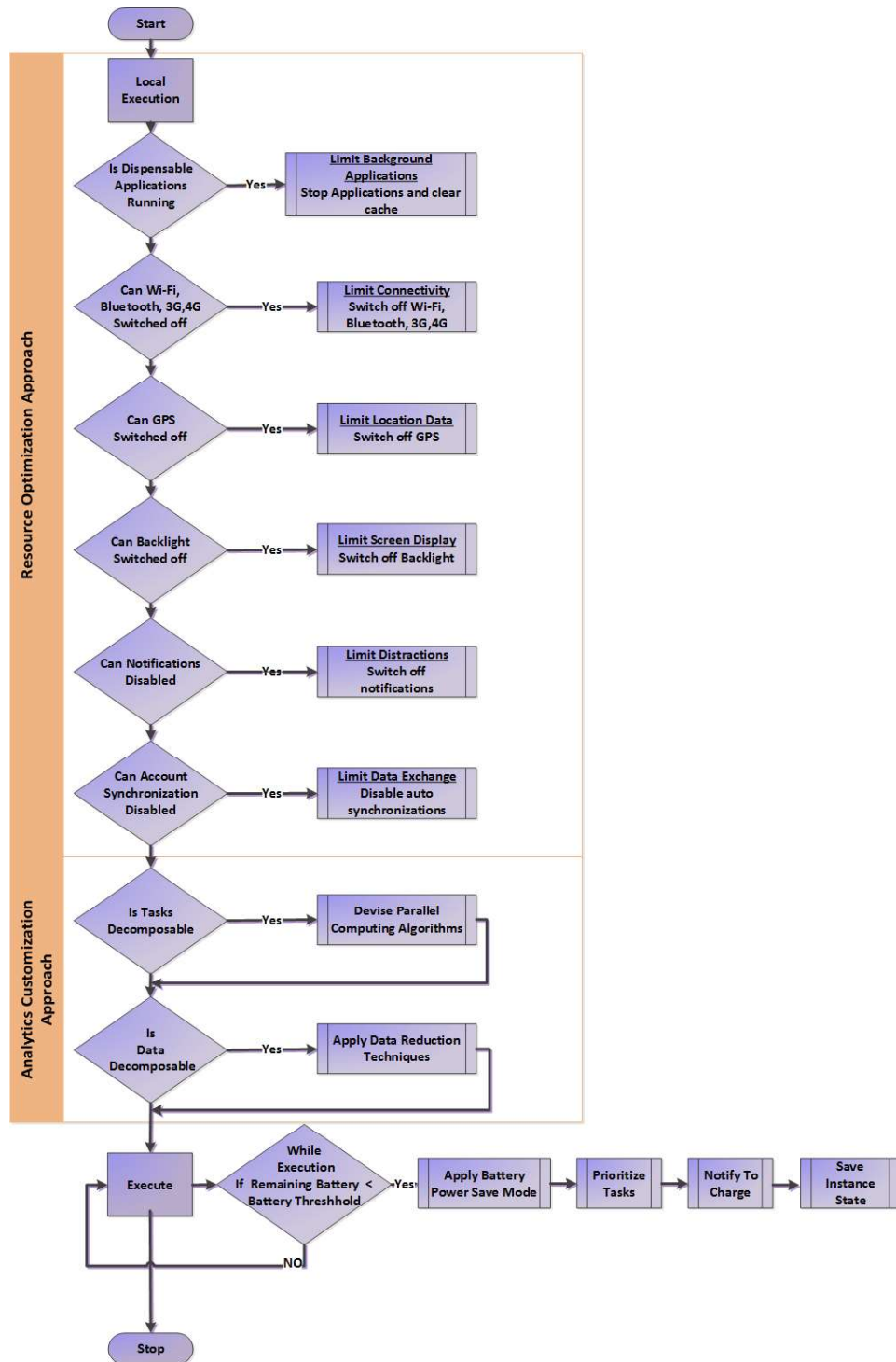


Figure 5: Flowchart Diagram Smart Energy Saving Approach in Mobile Devices

Analytics Customization approach is applied in the analytics phase. Tasks can be of two types: atomic which cannot be decomposed, and decomposable which consists of a lists of sub-tasks (Scerri, Vincent, & Mailler, 2006). The first phase is to check if tasks are decomposable or not and if it is, split the tasks into sub-tasks or activities and run them in parallel. Making proficient use of threads on Android can help in boosting the app's performance ("Better Performance through Threading", 2017). Feature selection techniques can be used to reduce data dimensionality. This involves removing attributes that are deemed to contain irrelevant information. Further technique includes the use of data sampling schemes. Sampling is a method for selecting a subset of data from the complete dataset in order to analyze and create models, where the subset sufficiently represents the whole data set (Nettleton, 2014). Finally, for time-series data sampling the data on consistent time intervals can be adapted to reduce data dimension.

4.2 Mobile Resources Optimization Algorithm

Mobile resources optimization algorithm is depicted in Figure 6. It starts with populating background applications vector list, with the running current applications details. Similarly, network connections list and various accounts list configured in the mobile device are populated. For each application in the application vector list (bgApps), is checked against the required or preferred application list by the user. If it is not a required application, the application is stopped. In the next step, if application cache files are available, it is cleared. Also, any notifications enabled for the application is disabled. For each connection in the network connections vector (netCon), if the connection is not required by the application, it is deactivated. For

instance, if Wi-Fi is not required for the analytic process, it is switched off. Likewise, if location data is not needed for the tasks, GPS is switched off. If the analytic tasks, requires no interaction with the user while processing, the backlight of the mobile device is switched off. Finally, for the accounts configured in the mobile device (i.e.: Google, iCloud etc.), the auto account synchronization is disabled to cut the synchronizations of mails, contacts, photos, and more. Thus a combination of all these tweaks together can realize significant energy savings.

Algorithm 1: Smart Energy Saving - Mobile Resources Optimization

Input:

\vec{rConn}	▷ Vector of Required Connections for processing
\vec{bgApps}	▷ Vector of Background Applications List
\vec{netCon}	▷ Vector of Network Connections List
$\vec{accList}$	▷ Vector of Accounts eg:google
// Following are updatable User Preferences	
$\vec{rAppList}$	▷ Vector of Required Apps
rGPS	▷ Boolean If GPS Required
rBlight	▷ Boolean If Backlight Required

Result: Mobile Resources Optimization**procedure** MOBRESOURCEOPTIMIZE(\vec{bgApps} , \vec{netCon} , rGPS, rBlight) $\vec{bgApps} \leftarrow \text{GETRUNNINGAPPLICATIONLIST}()$ $\vec{netConn} \leftarrow \text{GETNETWORKCONNECTIONLIST}()$ $\vec{accList} \leftarrow \text{GETAPPACCOUNTS}()$ **foreach** $app \in \vec{bgApps}$ **do** **if** app **is** $\notin \vec{rAppList}$ **then** stop(app) **end** **if** app **has** availableCache() **then** clearCache(app) **end** **if** app **has** notificationEnabled() **then** disableNotification(app) **end****end****foreach** $con \in \vec{netCon}$ **do** **if** con **is** active() **and** $con \notin \vec{rConn}$ **then** deactivate(con) **end****end****if** rGPS **is** false **then**

switchOffGPS()

end**if** rBlight **is** false **then**

switchOffBackLight()

end**foreach** $acc \in \vec{accList}$ **do** **if** acc **is** autoSyncEnabled() **then** disableAccAutoSync(acc) **end****end****end procedure**

Figure 6: Mobile Resources Optimization Algorithm

4.3 Mobile Analytics Customization Algorithm

In the mobile analytics customization algorithm represented in Figure 7, we introduce the minimum required battery threshold which is calculated dynamically. Battery discharge is a variable which depends on the network connectivity, number of processes, number of code instructions, data size, and screen brightness. Minimum required battery dynamic threshold is a function of battery availability and the rate of battery discharge. When the minimum battery threshold event is triggered, the tasks which need to be finished with high priority are processed and the user is notified to charge the device. Also, the instance state of the application is saved. This is to restore the application process if the device is shut down unexpectedly, by storing the task variables, state, and intermediate values.

In the analytics customization procedure, if the data is decomposable, data reduction techniques such as dimensionality reduction or sampling is conducted. Similarly, if tasks are decomposable to sub-tasks, parallel processing with multi-threading is performed. The analytics customization approach fully depends on the contextual data, the analytics tasks, and the intended results. For instance, in ECG analytics, QRS detection may be necessary, while EEG analytics may require classification based on waveform frequency (e.g., alpha, beta, theta, and delta).

The two algorithms represent two consecutive stages as shown in the flowchart diagram of Figure 5, and are designed this way for modularity purpose. Most of the smart phones incorporate automatic power saving when reaching the battery threshold around 15%. However, since we know beforehand, the task is heavy we can save battery even if it is at 100% battery level, or from the very beginning of the task execution, thus saving energy for longer task execution.

Algorithm 2: Smart Energy Saving - Mobile Analytics Customization

Input:

sData ▷ Sensed data from Body Sensors as Datastreams
 $\tilde{v}T$ ▷ Vector of Tasks/ Processes
 BLevel ▷ Available Battery Level of Mobile Device
 BMinThreshold ▷ Minimum Threshold Battery Required

Result: Mobile Analytics Customization

 BMinDynamicThresholdEvent \leftarrow RECIEVEBATTERYLOWEVENT()

procedure MOBANALYTILICSCUSTOMIZATION(sData)

 if *checkDataDecomposable*(sData) then

DoDataReduction(sData) ▷ Depends on Contextual Sensor Data and Process
▷ Dimensionality Reduction, Sampling

end

 foreach $T_n \in \tilde{v}I$ do

 if *checkTasksDecomposable*(T_n) then

Decompose $T_n \rightarrow T_{n_i}$ ▷ Depends on Contextual Sensor Data and Process
▷ Task Decomposition T_{n_i} is a unit task

runParallelProcessing(T_{n_1}, T_{n_n}) ▷ Multithreading

else

runSequentialProcessing(T_n)

end

end

end procedure

 upon event $< BMinDynamicThresholdEvent >$ do

ApplyBatteryPowerSaveMode()

▷ System built in power save mode

PrioritizeTasks()

▷ Finish the prioritized tasks

NotifyToCharge()

▷ Battery Charge Notification

saveInstanceState()

▷ Write the application current state values

end event

function RECIEVEBATTERYLOWEVENT(

NetwokConn = fn(GPS,WiFI,Bluetooth,MobileData)

BatteryDischarge= fn(NetwokConn,NoOfProcess,NoOfInstructions,DataSize,ScreenBrightness)

BatteryAvailable = BatteryMeter.Availabity()

DynThreshold = fn(BatteryAvailable/BatteryDischarge)

)

 end function

Figure 7: Mobile Analytics Customization Algorithm

Chapter 5: Mobile Data Offloading

Mobile computing allows healthcare providers to utilize application and resources with minimum investment on software and hardware, henceforth reducing capital expenditures. Also, healthcare providers and receivers now have access to more features on their mobile phones with online transaction capabilities, from anywhere and at any time. Modern healthcare is enriched with m-commerce which allows doing advertising, shopping and procurement with secure m-payment options. M-learning and training options are also now viable and vastly adopted by institutions. M-gaming is another potential revenue making business for service providers, where offloading can save mobile energy and increase game playing time. M-health is also a promising area where patients can perform mobile health monitoring, access health data on the move and also interact with care givers, thanks to the advancements in communication technology, in Wi-Fi, 3G/4G which enables faster communications and transactions with improved security. Large geographical markets can be covered by a business with use of mobile computing (Desai, 2016). Lately, the mobile experience is tied to the ‘BYOD’ (Bring Your Own Device) phenomenon, as an evolving aspect of the work organization which has an impact on the business. Organizations that embrace BYOD have some advantages over competitors as the devices tend to be more advanced and organization gets the benefit of the latest features and capabilities (Cognini, Gagliardi, & Polzonetti, 2013).

When a healthcare application must go mobile, the aim is to offer the same services on the mobile platform as on the desktop and to allow the users to be comfortable and have full flexibility. When health information is stored both on the

mobile device and on the server, intermittent network access may lead to accidental partition. Hence, the quality attributes to address with the mobile version of the healthcare applications are usability, performance, maintainability, consistency of application's data, availability of service and partition tolerance (Dugerdil, 2013). Mobile applications are broadly classified into five categories: mobile broadcast (m-broadcast), information (m-information), transaction (m-transaction), operation (m-operation), and collaboration (m-collaboration) (Unhelkar & Murugesan, 2010).

Offloading is an effective technique for extending the lifetime of a handheld mobile device by executing some or all components of mobile applications remotely (Huang, Wang, & Niyato, 2012). Offloading is not only for computation but also for mobile data backup/synchronization (Barbera, Kosta, Mei, & Stefa, 2013) which is of prime importance to organizations providing mobile device facilities to employees. In this case SMS (Short Message Service), calendar events, mail and contacts are required to be synchronized with enterprise software such as ERP (Enterprise Resource Planning) and CRM (Customer Relationship Management).

5.1 Offloading Scheme

In this section, we propose an offloading scheme as illustrated Figure 6. The initial step is to acquire the Big data to be processed and identify the data analytics processes to be conducted. In the following we detail each of these steps:

1. *Analytics Segmentation*: The breakdown of the analytics tasks to be performed, and information with details regarding the tasks, subtasks, number of instructions and data required for the task are identified.

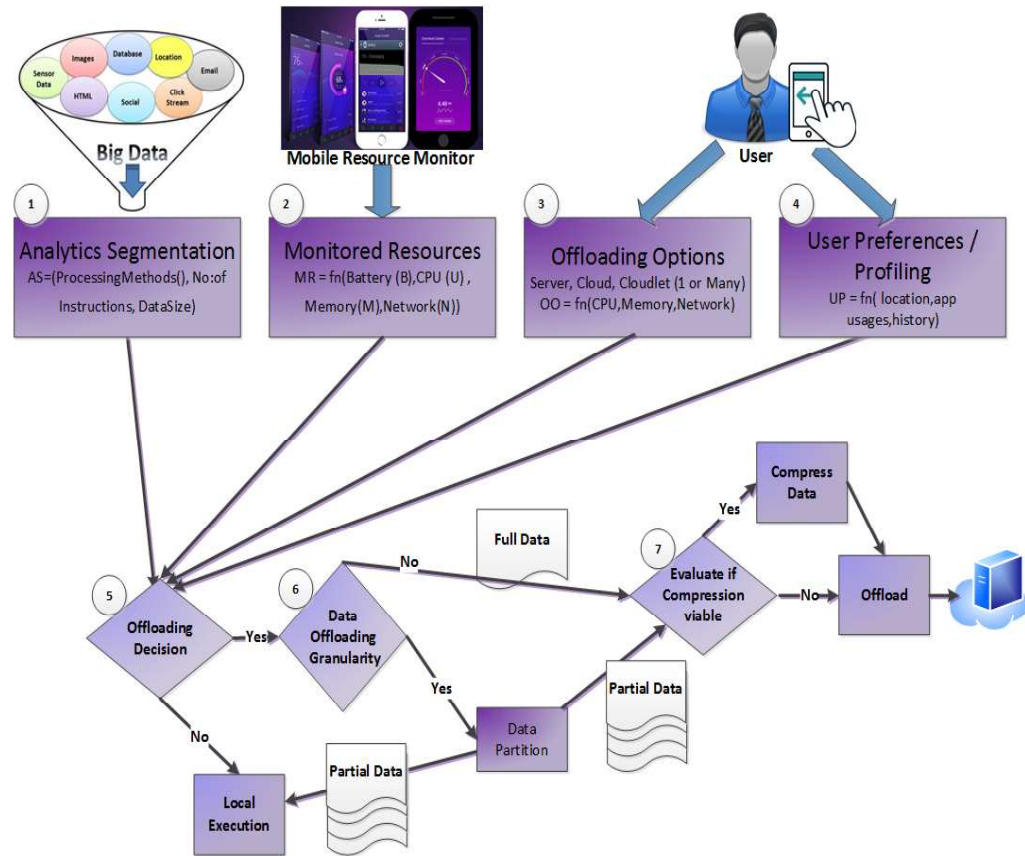


Figure 8: Proposed Mobile Offloading Scheme

2. Mobile Resource Monitoring: This step is to monitor the current resource usage information of the device. Mobile hardware resources which need to be monitored are: network availability and bandwidth, CPU usage, battery usage, memory usage, disk usage, and sensors (Patil, Hakiri, & Gokhale, 2016). In mobile devices, when using 2G/3G technology, they switch from 3G to 2G whenever the 3G signal is not present or lower than a certain threshold determined by the device type. The 2G upload speeds are much lower than the 3G. Wi-Fi network is quicker than 3G in terms of download speeds; however, the average upload speeds are higher in the 3G with respect to Wi-Fi (Barbera et al., 2013).

3. *Offloading Options*: The available offloading options (i.e. cloud, server, cloudlet and the CPU processing speed, network availability, bandwidth of each network), is an input for offloading decisions. Also, multiple offloading (i.e. offloading to one or more locations) can also be a point of consideration. The author of Mobile Cloud Computing (Debashis De, 2015), suggests ways to determine the optimal offloading path, when multiple offloading options are available. The options where random, Bandwidth dependent, Link failure rate dependent, Speedup factor dependent, Security dependent and Cost dependent. The optimal offloading path selection consumes time and energy and hence rerunning the computation every time, for the same data, application and offloading choices may be unnecessary. Path selection based on ant colony optimization would be ideal, with caching the path and using it whenever a similar offloading request is issued (Debashis De, 2015). The offloading options should be set in application settings by the user, which should be modifiable; for instance, to add a new cloud server, with required connection parameters and authentication details.

4. *User Preferences and Profiling*: Mobile applications should also adapt to the user's dynamic profile which might change with location, the context of requested information and applications data use (Unhelkar & Murugesan, 2010). Also, options for users to feed their preferences are considered. For instance, an individual user preference is to select at workplace, a specified Wi-Fi connectivity.

5. *Offloading Decision*: The offloading decision, is a context aware calculation based on all the above-mentioned inputs from step 1,2,3, and 4. The offloading decision calculation itself might add an overhead for the battery resources, and hence a valid determinant for offloading.

6. *Data Offloading Granularity*: With full offloading architecture, the whole application and its dataset are offloaded to a server/cloud and the mobile device awaits results from the server. With partial offloading architecture, a part of the application and dataset manipulated by this part are offloaded. Typically, a CPU-intensive energy consuming component is offloaded. When streamlined, this architecture yields parallelism and improves performance. An overhead with partial offloading is data redundancy at the device and server (Kim, 2012).

7. *Data Compression*: Further the data to be offloaded can be checked for compression viability. If data compression saves storage space and transmission costs, then compression may be a better choice, else the data is offloaded without compression.

5.2 Offloading Algorithm

Our proposed mobile offloading algorithm is shown in Figure 9. As described in the previous section (i.e. Offloading scheme), in the first four steps, the corresponding vectors are populated. Each task in the analytics information vector (vAI) is first decomposed in to unit tasks. Each of the unit tasks is evaluated for offloading. If the execution cost of the unit task with the available mobile device resources is greater than the execution of same unit task at a user preferred cloud/server using the server resources with added transmission cost, then the task is selected for offloading. Also, the data required for the offloaded task is partitioned for transmission.

Algorithm 3: Mobile Data Offloading**Input:**

\vec{vAI} ▷ Vector of Data Analytics Information
 \vec{vMR} ▷ Vector of Monitored Mobile Resources
 \vec{vOO} ▷ Vector of Offloading Options
 \vec{vUP} ▷ Vector of User Preferences
 \vec{vOT} ▷ Vector of To Be Offloaded Tasks
 \vec{vLT} ▷ Vector of To Be Locally Executed Tasks
BData ▷ Sensed Big data from various sources
 \vec{vOD} ▷ Vector of To Be Offloaded Data
 \vec{vLD} ▷ Vector of To Be Locally Stored Data

procedure MOBILEOFFLOADINGDECISION(BData)

$\vec{vAI} \leftarrow \text{ANALYTICSSEGMENTATION}()$ ▷ Get Analytics/Processing Tasks
 $\vec{vMR} \leftarrow \text{GETMONITOREDRESOURCES}()$ ▷ Monitored Resources: Battery, CPU, Storage, Network
 $\vec{vOO} \leftarrow \text{GETOFFLOADINGOPTIONS}()$ ▷ Connection parameters and resources of server/cloud
 $\vec{vUP} \leftarrow \text{GETUSERPREFERENCES}()$ ▷ Configurable user preferences on Network, Offloading

foreach $T_j \in \vec{vAI}$ **do**
 Decompose $T_j \rightarrow T_{ju}$ ▷ T_{ju} is a unit task

foreach $T_{ju} \in T_j$ **do**
 if $\text{evaluateOffloading}(T_{ju}, \vec{vMR}, \vec{vOO}, \vec{vUP}) = \text{True}$ **then**
 $\vec{vOT} \leftarrow T_{ju}$ ▷ To Be Offloaded
 $\vec{vOD} \leftarrow \text{SPLITDATA}(T_{ju}, \vec{vOT}, \text{BData})$
 else
 $\vec{vLT} \leftarrow T_{ju}$ ▷ To Be Locally Executed
 end
 end

end

foreach $T_{tb} \in \vec{vOT}$ **do**
 if $\text{evaluateCompressionViable}(T_{tb}, \vec{vOD}, \vec{vMR}, \vec{vOO}, \vec{vUP}) = \text{True}$ **then**
 performCompression(T_{tb}, \vec{vOD})
 end
 offload(T_{tb}, \vec{vOD})
end

end**end procedure**

function EVALUATEOFFLOADING($T_{ju}, \vec{vMR}, \vec{vOO}, \vec{vUP}$)
if $\text{executionCost}(T_{ju}, \vec{vMR}) \geq \text{executionCost}(T_{ju}, \vec{vOO}, \vec{vUP}) +$
 $\text{transmissionCost}(T_{ju}, \vec{vOO}, \vec{vUP})$ **then**
 return True
else
 return False
end function

Figure 9: Mobile Data Offloading Algorithm

Reducing the size of EEG data using compression algorithms has two main benefits: 1) reduce the size of locally stored data and 2) efficiently transmit over a communication network. We use the CVA (Compression Viability Algorithm) (Serhani, El Menshawy, Benharref, & Navaz, 2016), published in a previous work of ours, where the inputs are the offloading task, the corresponding EEG data, the available resources on the mobile device, the offloading options as well as the user preferences in order to evaluate the compression viability. If viable the data and the task is compressed before offloading.

There are two major approaches related to operational level issues when offloading is considered: application partition and offloading technology. Application partition decomposes complex workloads to atomic ones, to be processed concurrently (AbdElminaam, Abdul Kader, Hadhoud, & El-Sayed, 2013), which is the basis of analytics segmentation method. Offloading is clearly beneficial when the local completion time is greater than the remote completion time, the device battery resources are not adequate to process data locally, and/or the device storage is not sufficient. Our offloading algorithm helps in the key decision on when and where to accomplish the mobile analytics which may include the preprocessing and processing of data. At this point, we need to stress that the task decomposition and data partitioning depends mainly on the analytics tasks and the data itself.

Chapter 6: Implementation and Experimentation

In this chapter, we describe the implementation and experimentation we have conducted to evaluate: 1) Mobile Resources Optimization 2) Analytics Customization and 3) Mobile Offloading. We also, describe the dataset we have used in our experiments, the main technologies we have employed, and the set of scenarios we have selected to conduct the experiments and finally we discuss the results we have obtained.

6.1 Technologies

To implement the features of our solution/architecture, we employed a set of web services using their associated APIs. We developed REST (Representational State Transfer) web services to invoke remote services, JSON (JavaScript Object Notation) for lightweight data-interchange, and AJAX (Asynchronous JavaScript + XML) for data interchange and dynamic display. Such lightweight programming models would provide significant energy savings while transferring data (Gemson Andrew Ebenezer & Durga, 2015).

6.2 Dataset

We used both real-time streamed data retrieved directly from sensors and online available data bank for the purpose of testing. Data from sensors were checked for the viability of continuous monitoring samples from real-time EEG sensors such as Emotiv EPOC neuroheadset (“EMOTIV Epoc - 14 Channel Wireless EEG Headset,” 2016) and Muse, the brain sensing headband (“MUSE | Meditation Made Easy,” 2016). We used online available CHB-MIT database (Physionet.org, 2000) to

test and evaluate the effectiveness of the developed algorithms. This database is gathered by the Boston Children's Hospital. It consists of EEG recordings obtained from pediatric subjects having intractable seizures. The number of recorded events includes 198 seizures, with relevant annotation files. In particular, the recordings were composed from 22 subjects (5 males, ages 3 to 22; and 17 females, ages 1.5 to 19). EEG signals were sampled at 256 samples per second with 16-bit resolution. This database is a base for various machine learning researches (Shoeb & Gutttag, 2010), to detect the onset of an epileptic seizure through analysis of the scalp EEG, a non-invasive measure of the brain's electrical activity.

6.3 Experimental Setup

The main objective of the implementation phase is to experimentally test the effectiveness of our proposed algorithms and evaluate their ability to optimize resources and energy consumption. We developed an Android application to simulate a mobile health monitoring system to process EEG data. The simulation is implemented using Android Studio 2.2.3 and Samsung Galaxy Tab S SM-T805, with Wi-Fi, 3G, and 4G LTE and 7900mAh battery. For the offloading end, we require a cloud server or backend application server as well as a MySQL database server and Matlab for signal processing. We also need Big data processing environments such as Hadoop Ecosystem. Hadoop consists of two basic components: a distributed file system (HDFS) and the computational framework (MapReduce). Once Big data EEG analytics tasks are offloaded, these can be processed by Hadoop framework, which enables distributed and parallel computing (Sivaraman & Manickachezian, 2014).

Figure 10 describes the experimental environment which features the data, technologies, and the experimental configurations.

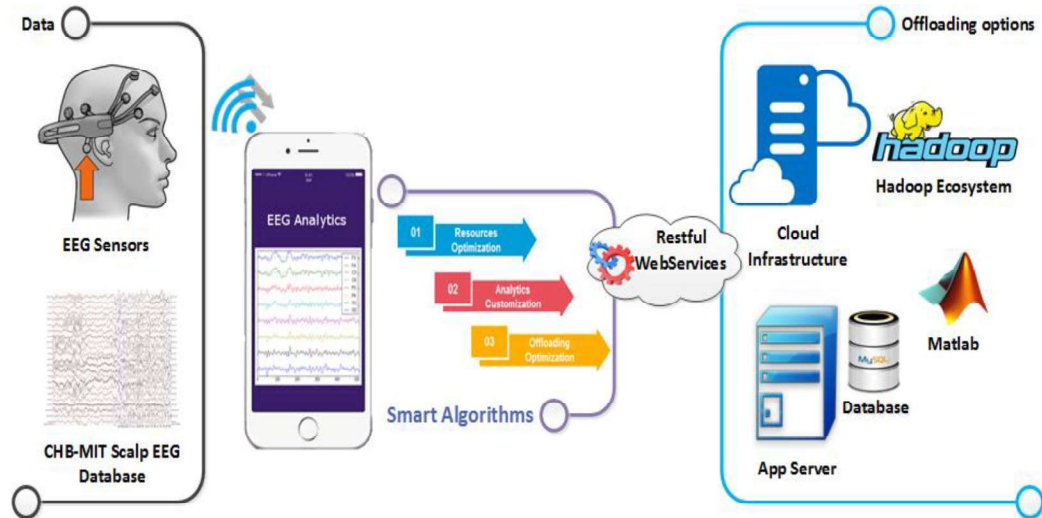


Figure 10: Experimental Environment

6.4 Scenarios

To validate the proposed smart algorithms, we implemented the algorithms on an Android mobile device. The application main menu is presented in Figure 11. The optimize menu implements the resources optimization, for energy saving purpose. The ‘battery meter’, meters the battery resources and execution time before and after the EEG processing. The third menu enable user to stop applications running in the background. The settings menu, gives the user the chance to set his/her preferred choices for optimization. We tabulated the experiments and analyzed the performance by measuring battery discharge rate and the execution time, before and after optimization. The analytics tasks chosen were to split EEG data, stored as EDF file

into header, annotation, and signal files. These files were written in JSON format and were stored in Android device.

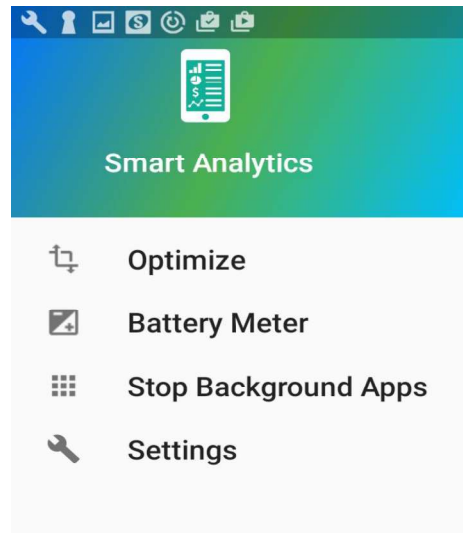


Figure 11: Application Main Menu

The battery meter option allows choosing the EEG File, and to start the data processing as shown in Figure 12.

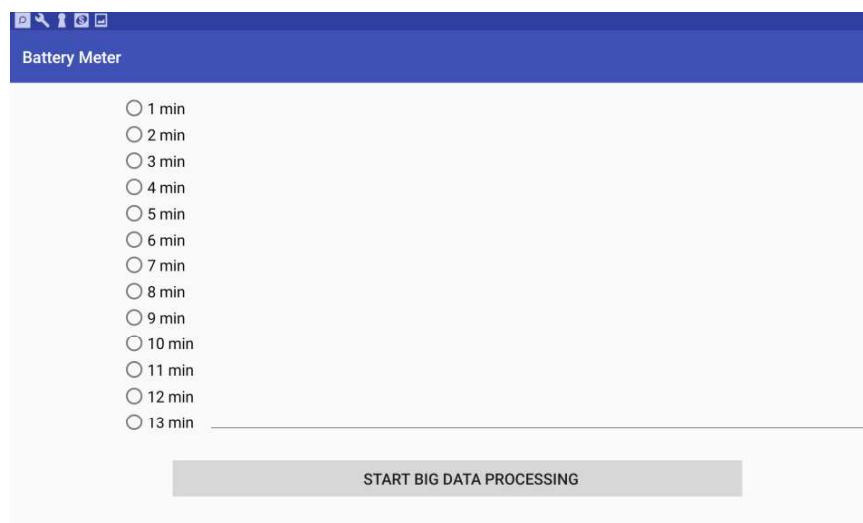


Figure 12: Experimental Setup to Meter Battery Resources.

6.4.1 Non-optimized Mobile Device Task Execution

The experimental values were noted, before optimization as shown in Table 1. USB Charging, Wi-Fi, Bluetooth, GPS and screen brightness indicate Boolean values (i.e. 0 \rightarrow False, 1 \rightarrow True). For instance, a value 1 for Wi-Fi refers to Wi-Fi is connected and a value 0 refers to Wi-Fi disconnected. A screen brightness of 1 indicates, full brightness and zero indicates minimum brightness.

EEG Data	USB Charging	Wi-Fi	Bluetooth	GPS	Screen Brightness	Battery Discharge Rate (%)	Execution Time (seconds)
1min	0	1	1	1	1	1	40.46
2min	0	1	1	1	1	1	77.58
3min	0	1	1	1	1	1	115.15
4min	0	1	1	1	1	1	195.32
5min	0	1	1	1	1	2	205.88
6min	0	1	1	1	1	2	556.182
7min	0	1	1	1	1	2	871.202
8min	0	1	1	1	1	2	1300.47
9min	0	1	1	1	1	2	3476.87
10min	0	1	1	1	1	2	5037.71
11min	0	1	1	1	1	3	8301.43
12min	0	1	1	1	1	3	9341.98
13min	0	1	1	1	1	3	12086.15

Table 1: Experimental Findings before Optimization

We conducted the experiment with the mobile device connected to the available networks, various applications running on the background and the screen backlight switched on. Battery discharge rate as well as the total execution time of the EEG data process is recorded. Battery discharge percentage is calculated by extracting the current battery level and scale from the battery status (“Monitoring the Battery

Level and Charging State | Android Developers,” 2017). Battery Discharge rate is the difference in battery discharge percentage noted at the commencement and end of the process (example: if battery discharge percentage at the beginning of the process is 65 and the end of the execution is 62, then there is a 3% discharge rate).

6.4.2 Optimized Mobile Device Task Execution

To optimize resources, the first step is to choose the preferred user settings as presented in Figure 13, primarily to switch on/off the network connections, screen brightness, GPS and account synchronization. Also, user preferences on background applications are captured. Then the optimize menu in Figure 10 is activated, which facilitates the execution of the user preferences chosen by the user.

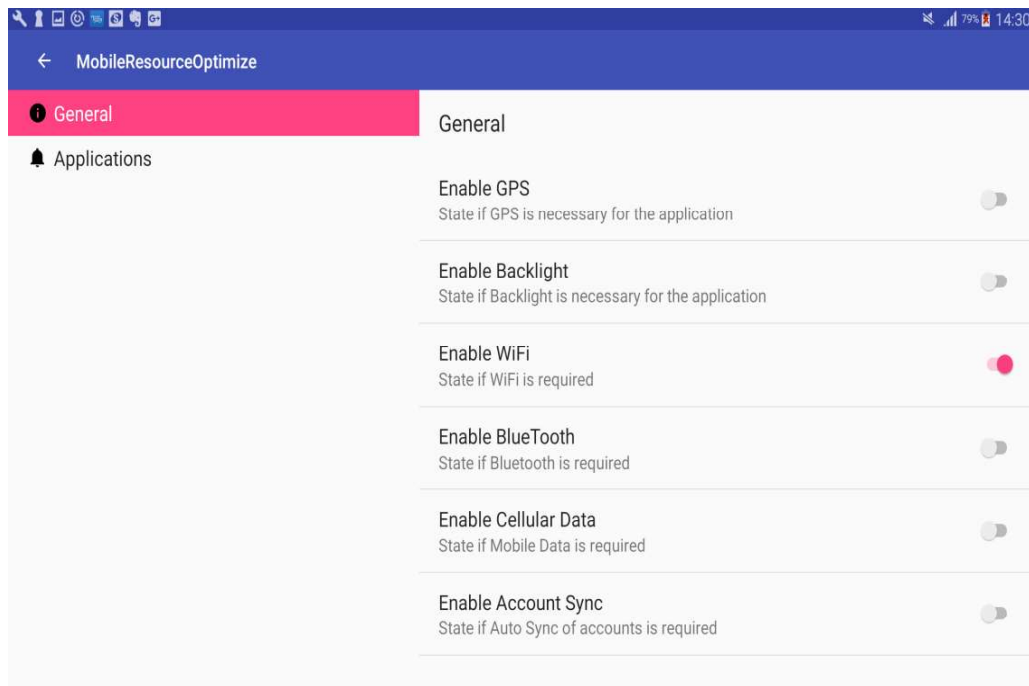


Figure 13: User Preferred Optimization Settings

The next step in resource optimization is to disable the background applications which are not preferred by the user. Initially, the running applications are listed as shown in Figure 14. When the button to kill the background, applications is triggered, the applications are stopped as depicted in Figure 15. However, the system and user preferred applications are exempted. In addition, the application cache files are automatically removed.

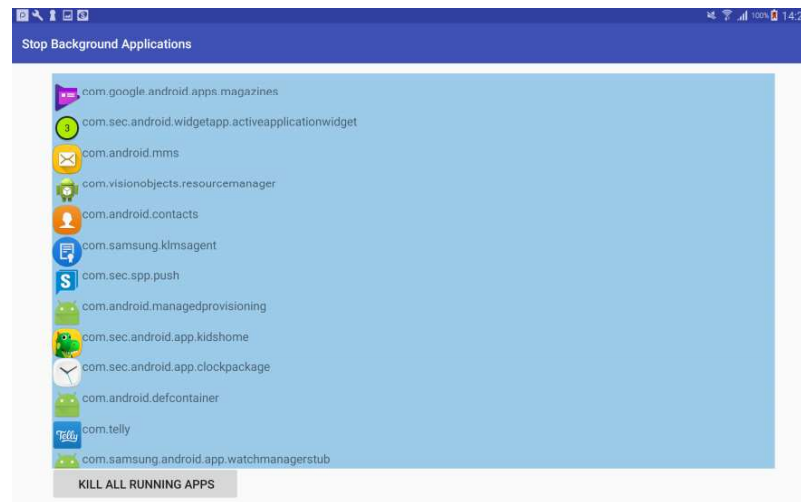


Figure 14: Background Applications Control Interface

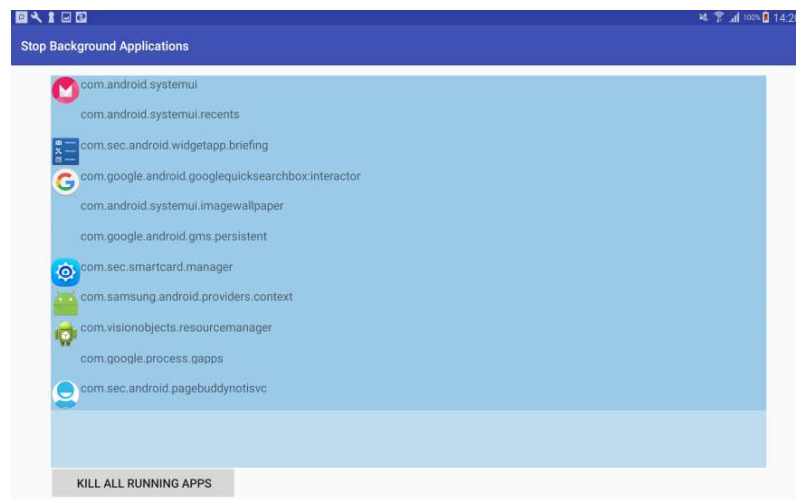


Figure 15: Control Interface - After Stopping the Non-Preferred Background Apps

Once all the resource optimizations techniques are performed, the experiment is repeated and the values recorded are depicted in Table 2. The monitored variables are the same as in Table 1, explained previously.

EEG Data	USB Charging	Wi-Fi	Bluetooth	GPS	Screen Brightness	Optimized Battery Discharge rate (%)	Optimized Execution Time (seconds)
1min	0	0	0	0	0	0	37.11
2min	0	0	0	0	0	0	73.39
3min	0	0	0	0	0	0	110.80
4min	0	0	0	0	0	0	147.08
5min	0	0	0	0	0	1	182.60
6min	0	0	0	0	0	1	219.35
7min	0	0	0	0	0	1	255.39
8min	0	0	0	0	0	1	294.92
9min	0	0	0	0	0	1	330.41
10min	0	0	0	0	0	1	371.71
11min	0	0	0	0	0	1	404.25
12min	0	0	0	0	0	2	446.59
13min	0	0	0	0	0	2	484.92

Table 2: Experimental Findings after Optimization

6.4.3 Evaluation of Resources Optimization on Battery Drainage and Execution Time

To better understand the results of the experiments, the battery discharge rate and the execution time of optimized and non-optimized runs are plotted as line graphs (Figure 16, 17).

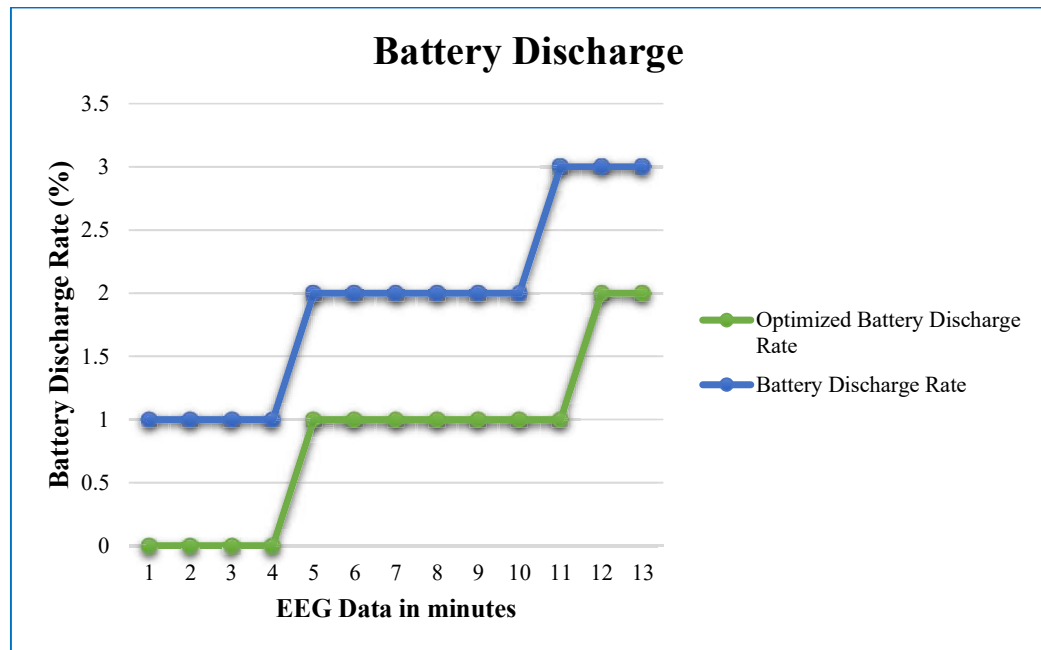


Figure 16: Optimized vs Normal Battery Discharge Rate

The mobile device battery discharge rate in Figure 16 indicates that, as EEG data file size increases in size/time, the battery discharge rate also increases. By viewing the optimized battery discharge rate, indicated by the green line, it is evident that optimized task execution consumes less battery, however it follows the same consumption pattern as the non-optimized run.

The zero-battery discharge, in the optimized run till the fourth minute may lead to confusion. However, it merely indicates that the battery discharge percentage at the beginning and end of the process remains same. This does not mean that there is no energy consumption, but it is a limitation in Android to represent battery level scale in decimals (i.e.: remaining battery capacity is represented as an integer percentage of total capacity with no fractional part) (“Battery Manager | Android Developers,” 2017).

Figure 17 depicts the optimized versus normal execution time, plotted with EEG data and time measured in seconds.

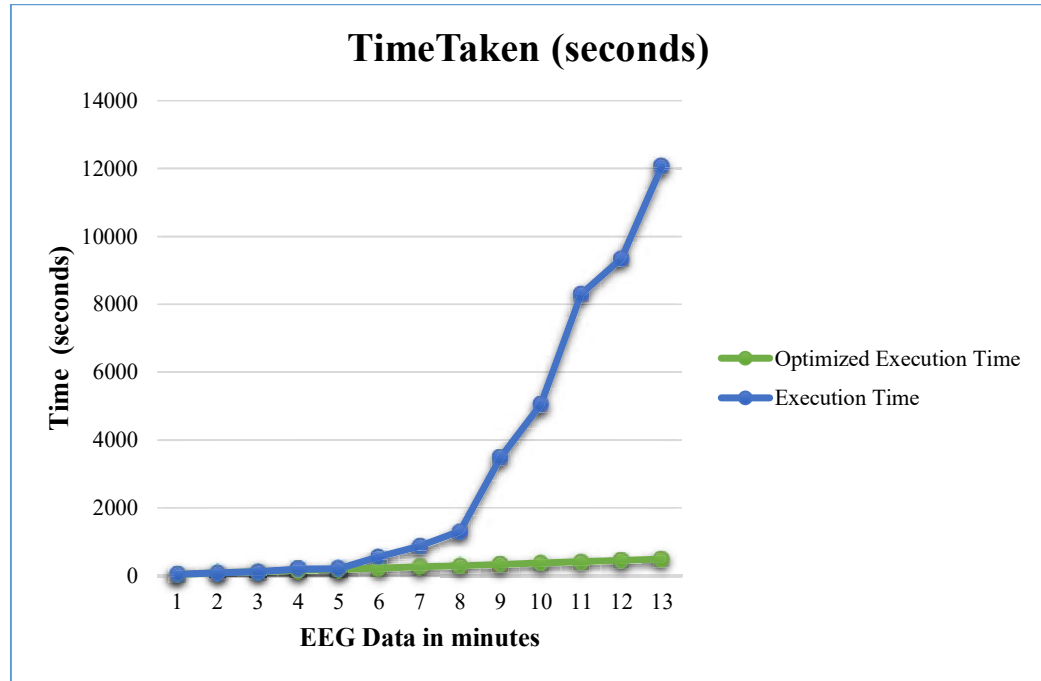


Figure 17: Optimized vs Normal Execution Time

The optimized execution time, depicted by the green plot in Figure 17, shows an upward linear steady trend, with increase in data. However, the non-optimized run shows a dramatic increase in execution time, thus highlighting a huge difference in execution time between the experiments, especially when the data is increasingly growing. A comparison of the optimized and non-optimized task execution is depicted in Table 3. The percentage gain in time and energy savings is calculated as:

Percentage gain = $\frac{\text{actual increase}}{\text{original value}} \times 100$, where actual increase is the difference in time /battery discharge rate between the non-optimized and optimized run and original value is the respective non-optimized values.

The execution time, in optimized experiment in an average saved processing time up to 56%. It is also noted that when the file sizes are small, the savings are negligible. However, with a comparably bigger EEG files specifically from 9th minute data, the execution time was significantly reduced to exceed 90%-time savings after optimization. For battery savings in the optimized experiment, on an average 64% of energy was saved.

EEG Data	Non-optimized		Optimized		Gain (%)	
	Battery Discharge Rate (%)	Execution Time (seconds)	Battery Discharge Rate (%)	Execution Time (seconds)	Time Savings (%)	Energy Savings (%)
1min	1	40.46	0	37.11	8.28	100
2min	1	77.58	0	73.39	5.40	100
3min	1	115.15	0	110.8	3.78	100
4min	1	195.32	0	147.08	24.70	100
5min	2	205.88	1	182.6	11.31	50
6min	2	556.182	1	219.35	60.56	50
7min	2	871.202	1	255.39	70.69	50
8min	2	1300.47	1	294.92	77.32	50
9min	2	3476.87	1	330.41	90.50	50
10min	2	5037.71	1	371.713	92.62	50
11min	3	8301.43	1	404.25	95.13	66.67
12min	3	9341.98	2	446.59	95.22	33.33
13min	3	12086.15	2	484.92	95.99	33.33
				Average	56.27	64.10

Table 3: Execution Time and Energy Savings Achieved

The experiments revealed that resources optimization techniques saved battery resources as well as processing time. It simply means, after resources optimization, the same process will consume less energy and the battery will last longer. Also, it has been noted that when the mobile device is on charging mode, the battery discharge rate is negligible.

6.4.4 Parallel Execution of Tasks for Analytics Optimization

For analytics customization experiments, we attempted parallel execution of tasks in Android. We employed, AsyncTask (“AsyncTask | Android Developers,” 2017), which enables proper and easy use of the UI thread. In our experiments, we found that the execution time increased, while the battery consumption remained almost same when not in charging. This was not the expected outcome, since with parallel execution; the time of completion should have been less. We learned that initially, Android AsyncTasks were executed serially on a single background thread. Later with Android Donut version, this was changed to a pool of threads allowing multiple tasks to operate in parallel. Finally, from the Android Honeycomb, tasks are executed on a single thread to avoid common application errors caused by parallel execution. Therefore, we had to devise parallel execution, with java.util.concurrent package with APIs such as ThreadPoolExecutor and FutureTask (“AsyncTask | Android Developers,” 2017). The results are presented in Table 4.

EEG	Serial		Parallel	
File	Execution Time (seconds)	BDR (%)	BDR (%)	Execution Time (seconds)
2min	73.39	0	0	41.75
3min	110.80	0	0	45.31
4min	147.08	0	0	84.18
5min	182.60	1	0	120.50
6min	219.35	1	1	127.33
7min	255.39	1	1	342.83
8min	294.92	1	1	455.15
9min	330.41	1	2	561.82
10min	371.71	1	2	2564.15

Table 4: Serial vs Parallel Execution

The execution time of serial and parallel execution of the task is plotted as in Figure 5. It indicates that till the sixth minute, the parallel execution time was lesser than the serial execution. However, from seventh minute the parallel execution time shoots up.

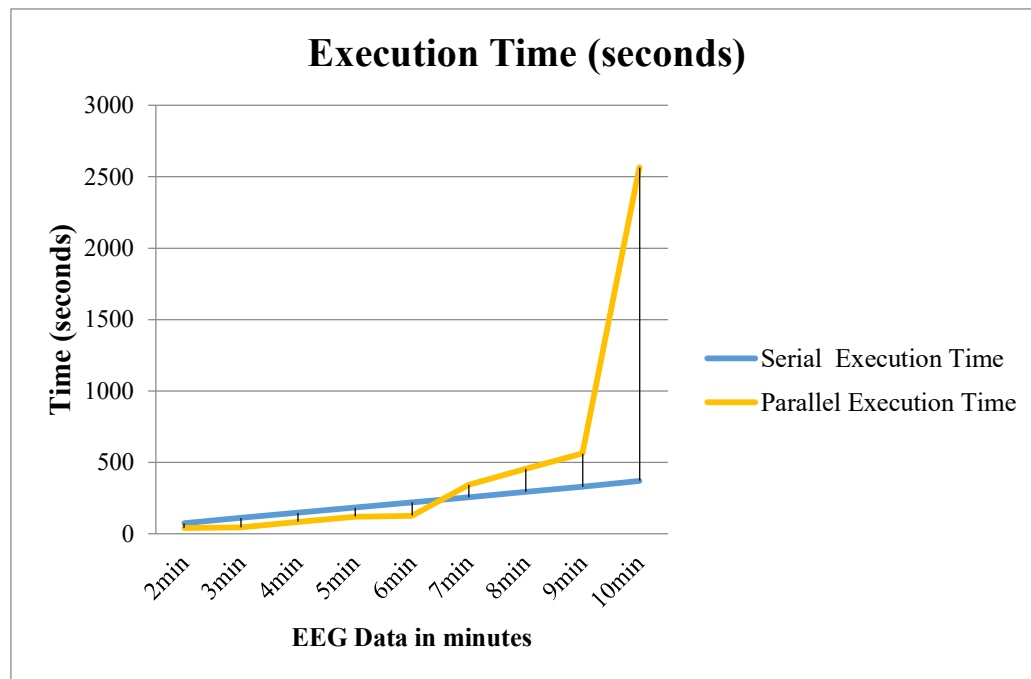


Figure 18: Serial vs Parallel Execution Time

This illustrates the problem of limited shared resource in our case the CPU, which is not scalable after a specific point. Threading, when the file load grows after a threshold, thus may cause performance degradation. In the field of parallel computing there is Amdahl's law (Amdahl, 1967) that can be applicable in this case. The point of Amdahl's law is that in any program there is always a portion that cannot be run in parallel (the sequential portion) and there is another portion that can be run in parallel (the parallel portion), and these portions always add up to 100%. Thus, the

winning technique is in balancing the sequential and parallel portions of the tasks. Battery Discharge Rate (BDR) experience the same phenomenon, initially saving the battery discharge, however after a threshold, it ends up discharging faster as indicated in Figure 18.

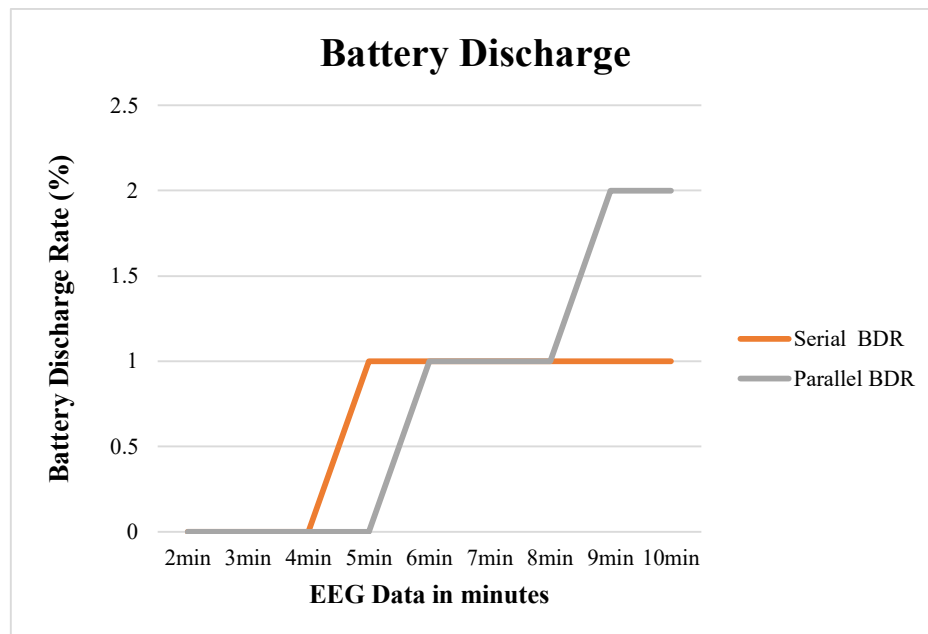


Figure 19: Serial vs Parallel Battery Discharge Rate

The second analytics customization technique is to perform data reduction, by selecting the most appropriate signals in diagnosing the disease, for instance epilepsy detection. These experiments require a background knowledge in diagnosing diseases and technical knowledge in signal processing, using tools such as Matlab. With limited time frame for development and implementation, these are considered for future works.

6.4.5 Mobile Offloading Evaluation

Mobile device applications with offloading provide several benefits over stand-alone applications such as reducing resource consumption on mobile devices and utilizing computing power of Big data processing tools on the server. In the experiment to be conducted, we will identify scenarios for health analytics, which comprises of several independent tasks. Each of these tasks will be evaluated for offloading, and if chosen, then checked for compression viability. With offloading options established with a cloud or server, Restful web services can be used to transfer files converted to JSON format. Two commonly known quality attributes for mobile offloading architecture are response time and energy consumption. In our experiments, we can record and evaluate the time and energy consumed at different phases of offloading as depicted in Figure 20.

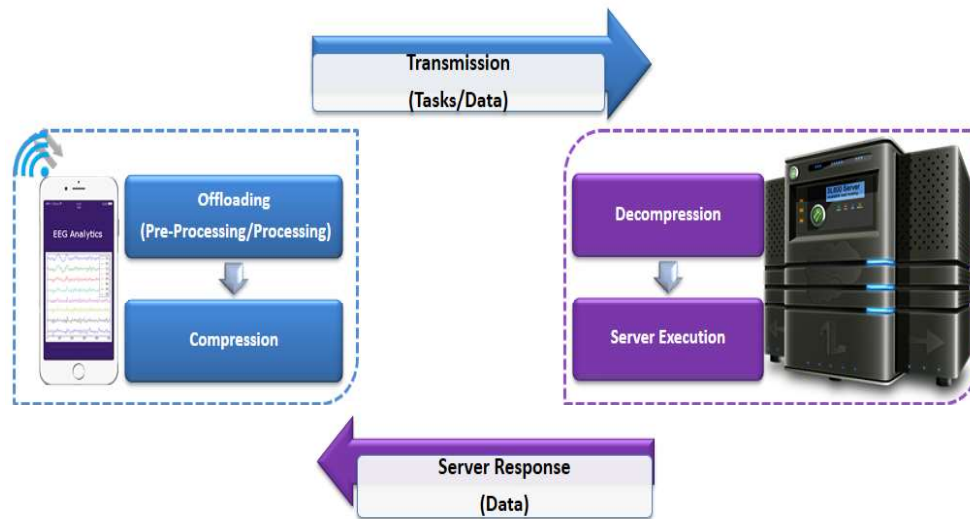


Figure 20: Mobile Offloading Evaluation Phases

The offloading algorithm at the mobile side evaluates the available resources and decides on the pre-processing, processing, and data compression tasks to be executed. Then, the decided tasks and data to be offloaded are transmitted to the server. The battery discharge rate, execution time of the above activities in the mobile device and the transmission time are recorded. In case data compression is done on the mobile device, the data decompression is performed at the server side and the tasks are executed. Finally, the execution results are transmitted to the server. The server execution time and response time will also be recorded which allow verifying the effectiveness of offloading algorithm. In addition, we can evaluate data storage reduction gained from data compression. Also, experiments for complete offloading, partial offloading and minimal offloading can be conducted and energy savings and execution time savings can be compared and evaluated.

6.5 Results and Discussion

We could successfully implement the resources optimization algorithm and we attempted parts of the second algorithm, analytics customization. We identified the phases for evaluating the mobile offloading algorithm. However, the experiment was left for future work. Regarding metering the resources, each time the results varied vastly when optimization techniques were not used. However, on optimized reruns, we found that there were only slight variations in battery consumption and execution time. We combined different approaches together to optimize the resources. First by stopping the background applications, second, by limiting connectivity, third by switching off context-location and backlight, and finally by disabling notifications and auto synchronizations of accounts. This combination tactics, together can realize

significant energy savings. The time taken for completions of the tasks were longer when more network connections were active and when more applications were running in the background. Although, the initial aim of optimization was to save the battery resources, the experiment highlighted that the total task time duration, also was greatly reduced. Hence, we can undoubtedly confirm that optimization not only save resources but also achieve quicker task completion.

Threads take up memory, with a minimum of 64k of memory each, which adds up quickly especially in handling Big data situations (“Better Performance through Threading,” 2017) in mobile devices. In case of multi-threading for parallel execution, from a software point of view, we can create hundreds of threads. But doing so can involve some performance limitations (e.g. threads scheduling and priority issues). The Android application shares limited CPU resources with background services, the renderer, audio engine, networking, and others. CPUs can handle only a small number of threads in parallel. Excess threads may lead to priority and scheduling issues. Therefore, it’s important to only create as many threads as manageable, and we can fix it by trial-and-error methods.

Chapter 7: Conclusion and Future Work

According to Forbes Magazine technology post (Bernard Marr, 2016), by the year 2020, about 1.7 megabytes of new information will be created every second for every human being on the planet. Within five years there will be over 50 billion smart connected devices in the world, all developed to collect, analyze and share data. By 2020, we will have over 6.1 billion smartphone users globally. Smartphones in recent years comes with sensor capabilities and connectivity's with location-based technologies Wi-Fi, Bluetooth and empowers patients and physicians in real-time. There is a need to focus on mobile and its next-generation analytics, providing improved user experience with resource conservation. Unraveling the Big data related complexities in the mobile environment can provide many insights and guide in making the right decisions at the right time in the healthcare arena.

This thesis proposes an energy efficient model for mobile Big data analytics and aims to overcome many limitations in mobile device analytics primarily, long execution time and limited battery resources, by integrating smart energy saving algorithms into the mobile environment. We proposed a scalable Big data mobile analytics architecture, which comprises of three smart analytics algorithms. Resources Optimization algorithm, focuses on energy saving by switching off non-required network connection, which is the main source of battery drainer. The second algorithm, emphasize energy savings by analytics customization, by devising parallel processing and sampling of data. Finally, the mobile offloading algorithm, attempts to increase efficiency of mobile device offloading considering the offloading options,

metered resources and user preferences. We believe that the three proposed algorithms, benefits in sustainable resource aware mobile device analytics.

7.1 Limitations and Future Research Directions

Given to the scope of this thesis and broadness of the proposed contributions, conducting an extensive implementation of the algorithms was not feasible. While the experimentation to evaluate the effectiveness of the algorithm, proved parts of the proposed optimization algorithms are relevant and provides good utility in minimizing the battery consumption and execution time, there are still a few limitations. During the implementation using Android, we faced the difficulty that some of the optimization techniques for instance switching off cellular data programmatically are no longer supported. Therefore, some of these hacks work only in rooted phones. Another limitation is the lack of experimental usage of mobile versions of Big data tools such as Hadoop and Spark (Bernardino, 2016).

In our research, optimizing mobile resources to efficiently perform Big data analytics has been systematically explored. We identified many research challenges on reviewing literature of related works. One of them was privacy and security, and a possible solution is to encrypt data before storage and then transmit. However, encryption alone will not solve the problem as performing encryption or steganography techniques (Johnson & Jajodia, 1998) before sending data to the cloud/server necessitates more additional processing on the mobile device and consumes further energy (Kumar & Lu, 2010). Tin- Man (Xia et al., 2015), a system that addresses the privacy and security challenge in security-oriented offloading, with a low-overhead tainting scheme called asymmetric tainting to track accesses to

confidential data. It also uses transparent SSL session injection and TCP pay load replacement to offload. Nevertheless, it adds overhead in terms of synchronization time and power consumption. Future work is in progress to address these challenges. Also, we will study the proposed algorithm's complexity to prove their effectiveness in terms of execution time and communication overhead. The current experiments were only based on Android. In the future, our optimization algorithms can be implemented and evaluated on iOS based mobile devices as well. An end-to-end application, serving a specific health issue within a concrete Big data project environment to implement the algorithms is also in the pipeline for future experiments.

7.2 Closing Remarks

In general, mobile Big data is still considered as a new subject and research area. Big data itself is termed a “buzzword” and considered a myth, but it is in fact a life changing reality. Analyzing data is always subjective, no matter how much data is available and is a process of individual choices and interpretation. It all starts from data collection, deciding what to measure and how to measure it and goes on with making observations and patterns within the data, creating a model, understanding and visualizing what the original data implied. Less than 0.5% of all data collected is ever analyzed and used. To draw valid conclusions from data it is also necessary to optimize the resources, to make it possible to perform the analytics in the first place.

Smarter algorithms can truly make Big data computations in mobile devices last longer by optimizing the resources. With innovations in bigger battery capacity, faster CPU's, memory which comes in smaller miniature packs combined with the power of OS platforms, on-the-move (mobile) analytics is surely the way to move

forward. Seamless information exchange between healthcare providers and patients, added with capabilities to make timely decisions for both patients and physicians, on the move, can create a huge performance boost and significance, thus unraveling Big data value. Probably in the future, mobile device embedded tiny sensors can detect heart attacks or seizures before they happen, and provide immediate artificial intelligence assisted diagnosis and treatment plans. Thus, improvements in mobile Big data analytics in healthcare, will surely facilitate an increase in individual life span.

References

- Abdelminaam, D. S., Abdul Kader, H. M., Hadhoud, M. M., & El-Sayed, S. M. (2013). Elastic framework for augmenting the performance of mobile applications using cloud computing. *2013 9th International Computer Engineering Conference (ICENCO)*, 134–141. <https://doi.org/10.1109/ICENCO.2013.6736489>
- Alexandros Labrinidis, & H. V. Jagadish. (2012). Challenges and Opportunities with Big Data. *Proceedings of the VLDB Endowment*, 1–15. <https://doi.org/10.14778/2367502.2367572>
- Alsheikh, M. A., Niyato, D., Lin, S., Tan, H., & Han, Z. (2016). and Apache Spark Mobile Big Data Analytics Using Deep Learning and Apache Spark, (April), 22–29.
- Alzamil, I., Djemame, K., Armstrong, D., & Kavanagh, R. (2015). Energy-Aware Profiling for Cloud Computing Environments. *Electronic Notes in Theoretical Computer Science*, 318, 91–108. <https://doi.org/10.1016/j.entcs.2015.10.021>
- Amdahl, G. M. (1967). Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities. *AFIPS Spring Joint Computer Conference, 1967. AFIPS '67 (Spring). Proceedings of the*, 30, 483–485. <https://doi.org/10.1145/1465482.1465560>
- App Dynamics. (2015). 16 Metrics To Ensure Mobile App Success, 8. Retrieved from <https://www.appdynamics.com/media/uploaded-files/1432066155/white-paper-16-metrics-every-mobile-team-should-monitor.pdf>
- AsyncTask | Android Developers. (2017). Retrieved from <https://developer.android.com/reference/android/os/AsyncTask.html>
- Balasubramanian, N., Balasubramanian, A., & Venkataramani, A. (2009). Energy consumption in mobile phones: a measurement study and implications for network applications. *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference - IMC '09*, 280–293. <https://doi.org/10.1145/1644893.1644927>
- Barbera, M. V., Kosta, S., Mei, A., & Stefa, J. (2013). To offload or not to offload? the bandwidth and energy costs of mobile cloud computing. *Proceedings - IEEE INFOCOM*, 1285–1293. <https://doi.org/10.1109/INFCOM.2013.6566921>
- Battery Manager | Android Developers. (2017). Retrieved from <https://developer.android.com/reference/android/os/BatteryManager.html>
- Beal, V. (2014). Big data. Retrieved from http://www.webopedia.com/TERM/B/big_data.html
- Benkhelifa, E., Welsh, T., Tawalbeh, L., Jararweh, Y., & Basalamah, A. (2015). User

- profiling for energy optimisation in mobile cloud computing. *Procedia Computer Science*, 52(1), 1159–1165. <https://doi.org/10.1016/j.procs.2015.05.151>
- Bernard Marr. (2016). 20 Mind-Boggling Facts Every Business Leader Must Reflect On Now. Retrieved from <https://www.forbes.com/sites/bernardmarr/2016/11/01/20-mind-boggling-facts-every-business-leader-must-reflect-on-now/#5d8e9be020dc>
- Bernardino, J. (2016). Personalization using Big Data Analytics Platforms, 5–6.
- Better Performance through Threading. (2017). Retrieved from <https://developer.android.com/topic/performance/threads.html>
- Beyer, M. a., & Laney, D. (2012). *The Importance of “Big Data”: A Definition. Gartner Publications* (Vol. i). <https://doi.org/G00235055>
- Bonnie Feldman. (2013). The Role of Big Data in Personalizing the Healthcare Experience: Mobile. Retrieved from <http://radar.oreilly.com/2013/09/the-role-of-big-data-in-personalizing-the-healthcare-experience-mobile-2.html>
- Castro, L. A., Beltrán, J., Perez, M., Quintana, E., Favela, J., Chávez, E., ... Navarro, R. (2014). Collaborative Opportunistic Sensing with Mobile Phones. *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, 1265–1272. <https://doi.org/10.1145/2638728.2638814>
- Chawda, R. K., & Thakur, G. (2016). Big data and advanced analytics tools. *2016 Symposium on Colossal Data Analysis and Networking, CDAN 2016*. <https://doi.org/10.1109/CDAN.2016.7570890>
- Cloud, V., Gateway, E., With, C., Storage, C., Cloud, E., & Security, S. (2015). Secure Data in Cloud Storage. Retrieved from <https://www.vormetric.com/sites/default/files/sb-vormetric-cloud-encryption-gateway.pdf>
- Cognini, R., Gagliardi, R., & Polzonetti, A. (2013). Business management and mobile experience. *2013 IEEE International Conference on Industrial Engineering & Engineering Management*, 1. Retrieved from <http://search.ebscohost.com/login.aspx?direct=true&db=edb&AN=101235084&site=eds-live>
- Comito, C., Falcone, D., Talia, D., & Trunfio, P. (2013). A distributed allocation strategy for data mining tasks in mobile environments. *Studies in Computational Intelligence*, 446, 231–240. <https://doi.org/10.1007/978-3-642-32524-3-29>
- Data Quality Management: Talend Enterprise Data Quality Tools. (2017). Retrieved from <https://www.talend.com/products/data-quality/>
- Debashis De. (2015). Mobile Cloud Computing: Architectures, Algorithms and Applications. *Chapman and Hall/CRC*. <https://doi.org/10.1371/journal.pone.0027263>

- Desai, N. S. (2016). Mobile Cloud Computing in Business. *International Journal of Information Sciences and Techniques*, 6(1/2), 197–202. <https://doi.org/10.1109/MITP.2012.64>
- Dugerdil, P. (2013). Architecting mobile enterprise app: a modeling approach to adapt enterprise applications to the mobile. *Proceedings of the 2013 ACM Workshop on Mobile ...*, 9–14. <https://doi.org/10.1145/2542128.2542131>
- EMOTIV Epoc - 14 Channel Wireless EEG Headset. (2016). Retrieved from <https://www.emotiv.com/epoc/>
- Gaber, M. M., Krishnaswamy, S., Gillick, B., Altaiar, H., Nicoloudis, N., Liono, J., & Zaslavsky, A. (2013). Interactive self-adaptive clutter-aware visualisation for mobile data mining. *Journal of Computer and System Sciences*, 79(3), 369–382. <https://doi.org/10.1016/j.jcss.2012.09.009>
- Gemson Andrew Ebenezer, J., & Durga, S. (2015). Big data analytics in healthcare: A survey. *ARPJ Journal of Engineering and Applied Sciences*, 10(8), 3645–3650. <https://doi.org/10.1155/2015/370194>
- George Firican. (2017). The 10 Vs of Big Data. Retrieved from <https://upside.tdwi.org/Articles/2017/02/08/10-Vs-of-Big-Data.aspx>
- Haghighi, P. D., Krishnaswamy, S., Zaslavsky, A., Gaber, M. M., Sinha, A., & Gillick, B. (2013). Open Mobile Miner: A Toolkit for Building Situation-Aware Data Mining Applications. *Journal of Organizational Computing and Electronic Commerce*, 23(3), 224–248. <https://doi.org/10.1080/10919392.2013.807713>
- He, Y., Yu, F. R., Zhao, N., Yin, H., Yao, H., & Qiu, R. C. (2016). Big Data Analytics in Mobile Cellular Networks. *IEEE Access*, PP(99), 1. <https://doi.org/10.1109/ACCESS.2016.2540520>
- Hönig, T., Eibel, C., Kapitza, R., & Schröder-preikschat, W. (2011). SEEP : Exploiting Symbolic Execution for Energy-Aware Programming, 58–62.
- Huang, D., Wang, P., & Niyato, D. (2012). A dynamic offloading algorithm for mobile computing. *IEEE Transactions on Wireless Communications*, 11(6), 1991–1995. <https://doi.org/10.1109/TWC.2012.041912.110912>
- Johnson, N. F., & Jajodia, S. (1998). Exploring steganography: Seeing the unseen. *IEEE Computer*, 31(2), 26–34. <https://doi.org/10.1109/MC.1998.4655281>
- Kargupta, H., Park, B., Pittie, S., Liu, L., Kushraj, D., & Sarkar, K. (2002). MobiMine: Monitoring the stock market from a PDA. *ACM SIGKDD Explorations Newsletter*, 3(2), 37–46. Retrieved from citeseer.ist.psu.edu/kargupta01mobimine.html
- Kchaou, H., Kechaou, Z., & Alimi, A. M. (2015). Towards an Offloading Framework based on Big Data Analytics in Mobile Cloud Computing Environments. *Procedia Computer Science*, 53, 292–297. <https://doi.org/10.1016/j.procs.2015.07.306>

- Kemp, R., Palmer, N., Kielmann, T., & Bal, H. (2012). Cuckoo: a computation offloading framework for smartphones. *Mobile Computing, Applications, ...*, 59–79. https://doi.org/10.1007/978-3-642-29336-8_4
- Kim, J. (2012). Design and evaluation of mobile applications with full and partial offloadings. *Advances in Grid and Pervasive Computing*, 172–182. Retrieved from <http://www.springerlink.com/index/L2058356342111P1.pdf>
- Kiukkonen, N., Blom, J., Dousse, O., Gatica-Perez, D., & Laurila, J. (2010). Towards rich mobile phone datasets: Lausanne data collection campaign. *Proceedings ACM International Conference on Pervasive Services (ICPS)*. Retrieved from <http://www.idiap.ch/~gatica/publications/KiukkonenBlomDousseGaticaLaurila-icps10.pdf>
- Kumar, K., & Lu, Y. H. (2010). Cloud computing for mobile users: Can offloading computation save energy? *Computer*, 43(4), 51–56.
- Lin, C. H., Hsiu, P. C., & Hsieh, C. K. (2014). Dynamic backlight scaling optimization: A cloud-based energy-saving service for mobile streaming applications. *IEEE Transactions on Computers*, 63(2), 335–348. <https://doi.org/10.1109/TC.2012.210>
- Liu, W., & Park, E. K. (2014). Big Data as an e-Health Service. *2014 International Conference on Computing, Networking and Communications (ICNC)*, 982–988. <https://doi.org/10.1109/ICCNC.2014.6785471>
- Loomba, R., Shi, L., Jennings, B., Friedman, R., Kennedy, J., & Butler, J. (2015). Energy-aware collaborative sensing for multiple applications in mobile cloud computing. *Sustainable Computing: Informatics and Systems*, 8, 47–59. <https://doi.org/10.1016/j.suscom.2015.09.001>
- Masud, M., Serhani, M., & Navaz, A. (2016). Resource-Aware Mobile-Based Health Monitoring. *IEEE Journal of Biomedical and Health Informatics*, XX(X), 1–1. <https://doi.org/10.1109/JBHI.2016.2525006>
- Monitoring the Battery Level and Charging State | Android Developers. (2017). Retrieved from <https://developer.android.com/training/monitoring-device-state/battery-monitoring.html>
- MUSE | Meditation Made Easy. (2016). Retrieved from <http://www.choosemuse.com/>
- Nettleton, D. (2014). *Commercial Data Mining. Commercial Data Mining*. <https://doi.org/10.1016/B978-0-12-416602-8.00014-5>
- Orsini, G., Bade, D., & Lamersdorf, W. (2016). Computing at the Mobile Edge: Designing Elastic Android Applications for Computation Offloading. *Proceedings - 2015 8th IFIP Wireless and Mobile Networking Conference, WMNC 2015*, 112–119. <https://doi.org/10.1109/WMNC.2015.10>
- Patil, P., Hakiri, A., & Gokhale, A. (2016). Cyber Foraging and Offloading Framework for Internet of Things. *2016 IEEE 40th Annual Computer Software and*

- Applications Conference (COMPSAC)*, 359–368.
<https://doi.org/10.1109/COMPSAC.2016.88>
- Physionet.org. (2000). CHB-MIT Scalp EEG Database. Retrieved from
<https://physionet.org/pn6/chbmit/>
- Quwaider, M., Jararweh, Y., Al-Alyyoub, M., & Duwairi, R. (2015). Experimental Framework for Mobile Cloud Computing System. *Procedia Computer Science*, 52, 1147–1152. <https://doi.org/10.1016/j.procs.2015.05.149>
- Routai, H., Badidi, E., Elmachour, M., Sabir, E., & Elkoutbi, M. (2014). Modeling and evaluating a cloudlet-based architecture for Mobile Cloud Computing. *2014 9th International Conference on Intelligent Systems: Theories and Applications, SITA 2014*, (Mc). <https://doi.org/10.1109/SITA.2014.6847290>
- Sarathchandra Magurawalage, C. M., Yang, K., Hu, L., & Zhang, J. (2014). Energy-efficient and network-aware offloading algorithm for mobile cloud computing. *Computer Networks*, 74(PB), 22–33. <https://doi.org/10.1016/j.comnet.2014.06.020>
- Satyanarayanan, M., Bahl, P., Cáceres, R., & Davies, N. (2009). The case for VM-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4), 14–23. <https://doi.org/10.1109/MPRV.2009.82>
- Scerri, P., Vincent, R., & Mailler, R. (2006). *Coordination of large-scale multiagent systems. Coordination of Large-Scale Multiagent Systems*. <https://doi.org/10.1007/0-387-27972-5>
- Schatz, B. R. (2015). National Surveys of Population Health: Big Data Analytics for Mobile Health Monitors. *Big Data*, 3(4), 219–229. <https://doi.org/10.1089/big.2015.0021>
- Serhani, M. A., El Menshawy, M., Benharref, A., & Navaz, A. N. (2016). Real time EEG compression for energy-aware continuous mobile monitoring. *Proceedings of the International Conference on Microelectronics, ICM, 2016–March*, 291–294. <https://doi.org/10.1109/ICM.2015.7438046>
- Serhani, M. A., Kassabi, H. T. El, Taleb, I., & Nujum, A. (2016). An Hybrid Approach to Quality Evaluation Across Big Data Value Chain. <https://doi.org/10.1109/BigDataCongress.2016.65>
- Shoeb, A., & Guttag, J. (2010). Application of Machine Learning To Epileptic Seizure Detection. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 975–982. <https://doi.org/10.1016/j.jneumeth.2010.05.020>
- Sivaraman, E., & Manickachezian, R. (2014). High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing Using Hadoop. *2014 International Conference on Intelligent Computing Applications*, 32–36. <https://doi.org/10.1109/ICICA.2014.16>
- Sun, J., & Reddy, C. (2013). Big data analytics for healthcare. *SIAM International*

- Conference on Knowledge Discovery and Data*, 1525. <https://doi.org/10.1145/2487575.2506178>
- Sun, Y., Song, H., Jara, A., & Bie, R. (2016). Internet of Things and Big Data Analytics for Smart and Connected Communities. *IEEE Access*, 4, 1–1. <https://doi.org/10.1109/ACCESS.2016.2529723>
- Swanson, S. (2016). *Self-Service Analytics - Making the Most of Data Access*. O'Reilly - Compliments of Talend.
- Unhelkar, B., & Murugesan, S. (2010). The enterprise mobile applications development framework. *IT Professional*, 12(3), 33–39. <https://doi.org/10.1109/MITP.2010.45>
- WorldBank, & InfoDev. (2012). *Information and Communications for Development 2012: Maximizing Mobile. 2012 Information and Communications for Development*. <https://doi.org/10.1596/978-0-8213-8991-1>
- Xia, Y., Liu, Y., Tan, C., Ma, M., Guan, H., Zang, B., & Chen, H. (2015). TinMan: Eliminating Confidential Mobile Data Exposure with Security Oriented Offloading. *Proceedings of the Tenth European Conference on Computer Systems - EuroSys '15*, 1–16. <https://doi.org/10.1145/2741948.2741977>
- Yazti, D. Z., & Krishnaswamy, S. (2014). Mobile big data analytics: Research, practice, and opportunities. *Proceedings - IEEE International Conference on Mobile Data Management*, 1, 1–2. <https://doi.org/10.1109/MDM.2014.73>
- Zaslavsky, A., Jayaraman, P. P., & Krishnaswamy, S. (2013). ShareLikesCrowd: Mobile analytics for participatory sensing and crowd-sourcing applications. *Proceedings - International Conference on Data Engineering*, 128–135. <https://doi.org/10.1109/ICDEW.2013.6547440>
- Zhang, X., & Jeon, W. (2012). Elastic HTML5: Workload Offloading Using Cloud-Based Web Workers and Storages for Mobile Devices. *Mobile Computing*, 1–10. <https://doi.org/10.1007/978-3-642-29336-8>

List of Publications

Masud, M., Serhani, M., & Navaz, A. (2016). Resource-Aware Mobile-Based Health Monitoring. *IEEE Journal of Biomedical and Health Informatics*, XX(X), 1–1. <http://doi.org/10.1109/JBHI.2016.2525006>

Serhani, M. A., El Menshawy, M., Benharref, A., & Navaz, A. N. (2016). Real time EEG compression for energy-aware continuous mobile monitoring. *Proceedings of the International Conference on Microelectronics, ICM, 2016–March*, 291–294. <http://doi.org/10.1109/ICM.2015.7438046>

Serhani, M. A., Kassabi, H. T. El, Taleb, I., & Nujum, A. (2016), An Hybrid Approach to Quality Evaluation Across Big Data Value Chain. <http://doi.org/10.1109/BigDataCongress.2016.65>

Navaz, A. N., Elfadil Mohammed, Serhani, M. A., & Nazar zaki. (2016). The use of data mining techniques to predict mortality and length of stay in an ICU. In: *Innovations in Information Technology (IIT), 2016 12th International Conference*. <https://doi.org/10.1109/INNOVATIONS.2016.7880045>

Appendix

Code Snippets

Few resource optimization code snippets deployed in the developed Android application are listed below.

// Wi-Fi

```
Boolean prefWiFi = getBoolPref("pref_switch3", this);
WifiManager wifi = (WifiManager) getSystemService(Context.WIFI_SERVICE);
boolean b=wifi.isWifiEnabled();
if(prefWiFi){
    if(!b) {
        wifi.setWifiEnabled(true); // "WiFI Enabled "
    }
}else{
    if(b) {
        wifi.setWifiEnabled(false); // "WiFI Disabled "
    }
}
```

// Bluetooth

```
public static boolean setBluetooth(boolean enable) {
    BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    boolean isEnabled = bluetoothAdapter.isEnabled();
    if (enable && !isEnabled) {
        return bluetoothAdapter.enable();
    }
    else if(!enable && isEnabled) {
        return bluetoothAdapter.disable();
    }
}
```

```

        return isEnabled;
    }

    // Cellular Data

    public boolean turnMobileDataOn(Boolean toSet) {
        ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo =
            connMgr.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
        boolean enabled = networkInfo.isConnected();
        if (toSet && !enabled) {
            showDialogMobData("Enable Mobile Data");
        }
        else if (!toSet && enabled){
            showDialogMobData("Disable Mobile Data");
        }
        enabled = networkInfo.isConnected();
        return enabled;
    }

    // GPS

    public boolean turnGPSON(Context context) {
        LocationManager service;
        service = (LocationManager) context.getSystemService(LOCATION_SERVICE);
        boolean enabled = service
            .isProviderEnabled(LocationManager.GPS_PROVIDER);
        if (!enabled) {
            showDialogGPS("Enable GPS");
        }
        enabled = service
            .isProviderEnabled(LocationManager.GPS_PROVIDER);
        return enabled;
    }

```

//Screen Backlight

```

Boolean prefBlight = getBoolPref("pref_switch2",this);
WindowManager.LayoutParams params = this.getWindow().getAttributes();
if(prefBlight) {
    /* Turn on: */
    params.flags = WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON;
    //TODO restoring from original value
    params.screenBrightness = 0.9f;
    this.getWindow().setAttributes(params);
}else {
    /** Turn off: */
    params.flags = WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON;
    //TODO Store original brightness value
    params.screenBrightness = 0.1f;
    this.getWindow().setAttributes(params);
}

```

// Account Auto Sync

```

Boolean prefAccSync = getBoolPref("pref_switch6",this);
if(prefAccSync) {
    ContentResolver.setMasterSyncAutomatically(true); // Auto Sync On
}else {
    ContentResolver.setMasterSyncAutomatically(false); // Auto Sync Off
}

```

// Clear Cache

```

public static void deleteCache(Context context) {
    try {
        File dir = context.getCacheDir();
        deleteDir(dir);
    } catch (Exception e) {}
}

```



```
public static boolean deleteDir(File dir) {  
    if (dir != null && dir.isDirectory()) {  
        String[] children = dir.list();  
        for (int i = 0; i < children.length; i++) {  
            boolean success = deleteDir(new File(dir, children[i]));  
            if (!success) {  
                return false;  
            }  
        }  
        return dir.delete();  
    } else if (dir != null && dir.isFile()) {  
        return dir.delete();  
    } else {  
        return false;  
    }  
}
```