Big Data Based Architecture of the ADL Recognition System

Yongan Liao
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/creativecomponents

Part of the Computer and Systems Architecture Commons

Recommended Citation
79.
https://lib.dr.iastate.edu/creativecomponents/79

This Creative Component is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Creative Components by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Big Data Based Architecture of the ADL Recognition System

by

Yongan Liao

A technical report submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Carl K. Chang, Major Professor
Simanta Mitra
Kevin Liu

Iowa State University
Ames, Iowa
2018

Copyright © Yongan Liao, 2018. All rights reserved.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES AND TABLES</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
</tr>
<tr>
<td>ABSTRACT</td>
</tr>
<tr>
<td>CHAPTER 1. INTRODUCTION</td>
</tr>
<tr>
<td>CHAPTER 2. THE ADL RECOGNITION SYSTEM</td>
</tr>
<tr>
<td>CHAPTER 3. BIG DATA BASED SOLUTION</td>
</tr>
<tr>
<td>Q1: Data Streaming</td>
</tr>
<tr>
<td>Q2: Distributed File System</td>
</tr>
<tr>
<td>Q3: Computation</td>
</tr>
<tr>
<td>Q4: Data Analysis</td>
</tr>
<tr>
<td>Q5: Data Storage</td>
</tr>
<tr>
<td>Q6: Data Presentation</td>
</tr>
<tr>
<td>CHAPTER 4. THE ARCHITECTURE OF THE SYSTEM</td>
</tr>
<tr>
<td>Grab Data</td>
</tr>
<tr>
<td>App Upload Directly</td>
</tr>
<tr>
<td>Data Center</td>
</tr>
<tr>
<td>Data Filter Module</td>
</tr>
<tr>
<td>Machine Learning Module</td>
</tr>
<tr>
<td>Platform</td>
</tr>
<tr>
<td>CHAPTER 5. PRESENTATION</td>
</tr>
<tr>
<td>Development Work</td>
</tr>
<tr>
<td>Evaluation</td>
</tr>
<tr>
<td>REFERENCES</td>
</tr>
<tr>
<td>APPENDIX A. SENSOR MODULE IN SMARTPHONE AND INFORMATIVE DATA</td>
</tr>
<tr>
<td>APPENDIX B. DATABASE FORMAT IN THE SERVER AND RAW DATA PACKAGES</td>
</tr>
<tr>
<td>APPENDIX C. VISUALIZED INTERFACE IN THE SERVER</td>
</tr>
</tbody>
</table>
**LIST OF FIGURES AND TABLES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td><em>An Overview of the Simplified Application Environment.</em></td>
<td>2</td>
</tr>
<tr>
<td>Figure 2</td>
<td><em>An Overview of the Overall Application Environment.</em></td>
<td>3</td>
</tr>
<tr>
<td>Figure 3</td>
<td><em>The Big Data Based Architecture of the ADL Recognition System.</em></td>
<td>12</td>
</tr>
<tr>
<td>Figure 4</td>
<td><em>The Way to Grab Data in the Data Center.</em></td>
<td>13</td>
</tr>
<tr>
<td>Figure 5</td>
<td><em>Pseudocodes of Workflows in Training the Label WorkOrSleep.</em></td>
<td>15</td>
</tr>
<tr>
<td>Figure 6</td>
<td><em>Pseudocodes of Workflows in Training the Label BodyAction.</em></td>
<td>16</td>
</tr>
<tr>
<td>Figure 7</td>
<td><em>An Overview of Platforms in the Big Data Based Architecture.</em></td>
<td>17</td>
</tr>
<tr>
<td>Figure 8</td>
<td><em>An Example: Status of Body Action for ne Day in Seconds.</em></td>
<td>19</td>
</tr>
<tr>
<td>Figure 9</td>
<td><em>An Example: Status of Work or Sleep for One Day in Seconds.</em></td>
<td>19</td>
</tr>
<tr>
<td>Figure 10</td>
<td><em>An Overview of the Big Data Based Technologies That Can Be Used in the System.</em></td>
<td>21</td>
</tr>
</tbody>
</table>

| Table 1 | *Comparison Between Existing System and Big Data Based System.*             | 5    |
ACKNOWLEDGMENTS

I would like to thank my committee chair, Dr. Carl K. Chang, and my committee members, Dr. Simanta Mitra, and Dr. Kevin Liu, for their guidance and support throughout the course of this research.

In addition, I would also like to thank my friends, colleagues, the department faculty and staff for making my time at Iowa State University a wonderful experience. I would like to give a special thanks to Yunfei Feng, who gives me many technical supports and suggestions on my work. I want to also offer my appreciation to those who were willing to help and provide suggestions in the lab.

Finally, I would like to thank my family especially my father, for their consistent support and encouragement in my life. Without them, this paper would not have been possible.
ABSTRACT

In the age of the Internet of Things (IoT), every motion in our daily life can be captured and modulated to a digital data with smart portable devices. These data are very valuable in the presentation of someone’s living status, and in analyzing its health condition. Based on that, suggestions by professionals can be given directionally to improve the living quality. An arising issue is that, in this way the amount of data in the filter is big and the velocity of data flow is high. The original structure is strainful in handling such a large amount of data or such detailed data. An urgent requirement is to build a structure to support big data in the aspect of grabbing, filtering, analysis, and presentation.

The ADL Recognition System collects information from elderly people, analyzes their behaviors, and presents them in a visualized way to specific users. It aims to provide a better nursing service for elderly people and a convenience in assessing health condition or nursing level for nurses and doctors.

My work is to design and implement a big data based architecture of the ADL Recognition System so that it can accept more users and data flowing in. Besides, the architecture will be expandable in computation and storage and adaptive to the scale of the real application environment. The necessity and methodology of importing big data based architecture will be justified in each process of data filtering from the aspects of technologies.
CHAPTER 1. INTRODUCTION

ADL stands for Activities of Daily Living, a term refers to the activities that a person need to do daily on its own. It commonly includes eating, bathing, dressing, working, walking, and etc. In the view of nursing science, the extent to which a person can perform ADL is considered an important measurement of his functional health status [1]. There are many types of researches from nursing scientists on how to assess and evaluate ADL behaviors since it expresses the extent of the requirement of nursing assistants. For example, if the patients are assessed through ADL information that they cannot move on their own, they will need the assistance in mobility from nurses. The extent to which the patients take a bath on their own decides whether they need bath assistances or bed baths.

Specifically, consider the application environment in gerontology on retirement communities or nursing communities. In many nursing communities, healthcare providers often collect ADL information on elderly people to determine what level of health care they need. Without the help of technologies, they have to follow their service targets all day in the interval of several hours to collect ADL information and assess ADL abilities. Inspired by that, we want to introduce technologies in this process to help the caregiver collect data conveniently as well as give the elderly people a better user experience. Recently these years, numerous sensors are installed in research for ADL behavior recognition purpose. For example, door contacts [2,3], infrared sensors [2], button sensors over lamp switches [2,3], microphones near faucets [2,3], and other vision technology [4]. Besides, in the same purpose, the ADL Recognition System is introduced with enhanced user experience [5].
CHAPTER 2. THE ADL RECOGNITION SYSTEM

Figure 1 If we simplify the ADL Recognition System as a box, the application environment with users can be presented. The arrow means the direction of data flow.

Figure 1 shows the basic application environment of the ADL Recognition System. The system divides their potential users into two groups: Observees, and Observers. The observees are targets who need to be observed and record on their ADL behaviors, including elderly people in nursing communities, children at home, and etc. The observers are those who have the intention to realize the condition of the observees, including family members who care about their parents or children, doctors who give professional healthy suggestions, observees themselves, and etc.

In the observees’ sides, there is an ADL Recorder App installed on the smartphones. The ADL Recorder App is user-friendly and functional in collecting instant ADL information of users. Since there are various embedded sensors bundled with a modern smartphone, the app is able to capture users’ multidimensional data, reflecting both behavioral and environmental contexts [9].

In the observers’ sides, users can observe the condition of observees from both websites and ADL Reviewer App. After data fusion and machine learning process in the
system, the result of classification will be presented as charts, bars, and other visualized forms. The presentation will be understandable for people without any knowledge of computer or statistic.

From Figure 2, we can see more details and all possible functions in the ADL Recognition System (Server). System Administrators manage the basic functions of the system, including grabbing data from observees, filtering into informative data, storing data in the cloud, subscribing to observers. In the hardware level, they need to manage computers...
for consistent computation and storage and maintain service for the system. Living Domain Experts and Data Analysts cooperate to extract valuable information by applying machine learning approaches. These approaches can classify various usable labels from data in the warehouse as features. These labels, as high-level information, can be stored or subscribe to the ADL Reviewer App through a broadcast agent.

Compared with different kinds of traditional sensors, the ADL Recognition System gives users a better experience in cost and comfort. Since the smartphone becomes a popular device with which users are carrying every day, they do not need to buy additional devices for the system. In order to capture data in dimensions, lots of sensors need to be carried on, which costs a lot and leads to the uncomfortable user experience. In the ADL Recognition System, modern smartphones are proved to have the ability to collect and recognize ADL information according to sensors embedded, including the microphone, Wi-Fi scan module, heading orientation of the device, light proximity, step detector, accelerometer, gyroscope, magnetometer, and time stamp. Appendix A shows some sensor modules embedded in the modern smartphone and the type of informative data which can be collected from the smartphone.
CHAPTER 3. BIG DATA BASED SOLUTION

The ADL Recognition System has been published and implemented in a way for research [5]. However, when it applies to the real market especially with a large number of users, it will counter some challenges. For example, with increased users the size of multidimensional data that represents ADL behavior and is extracted from the sensors listed above in the smartphone is large. For filtering this size of data, data analysis is in high computation. Since the system receives data without suspension, a large amount of historical data will need to be stored. These characteristics make the existing framework hard to maintain. Inspired by the fact, a big data based architecture of the ADL Recognition System is developed.

<table>
<thead>
<tr>
<th></th>
<th>Existing System [5]</th>
<th>Big Data Based System</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technology</strong></td>
<td>Python, Weka</td>
<td>Kafka, Spark MLlib</td>
</tr>
<tr>
<td><strong>Running Environment</strong></td>
<td>Local PC</td>
<td>Cloud Server Platform</td>
</tr>
<tr>
<td><strong>Data Volume</strong></td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td><strong>Computing Capability</strong></td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

Table 1 Comparison between existing system and big data based system. Big data based system have the similar application environment with the existing system but support high data volume and computing capability.

Big-data tends to be a popular term these years because in the age IoT data with various types are easily accessible. Here I list some characteristics of big data based architecture.
(1) Volume: The architecture should have the ability to grab and process a large amount of data generated from different sources. In other words, the scope is adaptive and expandable within the architecture to support the size of data.

(2) Velocity: The architecture should meet the requirement of fast data flow in data transfer, including collection, processing, and analysis. The system needs to accept and respond quickly.

(3) Variety: Type of data inflow can be multiple, including text, image, audio, video, and other structured or unstructured format.

(4) Value: Since the capability of computation is extensional, complicated and high-computational machine learning algorithm can be applied. It leads to the fact that prediction tends to be accurate, and additional hidden value will be discovered.

In software development, a big-data framework contains lots of technologies in different aspects. These technologies are changing so fast that no certain patterns can be followed. Some of them stand up then disappear, because of more effective new technologies. For better illustration, technical details of big data based architecture of the system will be introduced in a way of research questions and solutions

**Q1: Data Streaming**

Assuming the system will have 1,000,000 users, and each user instantly uploads a record of size 2K (or more). The instant throughput of the server could be at least 2G at the peak. For most case, HTTP/TCP will confirm that the record files will be received by the server. But in this high throughput, the server may not be powerful enough to respond every connection instantly even in the multi-thread mode. **How can the server receive these records without congestion so that every record file will not get lost?**
Steaming technologies need to be applied. In the old stage, ActiveMQ and RabbitMQ are two popular streaming technologies which are mature and reliable. Kafka [11], a younger steaming technology designed for the large volume of data, has better performance than these two and quickly becomes a popular streaming technology in big data framework. Actually, most business companies design their own streaming technology for better performance for specific purposes. For example, RocketMQ is an open source technology developed by Alibaba Company. Kafka is initially developed and deployed by LinkedIn Company.

For the ADL Recognition System, Kafka is a better choice for its high performance and compatibility in a distributed system. However, technology solution is not the only and the best choice when considering the application environment. Deployment of Kafka can ensure that every record files will not get lost in high throughput, but unavoidably it requires high hardware condition in storage and computation for the server.

Another solution in engineering is to build a data center to collect data from people in the same community through a local network. Then local data centers compress the data collected and upload to the server routinely. Considering the assumption in the question, if there is 1,000 data center, the size of data accepted by the server from the data center per day will be 2K*1M*150*0.9/1000 = 270M. The instant throughput of the data center is at most 1000*2K = 2M. Much less burden for the server.

The two solutions, in technology and in engineering, can combine to serve the users in different application environments.

Q2: Distributed File System

In order to support a large amount of data flow, the server needs to be built on a powerful computer. But the high-performance computer is expensive and inconvenient for use. **How can the server be organized and deployed in big data framework?**
Distributed computing has been published to solve that problem. Several computers group together and virtualize a single file system to support high computation. This is the main contribution of Hadoop. The file system visualized is called Hadoop Distributed File System (HDFS). HDFS makes it possible for small-and-medium-sized companies or communities to deploy their own server to support their project. HDFS is extensional in computation and storage by adding more CPUs and Memories, so is flexible in scale for deployment.

For now, the name Hadoop is not only for distributed computation but for the base of Big Data Framework. Many technologies in other aspects are developed on top of it, including data streaming, analysis, storage, and etc. Most of the technologies discussed in this report, like Kafka, can be built on HDFS. They all belong to the Hadoop Ecosystem.

Recently these years, Cloud Computing becomes popular and is a good alternative choice for deploying server. Microsoft Azure Cloud, Amazon Web Service Cloud (AWS), and Google Cloud are thes3 famous cloud platform. All of them support the Hadoop Ecosystem, support most technologies related to big data framework. The benefit is that developers do not need to care about the hardware layer, and cloud platform will assign suitable hardware for the web application. The price will be the main concern of the cloud service.

**Q3: Computation**

The computation work in the server is quite heavy in the big data framework. **How can the server do data transformation, like mapping and grouping by, effectively?**

Computation can be divided into two groups: offline computation, and streaming computation.
Offline computation is computing on local data and return a local result. Popular offline computation engine includes Hadoop MapReduce and Spark [12]. Spark is a new engine and proved to have better performance than MapReduce [7]. The principle of the Spark engine is to analyze the codes and optimize the flow of the data processing before actually running the codes. In this way, Spark can process data effectively by allocating computing resources properly.

Streaming computation is computing on a streaming data and return a streaming data. Storm, Spark Streaming, S4, and Heron are some popular streaming computation engine. Spark Streaming and Spark is similar in principle. Both of them is developed in Scala, a programming language friendly to data scientists.

Q4: Data Analysis

In this part of Data Analysis, I want to focus on Machine Learning engine in big data framework. As discussed above, most applications of big data framework are built on a distributed system. Offline machine learning library, like Weka, is not powerful enough to allocate resources in the distributed system to finish the jobs. **So how can the server run machine learning algorithms in a distributed system environment effectively?**

Apache Mahout is a machine learning engine which is developed specifically for running machine learning algorithm in HDFS. But now it is ignored compared with another recently frequently-used machine learning engine, Spark MLlib [13]. Spark MLlib is based on Spark engine and is famous for its high speed in running machine learning algorithms. Spark MLlib is 100x faster than Hadoop MapReduce in performing Logistic regression [8]. Spark MLlib have interfaces for common machine learning algorithm, including classification, regression, decision tree, clustering, and etc. Without caring about
compilation details, building workflow related to the application is the main concern the same as other machine learning tools. It even supports integrate existing workflows from other data science tools, such as R.

**Q5: Data Storage**

How can the server store history records so that read and write operation runs effectively?

Normally a relational database is a good choice, like MySQL, SQL Server, and Oracle Server. Relational database technologies are mature in organizing structured data effectively. The main question is to reduce the cost of accessing the database in a distributed system. JDBC/ODBC is the most general library for accessing the database, but a faster technology is needed for high velocity of data flow. Hive, Pig, and Spark SQL are introduced. Spark SQL is often used in accessing data for computation in Spark, or Spark MLlib. Hive and Pig are more general in reading from or writing to the database in HDFS.

If the records include unstructured data, like image, audio, and video, a non-relational database is needed. The popular non-relational database includes Redis, Apache Cassandra, and MongoDB. Redis is a light database with the fastest performance among them. Cassandra and MongoDB are more stable in storing a large amount of data, with the flexibility to handle SQL and non-SQL.

**Q6: Data Presentation**

How can the server present big size of data in complicated structures in a user-friendly view?

To present data in a website, Javascript is a good technology in data visualization. There are many alternative data visualization tools based on Javascript, including Chart.js,
D3.js, Google Chart, and etc. These tools work well and have the ability to perform an understandable and artful visualization, as long as the developers are experienced with them.
CHAPTER 4. THE ARCHITECTURE OF THE SYSTEM

Figure 3 presents the overview of the big data based architecture I have designed for the ADL Recognition System. Several big data based technologies have been applied to support a high volume of data flow. From the figure, we can see that data is grabbed from three sources to the server framework. Then the data flow through the Data Filter Module and the Machine Learning Module and finally is visualized in the terminal.

Figure 3 An overview of big data based architecture of the ADL Recognition System. There are three ways to grab data. The server can be divided into Data Filter Module, Machine Learning Module, and Visualization Module.

Grab Data

There are three ways to grab data in the architecture, including Upload Manually, App Upload Directly, and Data Center. Upload Manually means that users can log in to our server and upload their ADL data captured by the ADL Recorder APP. This is an intuitive way to save your formatted data in the server and supports a large amount of data but is manual and inconvenient for use. Then I will introduce the other two more convenient ways to grab data.
**App Upload Directly**

After the ADL Recorder APP produce a record, it will get an HTTP/TCP connection with our server and upload the record instantly. The main issue here is that if a large number of users intend to upload their data, the server will have heavy connection pressure and finally lead to congestion if the server is not powerful to handle every connection. So the server needs a streaming layer to save the data temporarily so that the data will not get lost. In this way, Kafka has been applied in the server for streaming to prevent upload congestion from app to the server. The server can asynchronously accept every incoming connection, put the data in the Kafka Cluster, and response success. At the same time, the Data Filter Module can take the data from the Kafka Cluster and continue the filtering work. Kafka separates data in process of handling connections and filtering, and effectively control the velocity of high volume data flow.

![Diagram of data flow](image)

*Figure 4 We can build a local data center to collect data from users living in neighbor. The data centers will compress the data collected and upload to the server routinely.*
**Data Center**

Considering if the amount of users is million level or larger, the server will still face pressure with Kafka as a streaming layer. Under this pressure, the cost to build such a powerful server is high. Motivated by that, a cheaper approach is to build a physical streaming layer. Data Center means building a local data center for collecting data from users in neighbor as a buffer. The data centers will compress the data collected and upload to the server routinely. This approach is motivated by the fact that part of elderly people live in the retirement community or nursing community where the data center can be built in for service. This is an engineering solution to solve the streaming problem by applying a physical streaming layer.

Compared with App Upload Directly, this solution only suitable for certain application environment, such as the retirement community. Because it takes time for data center to buffer data, the streaming is not instant and users have to wait for their record to be stored in the server. But the worth benefit is that it has a much lower cost to support high volume streaming in high privacy and security.

**Data Filter Module**

The function of this module is to filter raw data from the sensors in smartphones into a recognized format and then save them into the database. Modern smartphones can capture multidimensional information from different sensor modules, including Accelerometer Module, Orientation Module, GPS Module, Battery Module, Time Module, Weather Module, Cellular Module, Phone Status Module, Geomagnetic Field Module, and etc. These sensor modules can capture various types of data, such as the light condition of the environment, motion, location, and operations on the phone, which reflect the living condition of the users.
Appendix A makes a conclusion of the informative data we can capture from some popular smartphones.

In the experiment, I extract part of the features and save them into a MySQL database. Appendix B presents the database format of records in the server. The data includes static status data which is directly captured from sensors, like light, axis acceleration, angle, location, and phone status, and motion data which is calculated as a representation of changing, like moving, turning, lightChanging, and acceleration.

**Machine Learning Module**

This module do machine learning analysis on the data retrieved from the database. The main technology is Spark and Spark MLlib. In chapter 3, we have discussed the benefit of Spark MLlib and related technologies running on a distributed architecture. In this module, both supervised and unsupervised machine learning algorithm can be applied. For unsupervised machine learning, the model adjustment is applied when the prediction hit rate is decreased.

- **Label**: workOrSleep
- **Features**: hour, light, moving, screenOn, turning, earplug, actionLabel

i. actionLabel => indexedActionLabel
ii. new workOrSleepFeatures(hour, light, moving, screenOn, turning, earplug, indexedActionLabel)
iii. workOrSleepFeatures => indexedWorkOrSleepFeatures
iv. workOrSleep => indexedWorkOrSleep
v. DecisionTree(indexedWorkOrSleepFeatures, indexedWorkOrSleep)
vi. indexedWorkOrSleep => workOrSleep

Figure 5 *Pseudocodes of workflows in training the label workOrSleep*
In the experiment, the Decision Tree Algorithm is applied to create two labels: workOrSleep, and bodyAction. The features for the Label workOrSleep are hour, light, moving, screenOn, turning, earplug, and actionLabel. The features for the Label bodyAction are x-Acceleration, y-Acceleration, z-Acceleration, moving, angle, azimuth, pitch, and roll.

As R or other data science tools, I have created a workflow to pre-process data, format data, train models, and evaluate models. Figure 5 and 6 present the pseudocodes of the workflows.

The details about the experiment data and the corresponding result will be shown in Section 5.

**Platform**

Figure 7 shows the platforms in the architecture, which provide a running environment and a development framework. Play Framework is a web application framework usable in Scala and Java to handle web connection. It follows the model-view-controller (MVC) architectural pattern and aims to optimize developer productivity by using convention over configuration, hot code reloading, display of errors in the browser, and constraint of stateless development for the server. There are some characteristics that attract us to choose it. It is lightweight, stateless, web-friendly, and supportable for the highly-
scalable application. The most important reason is that it is friendly and efficient to Spark Ecosystem in data transforming and analysis because of the deployment of Scala.

Docker Container is provided by Docker, which is a tool to make the web application flexible to migrate from a development system to an application system. It functions like a virtual machine containing applications and its running environment except for the operating system, so Docker is lightweight and cost less resource.

Azure Cloud is a cloud platform providing hardware and service to manage them. It constitutes a virtual file system for expandable computation capability and provides a separated database for high volume storage. It is easy to set up server configuration on it. Besides, it sets up a distributed computing environment and supports Hadoop Ecosystem. It means that many big data based technologies based on the distributed system can run on it. Developers only need to care about the price for deployment of the server. For the ADL Recognition System in the experiment, I make a simple calculation for the price. A web application using an instance with 4 cores, 7GB RAM, 50GB storage needs at least 0.40$ per
hour. An SQL Database supporting 56GB memory and 32GB per month storage needs at least 0.60$ per hour. This is the basic price for simple function, if the amount of users continues to expand, the price will be more.
CHAPTER 5. PRESENTATION

Development Work

For presenting the architecture, I develop a web-based framework to realize the basic function of the ADL Recognition System. The data is captured from the ADL Recorder APP by [5]. The app will produce a record for 15 seconds after every 3 minutes or when a giant fluctuation is detected. The data package contains 6 months of the ADL data recording in real time and real living environment. Following the architecture described in Chapter 4, all the raw ADL data are imported to our server, filtered, and stored in the cloud database. For presentation, I create two labels: WorkOrSleep and BodyAction by applying the decision tree classifier algorithm. I create an interface for reviewing so that the user and the date in the presentation can be selected. The data package and data format in the database are presented in Appendix B. The interface for reviewing is presented in Appendix C.

In the terminal, I apply D3.js [10], a JavaScript visualization tool, to manipulate and display documents as results after machine learning. Figure 8 and 9 are two examples. The
whole circle represents one day of 24 hours. In each pie, the left side of notation is the label, and the right side is the corresponding time in seconds during which the user is performing the activity. For example, Sleeping: 11921s means that the user spends 11921 seconds on sleeping on that day. There is an Unrecorded label because some moments are not recorded or the records get lost. Different types of features are classified into the two labels. WorkOrSleep is determined by static status data, like record time, charging or not, and etc. BodyAction is determined by motion data, like data from accelerometers. The figures only show the classification result of one day of a user. Users can select the specific date they want to review. Here is just a representation of applying machine learning algorithm on the architecture. Since a large amount of multidimensional data are stored in the server, more labels can be created by domain experts and data analysts.

**Evaluation**

Now the big data based architecture of the ADL Recognition System has the ability to filter a large amount of data, as long as the system can be built on some powerful servers. From the experiment above, the amount of data we use for training is large, covering any changes for every three minutes of many sensors within 6 months. The three approaches to grab the data ensure that big data from different sources can flow in without congestion. And the data flow works well to lead to the classification results. Since we can set up our server on the cloud platform, the capability of computation is expandable, depending on the number of users and the capacity of data flow.
In the future, there will be more improvements in the architecture. Figure 10 shows some potential technologies that can be applied to the system. One concern is that, if there are some unstructured data type like audio and video it needs an unstructured database to store these data. The smartphone can record some audio files which give some possibilities to give a more accurate prediction. MongoDB can help save these files in the server and give efficient I/O service for further use. The other concern is that, the access to the database is in low efficiency especially in big data storage. In a distributed system, the data is stored in multiple computers. Hive and Pig can help manage the data scattered in different hardware as a central data warehouse in software level, and improve access efficiency. On improved big data based architecture, I am optimistic that users of the ADL Recognition System will have a better experience on the service.

Figure 10: An overview of the big data based technologies that can be used in the system. The non-filled rectangles mean the technologies I have applied to the system. The filled rectangles mean the technologies for future improvement.
REFERENCES


[8] "Spark runs workloads 100x faster / Logistic regression in Hadoop and Spark", http://spark.apache.org


[10] https://d3js.org


APPENDIX A. SENSOR MODULE IN SMARTPHONE AND INFORMATIVE DATA

Figure A.1 There are many sensor modules integrated in the modern smartphones. Here list some informative data we can directly capture from the sensors and deduced informative data inferred from other informative data.
APPENDIX B. DATABASE FORMAT IN THE SERVER AND RAW DATA PACKAGES

Figure B.1 In the experiment, I select some features that is captured from the ADL Recorder APP and save them in the database. The figure shows the database format of records in the server. Further, these features will be used in data analysis and presentation.

Figure B.2 In the experiment, raw data packages are used in our application. These data are recorded in real time and real living environment. The name of folders shows the date intervals and the user of the records.
APPENDIX C. VISUALIZED INTERFACE IN THE SERVER

Welcome to ADL Upload Data Service!

Upload single pair of files

Status File
Choose File No file chosen
Summit

Motion File
Choose File No file chosen

Upload multiple pairs of files

Status File
Choose Files No file chosen
Summit

Motion File
Choose Files No file chosen

Figure C.1 The figure shows the visualized interface for uploading the data packages. Besides, I have developed an interface for upload connection between the ADL Recorder APP and the server with Kafka streaming.

Welcome to ADL Analyze Data Service!

User name: 
Date: mm/dd/yyyy
Summit

Figure C.2 The figure shows the visualized interface for selecting records of user and data to analyze. After summitting, the server will search records in the database, analyze the data, and show the result as Figure 8 and Figure 9.