Int. J. Advance Soft Compu. Appl, Vol. 8, No. 2, July 2016
ISSN 2074-8523

Mobile Cloud Computing Architecture on Data Management for Big Data Storage

Nur Syahela Hussien[1,2], Sarina Sulaiman[1,2]
[1] UTM Big Data Centre
Universiti Teknologi Malaysia, Skudai Johor, Malaysia.
[2] Faculty of Computing
Universiti Teknologi Malaysia, Skudai Johor, Malaysia.
e-mail: nursyahela_90@yahoo.com, sarina@utm.my, mariyam@utm.my

Abstract

The rapid development of Mobile Cloud Computing (MCC) in this era is very significant, as the number of users accessing data keeps growing with time. Large data storage has to serve high volume transactions of data everyday when users request the data, therefore intelligent methods are required to solve the insufficient data storage experienced by some providers. Hence, a pre-fetching technique and Machine Learning (ML) technique are used to provide cloud data storage management for users to easily access their data anywhere at high speed to avoid latency; the total of response time during user access the data. This paper discusses the use of MCC technology with the pre-fetching technique to overcome the issue of latency in data management when there are large amounts of data being stored in the cloud. The pre-fetching technique is used to optimise the performance of Cloud Computing (CC) when handling data access by users. An enhancement of the MCC architecture is proposed to support data management for the efficient and effective performance of MCC services.

Keywords: *Mobile Cloud Computing architecture, data storage, pre-fetching, Machine Learning, latency, data management.*

1 Introduction

Big Data in Mobile Cloud Computing (MCC) describes a collection of data from users which is stored in the cloud, users can request these data at anytime from anywhere and hence a good data management system is needed due to the high data transfer. Billions of data transactions and streams come from devices worldwide; one of the challenges for current data management is to provide a service without any loss and with low throughput latency. However, after enabling and integrating a cloud management system, there are still difficulties serving all data streams and transactions [1]. The latency problem occurs if data management is not handled wisely when there is big data storage in a cloud, so pre-fetching and Machine Learning (ML) techniques are used to optimise performance, especially for big data. Latency, which is the total response time, is measured in this paper to evaluate the CC performance.

Pre-fetching technologies promise enhanced speed for users accessing data; in this research, the tree-view was used. Pre-fetching is one of the best techniques by which to overcome the latency problem [2, 3, 4]. In addition, ML can be used to extract information from these large volumes of data. ML is used to handle data management in MCC because it can automatically predict the data that a user will request; hence, it speeds up the user time when accessing data. For ML, the J48 algorithm is chosen to apply to the proposed architecture based on the results of analysis to provide the highest accuracy compared with other ML techniques for this research. J48 is the best algorithm, as reported by previous researchers [5, 6]. Hence, we proposed using the Intelligent Mobile Cloud Computing (IMCC) architecture that consists of pre-fetching and ML techniques to enhance the CC performance for big data storage.

In this paper, the use of cloud computing technology and the pre-fetching technique with regard to optimising large numbers of data requests and capacity for data storage will be discussed to support the proposed IMCC architecture. Data management and the speed of data access requested by users have to provide a large amount of data in data storage or many issues facing and slowing the performance of cloud storage.

The remainder of this paper is organised as follows: Section 2 discusses the related work of MCC architecture. Section 3 provides an explanation of pre-fetching for the CC framework, as this framework is a favourite topic of study. Section 4 presents the big data in CC, followed by details of the development of proposed architecture for IMCC. In section 6, the proposed implementation of ML is discussed, and then the evaluation results and discussion will be clarified in section 7. Finally, a conclusion of the study is provided in section 8.

2 Related Works

MCC offers better services to users by reducing costs, providing more reliability, and allowing users to access data everywhere, with more scalability and efficiency. Hence, it increases the demand of users for accessing many data on clouds, which can slow down performance [7, 8]. Thus, the proposed architecture is important for handling the data management to manage big data storage and avoid the problem of latency. Several papers have discussed the current MCC architecture.

The basic MCC architecture consists of components like a Mobile Terminal, which is a laptop, PDA or Smartphone. These terminals connect with a hotspot or base station via 3G, Wi-Fi, an authentication server, or CC platform, like the application server and data centre [9-11]. Fig. 1 shows the general MCC architecture and the associated components.

Esseradi *et al.* in [11] have proposed the MCC architecture shown in Fig. 2. The researchers proposed a clone architecture that is used to store the application and big data from the user. A clone is a set of programs on both Mobile and virtual platforms, to retain the offline and online application packages and user data at a similar level. These agents use mirroring files in the clone that are transmitted virtually to the clone. Hence, all type of applications can run on the platform.

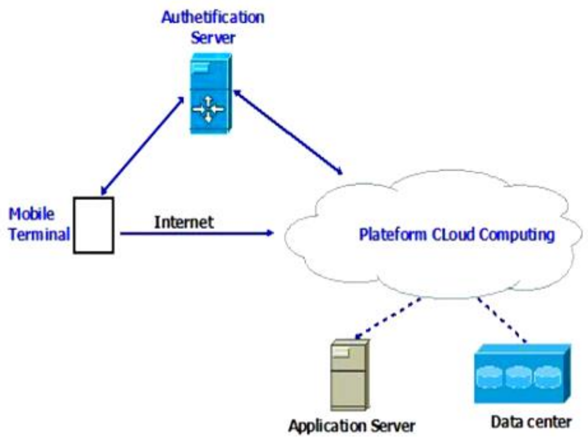

Fig. 1: MCC architecture by Esseradi *et al*. [11]

In line with Bajad *et al*. [12], architecture was introduced for secure outsourcing of data and arbitrary computations to an un-trusted commodity cloud. Bajad *et al*. [12] split the computations such that the trusted cloud is mostly used for security-critical operations in the less time-critical setup phase, whereas queries to the outsourced data are processed in parallel by the fast commodity cloud on encrypted data. CC is an emerging concept combining many fields of computing. The foundation of CC is the delivery of services, software and processing capacity over the Internet, reducing cost, increasing storage size for big data, automating systems, decoupling service delivery from the underlying technology, and providing flexibility and mobility of information. However, the actual realisation of these benefits is far from being achieved for mobile applications and open up many new research questions [12]. Hence, other research has the opportunity to study CC further.

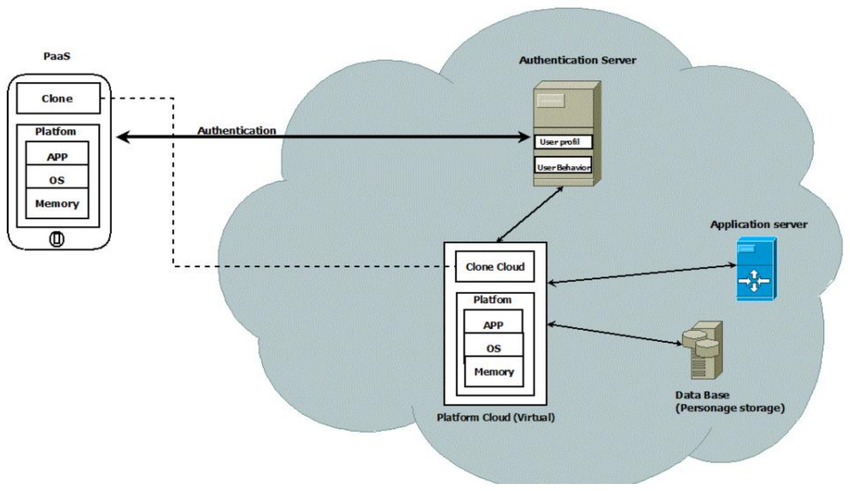

Fig. 2: Clone architecture for cloud mobile device by Esseradi *et al.* [11]

Chang *et al.* [13] described fair schedule architecture for a cloud-based TV program recommendation system. A cloud-based TV program may provide a big space on which to store TV program data. They have applied the intelligent technique with the k-Nearest Neighbour (k-NN) algorithm to recommend popular programs to new users. The results show the improvement in processing performance and increased utilisation of resources in the cloud environment. The intelligent technique is needed for handling big data. The program recommendations were available in real-time. Nevertheless, due to hardware resource limitations, this study did not use a large number of computers as CC nodes to perform the experiment and analysis. Hence, the research could not investigate the impact of increasing the number of computing nodes on the performance of the proposed system. In addition, the researcher mentioned that the possibility of applying other algorithms for better performance or accuracy can be examined in future works. Fig. 3 illustrates the proposed architecture by Chang *et al.* [10].

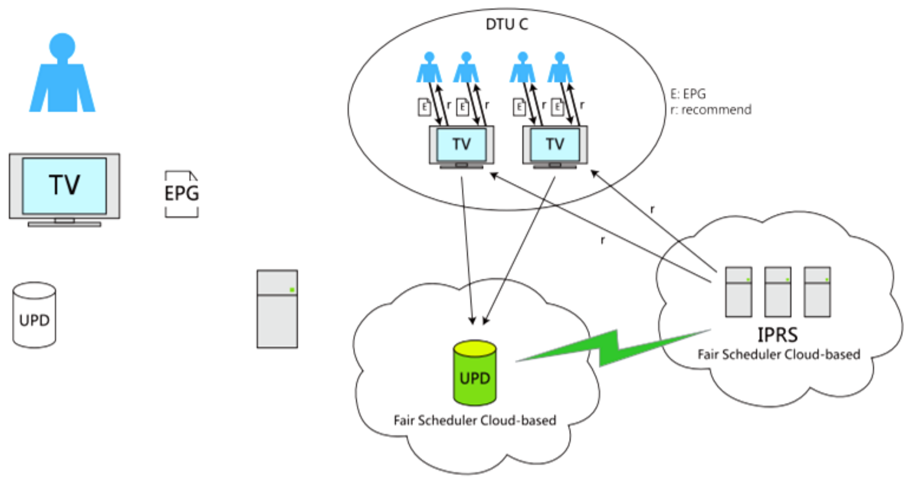

Fig. 3: Architecture proposed program recommendation system by Chang *et al.* [10]

Koukoumidis *et al.* [14] proposed an effective architecture that leverages abundant Non-volatile memories (NVM) in mobile devices, as in Fig. 4. They first analysed NVM technology scaling trends, and then propose a cloud service cache's architecture that resides on the mobile device's NVM (pocket cloudlet). The architecture enhances user experience, reducing the latency and battery life. They presented a case study of a search pocket cloudlet. Pocket cloudlet architecture allows mobile devices to capably host cloud services, but this poses diverse challenges. First, the amount of data being stored locally on the device needs to be determined for each cloud service because big data can be an issue, especially data on mobiles. Second, a mechanism to manage the locally stored cloud data is required as this data might change over time, like web content changes over time. Third, storage architecture for efficiently storing and accessing this large amount of data is needed. Mobile users need to be able to quickly search and access data across services while still having enough space to store their personal data. However, the architecture requires an expensive non-volatile memory to store big data from clouds.

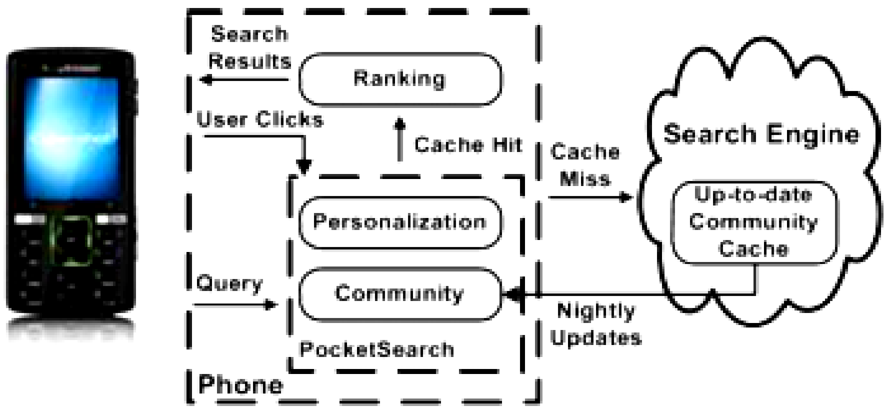

Fig. 4: Pocket Cloudlet Architecture by Koukoumidis [14]

Moreover, Mishra *et al.* [15] described MCC architecture to join mobile applications with a variety of cloud services by proposing three-tier architecture of Mobile-based cloud computing (Fig. 5). They put forward the different components of the architecture along with their services. Their aims using CC are for the storage and processing of data on mobile devices, thus reducing their limitations. However, their works have a variety of cloud services, which big data is a form of. Hence, during the transmission of big data on mobile devices, latency can occur which may disturb the MCC performance.

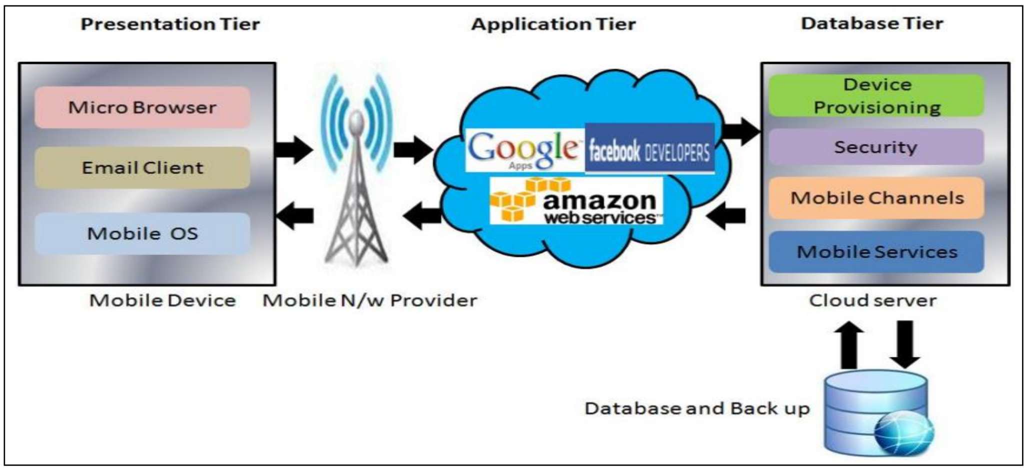

Fig. 5: MCC architecture by Mishra *et al*.[15]

The summary of all researchers on the proposed architecture is shown in Table 1. Additionally, the research has found that some other researchers have tended to study proposed frameworks. Hence, in the next section, the proposed framework from previous works will be discussed.

Table 1: Proposed Architecture by other Researchers

Author(s)	**Proposed Architecture**	**Benefit**	**Limitation**
Esseradi *et al*. [11]	Clone architecture for cloud mobile device to store the application and user data	Keep the offline and online applications packages and user data on the same level All kinds of applications can run on the Platform Cloud (Virtual).	Use a clone cloud
Chang *et al*. [13]	Fair scheduler cloud-based system architecture	Improved processing performance and increased utilisation of resources in the cloud environment.	Could not perform experiments to investigate the impact of increasing the number of computing nodes on the performance of the proposed system

Koukoumidis *et al.* [14]	Develop architecture for the Pocket Cloudlet including data selection and data management to determine the amount of data to be stored on the device for each cloud service.	Increase the efficiency of accessing data on the cloud using a local storage cache.	Architecture requires an expensive non-volatile memory to store data from clouds
Mishra *et al.* [15]	Propose Mobile Cloud Computing architecture to integrate mobile application with various cloud services.	Using cloud computing techniques for storage and processing of data on mobile devices, thereby reducing the limitation of mobile device	Users demand quality service at anytime and anywhere with speed and accuracy, so it could be a possible solution up to some extent

3 Pre-fetching on a Cloud Computing Framework

This paper covers previous works that have proposed the CC frameworks. These studies provide some idea on how to enhance the proposed IMCC architecture to optimise the CC performance. Wang *et al*. [16] proposed the AMES-Cloud framework for complete video storing and streaming system in a cloud known as the Video Cloud (VC) (refer Fig. 6). It is a very large scale base (VB), which will store the most popular and important video clips for the video service providers (VSPs).For popular videos, a temporal video base (tempVB) is used to cache new candidates; here, tempVB counts the access frequency of each video. The VC keeps running a collector to seek videos which are already popular in VSPs, and will re-encode the collected videos into SVC format and store them into tempVB first. By using this 2-tier storage, the AMES-Cloud can keep serving most of the popular videos forever. Note that the controller in the VC will handle the management work.

Specialised for each mobile user, a sub-video cloud (subVC) is created dynamically if there is any video streaming demand from the user. The subVC has a sub video base (subVB), which stores the recently fetched video segments. Note that the video deliveries among the subVCs and the VC in most cases are actually not “copy”, but just “link” operations on the same file eternally within the cloud data centre. There is also encoding function in subVC (actually a smaller-scale encoder in- stance of the encoder in VC), and if the mobile user demands a new video, which is not in the subVB or the VB in VC, the subVC will fetch, encode and transfer the video. During video streaming, mobile users will always report link conditions to their corresponding subVCs, and then the subVCs can offer adaptive video streams. Note that each mobile device also has a temporary caching storage, which is called local video base (localVB), and is used for buffering and pre-fetching.

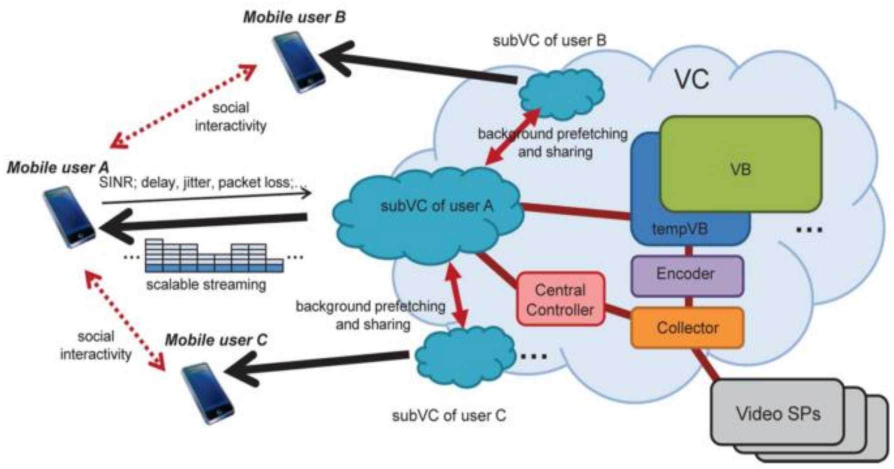

Fig. 6: AMES-Cloud framework with the Video Cloud (VC) subVCs for mobile users, the Video Base (VB), and the Video Service Providers (SPs) by Wang *et al.*[16].

Note that as the cloud service may across different places, or even continents, as in the case of a video delivery and pre-fetching between different data centres, a transmission will be carried out which can then be called a “copy”. Besides, because of the optimal deployment of data centres, as well as capable links among data centres, the “copy” of a large video file incurs a tiny delay.

Moreover, in 2014, Wang and Chen [17] improved this by using PreFeed based on RSS for pre-fetching video cloud. An RSS source is the RSS website with the original content and RSS feed is a set of many RSS updates or, for example, new articles. In the RSS, there are multiple instances of abstracted information of the news or articles from the RSS source; a particular piece of news or article is called content, which may be a mixture of text, images, audio, and video.

As illustrated in Fig. 7, PreFeed has one important centralised cloud called the feed cloud (FC), which is the key role of feed pre-fetching and pushing. In the FC, there is the main feed base (FB), which stores all pre-fetched feed contents, including the XML-based updates of the feed and all text, images, and multimedia contents of the original website. There is also a tempFB, which temporarily stores new feeds, and once it is accessed by a required number of users, it will be moved to FB. There is the organiser, to provide a summary of all RSSs of all users by filtering out duplicates, and then the collector, which will fetch the RSS updates and the original content from the feed content SPs. In particular, for each active subscribing user, the cloud virtualises a smart agent called subFC to monitor the user’s RSS requirement, as well as the wireless link quality, enabling them to decide how to request the RSS content and how to push content to the mobile user. Due to the elastic computation of dynamic resource allocation of CC, FCs with subFCs works with optimal performance adaptively to the user’s demands.

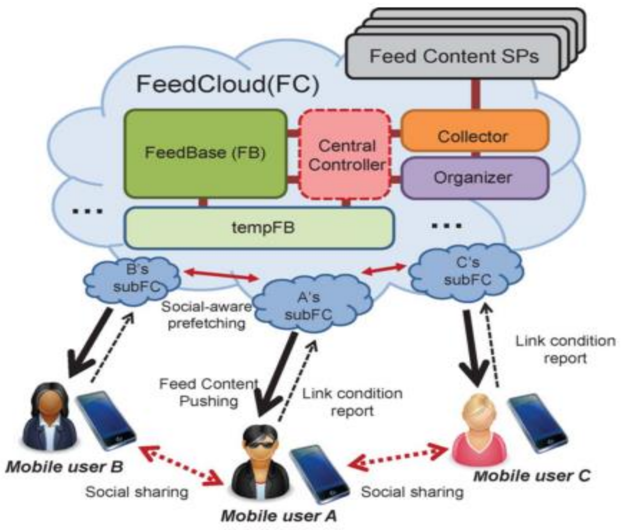

Fig. 7: PreFeed framework by Wang and Chen [17]

Regarding RSS content pushing, because mobile users have different mobility patterns and link conditions, the subFC will receive periodic updates of wireless link conditions from the users and selectively push content to users adapting to the link conditions. Due to the social sharing activities in RSS feeds, for example Google Reader, one user's friends can mark stars for particular RSS feeds, which shows that the user may have a high probability of accessing the content, so the content can be further retrieved for delivery.

Huang *et al.* [18] designed and implemented a framework to reduce web browsing latency for mobile users with a smart pre-fetching strategy and caching mechanism. The pre-fetching strategy leverages the skSLRUmodel, which predicts and pre-fetches web pages based on their contents with the consideration of user contexts and the devices' status, such as power consuming and cellular data usage. Fig. 8 depicts the architecture of their framework, which consists of a web browser and a server for accelerating the computing of pre-fetching. The pre-fetching plan relies on the support of both the statistics in the browser and the pre-fetching model on the server side; the caching mechanism is implemented in the Web browser. The result shows an improvement in the web page loading time. Their research is based on web page browsers applying the pre-fetching strategy when handling the latency issues.

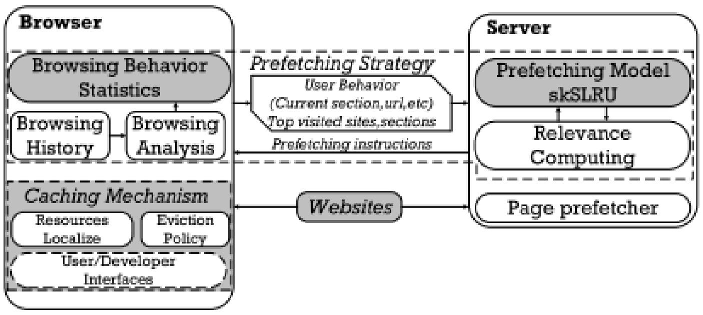

Fig. 8: Architecture of a framework by Huang *et al.* [18]

Based on those frameworks, this enables architecture to be proposed and enhanced by applying the ML techniques in the context of handling the data management issues involved in big data storage. From previous studies, it can be seen that in order to develop MCC architecture, the structure should include mobile devices, the server to manage MCC activity, databases in which to store data and some kind of user behaviour. Hence, to improve the architecture, the techniques that need to be applied should also be included in the architecture as well. The next section will discuss big data in CC, which is significant for CC research. Then, we will determine why data management is important by proposing architecture to provide greater efficiency for data management in CC.

4 Big Data in Cloud Computing

Big data is being produced all around us at all times; any information must have a large amount of data that needs to be kept or stored in a database. Today, many users like to store these data in clouds, which is known as Cloud Computing (CC). According to Balasubramaniyan and Ramachandran [19] and the European Commission [20], CC has various issues when managing the data deluge. This big data usually comes from the consumers using a digital form of data which will increase the network bandwidth and reduce the CC performance. This issue requires improvements in the structuring of data and the mechanisms required to handle it. It also leads to latency problems, meaning that it will slow down the overall progress of CC services.

According to the European Commission (2012) [20] there are several topics that need to be supported so that the European IT industry can overcome current barriers to entry into the market and maintain the CC performance for businesses. The first list from Europe is to manage the data deluge, which is in the context of volume, media types and streaming, in order to provide confidence that a cloud can maintain the best performance when dealing with different media. Hence, it requires improvements in structuring data and mechanisms to handle big data.

Today, there are high demands on using the CC service that need the CC to manage the increasing number of jobs with stable performance. As a result, this causes a challenge for data management systems. CC is often used in normal websites, but can also be used in the Mobile CC environment. The use of this technology is predicted to grow according to the increasing growth of technology. Many users take their mobile phones everywhere, thus increasing the demand for MCC to ease their work due to Internet availability. Therefore, MCC architecture is proposed in this paper to manage the challenge of data management systems in big data cloud storage. The detail of the proposed architecture for MCC is provided in the next section.

5 Proposed Architecture for IMCC

From the previous study on MCC architecture, the basic components are mobile user, server, database and platform. This research improves the architecture, which consists of entire basic components, by implementing the ML technique. Fig. 9 shows the proposed architecture that consists of mobile users, the internet, servers, databases and CC services. The mobile users connect with the server that controls the IMCC system. The CC services can access mobile phones via internet access. The most common CC services are Dropbox, Skydrive, Googledrive and Sugarsync [21-23]. All of these CC services consist of big data stores which need to be handled wisely because it will lead to latency when transferring big data on the cloud. Users find it difficult to access their data faster when there are too much data on a CC service. Users can take a long time to find their data. Hence, in this paper, architecture called IMCC is proposed. Based on the proposed architecture show in Fig. 9, users can access the CC service using the pre-fetching technique that reduces the latency by speeding the loading time access. Dynamic tree view is used as it is easier for users to directly access their most regularly viewed pages, thus increasing the speed time access, even if there are a lot of data. Besides, the J48 algorithm is proposed to enhance the data management performance in this architecture. In the J48 algorithm, it will predict the web object based on generating the rules of the algorithm. The database in this architecture acts as data storage by collecting the previous pre-fetch data and storing it in a database. When a user requests the data, it will be fetched from the database to speed up the access time.

The purpose of improving the MCC architecture is to handle the storage problem for big data. Big data is a structure built to handle high volumes of data that have to be stored; therefore, a big database is part of the big data system. With high traffic and a huge number of data transactions every day, a good monitoring system is required to provide an alert if anything happens in order to solve the issue within the quickest time. Hence, data management needs to be constructed wisely in architecture to provide a smooth performance when accessing data. When there are too much data being stored in CC services, it forms big data storage which is hard to access and search for user data within. Big data storage needs to be handled wisely, as it could lead to a latency problem. The proposed architecture supports multiple accesses on CC services. including the Dropbox, Skydrive, GoogleDrive etc. at one time; hence, users can use multiple CC services at one time and store more data easily by using the propose architecture that is applied to the system. As a result, it solves the storage problem for big data because it supports multiple CC services on one system at a time. Besides, the architecture provides the function of the pre-fetching technique by using the tree-view, which reduces the latency caused by big data. Then, the ML technique is included in the architecture structure to optimise the CC service by predicting the storage space available. Hence, this speed up the user’s time spent on CC services when users have multiple CC services compared to those individuals using traditional CC services, where users need to spend more time and are faced with latency issues when trying to access their data. Thus, the improvement in the MCC architecture is significant in this research. The next section discusses the proposed implementation of ML.

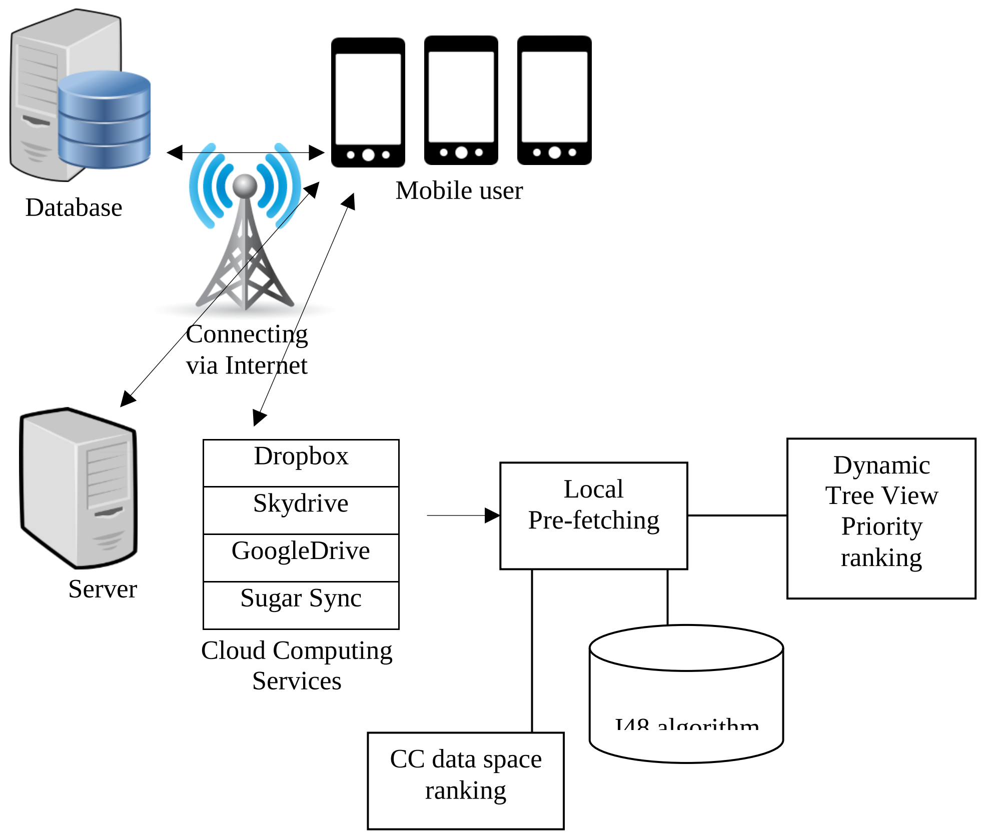

Fig. 9: Proposed Architecture of Intelligent Mobile Web Pre-fetching on Cloud Computing Environment (IMCC)

6 Machine Learning Implementation on IMCC

The proposed implementation is based on a CC environment. Fig. 10 shows the process of mobile application for the intelligent mobile Web Pre-fetching implemented in the CC services. The figure shows the different flows with normal browsing or with pre-fetching browsing. The user can access the data more effectively and rapidly, even when there are big data on a CC service, by using pre-fetching browsing as the data are already in the local storage before the user requests them. The figure also illustrates the basic sample of application by applying the pre-fetching technique to reduce the latency problem. Furthermore, the figure shows the implementation of the J48 algorithm to predict the storage space available. The calculation using theJ48 algorithm is based on the space used, the time spent on the CC service and the free space available. The resulting output shows the availability with the highest space and favourite CC services.

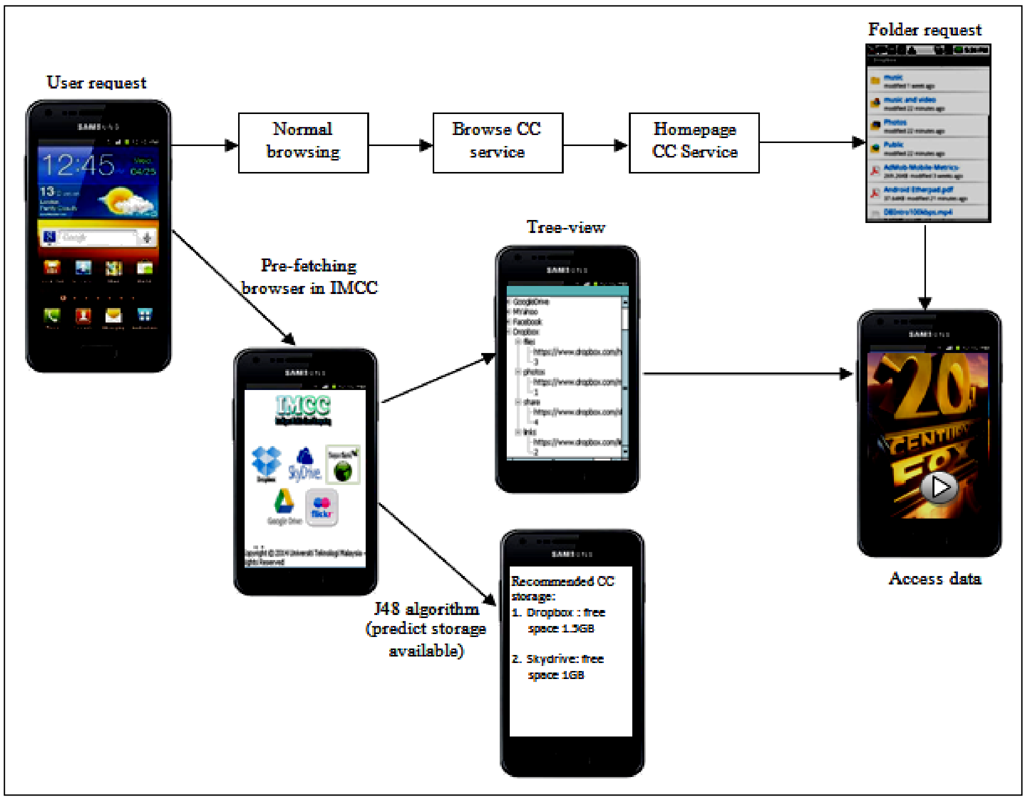
User request
Normal browsing
Browse CC service
Homepage CC Service
Folder request
Pre-fetching browser in IMCC
Tree-view
J48 algorithm (predict storage available)
Recommended CC storage:
1. Dropbox : free space 1.5GB
2. Skydrive: free space 1GB
Access data

Fig. 10: Process flow on mobile application for the IMCC

Fig. 11 shows the process flow diagram of an aspect method to request metadata on a web page linked to a device. The process flow starts with data collection from users. Then, the user requests metadata from the web page. The user sends a request from a mobile device to the web page, at which point the pre-fetching technique and ML technique (J48 algorithm) are applied to optimise the performance. Then, the system will retrieve the access pattern to be learned. The system will recognise the available free space storage and the favourite CC service. After that, it checks whether the storage space is greater than the target value; if not, it will return to the previous step by sending the request again from the user's mobile device, but if the value is higher than the target value the system

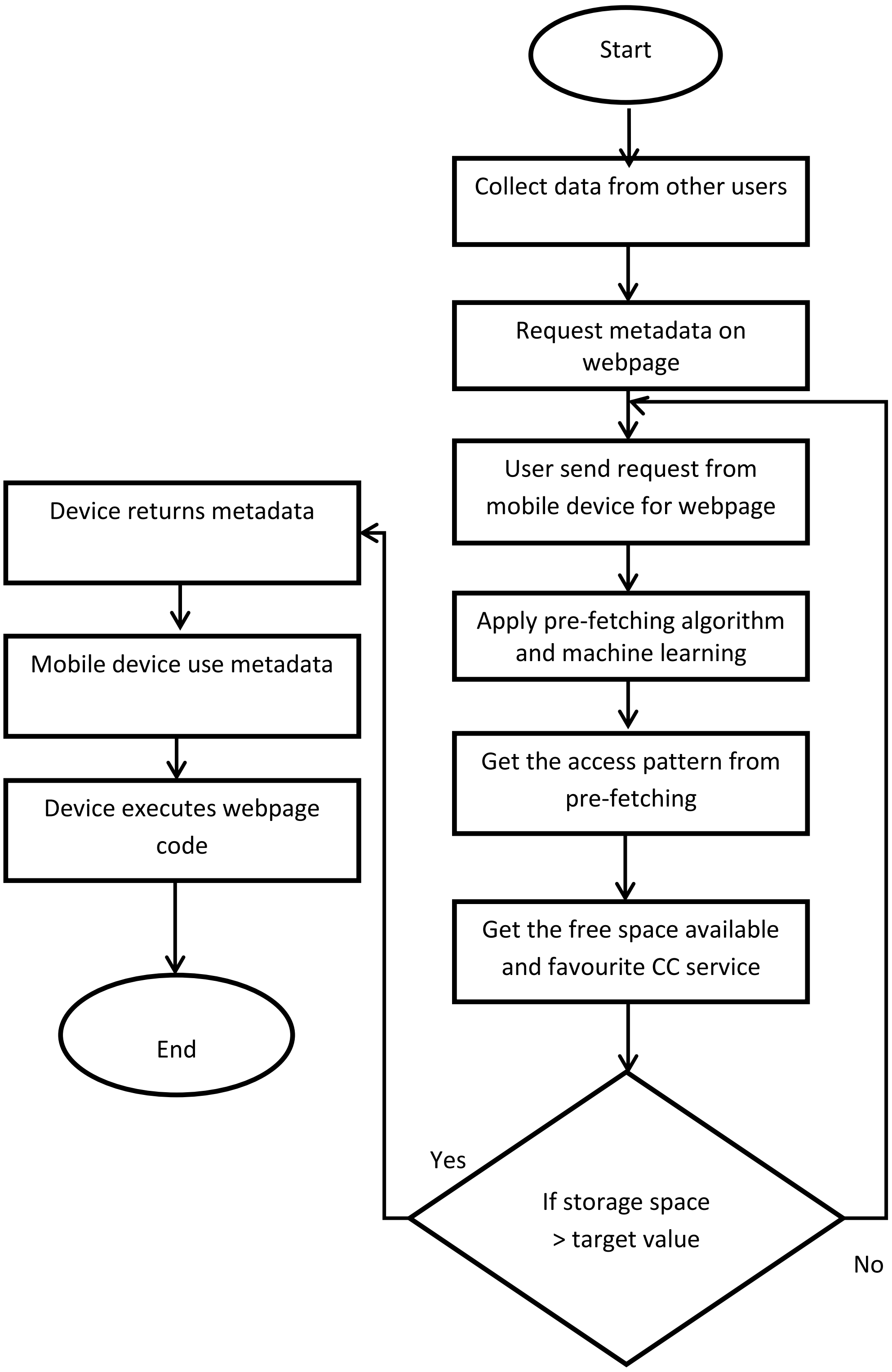
Start
Collect data from other users
Request metadata on webpage
User send request from mobile device for webpage
Apply pre-fetching algorithm and machine learning
Get the access pattern from pre-fetching
Get the free space available and favourite CC service
If storage space > target value
Yes
No
Device returns metadata
Mobile device use metadata
Device executes webpage code
End

Fig. 11: Process flow diagram of an aspect method to request metadata on Web page that linked to device

will proceed by returning the metadata to the device. The device uses the metadata and then executes the web page code. From this, it provides data management control for big data storage by allowing and suggesting the best place for the user to store the data, i.e. on which CC services.

The performance of pre-fetching that is applied in the proposed architecture is evaluated using the tree-view-based method. In addition, an advanced Keystroke Level Model (KLM) is used to compare four different situations. The advanced KLM model is used to calculate the time required for a user to access the data based on four different situations. The next section clarifies the detail of this evaluation.

7 Evaluation based on Advance KLM Model

An experiment was conducted to compare the execution time of the Dropbox application using IMCC and without an IMCC system based on the proposed architecture. Dropbox was used for the evaluation as it is the favourite CC service. A KLM was used to compare the performance proposed by Card *et al.* [20]. This model has also been discussed by other researchers [25-30]. In addition, in order to be able to describe and examine more advanced interactions on mobile phones, Holleis *et al.* [31] and Holleis *et al.* [32] performed an extensive set of studies and provided an updated Mobile KLM. This added new operators modelling novel types of interactions, as shown in Table 2 in bold.

To obtain results for this experiment, four situations will be applied in this experiment.

There are some factors that need be maintained:
Average non-secretariat typist (40 Web Pre-fetching): 0.28 seconds
User A, username (sample default name): syahela
Password (sample password): 1234

Table 2: Advanced Keystroke-Level Model (KLM) for mobile phones by Holleis *et al.* [31]

Operator	Remarks	Time (s)
A	Certain interactions require additional actions	1.23
I	Find and start interaction with device	
	Externally	5.32
	Internally	3.89
	Optimal setting	1.18
	no assumptions	4.61
F	Finger movement on a small screen	0.23

G	Gesture interaction with finger, hand, etc.	0.80
K	Press key	
	good typist (90 wpm)	0.12
	poor typist (40 wpm)	0.28
	non-typist	1.20
B	Mouse button press	
	down or up	0.10
	click	0.20
P	Moving the mobile device	1.00
H	Home hands to and from keyboard	0.40
D	Drawing – domain dependent	-
M	Mental preparation for a subtask	1.35
R	Response from system – measure	-
$\mathbf{S_{Macro}}$	Attention shift world ↔ device	0.36
$\mathbf{S_{Micro}}$	In general	0.14

(i) First Situation:

This Situation is for a user who is not using Intelligent Mobile Web Pre-fetching at all. The user must open the web browser, and then type in the URL of Dropbox mobile (http://dropbox.com). A calculation of the time duration is provided as Table 3.

Table 3: 1st situation to get the total time

Action Sequence	**Operator Sequence**	**Calculation of Total Time**
1. Initial Act	1. Start using mobile phone, I	= I+5F+7K+4K+2B+0.36
2. Point to URL text box by finger movement	2. Finger move to URL link box, F	
3. Write the URL	3. Write the URL link (http://dropbox.com)	= 1.18+(5*0.23) + (18* 0.28)+(7*0.28) + (4*0.28)+2*0.1+0.36
4. Point to username text box by finger movement	4. Finger move to username text box, F	
5. Write username	5. Write username (Assume 7-letter word), K	= 1.18+1.15+5.04+ 1.96+1.12+0.2+0.36
6. Point to password text box by finger	6. Finger move to password text box, F	= **11.01 seconds**

movement	7. Write password (Assume 4-letter word), K
7. Write password	
8. Point to sign in button by finger movement	8. Finger move to sign in button, F 9. Click sign in button, B
9. Click the files link element	10. Finger move to files link, F 11. Click the files link feature, B
10. Look at phone	12. Look at phone S_{Macro}

(ii) Second Situation:

The user logs into Dropbox Mobile without the Intelligent Mobile Web Pre-fetching system and the home element display. Then, the user continues to browse other elements. For instance, user A wants to see their files. Users do not need to key in their username and password to the system; it is assumed that the users have a similar username and password, which saves time writing the username and password and clicking on the login button. As explained in the third situation, the users do not need to insert their Dropbox Mobile password again (Table 4).

Table 4: 2nd situation to get the total time

Action Sequence	Operator Sequence	Calculation of Total Time
1. Initial Act 2. Point to username text box by finger movement 3. Write username 4. Point to password text box by finger movement 5. Write password 6. Point to login button by finger movement 7. Click the files link feature 8. Look at phone	1. Start using mobile phone, I 2. Finger move to username text box, F 3. Write username (Assume 7-letter word), K 4. Finger move to password text box, F 5. Write password (Assume 4-letter word), K 6. Finger move to login button, F 7. Click login button, B 8. Finger move to files link, F 9. Click the files link feature, B 10.Look at phone S_{Macro}	= I+4F+7K+4K+2B+0.36 = 1. 18 + (4*0.23) + (7*0.28) + (4*0.28)+2*0.1+0.36 = 1.18+0.92+1.96 +1.12+0.2 +0.36 = **5.74 seconds**

(iii) Third Situation:

For the first time login, a user must complete a number of steps, as in the first situation. Even though, for the second and the next logins, the user is only required to fill in the username and password in the system, the users will directly access the file pages as it assumes the files have the highest hit number (Table 5).

Table 5: 3rd situation to get the total time

Action Sequence	Operator Sequence	Calculation of Total Time
1. Initial Act	1. Start using mobile phone, I	= I + 3F + 7K + 4K + B+0.36
2. Point to username text box by finger movement	2. Finger move to username text box, F	= 1.18+(3*0.23)+ (7*0.28) + (4*0.28) +0.1+0.36
3. Write username	3. Write username (Assume	

4. Point to password text box by finger movement	7-letter word), K	
5. Write password	4. Finger move to password text box, F	= 1.18 +0.69+1.96 +1.12+0.1
6. Point to login button by finger movement	5. Write password (Assume 4-letter word), K	**= 5.41 seconds**
7. Click login button	6. Finger move to login button, F	
8. Look at phone	7. Click login button, B	
	8. Look at phone, S_{Macro}	

(iv) Fourth Situation:
Users open the tree-view element, check the data information and click on the link that they want to visit or view (Table 6).

Table 6: 4th situation to get the total time

Action Sequence	Operator Sequence	Calculation of Total Time
1. Initial Act	1. Start using mobile phone, I	= I+2F+2B+0.36
2. Point to tree view feature by finger movement	2. Finger move to tree view feature, F	= 1.18+(2*0.23)+(2*0.1)+0.36
3. Click tree view feature	3. Click tree view feature, B	= **2.2 seconds**
4. Point to files feature by finger movement	4. Finger move to files feature, F	
5. Click the files feature	5. Click the files feature, B	
6. Look at phone	6. Look at phone, S_{Macro}	

The results for the proposed IMCC architecture increase the speed when accessing information using the mobile-based technology for the MCC. Moreover, an access time comparison of the four situations is presented to explain the performance time difference in Fig. 12.

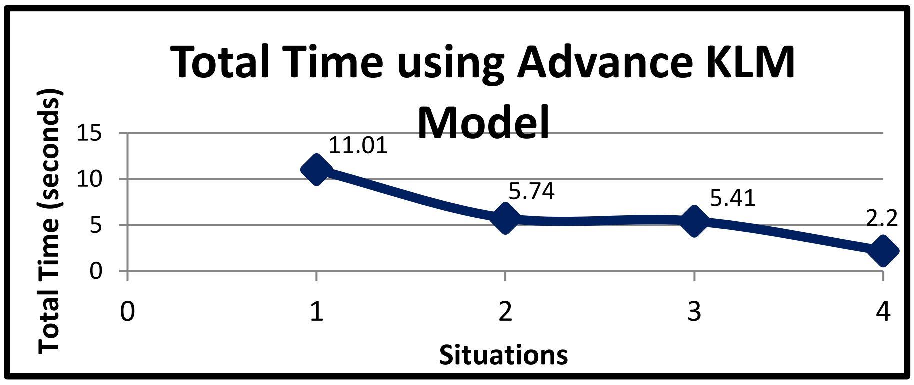

Fig. 12: Total time using KLM model in four situations

The total time using advanced KLM is plotted in Fig. 11; approximately 11.01 seconds is needed for situation 1, which uses the normal browsing via mobile phone. It is different when using the Intelligent Mobile Web Pre-fetching that is implemented in a mobile environment, as this required only 2.2 seconds for the user to access the data, which the faster time taken compare with others. Based on this result, the proposed architecture for the intelligent technique can be seen to be faster than the traditional Dropbox Mobile application access.

8 Conclusion

This paper proposes the IMCC architecture, which supports the pre-fetching technology and ML technique; these were included in the architecture to optimise the current situation and produce more efficient IMCC services. The use of pre-fetching and ML technology in big data cloud storage for IMCC architecture was discussed and found to give better efficiency and optimised performance for data management. Data management in CC acts as a database for data stored in a cloud for the user to access their data everywhere at anytime efficiently. Increasing the volume of data stored must be handled wisely to avoid latency; hence, support for the infrastructure to provide high quality services is required, otherwise the system will provide bad services. The structure of the architecture is important in order to provide good data management for big data. It needs the right IMCC architecture for the correct implementation to ensure better performance of CC services

ACKNOWLEDGEMENTS

This work is supported by Ministry of Higher Education and Universiti Teknologi Malaysia (UTM). This paper is financially supported by MyBrain15 and UTM Flagship Grant Q.J130000.2428.02G70, FRGS Grant, R.J130000.7828.4F634 and E-Science Fund, R.J130000.7928.4S117. The authors would like to express their deepest gratitude to the UTM Big Data Centre (BDC), Research Management Centre (RMC), UTM for the support in R & D, and Soft Computing Research Group (SCRG) for the inspiration in making this study a success. The authors would also like to thank the anonymous reviewers who have contributed enormously to this work.

References

[1] Abdul Kadir, E., Shamsuddin, S.M., Abdul Rahman, T., and Ismail, A.S. 2015. Big Data Network Architecture and Monitoring Use Wireless 5G Technology, Int. J. Advance Soft Compu. Appl, 7(1), 1-14.

[2] Hussien, N. S., Sulaiman, S., and Shamsuddin, S. M. 2014. A Review of Intelligent Methods for Pre-fetching in Cloud. Advances in Intelligent

Systems and Computing Springer International, 287, 647–656. http://doi.org/10.1007/978-3-319-07692-8

[3] Sulaiman, S., Shamsuddin, S. M., and Abraham, A. 2009. Rough Neuro-PSO Web caching and XML prefetching for accessing Facebook from mobile environment. 8th International Conference on Computer Information Systems and Industrial Management (CISIM 2009), IEEE Press, ISBN: 978- 1-4244-5612-3, 884–889.

[4] Sulaiman, S., Shamsuddin, S. M., and Abraham, A. 2013. A Survey of Web Caching Architectures or Deployment Schemes. International Journal of Innovative Computing, 01(1), 5–14.

[5] Patil T. R. and Sherekar S. S. 2013. Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification, Int. J. Comput. Sci. Appl., 6(2), 256–261.

[6] Kumar P. N. V., and Reddy V. R. 2014. Novel Web Proxy Cache Replacement Algorithms using Machine Learning. International Journal of Engineering Sciences and Research Technology, 3(1), 339–346.

[7] Jeong, Y., Park, J. S., and Park, J. H. 2015. An efficient authentication system of smart device using multi factors in mobile cloud service architecture. International Journal Communication System, 28, 659–674.

[8] Shiraz, M., Gani, A., Shamim, A., and Khan, S. 2015. Energy Efficient Computational Offloading Framework for Mobile Cloud Computing. Journal of Grid Computing, Springer Berlin Heidelberg, 13, 1–18.

[9] Alizadeh, M., Hassan, W. H., Zamani, M., and Khodadadi, T. 2013. A Prospective Study of Mobile Cloud Computing. Mobile Cloud Computing. International Journal of Advancements in Computing Technology (IJACT), 5(11), 8–19.

[10] Cui, Y., Ma, X., Wang, H., Stojmenovic, I., and Liu, J. 2012. A Survey of Energy Efficient Wireless Transmission and Modelling in Mobile Cloud Computing. Mobile Networks and Applications, 18(1), 148–155.

[11] Esseradi, S., Badir, H., Abderrahmane, S., and Rattrout, A. 2013. Mobile Cloud Computing: Current Development and Research Challenges. ICIT 2013 The 6th International Conference on Information Technology, 6, 1–9.

[12] Bajad R. A., Srivastava, M., and Sinha A. 2012. International Journal of Engineering Sciences and Emerging Technologies, Feb 2012, 1(2), 8–19.

[13] Chang, J.-H., Lai, C.-F., Wang, M.-S., and Wu, T.-Y. 2013. A cloud-based intelligent TV program recommendation system. International Journal Computers and Electrical Engineering, 39(7), 2379–2399.

[14] Koukoumidis, E., Lymberopoulos, D., Strauss, K., Liu, J., and Burger, D. 2012. Pocket cloudlets. ACM SIGPLAN Notices, 47(4), 171-184.

[15]Mishra, J., Dash, S. K., and Dash, S. 2012. Mobile-Cloud : A Framework of Cloud Computing for Mobile Application, CCSIT, 3, 347–356.

[16]Wang, X., Kwon T.T., and Choi Y. 2013. Mobile Cloud Computing Cloud-assisted Adaptive Video Streaming and Social-Aware Video Pre-fetching for Mobile Users, Seoul National University, (June), 72–79.

[17]Wang, X., and Chen, M. 2014. Pre-Feed: Cloud-Based Content Pre-fetching of Feed Subscriptions for Mobile Users, IEEE Systems Journal, 8(1), 202-207.

[18]Huang, H., Sun, H., Ma, G., Wang, X., and Liu, X. 2014. A Framework for Instant Mobile Web Browsing with Smart Prefetching and Caching. MobiCom ACM, 367–369.

[19]Balasubramaniyan J. and Ramachandran S. 2012. An Intelligent Cloud System Adopting File Pre-fetching, Proceeding ADCONS'11 Proceedings of the 2011 international conference on Advanced Computing, Networking and Security, 19-27.

[20] European Commission, 2012. A Roadmap for Advanced Cloud Technologies Under H2020. Recommendations by the Cloud Expert Group, 1-35.

[21]Lilly P. Top 20 Cloud Storage Service. Available at: http://www.pcadvisor.co.uk/features/storage/3421715/top-20-cloud-storage-services. [Accessed April 16, 2015].

[22]Mitroff S. 2014. Which cloud storage service is for you. Available at:

http://www.cnet.com/news/onedrive-dropbox-google-drive-and-box-which-cloud-storage-service-is-right-for-you. [Accessed April 16, 2015].

[23]Casserly M. 2014. 7 best cloud storage services - 2014's best online storage sites revealed. Available at: http://www.pcadvisor.co.uk/features/internet/

3506734/best-cloud-storage-services-review. [Accessed April 16, 2015].

[24]Card, S. K., Moran, T. P. and Newell, A. 1983. The Psychology of Human Computer Interaction. Hillsdale, NJ, USA: Lawrence Erlbaum Associates.

[25]Holleis, P., Otto, F., Hußmann, H., Iais, F., and Augustin, S. 2007. Keystroke-Level Model for Advanced Mobile Phone Interaction. CHI 2007 Proceedings Models of Mobile Interaction, 1505–1514.

[26]Jimenez, Y., and Morreale, P. 2013. Design and Evaluation of a Predictive Model for Smartphone Selection. Springer-Verlag Berlin Heidelberg 2013, (4), 376–384.

[27]Karousos, N., Katsanos, C., Tselios, N., and Xenos, M. 2013. Effortless Tool-based Evaluation of Web Form Filling Tasks using Keystroke Level Model and Fitts Law. Web and Ecommerce CHI 2013 Changing Perspectives, Paris, France, 1851–1856.

[28] Li, H., Liu, Y., Zhonglu, D., Liu, J., Li, Y., Rau, P. P., and Wang, X. 2010. Extended KLM for Mobile Phone Interaction : A User Study Result, 3517–3522.

[29] Michael, P., Gary, B., and Alan, S. 2007. An Extended Keystroke Level Model (KLM) for Predicting the Visual Demand of In-Vehicle Information Systems. CHI Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 6, 1515–1524.

[30] Hussien, N. S., Sulaiman, S., and Shamsuddin, S. M. 2014. Evaluation of Intelligent Mobile Web Pre- fetching System for Mobile Cloud Environment. Frontiers in Artificial Intelligence and Applications New Trends in Software Methodologies, Tools and Techniques, 265, 374 – 387.

[31] Holleis, P., Otto, F., Hußmann, H., and Schmidt, A. 2007. Keystroke-Level Model for Advanced Mobile Phone Interaction. CHI 2007 Proceedings Models of Mobile Interaction, 1505–1514.

[32] Holleis, P., Scherr, M., and Broll, G. 2011. A Revised Mobile KLM for Interaction with Multiple NFC-Tags. International Federation for Information Processing IFIP, 4, 204–221.