International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 4 (2015) pp. 10041-10054

http://www.ripublication.com

The Effect Of Cloudlets In Offloading Application To Cloud From Mobile Devices For Energy Efficiency

N.M Dhanya[a] and Dr. G. Kousalya[b]

[a]*Department of Computer Science and Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore-641 112, India*
e-mail: nm_dhanya@cb.amrita.edu
[b]*Department of Computer Science, Coimbatore Institute of Technology, Coimbatore - 641014 ,India*
e-mail: kousir@gmail.com

Abstract

Mobile is a device which is increasing in popularity day by day with powerful cameras and other facilities. We can utilize the power of mobiles in lot of applications. But battery capacity is a scarce resource in mobiles till now. So utilizing the high resolution cameras and using minimum battery power applications are advisable for mobile devices. Similarly we are considering two more application one a chess game and another is a sorting applications with different constraints. In this paper we propose a three tier architecture. Instead of depending on the distant cloud resources alone we are utilizing the nearby resource called cloudlet which can be accessed through wifi connectivity which is available everywhere nowadays. The cloudlet is a powerful resource which is in close proximity and by using this, the overall performance such as response time, efficiency can be increased. This paper analyses the effect of cloudlet in each application

Index Terms : Mobile cloud computing, cloudlet, offloading, Energy efficiency

I. INTRODUCTION

Face recognition [1] is an important application in our daily lives. It captures an images from a group , detect the faces in the picture and recognize the faces and give the detail about the detected faces from the database. The face recognition algorithms do analysis on the detected faces such as identifying nose, eyes, mouth, etc and use this features to analyze the detected and already present faces in the database. If a matching face is found on the database the associated information such as nationality, name, etc about the face is displayed. These algorithms give more than 90% accuracy. But all these algorithms present are very computational and resource intensive. Chess game is another application under consideration where one player is the mobile user and another player is the mobile itself. The data which is to be communicated is the current move of the client. Rest of the calculations are happening at the cloud or cloudlet and the resulting movement is returned to the mobile and displayed to the user. In sorting application the data to be transferred is larger than the face recognition and chess application. The communication cost is larger compared to other applications. The arry is passed to the cloud or cloudlet and the sorted array is getting back to the mobile for diplay.

Smartphones or high end mobile phones with great computational facilities are increasingly dominating in the mobile industry in recent years. The development of sensors and cameras in smart phones enable new application development such as face recognition, gesture recognition, and mobile health care and so on. These applications need continuous sampling and processing of the data which will take a large computation time, battery charge and memory. Even though smart phones are becoming more efficient in memory and battery life these two are the main constraints in data and process intensive applications which are running on mobile devices. So efficient utilization of these resources plays a major role in mobile technology. A solution to this problem can be derived using a combination of mobile technology and cloud computing called Mobile Cloud Computing (MCC). Through MCC the resource intensive or process intensive applications can be offloaded to cloud, making the efficient utilization of the resources and battery life time in mobile devices. This is called Mobile Computation Offloading (MCO). For example feature extractions using mobile from Flikr the work can be offloaded to a cloud server which is having fast access to Flick's data and transmit back the features to the mobile device.

Cloud computing is a new technology where we can get everything as a service. The data which we are giving is processed and stored in remote data centers of the cloud.[2][3][4] are the major cloud service provides today. A lot of other services are also available with specific services. In order to assist the mobile phones with limited processing power and battery capacity the smartphones can be augmented with cloud technologies so that execution of resource intensive tasks are not difficult to mobile phones any more.

Cloudlet is a nearby powerful computational machine with high processing power. It acts as an intermediate between actual cloud and the mobile device. Cloudlets avoid the high communication latency and high data transfer rate with the following techniques. The cloudlet can be connected to the mobile device through Wi-Fi, Bluetooth or 3G.

The contributions in this paper are

1. An architecture of mobile-cloudlet-cloud infrastructure which utilizes the power of cloud and cloudlet in offloading an application
2. The effect of cloudlet in applications like face recognition, chess game and sorting large array.
3. Energy efficiency in mobile phones where the high processing and energy utilizing operations are moved to cloudlet and cloud thus mobile side processing lighter.

II. RELATED WORKS

Smartphones are become increasingly popular. These devices are coming with high resolution cameras, processor and associated devices. Applications that can run in mobile devices are also increasing starting from the gaming applications to very complicated augmented reality applications. But these applications need high processing capacity and energy. But the processing capacity is limited to few GB's which are comparatively smaller compared to PC's and laptops. Energy is another important factor, where the battery technology is not increasing in a very slow speed as the associated mobile technologies ie, only 5% annually. Bigger batteries results in bigger devices which are not attractive.

For solving all these issues there exists a technology called offloading. The term offloading means transferring the process and resource intensive operations from resource poor smartphones to the surrogate. A rapid growth of mobile and cloud technology leads to the combination of Mobile Cloud Computing and offloading. Here are some works related to offloading.

Clone cloud[8] proposes a thread level granularity, where the device dependent classes are maintained in the mobile and rest of the application are offloaded. It uses static and dynamic code analysis to find out which part of the code should be offloaded. It does not need programmer support for conversion of the application. The application partitioning technique is fully dynamic in nature. The migration points is calculated by static analysis and dynamic profiling. Cuckoo[10] is targeted for android platform where it supports local and remote execution. It is integrated with the existing development platform. It is having a service re-writer and remote service deriver for supporting the offloading. Partially the applications are offloaded to the nearby infrastructure. The main modules are Cuckoo Remote Service Deriver(CRSD) and Cuckoo Service Rewriter(CSR).A dummy remote implementation is created by CRSD and stub/proxy is created by CSR for each AIDL interface so that the methods can be invoked remotely or locally. The application which are offloaded can be run in a JVM on the remote infrastructure or the cloud. The main disadvantage of Cuckoo is that it needs programmer support to rewrite the applications.

MAUI[9] describes the code offloading in Common Language Run (CLR) in .NET platform. MAUI creates offloading decision based on the classes with *@remotable* annotation. It takes a runtime decision of which method to offload. it creates an offloading decision which maximizes the energy saving and minimizes the programmer burden. COFA[13] Automatic and dynamic code offloading for android describes a system which can execute dynamically depending on the network conditions. It works on Android platform and is aiming for faster response and performance. This uses a thread level partitioning approach for splitting the application. The execution of the current thread is suspended at the client side and the value of program counter and other states are captured and transferred to the remote server. At the remote server VM these states are restored and the execution is restarted at that point.

ECOS[14] is an offloading technique based on security aspects. It is having an enterprise wide controller to control all offloading tasks. It is evaluated in the android platform A TLS encrypted connection is established between mobile device and the cloud for private data and normal connection is established for other data. This helps to save time and energy. This is the only offloading work which is taken into consideration of enterprise centric private offloaidng.

COMET[15] is based on Dalvic Virtual Machine. It works on the principle of distributed shared memory. In μ-Cloud[16] is based on cloud service composition for mobile devices. The mobile application is considered as a graph which is distributed among mobile devices and the cloud. This work concentrate on offline usability, scalabilty, portability and energy optimization.

Zang et al[17] proposes an elastic application model which is helpful in augmenting the mobile application with the cloud. They are introducing a term called weblet which can be launched on a mobile device or cloud. The decision on where to launch is dynamic and at run time it can change the decision. It mainly focuses on performance and energy. The main goal of ThinkAir[11] architecture is QoS. The programmer should identify all the resource intensive tasks. When a remotable method is called it is the duty of the profiler to monitor these methods and store the offloadable information for the future use. it supports three profilers. Device profiler for energy consumption, program profiler for program parameters and network profiler for monitoring network related parameters. The Execution controller is responsible for taking the decision of where to execute the methods, either locally or remotely based on energy consumption, previous execution history by profiler. At the cloud side there is a server handler which receives request from the client and sends the result back and an application server for managing the offloading code.

VM-Synthesis[18] is based on virtual machines which is running on the cloud to augment with the mobile phones. It is based on the concept of cloudlet. The process at the mobile device is suspended and resumes the execution at the virtual machine which is running on the cloud. This is called VM synthesis.

III. THE THREE TIER ARCHITECTURE

Here we are introducing a three tier architecture where the tier 1 in the mobile client for which we have to do the offloading. The middle tier is the cloudlet which is a nearby computational resource[7]. If such a resource is available the pre processing part and detection of faces in the image send from the mobile is done here and the information which is necessary for the face recognition alone is send to the cloud. The cloud tier is the place where the actual face recognition is happening. The results are send back to the mobile through the same path. This is represented in figure - 1.

If there is no cloudlet like structure is available in the close proximity the data send from the mobile is directly transferred to the cloud and all the operations (pre processing, face detection, face recognition) are happening at the cloud itself. This architecture is helpful if the database size is larger for example fraud detection in airports, where the pictures of the person is in the IB database . After preprocessing only the essential features are transferred to the cloud for comparing with the database images which is in the Amazon S3 data storage. Figure - 2 depicts this scenario.

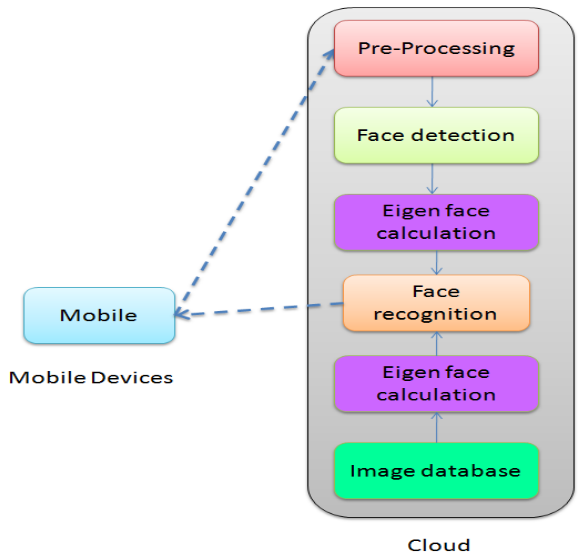

Figure 1 : Three tier architecture for face recognition

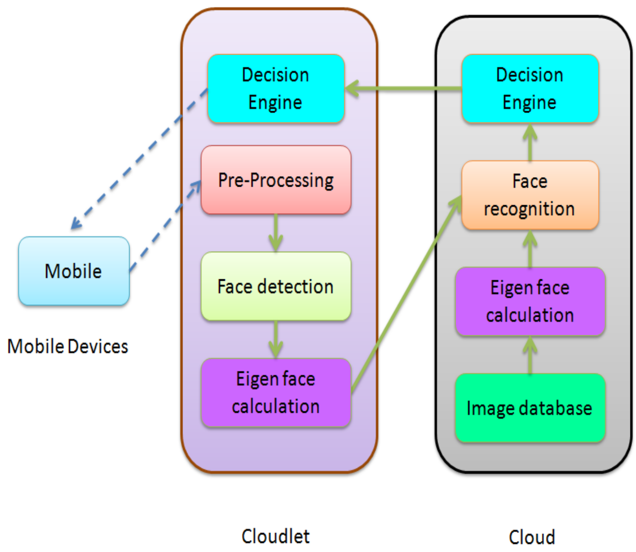

Figure 2 : Two tier architecture for face recognition

A. Mobile Device

In our architecture the duty of the mobile device is to take a picture and send this to the cloudlet. If the faces identified in this picture are there in the cloud database the details about the person is returned to the mobile device. Here we used a Samsung Galaxy phone with 1.2GB RAM and 4GB internal storage.

B. Cloudlet

A solution to resource poverty in mobile phones is to augment the mobiles with cloud computing which is having infinite resources available. But there are lots of limitations to use cloud as a support from mobile devices. The main problem is the latency to access the cloud resources which are usually far away from the mobile client. The next issue is the intermittent connectivity. The 3G technologies can make the data transmission delays. Another issue is the bandwidth which may not be sufficient to transfer a large data to the distant cloud. In order to avoid all these difficulties we are going for a concept called cloudlet.

Cloudlet is a nearby computational resource which can be connected through Wi-Fi. It can be called as datacenter in a box with enough computational and storage capacity. It can be placed in large malls, coffee shops, etc with Wi-Fi connectivity so that mobile devices can connect to the cloudlet through comparatively faster Wi-Fi compared to 3G connectivity to the distant clouds. The delay and bandwidth issues can be avoided with nearby cloudlet. All the preprocessing function can be done with the cloudlet and only limited information can be communicated to the cloud.

In our architecture the image preprocessing is done at the cloudlet. The noise and all other irrelevant information is deleted from the image and this is given to the face detection module. The face detection is implemented using JavaCV. Viola Jones algorithm is used for face detection where the Haar features and Haar classifiers are used. This produces a large number of detected faces compared to other methods. It iteratively eliminates the extra faces and the output will be more accurately detected faces. The Eigen values of the faces are calculated. The detected faces are sent to the cloud for further processing. The

In our experiments the cloudlet is a 2GB RAM system with minimum configurations. Generally cloudlet can be a system with low price and Wi-Fi connectivity.

C. Cloud

At the cloud side we use EC2 instance for processing and S3 for storing the image database. The Eigen faces which we are getting from the cloudlet is compared with the Eigen faces of the database images. if a match occurs the information pertaining to the image is taken and send back to the mobile through the cloudlet. To recognize the face , the Euclidean distance between the points of faces of the database images and the Eigen faces from the cloudlet is compared. if the distance is above a large threshold the algorithm determines that the detected is not a face and if the threshold is above a small threshold to all database faces then the face is not in the database. If the distance is below a small threshold a match is found and the information is passed to the mobile client.

IV. IMPLEMENTATION

A. Pre-Processing

The image is pre processes at the cloudlet to extract the specific features which is needed for the face detection and recognition procedures.

B. Face Detection

Face detection is a technique where from a group photo with a number of faces all faces are detected and a square box is shown for all detected faces.

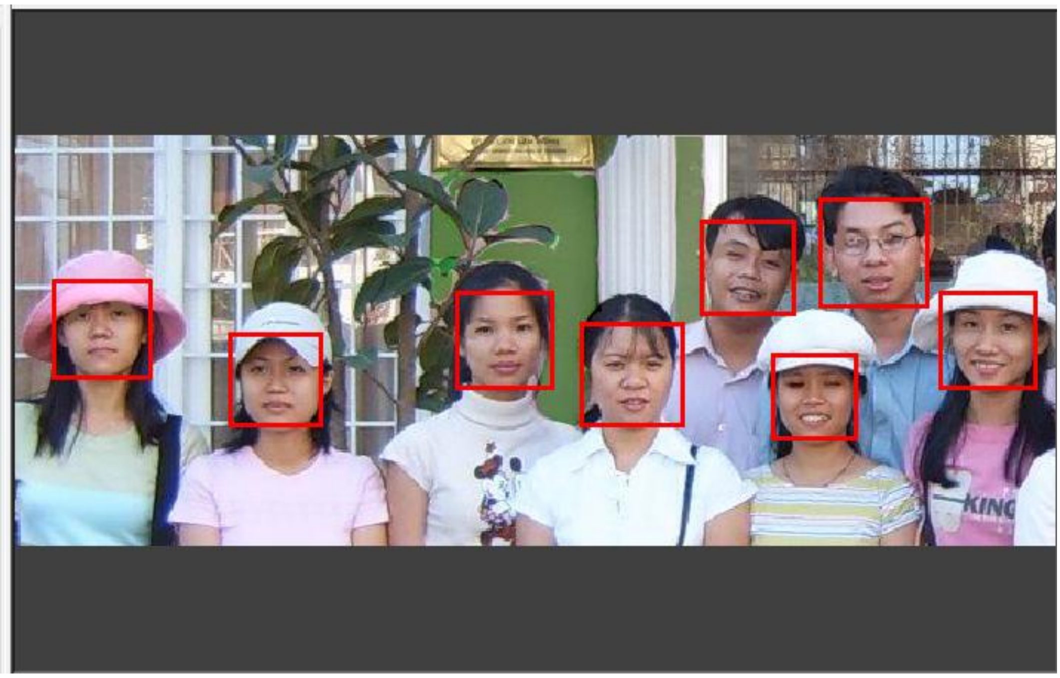

Figure 3 : Detected faces

C. Face Recognition

Figure 4 : Eigen faces

The face recognition is working as follows[5][6][12]

Eigenface computation

- The database is loaded with images.
- Set initial training images
- Calculate the eigenfaces from the training images and select those images which are having high Eigen value. Figure - 4 shows the eigenfaces calculated from our training set.
- calculate the allotment in weight space for each known individual by projecting the images into face space

Eigenfaces Recognition

- calculate a set of weights based on input image which is given from the mobile phone and eigen faces by projecting the input image onto each eigen face
- Check and see whether the image is a face by comparing to face space which is already created.
- If the detected item is a face, group the weight pattern as either a known person and return the corresponding details or as unknown

Chess game application

In chess application if the user is making a move in the mobile the system has to select the next movement. The calculation engine is the part which needs a large calculation to decide the next move. If this calculation engine can be offloaded into the cloud or nearby cloudlet the battery power can be saved significantly. In our two tier architecture the calculation engine is offloaded to the cloud directly. The calculation is happening at the cloud side and the result of the movement is returned back to the mobile. the data which is transferring is the user movement from the mobile. Only a few bandwidth is only needed for this transfer and the only delay into consideration is the mobile to cloud data transfer delay and calculation at the cloud side and returning the results back to the mobile. This application is an example for calculation intensive application but the amount of data to be transferred is small that is only the current movement of the user.

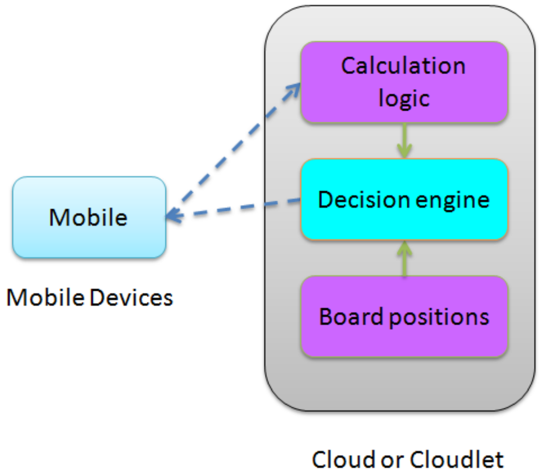

Figure 5 Chess game architecture

In the three tier architecture if we have a nearby cloudlet available no need to go for a distant cloud at all. the calculation can be offloaded to the cloudlet and get the results from there. The cloud can be replaced with the cloudlet. The results are as follows

Sorting application

We created an application which will sort a 1000 element array of random numbers 100 times. This application is a typical example for data intensive calculation along with a large data size. In our architecture the array which consists of random numbers are transferred to the cloud directly and to the cloudlet instead of cloud. In the case of nearby cloudlet the data can be transferred quickly and the sorting process is happening in the cloudlet and the result is transferred back to the mobile devices. The importance of this application is the amount of data to be transferred. there is a large amount of data to be transferred to the cloud or cloudlet for processing. so the speed up and energy efficiency depends on the bandwidth available for transferring the data. For low bandwidth the local communication will give better results than the remote execution. and if a nearby cloudlet is available with high bandwidth(usually with Wi-Fi connection) we can utilize that other than bandwidth constrained distant cloud.

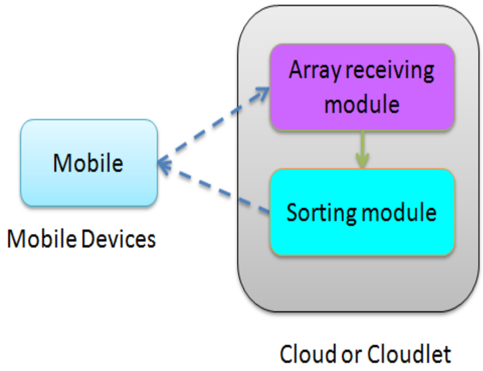

Figure 6 -Sorting Application

V. APPLICATION AND EVALUATION

Application Scenario : In airport security system a software like this allows to identify persons who are in the wanted list of criminals. The criminals can come with an altered face so that they cannot be recognized in the security systems. With a phone augmented with cloud and a local cloudlet can identify even a surgically altered images. The image of the person is taken and is send to the nearby cloudlet which is connected to the Airport. The preprocessing and the Eigen face calculation is done t the local cloudlet. this helps to avoid transferring a large amount data to the cloud. Only necessary data is transferred to the cloud where the database contain the images of all criminals. the data which is coming can be compared with the database and if a match occurs it is immediately sends back to the mobile client. The security features in airport can be strengthen using a technology like this.

Several experiments are conducted for the evaluation of the three tier architecture. The experiments including various aspects like battery utilization, performance and speedup.

Energy: The energy efficiency of the three tier architecture is compared with the two tier architecture.

Speedup : The application is running faster in the cloud execution

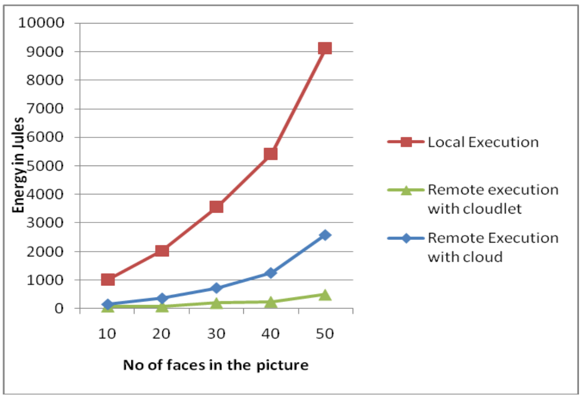

Figure 7 - Energy savings in face recognition

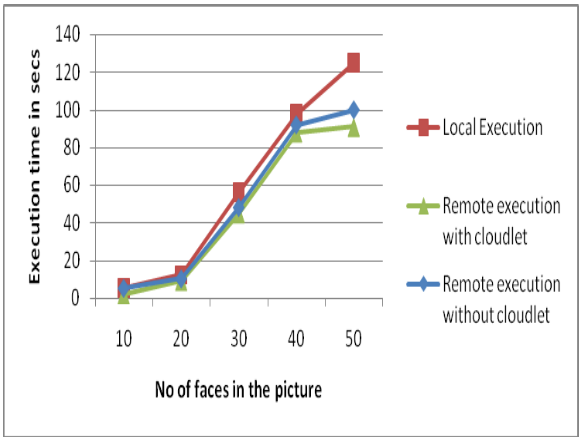

Figure 8- execution time in face recognition

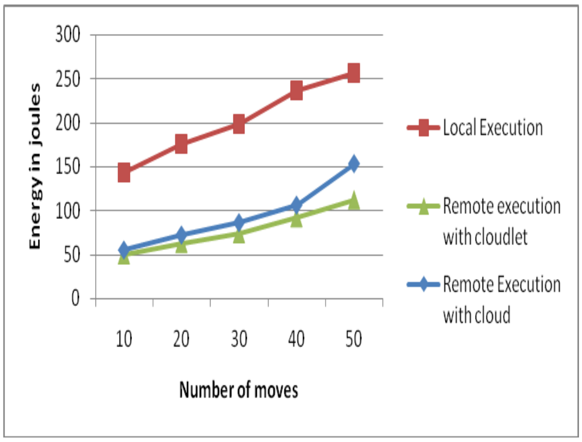

Figure 9 - energy savings in chess game

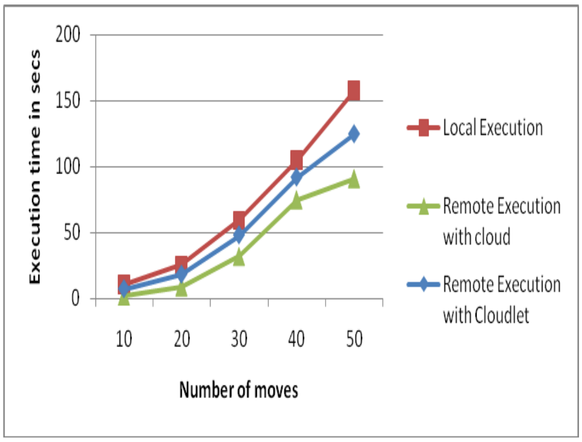

Figure 10 - execution time in chess game

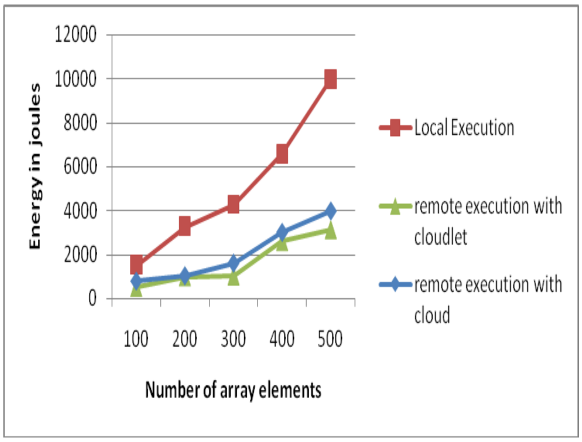

Figure 11 - energy savings in sorting

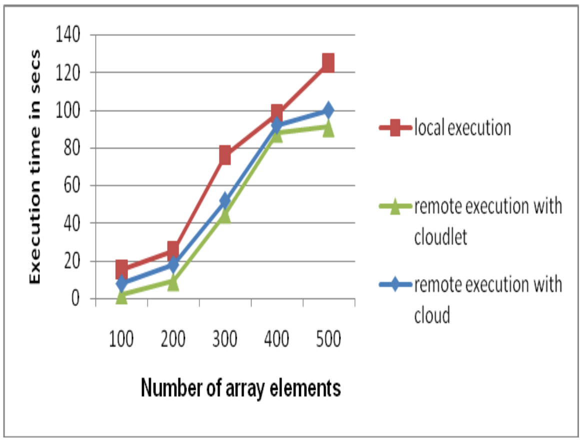

Figure 12 - execution time in sorting

VI. CONCLUSION AND FUTURE WORKS

The Three tier architecture including Mobile-Cloudlet-Cloud architecture is giving better results than the two tier architecture of Mobile-Cloud architecture. The work proposes a three tier architecture where the image capturing alone takes place at the mobile side the face detection and Eigen face calculation is happening at the cloudlet and the rest of the operation of face recognition with the database image is taking place at the cloud. In chess game application the cloudlet architecture show better performance than the distant cloud application in terms of energy and speedup. No intermediate actions can be done by the cloudlet just in the case of face recognition application. But the entire application can be done at the cloudlet and the results can be forwarded to the mobile. If no cloudlet is available in the nearby premises only distant cloud is the alternative and even if the energy and speedup is improved than local execution, the rate is less.

The experimental evaluation shows that the three tier architecture gives better results than the two tier architecture. in two tier architecture the entire information has to go to the cloud for processing. There the preprocessing , face detection and face recognition is taking place. Both response time and energy utilization is less in the case of three tier architecture.

In future this can be extended with a virtual machine approach. Context awareness can be added with this work to make it more efficient.

REFERENCES

[1] Ming-Hsuan Yang, David Kriegman, and Narendra Ahuja, 2002 ,"Detecting faces in images : A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34–58.

[2] Amazon, "Amazon Web Services (AWS)," http://aws.amazon.com.

[3] Microsoft, "Windows Azure," http://www.microsoft.com/windowazure.

[4] Google, "Google App Engine," http://code.google.com/appengine.

[5] R. Singh, M. Vatsa, H.S. Bhatt, S. Bharadwaj, A. Noore and S.S. Nooreyezdan, 2010, "Plastic Surgery: A New Dimension to Face Recognition", *In IEEE Transaction on Information Forensics and Security*, Vol. 5, No. 3, pp. 441-448.

[6] Himanshu S. Bhatt, Samarth Bharadwaj, Richa Singh, Mayank Vatsa, 2012, "Recognizing Surgically Altered Face Images using Multi-objective Evolutionary Algorithm", IEEE Transactions on Information Forensics and Security, (Volume:8 , Issue: 1) , Page(s): 89 - 100

[7] Tolga Soyata, Rajani Muraleedharan-Sreekumaridevi, Colin Funai, Minseok Kwon, Wendi Heinzelman,2012, "Cloud-Vision: Real-time Face Recognition Using a Mobile-Cloudlet-Cloud Acceleration Architecture", IEEE Symposium on Computers and Communications (ISCC),pp. 59 - 66

[8] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, 2011," Clonecloud: elastic execution between mobile device and cloud", in: Proceedings of the Sixth Conference on Computer Systems, EuroSys'11, ACM, New York, NY, USA, pp. 301–314.

[9] E. Cuervo, A. Balasubramanian, D.-K. Cho, A. Wolman, S. Saroiu, R. Chandra, P. Bahl, 2010, "MAUI: making smartphones last longer with code offload", Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys'10, ACM, New York, NY, USA, pp. 49–62.

[10] R. Kemp, N. Palmer, T. Kielmann, H. Bal, 2010, "Cuckoo: a computation offloading framework for smartphones", Proceedings of The Second International Conference on Mobile Computing, Applications, and Services, MobiCASE'10.

[11] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang,2012, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading." in Proc. of IEEE INFOCOM 2012.

[12] Mathew Turk and Alex Pentland ,2000," Eigenfaces for recognition", Journal of Cognitive Neuroscience Volume 3 Issue 1, Pages 71-86

[13] Deepak Shivarudrappa, Shashank Bharadwaj and MingLung Chen , 2011 "COFA : Automatic and Dynamic Code Offload for Android" , Boulder, CO, USA

[14] Aaron Gember, Chris Dragga, Aditya Akella University of Wisconsin, Madison 2012 "ECOS: Practical Mobile Application Offloading for Enterprises", Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems Pages 199-210

[15] Mark S. Gordon, D. Anoushe Jamshidi, Scott Mahlke, Z. Morley Mao and Xu Chen 2012 ,"COMET: Code Offload by Migrating Execution Transparently" 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI '12)

[16] Verdi Marcha, Yan Gua, Erwin Leonardia, George Goha, Markus Kirchberga, Bu Sung Leea ,2011, " µCloud: Towards a New Paradigm of Rich Mobile Applications", The 8th International Conference on Mobile Web Information Systems (MobiWIS), ScienceDirect Procedia Computer Science 5 618–624

[17] Xinwen Zhang · Anugeetha Kunjithapatham · Sangoh Jeong · Simon Gibbs ,2011, "Towards an Elastic Application Model for Augmenting the Computing Capabilities of Mobile Devices with Cloud Computing Mobile Networks and Applications", Volume 16 Issue 3, Pages 270-284

[18] Mahadev Satyanarayanan, Paramvir Bahl, Ramon Caceres, Nigel Davies Carnegie Mellon University, Microsoft Research, AT&T Research, Lancaster University 2009 "The Case for VM-based Cloudlets in Mobile Computing" , IEEE Pervasive Computing (Volume:8 , Issue: 4) 14 - 23