Minimizing Energy Consumption Of Image Data Offloading In Mobile Cloud Computing Application

Yuricha, Gede Putra Kusuma, Brian Sebastian Salim

Abstract: The tradeoff in mobile cloud computing is energy consumption on mobile devices that occurs when carrying out the offloading process. The purpose of offloading is to save energy on the mobile devices. When offloading is being carried out, data size is the determining factor of the energy consumption both in mobile device and bandwidth consumption. To minimize the data size factor, data size can be minimized by using compression techniques before it is being offloaded. But when the data is being compressed, mobile devices also consumes energy. The energy consumption required for compression depends on the data size and the quality of the data compression. The system requires a decision-making model to determine whether compression is needed or not. The proposed method for this tradeoff is using machine learning regression methods to make decision based on predictions of the smallest energy consumption value. The test uses 1,210 data which is trained by 5-fold cross validation to find the smallest RMSE value. Regression method with the smallest RMSE value is applied to test data as an evaluation. The results showed that energy consumption efficiency had an average value of 38.68% compared to sending compressed image data without decision-making model. The efficiency value of energy consumption with decision-making model versus direct offloading has an average value of 8.34%.

Index Terms: Cross Validation, Decision-making, Energy Efficiency, Machine Learning, Prediction, Regression Method, RMSE

1 INTRODUCTION

ENERGY consumption of batteries when mobile device carry out computational process is a limitation of resource in mobile computing [1]. Mobile Cloud Computing (MCC) technology appeared as one of the solutions which can solve that limitation [2] by combining mobile computing and cloud computing [3]. MCC allows computing to be done on mobile devices and transfer to the cloud, whether partially or completely from the computational process known as offloading. However, in carrying out the process, offloading also requires energy consumption on mobile devices when transmitting data to the cloud. The size of the bandwidth and the size of the offloading data are the factors that influence the offloading process [4]. Energy consumption affected by the size of bandwidth which needed to move computing data [5]. Data size also affects energy consumption when offloading process is being carried out. The larger the size of the data, more energy is needed for the data transmission, because the process of data offloading needs to be broken down into several parts, in which the transmission time is longer, and the consumed energy is greater [6]. Between bandwidth size and data size, the size of the data is more flexible to be modified before offloading. One of the ways to reduce the size of the data without reducing the quality is to use compression techniques. This is not a new thing to do. Sadler and Martonosi [7] conducted research with data compression before offloading to save energy in mobile device. Wibowo [8] stated that compression can increase the speed of data processing when transmitting and cloud server processing. Bandwidth and memory usage can be optimized by using compression and saving bandwidth in network [9], [10].

- *Yuricha is currently pursuing a master's degree program in computer science in Bina Nusantara University, Indonesia. E-mail: yuricha@binus.ac.id*
- *Gede Putra Kusuma is a lecturer in Bina Nusantara University, Indonesia. E-mail: inegara@binus.edu*
- *Brian Sebastian Salim is currently pursuing a master's degree program in computer science in Bina Nusantara University, Indonesia. Email: brian.salim@binus.ac.id*

However, the greater the data that needs to be compressed before transmitting, more energy is consumed with a value directly proportional to the execution time on the client side [11]. While the demands of compression are to produce files with a size that is as small as possible, also the results of the compression must be computed well on the cloud side. It called as the tradeoff in realizing energy efficiency in the MCC environment. A model is needed to determine whether it is necessary or not to compress the data before it is offloaded by obtaining the function to calculate energy consumption on compression, offloading, and energy consumption without compression. Image data that is directly offloaded certainly has a higher accuracy than the compressed image data. But if the compressed image data has a smaller energy consumption, then the compression ratio greatly affects computing on the cloud later. Evaluation is carried out on image data that has pixels containing bits where each bit affecting the output of data processing. Data compression uses the perceptual JPEG Encoder which has many modifications and derivatives in order to streamline data information whether it is included in the image data or in the framework of the compression speed and size of image data after compression. Modification of the JPEG Encoder uses the Discrete Cosine Transform (DCT) technique that has advantages in terms of compression speed and smaller image size compared to other techniques that can be used as a test on the offloading method. The problem to be resolved in this study is how to make the model of energy consumption for image compression before the data is transferred / sent (offloaded). By using machine learning regression method, the best model with smallest RMSE (Root Mean Square Error) value will be implemented to the MCC application to decide whether the image needs to be compressed or not before the offloading process. The smaller the size of image data, the more bandwidth consumption can be saved, also a fast execution time of compression can save energy on mobile devices. Tests can produce image data that is transmitted and computed in the cloud server section. Computational results can provide output with the right accuracy after being processed using the image recognition method.

2 Related Works

In this section, we briefly review some related works used to address the energy consumption by image compression before offloading. The use of offloading method is effective to be applied to MCC and used in image recognition [7] so as to minimize the use of battery power with the highest level of 1% and improve performance with an average execution time savings of 2.428 seconds for each workload. Offloading depends very much on the data transmission that is carried out and the size of the data to be offloaded. Balakrishnan et al [12] proposed the energy model by considering the energy consumption of both processor and memory which majorly affects the offloading decision. They conclude that there is a considerable amount of energy saving using the proposed approach. Li et al [13] have examined the advantages of using offloading on mobile devices to save energy when data is transmitted. The offloading method uses image compression in pre-processing process. The evaluation method uses computer vision - face detection. This research produces a model with maximize accuracy in images which also has an impact on energy consumption and response time on mobile devices. To deal with the problem of resource limitations on mobile devices such as memory capacity, network bandwidth, processor speed and battery have also been used as research by Wu [1]. Wu focuses on modeling the decision-making in offloading, concerning the right time to do offloading to the extent of whether the data will be offloaded all or partially by using multi-criteria. Profiling the offloading method can also be made effective by pre-processing data before it is offloaded. In terms of the pre-processing offloading technique used in this case the previous compression technique has been done by Soni et al [14] and Senthilkumar et al [15]. Compression has proven to have helped a lot in saving storage on both mobile devices and devices on the cloud side and shortening the time of transmission / transmission of data and saving memory capacity. Because energy consumption (E) is directly proportional to the power (P) multiplied by the execution time of a process (T). $E = P \times T$ (1)Compression is carried out not only in the context of energy efficiency on mobile devices but also in term of optimization of computing in the cloud [16]. There are many compression techniques that are currently developing. The choice of compression technique has an impact on the size of the data. The greater the size of the data, the longer the compression execution time will take place. The longer the compression execution time, the greater the energy needed [17]. By choosing the right compression technique, it can help minimize energy consumption for the pre-processing process. However, data with smaller size can also has its downside on the image quality considering whether it can still be computed on the server side [18]. Apart from the matter of pre-processing and offloading itself, it needs to be considered in term of bandwidth consumption which also influences energy consumption. The smaller the size of the data, the smaller the bandwidth consumption is used on data transmission. Segata et al [19] wrote about the method of measuring energy consumption for mobile devices. At the writing, it was said that energy consumption for WiFi, 3G and 4G had an influence on energy consumption on mobile devices. And offloading can only be done if the device is connected to the internet. The biggest influence lies in uploading activities (offloading from mobile devices to the cloud) and downloading activities (the process of getting responses from cloud computing). In the MCC concept that involves two parties, namely mobile and cloud devices. The main focus in this research is on energy savings on mobile. But saving energy by producing minimal image data might also not mean if the data on the cloud side cannot be computed. One type of computing that can be used in addition to using face recognition as in the evaluation method [13], another method that can be used is image recognition [20]. Image recognition involves datasets that have been provided to be used as test material to obtain accuracy from the compressed images before they are sent to the cloud. The calculation of energy consumption on mobile devices can be done by looking at the power management that is on the mobile and calculated on each workload, especially when the file is compressed and before it is offloaded to the cloud. Or look at the research conducted by Liu et al [21] which focuses on cloud computing for deep learning. Before the data is computed by the cloud, the data is first compressed so that accuracy is also taken into account. The compression used is a modification of DeepN-JPEG which is proven to consume energy by 30% without reducing the accuracy of the data to be consumed.

3 Proposed Image Data Offloading Method

The proposed method of image data offloading is using machine learning regression method to predict energy consumption for compression and offloading. The prediction will be used to decide whether making compression to the image data or not as decision making model. In determining whether the MCC application needs to do compression or not before the offloading process, the system needs to predict the size of the image data if it is to be compressed with certain quality and get the length of compression that will occur and the time of offloading that will occur.If the energy consumption of compression plus the energy consumption of offloading is greater than energy consumption of sending the image without compressing, mobile device will directly send the image without compressing and vice versa. The compression process is using perceptual JPEG encoder. The system has to be able to automatically decide whether to

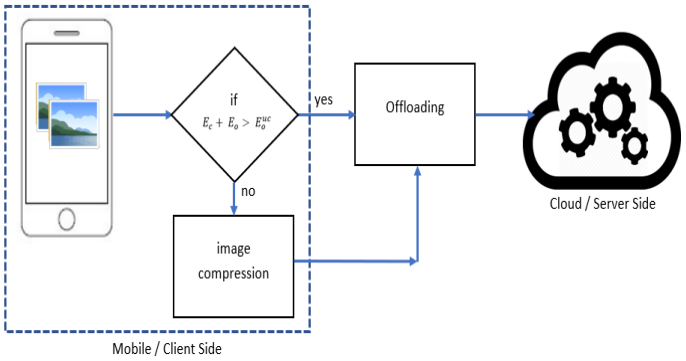

***Fig. 1**. Research Scratch Concept*

compress or not before offloading the image to the cloud as shown in Figure 1. the process of compression or offloading, the size of the image data greatly determines how much the energy consumption. The first process is profiling variables that affect the size and time of compression. Further, profiling process by taking several independent variables which are estimated to have impact on dependent variable (compression

size or compression time). The estimated independent variables: image compression quality (range used is 10 - 100% using multiple 10), image intensity, CPU usage before compression and free available memory before compression as shown in Figure 2.

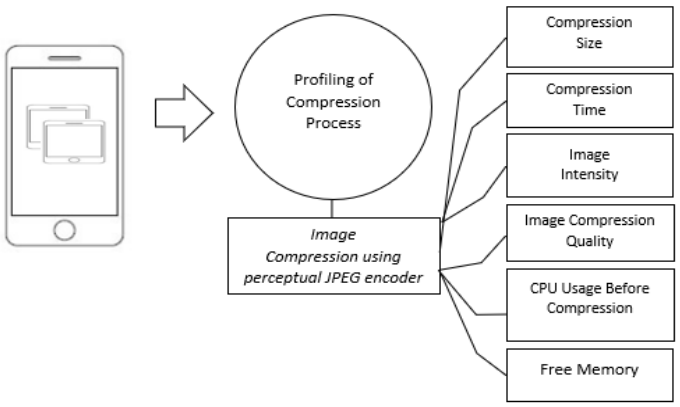

Fig. 2. *Profiling of Compression Process*

By modeling the compression size and compression time, energy consumption of compression can be countable. And the energy consumption of offloading also require function, so the system can make decision. To get the energy consumption of offloading function, the second process is profiling the offloading process. When profiling the offloading process, the targeted value is offloading time and the estimated

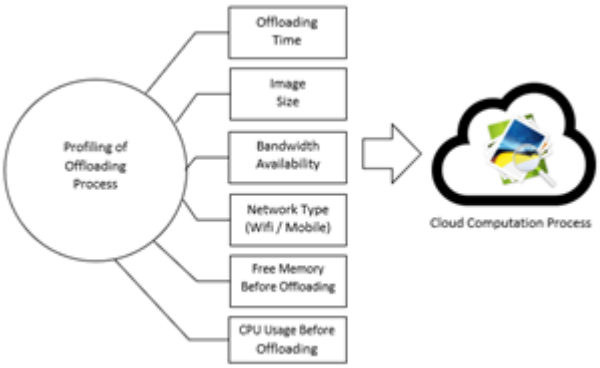

Fig. 3. *Profiling of Offloading Process*

independent variables are image size, bandwidth availability, network type, free memory and CPU usage before offloading as shown in Figure 3. Both compression and offloading process depend on the image size. The larger size of the image, the greater the process when compressing and offloading. This paper is using 640 x 480 dimension with constant size is 1200 KB (640 x 480 x 4 / 1024). Followed by data exploration process involving all existing variables. Four applicable models that are widely used by the Rapidminer community (http://mod.rapidminer.com) with numerical attributes and numerical targets (compression size, compression time, and offloading time) : k-NN (accounts for 11.2%), Neural Net (ANN) (accounts for 8.2%), Linear Regression (accounts for 7.3%), and Support Vector Regression (SVR) (accounts for 5.4%). Among the four methods, one method is chosen as a regression method that provides smaller RMSE to predict energy consumption on a mobile device. Therefore, the system can decide whether the input image data received needs to be compressed or not before the image is being offloaded to the cloud.

4 Experimental Design

4.1 Experimental Design

Modeling using machine learning regression as a tool begins by collecting the dataset. After getting the dataset, the collected dataset has to be explored. Normalization data is the part of data exploring by using min-max method. The next process is selecting model. The training process is the next step by using 5-fold cross validation and linear of sampling type. Each model is applied to data training for measurement to choose the right model by finding the lowest RMSE (Root Mean Square Error) value. It is used to predict the value for compression time, compression size and offloading time. MCC application is used to collect the dataset. The dataset is obtained by two profiling processes, compression process and offloading process. There are 1,210 data records which consists of 1,100 compressed image data and 110 uncompressed image data. 1,100 compressed image data records are used to model the size and time of compression function. 110 uncompressed image data are not included to model the size and time of compression function. The compressed image data consist of 10 type image compression quality 10 – 100% (multiple 10). 1,100 compressed image data are divided to 955 data records for training data of 5-fold cross validation and 145 compressed data records for testing data. All distributions to get the size and time compression model are also reused as training data and testing data at the time of loading the model. To train the dataset for offloading time model, the dataset is divided to 1,010 data records for training data 5-fold cross validation and 200 data records for testing. The testing data contains 145 compressed image data and 55 uncompressed image data. Due to 55 uncompressed image data on testing data, the other 55 uncompressed image data also be a part of 200 testing data. The distribution summary of cross validation training data and testing data can be seen in Table 1. The mobile device which being used in this paper is Samsung Note 3 as the Device under Test (DuT) which can support raw image. We implemented a testing system to get a dataset 110 times. Each time taking image data, 11 images are being captured and offloaded to cloud and recorded as dataset. Each of the 11 images has different image compression quality from the range 10-100 and one raw image. In order to pre-process images on Android, we use BitmapFactory to decode an image into a Bitmap and invoke createScaledBitmap to downscale the image. Decision maker is implemented as a background service on Android and a function called System.currentTimeMillis() is invoked to get execution time value in mili unit. The instance.freeMemory() function is also used to get the value of memory availability before executing an execution. Whereas to get the CPU usage value, the RandomAccessFile("/proc/stat") function is called. In offloading process, ConnectivityManager is invoked to get network type automatically and bandwidth availability value.To fetch the test images, system will send out a HTTP request to cloud. By marking the starting time and finishing time of the image fetching, the transmission time can be calculated. Another variable that needs to be obtained to calculate energy consumption is the value of power. The power value is obtained using the batteryManager function by getting the BATTERY_PROPERTY_CURRENT_AVERAGE value. We use RapidMiner v9.1 to apply the model: MLR, SVR, ANN and k-NN. We break the dataset down into 3 parts with 3 different labels with compression time, compression size and sending

time as targets. 1,210 data collected was divided into training data and testing data taken linearly and trained using 5-fold validation. In the data exploration process, the attribute data are normalized using the min-max method. Normalization is

Image Compression Quality (%)	Average of Energy Consumption (mJ)
10	24134.9996
20	24135.0013
30	24944.9994
40	24630.0001
50	24915.0000
60	24360.0004
70	24494.9989
80	24659.9998
90	25155.0010
100	25155.0025
Total Average	**24658.5003**

applied to attributes that are not targeted. The first function to be model is compression size. The original image size is set to be fixed at 1200 KB, and the compressed image size is uncertain depends on the quality of image compression and the other variables.

4.2 Experimental Results

The first test is to get the smallest RMSE value in the profiling process for compression size. The target value has a minimum value of 6 KB and a maximum value of 250 KB with an average value of 31.110 KB. Comparison of RMSE values on the four proposed models for compression sizes as shown in Figure 4.

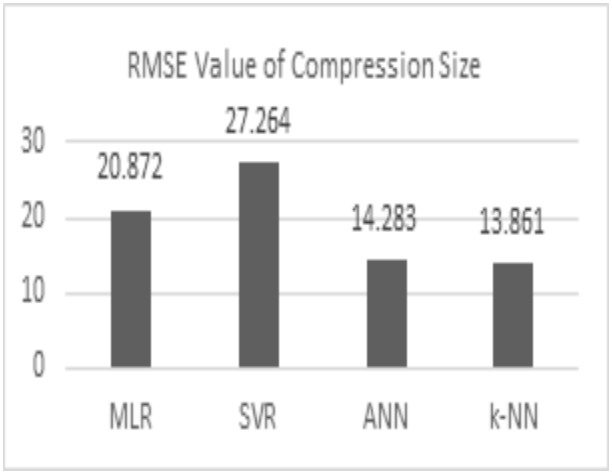

***Fig. 4**. RMSE Value of Compression Size*

In compression size testing, the MLR model has been tested with no feature selection, the RMSE value is 20.878 while using feature selection of M5 Prime obtained RMSE value is 20.872. The smallest RMSE value in MLR is gotten by feature selection of M5 Prime. The selection has been eliminated image intensity to get the higher accuracy. In the SVR model, the RMSE value is lowest when tested using the rbf kernel type: 27.264. While using the other kernel type such as linear (28.237) and poly (28.205), the RMSE value is higher than using rbf kernel type. All attribute that profiled are involved in this test.Whereas in the ANN model, the hidden layer is set to 4 with learning rate of 0.01 and training cycles of 200 delivered RMSE value of 14.283. If the learning rate is set to 0.05, the RMSE values increase to 16.586. The better result can be obtained if the hidden layer is set to 4. The last compression size testing is using the k-NN model. The value of k is set to 3 by sorting the data linearly obtained RMSE value of 13.861. Based on the four tests on compression size, k-NN is the best model for predicting the value. The second test focuses on the RMSE value of compression time. The minimum value of data training is 18 ms, the maximum value is 79 ms and the average value is 54.688 ms. The test results for each model delivered as shown in Figure 5.

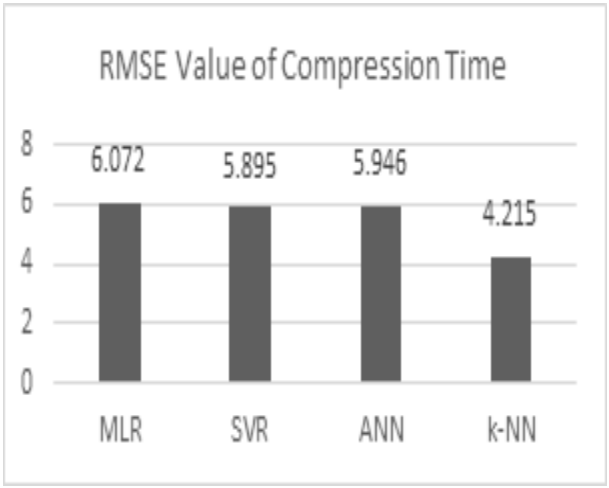

***Fig. 5**. RMSE Value of Compression Time*

On test that use MLR model, the lowest RMSE value can be delivered if no feature selection is used so all attributes are included in the test. It is RMSE value of 6.072. However, when using the SVR model, the lowest RMSE value is delivered when using the kernel type of rbf, it is 5.895. The RMSE value increases when tested using poly and linear kernels. The ANN model applied on dataset for compression time was also tested from the range learning rate of 0.01 until 0.05 and got the best result was learning rate 0.01 and the hidden layer is set to 3. While in the k-NN model, the best k-value to decide the nearest neighbor is set to 10. It due to k < 10 result is higher than k = 3. After testing the RMSE value for each model, the next is comparing the best results from each model and getting the best value is the k-NN model to get the compression time function that is equal to 4.215. The last test is testing of offloading time value. Training data is 1,010 data records with minimum value is 58 ms, maximum value is 37,613 ms and average value is 766.453 ms. The comparison of RMSE value for offloading time as shown in Figure 6. The lowest RMSE value on the MLR model is not using the result feature selection: 1,343.821 compared to the feature selection M5 Prime RMSE value is 1,345.524. When using SVR model with kernel type linear, the value is 1,760.615 and using poly kernel type has a value of 1,767.085 which is higher than the type of linear kernel. The lowest selected SVR model uses the rbf kernel type with a RMSE value of 1,758.926. When testing uses the ANN model, the lowest value of the lowest RMSE when trained is 200 times

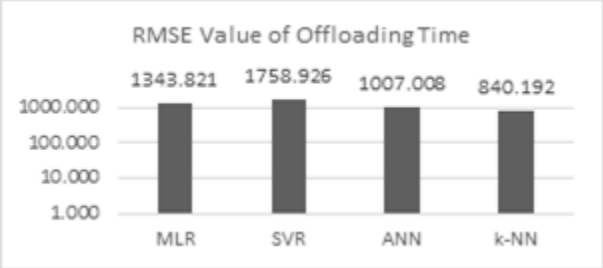

Fig. 6. RMSE Value of Offloading Time

while the RMSE value is higher when the training cycle is multiplied more than 200 times. The delivered RMSE value is 1,007.088. While the k-NN model, the training data is tested with a k value in the range 1 - 10. The best k value (set to 3) that has the lowest RMSE value of 840.192. This is because the value is higher when the k value is above 3. So, the best model to be applied for offloading time is k-NN with weighted vote and mixed measurement. Based on all testing results, all targets have the lowest RMSE value on k-NN model of machine learning. After getting the best model for each target. The next step to do is evaluating the model to data testing and calculate the energy consumption for making decision in MCC application. 145 data records as training data to evaluate the compression size model and compression time, while 200 data records as training data to evaluate the offloading time model. Training data for size and time of compression has each image compression quality with 10 data records. Whereas for training data for offloading time has 55 uncompressed data and 145 compressed data with 10 data records. Data testing was tested with each regression model obtained for compression size, compression time, and offloading time. The RMSE value obtained from testing using k-NN model of testing data results in a comparison of data as shown in Figure 7. The average value of the actual compression size for testing data is 34.16 KB and the compression time is 54.63 ms while the compression size predictive value is 31.46 KB and the

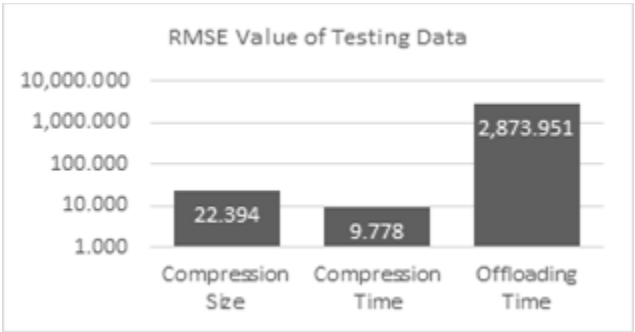

Fig. 7. *RMSE Value of Testing Data*

compression time is 54.80 ms. Evaluation for compression energy involves the value of the size and time of compression, while the delivery energy involves the time of offloading. Energy compression (E_c) value is obtained from compression time (CT) multiplied by the value of power (P_c) that needed by the mobile device for compressing the image with: $E_c = (CT * P_c)$ (2)By using existing prediction data on testing data, the results of predictions of compression energy consumption on the data delivered as in Table 2. The data presented is in the form of an average value of each image compression quality value. As well as the offloading energy (E_o) value obtained from the offloading time (OT) multiplied by the power (P_o) value on the mobile device when offloading the image data : $E_o = (OT * P_o)$ (3) In minimizing energy consumption in the MCC application a decision-making system is used to determine whether compression is necessary or not. Energy consumption when compression (E_c) is added to energy consumption when offloading (E_o). The sum of them is compared with the value of energy consumption if direct delivery without compression means that it only involves energy consumption when offloading ($E_o{}^{uc}$). If $E_o{}^{uc}$ is greater than E_c plus E_o, then the system runs compression on the captured image before sending. If the opposite, then the captured image is sent directly to the cloud. Through decision making, the MCC application look for the smallest energy consumption to save energy. By using 200 testing data applied to offloading time model which contain 145 compressed image data and 55 uncompressed image data, ten samples of data were taken randomly. The sample data taken is image data that has 10-100% image compression quality and uncompressed data. The energy value consumed if not using decision making results in an average value of 198,581.250 mJ. The value of energy consumption per quality image compression can be seen in summary Table 3. *ABLE 2* Energy Consumption for Compression Process Per Quality Energy consumption when the image data is directly sent without compression and without decision making has an average value of 921,195 mJ. When using decision making that implements a regression model in it, the average value of energy consumption efficiency is 76,805 mJ. Energy consumption efficiency uses decision making compared to direct compression without decision making of 38.68%. The efficiency of energy consumption also reaches 8.34% using decision making compared to not using decision making and directly sending image data.

5 Conclusions

This paper presents models to predict the energy consumption in mobile device using machine learning regression for minimizing energy consumption. The model has been built with k-NN that has smallest RMSE value of 5-fold cross validation training data. When image is being captured, the MCC application automatically make decision whether the image needs to be compressed or directly offload the image to the cloud. Energy consumption efficiency is 38.68% uses decision making compared to direct compression without decision making. The efficiency of energy consumption also reaches 8.34% using decision making compared to not using decision making and directly sending image data.

References

[1] H. Wu, "Multi-Objective Decision-Making for Mobile Cloud Offloading : A Survey," *IEEE Access,* pp. 3962-3976, 2018.

[2] E. Cuervo and &. B. A., "MAUI : making smartphones last longer with code offload," *ACM,* pp. 49-62, 2010.

[3] D. Debashis, "Mobile Cloud Computing: Architectures, Algorithms and Applications," *Chapman and Hall/CRC Press,* vol. 4, 2015.

[4] H. Qi and A. Gani, "Research On Mobile Cloud

Computing : Review, Trend and Perspectives," in *2nd International Conference on Digital Information and Communication Technology and its Applications, DICTAP 2012*, 2012.

[5] A. Kaur and K. Kaur, "A Comparative Study of Code Offloading Techniques and Application Partitioning Methods in Mobile Cloud Computing," *International Journal of Computer Applications,* 2016.

[6] Lin et al, "Time-and-Energy-Aware Computation Offloading in Handheld Devices to Coprocessors and Clouds," *IEEE Systems Journal,* pp. 1-13, 2013.

[7] C. M. Sadler and M. Martonosi, "Data Compression Algorithms for Energy-Constrained Devices in Delay Tolerant Networks," in *Proceeding of ACM SenSys*, 2006.

[8] P. S. Wibowo, "Rancang Bangun Mobile Computation Offloading Framework untuk Peningkatan Kinerja dan Penghematan Daya pada Smartphone (Studi Kasus Leaf Recognition)," Institut Teknologi Sepuluh November, Surabaya, 2017.

[9] H. C. Rustamaji, Mariani and B. Yuwono, "Aplikasi Kompresi Data Menggunakan Metode Huffman Statik pada Perangkat Mobile Berbasis Android," *Telematika,* pp. 9-18, 2014.

[10] M. Singh, S. Kumar, S. Chouhan and M. Shrivastava, "Various Image Compression Techniques: Lossy and Lossless," *International Journal of Computer Applications,* 2016.

[11] Tawalbeh et al, "Studying the Energy Consumption in Mobile Devices," in *The 13th International Conference on Mobile Systems and Pervasive Computing*, 2016.

[12] Balakrishnan et al, "An Energy Efficient Code Offloading Approach for Mobile Cloud Computing," *Indian Journal of Science and Technology,* vol. 9, no. 48, pp. 1-5, 2016.

[13] Li et al, "Make Smartphones Last A Day : Pre-processing Based Computer Vision Application Offloading," in *12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, Seattle, WA, USA, 2015.

[14] M. Soni and N. Shukla, "Data Compression Techniques in Cloud Computing," *IJARCCE Journal,* pp. ISSN: 2319-5940, 2016.

[15] M. Senthilkumar and V. Mathivanan, "Analysis of Data Compression Techniques using Huffman Coding and Arithmetic Coding," *International Journal of Advanced Research in Computer Science and Software Engineering,* 2016.

[16] Govinda K, Yuvaraj Kumar, "Storage Optimization in Cloud Environment using Compression Algorithm," *International Journal of Emerging Trends & Technology in Computer Science,* pp. ISSN: 2278-6856, 2012.

[17] S. Kasireddy and V. Bojja, "Measurements of Energy Consumption in Mobile Applications with respect to Quality of Experience," Blekinge Tekniska Hogskola, Karlskrona, 2012.

[18] S. Kasireddy, "Measurements of Energy Consumption in Mobile Applications with Respect to Quality of Experience," Blekinge Institute of Technology, Sweden, 2012.

[19] Segata et al, "Towards energy efficient smart phone applications: Energy models for offloading tasks into the cloud," in *IEEE International Conference on Communications (ICC)*, Sydney, NSW, Australia, 2014.

[20] V. Chandrasekhar, Chen, Tsai, Cheung, H. Chen, G. Takacs, Reznik, R. Vedantham, R. Grzesczcuk, Bach and Girod, "The Stanford Mobile Visual Search Dataset," in *Proceedings of the First ACM Multimedia Systems Conference (MMSys)*, San Jose, 2011.

[21] Liu et al, "DeepN-JPEG: A Deep Neural Network Favorable JPEG-based Image Compression Framework,"in *55th Annual Design Automation Conference*, 2018.