The Javanese Letters Classifier with Mobile Client-Server Architecture and Convolution Neural Network Method

https://doi.org/10.3991/ijim.v13i12.11492

Yulius Harjoseputro (✉), Yonathan Dri Handarkho, Heronimus Tresy Renata Adie
Universitas Atma Jaya Yogyakarta, Yogyakarta, Indonesia
yulius.harjoseputro@uajy.ac.id

Abstract—The rapid development of mobile technologies allows platform devices to perform sophisticated tasks, including character recognition. These identification systems are notable techniques that required high computation cost, in order to achieve acceptable accuracy resulting from diversity in alphabet shape and method of writing, especially for the non-Latin alphabet, e.g., Javanese letter. In addition, numerous studies have attempted to address these issues by employing a Convolution Neural Network (CNN) due to its ability to provide high accuracy in character detection. However, the performance on mobile devices is possibly faced with problems resulting from the limitation of computation resource on the platform that also affect computation cost. This study, therefore, proposes a 2-tier architecture by placing the mobile app as a client that invokes a Javanese letters classifier service, which is based on CNN, and implemented in the web-server through the Application Program Interface (API). The results show that the letter classification was successfully implemented in a mobile platform, with an accuracy rate of 86.68%, utilizing training for 50 epochs, and an average time of 1935 ms.

Keywords—CNN, Javanese letter, character recognition, Mobile client, 2-tier architecture, API.

1 Introduction

The use of classification image as a part of pattern recognition and machine learning has recently become an exciting topic. Meanwhile, with the growth of mobile technology, many scholars have attempted to develop a detecting system that is friendly with the limitations of device resource and also capable of producing high accuracy in the process of recognizing specific objects, e.g., character on mobile devices [1]. In addition, a major challenge in this development is related to the cost sacrificed for mobile computation resource, therefore, numerous scholars attempted techniques that are capable of resolving the problem, including Deep Learning, which encompasses Convolution Neural Network (CNN). This has been used to develop apps that perform character recognition tasks on mobile device platforms, and several modification and combination with other technique have also been adopted to address

computation cost issue [1] [2] [3]. Meanwhile, other reports stated the approach of utilizing the mobile device technology to minimize the resource applied in CNN algorithm, through their capacity to transmit and receive data that permits the involvement of other external resources in the computation process [4].

With the evolution of mobile device technology, there has been an increase in the development and adoption of character recognition on the platform in the numerous genres of the app, including within the educational field [5] [6]. According to [7], children tend to be interested in an application that offers both educational aspects and entertainment, therefore, the integration of engaging content and the advantages of smartphone features, tends to motivate the user to engage with the app, enhancing its perception as a productive tool in the process of learning about a specific subject [7] [8]. This enhances the potential of adopting character recognition as an entertainment element, with content that are engaging, possessing the tendency of attracting and encouraging students to learn particular topic related to alphabetic principle [5]. However, these methods are already known to require high computation cost, which is due to the diversity in alphabet shape and the technique adopted in writing, especially for the non-Latin alphabet [9] [10]. In addition, [4] argued on the competency consideration of resources in the development of mobile technologies and network today in the conduction of computation in character recognizing task. However, the alternative approach also needs to be observed, in order to ensure smooth and accurate operation in a mobile platform.

In relation with non-Italic alphabet recognition identified with the complexity of shape and style in contrast with Roman alphabets and previous studies have attempted to address this problem by proposing several methods. These were based on deep learning technique for character recognition, in Chinese, Japan, Korean, Thai, and Arabic letters [11] [12] [13] [14] [15]. In addition, as a country consists of diverse culture and ethnicities, Indonesia also has traditional characters included in the non-Italic alphabets, e.g., Javanese, which are different from Roman alphabets. These tend to possess different structures, complexity and shapes, making them a challenge to the system [16]. Moreover, there are about 20 syllables construct, equipped with special signs, related to the pronunciation and other complementary characters with different functions [16] [17], although Javanese people rarely use them in their daily life [18]. Therefore, for a specific intention, people tend to utilize these letters in education and cultural purpose. For example, in the Province of Yogyakarta, Javanese letters are applied as street names, in the menu of restaurants, around the Palace in Yogyakarta, and also taught in elementary and secondary schools [17]. Therefore, the maintenance of this cultural heritage, especially for a new generation requires the involvement of technology, consisting of character recognition that is possibly proposed as an alternative solution to encourage young people to initiate the learning process with new levels of excitement.

In addition, several prior studies have already addressed these issues, utilizing several methods, encompassing the Hidden Markov Model [19], Artificial Neural Network [18], Multiclass Support Vector Machine [16], as well as CNN technique [17]. However, this study tends to focus on the means of applying the CNN algorithm in a mobile device platform without suffering computation cost, alongside maintaining an

acceptable accuracy in recognizing Javanese character. This research is concerned with the capability of mobile computing, which relies on not only smartphone resource but also on the architecture of a network that significantly supports this platform, especially in terms of sharing and balancing the resource usage, as seen in mobile client-server architecture. Furthermore, there is also an attempt to purpose Javanese letters image classifier on standard web server specification and mobile devices client, in order to divide a load of computation resources, with the aim of providing high accuracy in the process. This approach has not previously been considered extensively in Javanese character recognition studies, thus, the view as a contribution in this specific context, particularly in development based on mobile platforms.

2 Review of Related Literature

In this section, several studies related to Javanese Alphabet recognition and the use of CNN as an algorithm of detection is discussed. Widiarti and Wastu [19] were amongst the earliest researchers to study the script, and the Hidden Markov Model was applied as a stochastic to understand its features. In addition, 50 images were adopted for each character, producing an accuracy of 85.7%, while [18] also conducted an investigation on handwritten character identification, through the neural network. Therefore, the results indicated that a 2-way Association method and counter propagation did not match the Javanese letters used for identification because the resulting accuracy was very low. This leads to the recommended use of a combination of Chi2, and back propagation neural networks with a resulting accuracy of 98%.

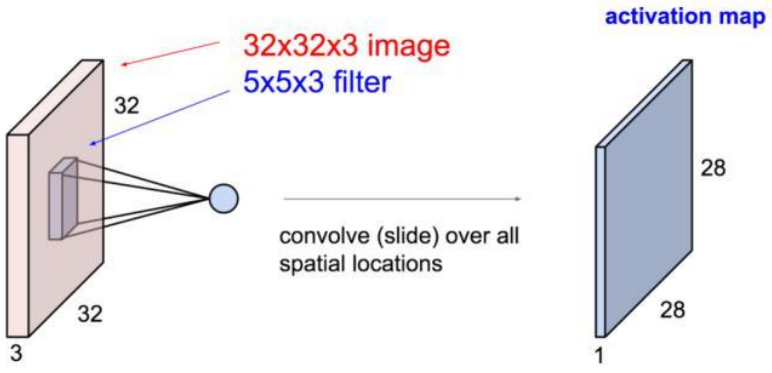
In the development of studies related to character recognition, especially for the non-Latin alphabets known to possess a complex varied structure and shape, several scholars propose deep learning techniques with extraction and modeling feature [20]. This method adopts an advanced algorithm, which is supported by a reliable training data set and powerful computation proses, believed to capably provide a preferred solution in machine learning field, including image and pattern detection [21]. With respect to Character recognition, Convolution Neural Network (CNN) is specified as an example of a Deep learning method utilized in prior investigations, due to the ability to perform a recognition process with high levels of accuracy. For instance, Dutt and AashiDutt [22] adopted it as a technique to develop a handwritten number recognition system, employing the 'Tensorflow' library for its implementation, subsequently producing an accuracy level of 99.70%. However, the code complexity and the slightly higher processing time tend to be the major weaknesses.

Furthermore, there has also been a wide application in the identification of non-italic alphabet, which happen to be complex and challenging to recognize, in contrast with the Roman alphabet [3] [9] [12] [13] [14] [15] [23] [24] [25]. For example, a report by Younis [24] focused on the identification of handwritten Arabic characters that utilize several datasets, generating a 94.6% level of accuracy through the total of 8738 AIA9K, comprising 85% and 15% of training and test images, respectively. Also, 97.6% accuracy was measured, using AHCD, with the total number of images in the dataset being 16800, consisting of 13,440 training and 3360 testing images [24].

In addition, related to mobile computation, several studies have successfully exhibited the strength of CNN in the conduction of character detection in the mobile platforms, without suffering computational loss [2] [3] [11]. This technique has previously been used by [17] with a different goal and context, focused on the development of web-based systems, and a marked level of accuracy in comparison with Multilayer Perception (MLP). In addition, another study specifically adopts CNN for Javanese letters detection by utilizing the GPU resource in a non-mobile platform, resulting in accuracy of 85%, with overall training test conducted in 409, 25 second [26]. This research, however, has a different objective, proposing the implementation of Javanese recognition system on a mobile device, by employing client-server architecture, in order to provide acceptable accuracy and computation load, which is suitable with a low specification of the web server and mobile devices.

2.1 The concept of the convolution neural network

Convolution Neural Network (CNN) is a model that is expanded based on a traditional neural network, which is widely used for image detection and recognition [27] [17], specifically developed to process two-dimensional data. However, its weakness is identified in the complexity of the constructed number of layers [28], which has implications for the time of data training, hence, numerous studies recommend the use of additional computation resources, e.g., GPU (Graphics Processing Unit), in order to avoid suffering training cost [29] [27]. In the CNN, several layers have been identified for possible use in the filtration of each process, e.g., in training, which generally has three stages, encompassing the Convolution, Pooling, and fully connected layers [30]. Furthermore, the convolution tends to perform operations that are based on the output of the previous layer, being the main process underlying a CNN, which in this case is a mathematical term that means applying a function on the output of others conducted repeatedly [31]. Therefore, a layer of convolution tends to significantly experience the complexity of a model through the optimization of its output.



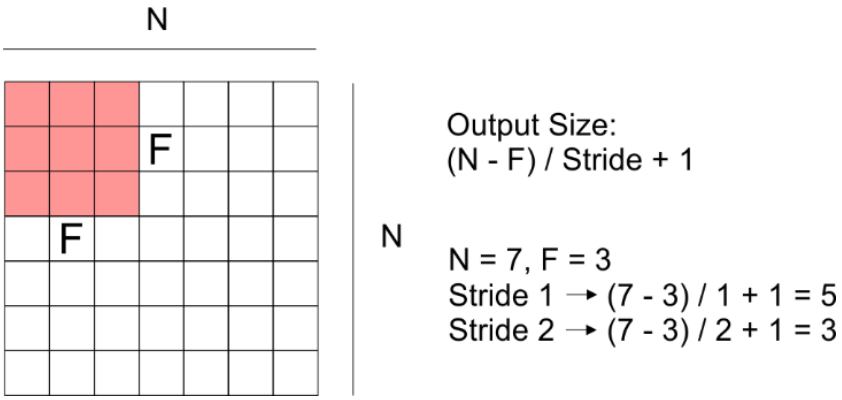


Fig. 1. Convolution Layer [30]

Pooling Layer tends to use functions with Feature Map as its input, subsequently processing with various statistical operations available, using nearby pixel values [29]. In addition, the application often requires the use of (1) Max Pooling with size 2x2 and Stride 2, with a value taken for each shift of this filter, and the largest in the 2x2 area, (2) Average Pooling, which takes the average value [30]. The most important aspect in the creation of a CNN model is the choice amongst different pooling layers, which capably enhance the model performance [32].

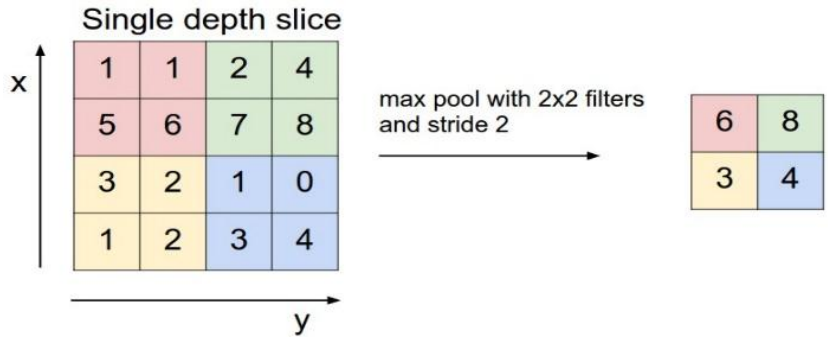


Fig. 2. Pooling Layer [33]

In the Fully Connected system, all the activation neurons from previous layer are linked with the neurons in the next, as well as the concept of ordinary artificial neural networks. Therefore, the security on activating the previous layer is converted into a 1 Dimension data prior to the connection with all neurons attached to the Fully Connected Layer, and the difference between both is observed in the aspect that the convolution type is only associated with certain regions, while the latter possesses the entire connection. In addition, they also tend to have a 1 x 1 kernel, which perform the similar functions, while maintaining the spatial character of the data [34]. Moreover, the Fully Connected type tends to have several Hidden and Output Layers, as well as Action, and Loss Functions [30].

Due to this model complexity, several studies have proposed the need for modifying CNN, through a combination with other approaches, in order to solve the computation cost problem [1] [2] [3], or dividing the burden, using another resources [4] [27] [26]. In addition, mobile computation context adopted in prior studies tend to address this issue by utilizing the Mobile GPU, in an attempt to minimize computation latency [27]. However, rather than place the entire load and process on a mobile device, it was proposed that another alternative, which exploits the capability of mobile internet service by introducing mobile client-server architecture. This provides a Javanese character recognition system with acceptable level of accuracy and computation load.

3 Methodology

This section discusses the concept of interface API and mobile client-server architecture in the building of a Javanese letter classifier, which was subsequently divided into two parts:

- The general description about its implementation in this context of study
- The architecture and a more detailed explanation of the Javanese letters image classifier.

3.1 The concept of interface API and mobile client for javanese letters recognition system

The final model outcome from the training process of this study needs to be accessible to the mobile clients, thus, the interface of HTTP based API is required for receiving and sending the result of the image classification process. In addition, the mechanism is initiated when the server obtains a request to classify the specific image that contains Javanese characters from the mobile client, followed by the server sending back the response in the JSON-formatted, containing classification results from the model. In addition, python-based Flask software was adopted in this interface implementation.

The mobile client app developed uses the Android framework, which possesses the capability to invoke and receive the response from Javanese letters models from the server-side, through API interface, based on the Http protocol. Therefore, the process flow is initiated when the mobile app obtains an image of Javanese letters through smartphone camera devices where it is installed, and then the data is subsequently sent to the server through API, which invokes the models that eventually process the image based on the dataset model. In addition, the result of classification is sent back to the API server, forwarded to the mobile client, and shown on the app, as seen in Fig. 3, which shows an overview of the entire process.

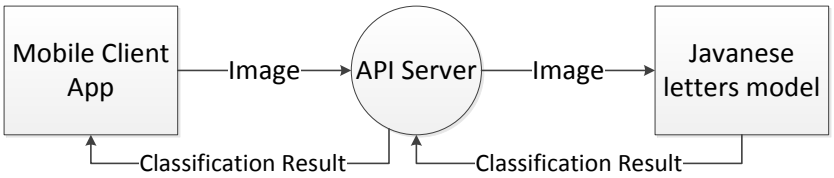


Fig. 3. Mobile client app flowchart when sending image and get the results of the classification

3.2 The implementation of javanese letters classifier with mobile client-server architecture

Typically, character recognition systems that are based on image detection requires a machine learning algorithm that demands high computation power to run its classifier model, which is not suitable with a mobile device that also has lower computation capabilities, with smaller memory and storage. This study, therefore, proposes a Javanese letter object class that is able to comply with a minimum specification web server and mobile devices client, and a pre-trained model classifier adapted to work on low computation power devices is also examined. These adopt 11 layers, which is based on Convolution Neural Network, as shown in Table 1, already tested with 86.68% accuracy, after 50 epoch training and 0.37-second average inference process. In addition, its properties are suitable to run on a low specification web server or mobile devices.

Table 1. Model Summary

Layer [(a)]	**Output Shape** [(b)]	**Parameter**
Convolution 2D (ReLU activation)	(None, 32, 32, 32)	320
Dropout	(None, 32, 32, 32)	0
Convolution 2D (ReLU activation)	(None, 32, 32, 32)	9248
Max Pooling	(None, 32, 16, 16)	0
Flatten	(None, 8192)	0
Dropout	(None, 8192)	0
Dense (ReLU activation)	(None, 500)	4096500
Dropout	(None, 500)	0
Dense (ReLU activation)	(None, 200)	100200
Dropout	(None, 200)	0
Dense (Soft max activation)	(None, 20)	4020
Total Parameter :		4210288

Notes: (a) Layer listed in a sequential order. (b) "None" on the first column shape means the batch size

Figure 4 presents how the model of Javanese letters classifier is built, with the preprocessing stage required the use of syntax Conv2D with kernel size 3x3 in the convolution process. Furthermore, the activation function was also adopted to transform the neural network into a non-linear variety, and the ReLU (Rectified Linear Unit) activation function and "same" padding were applied in this phase, due to the fact that the threshold range used was between 0 and infinity. Therefore, after the completion

of this process, the next step involves the reduction of matrix size, which requires the use of max pooling operation with filter size of 2x2. This was continuously inserted between the convolution layers in the CNN model, with the aim of reducing the number of parameters, computation network, and also to control over-fitting.

```
def create_model(ep_model):
    epochs = ep_model
    lrate = 0.01
    decay = lrate / epochs
    model_javaneseLetters = Sequential()

    model_javaneseLetters.add(Conv2D(32, (3, 3), input_shape=(32, 32, 1), activation='relu', padding='same'))
    model_javaneseLetters.add(Dropout(0.2))
    model_javaneseLetters.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
    model_javaneseLetters.add(MaxPooling2D(pool_size=(2, 2)))
    model_javaneseLetters.add(Flatten())
    model_javaneseLetters.add(Dropout(0.2))
    model_javaneseLetters.add(Dense(500, activation='relu', kernel_constraint=maxnorm(3)))
    model_javaneseLetters.add(Dropout(0.2))
    model_javaneseLetters.add(Dense(200, activation='relu', kernel_constraint=maxnorm(3)))
    model_javaneseLetters.add(Dropout(0.2))
    model_javaneseLetters.add(Dense(20, activation='softmax'))

    model_javaneseLetters.compile(loss='categorical_crossentropy', optimizer='adadelta', metrics=['accuracy'])

    return model_javaneseLetters
```

Fig. 4. Implementation of Javanese Letters Model.

This study uses the 2-tier(client/server) architecture to propose the Javanese letters recognizer, placing mobile apps as the client that invokes a service implemented by the server (Javanese letters model), through the Application Program Interface (API). In this context, the process involves sending an image of the specific character as a parameter to the server-side through API, which receives the request, and asks the model to recognize the client request, and the results are subsequently sent back to the app. Furthermore, separating the architecture into 2-tier prompts the appropriate allocation of resources and optimal balance needed to conduct the calculation, especially for mobile devices with limited computation and storage resources [35]. This also ensures that clients do not need to save the entire model, and also perform the computation on personal devices, but by simply placing the load on a server-side with better resource. In addition, separating the client and classifier model also tends to be beneficial in terms of compatibility of those that allow developers to implement the client side in various app platforms. Therefore, the presentation layer was set for this purpose, then the logic and resource layer were placed in the web server-side.

Moreover, the communication between both parties, as well as the devices with which clients send API with a request body containing an image in base 64 format to /api/detect/ in post mode, the server responses to the classification, and the score result, are in JSON format. Moreover, the API JSON format is shown in Fig 5, while the big picture of the proposed Javanese letters classifier with mobile client architecture is described in Fig. 6.

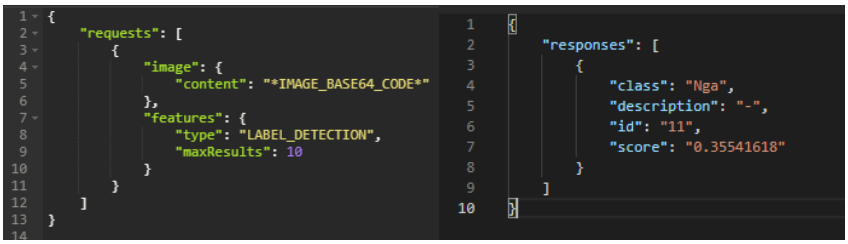


Fig. 5. Sample of the API request and response body between client and server.

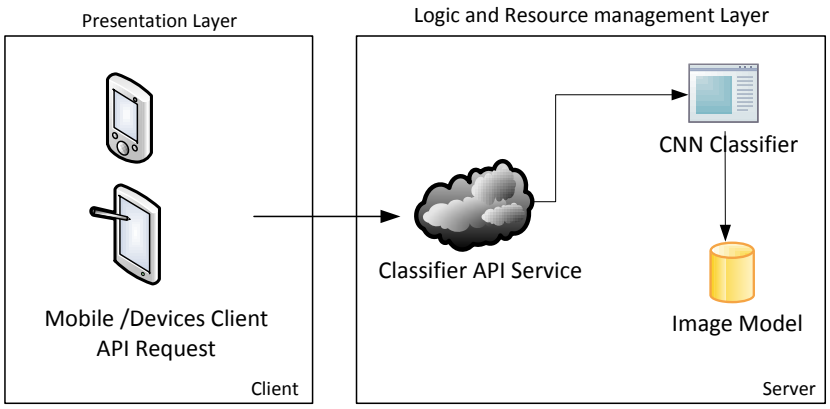


Fig. 6. Sample of the API request and response body between client and server.

4 Result and Discussion

The proposed Javanese letters classifier for mobile platform using 2-tier architecture has been built and tested in this study, using built-in server Python and g-unicorn as web API. In addition, the device client was created with Android Studio, including the camera and media access to capture images of Javanese letters, as well as internet permission to send the classification request to the server. Fig. 7 is the screenshot of android client devices' application, which shows the process of classification, and API response time was also examined, which is the estimate of the total stretch from the client to send a request to the server prior to the receipt of classification result, which only shows the need of 1935 ms on average for a single request. Meanwhile, based on accuracy, though it is a simple model and dataset, the result was fairly around 86.68% of accuracy on the model with 50 training epoch, and details of testing accuracy and cost results are summarized in Table 2.

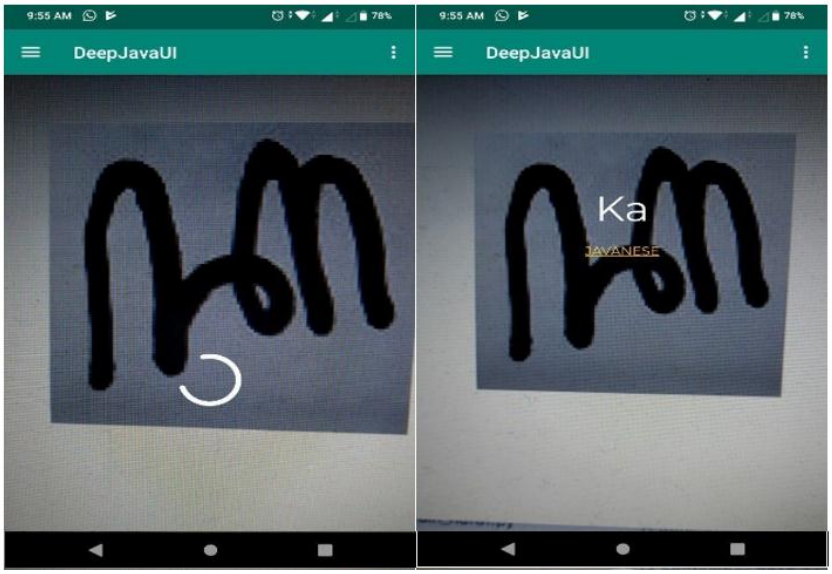


Fig. 7. Screenshots of Android Client Devices for the Javanese Letters

Table 2. API response and accuracy testing

API Response Time Testing		**Testing Accuracy of Model Training**	
N	***Response Time (ms)*** [(a)]	***Epoch***	***Accuracy*** [(b)]
1	1754	5	26.74%
2	2050	10	57.44%
3	1800	25	75.60%
4	2040	50	86.68%
5	2030	75	82.12%
Average	1935		

Notes: (a) Response time based on a network with 20 Mbps and 30 ms latency using 100 kb image size per request; (b) Accuracy based on the selected 100 images testing dataset

The result indicates that the implementation of 2-tier architecture in mobile character recognition possesses the capacity to produce accuracy results with around 86.68%. In addition, a comparison to prior studies developed based on CNN algorithm in non-mobile platform by [17] showed the accuracy testing value to be around 89 %, and the outcome of this research was acceptable with not so much difference. Furthermore, the response time also indicated the tolerated computation time, showing the combination of mobile server-client architecture and CNN technique, which is acceptable for implementation.

5 Conclusion and Future Research

The result of this study shows the possibility of conducting Javanese letters classifier with 2-tier architecture, and testing on the Android mobile operating system. In addition, the overall performance for a single classification request was observed to be fast, with an average response time of 1935 ms each, and a fair accuracy result, with about 86.68%, using 50 training epoch. Furthermore, the letters classifier with client-server architecture tends to enhance the device client flexibility, based on the propensity of being built on various operating platform. However, there is need for support by the interface, in order to invoke the classifier system from outside, due to the fact that most computation processes are carried out on the server. Therefore, the only downside recorded was that every request must be sent to the network, which demands a reliable network connection.

This study proposes the use of 2-tier architecture to provide a system that is capable of providing an acceptable level of accuracy in recognizing Javanese alphabet in a mobile platform using the CNN method, without sacrificing the computational resources needed. The result, therefore, proves the high propensity of system development using mobile client-server architecture and CNN to provide a tolerable level of accuracy without sacrificing high cost on computation resource and time. This study allocates almost all workings on the server-side, while neglecting the advantage of client resource in handling other significant aspects. Hence, there is a possibility of implementing a fat client by allocating a proportional load to the mobile machine. This is, however, a recommendation for further study, especially for non-italic character recognition apps in the platform.

6 References

[1] X. Zhang, M. Lin and J. Sun, "ShuffleNet : An Extremely Efficient Convolutional Neural Network for Mobile Devices," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6848–6856, 2018. https://doi.org/10.1109/cvpr.2018.00716

[2] Y. Kim, E. Park, S. Yoo, T. Choi, L.Yang and D. Shin, "Compression of Deep Convolutional Neural Networks for Fast and Low Power Mobile Applications," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–16, 2016.

[3] J. Wu, C. Leng, Y. Wang, Q. Hu and J. Cheng, "Quantized Convolutional Neural Networks for Mobile Devices," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4820–4828, 2016. https://doi.org/10.1109/cvpr.2016.521

[4] Y. Weng and C. Xia, "A New Deep Learning-Based Handwritten Character Recognition System on Mobile Computing Devices," Mobile Networks and Applications, 2019. https://doi.org/10.1007/s11036-019-01243-5

[5] M. Ati and M. Ahmed, "Augmented Reality Enhanced Computer Aided Learning for Young Children," in IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), pp. 129–133, 2018. https://doi.org/10.1109/iscaie.2018.8405457

[6] M. J. C. Samonte, A, S. B. Bahia, J. G. J. Del Monte, J. A. J. Gonzales and M. V. Sultan, "Assistive Mobile App for Children with Hearing & Speech Impairment Using Character and Speech Recognition," in ICIBE' 18 Proceedings of the 4th International Conference on

Industrial and Business Engineering, pp. 265–270, 2018. https://doi.org/10.1145/3288155.3288182

[7] S. Papadakis and M. Kalogiannakis, "Mobile educational applications for children: what educators and parents need to know," Int. J. Mobile Learning and Organisation, vol. 11, no. 3, p. 256–277, 2017. https://doi.org/10.1504/ijmlo.2017.085338

[8] J. Kuo, C. Huang, W. Liao and C. Huang, "HuayuNavi : A Mobile Chinese Learning Application Based on Intelligent Character Recognition," Edutainment Technologies. Educational Games and Virtual Reality/Augmented Reality Applications, p. 346–354, 2011. https://doi.org/10.1007/978-3-642-23456-9_63

[9] Z. Alom, P. Sidike, M.Hasan, T. M. Taha and V. K. Asari, "Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks," Computational Intelligence and Neuroscience, 2018. https://doi.org/10.1155/2018/6747098

[10] A. Jain and B. K. Sharma, "Analysis of Activation Functions for Convolutional Neural Network based MNIST Handwritten Character Recognition," International Journal of Advanced Studies of Scientific Research, vol. 3, no. 9, p. 68–74, 2018.

[11] X. Zhang, F. Yin, Y. Zhang, C. Liu and Y. Bengio, "Drawing and Recognizing Chinese Characters with Recurrent Neural Network," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, p. 849–862, 2018. https://doi.org/10.1109/tpami.2017.2695539

[12] S. Misawa, M. Taniguchi, Y. Miura and T. Ohkuma, "Character-based Bidirectional LSTM-CRF with words and characters for Japanese Named Entity Recognition," in Proceedings of the First Workshop on Subword and Character Level Models in NLP, pp. 97–102, 2017. https://doi.org/10.18653/v1/w17-4114

[13] S. Lee, Y. Sung, Y. Kim and E. Cha, "Variations of AlexNet and GoogLeNet to Improve Korean Character Recognition Performance," Journal of Information Processing Systems, vol. 14, no. 1, p. 205–217, 2018.

[14] P. Chomphuwiset, "Printed Thai Character Segmentation and Recognition," in 2017 4th IEEE International Conference on Soft Computing and Machine Intelligence Printed, pp. 123–127, 2017. https://doi.org/10.1109/iscmi.2017.8279611

[15] A. El-sawy, M. Loey and H. El-Bakry, "Arabic Handwritten Characters Recognition using Convolutional Neural Network," WSEAS Transactions on Computer Research, vol. 5, 2017.

[16] Y. Sugianela and N. Suciati, "Javanese Document Image Recognition Using Multiclass Support Vector Machine," Communication & Information Technology Journal, vol. 13, no. 1, pp. 25-30, 2019. https://doi.org/10.21512/commit.v13i1.5330

[17] C. K. Dewa, A. L. Fadhilah and Afiahayati, "Convolutional Neural Networks for Handwritten Javanese Character Recognition," Indonesian Journal of Computing and Cybernetics Systems, vol. 12, no. 1, p. 83–94, 2018. https://doi.org/10.22146/ijccs.31144

[18] G. S. Budhi and R. Adipranata, "Handwritten Javanese Character Recognition Using Several Artificial Neural Network Methods," Journal of ICT Research and Applications, vol. 8, no. 3, p. 195–212, 2015. https://doi.org/10.5614/itbj.ict.res.appl.2015.8.3.2

[19] A. Widiarti and P.N. Wastu, "Javanese Character Recognition Using Hidden Markov Mode," International Journal of Computer, Electrical, Automation, Control and Information Engineering, vol. 3, no. 9, p. 2201–2204, 2009.

[20] X. Ji, Q. Yu, Y. Liu and S. Kong, "A Recognition Method for Italian Alphabet Gestures Based on Convolutional Neural Network," Huang DS., Bevilacqua V., Premaratne P. (eds) Intelligent Computing Theories and Application. ICIC 2019. Springer International Publishing, vol. 11643, p. 653–664, 2019. https://doi.org/10.1007/978-3-030-26763-6_63

[21] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[22] A. Dutt and AashiDutt, "Handwritten Digit Recognition Using Deep Learning," International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 6, no. 7, pp. 990-997, 2017.
[23] T. Pham and P. Le-hong, "End-to-End Recurrent Neural Network Models for Vietnamese Named Entity Recognition: Word-Level," Communications in Computer and Information Science, vol. 781, p. 219–232, 2018. https://doi.org/10.1007/978-981-10-8438-6_18
[24] K. S. Younis, "Arabic Handwritten Character Recognition Based on Deep Convolution-al Neural Networks," Jordanian Journal of Computer and Information Technology (IJCIT), vol. 3, no. 3, pp. 186-200, 2017.
[25] N. Ly, C. T. Nguyen, K. C. Nguyen and M. Nakagawa, "Deep Convolutional Recurrent Network for Segmentation-free Offline Handwritten Japanese Text Recognition," in 14th IAPR International Conference on Document Analysis and Recognition, 2017. https://doi.org/10.1109/icdar.2017.357
[26] Y. Harjoseputro, "Classifying Javanese Letters with Convolutional Neural Network (CNN) Method," in The First International Conference and Exhibiton on Sciences and Technology (ICEST), Labuan Bajo, 2018.
[27] L. N. Huynh, R.K. Balan and Y. Lee, "DeepSense: A GPU-based Deep Convolutional Neural Network Framework on Commodity Mobile Devices," in WearSys '16 Proceedings of the 2016 Workshop on Wearable Systems and Applications, pp. 25–30, 2016. https://doi.org/10.1145/2935643.2935650
[28] L. Tob, A. Ducournau, F. Rousseau, G. egoire Mercier and R. Fablet, "Convolutional Neural Networks for Object Recognition on Mobile Devices : a Case Study," in 23rd International Conference on Pattern Recognition (ICPR), pp. 3530–3535, 2016. https://doi.org/10.1109/icpr.2016.7900181
[29] K. P. Danukusumo, Pranowo and Maslim, "Indonesia ancient temple classification using convolutional neural network," in The 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), Yogyakarta, 2017. https://doi.org/10.1109/iccerec.2017.8226709
[30] A. Santoso and G. Ariyanto, "Implementasi Deep Learning Berbasis Keras untuk Pengenalan Wajah," Jurnal Emitor, vol. 18, no. 1, pp. 15-21, 2018. https://doi.org/10.23917/emitor.v18i01.6235
[31] I. W. S. E. Putra, A. Y. Wijaya and R. Soelaiman, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101," Jurnal Teknik ITS, vol. 5, no. 1, pp. A65-A69, 2016. https://doi.org/10.12962/j23373539.v5i1.15696
[32] C.-Y. Lee, P. W. Gallagher and Z. Tu, "Generalizing Pooling Functions in Convolution-al Neural Networks:Mixed, Gated, and Tree," Artificial Intelligence and Statistics, pp. 464-472, 2016.
[33] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein and A. Berg, "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211-252, 2015. https://doi.org/10.1007/s11263-015-0816-y
[34] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár and C. Zitnick, "Microsoft COCO: Common Objects in Context," Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol. 8693, 2014. https://doi.org/10.1007/978-3-319-10602-1_48
[35] M. R. Rahimi, N. Venkatasubramanian, S. Mehrotra and A. Vasilakos, "MAP Cloud : Mobile Applications on an Elastic and Scalable 2-Tier Cloud Architecture," in IEEE/ACM Fifth International Conference on Utility and Cloud Computing MAPCloud:, pp. 83–90, 2012. https://doi.org/10.1109/ucc.2012.25

7 Authors

Yulius Harjoseputro is a lecturer at Department of Informatics, Faculty of Industrial Technology, Universitas Atma Jaya Yogyakarta-Indonesia, where he completed a Graduate studies in a Master study Program on Informatics Engineering. The focus of his research is on image processing, deep learning and mobile computing.

Yonathan Dri Handarkho is a lecturer at Department of Informatics, Faculty of Industrial Technology, Universitas Atma Jaya Yogyakarta-Indonesia. He is a Ph.D. candidate at Vincent Mary School of Science and Technology, Assumption University of Thailand, holding a Master of Engineering degree from Universitas Gadjah Mada-Indonesia. In addition, his research interests are based on Information Technology acceptance, CRM Technology, and mobile computing.

Heronimus Tresy Renata Adie is on the Bachelor of Engineering on Informatics program, Faculty of Industrial Technology, Universitas Atma Jaya Yogyakarta-Indonesia.

Article submitted 2019-08-11. Resubmitted 2019-09-24. Final acceptance 2019-09-24. Final version published as submitted by the authors.