JAVANESE CHARACTER RECOGNITION WITH REAL-TIME DETECTION USING CONVOLUTIONAL NEURAL NETWORK

Submitted as partial Fulfillment of the Requirements for Getting Bachelor Degree of Informatics
Departement Faculty of Communication and Informatics

By:

WAHYU PURNAMA WIDYA LEKSANA

L200164012

DEPARTMENT OF INFORMATICS

FACULTY OF COMMUNICATION AND INFORMATICS

UNIVERSITAS MUHAMMADIYAH SURAKARTA

2023

APPROVAL

JAVANESE CHARACTER RECOGNITION WITH REAL-TIME DETECTION USING CONVOLUTIONAL NEURAL NETWORK

SCIENTIFIC PUBLICATION

by:

WAHYU PURNAMA WIDYA LEKSANA
L200164012

Has been examined and approved for testing by:
Supervisor

Fajar Supriawan, S.T., M.Eng.Sc., Ph.D
NIK.924

ACCEPTANCE

JAVANESE CHARACTER RECOGNITION WITH REAL-TIME DETECTION USING CONVOLUTIONAL NEURAL NETWORK

BY

WAHYU PURNAMA WIDYA LEKSANA

L200164012

Has been maintained in front of the Board of Examiners
Faculty of Communication and Informatics
Universitas Muhammadiyah Surakarta
On Saturday, January 21th **2023**
and declared to have fulfilled the requirements

Team of Examiners:

1. **Fajar Suryawan, S.T., M.Eng.Sc., Ph.D.** (.....)
   (Chief of Examiners Board)
2. **Nurgiyatna, S.T., M.Sc., Ph.D.** (.....)
   (Member I of Examiners Board)
3. **Maryam, S.Kom., M.Eng.** (.....)
   (Member II of Examiners Board)

Dean
Faculty of Communication and Informatics

Nurgiyatna, S.T., M.Sc., Ph.D.
NIK.881

Head
Department of Informatics

Nurgiyatna, S.T., M.Sc. Ph.D.
NIK.1305

ii

STATEMENT

I hereby declare that in this thesis no work has ever been submitted to obtain a degree in college and to the best of my knowledge no work or opinion has ever been written or published by anyone else, published here in documents and in References.

If later there is proven untruth in my statement above, then I will take full responsibility.

Surakarta, January 31th 2023

Writer

WAHYU PURNAMA WIDYA LEKSANA

L200164012

JAVANESE CHARACTER RECOGNITION WITH REAL-TIME DETECTION USING CONVOLUTIONAL NEURAL NETWORK

Abstrak

Huruf jawa merupakan warisan budaya yang perlu dilestarikan sehingga tidak menghilang di masa yang akan datang. Dalam hal ini banyak peneliti mengambil huruf jawa sebagai objek penelitian mereka. Berbagai metode dikembangkan guna menciptakan model untuk pengenalan huruf jawa yang paling optimal. Model jaringan saraf konvolusional dipilih oleh peneliti dikarenakan kemampuan model ini dikenal baik dalam mengklasifikasikan citra kedalam kelas klasifikasi yang diinginkan. Citra huruf jawa sebanyak 3240 dikumpulkan dan diklasifikasikan secara manual kedalam 20 label untuk digunakan sebagai himpunan data. Model ini kemudian dilatih menggunakan himpunan data tersebut sehingga menghasilkan model yang siap digunakan untuk mengenali huruf jawa baru yang didapatkan dari pengambilan citra secara real-time menggunakan webcam. Model jaringan saraf konvolusional ini mendapatkan akurasi 99.17% dalam 7 kali epoch. Sistem yang dirancang berhasil mengenali aksara jawa dengan kecepatan untuk 1 *frame* selama 0.13 detik. Uji coba menggunakan citra dari webcam kedalam model jaringan saraf konvolusional ini mendapatkan hasil mampu mengenali ke 20 huruf jawa dengan akurasi tertinggi 96.67% untuk aksara *ga* dan 56.24% untuk aksara *wa*.

Kata Kunci: Huruf Jawa, Jaringan Saraf Konvolusional, Tensorflow, Real-time

Abstract

Style Javanese letters are a cultural heritage that needs to be preserved so they don't disappear in the future. In this case, many researchers take Javanese letters as their research object. Various methods were developed to create a model for the most optimal recognition of Javanese letters. The convolutional neural network model was chosen by researchers because of its well-known ability to classify images into the desired classification class. 3240 Javanese letter images were collected and classified manually into 20 labels to be used as data sets. This model is then burned using the data set to produce a model that is ready to be used to identify new Javanese letters obtained from real-time image capture using a webcam. This convolutional neural network model achieves 99.17% accuracy within 7 epochs. The designed system successfully recognize the javanese character at a speed of 1 frame for 0.13 seconds. The trial using the image from the webcam into the convolutional neural network model obtained results that were able to identify 20 javanese characters with the highest being *ga* character with 96.67% accuracy and the lowest being *wa* character with 56.24%.

Keywords: Javanese Character, Convolutional Neural Network, Tensorflow, Real-time

1. PRELIMINARY

Javanese have many cultural heritage and customs that have been applied in the daily lives of the society. Some problem arise when the effect of development era and cultural exchange engage the society to advanced their cultural and it result in some of javanese custom got forgotten gradually. Javanese culture has inherent identity, one of which is the character chosen by studyers by combining technology to create media that can facilitate users in learning the script mentioned. The characters of Javanese script themselves are very diverse and not the same with each other used in the past before the congress in 1922 which discussed the writing rules of Javanese characters so that there was harmony in Javanese characters in their writing.(Jawi, 1926)

"Javanese Character Recognition" is a theme of character analysis of Javanese characters aimed specifically at young Javanese generations so as not to forget their identity as Javanese because after all it can be seen directly that the characteristics of Javanese culture in Javanese society began to fade. The use of Javanese script is still used in Javanese Language Education materials, writing on arches, as well as writing on Javanese cultural things. Weakening on cultural problem such the javanese can't read the local script can be appointed as study for recognizing the javanese script (Wibowo et al., 2012). Study of recognition of the script into a computer's data can be compiled using image recognition algorithms namely CNN (Convolutional Neural Network) which can be arranged into a simple program, this already implemented before such the study accomplished by (Dewa et al., 2018) CNN selected as model to recognize javanese character in their softtware. The CNN model will be trained along with one hidden layer MLP model and two hidden layer MLP model, all of model will train the same dataset of javanese character. The dataset manually collected into 2000 data belongs to 20 classes of basic javanese character. Based on the study by quantify the performance with k-folds cross validation to acquire accuracy and training time, the CNN model's accuracy is better than the MLP model but take longer time to train the data and the accuracy of CNN model cannot reach 90% for all folds.

Some previous studies that specifically discussed the character recognition of Javanese characters using a variety of algorithms including the introduction of Javanese letters using CNN in scene image. The study using CNN with ICDAR 2003 as training set and synthetic data then mixes with the collected Javanese letter contain of 90 scene images. The result show good result on detecting the letter by creating bounding boxes on the scene images with 96% accuracy (Afakh, Risnumawan, Anggraeni, Tamara, & Ningrum, 2017).

The use of HMV (Hidden Markov Model) that uses *legena/carakan* characters with a total of 20 characters with 50 data that has been processed into 72x72 pixels and then extracted through discrete pixel information from each vector. The extraction of the character specified into 4 different type. The study uses 1000 javanese character of *legena* characters using 5-fold cross validation method and divide 800 for training set and 200 for test set. The work giving good result on all 4 types with the best extraction being 1V feature with 85.7% (Widiarti & Wastu, 2009)

The study done by(Wibowo et al., 2017) introduce their work on solution to save Javanese manuscript by digitalizing it. Using 11,000 dataset of javanese character written by 50 different people with their own writing types and styles with 20 character types. 2 CNN model used in this study with 8 different configuration. Its result show that some character such as Ha, Ka, Ta, La, Ya, and Nya need more width to avoid misclassification during training using CNN, able to recognize every character with no less than 85% , and conclude the best model to recognize javanese character with accuracy, precision, recall and F-1 score respectively 94.57%, 94.75%, 94.57%, 94.66%.

The study done by (Budhi & Adipranata, 2015) using 5 different method of ANN(Artificial Neural Network) they are bidirectional associative memory network, a counterpropagation network, an evolutionary neural network, and a combination of chi2 and a backpropagation neural network and using 31 classes with 620 datasets of javanese characters. The conclusion of this study conclude some neural networks that can't be used to recognize javanese character, and show the best method in the tested neural network is the combination of the Chi2 method and the back propagation neural network with 98% accuracy rate on data trained beforehand and 73% on data trained not beforehand.

The use of LVQ (Learning Vector Quantizations) which uses 32 dataset divided into 30 training data and 2 data as initial weight for 2 javanese characters which are "*Ra*" and "*Ga*". The data processed before entered into the neural network with cropping, thresholding, scalling and feature extraction. The study concluded that the LVQ have 56.61% accuracy to recognizing the mentioned character (Mafrur et al., 2015)

Therefore CNN will be used as the model to create program for recognize the javanese character as its shown to be better image classification model compared to other especially on the

accuracy to train the dataset. To differentiate with previous study, this study will create a real-time video captured by webcam to test the trained CNN model.

2. METHOD

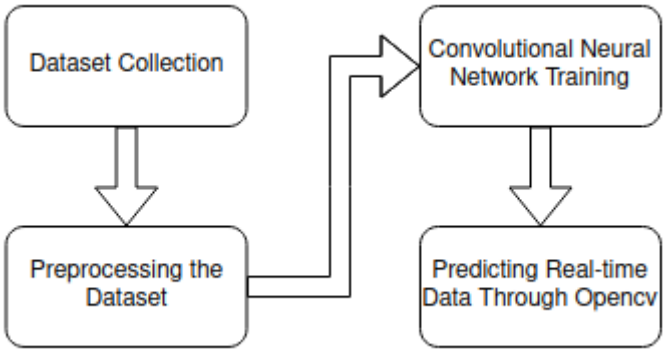

Figure 1. Flowchart of proposed method

Method used in this study shown in Figure 1 will be explained as follow.

2.1 Dataset Collection

In this study, researcher collecting 20 types of javanese character representing *ha, na, ca, ra, ka, da, ta, sa, wa, la, ma, ga, ba, ta, and nga* this basic character usually called as *legena/carakan* characters each of them represent as 20 label consist 176 images as dataset, totaling 20x176 equal to 3520 data will become the training dataset and another data with 20x13 data totaling 260 data will become the testing data. The training dataset obtained by painting manually on Pinta(Application for drawing), by separating fullset characters i.e. Figure 2, and dataset collected by (Phiard,2021) and (Hunafa, 2022). As for the testing data obtained by manually by writing with different style on paper by 5 person. An example of every shape of *legena* characters with its pronunciation can be seen in Figure 2.

Figure 2. *Legena* Characters.

2.2 Preprocessing the Training Dataset

The variety of images type in dataset will be simplified to become binarized, trimmed and resized data.

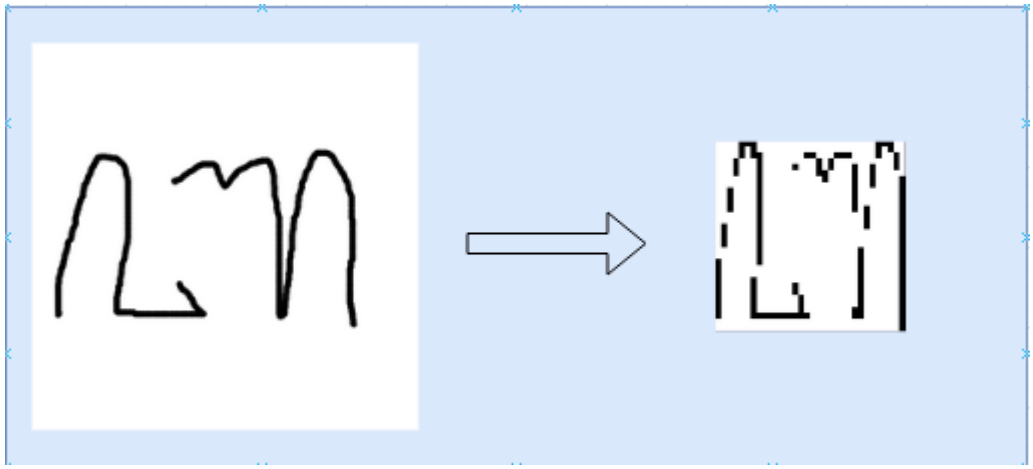

Figure 3. One of data in the dataset simplified

Converting data to binary is required to make CNN model can have dataset that have the same amount of channel with 1 or 0 number on every pixel. Trimming will cut unnecessary space making the later CNN model training become more clear and accurate. Resizing the image to 32x32 pixel for it can fit into the CNN model. Figure 3 ilustrate a raw data that have been processed to become prepared training data. Prepared training data obtained by processing every data on dataset with converting, trimming and resizing. The data will have structure 32x32 pixel with 1 channel stated as (32,32,1).

2.3 Convolutional Neural Network Training

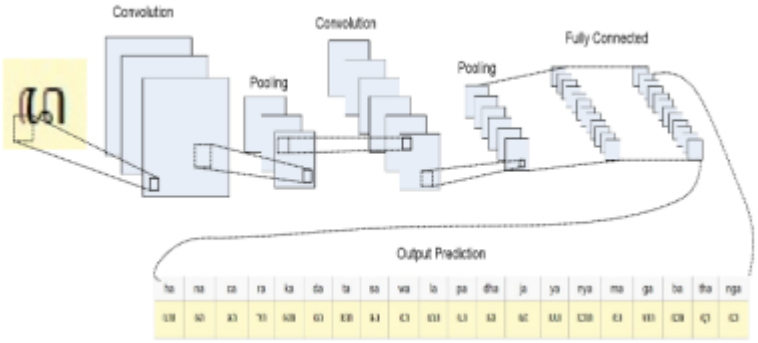

Figure 4. Example of CNN model

Figure 4 is an example of CNN model which explained by (Wibowo et al., 2012). The CNN model also explained by (Dewa, Fadhilah, & Afiahayati, 2018) and (Lorentius et al., 2019) in their study. As for the model in this study will use a CNN model with specification specified in Table 1.

Table 1. The CNN structure used in this research using input image 32x32 pixel with binary pixel

Number	Layer Type	Operation	Output Shape
1	Convolutional +Relu	32 (3x3)	(30, 30, 32)
2	Max Pooling	(2x2)	(15, 15, 32)
3	Convolutional +Relu	64 (3x3)	(13, 13, 64)
4	Max Pooling	(2x2)	(6, 6, 64)
5	Convolutional +Relu	64 (3x3)	(4, 4, 64)
6	Flatten		(1024)
7	Dense + Relu	64	(64)
8	Dense + Softmax	20	(20)

Table 1 ilustrate the model used in this study created by using Tensorflow(Tensorflow, 2021), the dataset will be fed to this model to create trained model which will be used for testing. This CNN model have 8 layer. 1st layer is convolutional+relu, the data (32,32,1) will be fed into the 1st layer. it will create 32 filters and calculating the input with 3x3 pixel called kernel with 1 strides to create output of tensor with size (30,30,32).

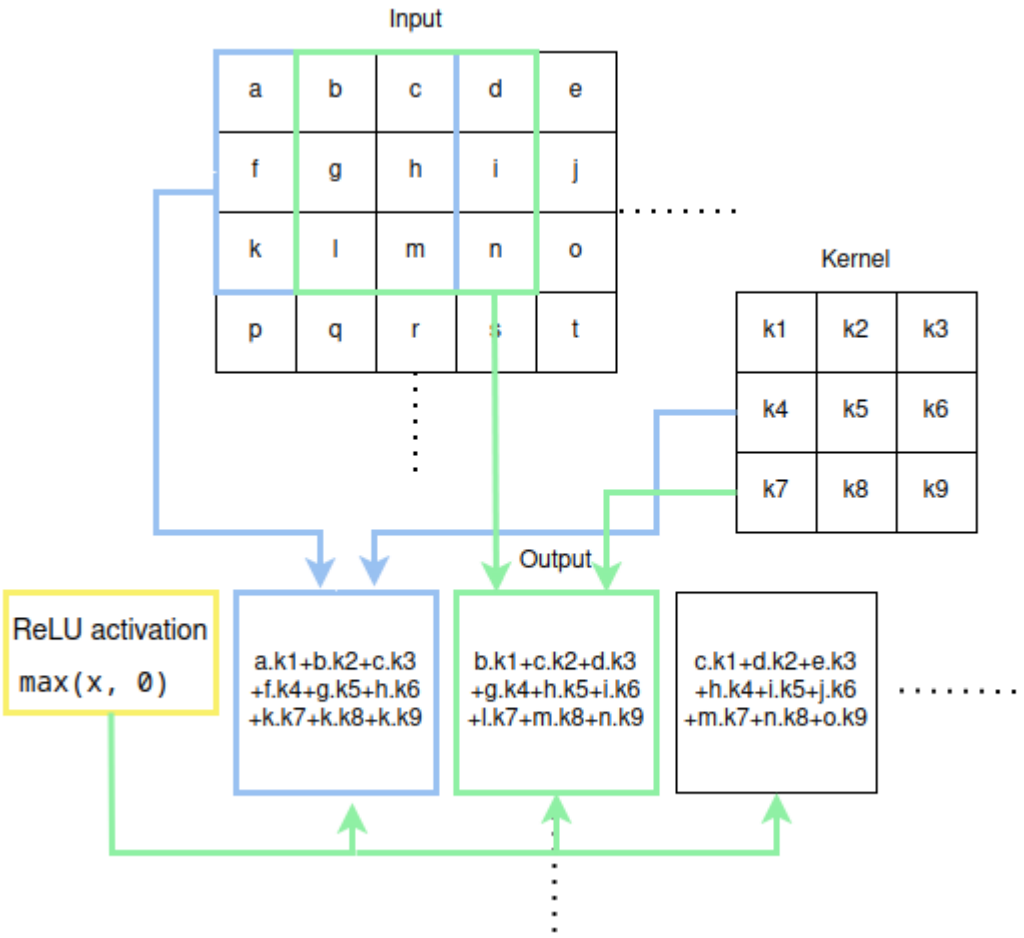

Figure 5. Convolutional operation graph with ReLu activation

In Figure 5 based on (Goodfellow et al, 2016) explained the 1st layer work on input image, the Tensorflow will compute the 3x3 pixel of image multiplied by the kernel with 3x3 pixel then the result will be included to ReLu activation function to create the output data before going to the next layer. ReLU activation function will take the maximum value but the value as 0 if the value less than 0. The strides mean how 3x3 pixel on the image shift to the next 3x3 pixel which in this case to be (1,1) as default of the tensorflow it will move 1 pixel to the right. 2nd layer is max pooling, its selecting the maximum value of 2x2 pixel over (1,1) stride in entire input image and will create tensor size become half of general size so it became (15,15,32). 3th layer do same thing as layer 1 but creating 64 filter so it get tensor (13,13,64). 4th layer will do the same as the 2nd layer by taking the maximum value over 2x2 pixel to get tensor (6,6,64). 5th layer do the same as 1st layer but creating 64 filter to get tensor (4,4,64). On 6th layer flatten the tensor making every element rewrited into 1 dimensional tensor with 1024 element. 7th layer making the array became 1 dimensional array with 64 elements by using dense function on Tensorflow and using relu activation which taking element that have max value over the calculated array. 8th layer dense with softmax activation making the array became 1 dimensional array

with 20 element representing label that used in this study. The 8th layer determine which label the input image is classified. The train accuracy and loss will be shown as the result of the trained model.

2.4 Predicting Real-time Data Through Opencv

The system for detecting the image from the testing data will be created by using Opencv. The system will have 2 different viewpoint, the first viewpoint used to show full image of the frame in rgb color that captured continously by webcam in real time with size 512x512 pixel and static bounding box in the center with size 172x172 pixel then the second viewpoint used to show the prepared testing data to be predicted through the model. The prepared testing data will be obtained by firstly taking the static bounding box image from the first viewpoint. The image then preprocessed with same process explained on section 2.2.

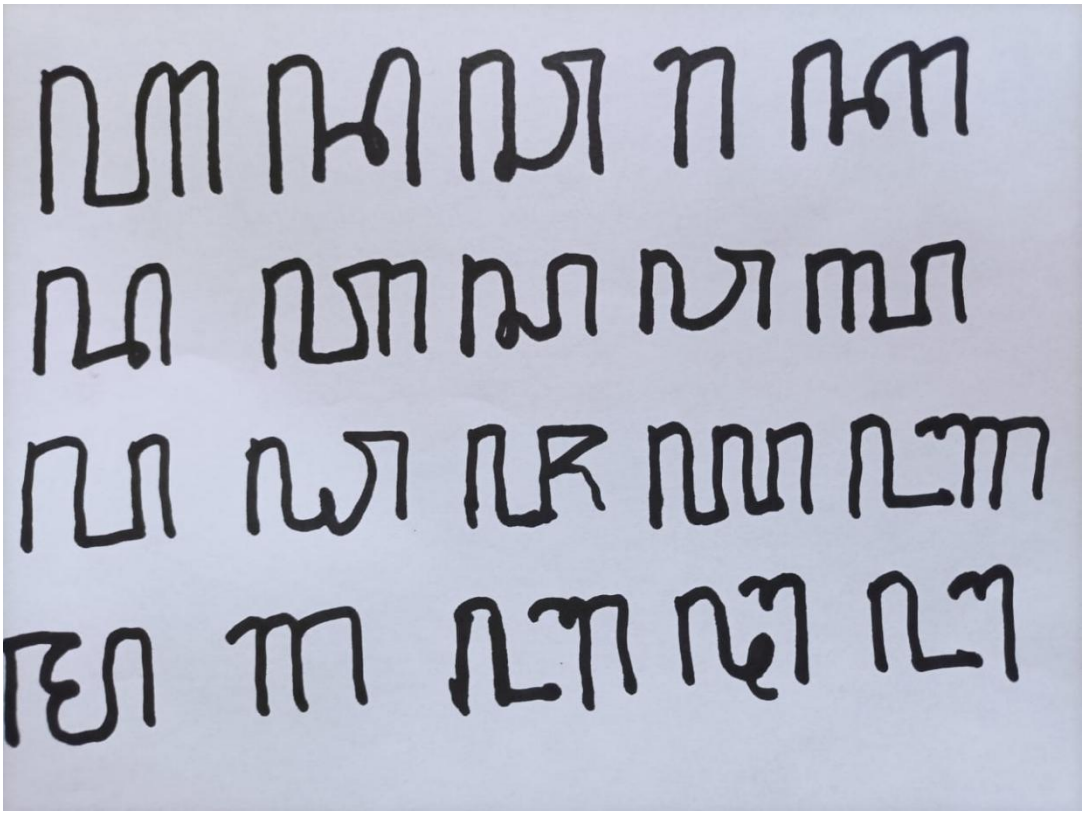

Figure 6. Testing data in form of sheet of paper

For testing researcher will use the testing data which have form of sheet of paper with 20 javanese characters that can be viewed on Figure 6.

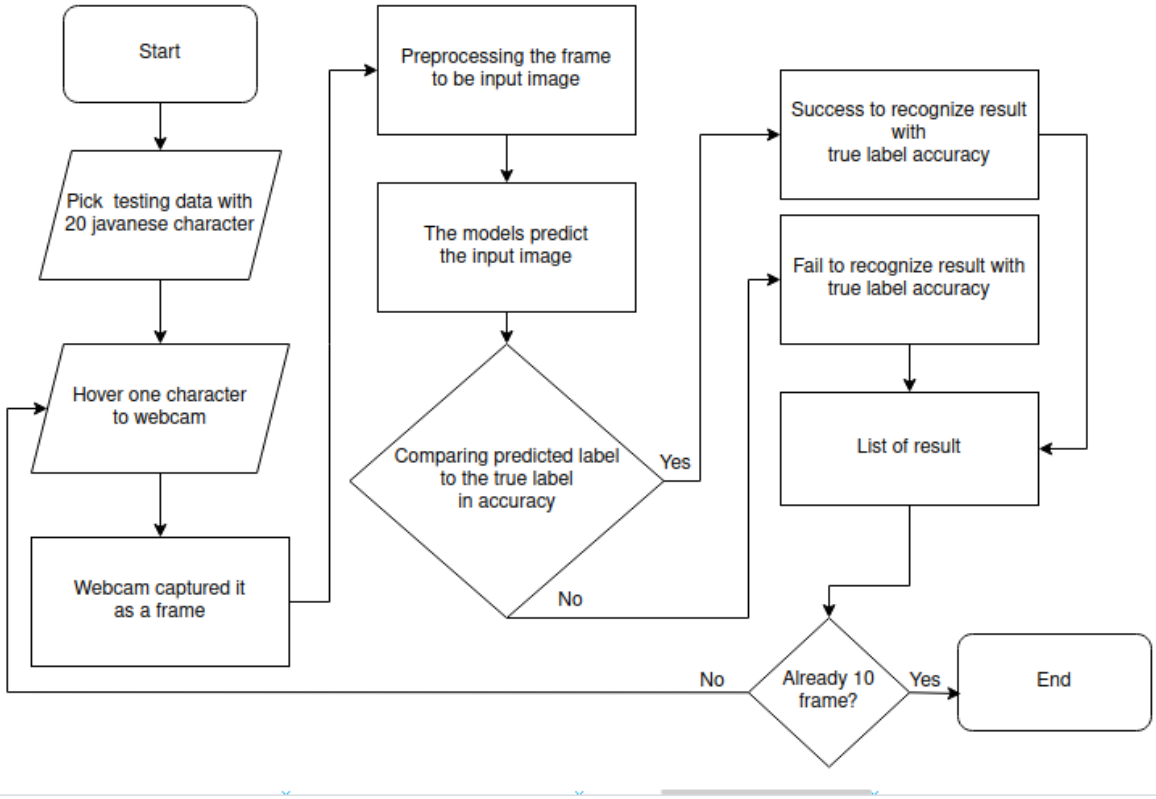

Figure 7 Flowchart of testing the model on the system

The testing illustrated as Figure 7. The researcher pick one sheet of paper of the testing data. Then researcher will hover 1 character per 1 character to the webcam. The hovering of 1 character to the webcam will be positioned on the center of first viewpoint and shown clearly on second viewpoint as 1 javanese character ready to be predicted. When the character have been positioned correctly, images of 10 frame will be taken from the second viewpoint to be fed to the trained model for predicting. One by one image will be predicted by the trained model resulting in 10 predicted label and its accuracy. Researcher will take comparation of these predicted label with its true label and make percentages of the average of accuracy to see the ability of trained model to be able to predict the javanese character correctly.

3. RESULTS AND DISCUSSION

The experiment was performed using laptop Ba004ax A10-9600P 2.4-3.3 GHz. The implementation code used python as the programming language with some library included Tensorflow as the main framework used to make the CNN model, Mathplotlib used to make graph showed in this study, Numpy used to do scientific calculation, Pillow used to process collection of images into the required dataset and Opencv used to make the input data for the prediction. Dataset classified into 20 different

label, converted into binary image, trimmed and resized. Then the CNN model were created and defined as Table 1 on the method section.

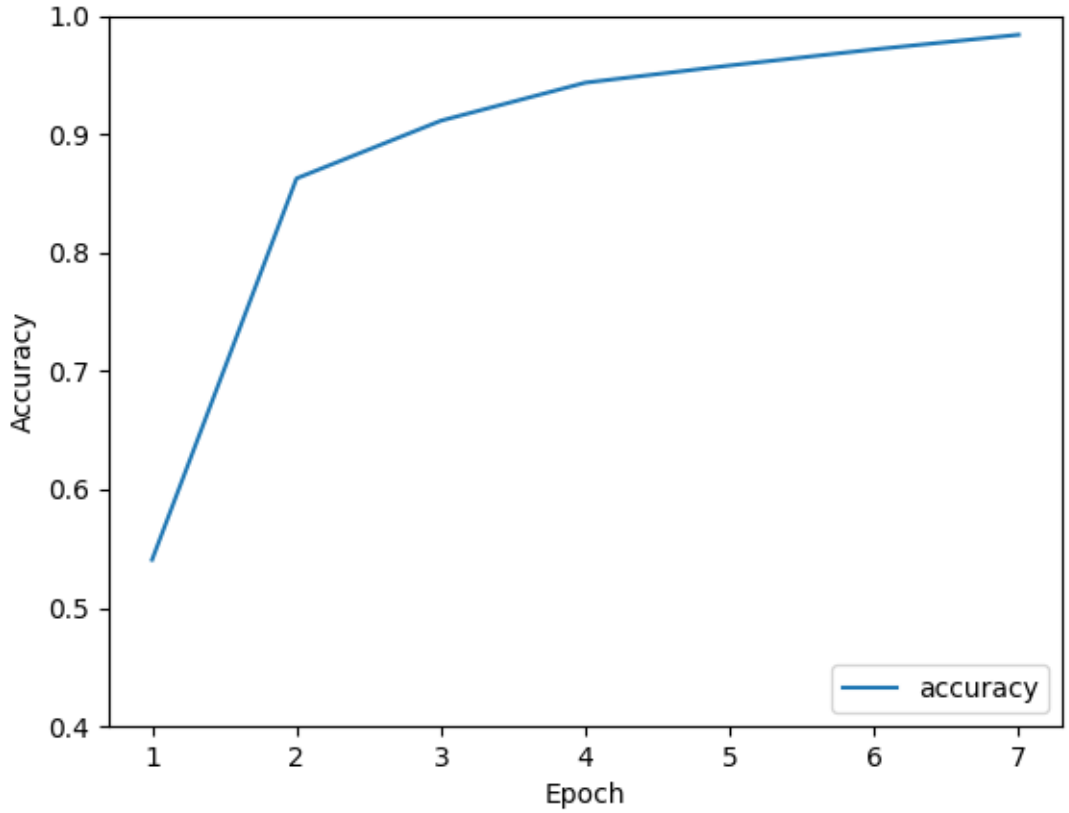

Figure 8. The trained model graph

Feed each preprocessed data to the created model to start model training for 7 epoch. The result of the training can be viewed From Figure 8 it show that model learn the dataset with graph of accuracy on trained data. Accuracy obtained from taking the frequency of matched train data true label with the prediction label of the same train data through the last layer(Tensorflow, 2021). The graph show the accuracy increased on every epoch mean the model training got better to predict javanese character from the train data every time training occur. The Accuracy on 7th epoch is 99.17%.

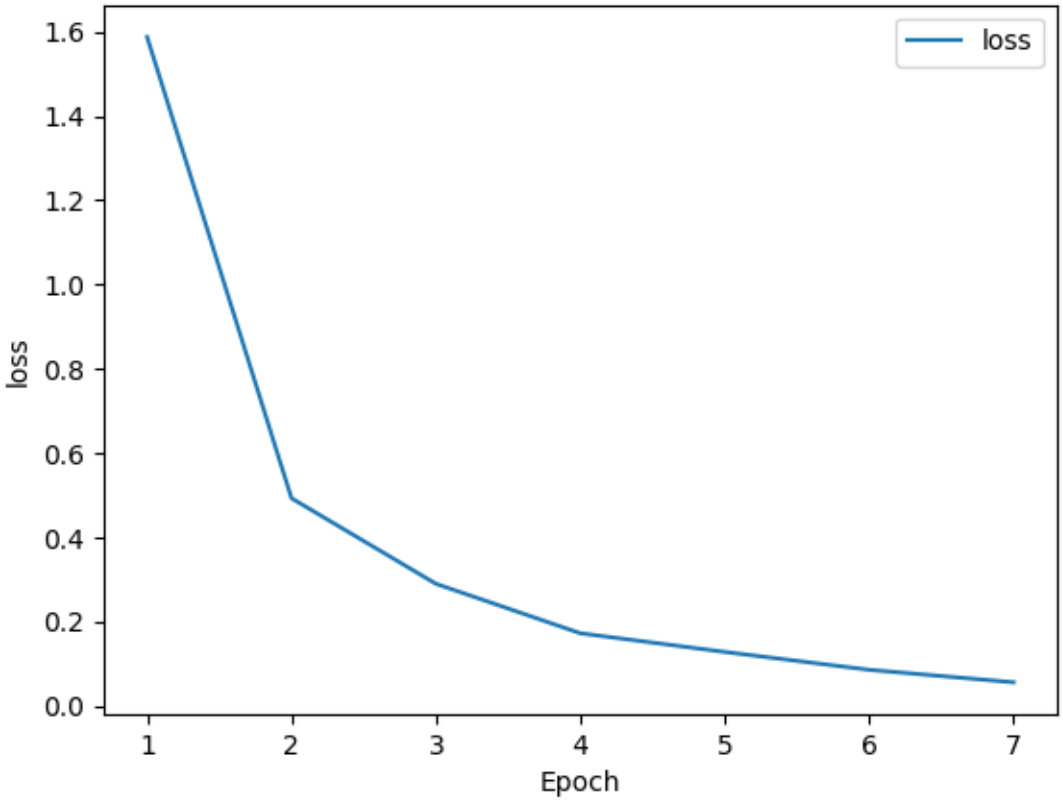

Figure 9. The loss graph of the model

The loss graph on Figure 9 show the learning process on the model. Loss obtained from the calculating crossentrophy function between trained data label fail with the data true label(Tensorflow, 2021). The loss gradually decreased over the epoch indicate the model ability to recognize the true label on trained data. The loss on the 7th epoch is 3.91%. The time to training the model is 28.87 second.

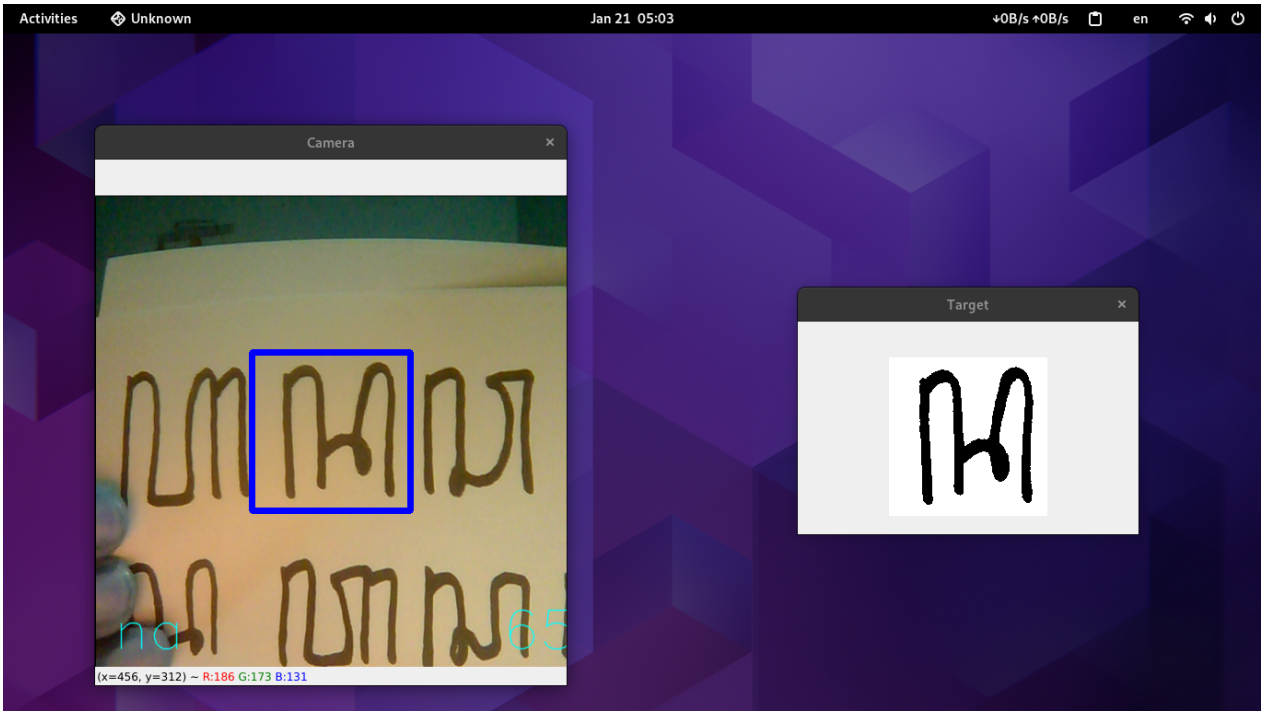

Figure 10. System with first viewpoint and second viewpoint

After the training process done, the system for predicting succesfully created with Opencv and have look as Figure 10. The right side is the first viewpoint show real time frame and the left side is the second view as input for the testing. In the first viewpoint there are predicted label on bottom left and accuracy on the bottom right. After 500 frame were tested resulting on average of time to predict the test input is 0.13 second.

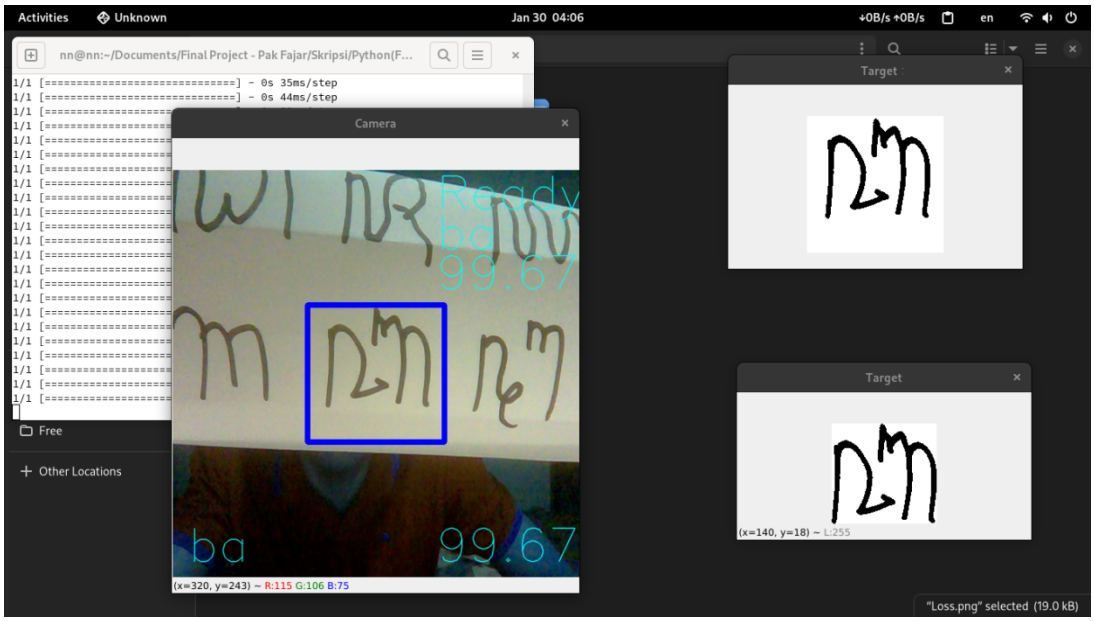

Figure 11 System with additional viewpoint for testing

For testing the third viewpoint on bottom which titled “Target” were created and can be seen on Figure 11. The second viewpoint on top titled “Target” is actually the same as second viewpoint on Figure 10. The addition use for easier the researcher to hover the designated javanese character so it is not change the system on the way model predict the input data or the testing data. The third viewpoint is preprocessed image from second viewpoint to be input data for the model. Then the additional text on top right of first viewpoint is the true label and true accuracy of the model on testing the input data. The testing data on this study consist of 13 data on each 20 labels. The result from predicting the testing data to the trained model on each data creating 10 frame with its predicted label and its accuracy. 13 data x 10 frame totaling 130 predicted label with its accuracy on one label of javanese character were created. Over all 20 label will be created 2600 predicted label with its accuracy. From those total researcher take the result of sysyem on compare predicted label to its true label if it matched then calculated as success or else will be calculated as fail and take the calculated average of accuracy on every label. The final result can be seen on Table 2, the percentage is the percentage from the average of accuracy this can determine the ability of trained model to predicting the testing data.

Table 2. The result of javanese character predicted to the CNN model

Character/Label	Percentage	Success | Fail	Character/Label	Percentage	Success | Fail
ha	91.41%	118|12	pa	66.81%	89|41
na	91.13%	122|8	*dha*	88.98%	117|13
ca	77.22%	105|25	*ja*	88.93%	117|13
ra	81.47%	105|25	*ya*	87.83%	114|16
ka	62.97%	83|47	*nya*	93.0%	125|5
da	92.85%	123|7	*ma*	70.89%	97|33
ta	63.11%	88|42	*ga*	96.67%	128|2
sa	90.15%	120|10	*ba*	88.96%	121|9
wa	56.24%	87|43	*tha*	92.26%	121|9
la	91.92%	121|9	*nga*	69.94%	98|32

The low accuracy on several character be caused by the similar image shape of the character to another and the lack of training image on the similar image shape of the tested character.

4. CONCLUSION

In this study, program which employs CNN model to perform classification task for recognizing javanese character recognition were succesfully created. The experiment conducted using Tensorflow framework to make the model trained with 3240 data that classified into 20 labels. Result of the graph show accuracy 99.17% with 3.91% loss in 29.87 second which its always detect the character and classified it into one of 20 label initiate before. In real-time detection the image of javanese characters acquired from camera are always detected which making the result and the accuracy of the result changed continously. The data sometimes recognize different result on one image if the image shake even just a little. The result were done on the testing data with 13 data on each label with the biggest percentage at character *ga* with 96.67% accuracy and lowest percentage at character *wa* 56.24% accuracy. This study result can be improved by increasing the dataset for training with more variety of font in javanese character.

References

Afakh, M. L., Risnumawan, A., Anggraeni, M. E., Tamara, M. N., & Ningrum, E. S. (2017). Aksara jawa text detection in scene images using convolutional neural network. *Proceedings - International Electronics Symposium on Knowledge Creation and Intelligent Computing, IES-KCIC 2017*. https://doi.org/10.1109/KCIC.2017.8228567

Budhi, G. S., & Adipranata, R. (2015). Handwritten javanese character recognition using several artificial neural network methods. *Journal of ICT study and Applications*, *8*(3), 195–212. https://doi.org/10.5614/itbj.ict.res.appl.2015.8.3.2

Dewa, C. K., Fadhilah, A. L., & Afiahayati, A. (2018). Convolutional Neural Networks for Handwritten Javanese Character Recognition. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *12*(1), 83. https://doi.org/10.22146/ijccs.31144

Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press. https://www.deeplearning.org.

Hunafa, H.(2022).Javanese Script(Aksara Jawa) Augmented | Kaggle .https://www.kaggle.com/datasets/hannanhunafa/javanese-script-aksara-jawa-augmented

Jawi, P. (1926). Sastra Sriwedari. *Yapa Insêtitut (Java Instituut)*, 65–80. Retrieved from https://www.sastra.org/bahasa-dan-budaya/pengetahuan-bahasa/2520-sastra-sriwedari

Lorentius, C. A., Adipranata, R., Tjondrowiguno, A., Studi, P., Informatika, T., Industri, F. T., … Siwalankerto, J. (2019). Pengenalan Aksara Jawa dengan Menggunakan Metode Convolutional Neural Network. *Jurnal Infra Petra*.

Mafrur, R., Andestoni, M., Ahdi, M. S., Fajri, N. S., & Muhantini, A. (2015). *Pengenalan Huruf Jawa Menggunakan Metode Learning Vector Quantization (Lvq)*. 1–6. https://doi.org/10.13140/RG.2.1.4354.0001

Sugianela, Y., & Suciati, N. (2019). Javanese Document Image Recognition Using Multiclass Support Vector Machine. *CommIT (Communication and Information Technology) Journal*, *13*(1), 25–30. https://doi.org/10.21512/commit.v13i1.5330

Phiard (2021). Aksara Jawa | Kaggle . https://www.kaggle.com/datasets/phiard/aksara-jawa

Tensorflow (2021, April 22). API Documentation | TensorFlow v2.11.0 . https://www.tensorflow.org/api_docs/

Wibowo, M.A., Soleh, M., Pradani, W., Hidayanto A.N., & Arymurthy, A.M. (2017). *Handwritten Javanese Character Recognition using Descriminative Deep Learning Technique*.

Widiarti, A., & Wastu, P. (2009). Using Hidden Markov Model Based. *International Scholarly and Scientific study & Innovation*, *3*(9), 2201–2204.