International Journal of Informatics and Computation (IJICOM)
Vol. 1, No.2, December, 2019
ISSN: 2685-8711, E-ISSN: 2714-5263

Design and Building Javanese Script Classification in The State Museum of Sonobudoyo Yogyakarta

Mohammad Diqi[1], Mujastia Feliati Muhdalifah[2]

Abstract

The Sonobudoyo State Museum is one of the state museums in Yogyakarta where stores historical objects like the Javanese script. This Javanese script presents in street names, especially in the city of Yogyakarta to represent local content for elementary, middle, and high schools. To read and understand Javanese script, people must learn it within a specified period, whereas with Latin letters is easier and faster to understand. The purpose of this paper is to design and build a Javanese script classification dataset to attract both adults, children, and parents as effective learning media. We construct the dataset by using Deep Learning with the Convolutional Neural Network (CNN). Stages of making a dataset are input data, the process of building models, and training can then recognize Javanese script images. We collect the dataset from the internet and several different people to train the computer machines. In this paper, we construct the Javanese script classification dataset to help users to detect Javanese characters. The results of this training the application of Javanese script classification can produce a certain level of recognition of Javanese script patterns in a real application.

Keywords:
Javanese script, dataset, deep learning, CNN

1. Introduction.

The museum is a permanent institution that serves the public needs, is open, does the collecting, conserving, resetting, communicating, and exhibiting tangible objects to the public. One of them is the Sonobudoyo State Museum. The Sonobudoyo State Museum is a museum of Javanese history and culture, including classical Javanese architectural buildings. This museum keeps a collection of Javanese culture and Javanese history. One of them is the Javanese script as one of the historical relics. The Javanese script is an icon of the city of Yogyakarta, every name of a public building and street by using the Javanese script. But if the Javanese script is not well studied, there will be mistakes in writing. And the need for preservation to keep the Javanese script still exists and is in demand by young people in this modern era. One way is the digitalization of Javanese script not only to preserve but as a media for learning that is more efficient and practical and can attract interest among children, adults, adolescents, and parents.

Machine Learning is a field of computer science that uses statistical techniques to provide the ability of computer information systems to be able to learn from data, without being explicitly programmed [2]. ML is widely applied in many cases [12]. The learning algorithm is very dependent on input dataset. The dataset is a source of knowledge that has been grouped and processed according to its purpose. The dataset contains the process of training data and tests dataset. The training dataset is to create a model as a learning process while the test dataset is used by computers to test the model.

By utilizing Deep learning that uses the CNN, it will be easier to train models neural network with layers more than the previous models. A Neural network multilayer has a massive algorithmic

[1] Mohammad Diqi (Corresponding Author: diqi@respati.ac.id, University of Respati Yogyakarta)
[2] Mujastia Feliati Muhdalifah (Author, mujastia@gmail.com, University of Respati Yogyakarta)

complexity, so it requires a high specification computer. Like a computer equipped with a GPU, the neural network will run faster compared to using a CPU. In addition to using the GPU on a computer, you can use a services cloud that provides high-spec GPUs for free as advanced computing up to 12 hours or more. One of them is Google Collaboratory or Collab [4]. Based on the problem, this paper constructs the Design of Java-Based Mobile Literacy Classification Dataset in the Yogyakarta Sonobudoyo State Museum. It is to detect Javanese script patterns so that later it can be used to translate Javanese script.

2. Object and Basis Theory

In this experiment, we symbolize the dataset by a letter. In making the dataset, the machine learning algorithm does the training of the dataset, the test dataset, and the validation of the dataset. There are no fixed rules for choosing and determining a training set, validation set, and test set. Generally, experts will choose these three parts of the dataset randomly from an intact database with a portion of the training set of 50%, a validation set of 25%, and a test set of 25%. If the training set and validation set are grouped in the phase training while the test set in the phase testing set, then the portion of the training phase and the testing phase ranges from 75% / 25% or 80% / 20% [2].

Deep Learning is a part of Machine Learning that consists of many layers (hidden layer) and stack. That layer is method that classifies specific commands from input to produce the output [1]. One of the Deep Learning algorithm is the Convolution Neural Network. It is a network that widely used on images processing. The architecture going through the convolution layer and processed based on the specified filters. Each of these layers produces patterns from several parts of the image to do the classification process.

3. The Method

Convolutional Neural Network (CNN) is one of the developments of artificial neural networks inspired by human neural networks and commonly used in image data to detect and recognize an object in an image [2]. CNN consists of neurons that have weight, bias, and activation functions. CNN process flow in Figure 1 [9].

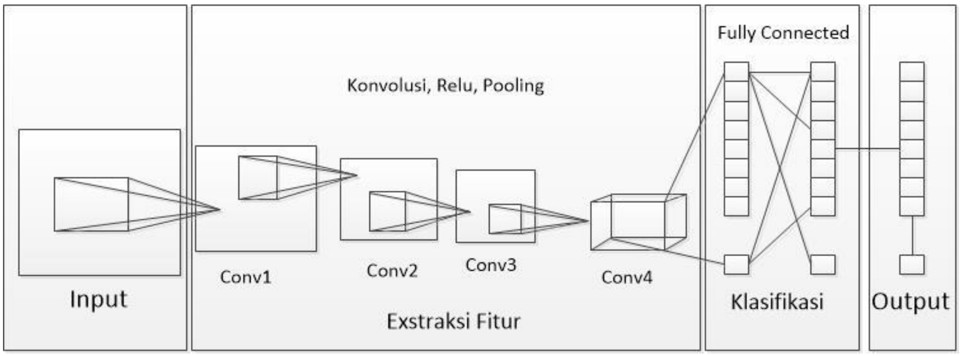

Figure 1 CNN Process Flow

a. Convolution Layer

Convolution Layer is the NN part to performs convolution operations that combines linear filters to the local area. This layer first receives the image at input right on the architecture. The shape of this layer is a filter with a length(pixels), width(pixels), and full accordance with channel image the data in the input right. These three filters will shift to all parts of the picture. The shift will perform a "dot" operation between the input and the value of the filter so that it will produce an output called an activation map or feature map. The convolution process can be seen in Figure 2.

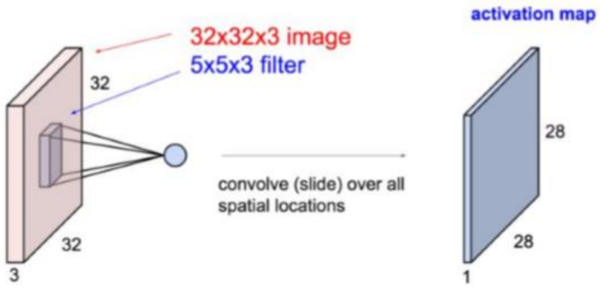

Figure 2 Convolution Process

Figure 2 shows the image size is 32x32x3, the model will convolve the image in the convolution layer, the image cutting process will be carried out with a 5x5x3 filter so that the resulting image with a smaller size is 28x28x1 [9]. Figure 3 shows the convolution formula in a simple form.

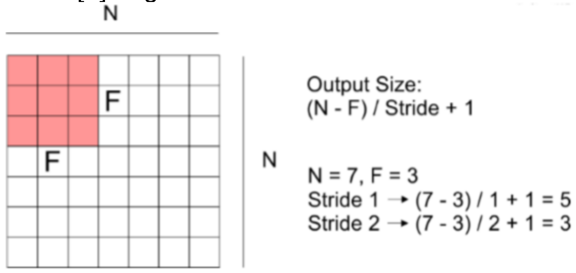

Figure 3 Convolution Formula

Figure 3 shows N is the total number of rows and columns, F is the number of rows and columns to be taken, and stride is the shift made.

b. Pooling Layer

It has many types of pooling in neural network like RunPool pooling as new way to downsize the feature map [14]. The pooling layer receives output from the convolution layer. At this layer, the size of the image data will be reduced. The principle of the pooling layer consists of filters of a specific size and stride or step then shifts to the entire area feature map. For most CNN architectures, the method pooling used is Max pooling. Max pooling divides the convolution layer output into several grids, and then each filter shift will take the most substantial value from each grid. Depending on the step, the method can propose a fraction of its original size to reduce other dimensions of the data, thereby reducing the number of parameters to the next level. Figure 4 depicts the process of the Polling layer.

Figure 4 The Polling Layer

The layer in the image above shows the output convolution layer is divided into several grids and then a 2x2 filter shift with two strides will take the largest value from each grid [10].

c. Fully Connected Layer The

A fully connected layer takes input from the output pooling layer as a feature map. The feature map is a multidimensional array to reshape and produce many n-dimensional vectors. For example, the layer consists of 500 neurons, then will be applied softmax that returns the most massive probability list for each of the 10 class labels as the final classification of the network. Figure 5 shows the processes in the fully connected layer [9].

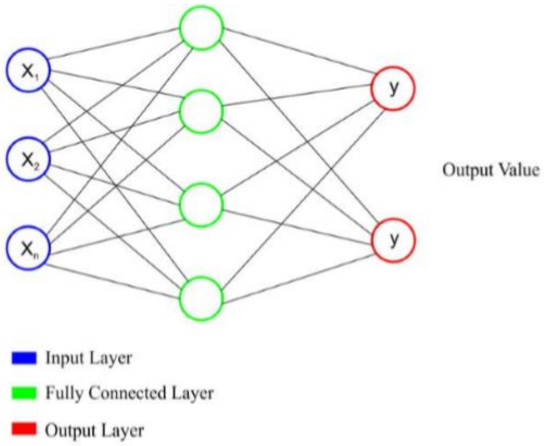

Figure 5 Fully Connected Layer

4. System Design

Figure 6 shows the use case diagram.

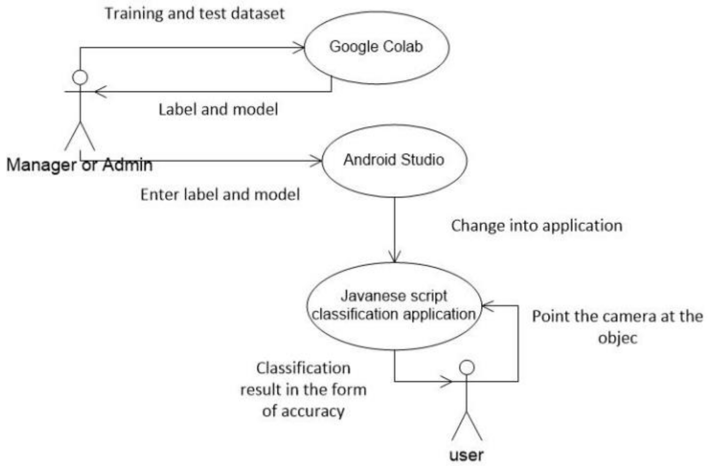

Figure 6 Use Case Diagram

a. Preparing The Dataset

We utilize the dataset as the input picture of Java character script or basic script. The dataset consists of 2000 characters drawn with 20 letters, and each letter has 100 images. 300x300 max pixel size. Examples of images can be seen in Figure 7.

Figure 7 Examples of ways to do

b. Dataset Training

At the training stage, we train the dataset by using the method convolution neural network (CNN). This training process is the stage to obtain high accuracy from the classification conducted. Training the dataset through 2 stages, namely the primary training stage and training to improve the model. Both of these training use batch size 200 with epoch 80. Figure 8 depicts the flow of making the dataset.

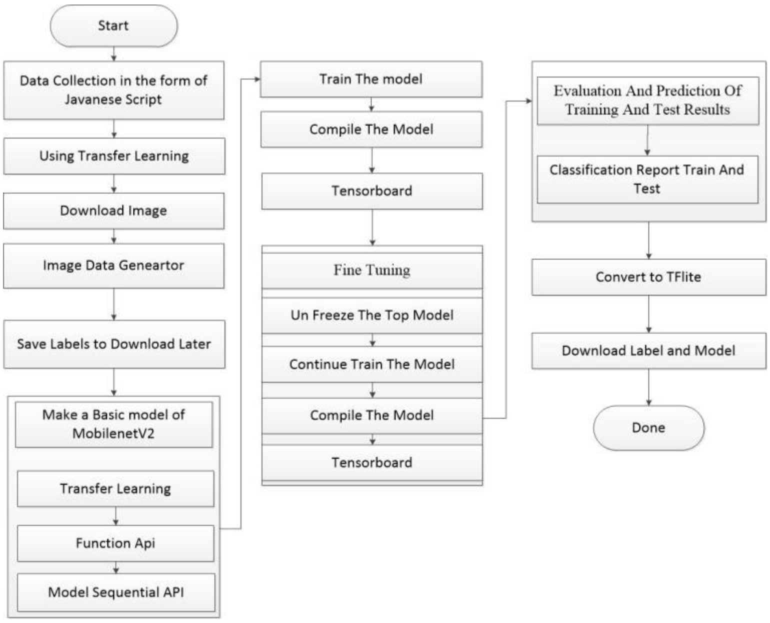

Figure 8 Flowchart or Dataset Training

5. Results and Discussion

The following results and discussion refer to the training process in figure 9.

a) The collection of Javanese script drawings is done by collecting images from different handwriting and also obtained from the internet. Then the image is resized to the same size and compressed (zipped) then uploaded to Mediafire.
b) Using Transfer Learning or importing modules needed through Google Collab.
c) Download images that were previously uploaded.
d) Image Data Generator is used to determine the size of the image scale. Starting from the image size, batch size and share data for the process training and test. The batch size itself is a grouping of data to be sent to a neural network to speed up repetition (epoch).

```
IMAGE_SIZE = 224
BATCH_SIZE = 200

datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    width_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    validation_split=0.2)

train_generator = datagen.flow_from_directory(
    base_dir,
    target_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    subset='training')

val_generator = datagen.flow_from_directory(
    base_dir,
    target_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    subset='validation')
```

Figure 9 Script Image Data Generator

From Figure 9 we can find out research using image size = 224, and batch size = 200. The results of the script can be seen in Figure 10.

```
Found 1600 images belonging to 20 classes.
Found 400 images belonging to 20 classes.
```

Figure 10 Image Data Generator The

Results of Figure 10 show that there are 1600 images in training and there are 400 images tested from 20 classes.

e) At this stage, the system will download labels for the testing process. It obtains the output in the form of classes or letters. The label contains the data letters used in the dataset, based on Figure 11.

```
{'BA': 0, 'CA': 1, 'DA': 2, 'DHA': 3, 'GA': 4,
'HA': 5, 'JA': 6, 'KA': 7, 'LA': 8, 'MA': 9,
'NA': 10, 'NGA': 11, 'NYA': 12, 'PA': 13, 'RA': 14,
'SA': 15, 'TA': 16, 'THA': 17, 'WA': 18, 'YA': 19}
```

Figure 11 Labels

f) Create a basic model of MobileNetV2, the model to optimize training uses two ways, namely with Transfer Learning and Fine Tuning Transfer Learning has two methods to build a model that is the function API and the model sequential API. Model function API is one way to create a model that has more flexibility because we can quickly determine the model where layers are connected to various layers. This model is used to identify a model multi-output, directed graph. While the sequential model API is one way of making a model by allowing the model to be made layer by layer for most problems. There is a limitation of this model when creating models with multiple layers and has many inputs or outputs. Figure 12 depicts the model

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenetv2_1.00_224 (Model) (None, 7, 7, 1280)        2257984
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 5, 5, 32)          368672
_________________________________________________________________
dropout_1 (Dropout)          (None, 5, 5, 32)          0
_________________________________________________________________
global_average_pooling2d_1 ( (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 20)                660
=================================================================
Total params: 2,627,316
Trainable params: 369,332
Non-trainable params: 2,257,984
```

Figure 12. Sequential Model

From Figure 8, the total parameters obtained from the model created are 2,627,316, with the usual settings in training that is 369,332, while those that cannot be trained are 2,257,984. We create a model before the process of training.

g) Train the model is a process of training data by using epoch 50 based on Figure 13.

```
logdir="logs/scalars/"+dt.datetime.now().strftime("%Y%m%d-%H%M%S")tensorboard_callback=
keras.callbacks.TensorBoard(log_dir=logdir)
epochs = 50

history = model.fit_generator(train_generator,
                    epochs=epochs,
                    validation_data=val_generator,
                  callbacks=[tensorboard_callback])
```

Figure 13. The Model Train Script

h) Learning Curves Using Tensorboard
The graph taken refers to the process of the trained model previously. Where epoch is used epoch 49 with a loss of 0.0810 and accuracy of 0.9844 while validation is loss 3.1805 and validation is accuracy 0.2750. Figure 14 depicts the accuracy result in this process.

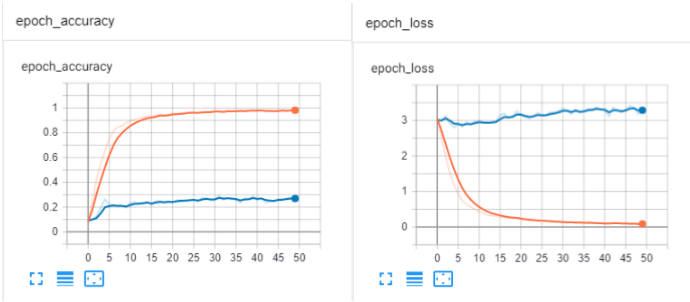

Figure 14 Graph Of Accuracy and Loss

i) We do the Fine Tuning to improve the results of training conducted on transfer learning. We need to take
   1. Steps to Unfreeze Top Model to melt the frozen model at the end of the training.
   2. We compile models to find out how many parameters are trained and which cannot be trained. Figure 15 illustrates the compiling result.

```
Model: "sequential_1"
Layer (type)                 Output Shape              Param #
=================================================================
mobilenetv2_1.00_224 (Model) (None, 7, 7, 1280)        2257984
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 5, 5, 32)          368672
_________________________________________________________________
dropout_1 (Dropout)          (None, 5, 5, 32)          0
_________________________________________________________________
global_average_pooling2d_1 ( (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 20)                660
=================================================================
Total params: 2,627,316
Trainable params: 2,231,924
Non-trainable params: 395,392
```

Figure 15 Results Compile Model

From Figure 13, it can be concluded that the model that has been through the process training with a total parameter of 2,627,316 that cannot be trained is only 395,392 and the usual in training is 2,231,924. The figure shows that the model used is getting better than the model before going through the process training.

   3. Continue Train Model is the second training or improvement process from the training. We conduct the training process by tuning the epoch 50. Figure 16 depicts the results of the train.

```
Epoch 49/50
8/8 [==============================] - 222s 28s/step - loss: 0.0160 - accuracy: 0.9975
                                     - val_loss: 2.4812 - val_accuracy: 0.3775
Epoch 50/50
8/8 [==============================] - 222s 28s/step - loss: 0.0177 - accuracy: 0.9969
                                     - val_loss: 2.5291 - val_accuracy: 0.3525
```

Figure 16 Results Continue Train Model

   4. Learning Curves with Tensorboard
   We obtain the second graph from the second training, which is training refinement. The graph refers to the 49th epoch with a loss 0.0160and an accuracy of 0.9969 while validation of loss 2,5291 and validation of accuracy 0.3525. Figure 17 depicts the result of the process in this stage.

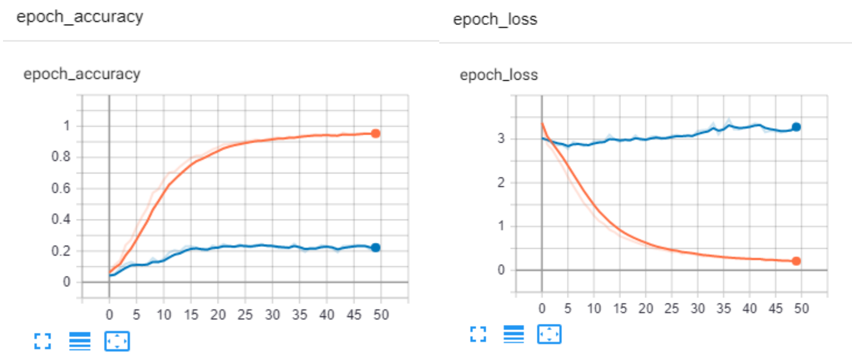

Figure 17 Chart Accuracy and Loss Training Second Stage

j) Evaluation and Prediction of Results Training and Test.
After going through a long process, it will produce a hassle classification of the train and test as a reference to find out the accuracy of each letter.

1. Classification Report Training
We calculate accuracy per class or per letter using the formula [1]

$$\text{accuracy} = \frac{\text{Amount of Data Per class}}{\text{amount of Training Data}} \quad [1]$$

Example of calculation

$$\text{classes} = \frac{317}{80} = 3.9625$$

Figure 18 depicts the calculation accuracy refers to the confusion matrix in train process.

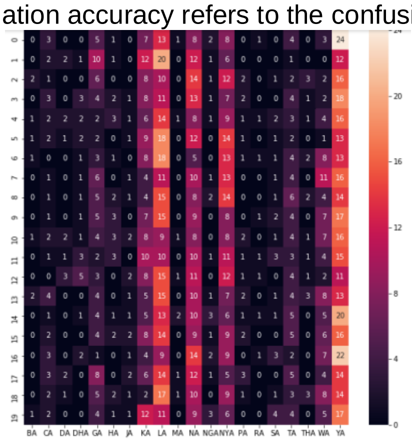

Figure 18 Confusion Matrix Train

2. Classification Report Text
We calculate accuracy per class or letter using the formula [1]

$$\text{accuracy} = \frac{\text{Amount of Data Per class}}{\text{amount of Data Test}} \quad [1]$$

Example of calculation

$$\text{classes} = \frac{95}{20} = 4,75$$

The Calculation accuracy refers to the confusion matrix test can be seen in Figure 19.

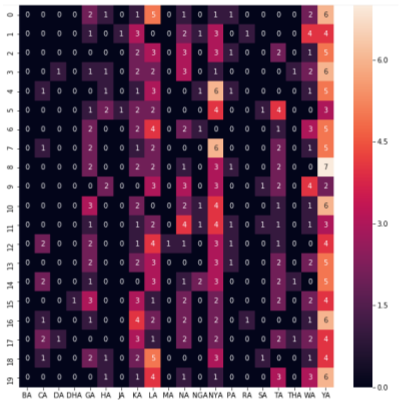

Figure 19 Confusion Matrix Test

k) Convert To TF lite
in this paper, we can proceed to the next stage to save the model using tf_save_mode.save and then convert the model to the TF lite compatible format for Android.

l) Downloading labels and models is the last step in creating a dataset. Downloaded labels and models will be used for manufacturing applications through Android Studio.

m) Implementation
we construct the applications using one interface. The user only exposes the camera to the object (Javanese script). The application will automatically detect the image and categorize the letters according to the picture and the application can display the percentage level of accuracy. Figure 20 depicts an appearance in identifying Javanese characters.

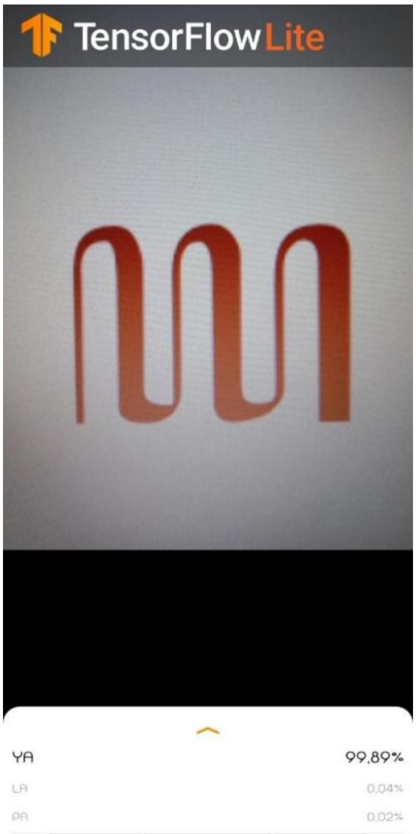

Figure 20 Application Display

Figure 20 shows that the application can detect the letter Yes with an accuracy of 99.89%. It is a promising result because the level of accuracy of the letter Yes is almost 100%.

6. Conclusion

In this paper, we construct the Javanese script classification dataset to help users to detect Javanese characters by using deep learning. It also can be used as a learning media that is more interesting and more effective. We establish the dataset by using the method Convolutional Neural Network (CNN). Stages of making a dataset are input data, the process of building models, and training to recognize Javanese script images. The experiment produces the application of Javanese script classification to provide an accurate level of recognition of Javanese script patterns. This accuracy can be a benchmark of how accurate learning is received to recognize Javanese script patterns in the real application.

7. References

[1] Darma, Royani and G. Ariyanto, "Implementation of Deep Learning Tensorflow Based for Fingerprint Recognition", Emitter Journal: Vol. 18 No. 01 ISSN 1411-8890, Surakarta: Surakarta Muhammadiyah University (UMS).

[2] B, Purnama. "Introduction to Machine Learning Concepts and Practices with Examples of Based Training R and Python", Bandung: Bandung Informatics.

[3] Rena, and P. Navia, "Thesis-Application of Method in Convolutional Neural Network Beam Notation Image Detection", Jakarta: Syarif Hidayatullah State Islamic University. 2019.

[4] Strictly, Get to know Google Colab. [online] https://structilmy.com/2019/05/ know-collab/, accessed December 6, 2019

[5] Sonobudoyo State Museum, [online] www.sonobudoyo.com, accessed November 15, 2019.

[6] L. Marifatul Azizah, S. Fadillah Umayah, and F. Fajar, "Detection of the Mangosteen Surface Surface Using the Method Dept Learning with Multilayer Convolution", Semesta Teknik Vol.21, No.2,230-236, November 2018.

[7] Chairani, Widyawan, and S. Kusumawardani, "Machine Learning for Position Estimation Based on RSS Fingerprint Using IEEE 802.11g in Lantau 3 of JTETI Building UMG ", Journal of Electrical Engineering and Information Technology at Gadjah Mada University, vol.7, No.1 May 2015, e- ISSN: 2460-0997.

[8] A. Santoso and G. Ariyanto, "Implementation of Hard Based Deep Learning for Facial Recognition", Muhammadiyah University of Surakarta (UMS), Emitter Journal, Vol. 18, No. 01, ISSN 1411-8890.

[9] J. Pujoseno, "Implementation of Deep Learning Using Convolutional Neural Networks for Classification of Stationery", Final Project: Indonesian Islamic University, 2018.

[10] I Wayan Suartika EP, A. Yudhi Wijaya, and R. Soelaiman, "Image Classification Using Convolutional Neural Network (CNN) on Caltech 101 ", Sepuluh November Technology Institute (ITS), ITS Engineering Journal, Vol.5, No.1, (2016), ISSN: 2337-3539 (2301-9271 Print).

[11] A. RidnO Prabu Pratma and Z. Jin, Foreign Students' Intention Towards a China's Third-Party Mobile and Onlie Payment Platform Based on Alipay, International Journal of Informatics and Computation: Nanjing University of Information Science and Techonology, Vol.1, No.1, 2019.

[12] P. Wanda, A Survey of Intrusion Detection System. International Journal of Informatics and Computation, [S.l.], vol. 1, no. 1, p. 1-10, Jan. 2020.

[13] S. Lestari, M. Diqi and R. Widyaningrum. "Measurement of maximum value of dental radiograph to predict the bone mineral density." 2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI) (2017): 1-4.

[14] Putra Wanda, Huang J. Jie." RunPool: A Dynamic Pooling Layer for Convolution Neural Network Authors, *, Vol. 13, Issue 1, Pages 66 – 76, 2020.