2nd International Conference on Information Science and Technology Innovation (ICoSTEC)
February, 25 2023. Yogyakarta, Indonesia

Combination Of SqueezeNet And Multilayer Backpropagation Algorithm In Hanacaraka Script Recognition

Yuni Franciska br Tarigan[1], Teddy Surya Gunawan[2], B. Herawan Hayadi[3]

[1,2,3]*Magister of Computer Science, Potensi Utama University*
JL. KL. Yos Sudarso Km. 6,5 No. 3-A, Medan

[1]yuni.franciska@gmail.com

[2]tsgunawan@gmail.com

[3]b.herawan.hayadi@gmail.com

***Abstract*—Javanese script is one of Indonesia's cultural heritages that are increasingly rarely used today. The difficulty of recognizing the shapes of letters, let alone writing them, is the main obstacle in using the Hanacaraka script. This research offers an alternative to Hanacaraka script recognition using a combination of image feature extraction and machine learning, where we utilize a pre-trained SquzeeNet model and Multilayer Backpropagation algorithm. Of the 18 models built using ReLu, Sigmoid, and Tanh activation functions, we found that the Tanh activation function, using the combination of 50-50-100 neuron configuration and 25 epochs, was the most optimal function used to classify the training data with accuracy, precision, and recall values of 93.8%. Meanwhile, the Tanh activation function, using the 50-100-50 neuron configuration and 50 epochs, is the most optimal function to classify the testing data, with accuracy, precision, and recall values of 89.1%, 89.5%, and 89.5%. All built models show a training and testing performance ratio below 10%. From this result, we conclude that all models have good reliability in the training and testing classification process.**

***Keywords*—classification, hanacaraka script, multilayer backpropagation, squeezenet, transfer learning**

I. INTRODUCTION

A group of scripts known as Hanacaraka (carakan) was developed from the first five series of letters in the Javanese alphabet and is used extensively on the islands of Java and Bali [1]. The Hanacaraka script is in danger of becoming extinct due to its lack of use in modern times, particularly in visual communication media [2]. The difficulty of differentiating and writing the letters in the Hanacaraka script is the prime reason for its lack of usage by the Indonesian people [3]. In this study, we identify the Hanacaraka script using a machine learning approach by using digital images as an alternative way to recognize and re-introduce it to the public.

Some studies have proven the performance of machine learning in digital image-based object recognition: the cat breed classification using the Support Vector Machine (SVM) and Naive Bayes (NB) algorithms, with accuracy values of 88.4% and 79.5% [4]; the breast cancer classification using the Multilayer Perceptron (MLP) and SVM algorithm, with the highest accuracy value of 97.7% [5]; signature identification using the K-Nearest Neighbors (K-NN) algorithm, with 96.7% accuracy [6].

Feature extraction is a crucial step in digital image-based object recognition since the extracted features serve as the data for machine learning algorithms to recognize the digital image's object. In this research, we use a pre-trained model-based feature extraction to extract features in the image of Hanacaraka letters to be used as a dataset in the classification process using machine learning algorithms. A pre-trained model is a model that has been trained in advance using a large amount of data so it can classify using fewer data.

SqueezeNet is a reliable pre-trained Convolutional Neural Network (CNN) model in training digital image-based object recognition, with the same accuracy as the AlexNet model [7]. Several transfer learning-based studies used this model in their research, such as in the Fundus classification with a combination of the Gradient Descent algorithm [8], the combination with Support Vector Machine (SVM) and Naive Bayes algorithms in the face mask classification [9], the maize leaf disease classification with the combination of Stochastic Gradient Descent (SGD) algorithm [10], the vegetable and fruit image classification in pair with the Linear Discriminant Analysis (LDA) [11], and a combination with MLP for COVID-19 diagnosis [12]. Based on the compatibility shown by this model with various types of machine learning algorithms, we choose SqueezeNet as a model for feature extraction of Hanacaraka script images and then transfer the results the Backpropagation algorithm as classification datasets.

Backpropagation is a machine learning algorithm with a back-passing ability to correct errors in the classification process using its three layers: the input layers, the hidden layers, and the output layers [13]. Backpropagation is a widely utilized technique in classification research, as demonstrated in the study about signature image identification with 83% accuracy [14], the white blood and lymphoblast cells classification with 91.43% accuracy [15], and the eyeball movement patterns classification with 88.24% accuracy [16]. Backpropagation algorithms that use three or more hidden layers are known as Multilayer Backpropagation, and neural networks, such as the

Multilayer Perceptron, use it in their architecture [17]. Multilayer Backpropagation has an advantage in terms of a repeated weight modification process in each hidden layer, resulting in a more accurate weight calculation [18]. We choose this approach as a combination with the SqueezeNet model.

The combination of SquezeeNet and MB in this study produces 18 models, where each model uses different activation functions, namely ReLu, Tanh, and Sigmoid. We also aim to analyze the best epoch and neuron configuration using the combination of 25 and 50 epochs and 50-50-100, 50-100-50, and 100-50-50 neuron configurations. We then analyze the performance of these 18 models using the 10-fold cross-validation to produce accuracy, precision, and recall values. Based on these three values, we choose the best and worst models in classifying the Hanacaraka script based on the dataset generated from the feature extraction process using the SquezeeNet model.

II. METHODS

A. *Dataset*

In this research, we use data in the form of digital images of handwritten Hanacaraka letters obtained from the GitHub site [19]. Figure 1 shows the sample images from each of the Hanacaraka script letters.

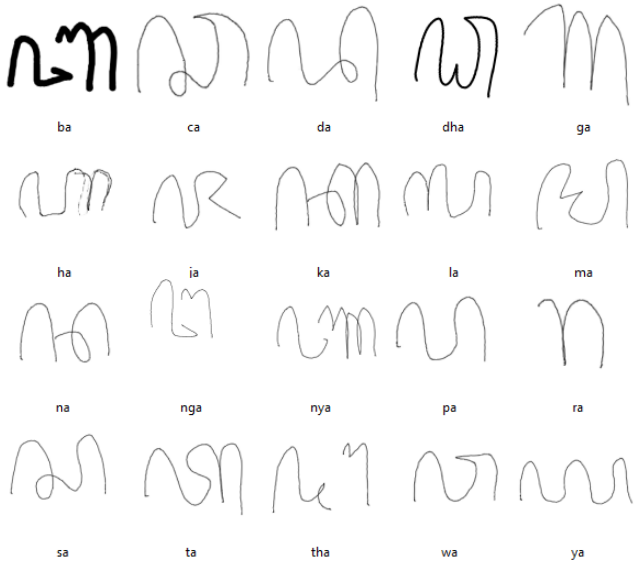

Fig. 1 Hancaraka Letters Sample

Figure 1 shows an example of the 20 letters in the Hanacaraka script used in this study. Each letter has 160 pictures, which we split in half by an 80:20 ratio to provide 128 photos for training data and 32 images for testing data.

Previous research has done some classification using this Hanacaraka script, such as applying a Convolutional Neural Network (CNN), with the highest accuracy generated value of 86.68% [20], using the Backpropagation algorithm with the best accuracy value of 77% [21], and implementing the Backpropagation algorithm, with an accuracy up to 90% [22].

We use the images in the dataset to train and test Hanacaraka script recognition, using a combination of SqueezeNet and Backpropagation algorithms to see if we can get better results.

B. *SquezeeNet*

SqueezeNet is a CNN architecture whose dimensionality reduction method reduces the size of the activation map from 3x3 to 1x1 in the convolution layer [23]. This pre-trained CNN model can produce a base network output for class prediction at the final layer level [24]. The SqueezeNet architecture, shown in Figure 2, consists of fire basic module stacks, the hybrid kernel 1x1 and 3x3 in the expansion layer, and a 1x1 convolution kernel, with the feature map output the size of the original image [25].

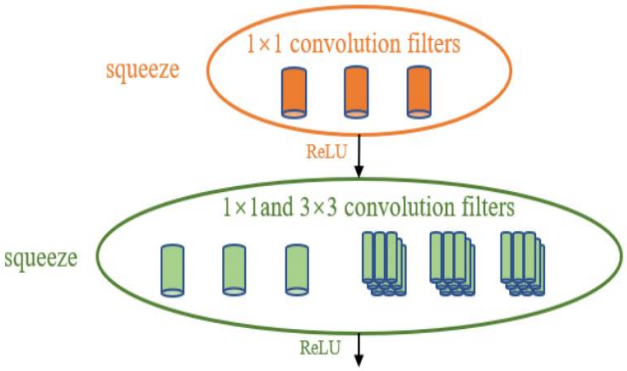

Fig. 2 SqueezeNet Architecture

Some research has employed SqueezeNet as a feature extractor in image classification, as seen in Parkinson's disease detection, with an accuracy of 90% [26], in production plants' surface defect detection, with an average MAE of 0.017533 [27], and in hand and gesture detection, with a precision of 84.1% [28].

In this research, we employ the SqueezeNet architecture in extracting the Hanacaraka script images to produce 1000 features as the dataset. Table I shows the extracted feature samples from the training data, while Table II is from the testing data.

TABLE I
TRAINING DATA SAMPLES

n0 - n994	n995	n996	n997	n998	n999	Script
…	7.789	8.428	2.759	2.77	11.785	ba
…	0.622	3.43	1.238	3.479	5.649	ca
…	0.844	2.192	0.363	4.169	6.8004	da
…	4.074	5.728	0.271	5.044	8.307	dha
…	-0.209	1.193	0.865	3.817	1.4674	ga
…	1.89	2.151	-0.981	4.925	5.576	ha
…	1.334	1.981	-0.23	4.594	6.111	ja
…	1.093	3.042	-0.327	5.585	4.112	ka
…	1.619	3.285	-0.598	5.219	6.237	la
…	-0.03	2.638	0.105	3.205	7.382	ma
…	1.324	1.231	-0.412	3.355	4.908	na
…	0.837	1.228	-0.418	2.855	6.675	nga
…	1.129	3.798	-0.733	4.422	6.458	nya

…	0.798	2.661	0.058	3.891	5.552	pa
…	1.439	2.145	-1.28	3.469	3.889	ra
…	0.052	3.133	-0.049	3.931	5.775	sa
…	-0.097	3.588	0.546	4.107	3.189	ta
…	1.033	2.568	-0.372	2.519	6.0722	tha
…	0.145	2.835	0.732	3.236	5.871	wa
…	0.927	4.159	0.2111	3.785	8.115	ya

From Table I above, the feature extraction with the SqueezeNet produces n0, n1, ..., and n999 attributes used in the training data with the combination of the script target. In Table II, the feature extraction with the SqueezeNet also produces n0, n1, ..., and n999 attributes used in the testing data with the combination of the script target.

TABLE II
TESTING DATA SAMPLES

n0 - n994	**n995**	**n996**	**n997**	**n998**	**n999**	**Script**
…	3.095	5.299	-1.464	3.904	5.688	ba
…	1.195	4.792	-0.410	3.521	6.581	ca
…	3.019	5.603	-1.401	4.771	8.099	da
…	1.834	6.815	0.346	4.616	6.571	dha
…	1.212	1.790	-0.007	5.463	5.357	ga
…	3.506	6.638	-1.247	5.249	4.778	ha
…	0.641	0.259	-0.059	5.932	4.567	ja
…	2.926	3.849	-0.854	6.119	2.872	ka
…	0.749	1.058	0.203	5.087	3.680	la
…	4.731	11.898	1.748	-0.478	15.042	ma
…	4.364	4.722	-1.870	4.669	6.802	na
…	0.785	0.327	-1.363	1.564	6.056	nga
…	4.239	7.427	-0.300	4.969	4.254	nya
…	1.877	1.049	-0.544	3.745	6.410	pa
…	4.946	2.053	-2.063	4.439	5.940	ra
…	2.832	4.664	-1.318	5.157	5.569	sa
…	1.341	1.027	0.271	4.054	4.610	ta
…	0.692	2.094	-1.598	3.325	5.282	tha
…	2.405	4.585	-0.923	3.980	6.934	wa
…	0.352	2.896	0.321	4.615	4.095	ya

C. Model Configuration

Finding an optimal activation function for a specific model in a neural network is a core task to improve the model's performance [29]. In multilayer neural networks, activation functions, such as ReLu, Sigmoid, and Tanh, play a crucial role in controlling the output response of the neurons [30].

In this research, we implement three activation functions in the models, ReLu, Sigmoid, and Tanh, in limiting the neuron output by using equations (1) to (3) [31].

$$f(x)_{ReLu} = max(0,x) \begin{cases} x, if\ x > 0 \\ 0, if\ x \leq 0 \end{cases} \quad (1)$$

$$f(x)_{sigmoid} = \frac{1}{e^{-x}} \quad (2)$$

$$f(x)_{Tanh} = \frac{2}{1+e^{-2x}} - 1 \quad (3)$$

We employ three hidden layers with a combination of 50-50-100, 50-100-50, and 100-50-50 neuron numbers for all models. While the MB-R model uses the ReLu activation function, the MB-S, and the MB-T use the Sigmoid and Tanh activation function, respectively. Table IV shows the model's configuration used in this research.

TABLE IV
MODEL'S CONFIGURATION

Model	Configuration
MB-R1	Activation Function: ReLu Neuron: 50-50-100 Epoch: 25
MB-R2	Activation Function: ReLu Neuron: 50-50-100 Epoch: 50
MB-R3	Activation Function: ReLu Neuron: 50-100-50 Epoch: 25
MB-R4	Activation Function: ReLu Neuron: 50-100-100 Epoch: 50
MB-R5	Activation Function: ReLu Neuron: 100-50-50 Epoch: 25
MB-R6	Activation Function: ReLu Neuron: 100-50-50 Epoch: 50
MB-S1	Activation Function: Sigmoid Neuron: 50-50-100 Epoch: 25
MB-S2	Activation Function: Sigmoid Neuron: 50-50-100 Epoch: 50
MB-S3	Activation Function: Sigmoid Neuron: 50-100-50 Epoch: 25
MB-S4	Activation Function: Sigmoid Neuron: 50-100-100 Epoch: 50
MB-S5	Activation Function: Sigmoid Neuron: 100-50-50 Epoch: 25
MB-S6	Activation Function: Sigmoid Neuron: 100-50-50 Epoch: 50
MB-T1	Activation Function: Tanh Neuron: 50-50-100 Epoch: 25
MB-T2	Activation Function: Tanh Neuron: 50-50-100 Epoch: 50
MB-T3	Activation Function: Tanh Neuron: 50-100-50 Epoch: 25
MB-T4	Activation Function: Tanh Neuron: 50-100-50 Epoch: 50
MB-T5	Activation Function: Tanh Neuron: 100-50-50 Epoch: 25
MB-T6	Activation Function: Tanh Neuron: 100-50-50 Epoch: 50

D. Evaluation

We use 10-fold cross-validation to evaluate each model's performance in classifying the Hancaraka script. The accuracy, precision, and recall values gained using equations (4) to (6) will be the reference in ranking the models from best to worst [32].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

We will use the accuracy, precision, and recall values to analyze the activation function's performance and the neuron placement in the training and testing process.

By comparing the training and testing data performance, we calculate the difference percentage to analyze the reliability of the models. We use a standard of 10% as a threshold for the comparison. If the difference percentage is lower than 10%, we conclude that the model produced is reliable enough to classify the given data.

III. RESULTS

Using the extracted feature from the Hanacaraka script images, we classify the dataset using the built model, which produced a performance result from each model, displayed in Table IV for the training data and Table V for the testing data.

TABLE IV
PERFORMANCE RESULT FOR TRAINING DATA

Model	Accuracy	Precision	Recall
MB-R1	0.909	0.909	0.909
MB-R2	0.919	0.919	0.919
MB-R3	0.918	0.919	0.918
MB-R4	0.921	0.921	0.921
MB-R5	0.912	0.913	0.912
MB-R6	0.920	0.920	0.920
MB-S1	0.521	0.489	0.521
MB-S2	0.740	0.740	0.740
MB-S3	0.590	0.559	0.590
MB-S4	0.745	0.749	0.745
MB-S5	0.747	0.748	0.747
MB-S6	0.838	0.834	0.838
MB-T1	0.927	0.928	0.927
MB-T2	0.938	0.938	0.938
MB-T3	0.927	0.927	0.927
MB-T4	0.936	0.936	0.936
MB-T5	0.926	0.927	0.926
MB-T6	0.934	0.934	0.934

From the result shown in Table IV, we get the best performance from the MB-T2 model, with an accuracy value of 93.8%, a precision value of 93.8%, and a recall value of 93.8%. This result shows that in the classification of the training data, the Tanh activation function works best in the 50-50-100 neurons configuration and with 25 epochs.

TABLE IV
PERFORMANCE RESULT FOR TESTING DATA

Model	Accuracy	Precision	Recall
MB-R1	0.831	0.857	0.831
MB-R2	0.842	0.865	0.842
MB-R3	0.872	0.881	0.872
MB-R4	0.881	0.888	0.881
MB-R5	0.825	0.84	0.827
MB-R6	0.827	0.841	0.827
MB-S1	0.491	0.423	0.491
MB-S2	0.711	0.716	0.711
MB-S3	0.623	0.606	0.623
MB-S4	0.72	0.742	0.72
MB-S5	0.68	0.686	0.68
MB-S6	0.781	0.791	0.781
MB-T1	0.88	0.89	0.88
MB-T2	0.884	0.894	0.884
MB-T3	0.872	0.879	0.872
MB-T4	0.891	0.895	0.891
MB-T5	0.872	0.876	0.872
MB-T6	0.875	0.88	0.875

From the result shown in Table V, we get the best performance from the MB-T4 model, with an accuracy value of 89.1%, a precision value of 89.5%, and a recall value of 89.1%. This result shows that in the classification of the testing data, the Tanh activation function works best in the 50-100-50 neurons configuration and with 50 epochs.

Next, we calculate the model's average performance for training data classification using the epoch's variation, resulting in data displayed in Table VII and Table VIII.

TABLE VII
AVERAGE TRAINING PERFORMANCE PER EPOCH

Activation Function	Epoch	Accuracy	Precision	Recall
ReLu	25	0.913	0.914	0.913
	50	0.920	0.920	0.920
Sigmoid	25	0.619	0.599	0.619
	50	0.774	0.774	0.774
Tanh	25	0.927	0.927	0.927
	50	0.936	0.936	0.936

From the result shown in Table VII, we get the best performance for the ReLu activation function yields using the 50 epoch, with an accuracy value of 92%, a precision value of 92%, and a recall value of 92%. This result shows that the ReLu activation function works best with 50 epochs in the training process.

From the result shown in Table VII, we get the best performance for the Sigmoid activation function yields using the 50 epoch, with an accuracy value of 77.4%, a precision value of 77.4%, and a recall value of 77.4%. This result shows that the Sigmoid activation function works best with 50 epochs in the training process.

From the result shown in Table VII, we get the best performance for the Tanh activation function yields using the 50 epoch, with an accuracy value of 93.6%, a precision value of 93.6%, and a recall value of 93.6%. This result shows that

the Tanh activation function works best with 50 epochs in the training process.

From the above results, we found that the best epoch to use in the Hanacaraka script classification using the training data is 50. We also see that as the epoch value increases, the classification performance of the training data of each activation function improves.

Next, we calculate the model's average performance for testing data classification using the epoch's variation, resulting in data displayed in Table VIII.

TABLE VIII
AVERAGE TESTING PERFORMANCE PER EPOCH

Activation Function	Epoch	Accuracy	Precision	Recall
ReLu	25	0.843	0.859	0.843
	50	0.850	0.865	0.850
Sigmoid	25	0.598	0.572	0.598
	50	0.737	0.750	0.737
Tanh	25	0.875	0.882	0.875
	50	0.883	0.890	0.883

From the result shown in Table VIII, we get the best performance for the ReLu activation function yields using the 50 epoch, with an accuracy value of 85%, a precision value of 86.5%, and a recall value of 85%. This result shows that the ReLu activation function works best with 50 epochs in the testing process.

From the result shown in Table VIII, we get the best performance for the Sigmoid activation function yields using the 50 epoch, with an accuracy value of 73.7%, a precision value of 75%, and a recall value of 73.7%. This result shows that the Sigmoid activation function works best with 50 epochs in the testing process.

From the result shown in Table VIII, we get the best performance for the Tanh activation function yields using the 50 epoch, with an accuracy value of 88.73 a precision value of 89%, and a recall value of 88.3%. This result shows that the Tanh activation function works best with 50 epochs in the testing process.

From the above results, we found that the best epoch to use in the Hanacaraka script classification using the testing data is 50. We also see that as the epoch value increases, the classification performance of the testing data of each activation function improves.

Next, we calculate the model's average performance for testing data classification using the neuron's configuration, resulting in data displayed in Table IX.

TABLE IX
AVERAGE TRAINING PERFORMANCE PER NEURONS

Activation Function	Neurons	Accuracy	Precision	Recall
ReLu	50-50-100	0.914	0.914	0.914
	50-100-50	0.920	0.920	0.920
	100-50-50	0.916	0.917	0.916
Sigmoid	50-50-100	0.631	0.615	0.631
	50-100-50	0.668	0.654	0.668
	100-50-50	0.793	0.791	0.793
Tanh	50-50-100	0.933	0.933	0.933
	50-100-50	0.932	0.932	0.932
	100-50-50	0.930	0.931	0.930

From the result shown in Table IX, we get the best performance for the ReLu activation function yields using the 50-100-50 neuron's configuration, with an accuracy value of 92%, a precision value of 92%, and a recall value of 92%. This result shows that the ReLu activation function works best with the 50-100-50 neuron's configuration in the training process.

From the result shown in Table VII, we get the best performance for the Sigmoid activation function yields using the 100-50-50 neuron's configuration, with an accuracy value of 79.3%, a precision value of 79.1%, and a recall value of 79.3%. This result shows that the Sigmoid activation function works best with 100-50-50 neuron's configuration in the training process.

From the result shown in Table IX, we get the best performance for the Tanh activation function yields using the 50-50-100 neuron's configuration, with an accuracy value of 93.3%, a precision value of 93.3%, and a recall value of 93.3%. This result shows that the Tanh activation function works best with 50-50-100 neuron's configuration in the training process.

Using the training data, we discovered that each activation function performs better with distinct neuron configurations in the Hanacaraka script classification.

Next, we calculate the model's average performance for testing data classification using the neuron's configuration, resulting in data displayed in Table X.

TABLE X
AVERAGE TESTING PERFORMANCE PER NEURONS

Activation Function	Neurons	Accuracy	Precision	Recall
ReLu	50-50-100	0.837	0.861	0.837
	50-100-50	0.877	0.885	0.877
	100-50-50	0.826	0.841	0.827
Sigmoid	50-50-100	0.601	0.570	0.601
	50-100-50	0.672	0.674	0.672
	100-50-50	0.731	0.739	0.731
Tanh	50-50-100	0.882	0.892	0.882
	50-100-50	0.882	0.887	0.882
	100-50-50	0.874	0.878	0.874

From the result shown in Table X, we get the best performance for the ReLu activation function yields using the 50-100-50 neuron's configuration, with an accuracy value of 87.7%, a precision value of 88.5%, and a recall value of 87.7%. This result shows that the ReLu activation function works best with 50-100-50 neuron's configuration in the testing process.

From the result shown in Table X, we get the best performance for the Sigmoid activation function yields using the 100-50-50 neuron's configuration, with an accuracy value of 73.1%, a precision value of 73.9%, and a recall value of 73.1%. This result shows that the Sigmoid activation function

works best with 100-50-50 neuron's configuration in the testing process.

From the result shown in Table X, we get the best performance for the Tanh activation function yields using the 50-50-100 neuron's configuration, with an accuracy value of 88.2%, a precision value of 89.2%, and a recall value of 88.2%. This result shows that the Tanh activation function works best with 50-50-100 neuron's configuration in the testing process.

Using the testing data, we discovered that each activation function performs better with distinct neuron configurations in the Hanacaraka script classification..

Finally, we calculate the difference percentage for training and testing data classification result on each model. Table XI shows the comparison result with the final reliability evaluation.

TABLE XI
PERFORMANCE REABILITY

Model	Δ Accuracy	Δ Precision	Δ Recall	Reliable
MB-R1	0.078	0.052	0.078	Yes
MB-R2	0.077	0.054	0.077	Yes
MB-R3	0.046	0.038	0.046	Yes
MB-R4	0.04	0.033	0.04	Yes
MB-R5	0.087	0.073	0.085	Yes
MB-R6	0.093	0.079	0.093	Yes
MB-S1	0.03	0.066	0.03	Yes
MB-S2	0.029	0.024	0.029	Yes
MB-S3	0.033	0.047	0.033	Yes
MB-S4	0.025	0.007	0.025	Yes
MB-S5	0.067	0.062	0.067	Yes
MB-S6	0.057	0.043	0.057	Yes
MB-T1	0.047	0.038	0.047	Yes
MB-T2	0.054	0.044	0.054	Yes
MB-T3	0.055	0.048	0.055	Yes
MB-T4	0.045	0.041	0.045	Yes
MB-T5	0.054	0.051	0.054	Yes
MB-T6	0.059	0.054	0.059	Yes

Table XI shows that all models have less than a 10% difference percentage, showing that all the models built have enough reliability to classify the Hanacaraka script using the given image data and are feasible to be implemented in a finished application.

IV. CONCLUSIONS

From this research, we acknowledge that using the pre-trained SqueezeNet model helps make feature extraction easier. The resulting features, numbering 1000, can be transferred to other machine learning algorithms, in this case, the Multilayer Backpropagation, as the data for the Hanacaraka script classification. From the training data classification, we found that the Tanh activation function with 25 epochs and 50-50-100 neuron configuration is the most optimal for this case, proven by the highest accuracy (93.8%), precision (93.8%), and recall (93.8%) values generated from this model. From the testing data classification, we found that the Tanh activation function with 50 epochs and 50-100-50 neuron configuration is the most optimal for this case, proven by the highest accuracy (89.1%), precision (89.5%), and recall (89.1%) values generated from this model.On average, the ReLu activation function works best with 50 epochs in the training process, generating an accuracy, precision, and recall values of 92%, respectively. On average, the Sigmoid activation function works best with 50 epochs in the training process, generating an accuracy, precision, and recall values of 77.4%, respectively. On average, the Tanh activation function works best with 50 epochs in the training process, generating an accuracy, precision, and recall values of 93.6%, respectively. Based on the testing data, the ReLu activation function performs better using 50 epochs, generating an accuracy, precision, and recall values of 85%, 86.5%, and 85%. The Sigmoid activation function performs better using 50 epochs in the testing data classification, generating an accuracy, precision, and recall values of 73.7%, 75%, and 73.7%. Meanwhile, the Tanh activation function performs better using 50 epochs in the testing data classification, generating an accuracy, precision, and recall values of 88.3%, 89%, and 83.3%. Based on the neuron configuration's average performance, the ReLu activation function yields the best performance in the training data classification using the 50-100-50 neuron configuration, with an accuracy, precision, and recall values of 92%, respectively. The Sigmoid activation function yields the best performance using the 100-50-50 neuron configuration, with an accuracy, precision, and recall values of 79.3%, 79.1%, and 79.3%. Meanwhile, the Tanh activation fuction generates the highest performance value of 93.3% accuracy, precision, and recall, using the 50-50-100 neuron configuration. Overall, all models have excellent reliability, as shown by the less than 10% difference percentage between training and testing classification performance. We hope that the findings of this study will prompt more research into promoting Indonesian cultural heritage in the future to prevent its extinction.

REFERENCES

[1] A. S. Priyanto, "Javanese Script Ca, Ra, Ka as a Source of Idea for the Creation of Ceramic Works," *Karya Seni*, p. 15, 2019.

[2] D. P. Gumiwang, I. P. Windasari, and R. Septiana, "Educational Game for Learning Javanese Script Using Android-Based Augmented Reality Technology at SD Negeri Sinomwidodo Tambakromo Pati," *J. Ilmu Tek. dan Komput.*, vol. 6, no. 2, pp. 163–173, 2022.

[3] I. I. Widihastuti and M. Khosyiâ, "Interactive Application for Learning Javanese Script," *Maj. Ilm. Sultan Agung*, vol. 50, no. 127, pp. 45–52, 2021.

[4] J. Kusuma, A. Jinan, M. Z. Lubis, and R. Rosnelly, "Comparison of Support Vector Machine and Naive Bayes Algorithms on Cat Breed Classification," *J. Generic*, vol. 14, no. 1, pp. 8–12, 2022.

[5] J. Kusuma, B. H. Hayadi, and R. Rosnelly, "Comparison of Multi-Layer Perceptron (MLP) and Support Vector Machine (SVM) Methods for Breast Cancer Classification," *MIND (Multimedia Artif. Intell. Netw. Database) J.*, vol. 7, no. 1, pp. 51–60, 2022.

[6] M. S. Simanjuntak, R. Rosnelly, and Wanayumini, "Identifikasi Tanda Tangan menggunakan Metode Fitur Ekstrasi Biner dan K Nearest Neighbor," *CSRID (Computer Sci. Res. Its Dev. Journal)*, vol. 12, no. 3, p. 191, 2021.

[7] S. Kumar, S. Mishra, and S. K. Singh, "Deep Transfer Learning-Based COVID-19 Prediction Using Chest X-Rays," *J. Health Manag.*, vol. 23, no. 4, pp. 730–746, 2021.

[8] W. Setiawan, "Comparison of Convolutional Neural Network Architectures for Fundus Classification," *J. Simantec*, vol. 7, no. 2, pp. 48–53, 2020.
[9] P. R. Togatorop and A. Fauzi, "Classification of Face Mask Usage Using Squeezenet," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 9, no. 1, pp. 397–406, Mar. 2022.
[10] W. Setiawan, A. Ghofur, F. Hastarita Rachman, and R. Rulaningtyas, "Deep Convolutional Neural Network AlexNet and Squeezenet for Maize Leaf Diseases Image Classification," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, pp. 0–7, Nov. 2021.
[11] M. BAYĞIN, "Vegetable and Fruit Image Classification with SqueezeNet based Deep Feature Generator," *Turkish J. Sci. Technol.*, vol. 17, no. 1, pp. 121–134, 2022.
[12] I. Zabbah, K. Layeghi, and R. Ebrahimpour, "Improving the Diagnosis of COVID-19 by using a combination of Deep Learning Models," *J. Electr. Comput. Eng. Innov.*, vol. 10, no. 2, pp. 411–424, 2022.
[13] D. Pardede, B. H. Hayadi, and Iskandar, "Multi-Layer Perceptron Literature Review How Well This Algorithm Performs," *J. ICT Apl. Syst.*, vol. 1, no. 1, pp. 23–35, Jun. 2022.
[14] J. Kurniati, "Optimization of Artificial Neural Networks Using Ant Colony Optimization To Identify Signature Images," *Transmisi*, vol. 21, no. 4, pp. 128–134, 2019.
[15] A. N. Liyantoko, I. Candradewi, and A. Harjoko, "Classification of White Blood Cells and Lymphoblast Cells Using Multilayer Perceptron Backpropagation Method," *IJEIS (Indonesian J. Electron. Instrum. Syst.*, vol. 9, no. 2, p. 173, 2019.
[16] K. Amadea, F. A. Bachtiar, and G. Pangestu, "Classification of Eyeball Movement Patterns Using the Multilayer Backpropagation Method," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 391–400, 2022.
[17] M. Handayani, M. Riandini, and Z. Zakarias, "Comparison of Neural Network Optimization Functions in Candidate Husband Eligibility Classification," *J. Inform.*, vol. 9, no. 1, pp. 78–84, Apr. 2022.
[18] D. Pardede, I. Firmansyah, M. Handayani, M. Riandini, and R. Rosnelly, "Comparison Of Multilayer Perceptron's Activation And Optimization Functions In Classification Of Covid-19 Patients," *JURTEKSI (Jurnal Teknol. dan Sist. Informasi)*, vol. 8, no. 3, pp. 271–278, Aug. 2022.
[19] R. P. Nugroho, "Aksara Jawa - Hanacaraka," 2020. [Online]. Available: https://github.com/vzrenggamani/aksarajawa-hanacaraka.
[20] Y. Harjoseputro, "A Classification Javanese Letters Model using a Convolutional Neural Network with KERAS Framework," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 10, pp. 106–111, 2020.
[21] A. N. Handayani, H. W. Herwanto, K. L. Chandrika, and K. Arai, "Recognition of Handwritten Javanese Script using Backpropagation with Zoning Feature Extraction," *Knowl. Eng. Data Sci.*, vol. 4, no. 2, p. 117, 2021.
[22] I. Prihandi, I. Ranggadara, S. Dwiasnati, Y. S. Sari, and Suhendra, "Implementation of Backpropagation Method for Identified Javanese Scripts," *J. Phys. Conf. Ser.*, vol. 1477, no. 3, 2020.
[23] F. Angga Irawan, M. Sudarma, and D. Care Khrisne, "Design of Android-based California Papaya Plant Disease Identification Application Using CNN Method Squeezenet Architecture Model," *Spektrum*, vol. 8, no. 2, pp. 18–26, 2021.
[24] D. Arifianto and A. S. Agoes, "Cervical Cancer Image Classification Using CNN Transfer Learning," *Proc. 2nd Int. Semin. Sci. Appl. Technol. (ISSAT 2021)*, vol. 207, no. Issat, pp. 145–149, 2021.
[25] D. Yang, W. Bao, and K. Zheng, "Lane Detection of Smart Car based on Deep Learning," *J. Phys. Conf. Ser.*, vol. 1873, no. 1, 2021.
[26] L. S. Bernardo, R. Damaševičius, S. H. Ling, V. H. C. de Albuquerque, and J. M. R. S. Tavares, "Modified SqueezeNet Architecture for Parkinson's Disease Detection Based on Keypress Data," *Biomedicines*, vol. 10, no. 11, p. 2746, Oct. 2022.
[27] G. Fu *et al.*, "A Surface Defect Inspection Model via Rich Feature Extraction and Residual-Based Progressive Integration CNN," *Machines*, vol. 11, no. 1, pp. 1–19, 2023.
[28] B. Qiang *et al.*, "SqueezeNet and Fusion Network-Based Accurate Fast Fully Convolutional Network for Hand Detection and Gesture Recognition," *IEEE Access*, vol. 9, pp. 77661–77674, 2021.
[29] B. Yuen, M. T. Hoang, X. Dong, and T. Lu, "Universal activation function for machine learning," *Sci. Rep.*, vol. 11, no. 1, pp. 1–12, 2021.
[30] Q. Jiang, L. Zhu, C. Shu, and V. Sekar, "Multilayer perceptron neural network activated by adaptive Gaussian radial basis function and its application to predict lid-driven cavity flow," *Acta Mech. Sin. Xuebao*, vol. 37, no. 12, pp. 1757–1772, 2021.
[31] A. Nguyen, K. Pham, D. Ngo, T. Ngo, and L. Pham, "An analysis of state-of-the-art activation functions for supervised deep neural network," *Proc. 2021 Int. Conf. Syst. Sci. Eng. ICSSE 2021*, pp. 215–220, 2021.
[32] I. Firmansyah, J. T. Samudra, D. Pardede, and Z. Situmorang, "Comparison Of Random Forest And Logistic Regression In The Classification Of Covid-19 Sufferers Based On Symptoms," *J. Sci. Soc. Res.*, vol. 5, no. 3, p. 595, Oct. 2022.