Implementation of The CNN Deep Learning Method in Tajong (Sarung) Samarinda Classification

Sitti Muhartini[1], Andi Sunyoto[2], Alva Hendi Muhammad[3]
[1,2,3] Master of Informatics Engineering Department, Universitas Amikom Yogyakarta, Yogyakarta

Email: [1]12tini@students.amikom.ac.id

Received: 2024-08-06 Received in revised from 2024-08-20 Accepted: 2024-09-30

Abstract

Samarinda sarongs are one of Indonesia's traditional fabrics that are famous for their beautiful motifs and textures. This fabric is made using traditional weaving techniques using non-machine looms (ATBMs), resulting in a unique and distinctive diversity of textures. The difference between the loom, namely the machine and the non-machine, resulting in a difference in the texture of the Samarinda sarong. This difference can be seen from the thread density, texture smoothness, and sharpness of the motif. On certain Samarinda sarong motifs that do not require special details. This study aims to develop a classification model of Samarinda sarong texture based on the loom (machine and non-machine) using the Deep Learning method. This model is expected to help, increase the selling value of Samarinda sarongs, preserve and promote traditional fabrics In this context, the choice between DenseNet121 and VGG16 can depend on user preferences or specific needs, such as computing speed or model size.

Keywords: *ATBM; CNN; DenseNet121; Sarong; VGG16*

1. Introduction

The Samarinda sarong is a traditional Indonesian fabric which is famous for its beautiful motifs and textures. This fabric is made using traditional weaving techniques using non-machine looms (ATBM), producing a variety of unique and distinctive textures. The differences in looms, namely machines and non-machines, result in differences in the texture of the Samarinda sarong. This difference can be seen from the density of the thread, the smoothness of the texture, and the sharpness of the motif. For certain Samarinda sarong motifs that do not require special details, machine weaving results in the appearance of woven sarongs that almost resemble non-machine woven products. This can of course be an opening for irresponsible sellers to deceive consumers who don't really understand the differences between these fabrics.

The proposed deep learning-based model with data augmentation and transfer learning for automatic woven fabric classification [1], was proven to be more accurate and reliable than other methods with the application of the ResNet-50 method which was proven to be more accurate than VGGNet. [2] applies a pre-processing stage where the fabric image is converted to grayscale format, blurred using Gaussian blur, and binarized to remove noise. Then, fabric texture features are extracted using the LSTM model. The extracted fabric texture features are features that represent dark and light patterns in the fabric image. Test results show that this method can classify fabric texture with an accuracy of 98.8% and detect fabric defects with an accuracy of 97.2%.

This research aims to develop a texture classification model for Samarinda sarongs based on the loom (machine and non-machine) using the Deep Learning method. It is hoped that this model can help increase the selling value of Samarinda sarongs, preserve and promote traditional fabrics. Based on previous research, this study used 500 images of Samarinda sarongs (250 machines, 250 non-machines). The main model used is ResNet, and compared with other models such as VGGNet and LeNet. In previous research, Gaussian blur was used to clarify defects in fabric, but it is more appropriate to use it on images that need to be smoothed. Meanwhile, this research applies and compares the evaluation results of the CNN LeNet, VGG16, ResNet50 and DenseNet50 models. Using 1000 texture image data of Samarinda woven fabric produced by non-machine looms and 1000 texture image data of Samarinda woven fabric produced by machine looms. This image was taken manually using a digital microscope X4 1600X. It is hoped that the results of this research can contribute to the development of texture detection and fabric detection systems for traditional Indonesian fabrics.

2. Method

Provide sufficient detail methods to allow the work to be reproduced. Methods already published should be indicated by a reference: only relevant modifications should be described.

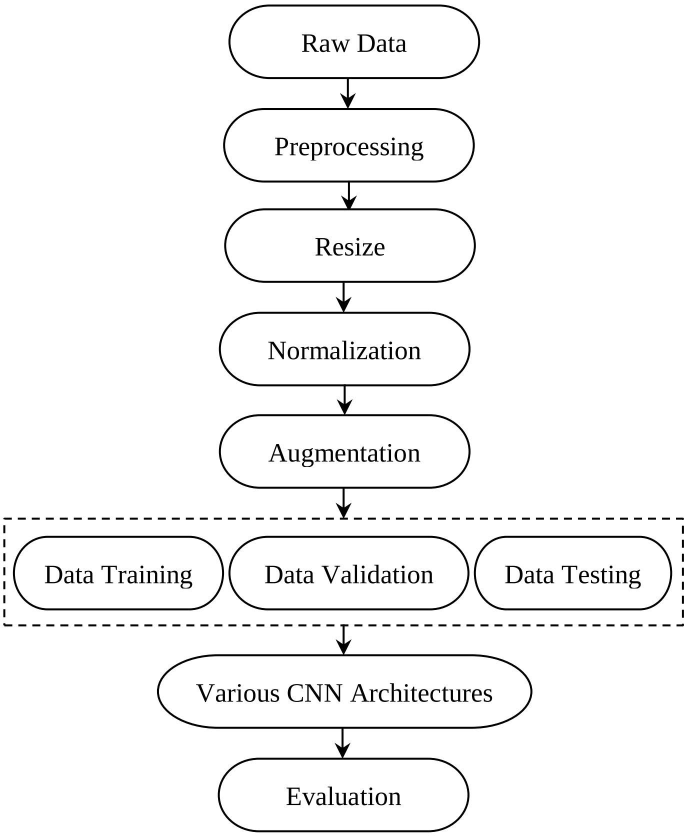


Figure 1. Flowchart of Research Methodology

Based on Figure 1. above regarding the flow of the research process, it can be briefly explained as follows:

1. Raw Data
   Sarong texture images are divided into two types, which will later be separated into each folder to help the classification process. To take the picture itself, use a digital microscope model X4 with a resolution of 1920x1440. For sarong texture images produced from machine looms, add them to the ATM folder and for sarong texture images produced from non-machine looms, add them to the ATBM folder.

2. Pre-Processing
   The texture image of ATBM and ATm sheaths becomes 256x256 pixels and is pre-processed using various techniques such as rotation, scale and translation.
3. Training and Training Model
   Carrying out the training process using the Samarinda sarong dataset which has been classified based on the loom. At this stage two scenarios are implemented with the first scenario using

80% of the data for training and 20% for testing. Meanwhile, the second scenario uses 70% of the data for training and 30% of the data for testing.

4. Testing and Testing Models
   The testing process of the test image uses CNN classification, and is evaluated using a confusion matrix. Where the previous data has been resized, normalized and augmented and Gabor Filter and G2RGB have been applied as additional data variants.
5. Trial and Evaluation
   Carry out the testing process by testing several test process flows as an evaluation process to determine the accuracy of model performance.

3. Results and Discussion

This research uses the Deep Learning CNN method with DenseNet121 and VGG16 which will be processed using Google Colab: https://colab.research.google.com/ using Python language. The scenario used is in accordance with the explanation regarding the Training and Model Training stages where the first scenario uses 80% of the data for training and 20% for testing. Meanwhile, the second scenario uses 70% of the data for training and 30% of the data for testing.

1. *Discussion for the VGG16 Split Data Model*
   The research used the CNN VGG16 Deep Learning method by applying two scenarios to obtain the following confusion matrix and graph results.
   a. Deep Learning CNN Model VGG16 Split Data (70:15:15)

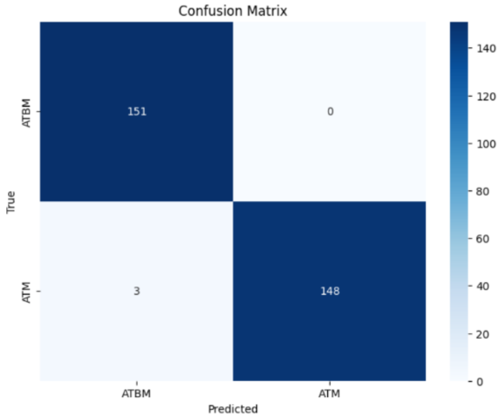


Figure 2. ***Confusin Matrix*** **Model VGG16** ***Split Data*** **(70:15:15)**

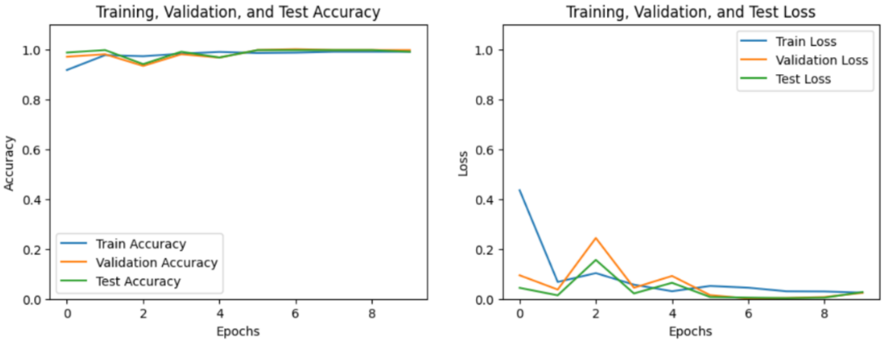


Figure 3. VGG16 ***Split Data*** **Model Graphic (70:15:15)**

Based on Figure 2 and Figure 3 using Google Colab, the following explanation is obtained:

Table 1. Description of Confusin Matrix VGG16 Split Data Model (70:15:15)

	Precision	Recall	F1-Score	Support
ATBM	0.980519	1.000000	0.990164	151.000000
ATM	1.000000	0.980132	0.989967	151.000000
Accuracy	0.990066	0.990066	0.990066	0.990066
Macro avg	0.990260	0.990066	0.990065	302.000000
Weighted avg	0.990260	0.990066	0.990065	302.000000

Table 2. Description of VGG16 Split Data Model Graphic (70:15:15)

	Epoch	Train_Accuracy	Val_Accuracy	Test_Accuracy	Train_Loss	Val_Loss	Test_Loss
0	1	0.917024	0.970000	0.986755	0.435530	0.094685	0.044300
1	2	0.976395	0.980000	0.996689	0.067977	0.037385	0.014439
2	3	0.972103	0.933333	0.940397	0.103384	0.243958	0.156250
3	4	0.982117	0.980000	0.990066	0.057026	0.044664	0.021917
4	5	0.989270	0.966667	0.966887	0.030687	0.091958	0.064515
5	6	0.984979	0.996667	0.996689	0.051840	0.015214	0.007757
6	7	0.986409	1.000000	0.996689	0.044898	0.001467	0.005290
7	8	0.989986	0.996667	0.996689	0.030493	0.003727	0.003415
8	9	0.989986	0.996667	0.996689	0.029874	0.007417	0.005122
9	10	0.989986	0.996667	0.990066	0.025045	0.024459	0.027218

b. Deep Learning CNN VGG16 Split Data Model (80:10:10)

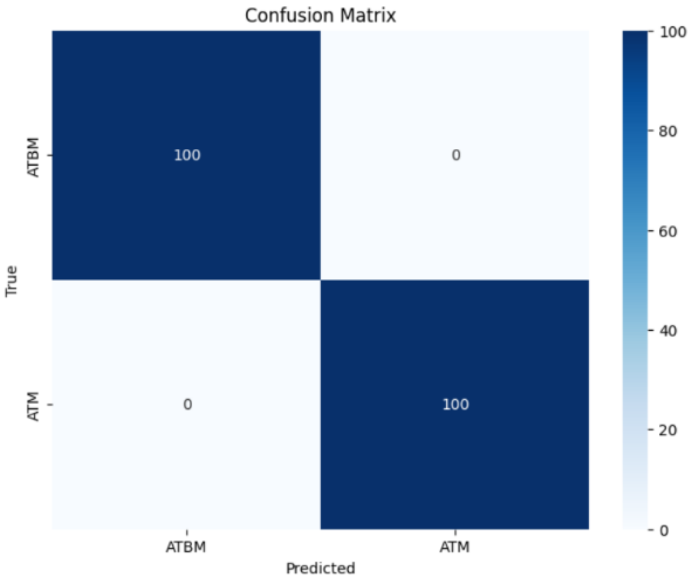


Figure 4. Confusin Matrix VGG16 Split Data Model (80:10:10)

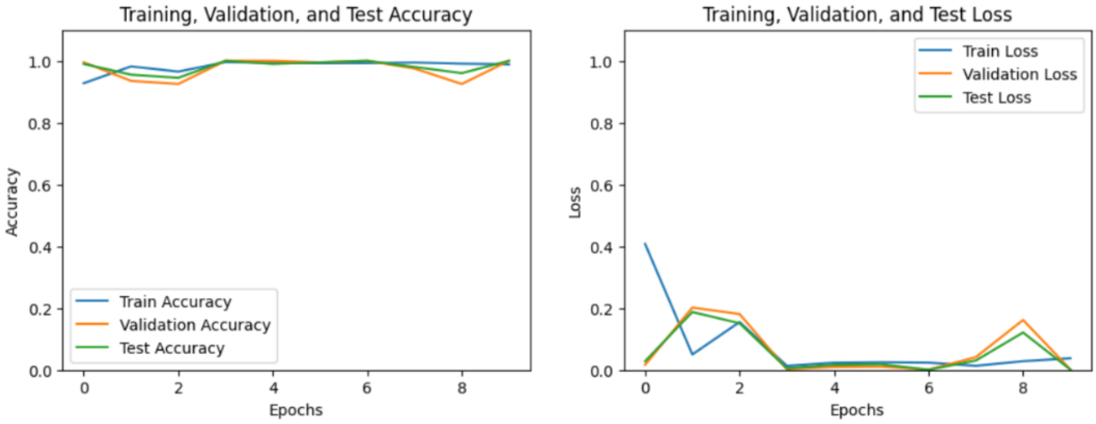


Figure 5. VGG16 Split Data Model Graphic (80:10:10)

Based on Figure 4 and Figure 5 using Google Colab, the following explanation is obtained:

Table 3. Description of Confusin Matrix Model VGG16 Split Data (80:10:10)

	Precision	Recall	F1-Score	Support
ATBM	1.0	1.0	1.0	100.0
ATM	1.0	1.0	1.0	100.0
Accuracy	1.0	1.0	1.0	1.0
Macro Avg	1.0	1.0	1.0	200.0
Weighted Avg	1.0	1.0	1.0	200.0

Table 4. Description of VGG16 Split Data Model Graphic (80:10:10)

	Epoch	Train_Accuracy	Val_Accuracy	Test_Accuracy	Train_Loss	Val_Loss	Test_Loss
0	1	0.927500	0.995	0.990	0.407685	0.016507	0.028084
1	2	0.981875	0.935	0.955	0.050083	0.201722	0.187348
2	3	0.965000	0.925	0.945	0.154959	0.180818	0.150915
3	4	0.995625	1.000	1.000	0.013188	0.002408	0.005206
4	5	0.993750	1.000	0.990	0.023375	0.009643	0.016633
5	6	0.991875	0.995	0.995	0.024857	0.010701	0.018224
6	7	0.992500	1.000	1.000	0.023591	0.001030	0.000527
7	8	0.994375	0.975	0.980	0.013314	0.041238	0.030434
8	9	0.990625	0.925	0.960	0.028279	0.161574	0.121092
9	10	0.988750	1.000	1.000	0.037643	0.001578	0.001363

2. *Discussion for the DenseNet121 Split Data Model*

   The research used the CNN DenseNet121 Deep Learning method by applying two scenarios to obtain the following confusion matrix and graph results.

   a. Deep Learning CNN Model DenseNet121 Split Data (70:15:15)

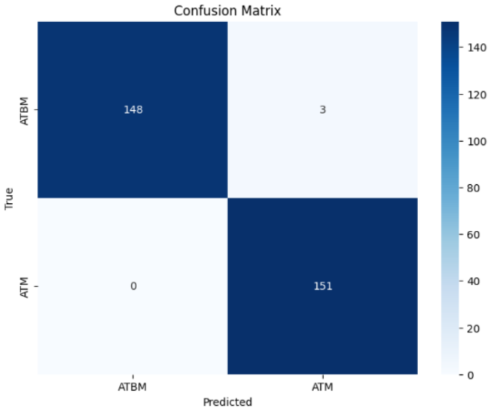


Figure 6. Confusin Matrix DenseNet121 Split Data Model (70:15:15)

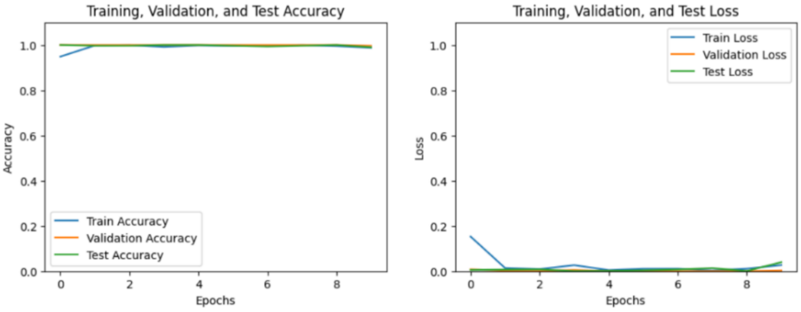


Figure 7. DenseNet121 Split Data Model Graphic (70:15:15)

Based on Figure 6 and Figure 7 using Google Colab, the following explanation is obtained:

Table 5. Description of Confusin Matrix DenseNet121 Split Data Model (70:15:15)

	Precision	Recall	F1-Score	Support
ATBM	1.000000	0.980132	0.989967	151.000000
ATM	0.980519	1.000000	0.990164	151.000000
Accuracy	0.990066	0.990066	0.990066	0.990066
Macro avg	0.990260	0.990066	0.990065	302.000000
Weighted avg	0.990260	0.990066	0.990065	302.000000

Table 6. Description of DenseNet121 Split Data Model Graphic (70:15:15)

	Epoch	Train_Accuracy	Val_Accuracy	Test_Accuracy	Train_Loss	Val_Loss	Test_Loss
0	1	0.948498	1.000000	1.000000	0.153032	0.007957	0.005389
1	2	0.997854	1.000000	0.996689	0.013773	0.002115	0.006867
2	3	0.999285	1.000000	0.996689	0.009040	0.001084	0.007771
3	4	0.991416	1.000000	1.000000	0.026583	0.005156	0.000676
4	5	0.997854	1.000000	1.000000	0.004535	0.000318	0.001566
5	6	0.995708	1.000000	0.996689	0.010445	0.000157	0.003739
6	7	0.996423	1.000000	0.993378	0.010755	0.000055	0.007287
7	8	0.999285	1.000000	0.996689	0.003271	0.000113	0.013394
8	9	0.994993	1.000000	1.000000	0.010817	0.000126	0.000776
9	10	0.987840	0.996667	0.990066	0.026959	0.003237	0.039668

a. *Deep Learning* CNN Model DenseNet121 *Split Data* (80:10:10)

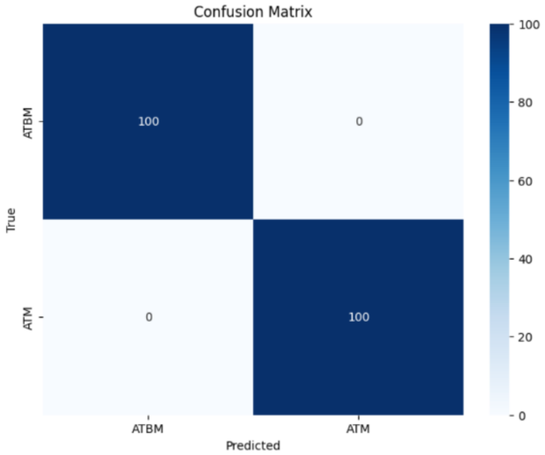


Figure 8. Confusin Matrix DenseNet121 Split Data Model (80:10:10)

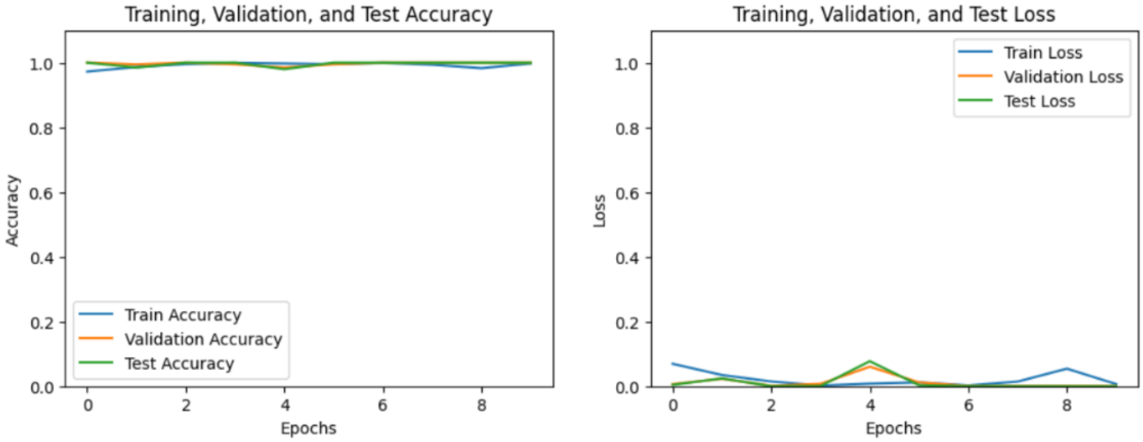


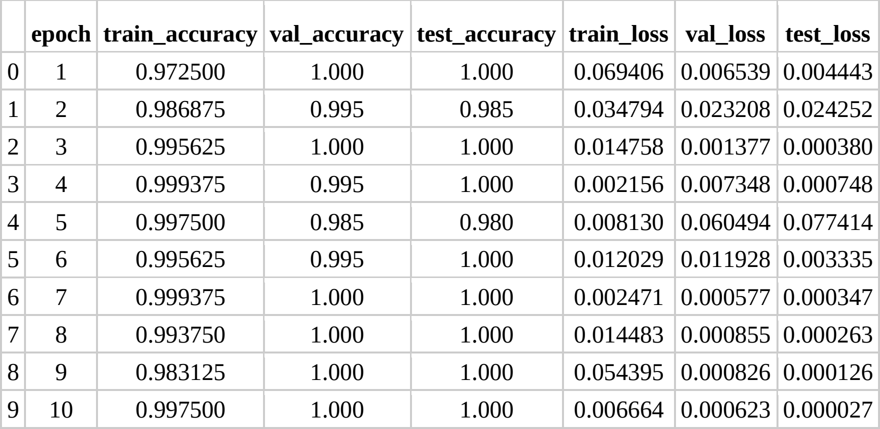
Figure 9. DenseNet121 Split Data Model Graphic (80:10:10)

Based on Figure 8 and Figure 9 using Google Colab, the following explanation is obtained:

Table 7. Confusin Matrix Model DenseNet121 Split Data (80:10:10) Explanation

	Precision	Recall	F1-Score	Support
ATBM	1.0	1.0	1.0	100.0
ATM	1.0	1.0	1.0	100.0
Accuracy	1.0	1.0	1.0	1.0
Macro avg	1.0	1.0	1.0	200.0
Weighted avg	1.0	1.0	1.0	200.0

Table 8. Description of DenseNet121 Split Data Model Graphic (80:10:10)



	epoch	train_accuracy	val_accuracy	test_accuracy	train_loss	val_loss	test_loss
0	1	0.972500	1.000	1.000	0.069406	0.006539	0.004443
1	2	0.986875	0.995	0.985	0.034794	0.023208	0.024252
2	3	0.995625	1.000	1.000	0.014758	0.001377	0.000380
3	4	0.999375	0.995	1.000	0.002156	0.007348	0.000748
4	5	0.997500	0.985	0.980	0.008130	0.060494	0.077414
5	6	0.995625	0.995	1.000	0.012029	0.011928	0.003335
6	7	0.999375	1.000	1.000	0.002471	0.000577	0.000347
7	8	0.993750	1.000	1.000	0.014483	0.000855	0.000263
8	9	0.983125	1.000	1.000	0.054395	0.000826	0.000126
9	10	0.997500	1.000	1.000	0.006664	0.000623	0.000027

4. Conclussion

Both models are able to perform the task of fabric texture classification very well. There is no significant difference in the overall effectiveness of these two models, making them both excellent choices for this task. However, VGG16 is slightly simpler in architecture than DenseNet121, which may make it faster in training and inference. In this context, the choice between DenseNet121 and VGG16 may depend on user preferences or specific needs, such as computing speed or model size.

Reference

[1] Hussain, M. A. I., Khan, B., Wang, Z., & Ding, S, "Woven fabric pattern recognition and classification based on deep convolutional neural networks," *Electronics (Switzerland)*, 9(6), 2020. https://doi.org/10.3390/electronics9061048

[2] Kumar, K. S., & Bai, M. R, "LSTM based texture classification and defect detection in a fabric. *Measurement: Sensors"*, 26, 2023. https://doi.org/10.1016/j.measen.2022.100603

[3] Agastya, I. M. A., & Setyanto, A, "Classification of Indonesian batik using deep learning techniques and data augmentation", *Proceedings - 2018 3rd International Conference on Information Technology, Information Systems and Electrical Engineering*, ICITISEE 2018. https://doi.org/10.1109/ICITISEE.2018.8720990

[4] Andrian, R., Hermanto, B., & Kamil, R. (2019). The Implementation of Backpropagation Artificial Neural Network for Recognition of Batik Lampung Motive. Journal of Physics: Conference Series, 1338(1), 0–10. https://doi.org/10.1088/1742-6596/1338/1/012062

[5] Aras, S., Setyanto, A., & Rismayani. (2022). Classification of Papuan Batik Motifs Using Deep Learning and Data Augmentation. 2022 4th International Conference on Cybernetics and Intelligent System, ICORIS 2022. https://doi.org/10.1109/ICORIS56080.2022.10031320

[6] Fadlil, A., Riadi, I., & Purwadi Putra, I. J. D. E. (2023). Comparison of Machine Learning Performance Using Naive Bayes and Random Forest Methods to Classify Batik Fabric Patterns. Revue d'Intelligence Artificielle, 37(2), 379–385. https://doi.org/10.18280/ria.370214

[7] Haralick RM, S. K. (1973). IEEE Transactions on systems, man, and cybernetics:610– 621, 1973. Textural Features for Image Classification, 3, 610–621.

[8] Hu, Y., Long, Z., Sundaresan, A., Alfarraj, M., AlRegib, G., Park, S., & Jayaraman, S. (2021). Fabric surface characterization: assessment of deep learning-based texture representations using a challenging dataset. Journal of the Textile Institute, 112(2). https://doi.org/10.1080/00405000.2020.1757296

[9] Juwita, A. R., & Solichin, A. (2018). Batik pattern identification using GLCM and artificial neural network backpropagation. Proceedings of the 3rd International Conference on Informatics and Computing, ICIC 2018. https://doi.org/10.1109/IAC.2018.8780412

[10] Khasanah, C. U., Utami, E., & Raharjo, S. (2020). Implementation of Data Augmentation Using Convolutional Neural Network for Batik Classification. 2020 8th International Conference on Cyber and IT Service Management, CITSM 2020, 20–24. https://doi.org/10.1109/CITSM50537.2020.9268890

[11] Kusanti, J., & Suprapto, A. (2019). Combination of Otsu and Canny Method to Identify the Characteristics of Solo Batik as Surakarta Traditional Batik. Proceedings - 2019 2nd International Conference of Computer and Informatics Engineering: Artificial Intelligence Roles in Industrial Revolution 4.0, IC2IE 2019. https://doi.org/10.1109/IC2IE47452.2019.8940884

[12] Lizarraga-Morales, R. A., Correa-Tome, F. E., Sanchez-Yanez, R. E., & Cepeda-Negrete, J. (2019). On the Use of Binary Features in a Rule-Based Approach for Defect Detection on Patterned Textiles. IEEE Access, 7. https://doi.org/10.1109/ACCESS.2019.2896078

[13] Putra, M. T. D., & Kusuma, G. P. (2019). Batik Classification using Deep Learning. International Journal of Recent Technology and Engineering (IJRTE), 8(4), 11416–11421. https://doi.org/10.35940/ijrte.d9039.118419

[14] Seçkin, A. Ç., & Seçkin, M. (2022). Detection of fabric defects with intertwined frame vector feature extraction. Alexandria Engineering Journal, 61(4). https://doi.org/10.1016/j.aej.2021.08.017

[15] Septiarini, A., Saputra, R., Tedjawati, A., Wati, M., & Hamdani, H. (2022). Pattern Recognition of Sarong Fabric Using Machine Learning Approach Based on Computer Vision for Cultural Preservation. International Journal of Intelligent Engineering and Systems, 15(5), 284–295. https://doi.org/10.22266/ijies2022.1031.26

[16] Simon, P., & Uma, V. (2020). Deep Learning based Feature Extraction for Texture Classification. Procedia Computer Science, 171. https://doi.org/10.1016/j.procs.2020.04.180

[17] Szeliski, R. (2021). Computer Vision : Algorithms and Applications 2nd Edition. In Springer.