# Identification of Batik Making Method from Images using Convolutional Neural Network with Limited Amount of Data

*by* Mohammad Arif Rasyidi

❏     1

# Identification of Batik Making Method from Images using Convolutional Neural Network with Limited Amount of Data

**Mohammad Arif Rasyidi[1], Ruktin Handayani[2], Fauzul Aziz[3]**
[1,2,3]Department of Informatics, Universitas Internasional Semen Indonesia, Indonesia

| Article Info | ABSTRACT |
|---|---|
| | This study aims to apply the Convolutional Neural Network (CNN) to classify batik based on its manufacturing method, namely *Batik Tulis* which are hand drawn, *Batik Cap* where stamps are used to create the pattern, and *Batik Printing* which are printed using textile printing machine. We collected 40 images for each type of batik with a total of 120 images. To speed up and simplify the model building process, we implemented transfer learning with 3 basic CNN model architectures, namely ResNet, DenseNet, and VGG with batch normalization. We also experimented with building a new dataset by breaking each image down into 30 smaller images. Image augmentation was also used to prevent overfitting as well as to provide variations in the training data. The experimental results with 5-fold cross validation show that densenet169 gives the best results on the original dataset with an accuracy of 79.17% while vgg13_bn shows the best performance on the modified dataset with an accuracy of 87.61%. All models showed an increase in performance when using the modified dataset, except densenet169 which did not show a significant difference in performance. |

***Corresponding Author:***

Mohammad Arif Rasyidi,
Department of Informatics,
Universitas Internasional Semen Indonesia,
Jl. Veteran, Gresik, Jawa Timur 61122, Indonesia.
Email: mohammad.rasyidi@uisi.ac.id

## 1.   INTRODUCTION

Since it was first created, perhaps hundreds to thousands of years ago, until it was named Masterpieces of the Oral and Intangible Heritage of Humanity by UNESCO in 2009 [1] and to this day, batik, a patterned cloth from Indonesia, has undergone many developments. Traditionally the process of making batik begins with drawing the desired design pattern on the fabric using a pencil. The manufacturing process is continued by applying liquid wax on both sides of the cloth according to the previously formed pattern using canting as shown in Figure 1. Canting is a pen-like instrument which consists of bamboo handle and copper wax pot with a narrow pipe spout. Next, the fabric is dyed by dipping it in a dye solution. In this process, the part of the cloth that has been covered with wax before will not get wet, so that the original color is maintained. The wax is then removed by scraping it with a knife or other metal object. The cloth is then washed and boiled to remove any lingering wax residue. The process of waxing, coloring, and removing wax was repeated for each required color combination [2], [3]. Because the batik maker draws the design pattern using the canting by hand, the batik produced by this method is called *Batik Tulis*, which comes from the Javanese language for "writing". Due to the complexity of the process, the process of making *Batik Tulis* takes a long time. It can take 30 to 50 days or even months for producing a piece of cloth [2], [3].

Around 1840, a method of making batik using *cap* (stamp) was developed to replace canting [3]. A *cap* is a block-shaped printing tool made of metal such as copper with a batik pattern. With a *cap*, the batik

maker can apply the entire batik pattern with one application (see Figure 1). This greatly accelerates the process of making batik from the previous method. Additionally, in general, the resulting design patterns are tidier than manual work using canting. However, unlike *Batik Tulis* which has a color pattern on both sides, most *Batik Cap* only has a color on one side of the fabric.

Along with technological developments and the need for more affordable batik, a new modern variation emerged, namely *Batik Printing*. Unlike the two previous variants, *Batik Tulis* and *Batik Cap*, which are produced using wax, the *Batik Printing* production process does not require this. The entire manufacturing process of *Batik Printing* is carried out using a textile machine, which often raises debate as to whether the fabric with the resulting batik pattern can be called batik.



(a) (b)
Figure 1. The process of making batik using *tulis* (a) and *cap* (b) method

There is no doubt that the variation in the method of producing batik creates a difference in the price of the batik fabric made. Batik produced by the hand drawn method (*tulis*) usually has a much higher market value than batik fabric produced by other methods due to the length of the production process and the artistic value it displays. While it can be easily identified visually by a skilled person, often it is difficult for average people to find out how the batik is produced because of their similar design patterns. Furthermore, in the online trade sector, buyers can only see photos of the batik cloths being sold without being able to hold and inspect them directly, which is prone to misidentification and can lead to fraud.

To solve this problem, in this study we propose a supervised learning approach by building a classification model to help ordinary people identify methods of making batik. The classification model for the batik making method will be built using the Convolutional Neural Network (CNN) and transfer learning. CNN is a special type of deep learning which is very often applied to the image processing field. Unlike traditional machine learning methods which require a feature extraction process from images prior to model training, CNN incorporates the process into the training phase, and thus does not require a separate feature extraction step. This approach is what makes it so powerful and popular today. Some examples of CNN applications include fashion image classification [4], classification of retinal tomographic images [5], and handwriting recognition [6].

One of the challenges we faced in building such a classification model was the insufficient amount of labeled data that we were able to collect. However, as we all know, deep learning methods such as CNN require a sufficient amount and variety of data to be able to produce high-quality models. To overcome this problem, the approach we take is to take advantage of CNN models that have been previously developed by researchers for other problems and adapt them for classifying batik making method. This technique is often known as transfer learning. With several proven and readily available CNN models, such as AlexNet [7], MobileNet [8], [9], ResNet [10], DenseNet [11], SqueezeNet [12], and VGG [13], it is possible to create new CNN models based on these models so that the training phase can be carried out more rapidly and easily.

In the literature, there have been many previous studies related to batik, for example, [14–28]. Most of these studies concentrate more on the feature extraction aspect of the classification. Generally, in the image recognition process, the features generated from the feature extraction process are used as input to train prediction models using traditional machine learning algorithms. Although this approach is very popular, with the advent of deep learning, especially with CNN, new classification methods that do not require manual feature extraction are starting to be widely applied to batik classification problems [29–34]. However, despite the vast number of studies related to batik, only a few studies have focused on predicting batik-making methods, with others more frequently aimed for batik pattern classification. One of the studies we found that focuses on the prediction of the batik making method is [24] which utilizes the Gray Level Co-occurrence Matrix (GLCM) for the feature extraction process from the image and Learning Vector

Quantization (LVQ) for classification. Even though the resulting performance of this study is quite high with almost 99% accuracy, the study uses even less data than what we used in this study, namely 10 images for each type of batik. Each image is then split into 135 images and then randomly divided into training and test size with a ratio of 2:1. The weakness of the method used is that because each image is broken down into smaller images before being divided into training and test data, the fractions of the same image will be scattered partly in the training data and partly in the test data. Thus, when there are some parts in the training data that are exactly the same or very similar to some images in the test set, the result will be misleading as the model may not have learned anything useful to classify new images. As the images come from the same source image, the probability of this happening is very high, so its applicability and result may need to be reviewed.

Accordingly, our contributions are as follows. We are one of the few who focus on predicting batik making methods and not on batik patterns. To the best of our knowledge, we are also the first apply CNN to classify batik making methods, albeit using only small amount of data. Finally, we also experimented by splitting our images in building the model, which is similar to the approach of [24]. However, we made improvements in the experimental design where we only split the image after dividing it into training and test sets. Thus, it is expected that the resulting model performance measurement results can be more accurate.

## 2. RESEARCH METHOD

As shown in Figure 2, our research methodology mainly consists of five main processes: data collection, cross validation, image augmentation, model training, and testing.
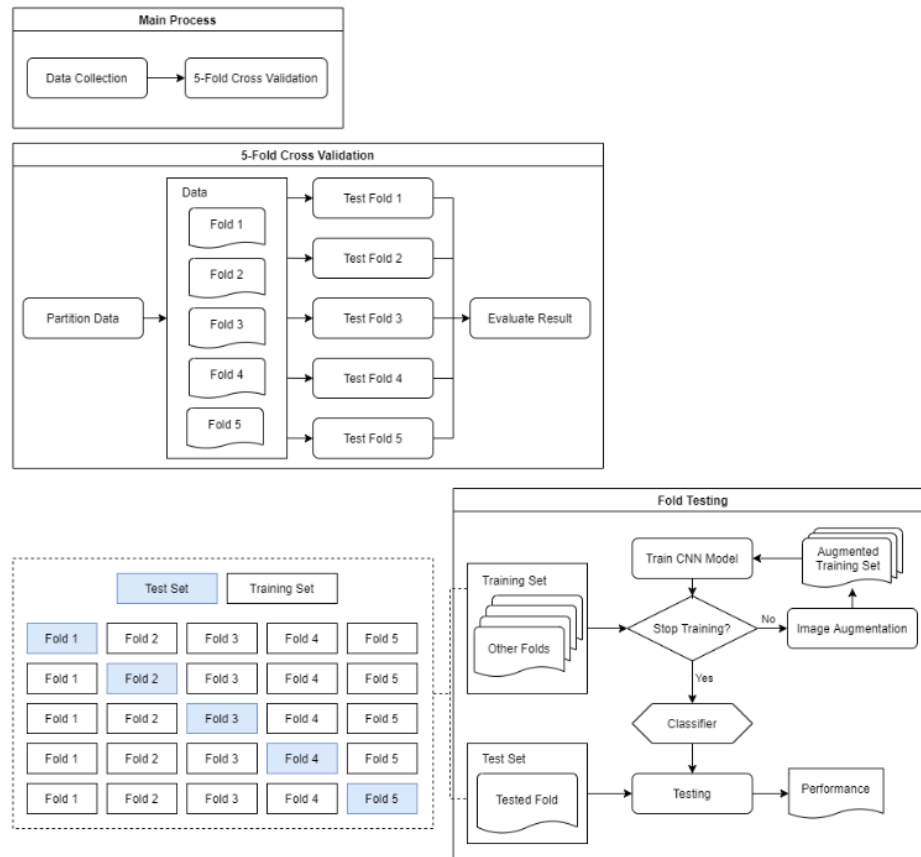


Figure 2. Research Method

## 2.1. Data Collection

We collected a total of 120 images of batik, which consist of batik *tulis*, *cap*, and *printing* (40 images for each type of batik). Each of the collected images has been cropped and resized to 540x630 pixels. Some examples of batik images are shown in Figure 3.

*Batik Tulis*              *Batik Cap*              *Batik Printing*

Figure 3. Some examples of images used in this study

## 2.2. Cross Validation

Due to the small amount of data collected, the test method using an ordinary holdout set may cause high bias and variation in the performance being measured. Therefore, in this study, we used the 5-fold cross validation method to overcome this. In 5-fold cross validation, the dataset is divided randomly into 5 parts called folds with each fold containing 24 images (8 images for each type of batik). Then, alternately each fold is used as a test-set while the other 4 folds will be used as training data to build a model that will be tested on the test fold as illustrated in Figure 1. The five test results obtained in each iteration will then be averaged to get the final performance.

Apart from using the original data directly, we also experimented by splitting each image into 30 different images with a size of 180x180 pixels. In this image splitting process, we first divide the image into 42 blocks, each measuring 90x90 pixels. Next, we formed new images by taking 2x2 blocks starting from the top left corner then shifting them to the right and down by 1 block. This is similar to the convolution and pooling operations on CNN with a filter size of 2x2 and stride 1. This process is illustrated in Figure 4. From this process, we obtained 720 images for each fold for a total of 3600 images. The reason we do the image splitting process after dividing the data into 5 folds and not before is so that the results of dividing each image are not separated in training and test sets which might cause the measured performance to be less reliable. When using the model built using this dataset in production, because the original image is divided into 30 smaller images, we will of course get 30 different predictions. The final prediction result for the original image can thus be obtained simply by a majority vote.
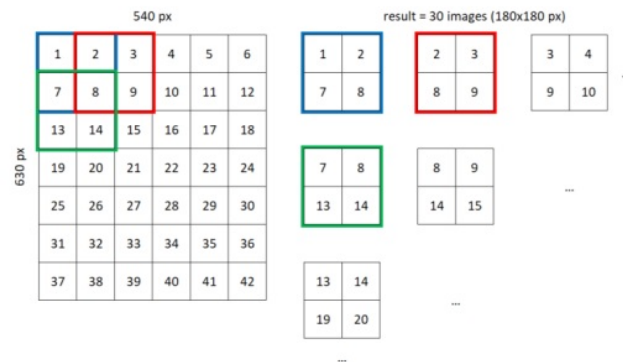
Figure 4. Illustration of image splitting process

## 2.3. Model Training

With the small amount of data we have collected, the trained model will be prone to overfitting. Like our previous research [34], image augmentation is used to overcome this problem. In the training process, in each iteration, the images given to the model will be altered with a small random transformation

without losing their original content. The transformations used in this study include horizontal and vertical flip, rotation, zoom, lighting, and symmetric wrap. Table 1 shows the parameters of each transformation.

Table 1. Parameters for image augmentation

| Transformation | Value |
|---|---|
| Horizontal Flip | True/False |
| Vertical Flip | True/False |
| Max Rotation | 45° |
| Max Zoom | 1.3 |
| Max Lighting | 0.3 |
| Max Symmetric Warp | 0.2 |

## 2.4. Image Augmentation

As previously stated, the model building was carried out using 4/5 parts of the total data. From this data, we further divide it into 3: 1 with 3 parts used as training data while the remaining 1 part is used for data validation. Validation data is used to determine when the training process must be stopped so that the resulting model does not overfit the training data.

There are 3 types of CNN architectures and their variations that we use, namely ResNet (resnet18, resnet34, resnet50, resnet101 and resnet152) [10], DenseNet (densenet121, densenet169, densenet201 and densenet161) [11], and VGG with batch normalization (vgg11_bn, vgg13_bn, vgg16_bn and vgg19_bn) [13]. The training process is carried out using these basic models which have been previously trained on the ImageNet dataset and modified to accommodate the identification problem of the batik making method. To speed up the training process, each model is trained using the 1cycle policy [35]. In addition, early stopping is also applied to avoid overfitting.

## 2.5. Testing

Testing is done by applying trained models to the test data. Confusion matrix as shown in Figure 5 is then used to summarize the result of the test. There are four criteria tested, namely accuracy, precision, recall, and F-measure which will be calculated according to the following formulas

$$accuracy = \frac{\sum_{i=1}^{k} TP_i}{n} \qquad (1)$$

$$precision = \frac{\sum_{i=1}^{k} \frac{TP_i}{TP_i + FP_i}}{k} \qquad (2)$$

$$recall = \frac{\sum_{i=1}^{k} \frac{TP_i}{TP_i + FN_i}}{k} \qquad (3)$$

$$F\text{-}measure = \frac{\sum_{i=1}^{k} \frac{2\,TP_i}{2\,TP_i + FP_i + FN_i}}{k} \qquad (4)$$

where:

TP   = True Positive
FN   = False Negative
FP   = False Positive
$n$    = Number of samples in the test set
$k$    = Number of classes

|  | C1 | C2 | C3 |
|---|---|---|---|
| C1 | **TP₁** | FP₂ / FN₁ | FP₃ / FN₁ |
| C2 | FP₁ / FN₂ | **TP₂** | FP₃ / FN₂ |
| C3 | FP₁ / FN₃ | FP₂ / FN₃ | **TP₃** |

**Actual Class** (vertical) — **Prediction** (horizontal, C1 C2 C3)

Figure 5. Sample Confusion Matrix. Here C1, C2, and C3 represent *Batik Tulis*, *Batik Cap*, and *Batik Printing* respectively

# 3.    RESULTS AND DISCUSSION

In this section, we discuss the overall model performance, the effect of layer count on model performance, and the effect of image splitting on model performance.

## 3.1.  Overall Performance of the Models

Table 2 shows the performance of each classification model built against the original full-size dataset as well as the modified dataset with the image broken down into smaller sizes. On the original dasaset, the best performance was shown by densenet169 with an accuracy of 79.17%, precision of 80.98%, recall of 79.17% and F-measure of 79.03%. Meanwhile, the worst performance is shown by vgg11_bn with accuracy, precision, recall, and F-measure below 50%, only slightly above the random baseline classifier (33.33%). In the modified dataset, the greatest performance was achieved by vgg13_bn, with an accuracy of 87.61%, 88.62% precision, 87.81% recall, and 87.32% F-measure which were the highest values for each performance metric. In general, all models perform fairly well, with above 75% performance across all models and measurement metrics.

Table 2. The performance of each model on the original and modified dataset

| Model | Original Dataset | | | | Modified Dataset | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F-Measure | Accuracy | Precision | Recall | F-Measure |
| resnet18 | 0.6917 | 0.7030 | 0.6917 | 0.6868 | 0.7794 | 0.7932 | 0.7794 | 0.7745 |
| resnet34 | **0.7500** | 0.7635 | **0.7500** | **0.7419** | 0.7981 | 0.8077 | 0.7981 | 0.7956 |
| resnet50 | 0.7083 | 0.7337 | 0.7083 | 0.7080 | 0.8303 | 0.8434 | 0.8303 | 0.8282 |
| resnet101 | 0.7167 | 0.7263 | 0.7167 | 0.7136 | 0.8347 | 0.8610 | 0.8347 | 0.8317 |
| resnet152 | 0.6917 | 0.7324 | 0.6917 | 0.6891 | 0.8408 | 0.8535 | 0.8408 | 0.8398 |
| densenet121 | 0.6833 | 0.6812 | 0.6833 | 0.6654 | **0.8550** | **0.8660** | **0.8550** | **0.8513** |
| densenet161 | **0.7500** | 0.8008 | **0.7500** | 0.7380 | 0.8278 | 0.8420 | 0.8278 | 0.8271 |
| densenet169 | *0.7917* | *0.8098* | *0.7917* | *0.7903* | 0.7972 | 0.8089 | 0.7972 | 0.7910 |
| densenet201 | 0.7417 | **0.7700** | 0.7417 | 0.7368 | 0.7989 | 0.8162 | 0.7989 | 0.7928 |
| vgg11_bn | 0.4417 | 0.4223 | 0.4417 | 0.4085 | 0.8481 | 0.8634 | 0.8481 | 0.8448 |
| vgg13_bn | 0.5917 | 0.5984 | 0.5917 | 0.5782 | *0.8761* | *0.8862* | *0.8761* | *0.8732* |
| vgg16_bn | 0.5250 | 0.4884 | 0.5250 | 0.4782 | **0.8550** | **0.8658** | **0.8550** | **0.8526** |
| vgg19_bn | 0.6500 | 0.6580 | 0.6500 | 0.6340 | 0.8392 | 0.8523 | 0.8392 | 0.8365 |

## 3.2.  The Effect of Increasing the Number of Layers in the Model on Performance

From Table 2, we can also see that there is no clear relationship between the number of layers in the model and the resulting performance. In the original dataset, DenseNet performance tends to get better with increasing layers. However, this does not apply to densenet201 where the performance actually decreases when compared to densenet169. The opposite occurs in the modified dataset, where the performance of DenseNet models tends to decrease as the number of layers in the model increases. The DenseNet variant with the least number of layer (densenet121) actually shows the best performance compared to other model variants in the modified dataset. ResNet also shows similar thing. In the modified dataset, ResNet's performance in terms of accuracy tends to increase as the number of layers increases. However, the same thing is not shown in the original dataset where the best performance is shown by resnet34. Whereas in VGG, model performance tends to fluctuate as the number of layers increases. Thus, we cannot conclude with certainty the relationship between the number of layers in the model and the resulting performance.

## 3.3.  Comparison of Model Performance on the Original and Modified Datasets

Next, we will look at the effect of splitting the image to form a new dataset on the performance generated by the classification model. Figure 6 shows the difference in the performance of the model built with the modified dataset against the performance of the model built with the original dataset. From this figure, it can be seen that almost all models show an increase in performance when compared to the model in the original dataset. The highest increase is shown by VGG with one of its variants, namely vgg11_bn, showing a performance increase of around 100% in terms of accuracy, precision, recall, and F-measure. This is quite surprising considering that vgg11_bn was the model that performed the worst on the original dataset. In terms of performance improvement, ResNet is in second place where it can be seen from the graph that along with the increase in the number of layers, the performance improvement tends to increase. Whereas in Densenet the opposite happened where the increase in the number of layers was followed by a decrease in the performance increase. For densenet169, there is almost no difference in performance between the models built with the original dataset when compared to the models with the new dataset. This shows that in most architectures such as VGG and ResNet, the number and size of images have a very significant effect on the performance of the resulting model. Whereas in other architectures such as densenet169, the resulting model performance is not too sensitive to the number and size of training images. Further experiments are needed to investigate this.
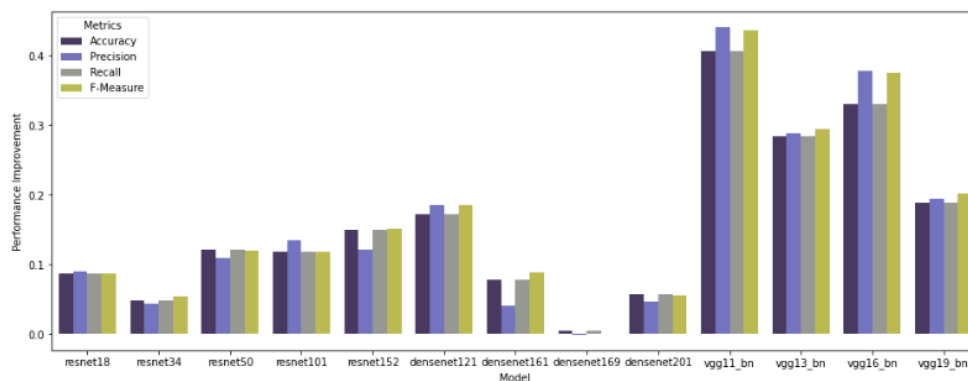
Figure 6. Difference in performance of models trained with the modified dataset against the models built using the original dataset

## 4.    CONCLUSION

From the results of this study, we have seen that CNN can be applied to the problem of identifying the method of batik making with a limited amount of data and has shown a fairly good performance. The best performance on native data is achieved by models with the densenet169 architecture. Whereas in the dataset with images broken down into smaller sizes, vgg13_bn shows the best performance. In most models, using a modified dataset will result in better performance when compared to using the original dataset. Although in certain variants such as densenet169, almost no change in performance was obtained from using the new dataset, which shows that densenet169 is not very sensitive to the number and size of images. In future research, we will try to implement CNN for similar problems with a larger amount of data and try other approaches such as anomaly detection to solve this problem. We will also try to apply CNN to the multiclass classification problem where one batik cloth can be made with a combination of different manufacturing methods, such as a combination of writing and stamp, writing and printing, and so on.

## REFERENCES

[1]    "Indonesian Batik - intangible heritage - Culture Sector - UNESCO." https://ich.unesco.org/en/RL/indonesian-batik-00170 (accessed Dec. 03, 2019).
[2]    N. Krevitsky, *Batik - Art and Craft*. New York, NY, USA: Reinhold Publishing Corporation, 1964.
[3]    I. M. Elliot, *Batik Fabled Cloth of Java*. Hong Kong: Periplus Editions (HK) Ltd., 2004.
[4]    Y. Seo and K. shik Shin, "Hierarchical convolutional neural networks for fashion image classification," *Expert Syst. Appl.*, vol. 116, pp. 328–339, Feb. 2019, doi: 10.1016/j.eswa.2018.09.022.
[5]    L. Fang, Y. Jin, L. Huang, S. Guo, G. Zhao, and X. Chen, "Iterative fusion convolutional neural networks for classification of optical coherence tomography images," *J. Vis. Commun. Image Represent.*, vol. 59, pp. 327–333, Feb. 2019, doi: 10.1016/j.jvcir.2019.01.022.
[6]    R. Ptucha, F. Petroski Such, S. Pillai, F. Brockler, V. Singh, and P. Hutkowski, "Intelligent character recognition using fully convolutional neural networks," *Pattern Recognit.*, vol. 88, pp. 604–613, Apr. 2019, doi: 10.1016/j.patcog.2018.12.017.
[7]    A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," Apr. 2014, Accessed: Dec. 04, 2019. [Online]. Available: http://arxiv.org/abs/1404.5997.
[8]    A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017, Accessed: Aug. 20, 2019. [Online]. Available: http://arxiv.org/abs/1704.04861.
[9]    M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018, doi: 10.1109/CVPR.2018.00474.
[10]   K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2016, vol. 2016-December, pp. 770–778, doi: 10.1109/CVPR.2016.90.
[11]   G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," Aug. 2016, Accessed: Dec. 04, 2019. [Online]. Available: http://arxiv.org/abs/1608.06993.

[12] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," Feb. 2016, Accessed: Dec. 04, 2019. [Online]. Available: http://arxiv.org/abs/1602.07360.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[14] I. Nurhaida, R. Manurung, and A. M. Arymurthy, "Performance comparison analysis features extraction methods for Batik recognition," in *2012 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 2012, pp. 207–212.

[15] I. Nurhaida, A. Noviyanto, R. Manurung, and A. M. Arymurthy, "Automatic Indonesian's Batik Pattern Recognition Using SIFT Approach," in *Procedia Computer Science*, 2015, vol. 59, pp. 567–576, doi: 10.1016/j.procs.2015.07.547.

[16] I. Nurhaida, H. Wei, R. A. M. Zen, R. Manurung, and A. M. Arymurthy, "Texture fusion for batik motif retrieval system," *Int. J. Electr. Comput. Eng.*, vol. 6, no. 6, pp. 3174–3187, 2016, doi: 10.11591/ijece.v6i6.12049.

[17] I. Setyawan, I. K. Timotius, and M. Kalvin, "Automatic batik motifs classification using various combinations of SIFT features moments and k-Nearest Neighbor," in *Proceedings - 2015 7th International Conference on Information Technology and Electrical Engineering: Envisioning the Trend of Computer, Information and Engineering, ICITEE 2015*, 2015, pp. 269–274, doi: 10.1109/ICITEED.2015.7408954.

[18] A. E. Minarno, A. S. Maulani, A. Kurniawardhani, F. Bimantoro, and N. Suciati, "Comparison of methods for Batik classification using multi texton histogram," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 16, no. 3, pp. 1358–1366, Jun. 2018, doi: 10.12928/TELKOMNIKA.v16i3.7376.

[19] A. A. Kasim, R. Wardoyo, and A. Harjoko, "The selection feature for batik motif classification with information gain value," in *Communications in Computer and Information Science*, 2017, vol. 788, pp. 106–115, doi: 10.1007/978-981-10-7242-0_9.

[20] A. A. Kasim, R. Wardoyo, and A. Harjoko, "Batik Classification with Artificial Neural Network Based on Texture-Shape Feature of Main Ornament," *I.J. Intell. Syst. Appl.*, vol. 6, pp. 55–65, 2017.

[21] R. Azhar, D. Tuwohingide, D. Kamudi, Sarimuddin, and N. Suciati, "Batik Image Classification Using SIFT Feature Extraction, Bag of Features and Support Vector Machine," in *Procedia Computer Science*, 2015, vol. 72, pp. 24–30, doi: 10.1016/j.procs.2015.12.101.

[22] Farida, R. E. Caraka, T. W. Cenggoro, and B. Pardamean, "Batik Parang Rusak Detection Using Geometric Invariant Moment," in *1st 2018 Indonesian Association for Pattern Recognition International Conference, INAPR 2018 - Proceedings*, Jan. 2019, pp. 71–74, doi: 10.1109/INAPR.2018.8627000.

[23] C. Jatmoko and D. Sinaga, "A Classification of Batik Lasem using Texture Feature Ecxtraction Based on K-Nearest Neighbor," *J. Appl. Intell. Syst.*, vol. 3, no. 2, 2019, doi: 10.33633/jais.v3i2.2151.

[24] N. Yunari, E. M. Yuniarno, and M. H. Purnomo, "Indonesian batik image classification using statistical texture feature extraction Gray Level Co-occurrence Matrix (GLCM) and Learning Vector Quantization (LVQ)," *J. Telecommun. Electron. Comput. Eng.*, vol. 10, no. 2–3, pp. 67–71, 2018.

[25] F. U. Karimah and A. Harjoko, "Classification of batik kain besurek image using speed up robust features (SURF) and gray level co-occurrence matrix (GLCM)," in *Communications in Computer and Information Science*, 2017, vol. 788, doi: 10.1007/978-981-10-7242-0_7.

[26] W. Herulambang, M. N. Hamidah, and F. Setyatama, "Comparison of SVM and BPNN Methods in the Classification of Batik Patterns Based on Color Histograms and Invariant Moments," 2020, doi: 10.1109/ICoSTA48221.2020.1570615583.

[27] A. E. Minarno, Y. Azhar, F. D. Setiawan Sumadi, and Y. Munarko, "A Robust Batik Image Classification using Multi Texton Co-Occurrence Descriptor and Support Vector Machine," 2020, doi: 10.1109/ICoIAS49312.2020.9081833.

[28] A. E. Minarno, F. D. S. Sumadi, H. Wibowo, and Y. Munarko, "Classification of batik patterns using K-nearest neighbor and support vector machine," *Bull. Electr. Eng. Informatics*, vol. 9, no. 3, 2020, doi: 10.11591/eei.v9i3.1971.

[29] I. M. A. Agastya and A. Setyanto, "Classification of Indonesian batik using deep learning techniques and data augmentation," 2018, doi: 10.1109/ICITISEE.2018.8720990.

[30] A. Y. Wicaksono, N. Suciati, C. Fatichah, K. Uchimura, and G. Koutaki, "Modified Convolutional Neural Network Architecture for Batik Motif Image Classification," *IPTEK J. Sci.*, vol. 2, no. 2, 2017, doi: 10.12962/j23378530.v2i2.a2846.

[31] Y. Gultom, A. M. Arymurthy, and R. J. Masikome, "Batik Classification using Deep Convolutional Network Transfer Learning," *J. Ilmu Komput. dan Inf.*, vol. 11, no. 2, 2018, doi: 10.21609/jiki.v11i2.507.

[32] H. Prasetyo and B. A. Putra Akardihas, "Batik image retrieval using convolutional neural network," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 17, no. 6, 2019, doi: 10.12928/TELKOMNIKA.v17i6.12701.

[33] J. Tristanto, J. Hendryli, and D. Erny Herwindiati, "Classification of Batik Motifs Using Convolutional Neural Networks," *SSRN Electron. J.*, 2018, doi: 10.2139/ssrn.3258935.

[34] M. A. Rasyidi and T. Bariyah, "Batik pattern recognition using convolutional neural network," *Bull. Electr. Eng. Informatics*, vol. 9, no. 4, 2020, doi: 10.11591/eei.v9i4.2385.

[35] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 -- learning rate, batch size, momentum, and weight decay," Mar. 2018, Accessed: Dec. 04, 2019. [Online]. Available: http://arxiv.org/abs/1803.09820.

# Identification of Batik Making Method from Images using Convolutional Neural Network with Limited Amount of Data

8  Waego Hadi Nugroho, Samingun Handoyo, Yusnita Julyarni Akri. "An Influence of Measurement Scale of Predictor Variable on Logistic Regression Modeling and Learning Vector Quntization Modeling for Object Classification", International Journal of Electrical and Computer Engineering (IJECE), 2018
Publication

<1%

9  Jani Kusanti, Agung Suprapto. "Combination of Otsu and Canny Method to Identify the Characteristics of Solo Batik as Surakarta Traditional Batik", 2019 2nd International Conference of Computer and Informatics Engineering (IC2IE), 2019
Publication

<1%

10  www.mikroskil.ac.id
Internet Source

<1%

11  Naja Aqilah Hisham, Sheikh Ahmad Zaki Shaikh Salim, Aya Hagishima, Fitri Yakub, Hafizah Farhah Saipan Saipol. "Statistical analysis of air-conditioning and total load diversity in typical residential buildings", Bulletin of Electrical Engineering and Informatics, 2021
Publication

<1%

12  export.arxiv.org
Internet Source

<1%

**13** hdl.handle.net
Internet Source
<1%

**14** www.mun.ca
Internet Source
<1%

**15** github.com
Internet Source
<1%

**16** Danis Aditya Mardani, Pranowo, Albertus Joko Santoso. "Deep learning for recognition of Javanese batik patterns", AIP Publishing, 2020
Publication
<1%

**17** content.iospress.com:443
Internet Source
<1%

**18** Juan Li. "Real-time image recognition algorithm and system design of Android mobile devices", International Journal of Computers and Applications, 2020
Publication
<1%

**19** Minarno, Agus Eko, Yuda Munarko, Fitri Bimantoro, Arrie Kurniawardhani, and Nanik Suciati. "Batik image retrieval based on enhanced micro-structure descriptor", 2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE), 2014.
Publication
<1%

**20** dokumen.pub
Internet Source
<1%

21 repository.tudelft.nl
Internet Source
<1%

22 www.archive.org
Internet Source
<1%

23 www.hindawi.com
Internet Source
<1%

24 Sepp Hochreiter, Jürgen Schmidhuber. "Long Short-Term Memory", Neural Computation, 1997
Publication
<1%

Exclude quotes          Off                    Exclude matches          Off
Exclude bibliography    On