# Classification of Batik Patterns Using Inception-ResNetV2 with Data Augmentation

Budi D. Satoto [1,*], Wahyudi Agustiono [1], Suraya B. Hamid [2], Novia P. Ramadhani [1], Fiola L. Rafelina [1], Chafi A. R. Zakiy [1], Budi Irmawati [3], and Deshinta A. Dewi [4]

[1] Department of Information Systems, Faculty of Engineering, University of Trunojoyo Madura, Indonesia
[2] Department of Information System, Fakulti Sains Komputer dan Teknologi Maklumat, Universiti Malaya, Malaysia
[3] Department of Informatics Engineering, Faculty of Engineering, University of Mataram, Indonesia
[4] Center for Data Science and Sustainable Technologies, INTI International University, Nilai, Malaysia
Email: budids@trunojoyo.ac.id (B.D.S.); wahyudi.agustiono@trunojoyo.ac.id (W.A.); suraya_hamid@um.edu.my (S.B.H.); putrinopiaa@gmail.com (N.P.R); 210441100066@trunojoyo.ac.id (F.L.R.); 210441100067@trunojoyo.ac.id (C.A.R.Z.); budi-i@unram.ac.id (B.I.); deshinta.ad@newinti.edu.my (D.A.D.)
*Corresponding author

***Abstract*—One of Indonesia's cultural heritages with significant artistic and historical value is batik. The background of this research is the diversity of cultures and customs in various regions of Indonesia, which is reflected in the diverse batik patterns currently used. Classifying batik patterns is very important in preserving, recognizing, and promoting batik as a valuable cultural asset of the country. The problem statement is how classification facilitates the identification of the history, significance, and characteristics of each batik pattern currently in use. This problem can be overcome by applying deep learning to help identify batik patterns throughout the archipelago. The key conclusion is that a deep learning strategy is necessary, which involves training with extensive visual data. The methods used include deep learning by utilizing the Inception-ResNetV2 architecture. Its contribution is adopting an architecture that is designed to minimize the number of parameters and improve computational performance. The network can perform more efficiently overall when combined with residual connections from ResNetV2. Various types of batik and classes from the archipelago form the dataset. After the calculation was carried out for 9 min and 6 s, a batik pattern model was obtained with an average accuracy of 98.19%, precision of 98.20%, recall of 98.19%, and F1-Score of 98.16%. Mean Squared Error (MSE) 0.0023, Root Mean Squared Error (RMSE) 0.0483, Mean Absolute Error (MAE) 0.0035. The experimental data were then used to test the confidence level, achieving an average accuracy of 76–99%.**

***Keywords*—batik pattern, deep learning, Inception-ResNetV2, data augmentation, process innovation, product innovation**

## I. Introduction

According to the Ministry of Tourism and Creative Economy, batik's appealing variety of motifs and patterns makes it an attractive advertising medium for Indonesian tourism. Emphasizing the distinctive motifs of each location is one method to preserve batik and boost tourism in Indonesia. The distinctiveness of the batik motifs created by artisans may also draw tourists to the batik's original location [1].

Batik has strong cultural ties to Indonesia and has emerged as an icon and a means of expressing national identity. The idea of tipping points—where small adjustments can produce significant and inevitable effects—is explored through descriptive methodology. This study focuses on narratives that have been created and perpetuated in the minds of people living in societies that have undergone historical change. This system can identify potential motivations, impacts, feedback loops, enabling circumstances, key actors, and interventions that lead to change, critical and inevitable changes, or critical moments by identifying key points in these narratives. The study's findings indicate that the introduction of batik into Indonesian culture, beginning around the end of the 19th century, marked a significant turning point in history. Understanding the complex history of Indonesian batik's transformation into a national symbol is possible by identifying key moments and examining the events that led up to and followed them. Small adjustments can have significant effects [2].

The United Nations Educational, Scientific and Cultural Organization (UNESCO) has recognized Indonesia's centuries-old handicraft of batik weaving as one of the most significant examples of cultural heritage. Batik has been passed down through the ages in Indonesia, and praise for its intricate themes and designs has come from around the world. Classifying batik patterns, based on elements such as production method, cultural history, and symbolic meaning, is gaining popularity. The topic explores batik over time and across various cultural contexts, as well as its historical and cultural significance in Indonesia. Accurate batik categorisation is achieved using deep learning (Convolutional Neural Network (CNN) and Efficient-Net) and fundamental machine learning methods. The results showed that the enhanced version of EfficientNet

Manuscript received May 23, 2025; revised July 4, 2025; accepted August 20, 2025; published January 8, 2026.

trained with an accuracy of 93.81%. This study also examines the value of batik festivals and museums in promoting travel to Indonesia. According to this article, the classification of batik could have a significant impact on the Indonesian economy and tourism. Indonesia can increase tourism and boost its economy by promoting and protecting its rich cultural heritage, particularly its batik legacy [3].

Some Indonesians are unable to recognize the names of the batik motifs they see or wear, despite it being one of the country's most revered cultural traditions. Research shows that the greater the number of batik variants, the harder it becomes to distinguish between batik designs. It makes automatic batik classification even more crucial in assisting the general public in identifying batik patterns. Additionally, the historical significance of batik patterns makes them essential to understand. To recognize batik, researchers employ the CNN, VGG-16 and VGG-19 batik classification methods to design an automatic system that can predict the correct identification of batik patterns with over 90% accuracy. The classifier cannot accurately determine the type of batik motif in photos that have been altered, such as those that have been magnified or rotated. Batik motifs are classified on a scale of 2.0, with a classification accuracy of less than 56%. The CNN is trained using more data to improve accuracy. The Rotated or zoomed photos and augmented data techniques can improve accuracy by 10% [4].

A common issue in picture classification tasks is dataset imbalance. This issue also affects batik pattern data, primarily due to the low quality of the available photographs and the scarcity of specific patterns. Current research employs advanced augmentation and oversampling methods to address unbalanced datasets, effectively tackling this issue in their study. The image becomes more diverse in color, contrast, wrinkles, and curves—all of which can be found in batik apparel. Three different training approaches and two CNN models—DenseNet169 and VGG-16—were used in this investigation. These methods consist of training with oversampling, training with advanced augmentation and oversampling, and training with both advanced augmentation and oversampling. The results show that DenseNet169 works best with our oversampled and supplemented datasets, with an accuracy of 84.62%. Furthermore, VGG-16 achieved an accuracy of 82.56% on the dataset, showing strong performance. Our findings demonstrate that the model performs better when oversampling and advanced augmentation are applied to the dataset, compared to when simple and oversampled data are used [5].

In Indonesia, batik is characterized by a variety of designs arranged in a repetitive pattern to convey the overall primary theme of the cloth. Various batik periodicals, the internet, and the direct use of a digital camera are sources used to gather different types of batik themes. There is still room for improvement in the automatic classification of batik patterns, particularly concerning rotation and scale invariance. To facilitate image classification, this dilemmatic shift in batik patterns also necessitates a reliable feature extraction approach. Several windows, including 6×6, 9×9, 12×12, and 15×15 windows, or a mixture of these windows, are used in this method, which is known as MU2ECS-LBP (local binary symmetric multiwindow and multiscale extended center pattern). Researchers use k-Nearest Neighbor (k-NN) and Artificial Neural Network (ANN) in a batik classification system to automatically recognize batik patterns. Following a series of tests, the accuracy value results obtained with the k-NN method on multiwindow 6-9-12-15 and other multiwindow 6-12-15 with overlapping images of 40 and 50 pixels, and the number of image classes for grades 5, 9, and 12 were 99.91% and 99.8%. These results were based on the influence of the training image and image conditions. The multiwindow 6-9-12-15 with the ANN 64-240-12 architecture has the highest accuracy value. The multiwindow effect and image overlap are the foundations of the ANN technique. The accuracy percentage is 98.43%. The novel method consistently yields high classification accuracy even though it is a development of the local binary pattern [6].

The author is interested in applying deep learning convolutional neural networks to approach the problem of batik classification, as indicated by the literature review above. This is because batik cultural heritage preserves and maintains cultural identity, encouraging learning about local traditions and customs. The gap with previous research is that this study focuses more on a concise architecture with optimal results. The weakness of prior research is the need to compile ground truth, while in deep learning, ground truth is prepared through an in-depth training process to obtain a model with high accuracy. The significance of this research lies in the use of the Inception-ResNetV2 architecture, which combines elements of Google's Inception and ResNetV2. Combining the two is expected to improve accuracy, reduce computational burden, solve the vanishing gradient problem by shortening training time, and improve convergence. In real-time use of this application, it is hoped that it can be used as a batik area entry identifier.

## II. Literature Review

### A. Introduction to Batik Patterns

In Southeast Asia, Malaysia and Indonesia are renowned for their prolific batik production. For one century, the histories of batik have been intricately linked between these two nations. Published works on batik production, innovations, and issues are examined within each nation's indigenous batik. This study aims to determine the extent to which batik has evolved as a creative sector in various nations since its inception to the present day, with a specific focus on the 21st century. Batik craftsmanship has endured remarkably well, as both countries continue to utilize comparable tools and techniques today [7]. The design and stylisation of batik designs have advanced significantly with the introduction of computer techniques, such as fractal geometry. The two nations' shared issues are emphasised and divided into internal and external problems. Integrating technologies from the Third Industrial Revolution (IR 3.0) primarily focuses on utilising computer-aided design and manufacturing to enhance the current batik output [8]. A recent study demonstrates how incorporating Fourth Industrial Revolution (IR4.0) technology, such as Augmented Reality (AR), can enhance batik-making skills and increase awareness of batik as an intangible cultural heritage. The lack of exposure to batik among younger generations remains a concern. Therefore, a concise framework is presented, along with examples of how IR4.0 technologies can be creatively applied to disseminate knowledge about batik through learning pathways. Motifs contain local philosophy and history; therefore, given their geographical origin, the model not only recognises the motif but also acknowledges where and why it emerged. This also maintains cultural context and prevents oversimplification by Artificial Intelligence (AI) [9].

### *B. Deep Learning*

A particular kind of machine learning is referred to as "deep learning". To evaluate and comprehend data, it utilizes artificial neural networks, which are neural networks equipped with sophisticated learning algorithms [10]. It extracts features using a CNN that utilizes a convolutional lattice [11]. The result of this method includes:

#### *1) Convolutional layer*

The basic process of CNN, a convolution, creates a feature map by applying a filter (kernel) to the input according to Eq. (1).

$$y(i,j) = (x \times h)(i,j) = \sum_m \sum_n x(i-m, j-m) \cdot h(m,n) \quad (1)$$

where: $y(i,j)$ =The pixel value of the convolution at position $(i,j)$ ; $x(i,j)$ =The pixel value of the input image at position $(i,j)$ ; $h(m,n)$ = The convolution kernel or filter of size $(m,n)$ ; $(i,j)$ =The position coordinates in the resulting image; $(m,n)$ = The position coordinates in the kernel [12].

#### *2) Padding*

The output size of the convolution layer is controlled by it. It is possible to use several forms of padding, as well as zero padding. If the padding of $p$ is applied, then the output size (if the input is of size $W \times H$ and the filter is of a size $F \times F$) [13]. The equation for padding is shown in Eqs. (2) and (3).

$$W_{out} = \frac{W - F + 2p}{S} + 1 \quad (2)$$

$$H_{out} = \frac{W - F + 2p}{S} + 1 \quad (3)$$

where: $s$ = a stride (shifting for filter); $p$ = padding; $W_{out}$ = width of image; $H_{out}$ = height of image; $F$ = Filter [14].

#### *3) Stride*

It is the number of steps the filter performs on the input. If $S$ steps are applied, the output size (without padding) is shown in Eqs. (4) and (5).

$$W_{out} = \frac{W - F}{S} + 1 \quad (4)$$

$$H_{out} = \frac{H - F}{S} + 1 \quad (5)$$

where: $S$ = Stride, $W_{out}$= Width of image, and $F$ = Filter [15].

#### *4) Pooling layer*

Pooling layers decrease the feature map's spatial dimension. The formula illustrates that there are two main types: pooling and average pooling, as shown in Eq. (6).

$$y(i,j) = \frac{1}{|R(i,j)|} \sum_{(m.n) \in R(i.j)} x(m,n) \quad (6)$$

where: $R(i, j)$ = Pooling Area; $y(i, j)$= Output Pooling; $x(m, n)$ = Input [16].

#### *5) Batch normalization*

It is utilized to accelerate training and increase model stability by normalizing the output of the preceding layer, as shown in Eqs. (7) and (8).

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} \quad (7)$$

$$y = \gamma \hat{x} + \beta \quad (8)$$

where: $\mu$ = mean, $\sigma^2$ = variants, $\varepsilon$ = small value to prevent division by zero, $\gamma$ and $\beta$ = parameters that can be studied [17].

#### *6) Rectifier linear unit*

The purpose of this function is to add non-linearity to the model in artificial neural networks. It functions by maintaining the positive input values while setting all negative input values to zero, as shown in Eq. (9).

$$f(x) = \begin{cases} 0, & if\ x > 0 \\ x, & if\ x \geq 0 \end{cases} \quad (9)$$

where: $x$ = input, and $F(x)$ = ReLU function. The ReLU function helps to overcome the problem of vanishing gradients or loss of information during the training process [18].

#### *7) Fully connected layer*

The output is flattened and sent to the fully linked layer following multiple layers of convolution and pooling, according to some reports. It is shown in Eq. (10).

$$y = \sigma(Wx + b) \quad (10)$$

where: $W$ stands for weight, $x$ for input, $b$ for bias and $\sigma$ for activation function (e.g., sigmoid, ReLU) [19].

*8) Softmax layer*

It transforms the final layer's output into a probability distribution in neural networks. Its formula is shown in Eq. (11).

$$\sigma(z)_i = \frac{e^Z i}{\sum_{i=1}^{C} e^Z j} \tag{11}$$

where: $\sigma(z)_i$ = Probability of $i$-Class; $z_i$ is the raw score (logit) for class-$i$ produced by the neural network; $C$ = number of $i$-Class [20].

*C. Confusion Matrix*

In Multiclass classification, a table evaluates the performance of the classification algorithm. This matrix provides a breakdown of the model's correct and incorrect predictions for each class [21]. The Multiclass confusion matrix is shown in Table I.

TABLE I. CONFUSION MATRIX

| **Class** | | **Prediction** | | | |
|---|---|---|---|---|---|
| | | **A** | **B** | **C** | **D** |
| **Actual** | A | TN | FP | TN | TN |
| | B | FN | TP | FN | FN |
| | C | TN | FP | TN | TN |
| | D | TN | FP | TN | TN |

where: TN = True Negatif, FP = False Positif, FN = False Negatif, TP = True Positif.

- Accuracy states the total proportion of correct forecasts out of all predictions.
- Out of all the occurrences projected as belonging to a class, the precision for each class indicates the percentage of accurate optimistic forecasts.
- The percentage of accurate optimistic predictions for a class out of all actual instances of that class is known as recall (sensitivity) for each class.
- Each class's F1-Score indicates the class's harmonic average of precision and recall [22].

*D. Value of Error Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE)*

*1) Mean Squared Error (MSE)*

The average squared error between the expected and actual values is measured by MSE. The discrepancy between the model's suggested value and the actual value is known as error. The formula is shown in Eq. (12).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 \tag{12}$$

where: $n$ = number of observations, $y_i$ = actual value, and $\hat{y}_i$ = prediction value [23].

*2) Root Mean Squared Error (RMSE)*

The square root of MSE is known as RMSE. Compared to MSE, RMSE is simpler to understand because it uses the same units as the projected value. It is shown in Eq. (13).

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{13}$$

where: $n$ = number of observations, $y_i$ = actual value, and $\hat{y}_i$ = prediction value [24].

*3) Mean Absolute Error (MAE)*

*MAE* is the average of the absolute values of the error between the predicted value and the actual value. Unlike MSE, MAE does not magnify the influence of significant errors, thus providing a more direct picture of the average error. It is shown in Eq. (14).

$$MAE = \frac{\sum_{i=1}^{n}\left|y_i - x_i\right|}{n} \tag{14}$$

where: $n$ = number of observations, $y_i$ = actual value, and $x_i$ = prediction value.

MAE is more straightforward to interpret because it shows the average absolute error. It doesn't magnify significant errors such as MSE and RMSE, so it can be fairer if considerable mistakes are not so critical [25].

*4) Training optimizer*

Reducing a cost or loss function is the primary goal of gradient descent. Iteratively, it modifies model parameters (such as a neural network's weights) to reduce the loss value, as shown in Eq. (15).

$$w_{t+1} = w_t - \eta \nabla L(w_t) \tag{15}$$

where: $w_t$ = weight on iteration-$t$, $w_{t+1}$ = weight on iteration-($t$+1), $\eta$ = learning rate, $\nabla L$ is the Gradient of the loss on the weights $w_t$ [26].

*E. Data Augmentation*

Implementing different changes or transformations to the initial data without altering the data's labels or contents [27].

*1) Contrast adjustment*

As shown in Eq. (16), enhancing the prominence of features in an image or reducing their visual intensity can make the machine learning model more robust to variations in lighting.

$$I' = \alpha \cdot (I - \mu) + \mu \tag{16}$$

where: $I$ is input image, $I'$ is adjusted image, $\mu$ is the mean intensity of the input image (calculated as the mean of all pixel values), $\alpha$ is contrast adjustment factor ($\alpha > 1$: increases contrast, $\alpha < 1$: decreases contrast) [28].

*2) Saturation tweaks*

Changing the intensity or strength of color in an image without changing its geometric components, as shown in Eq. (17).

$$I' = (1 - \beta) \cdot Igray + \beta \cdot I \tag{17}$$

where: $I$ is input image (in RGB format), $Igray$ is Grayscale version of the input image, calculated as: $Igray = 0.2989 \times R + 0.5870 \times G + 0.1140 \times B$ , $\beta$ is saturation adjustment factor ($\beta > 1$: increases saturation, $\beta < 1$: decreases saturation), $I'$ is adjusted image [29].

*3) Geometric transformations*

Modifying the geometric structure of an image or visual dataset involves changing the position, orientation, size, or shape of objects in an image without significantly altering the pixel values. Scaling is one technique, as shown in Eq. (18).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = s \times \begin{bmatrix} x \\ y \end{bmatrix} \tag{18}$$

where: $s$: scaling factor ($s > 1$: enlarges, $0 < s < 1$: shrinks), $(x, y)$ is the original pixel coordinates, and $(x', y')$ is scaled coordinates. Other implementations are shearing, rotation, translation, and combining augmentation [30, 31].

## III. Materials and Methods

The procedure begins by preparing the input image for training. To do this, the similarity of the training data format needs to be checked, as it was collected using different cameras. Different horizontal and vertical sizes are available, regardless of the file format (JPG, PNG, or JPEG). Sometimes, raw and high-resolution files require special handling. After the input image meets the standards, it proceeds with preprocessing. The use of pre-training weights aims to leverage knowledge already obtained from millions of images covering thousands of classes, thereby accelerating the training process and enhancing accuracy, especially when the new dataset is of limited size. K-Fold is a method of division that can be done by determining the portion of the training data or by setting the number of files. To split the training data. Following the selection of the architecture to be employed, training is conducted with 80% of the data, validation data at 15%, and test data at 5%. The hardware and software implementation used is Google Collab, with a laptop specification of an Intel i7 processor-based processing, a RAM capacity of 12 GB, and a NVIDIA GTX 1050 GPU, and Google Drive for permanent storage.

### *A. Inception Block*

The primary objective of the Inception is to enhance computational efficiency and image recognition performance by combining multiple filter sizes within a single block. The STEM block comprises several convolutional layers with pooling, designed to gradually reduce the input space size while increasing the complexity of the extracted features. The architecture used is shown in Fig. 1.

The task of 5×Inception-ResNet-A, shown in Fig. 2, is to extract visual features from images deeply and efficiently by combining the advantages of the Inception architecture and residual connections. The first block of the Inception-ResNet network, Inception-ResNet-A, operates at a high spatial resolution and is responsible for identifying local patterns in images, including edges, textures, and fundamental structures.

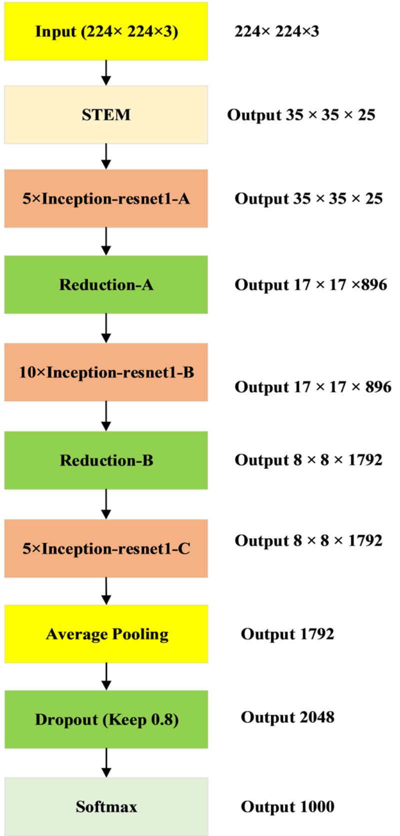


Fig. 1. Inception block diagram.

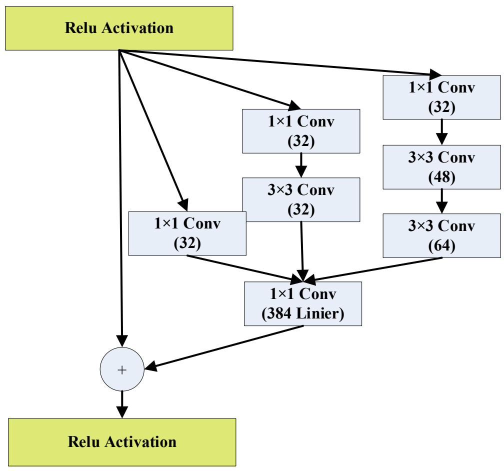


Fig. 2. The 5×Inception-ResNet1-A.

ResNet optimizes the training process of deep network models. This is achieved through residual connections, shortcuts that allow information and gradients to flow directly through multiple layers, as shown in Fig. 3.

By significantly reducing the vanishing gradient issue, this technique enables the network to add more layers without compromising stability or performance during training. Thus, even though the amount of computation remains large, the learning process becomes more efficient and converges faster.

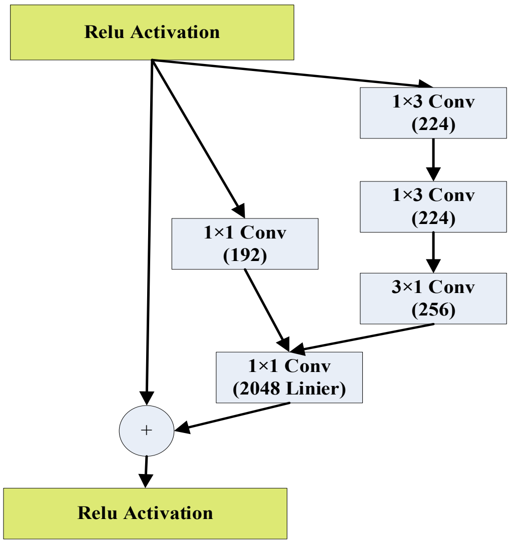


Fig. 3. The 5×Inception-ResNet1-C.

By repeating this block five times, the network can deepen its understanding of features without drastically increasing the number of parameters. The task of 5×Inception-ResNet-C is to extract high-level features from images while maintaining computational efficiency and training stability. Inception-ResNet-C is utilized in the final stage of a deep learning network, where the feature map size is already small, but the information complexity is high.

## B. Dataset

In this study, the batik motif data used are secondary data from Kaggle.com, with the link address available in the "Data Access" section. The fifteen classes consist of the following: Cendrawasih, Parang, Insang, Kawung, Megamendung, Geblek Renteng, Ikat Celup, Lasem, Pala, Dayak, Bali, Tambal, Poleng, Betawi, and Sekar Jagad. The batik data consists of batik motif images from several cities in Indonesia. The data is taken in various formats, including JPG, PNG, and JPEG. The original photos from each class numbered 70 images, totaling 1,050 images of varying sizes, which were taken with cameras and cellphones. After that, the fifteen classes are expanded to include 330 photos each, for a total of 4,950 training images. These photos are separated into three sets: 5% for testing, 15% for validation, and 80% for training.

# IV. Results and Discussion

## A. Preprocessing

To standardize the entire file format used in the training process, preprocessing is carried out, as shown in Fig. 4.

To the exact resolution makes the data more uniform. This reduces undesired dataset variances that may impact model performance. By preventing the model from learning from elements that are overly unique to the original image's various sizes, it also lessens the chance of overfitting.

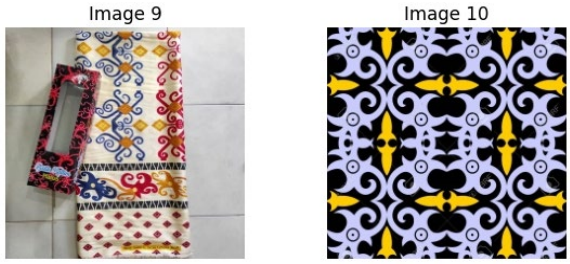


Fig. 4. Resizing the images into 224×224×3.

## B. Data Augmentation

It used to expand a dataset's size and variety without requiring the collection of new information. Augmentation introduces variance into the data, making the model more resilient and generalizable, thereby reducing the likelihood of overfitting and enhancing performance on previously unseen data. As shown in Fig. 5(a), additional data variations can be added, but only as needed, and the image remains clear. Shears and reflections are one part of geometric transformations.

The techniques used:

- Rotate the image by a certain angle, as shown in Eq. (19).

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{19}$$

where $\theta$ is angle of rotation [32].

- Translation is shifting an image to a specific position.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{20}$$

where: $t_x$ and $t_y$ are the sliding distance in the horizontal and vertical directions as shown in Fig. 5(b). The setting used in Python is rescale = 1/255, rotation_range = 20, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'nearest'.

## C. Training Process

The data is divided into training and testing data. The initial stage of parameter initialisation encompasses various factors, including the optimisation method used, learning rate, number of iterations, minimum epoch, validation set, and early stopping, among others. The training process in Fig. 6 requires computing time, which depends on the device and memory capacity. The setting of the hyperparameter is shown in Table II.

TABLE II. Hyperparameter Setting

| Hiperparameter | Set 1 (Before tuning) | Set 2 (After tuning) |
|---|---|---|
| Learning rate | 0.001 | 0.0001 |
| Batch size | 64 | 32 |
| Optimizer | SGD | Adam |
| Trainable layers | 0 | Top 50 layers |
| Epochs | 10 | 30+ early stopping |
| Data Augmentation | No | Yes |
| Dropout (classifier) | 0.0 | 0.5 |
| Accuracy | 96.80 | 98.19 |


Fig. 5. Augmentation with (a) scale and rotation, (b) shears and reflection.

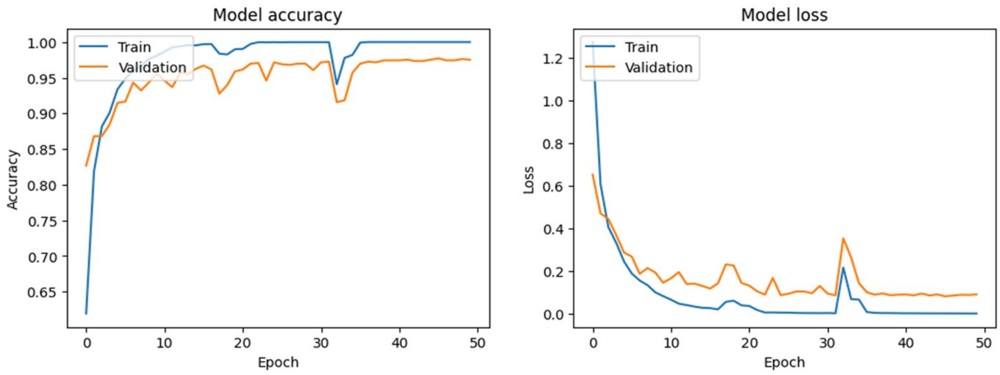


Fig. 6. Training process using Inception-ResNetV2.

Previous academic discoveries and experimental results demonstrating enhanced model generalization and stability served as the impetus for the hyperparameter changes from Set 1 to Set 2. Since lower learning rates have been shown to stabilize transformer-based model training, the learning rate was lowered from 0.001 to 0.0001 to avoid unstable updates and overfitting [33]. In line with research showing that smaller batch sizes can enhance generalization, the batch size was lowered from 64 to 32 to improve gradient estimation and enable better convergence in fewer datasets [34]. Adam, which offers an adjustable learning rate and momentum correction, and is frequently used in vision transformers training because of its quicker convergence, replaced SGD as the optimization approach [35]. By increasing the number of trainable layers from zero (frozen backbone) to fifty top layers, high-level representation refinement was possible while still making use of pre-trained weights. To provide the model with additional learning chances and prevent overfitting, epochs were extended from 10 to 30 with early stopping.

Results in Best hyperparameter tuning is Experiment 6 (LR = 0.00005, Epochs = 75, Dropout = 0.5), then best accuracy (92.4%) and the lowest loss (0.29). Effect of the Learning Rate is smaller learning rate will be smoother convergence and better accuracy. The effect of Batch Size is moderate batch size (32) tends to generalise better. Effect of Dropout: 0.5 helps regularize and avoid overfitting [36]. It shows in Table III.

TABLE III. HYPERPARAMETER TUNING

| Experiment ID | Learning Rate ($\eta$) | Batch Size | Dropout Rate | Epochs | Augmentation | Validation Accuracy (%) | Validation Loss |
|---|---|---|---|---|---|---|---|
| Exp-1 | 0.001 | 32 | 0.5 | 50 | Medium | 87.2 | 0.42 |
| Exp-2 | 0.0005 | 32 | 0.5 | 50 | Medium | 89.5 | 0.38 |
| Exp-3 | 0.0001 | 32 | 0.5 | 50 | Medium | 91.3 | 0.31 |
| Exp-4 | 0.0001 | 16 | 0.5 | 50 | Medium | 88.6 | 0.36 |
| Exp-5 | 0.0001 | 64 | 0.3 | 50 | Strong | 90.1 | 0.33 |
| Exp-6 | 0.00005 | 32 | 0.5 | 75 | Strong | 92.4 | 0.29 |

### *D. Confusion Matrix*

A classification model's performance is evaluated using this method, which measures explicitly how well the model predicts labels that match the actual labels. The validation data used in the training procedure is shown in the diagonal confusion matrix.

The other box contains the number of files that match the class in each class. However, if a class is wrong, the number of files will appear in a horizontal or vertical box. Errors can occur if there is a similarity between the prediction and the actual files. This can cause class accuracy to decrease, making visual differences necessary in each class. Building a model with validation data is demonstrated by the confusion matrix in Fig. 7. The outcomes are as follows: Accuracy: 0.9819, Weighted Precision: 0.9820, Weighted Recall: 0.9819, and Weighted F1-Score: 0.9816. The Mean Absolute Error (MAE) is 0.0035, the Mean Squared Error (MSE) is 0.0023, and the Root Mean Squared Error (RMSE) is 0.0483.

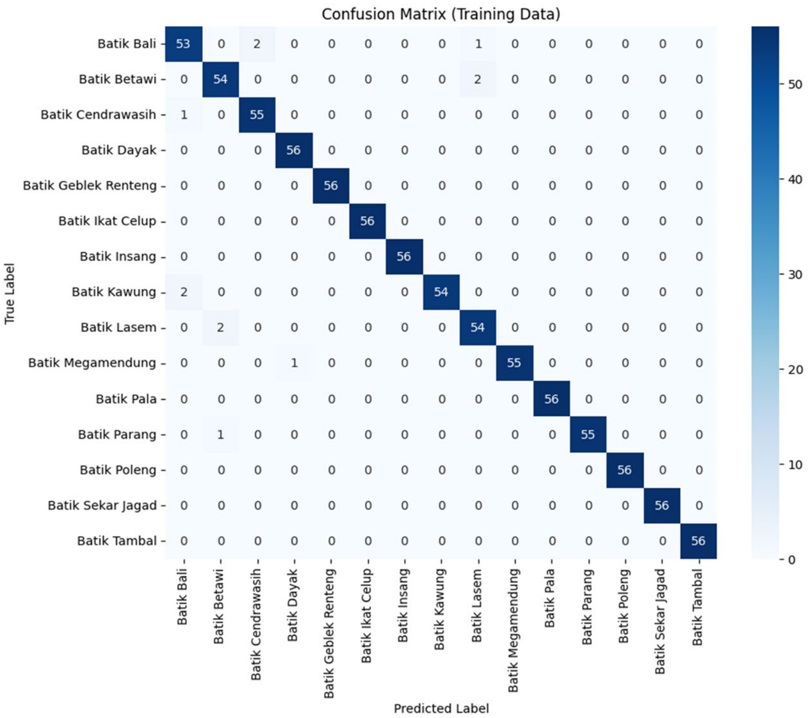


Fig. 7. Confusion matrix model.

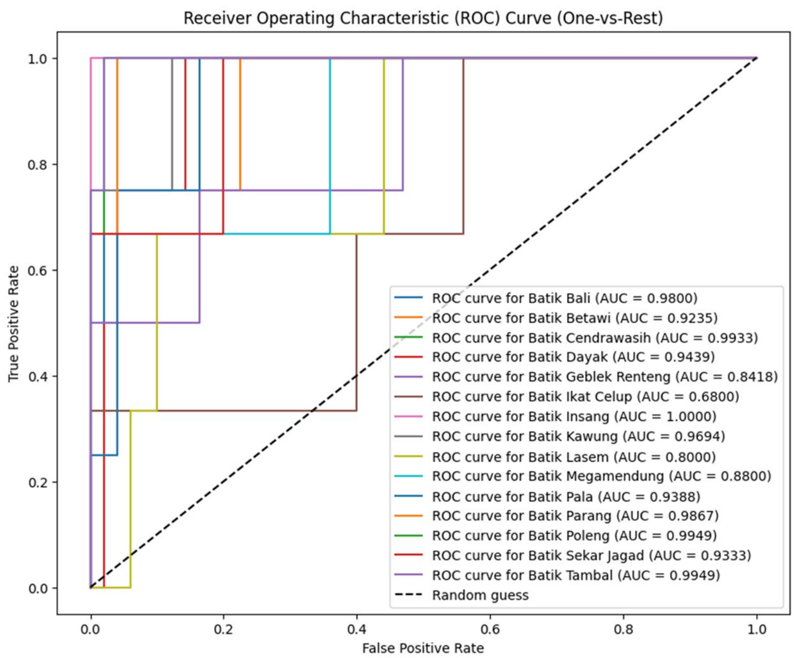


Fig. 8. Area Under Curve (AUC).

The model's capacity to differentiate between the positive and negative classes is demonstrated by the Area Under Curve (AUC) (Area Under the ROC Curve), as shown in Fig. 8. Excellent discriminative capability is indicated by a high AUC value, which approaches 1.0. This means that the model is successful in accurately distinguishing genuine positives and true negatives. On the other hand, an AUC nearer 0.5 indicates that the model's performance is no better than that of random guessing. The outcome shows that the AUC value is approximately 0.9800.

### *E. Prediction/Testing*

By contrasting the prediction outcomes on the test data with the model's performance during training, belief accuracy characterises the degree of confidence in the prediction. The prediction on data testing is shown in Figs. 9 and 10. The higher the confidence level, the more the prediction results match the patterns discovered during the training phase, suggesting that the model is not only accurate on the training data but also trustworthy on fresh data.


Fig. 9. Prediction of experimental test data 01.

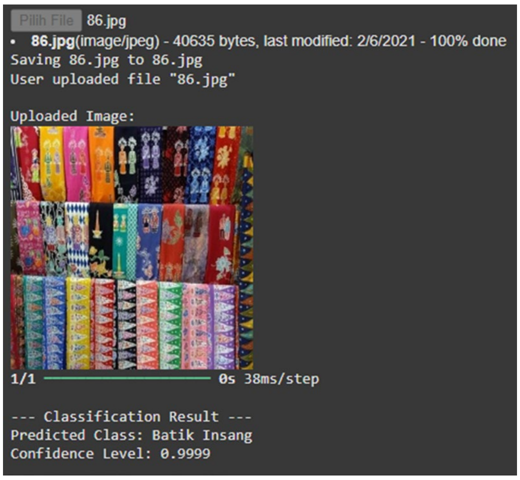


Fig. 10. Prediction of experimental test data 02.

### *F. K-Fold Validation*

K-fold validation analysis is very useful for minimizing evaluation bias that may occur if only one test dataset is used. A K-Fold validation scenario was used in the trial, and Table IV displays the findings. The model performs steadily and consistently across all data subsets, as indicated by the K-Fold study. By performing validation K times, each data subset has the opportunity to become the test data, thus making the model evaluation more comprehensive and unbiased towards certain data distributions. Evaluation measures, including accuracy, precision, recall, and F1-Score, in the context of the results, exhibit high and balanced average values of approximately 98.19%, with comparatively slight variances between folds. This suggests that the Inception-ResNetV2 model can effectively generalize to unseen data. The difference between the 10-Fold vs the random split is shown in Table V.

TABLE IV. COMPARISON OF K-FOLD VALIDATION

| Fold | Training Set | Validation Set | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | MSE | RMSE | MAE |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 4455 | 495 | 98.23 | 98.01 | 98.12 | 98.06 | 0.0177 | 0.1330 | 0.1052 |
| 2 | 4455 | 495 | 98.12 | 97.95 | 98.02 | 97.98 | 0.0188 | 0.1372 | 0.1090 |
| 3 | 4455 | 495 | 98.26 | 98.10 | 98.20 | 98.15 | 0.0174 | 0.1318 | 0.1040 |
| 4 | 4455 | 495 | 98.15 | 97.99 | 98.00 | 97.98 | 0.0185 | 0.1360 | 0.1073 |
| 5 | 4455 | 495 | 98.34 | 98.21 | 98.24 | 98.22 | 0.0166 | 0.1290 | 0.1025 |
| 6 | 4455 | 495 | 98.10 | 97.87 | 97.95 | 97.91 | 0.0190 | 0.1378 | 0.1100 |
| 7 | 4455 | 495 | 98.22 | 98.05 | 98.14 | 98.09 | 0.0178 | 0.1335 | 0.1060 |
| 8 | 4455 | 495 | 98.19 | 98.00 | 98.09 | 98.03 | 0.0181 | 0.1345 | 0.1068 |
| 9 | 4455 | 495 | 98.09 | 97.88 | 97.97 | 97.91 | 0.0191 | 0.1382 | 0.1102 |
| 10 | 4455 | 495 | 98.20 | 98.02 | 98.11 | 98.05 | 0.0179 | 0.1339 | 0.1061 |
| Average | | | **98.19** | **98.01** | **98.08** | **98.04** | **0.0179** | **0.1345** | **0.1067** |

TABLE V. COMPARISON 10-FOLD CV VS RANDOM SPLIT

| Protocol | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | AUC (%) | Train Time/fold (min) | Total Train Time (min) |
|---|---|---|---|---|---|---|---|
| 10-Fold CV (mean±std) | 93.1±1.2 | 92.8±1.3 | 92.5±1.4 | 92.6±1.2 | 96.4±0.8 | 7.8±0.6 | ±78 |
| Random Split (80/20)×5 | 94.0±0.6 | 93.7±0.7 | 93.3±0.8 | 93.5±0.7 | 96.9±0.5 | 9.5±0.4 | ±9.5 |

### G. Comparison of Inception and ResNet

ResNet50 outperformed InceptionV3 by a little margin on all evaluation metrics. Comparison is shown in Table VI. ResNet50 was marginally faster, but the training time for each epoch was essentially the same. These findings suggest that ResNet outperforms Inception in terms of feature representation of batik motifs.

TABLE VI. COMPARISON INCEPTION VS RESNET 50

| Model | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) | Training Time (s/epoch) |
|---|---|---|---|---|---|
| InceptionV3 | 89.4 | 88.7 | 87.9 | 88.3 | 52 |
| ResNet50 | 91.2 | 90.8 | 90.1 | 90.4 | 48 |

### H. Handle Overfitting

Cross-Categorical cross-entropy, a loss function frequently utilized for multiclass classification applications, is employed when training a batik image classification model with the Inception-ResNet-v2 architecture. This function calculates the probability difference between the model's predictions and the actual label distribution. To improve model generalization and avoid overfitting, a variety of regularization techniques are used. To prevent the model from becoming too complicated, one approach is L2 regularization (weight decay), which penalizes substantial model weights. Additionally, dropout is used to randomly deactivate neurons during training in many fully connected layers with a specific dropout rate, thereby avoiding over-reliance on particular attributes. Regularization strategies and the proper choice of loss function are essential for preserving training stability and enhancing model performance on untested data.

### I. Comparison between K-Fold vs Random Split

Assessed Inception-ResNetV2 using the same augmentation under two validation protocols: repeated random splits (80/20, five runs) and 10-fold cross-validation. The 10-fold CV produced a conservative and reliable estimate across folds with an accuracy of 93.1±1.2% and an F1-Score of 92.6±1.2%. The accuracy and F1-Score from repeated random splits were 94.0±0.6% and 93.5±0.7%, respectively. These results were somewhat higher on average but more susceptible to data partitioning. These findings imply that a 10-fold CV more accurately represents expected generalization on unknown data at the expense of more training time, even while a random split may exaggerate performance for fortunate splits.

### J. Comparison with Other Architectures

Compared to other deep learning architectures such as VGG16, ResNet50, or plain InceptionV3, the Inception-ResNetV2 model demonstrates superior performance for batik pattern classification because it combines the strength of Inception modules in capturing multi-scale features with the residual connections of ResNet that improve gradient flow and training stability. While VGG16 is simpler and easier to train, it lacks the depth and efficiency needed to capture the intricate textures and motifs of batik. ResNet50 provides good generalization but may miss fine-grained local features compared to Inception structures. InceptionV3, although effective in extracting multi-scale information, is less efficient in intense networks without residual shortcuts. Thus, Inception-ResNetV2 offers a balanced trade-off between accuracy, convergence speed, and robustness, making it more suitable for complex cultural pattern recognition tasks. The comparison with other architectures is shown in Fig. 11. The batik pattern test scenario shows that Inception ResNet is not the best. However, considering that this architecture is derived from Google's Inception, it offers improvements over pure Inception-ResNetV2, which requires significantly longer computation times. The addition of ResNet skip connections improves Inception's computation time, resulting in shorter computation times.

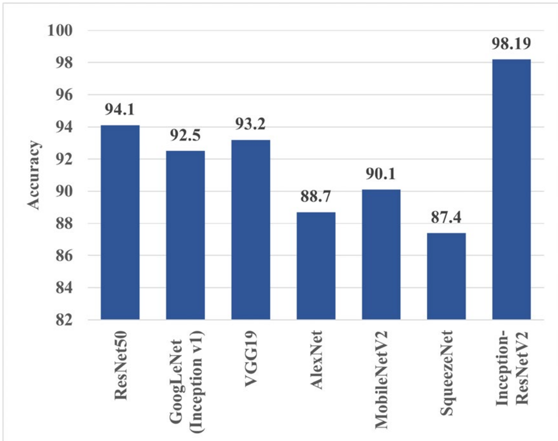


Fig. 11. Architectural comparison for the batik dataset.

### K. Research Comparison on Batik Pattern

One issue with comparing related studies on batik patterns in the publications is that not all of them display the computing time required to learn the model. The utilization and variations in hardware specs or usage are the cause of this. A comparison of the accuracy of numerous comparable investigations is presented in Table VII. The final model outperforms other researchers by a small margin. Additionally, taking into account the benefits and drawbacks of each architectural method, the average accuracy results are also impacted. According to the final results, the average accuracy of inception_ResNetV2 in the tested situation is 98.19%.

TABLE VII. COMPARATIVE OF BATIK PATTERN RESEARCH

| No | Authors | Method / Model | Accuracy (%) | Advantages | Limitations |
|---|---|---|---|---|---|
| 1 | Rasyidi and Bariyah [37] | CNN (DenseNet architecture) | 94% (journal.beei.org) | Good performance across 6 classes of batik motifs (Banji, Ceplok, Kawung, Mega Mendung, Parang, Sekar Jagad). | -Dataset size (~994 images).<br>-DenseNet is relatively heavy; might be harder to deploy on low. |
| 2 | Utaminingsih and Sahputra [38] | CNN (EfficientNet) | 98% (Journal of IT Science) | -Effective in capturing complex Aceh batik motifs. | -The dataset has only 3 classes. Requires a powerful model & more computation.<br>-Possibly sensitive to class imbalance. |
| 3 | Sari *et al*. [39] | MobileNetV3 + transfer learning + data augmentation | ≈ 93.88% (testing) (ejournal.isha.or.id) | -Model uses transfer learning: faster training, good feature extraction.<br>-Large dataset (4,284 images), with 5 Batik classes. | Data augmentation helps, but pattern diversity (lighting, fabric, etc.) might still affect performance. |
| 4 | A-Aziz and Saefurrohman [40] | CNN (Android app with TensorFlow Lite) | ≈ 94% (testing); ~92.25% for training (Open Journal) | -Deployment as an Android app (mobile application) increases usability.<br>-Real-time detection possibilities.<br>- Smaller dataset (500). | -Mobile constraints may limit model size / computational capacity.<br>-With fewer images per class, there is a risk of overfitting. |
| 5 | Karim *et al*. [41] | Combined GLCM + CNN (EfficientNetB0) | 97% (santika.upnjatim.ac.id) | -Combining texture (GLCM) + deep visual features (CNN) yields high predictive performance.<br>-Effective even with moderate data (augmented). | -The dataset is relatively small initially (before augmentation): risk of bias.<br>-Might not generalise to motifs far outside Yogyakarta / Pekalongan styles. |
| 6 | Rosalina *et al*. [42] | CNN (EfficientNet) on mobile platform | 83% (ijai.iaescore.com) | -Targeted for real-time, mobile usage: practical deployability.<br>-Covering 15 batik types (larger number of classes).<br>-Good first step for mobile/field applications. | - Lower accuracy compared to many heavy-model based studies. - More classes → more confusion / harder classification.<br>-Mobile constraints (memory, processing) likely limit model capacity. |
| 7 | This study | Inception-ResNetV2 + 10-Fold CV | 98.19% | -High accuracy;<br>-Rich, deep features.<br>-Strong transfer learning. | -Computational cost.<br>-Model complexity. |

### L. *Comparative Research Impact of Batik Management with Deep Learning Classification on Tourist Visits*

By leveraging deep learning techniques to automatically classify batik patterns based on region, style, and historical elements, local governments and cultural institutions can create more intelligent systems for archiving and promoting traditional textiles. The effect on tourism research is shown in Table VIII.

TABLE VIII. COMPARATIVE ANALYSIS OF BATIK PATTERNS IN TOURISM RESEARCH

| No | Authors | Tittle | Indicator | Result |
|---|---|---|---|---|
| 1 | Bhatia *et al*. [43] | Innovative Applications of Deep Learning in Cultural Heritage Development and Preservation: A Customization Perspective | Using AI features to recognize batik | Yes: 75%, No: 25%. |
| 2 | Ilieva *et al*. [44] | Effects of Generative AI in the Tourism Industry | Satisfaction with the feature | Beneficial: 55%, Helpful: 35%, Not very helpful: 7%, Not helpful at all: 3%. |
| 4 | Alyasiri *et al*. [45] | A Survey on the Potential of Artificial Intelligence Tools in Tourism Information Services | Recommend features to others | Yes: 85%, No: 15%. |
| 5 | Bi *et al*. [46] | Innovative Approaches to Preserving Intangible Cultural Heritage through AI-Driven Interactive Experiences | Increased appreciation of culture | Very much improved: 65%, Improved: 25%, Not much improved: 8%, Not many improved: 2%. |

This demonstrates how technology facilitates access to tourist attractions. Before visiting a tourist area, tourists' knowledge of batik motifs and their understanding of the surrounding cultural elements can be enhanced by using deep learning to categorize batik patterns. This deep learning-based model can identify a broader range of patterns thanks to a variety of augmentations. A mobile-based cultural heritage app that helps tourists learn about and recognize traditional batik designs is an example of how batik pattern classification can be used in the tourism industry. Classifying uploaded batik photos is then automatically tested for similarity, allowing visitors to gain insight into the historical and cultural significance of the patterns and identify aesthetic contrasts between them. Using data augmentation, the model is guaranteed to be robust to changes in lighting, orientation, and image quality—common issues in real-world situations when tourists use mobile phones to take pictures. This app increases visitor engagement, indirectly benefiting tourism and culture. Furthermore, it provides regional batik artists with a digital platform to advertise their work to a wider market.

## V. CONCLUSION AND FUTURE WORK

Batik motifs have been successfully identified using the developed technique. The Inception-ResNetV2 architecture serves as the foundation for a deep learning technique used in batik pattern detection. The fifteen classes consist of the following: Cendrawasih, Parang, Insang, Kawung, Megamendung, Geblek Renteng, Ikat Celup, Lasem, Pala, Dayak, Bali, Tambal, Poleng, Betawi, and Sekar Jagad. The research findings indicate that incorporating ResNet into the Inception architecture helps reduce the computational burden in the convolutional process. Additionally, the concatenation process reduces overfitting, and displaying unique batik patterns through technological advancements can attract more tourists. An average accuracy of 98.19%, precision of 98.20%, recall of 98.19%, and F1-Score of 98.16% were achieved by the model produced during the training phase using the Inception-ResNet v2 architecture. Mean Absolute Error (MAE) 0.0035, Mean Squared Error (MSE) 0.0023, and Root Mean Squared Error (RMSE) 0.048. Furthermore, the confidence level test was conducted using new data or experiments, achieving an average accuracy of 76–99%. Future research that can be developed is the integration of Inception-ResNetV2 based on Generative Adversarial Networks (GAN). The training data is improved, and model generalization is enhanced by using GAN results. Additionally, by combining qualitative and quantitative evaluations, they can be verified to ensure cultural and artistic validity. Validation involves traditional batik experts, art curators, or batik artisans who assess the suitability of motifs, colors, and symbolism of authentic cultural heritage.

## Data Access

The data is a public dataset at the following link: https://www.kaggle.com/datasets/alfanme/indonesian-batik-motifs-corak-app.

## Conflict Interest

The authors declare no conflict of interest.

## Author Contributions

B.D.S: Problem identification, drafting proposals, testing data validity, compiling software, conducting trial scenarios, validating results, and publishing. W.A.: International coordination, data validity, software testing, and coordinating students. S.B.H.: Strengthening international activities, information on research locations in Malaysia. D.A.D.: Validating software design and testing software. B.I. Students assist in fieldwork and data collection: N.P.R., V.L.S., C.A.R.Z. All authors had approved the final version.

## Funding

Grant funding for this research is through the International Collaborative Research scheme, Decree Number 299/UN46/HK.02/2024, and Contract Number of research 473/UN46.4.1/PT.01.03/RISMAN/2024.

## Acknowledgment

The researcher would like to thank the Institute for Research and Community Service of Trunojoyo University, Madura, East Java, Indonesia. Additionally, the author would like to thank the Department of Higher Education, Research, and Technology.

## Reference


[1] R. Febriani, L. Knippenberg, and N. Aarts, "The making of a national icon: Narratives of batik in Indonesia," *Cogent Arts Humanit*., vol. 10, no. 1, 2254042, 2023. doi: 10.1080/23311983.2023.2254042

[2] G. Andersson and J. Engren, "Narration, life and meaning in history and fiction," *Scand. J. Hist*., vol. 47, no. 1, pp. 62–82, 2022. doi: 10.1080/03468755.2021.2014356

[3] R. G. Tiwari, A. K. Agarwal, V. Jain, and A. Kumar, "Batik classification in Indonesia: Exploring its significance on tourism and economy," in *Proc. 2023 International Conference on Sustaining Heritage: Innovative and Digital Approaches (ICSH)*, 2023, pp. 119–124. doi: 10.1109/ICSH57060.2023.10482828

[4] I. M. A. Agastya and A. Setyanto, "Classification of Indonesian batik using deep learning techniques and data augmentation," in *Proc. 2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE)*, 2018, pp. 27–31. doi: 10.1109/ICITISEE.2018.8720990

[5] B. J. Filia, F. F. Lienardy, I. K. P. B. Laksana, *et al*., "Improving batik pattern classification using CNN with advanced augmentation and oversampling on imbalanced dataset," *Procedia Comput. Sci*., vol. 227, pp. 508–517, 2023. https://doi.org/10.1016/j.procs.2023.10.552

[6] A. H. Rangkuti, A. Harjoko, and A. Putra, "A novel reliable approach for image batik classification that invariant with scale and rotation using MU2ECS-LBP algorithm," *Procedia Comput. Sci.*, vol. 179, pp. 863–870, 2021. https://doi.org/10.1016/j.procs.2021.01.075

[7] W. J. E. Wahed, S. Azmi, H. Amin, C. Pindah, and A. Alia, "Malaysian batik, our pride: A systematic literature review," *Int. J. Acad. Res. Bus. Soc. Sci.*, vol. 12, pp. 942–955, 2022. doi: 10.6007/IJARBSS/v12-i10/14950

[8] S. I. Syed Shaharuddin *et al*., "A review on the Malaysian and Indonesian batik production, challenges, and innovations in the 21st century," *Sage Open*, vol. 11, no. 3, 2021. doi: 10.1177/21582440211040128

[9] Indarti, I. A. T. Rahayu, and L. H. Peng, "Sustainable batik production: review and research framework," in *Proc. International Conference on Research and Academic Community Services (ICRACOS 2019)* 2020, pp. 66–72. doi: 10.2991/icracos-19.2020.14

[10] B. D. Satoto, A. Yasid, B. Irmawati, M. Yusuf, M. Syarief, and S. O. Khairunnisa, "Facial expression in tourism destinations using a deep learning approach," in *Proc. 2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2022, pp. 254–259. doi: 10.1109/ISRITI56927.2022.10052854

[11] J. Chai, H. Zeng, A. Li, and E. W. T. Ngai, "Deep learning in computer vision: A critical review of emerging techniques and application scenarios," *Mach. Learn. with Appl.*, vol. 6, 100134, 2021. https://doi.org/10.1016/j.mlwa.2021.100134

[12] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual understanding of convolutional neural network-a deep learning approach," *Procedia Comput. Sci*., vol. 132, pp. 679–688, 2018. https://doi.org/10.1016/j.procs.2018.05.069

[13] G. Liu *et al*., "Partial convolution for padding, inpainting, and image synthesis," *IEEE Trans. Pattern Anal. Mach. Intell*., vol. 45, no. 5, pp. 6096–6110, 2023. doi: 10.1109/TPAMI.2022.3209702

[14] V. Andrearczyk and P. F. Whelan, "Deep learning in texture analysis and its application to tissue image classification," in *Proc. Biomedical Texture Analysis*, 2017, pp. 95–129. https://doi.org/10.1016/B978-0-12-812133-7.00004-1

[15] C. Yang, Y. Wang, X. Wang, and L. Geng, "A stride-based convolution decomposition method to stretch CNN acceleration algorithms for efficient and flexible hardware implementation," *IEEE Trans. Circuits Syst. I Regul. Pap*., pp. 1–14, 2020. doi: 10.1109/TCSI.2020.2985727

[16] J. Choi and D. Suh, "A depthwise convolutional neural network model based on active contour for multi-defect wafer map pattern classification," *Eng. Appl. Artif. Intell*., vol. 139, 109707, 2025.

https://doi.org/10.1016/j.engappai.2024.109707

[17] M. Segu, A. Tonioni, and F. Tombari, "Batch normalization embeddings for deep domain generalization," *Pattern Recognit*., vol. 135, 109115, 2023. https://doi.org/10.1016/j.patcog.2022.109115

[18] M. Zhao, S. Zhong, X. Fu, B. Tang, S. Dong, and M. Pecht, "Deep residual networks with adaptively parametric rectifier linear units for fault diagnosis," *IEEE Trans. Ind. Electron*., vol. 68, no. 3, pp. 2587–2597, 2021. doi: 10.1109/TIE.2020.2972458

[19] S. H. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of fully connected layers on performance of convolutional neural networks for image classification," *Neurocomputing*, vol. 378, pp. 112–119, 2020. https://doi.org/10.1016/j.neucom.2019.10.008

[20] Y. Gao, W. Liu, and F. Lombardi, "Design and implementation of an approximate softmax layer for deep neural networks," in *Proc. 2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5. doi: 10.1109/ISCAS45731.2020.9180870

[21] C. A. Setiawan *et al*., "Gestive: Evaluation of multi-class classification methods for gesture recognition to improve presentation experience," *Procedia Comput. Sci*., vol. 227, pp. 364–371, 2023. https://doi.org/10.1016/j.procs.2023.10.535

[22] S. Ruuska, W. Hämäläinen, S. Kajava, M. Mughal, P. Matilainen, and J. Mononen, "Evaluation of the confusion matrix method in the validation of an automated system for measuring feeding behaviour of cattle," *Behav. Processes*, vol. 148, pp. 56–62, 2018. https://doi.org/10.1016/j.beproc.2018.01.004

[23] D. G. Chachlakis, T. Zhou, F. Ahmad, and P. P. Markopoulos, "Minimum mean-squared-error autocorrelation processing in coprime arrays," *Digit. Signal Process*., vol. 114, 103034, 2021. https://doi.org/10.1016/j.dsp.2021.103034

[24] M. Nayak, S. Das, U. Bhanja, and M. R. Senapati, "Elephant herding optimization technique based neural network for cancer prediction," *Informatics Med. Unlocked*, vol. 21, 100445, 2020. https://doi.org/10.1016/j.imu.2020.100445

[25] L. Ying and L. Boqin, "Application of transfer learning in task recommendation system," *Procedia Eng*., vol. 174, pp. 518–523, 2017. https://doi.org/10.1016/j.proeng.2017.01.178

[26] R. Zhou and P. Quan, "Optimization ways in neural network compression," *Procedia Comput. Sci*., vol. 221, pp. 1351–1357, 2023. https://doi.org/10.1016/j.procs.2023.08.125

[27] A. Mumuni and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," *Array*, vol. 16, 100258, 2022. https://doi.org/10.1016/j.array.2022.100258

[28] I. S. Isa, S. N. Sulaiman, M. Mustapha, and N. K. A. Karim, "Automatic contrast enhancement of brain MR images using Average Intensity Replacement based on Adaptive Histogram Equalization (AIR-AHE)," *Biocybern. Biomed. Eng*., vol. 37, no. 1, pp. 24–34, 2017. https://doi.org/10.1016/j.bbe.2016.12.003

[29] Y. Li, "Research on neural network algorithm in artificial intelligence recognition," *Sustain. Energy Technol. Assessments*, vol. 53, 102691, 2022. https://doi.org/10.1016/j.seta.2022.102691

[30] M. I. Thusnavis Bella and A. Vasuki, "An efficient image retrieval framework using fused information feature," *Comput. Electr. Eng*., vol. 75, pp. 46–60, 2019. https://doi.org/10.1016/j.compeleceng.2019.01.022

[31] F. López de la Rosa, J. L. Gómez-Sirvent, R. Sánchez-Reolid, R. Morales, and A. Fernández-Caballero, "Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a ResNet50 convolutional neural network," *Expert Syst. Appl*., vol. 206, 117731, 2022. https://doi.org/10.1016/j.eswa.2022.117731

[32] K. Alomar, H. I. Aysel, and X. Cai, "Data augmentation in classification and segmentation: A survey and new strategies," *Journal of Imaging*, vol. 9, no. 2, 46 2023. doi: 10.3390/jimaging9020046

[33] V. Vasan, N. V. Sridharan, S. Vaithiyanathan, and M. Aghaei, "Detection and classification of surface defects on hot-rolled steel using vision transformers," *Heliyon*, vol. 10, no. 19, e38498, 2024. https://doi.org/10.1016/j.heliyon.2024.e38498

[34] S. Lee, C. He, and S. Avestimehr, "Achieving small-batch accuracy with large-batch scalability via Hessian-aware learning rate adjustment," *Neural Networks*, vol. 158, pp. 1–14, 2023. https://doi.org/10.1016/j.neunet.2022.11.007

[35] P. Wang and N. Zheng, "Convergence analysis of asynchronous stochastic recursive gradient algorithms," *Knowledge-Based Syst*., vol. 252, 109312, 2022. https://doi.org/10.1016/j.knosys.2022.109312

[36] M. Vilares Ferro, Y. D. Mosquera, F. J. R. Pena, and V. M. D. Bilbao, "Early stopping by correlating online indicators in neural networks," *Neural Networks*, vol. 159, pp. 109–124, 2023. https://doi.org/10.1016/j.neunet.2022.11.035

[37] M. Rasyidi and T. Bariyah, "Batik pattern recognition using convolutional neural network," *Bull. Electr. Eng. Informatics*, vol. 9, no. 4, pp. 1430–1437, 2020. doi: 10.11591/eei.v9i4.2385

[38] E. Utaminingsih and I. Sahputra, "Automated recognition of batik aceh patterns using machine learning techniques," *Brill. Res. Artif. Intell*., vol. 4, no. 2, pp. 619–624, 2024. doi: 10.47709/brilliance.v4i2.4831

[39] I. P. Sari, L. Elvitaria, and R. Rudiansyah, "Data-driven approach for batik pattern classification using Convolutional Neural Network (CNN)," *J. Mandiri IT*, vol. 13, no. 3, pp. 323–331, 2025. doi: 10.35335/mandiri.v13i3.361

[40] F. F. K. A. A. Aziz and S. Saefurrohman, "Deteksi dan pengenalan jenis corak batik nusantara menggunakan metode CNN berbasis Android," *J. Teknol. Sist. Inf. dan Apl.*, vol. 6, no. 2, pp. 191–201, 2023.

[41] M. S. A. Karim, A. A. Hakim, C. S. Rafika *et al*., "Klasifikasi motif batik yogyakarta dan pekalongan menggunakan metode GLCM dan CNN berbasis arsitektur EfficientNet," *Pros. Semin. Nas. Inform. Bela Negara*, vol. 5, no. 1, pp. 350–356, 2025. doi: 10.33005/santika.v5i1.747

[42] R. Rosalina, G. Sahuri, and H. Desriva, "A mobile-optimized convolutional neural network approach for real-time batik pattern recognition," *IAES Int. J. Artif. Intell*., vol. 13, 3018, 2024. doi: 10.11591/ijai.v13.i3.pp3018-3027

[43] N. Bhatia, G. Rani, and V. S. Dhaka, "Innovative applications of deep learning in cultural heritage development and preservation: A customization perspective," 2024, pp. 493–497. doi: 10.1109/MetroLivEnv60384.2024.10615278

[44] G. Ilieva, T. Yankova, and S. Klisarova-Belcheva, "Effects of generative AI in tourism industry," *Information*, vol. 15, no. 11, 671, 2024. doi: 10.3390/info15110671

[45] O. Alyasiri, K. Selvaraj, H. Younis, T. Sahib, M. Almasooudi, and I. Hayder, "A survey on the potential of artificial intelligence tools in tourism information services," *Babylonian J. Artif. Intell*., vol. 2024, pp. 1–8, 2024. doi: 10.58496/BJAI/2024/001

[46] H. Bi and N. Nasir, "Innovative approaches to preserving intangible cultural heritage through AI-driven interactive experiences," *Acad. J. Sci. Technol*., vol. 12, pp. 81–84, 2024. doi: 10.54097/98nre954