The Comparison of GAN and CNN Models in the Innovation of Coloring Madura and Bali Batik

Yohan Permana
Department of Information Systems
Universitas Trunojoyo Madura
Bangkalan, East Java, Indonesia
yhuntp@gmail.com

Arik Kurniawati
Department of Informatics Engineering
Universitas Trunojoyo Madura
Bangkalan, East Java, Indonesia
arik.kurniawati@trunojoyo.ac.id

Fitri Damayanti
Department of Information Systems
Bangkalan, East Java, Indonesia
Madura, Indonesia
fitrid@trunojoyo.ac.id

I Ketut Adi Purnawan
Department of Information Technology
Universitas Udayana
Badung, Bali, Indonesia
adipurnawan@unud.ac.id

Abstract—**This study aims to innovate the automatic coloring of batik patterns using deep learning models. Specifically, it compares the performance of Generative Adversarial Networks (GAN) with pre-trained Caffe-based Convolutional Neural Networks (CNN) in coloring images of Madura and Bali batik. The dataset consists of 388 Madura batik images for training, 97 for validation, and 20 distinct images of both Bali and Madura batik for testing. This dataset was obtained through web scraping from batik posts on social media platforms like Instagram, Bing Image Search using specific keywords, and Kaggle, followed by a manual combination and cleaning process. The GAN model was trained with varying epochs (40, 80, 150), while the CNN utilized pre-trained Caffe weights. The evaluation was conducted using the Peak Signal-to-Noise Ratio (PSNR), Fréchet Inception Distance (FID), Mean Squared Error (MSE), and Structural Similarity Index (SSIM). The results indicate that the GAN model with 150 epochs outperformed the CNN, achieving a PSNR of 29.702, an FID of 84.016, an MSE of 511.8812, and an SSIM of 0.9925, demonstrating superior color creation and artistic detail in batik. Conversely, the CNN model exhibited lower performance, with a PSNR of 28.218, an FID of 200.271, and an SSIM of 0.7925, indicating its limitations in preserving the intricate patterns and colors of batik. This research demonstrates the applicability of GAN in automatic batik coloring, potentially providing innovative solutions for the batik industry while maintaining the cultural and artistic integrity of traditional designs.**

Keywords— automatic coloring, batik, CNN, deep learning, GAN.

I. INTRODUCTION

Batik is not only very important as part of the Indonesian cultural heritage, but it bears many philosophical values representing the richness of local culture and traditions. [1]. There are many characteristic batik motifs and patterns in different places in Indonesia, one of them being Madura, from a mixture of flora and fauna motifs to bright colors that symbolize the nature and culture of Madura. [1]. While Balinese Batik uses more geometric motifs under the influence of Hindu culture, the colors are more subdued, such as black, white, and brown. [2].

The making of batik is still manually done, especially in the coloring process, which requires exceedingly high skills and much time. Through time, several other motives of batik have become complex and have added to the problems that come up when trying to maintain the traditional way of coloring. The dire need for innovation is that the efficiency of the coloring of batik might be reduced without reducing its aesthetic value and beauty.

With the development of the technologies of artificial intelligence and in particular deep learning, there is application to innovations in image coloring a Generative Adversarial Network. Specifically, the main constitution of the architecture involves a generator that is supposed to paint colors based on the pattern learned from the training data and a discriminator that evaluates the coloring results against the original images. [3]. This makes GAN a very potential technology that could be implemented effectively and innovatively during the automatic coloring of batik.

The first paper, "Automatic Colorization of Black and White Images Using GAN," was steered in a way to increase the effectiveness of coloring black and white photos to provide chromatic values that retain the original brightness. The Grayscale2Colorization Mixed Dataset used in this research project is part of Places365; it contains 10,000 grayscale images for training and 457 for testing, all of them being 128x128 pixels in size. In general, a U-Net generator and a Patch-GAN discriminator were used in the architecture. It was inferred from these works that the models performed great in coloring grayscale pictures. The value of MSE is 0.038, SSIM with 0.83, and an FID of 0.0118, hence getting a good realistic colored image. [4].

One more related, earlier research effort in a slightly different domain has been put into the study of GAN modeling for innovation in coloring applied to anime sketch images. Results confirmed that the auto-colorized images from the black-and-white sketches would bring great ease to color concerning manual coloring. It was tested at epochs 19, 37, and 50, the quality was taken through the Fréchet Inception Distance, and the value obtained was 58.389. The results support once again the potential of GANs to improve quality and efficiency in coloring images [5]. There have also been previous studies focusing on evaluating the image compression generated by GAN, assessing the quality of image reconstruction using the PSNR metric, which is effective in providing information regarding the quality of the reconstructed images from GAN. [6].

In addition, previous research using a different model, namely a pre-trained Caffe-based CNN, has also been conducted to achieve automatic coloring of grayscale images from the WordNet dataset. This study produced more realistic colored images compared to the original inputs, utilizing the Caffe model along with the OpenCV, NumPy, and Argparse libraries. This research showed an improvement in coloring accuracy of 32% compared to the previous model, with a reduction in desaturation in the final output [7].

Previous research shows that Generative Adversarial Networks (GANs) have been employed in image coloring experiments, involving both discriminator and generator networks and optimal epochs for realistic results. The colorings are evaluated using metrics like Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Fréchet Inception Distance (FID), and Mean Squared Error (MSE). In contrast, pre-trained Caffe-based Convolutional Neural Networks (CNNs) offer a simpler mechanism, using pre-trained models through the OpenCV library.

In batik image coloring, both GAN models and pre-trained Caffe-based CNNs offer efficient automatic coloring solutions that preserve the aesthetic value of batik motifs. This study aims to implement and compare these models for automatically coloring Balinese and Madurese batik images, evaluated using SSIM, FID, MSE, and PSNR metrics.

II. Methods

Experimental methodology is used in this study to implement Deep Learning frameworks, notably GAN and CNN using pre-trained Caffe models, for innovating the coloring of Balinese and Madurese batik images and making possible a comparative analysis of outcomes from both models.

A. Dataset

This research utilizes a batik image dataset obtained through a data scraping process, which involves data extraction from websites using Python programming. [8]. The library used for scraping is Instaloader, which functions to retrieve image posts from Instagram based on the username keyword. Subsequently, the Bing Image Downloader is used to download images from the Bing search engine based on keywords. [9] [10].

TABLE 1. PROPORTION OF SCRAPING DATASET

Source	Total Images	After Cleaning	Keyword with Most Images
Bing	152	122	Batik Madura, Batik Bangkalan, Batik Sumenep, Batik Pamekasan, Batik Sampang
Instagram	367	343	batikmadura_, batikmadura__, areya_batikmadura
Kaggle	30	20	-
Combined	519	485	-

Overall, the scraping results from Bing and Instagram contain Madurese batik data that will be used as training and validation data. This data is then manually cleaned to remove duplicate images. The dataset used as test data in this research was obtained through Kaggle, which includes images of Balinese and Madurese batik, ensuring that there are no images identical to those from the scraping results. The data proportions can be seen in Table 1, which shows the total data proportions before and after cleaning, as well as the keywords based on the largest number of images and their sources.

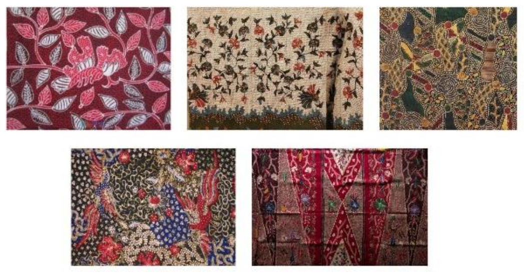


Fig. 1. Dataset Visualization

Fig. 1. is a visualization of the batik data that has been collected. In this image, various Madurese batik motifs are displayed, reflecting the cultural richness and traditions of the region. Each motif not only has a unique design but also consists of different values of color and brightness, which are closely related and serve as the main materials for modeling and generating new colorings.

B. Batik Image Coloring with the Generative Adversarial Network (GAN) Model

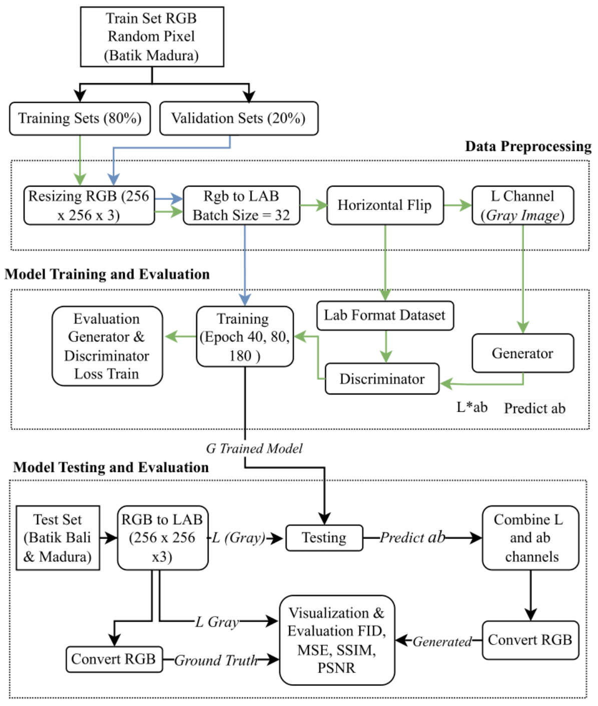

Train Set RGB Random Pixel (Batik Madura)
Training Sets (80%)
Validation Sets (20%)
Data Preprocessing
Resizing RGB (256 x 256 x 3)
Rgb to LAB Batch Size = 32
Horizontal Flip
L Channel (Gray Image)
Model Training and Evaluation
Evaluation Generator & Discriminator Loss Train
Training (Epoch 40, 80, 180)
Lab Format Dataset
Generator
Discriminator
L*ab
Predict ab
G Trained Model
Model Testing and Evaluation
Test Set (Batik Bali & Madura)
RGB to LAB (256 x 256 x3)
L (Gray)
Testing
Predict ab
Combine L and ab channels
L Gray
Visualization & Evaluation FID, MSE, SSIM, PSNR
Generated
Convert RGB
Convert RGB
Ground Truth

Fig. 2. Generative Adversarial Network (GAN) Staining research flow

Generative Adversarial Network (GAN) is a machine learning model that consists of two components: the generator and the discriminator. The discriminator's role is to distinguish between real data and synthetic data generated by the generator. Meanwhile, the generator's role is to create synthetic data that mimics the real data to the point where it becomes difficult to differentiate, attempting to fool the discriminator. [11].

Fig. 2. shows the process flow of the GAN model used for batik coloring. The process begins with preparing the

Madurese batik data, which has varying pixel sizes and uses RGB color channels. The data is then randomly split using random permutation, with 80% as training data and 20% as validation data. Next, preprocessing is performed by resizing the images to 256x256 pixels with 3 color channels. After that, the RGB colors are converted to LAB, where L represents luminance (black-and-white), a represents the green-to-red channel, and b represents the blue-to-yellow channel [12]. The Lab color space contains a dedicated channel to represent the brightness of the image, and the color information is fully encoded in the remaining two channels, making it an optimal choice for this task [13]. The converted data is normalized within the range of [-1, 1] to accelerate training, and the batch size is set to 32. In addition, random horizontal flips are applied to increase dataset variation and reduce overfitting. After preprocessing, the data is used to train and evaluate the GAN model.

The model training and evaluation are conducted after the preprocessing stage is completed, producing clean image data in the LAB channel. The next step involves feeding the L channel into the generator network. The generator then predicts new color values, represented by the ab channels, and combines them back into the new LAB image. This image is then passed to the discriminator, where the discriminator evaluates the LAB values generated by the generator against the original preprocessed LAB values. The discriminator classifies which images are real and which are synthetic, based on the original and generated LAB inputs. This process is repeated in a loop over several iterations (epochs) with specific batch sizes. During the training process, validation data is used for visualization to monitor the model's performance after each iteration or epoch. In this study, three test scenarios were conducted with epochs of 40, 80, and 150. The evaluation of the generator and discriminator losses is used to measure how well the generator can produce realistic images that fool the discriminator, and how effectively the discriminator can distinguish between real and generated images. In the GAN modeling used in this research, the discriminator and generator each have different architectures, as illustrated in Fig. 3. and Fig. 4.

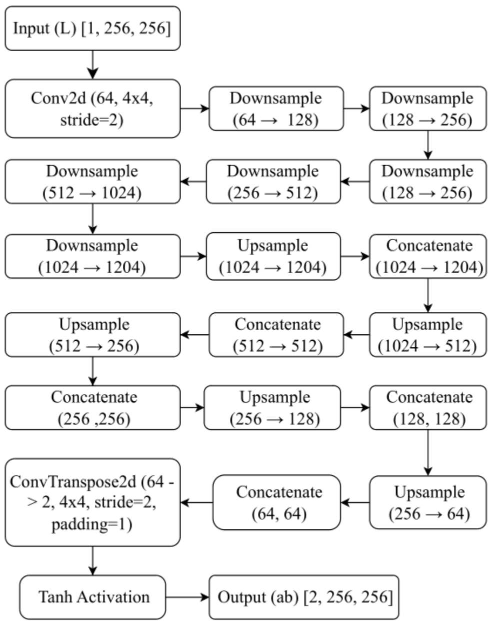
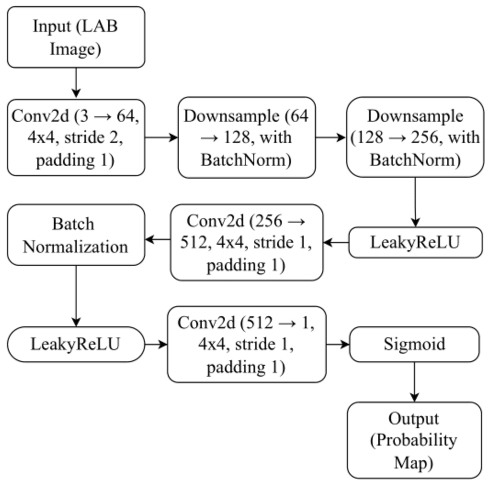


Fig. 3. Architecture diagram of the U-Net GAN generator with downsampling, upsampling and skip connections

Fig. 3. illustrates the Generator architecture diagram, which employs a U-Net approach. U-Net combines downsampling and upsampling layers with skip connections to preserve critical details during the coloring process. This architecture allows the model to retain high-resolution information from the input image as it passes through deeper layers, resulting in more accurate and realistic colorized images. The skip connections link the downsampling layers to the upsampling layers, helping to maintain spatial information throughout the image transformation process. [14]. The data processing flow begins with an input grayscale (L) and proceeds to a colored output (a, b) in the LAB format. The process starts with a convolutional layer (Conv2d), which captures the initial features from the input, followed by a series of downsampling layers aimed at reducing the spatial dimensions while increasing the number of feature channels, from 64 to 1024. Once the features are extracted, the data is passed through several upsampling layers, which progressively increase the spatial dimensions while incorporating information from the previous downsampling layers through concatenation operations. This ensures that important details are preserved as the image dimensions are expanded.



Input (LAB Image)
Conv2d (3 → 64, 4x4, stride 2, padding 1)
Downsample (64 → 128, with BatchNorm)
Downsample (128 → 256, with BatchNorm)
LeakyReLU
Conv2d (256 → 512, 4x4, stride 1, padding 1)
Batch Normalization
LeakyReLU
Conv2d (512 → 1, 4x4, stride 1, padding 1)
Sigmoid
Output (Probability Map)

Fig. 4. Architecture diagram of the PatchGAN in the GAN Discriminator

In Fig. 4. the architecture of the Discriminator used in this study is illustrated, where the model receives input images in LAB format with three channels (L, a, b). The data passes through a series of downsampling convolutional layers, starting from 64 channels and increasing up to 512 channels. Each layer is accompanied by LeakyReLU activation and batch normalization to enhance training performance. Finally, it reduces the dimensional features to one channel through a sigmoid activation after all downsampling and is ready to output the probability map.

With such an architecture, the network can appreciate the difference between the images being generated by the Generator and the real images to give the needed feedback for the improvement of the images. The Discriminator employed within the scope of this research is the Patch GAN model image is divided into many patch-small components and works on rating the realism of each patch independently. [15].

In this GAN model, therefore, losses for both the Discriminator and the Generator will be computed using BCE, which means that this measure will appropriately reflect how well the model can distinguish between real and generated images, and indicate how well each of its components performs during training [16]. The formula for calculating BCE can be found in Equation (1).

$$LossG = E_{P_{data}}[\log(D(x))] \; + E_{P_z}\;[\log(-D(G(z))] \quad (1)$$

Where:

- The first term $E_{P_{data}}[\log(D(x))]$ Represents the expected log probability of real images being classified as real.
- Finally, the second term E_{P_z} $[\log(1 - D(G(z))]$ Amounts to the log probability, with which generated images are classified as fake by the discriminator.

After the model training was completed, the results of the trained model at epochs 40, 80, and 150 were tested by loading the trained generator model (G) and using it to color test data consisting of Bali and Madura batik patterns selected to be different from the training data. Epochs 40, 80, and 150 were chosen to represent the early, middle, and late phases of model training. This selection aims to simplify the analysis without losing the main trends. Evaluation at higher epochs does not guarantee significant improvement, especially with small datasets, so these three epochs provide a sufficiently representative overview. The testing process involved converting the test data into the LAB color space, using the L channel (grayscale) as input for coloring by the model, then converting the output back to RGB and visualizing it as grayscale input images, ground truth (original image), and images generated by the model. The evaluation was conducted by comparing the generated images with the original image using metrics such as FID, SSIM, MSE, and PSNR, with detailed results explained in section D and final comparisons made with a CNN model based on a pre-trained CaffeModel.

C. Image Colorization of Batik Using Pre-trained Caffe Model Based on Convolutional Neural Network (CNN)

CNN, or Convolutional Neural Network, is a deep neural network technique developed for processing images. It uses several convolutional layers which, on their own, extract critical features such as edges and patterns. Consequently, this system can recognize and classify objects in pictures. [17]. A machine learning model pre-trained earlier on a large dataset for any activity is known as a pre-trained model. [18]. In this study, a pre-trained model based on a Convolutional Neural Network (CNN) is utilized, specifically the Caffe model for image colorization. This model has been trained on a large dataset of color images, enabling the neural network to learn natural coloring patterns effectively.

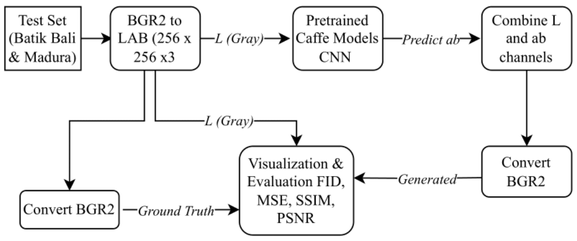

Test Set (Batik Bali & Madura)
BGR2 to LAB (256 x 256 x3
L (Gray)
Pretrained Caffe Models CNN
Predict ab
Combine L and ab channels
L (Gray)
Convert BGR2
Generated
Visualization & Evaluation FID, MSE, SSIM, PSNR
Convert BGR2
Ground Truth

Fig. 5. CNN-based pre-trained Caffe Model Staining research flow

In Fig. 5. CNN based Caffe model Staining research flow is presented, where the data used consists of batik images from Bali and Madura. The preprocessing of the images involves converting from BGR (Blue, Green, Red) to LAB color space, utilizing OpenCV for this task. Although the preprocessing steps differ from those in the GAN model, which uses PyTorch, the overall sequence of processes is quite similar. After preprocessing, the L channel is input into the pre-trained Caffe model to predict the ab values, which are then combined and converted back to BGR color space. The results are visualized, showcasing the original data (ground truth), the grayscale images, and the generated colorization results from the model. Subsequently, the outcomes are evaluated using FID, MSE, SSIM, and PSNR metrics, and these results will be compared with those from the GAN colorization model.

D. Model Comparison Evaluation

After successfully implementing the GAN model and the pre-trained Caffe model-based CNN, an evaluation was conducted based on the test data consisting of Balinese and Madurese batik. The evaluation results were then compared and analyzed. The evaluation consists of the following assessments:

1. Fréchet Inception Distance (FID) measures the distance between the feature distributions of the newly generated colored images produced by the model and the feature distributions of the original images [19]. The calculation of FID can be seen in equation (2). FID ensures that the resulting image is not only visually realistic but also retains the original patterns and characteristics of the batik, as this metric evaluates the similarity of the visual feature distribution between the new image and the original image.

$$FID \; = \| \mu 1 - \mu 2 \|^2 + Tr(C1 + C2 - 2\sqrt{C_1 C_2}) \qquad (2)$$

Where:

- $\mu 1$ and $\mu 2$ Are the mean vectors of the features extracted from the original images and the generated images.
- $C1$ and $C2$ Are the covariance matrices of the features of the original images and the generated images.
- Tr Tr is the trace operation on a matrix, which is the sum of the diagonal elements of the matrix.

2. Structural Similarity Index (SSIM) measures the structural similarity between two images. This

metric takes into account luminance, contrast, and structure in the images [20]. The formula for calculating SSIM can be found in Equation (3), which is used to ensure that the subtle nuances of color and shape are preserved so that the artistic meaning of batik is not lost, especially in reconstructing the coloring into specific patterns.

$$SSIM(xy) = \frac{(2\mu_x\mu_y + C1)(2\sigma_{xy} + C2)}{(\mu_x^2 + \mu_x^2\, C1)(\sigma_x^2 + \sigma_y^2 + c2)} \quad (3)$$

Where:

- μ_x and μ_y Are the averages of images x and y Respectively.
- σ_x^2 dan σ_y^2 Are the variances of images x and y Respectively.
- σ_{xy} Is the covariance between images x and y.
- $C1$ and $c2$ Are small constants used for numerical stability.

3. Mean Squared Error (MSE) calculates the average of the squared differences between the pixel values of the original image and the generated image. The formula for calculating MSE can be found in Equation (4). MSE provides a clear indication of the magnitude of the error in the new colorization results [21]. In the context of art, a low MSE indicates a more accurate and satisfying colorization that effectively reproduces the nuances of the original batik.

$$MS = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \quad (4)$$

Where:

- n Is the total number of pixels.
- x_i Is the pixel value of the original image.
- y_i Is the pixel value of the generated image.

4. Peak Signal-to-Noise Ratio (PSNR) measures the ratio between the maximum signal (original image) and the noise (the difference between the original image and the generated image) [21]. The formula for calculating PSNR can be found in Equation (5). The higher the PSNR, the better the quality of the generated image, ensuring that the newly produced colors remain close to the original without sacrificing important details or patterns.

Where:

- R Represents the maximum pixel value.

III. Results and Discussion

The data comparison in this study involves 388 images of Madura batik for training (80%), 97 images for validation (20%), and 20 images of Madura and Bali batik for testing. These 20 test images have been manually filtered to ensure they were not included in the training data and process. The testing scenario includes a Generative Adversarial Network (GAN) model trained using Madura batik images with varying numbers of epochs (40, 80, 150). Performance analysis is conducted through loss graphs for both the discriminator and generator, and the results of the trained model are used for testing on the Madura and Bali batik test data. This model is compared with a Convolutional Neural Network (CNN) based on Caffe that uses a pre-trained model. Evaluation is performed using metrics such as Peak Signal-to-Noise Ratio (PSNR), Fréchet Inception Distance (FID), Mean Squared Error (MSE), and Structural Similarity Index (SSIM) to assess how well each model maintains the quality and artistic details of batik while generating new colors as an innovation in automatic coloring.

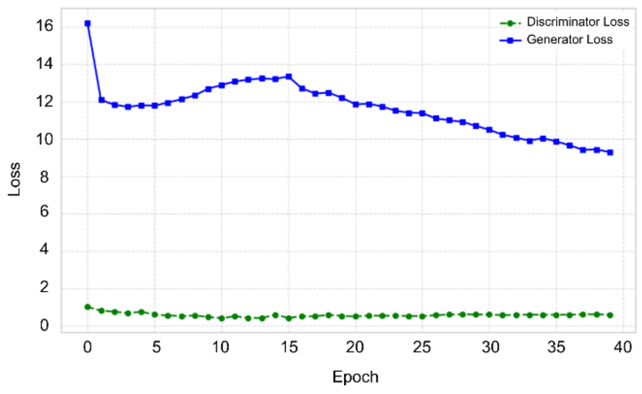


Fig. 6. Visualization of Generator and Discriminator Loss Graph at Epoch 40

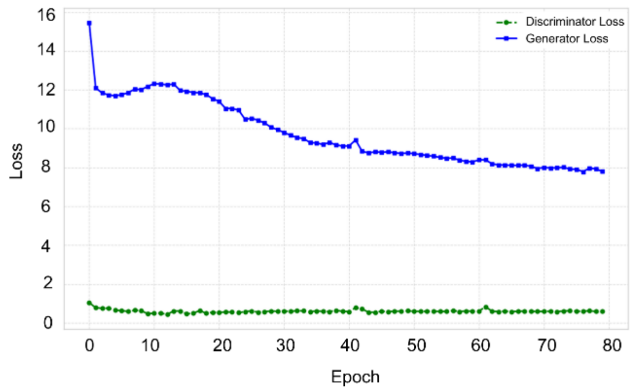


Fig. 7. Visualization of Generator and Discriminator Loss Graph at Epoch 80

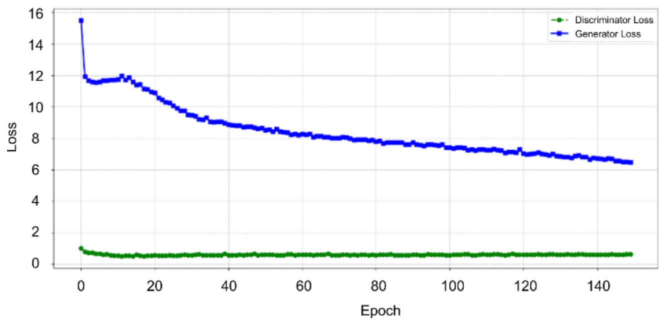


Fig. 8. Visualization of Generator and Discriminator Loss Graph at Epoch 150

The results of training the GAN model with epoch 40, 80 and 150 experiments can be seen in the generator and discriminator loss graphs in Figures 6, 7 and 8. Based on these results, it can be analyzed that at epoch 40, the generator loss value decreased from 16.6647 to 9.3321 with small fluctuations at the 15th epoch point, then at epoch 80, the generator showed an increase with a decrease in loss from 15.9999 to 7. 7903, and the loss discriminator decreased from 1.2783 to 0. 5928 with small fluctuations at the 10th epoch than at epoch 150, the generator reached stability with a gradual decrease from 12.0492 to 6.5088, while the loss discriminator remained stable in the range of 1.2495 to 0.6013 with small fluctuations at the 10th epoch. It can be seen that the training results show an improvement in the GAN model as the number of epochs increases. Although there is a decrease in loss in the generator and discriminator, small fluctuations at some epoch points indicate that the model is still not fully stable. In GAN, unlike traditional regression or classification models, the discriminator loss and generator loss must not outweigh each other, as this can cause the model to fail to converge [22]. Maintaining a balance between the two is crucial to ensure that the training process progresses effectively, with the generator continuously producing more realistic data while the discriminator remains capable of accurately distinguishing between real and generated data. An imbalance in the losses can cause one component to become too dominant, causing the model to lose direction and not achieve optimal results. In order to reduce these fluctuations and to accelerate a more stable convergence, it is recommended to apply some additional techniques in further research. One of these is learning rate decay, which can help to gradually reduce the learning rate during the training process.

Table 2. Comparison of GAN loss curve generator and discriminator

Model	Dataset	Loss Generator	Loss Discriminator
DCGAN [3]	Places365	11.52	0.52
Pix2pix GAN [23]	COCO (Common Objects in Context)	9.43	0.58
Our models	Batik	6.50	0.60

Based on Table 2 on the training of the GAN model used in this study, competitive results are obtained for batik coloring based on the comparison of the loss generator and loss discriminator values with previous research. In the first model, namely Deep Convolutional Generative Adversarial Network (DCGAN) on the grayscale image coloring with Places365 dataset, a loss generator value of 11.52 and a loss discriminator of 0.52 were obtained [3]. In the second model, Pix2pix GAN applied to the grayscale image coloring with the COCO dataset, the loss generator value reached 9.43 and the loss discriminator was 0.58 [23]. In comparison, our model on the Batik dataset produces the best value with a loss generator of 6.50 and a loss discriminator of 0.60. Although the dataset and the coloring task are different, the lower loss generator value indicates that our model has a competitive potential in producing batik coloring with reasonable detail, while the loss discriminator remains in a balanced performance range.

The test results of the GAN model using 20 images of Madura and Bali batik are shown in Figures 9, 10, and 11. Each figure consists of three parts: the original image, the grayscale image, and the generated image.

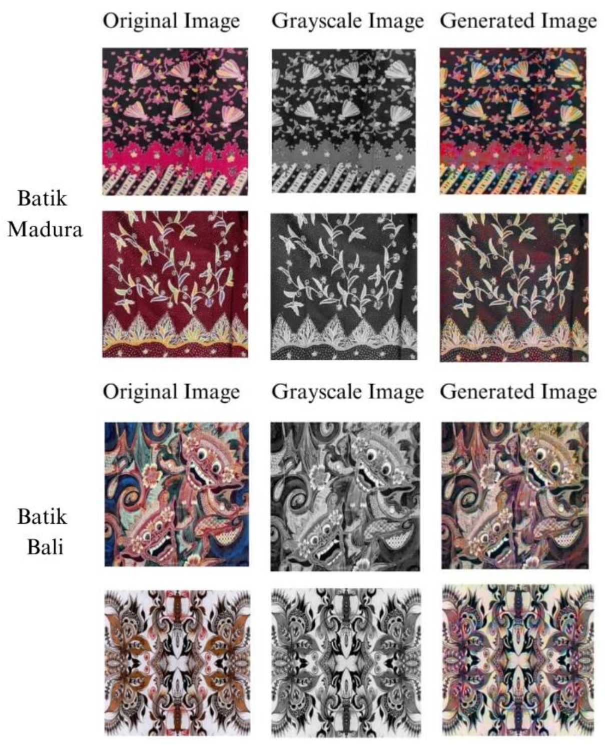


Fig. 9. Visualization of sample test results for Madura and Bali batik with Epoch 40

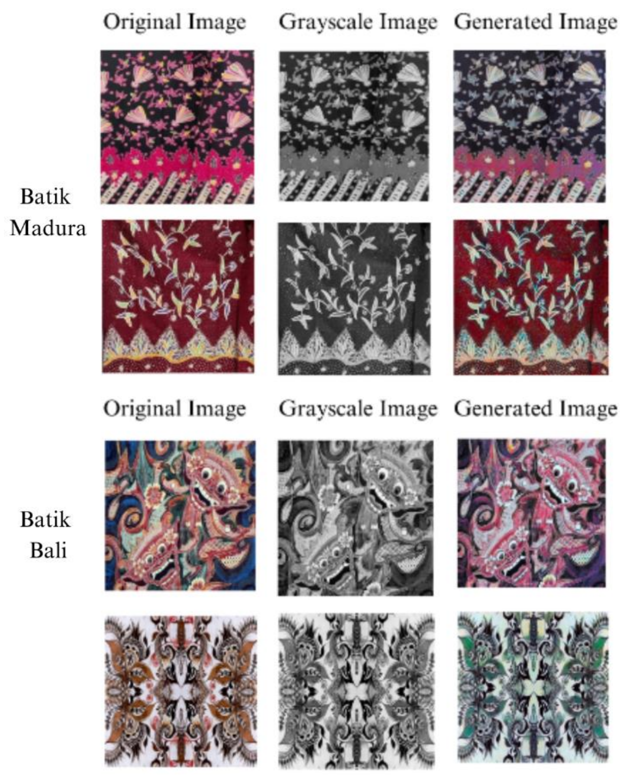


Fig. 10. Visualization of sample test results for Madura and Bali batik with Epoch 80

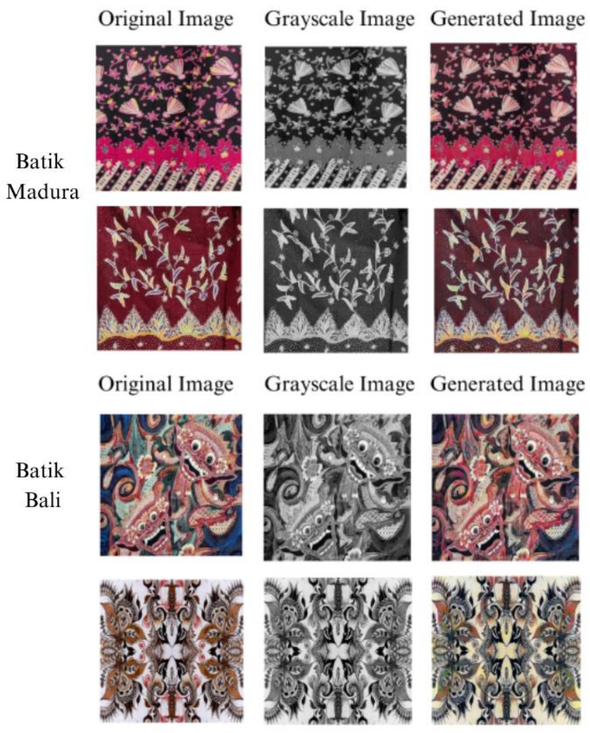


Fig. 11. Visualization of sample test results for Madura and Bali batik with Epoch 150

For the Caffe Model-based CNN model, the colorization results of Madura and Bali batik images can be seen in Fig. 12. These results were obtained directly from the pre-trained model without any additional training process.

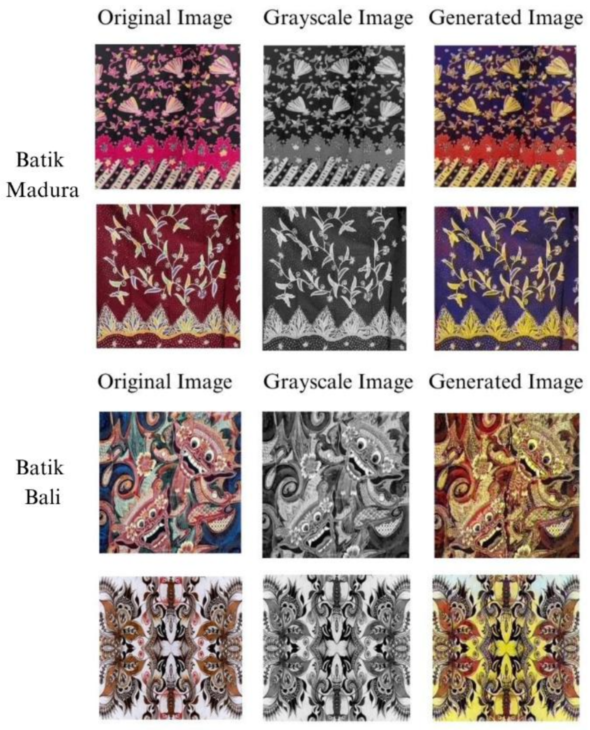


Fig. 12. Visualization of sample test results for Madura and Bali batik data using the Caffe-based pre-trained CNN model.

Based on the results from both models, quantitative evaluation can be conducted as presented in Table 2.

TABLE 2 QUANTITATIVE MODEL EVALUATION: PSNR, FID, MSE, SSIM

Model	Epoch	PSNR	FID	MSE	SSIM
CNN Caffe	N/A	28.21	200.27	660.50	0.792
GAN	40	29.03	95.01	21.06	0.992
GAN	80	29.54	92.98	23.82	0.987
GAN	150	29.70	84.01	21.33	0.992

Based on the model comparison evaluation in Table 2, it can be analyzed that the Caffe-based pre-trained CNN model shows suboptimal performance in batik coloring. The CNN recorded a lower PSNR value (28.21), a significantly higher FID (200.27), a large MSE (660.50), and a lower SSIM (0.7925). The coloring results from the CNN were inaccurate, mainly because this model used pre-trained weights that were not specifically trained for batik patterns and colors, making it less capable of accurately reproducing the colors and details of batik.

In contrast, the GAN model demonstrated superior results. At epoch 150, the GAN recorded the highest PSNR value (29.70) and the lowest FID (84.01), indicating better reproduction of colors and artistic details. The SSIM value was also the highest at 0.992, suggesting that the generated batik structure was more consistent. The success of the GAN model in producing new colors and deeper details has the potential to provide innovation in batik art, allowing artists to create richer and more varied works while still preserving the artistic meaning of batik culture.

IV. CONCLUSION

Based on the results of the research conducted, it can be concluded that the GAN model shows superior performance compared to the pre-trained Caffe-based CNN model in batik coloring tasks. Training the GAN model at various epochs (40, 80, and 150) shows a consistent decrease in loss generator and loss discriminator values, with the best value at epoch 150, namely loss generator 6.50 and loss discriminator 0.60, although there are still small fluctuations. Compared to previous studies, the GAN model in this study has a lower loss generator value, indicating its ability to produce batik coloring with better detail. The GAN model also recorded the highest PSNR (29.70), lowest FID (84.01), and highest SSIM (0.992) values, indicating a more accurate reproduction of batik colors and details as well as a consistent structure. This success confirms the potential of the GAN model as an innovative tool in batik coloring, enabling the creation of more diverse artworks without neglecting the artistic and cultural value of batik.

For further research on the GAN, we suggest the application of techniques such as learning rate decay to reduce fluctuations in loss values during training, as well as the exploration of the use of larger and more diverse datasets to improve the generalization of the model to various batik patterns and colors.

ACKNOWLEDGMENT

This research was financially supported by an Institutional Research Grant (No. 299/UN46/HK.02/2024) from the

Research and Community Service Institute (LPPM) of Universitas Trunojoyo Madura in 2024.

References

[1] R. . S. Suminto, "BATIK MADURA: Menilik Ciri Khas dan Makna Filosofinya," *Corak*, vol. 4, no. 1, pp. 1–12, 2015, doi: 10.24821/corak.v4i1.2356.

[2] I. R. Salma, M. Masiswo, Y. Satria, and A. A. Wibowo, "Pengembangan Motif Batik Khas Bali," *Din. Kerajinan dan Batik Maj. Ilm.*, vol. 32, no. 1, p. 23, 2016, doi: 10.22322/dkb.v32i1.1168.

[3] M. Ricky and M. E. Al Rivan, "Implementasi Deep Convolutional Generative Adversarial Network untuk Pewarnaan Citra Grayscale," *J. Tek. Inform. dan Sist. Inf.*, vol. 8, no. 3, pp. 556–566, 2022, doi: 10.28932/jutisi.v8i3.5218.

[4] P. Nepali, R. K. Sah, and S. Kc C, "Auto Colorization of Gray Images using GAN," *Proc. 12th IOE Grad. Conf.*, vol. 12, no. October 2022, pp. 992–1000, 2022.

[5] R. R. Kalendesang and D. H. Setiabudi, "Pewarnaan Otomatis Sketsa Gambar Menggunakan Metode Conditional GAN Untuk Mempercepat Proses Pewarnaan," *J. Infra*, vol. 10, no. 2, 2022.

[6] J. Lohdefink, A. Bar, N. M. Schmidt, F. Huger, P. Schlicht, and T. Fingscheidt, "On low-bitrate image compression for distributed automotive perception: Higher peak snr does not mean better semantic segmentation," *IEEE Intell. Veh. Symp. Proc.*, vol. 2019-June, pp. 424–431, 2019, doi: 10.1109/IVS.2019.8813813.

[7] R. Kapoor, B. Manocha, and A. Dobhal, "Black and White Image Color Restoration using Caffe Model," *Int. J. Res. Eng. Emerg. Trends*, vol. 4, no. 2, pp. 61–67, 2020.

[8] A. Himawan, A. Priadana, and A. Murdiyanto, "Implementation of Web Scraping to Build a Web-Based Instagram Account Data Downloader Application," *IJID (International J. Informatics Dev.*, vol. 9, no. 2, pp. 59–65, 2020, doi: 10.14421/ijid.2020.09201.

[9] Ridwang, A. A. Ilham, I. Nurtanio, and Syafaruddin, "Image search optimization with web scraping, text processing and cosine similarity algorithms," *2020 IEEE Int. Conf. Commun. Networks Satell. Comnetsat 2020 - Proc.*, no. December, pp. 346–350, 2020, doi: 10.1109/Comnetsat50391.2020.9328982.

[10] M. Vicente-Mariño and M. Varela-Rodríguez, "Scattered Images: scrapers and the representation of cancer on Instagram," *Cuadernos.info*, no. 49, pp. 72–97, 2021, doi: 10.7764/cdi.49.27809.

[11] A. Althbaity, M. M. Dessouky, and I. R. Khan, "Colorization Of Grayscale Images Using Deep Learning," *Proc. - 2022 14th IEEE Int. Conf. Comput. Intell. Commun. Networks, CICN 2022*, vol. 6, no. 11, pp. 131–138, 2022, doi: 10.1109/CICN56167.2022.10008319.

[12] W. Zhu, Z. Wang, Q. Li, and C. Zhu, "A Method of Enhancing Silk Digital Printing Color Prediction through Pix2Pix GAN-Based Approaches," *Appl. Sci.*, vol. 14, no. 1, 2024, doi: 10.3390/app14010011.

[13] K. Nazeri, E. Ng, and M. Ebrahimi, "Image colorization using generative adversarial networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10945 LNCS, pp. 85–94, 2018, doi: 10.1007/978-3-319-94544-6_9.

[14] R. Sankar, A. Nair, P. Abhinav, S. K. P. Mothukuri, and S. G. Koolagudi, "Image Colorization Using GANs and Perceptual Loss," *2020 Int. Conf. Artif. Intell. Signal Process. AISP 2020*, no. 1, pp. 0–3, 2020, doi: 10.1109/AISP48273.2020.9073284.

[15] A. A. Tolpadi *et al.*, "Synthetic Inflammation Imaging with PatchGAN Deep Learning Networks," *Bioengineering*, vol. 10, no. 5, 2023, doi: 10.3390/bioengineering10050516.

[16] C. Guo, X. Chen, Y. Chen, and C. Yu, "Multi-Stage Attentive Network for Motion Deblurring via Binary Cross-Entropy Loss," *Entropy*, vol. 24, no. 10, pp. 1–14, 2022, doi: 10.3390/e24101414.

[17] M. A. R. Ramadhan, T. Apriliyan, N. Ananta, and A. A. Zakkyfriza, "Perbandingan Jumlah Layer Pada Convolutional Neural Network Untuk Meningkatkan Akurasi Dalam Klasifikasi Gambar," vol. 2, no. 5, pp. 211–217, 2024.

[18] H. Chen *et al.*, "Pre-trained image processing transformer," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 12294–12305, 2021, doi: 10.1109/CVPR46437.2021.01212.

[19] M. Feuerpfeil, "Conditional Generative Adversarial Network : generate new face images based on Conditional Generative Adversarial Network : generate new face images based on attributes," no. July, 2020, doi: 10.13140/RG.2.2.32736.81925.

[20] Y. Wu, Y. Bai, Z. Lan, and S. Yao, "A Structural-Similarity Conditional GAN Method to Generate Real-Time Topology for Shell-Infill Structures," *Int. J. Comput. Methods*, no. May, 2023, doi: 10.1142/S0219876223410074.

[21] K. Fu, J. Peng, H. Zhang, X. Wang, and F. Jiang, "Image super-resolution based on generative adversarial networks: A brief review," *Comput. Mater. Contin.*, vol. 64, no. 3, pp. 1977–1997, 2020, doi: 10.32604/cmc.2020.09882.

[22] F. A. Putra, E. Y. Puspaningrum, and W. S. Saputra, "Otomatisasi Pewarnaan Citra Monokrom dengan Metode Generative Adversarial Network," *Pros. Semin. Nas. Inform. Bela Negara*, vol. 1, pp. 137–141, 2020, doi: 10.33005/santika.v1i0.37.

[23] O. B. Bruvik and M. Penfold, "Image Colorization using GAN and Inception-ResNet-v2," Stanford University, 2017, p. 4326. [Online]. Available: https://cs231n.stanford.edu/2024/papers/image-colorization-using-gan-and-inception-resnet-v2.pdf