Observing Pre-Trained Convolutional Neural Network (CNN) Layers as Feature Extractor for Detecting Bias in Image Classification Data

Amadea Claire Isabel Ardison[1], Mikhaya Josheba Rumondang Hutagalung[2*], Reynaldi Chernando[3], and Tjeng Wawan Cenggoro[4]
[1–4]Computer Science Department, School of Computer Science, Bina Nusantara University Jakarta 11480, Indonesia
[4]Bioinformatics and Data Science Research Center, Bina Nusantara University Jakarta 11480, Indonesia
Email: [1]amadea.ardison@binus.ac.id, [2]mikhaya.hutagalung@binus.ac.id, [3]reynaldi.chernando@binus.ac.id, [4]tjeng.cenggoro@binus.ac.id

***Abstract*—Detecting bias in data is crucial since it can pose serious problems when developing an AI algorithm. The research aims to propose a novel study design to detect bias in image classification data by using pre-trained Convolutional Neural Network (CNN) layers as a feature extractor. There are three datasets used in the research with varying degrees of complexity, those are low, medium, and high complexity. There are Modified National Institute of Standards and Technology (MNIST) Digits, batik collections (Parang, Megamendung, and Kawung), and Canadian Institute for Advanced Research (CIFAR-10) datasets. Then, the researchers make a baseline workflow and substitute a step-in feature extraction with a convolution using the first pre-trained CNN layer and each of its kernels. Then, the researchers evaluate the effect of the experiments using accuracy. By observing the effect of the individual kernel, the research can better make sense of what happens inside a CNN layer. The research finds that color in the image is an essential factor when working with CNN. Furthermore, the proposed study design can detect bias in image classification data where it is related to the color of the image. Detecting this bias early is important in helping developers to improve AI algorithms.**

***Index Terms*—Pre-Trained Convolutional Neural Network (CNN), Features Extractor, Data Bias, Image Classification**

I. Introduction

DETECTING bias in data is important since it can pose some serious problems when developing an AI algorithm [1–4]. Bias can exist in different types and forms [5]. One is from the training data that are inherently biased towards certain data groups. An example of one of the cases in that bias in data causes problems is in the Amazon hiring system [3]. It is biased towards male candidates because the training data are from ten years' worth of job applications that males dominate.

Bias in data can occur when certain groups of the data are more represented than others. For example, in the case of image recognition with color bias, data for the human face are mostly light-skinned, and data for the dark-skinned face are less prevalent. So, the model trained using the biased data classifies people with dark-skinned faces wrongly. Some research has studied ways to identify and quantify and methods to mitigate bias in data [6–8]. For example, previous studies use an approach that alternates values between potentially biased attributes such as gender and race [6] and re-weight the training examples [7].

To address the problem, the researchers propose a novel study design for detecting bias in data using a pre-trained Convolutional Neural Network (CNN) layer [9, 10] as a feature extractor in traditional Computer Vision workflow. While using a different approach, the study design complements the existing approaches for detecting bias. Meanwhile, CNN is commonly used for image recognition tasks, where the features are automatically extracted [11–15]. The researchers use the layer as a features extractor to detect bias and to make sense of what happens inside the CNN filters, as the current state of deep learning

Received: Jan. 30, 2022; received in revised form: May 23, 2022; accepted: May 24, 2022; available online: Sep. 14, 2022.
*Corresponding Author

is a black box [16, 17].

CNN is known to perform better than traditional computer vision by rapidly progressing due to the increasing computational power and the amount of available data to train its neural networks [18, 19]. Then, pre-trained models are introduced to be models which have been trained with a huge amount of data and can be used directly or as transfer learning and changed according to the needs. While there are many pre-trained CNN architectures, the researchers choose a pre-trained ResNet-50 model [20] for the experiments.

As a class of deep learning, CNN has feature extraction and classifier in one step of end-to-end learning. Inside the end-to-end learning, CNN's filter layers abstract the features of an image to get its edges, corner, contour, and object parts as feature representations with each layer deeper [21]. Although the process increases the performance of CNN, it is regarded as a black-box operation, which contrasts with the intent to analyze data in search of bias.

To use the CNN filters, the researchers take advantage of traditional Computer Vision's transparency and use the hybrid approach, which uses CNN filters as feature extractor in traditional Computer Vision workflow. Traditional Computer Vision workflow has separate steps for feature engineering of manual extraction, selection, and classifier [18]. In this way, the pre-trained CNN filters as features extractor can be used to complement other feature extraction methods. By using the hybrid approach, the researchers want to see the effect of CNN filters, analyze them, and detect the bias inside the data.

II. Research Method

A. Datasets

There are three datasets with different complexities in the experiments. The first dataset is Canadian Institute for Advanced Research (CIFAR-10) dataset [22]. The dataset contains 60,000 images of 10 classes with the size of 32 × 32 pixels. However, the experiments only use the training set, which amounts to 50,000 out of 60,000 images. About 40,000 images are used to train the model, and the other 10,000 images are to test the model. The classification of the images is either one of an airplane, automobile, bird, cat, deer, dog, frog, horse, ship, or truck. This dataset has raw images with many combinations of colors and edges, so it is considered a high-complexity dataset.

The second dataset is the Modified National Institute of Standards and Technology (MNIST) Digits dataset which is a subset of a larger set collected by NIST [23]. This dataset of handwritten digits is black and white (bi-level) images with the size of 28 × 28 pixels. The experiments use the training set of 60,000 images and the testing set of 10,000 images. As they are digits, the classification of the images is for each digit. Since this dataset consists of grayscale images with basic edges, it is considered a low-complexity dataset.

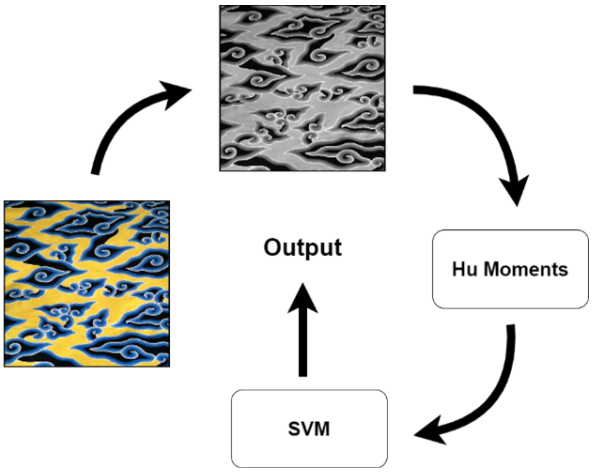

Fig. 1. Baseline pipeline.

The last dataset is a collection of batik images collected from the Internet. Because they are from different sources, the size of each image is different, with the smallest width and height of 120 pixels, so the researchers resize the images into 256 × 256 pixels with CV2. The collection can be classified into Kawung, Megamendung, or Parang, which are a few origins of batik. For training images, Kawung has 130 images, Megamendung has 130 images, and Parang has 148 images, totaling 408 images. Meanwhile, for testing images, Kawung has 40 images, Megamendung has 30 images, and Parang has 30 images, totaling 100 images. This dataset is considered a medium complexity dataset. Although it consists of raw images, they have a distinct pattern and are less complex than CIFAR-10.

B. Baseline

Before starting the experiments, the researchers create a baseline pipeline following the traditional Computer Vision flow, which the researchers use to make comparisons. After importing the dataset, the researchers extract their features in this pipeline by applying image grayscaling and Hu Moments [24]. Afterward, the Support Vector Machine (SVM) [25] is used as the classification model. The baseline pipeline of workflow is visualized in Fig. 1.

Grayscaling image is one way to reduce the amount of information provided in each image pixel. This

method converts the three channels of R (red), G (green), and B (blue) of the image into a binary image with intensity 0 to 1. The process is defined in Eq. (1).

$$I(x,y) = \frac{R+G+B}{3}. \tag{1}$$

$$\begin{aligned} h_0 &= \eta_{20} + \eta_{02}, \\ h_1 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \\ h_2 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2, \\ h_3 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2, \\ h_4 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \\ &\quad [(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ &\quad +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\ &\quad [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2], \\ h_5 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ &\quad +4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}), \\ h_6 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12}) \\ &\quad [(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ &\quad -(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03}) \\ &\quad [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]. \end{aligned} \tag{2}$$

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}}. \tag{3}$$

$$Y = \frac{p+q}{2} + 1. \tag{4}$$

$$H_i = -sign(h_i) \log |h_i|. \tag{5}$$

After grayscaling an image, the next step is to extract the images with Hu Moments. It is also called Geometric Invariant Moment (GIM), is a global feature extractor to extract a shape feature vector from the silhouette or outline of an image in an image to represent the shape of the object [24]. The features from Hu Moments are seven invariant moments under translations which are calculated using Eqs. (2)–(4) to calculate gamma. In those equations, h_0 to h_6 is the invariant moments, η_{pq} is the normalized moment order $(p+q)$, and μ_{pq} is the image moment order $(p+q)$. Since there is a big difference between the value of the first moment and the last moment, the researchers bring those moments into the same range using Eq. (5).

Lastly, the researchers apply SVM from scikit-learn as the classifier for the experiments. SVM is a supervised learning model which can be used for classification and regression problems [25]. For problems of classification type, scikit-learn has several SVM classes. Among those classes, the researchers choose to use Support Vector Classification (SVC) with gamma=1e-3 and C=100 as their hyperparameters.

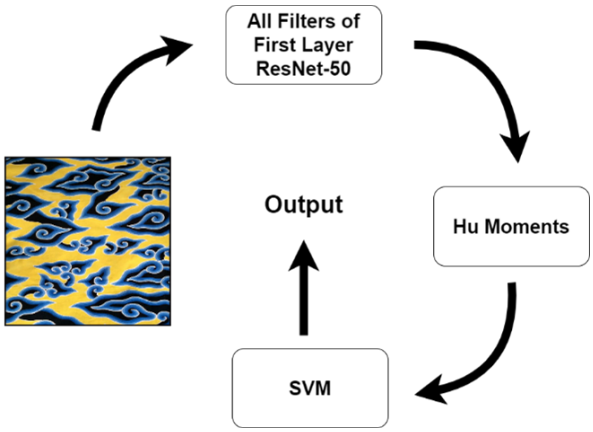

Fig. 2. Experiment with all filters of the first layer ResNet-50 pipeline.

C. Experiment with First Layer of ResNet-50

The experiments are divided into two parts, and the researchers focus on feature extraction. The first experiment is to see the effect of the ResNet-50 model's first layer filters (all 64 filters). It is done by replacing the image grayscale in the baseline pipeline, as visualized in Fig. 2.

Before applying the filters of the first layer ResNet-50, the researchers normalize the images using z-score normalization as defined in Eq. (6). It shows that x is the original value of data, μ is the mean of data, and σ is the standard deviation of data. The resulting standardized values are then convolved with the first layer of ResNet-50. It is known to recognize z-score normalized inputs better than other normalizations.

$$z = \frac{x - \mu}{\sigma}. \tag{6}$$

D. Experiment with Each Filter of the First Layer of ResNet-50

The second experiment is to see the effect of each of those first-layer filters. Once more, a filter is applied by replacing the image grayscale in the baseline pipeline as visualized in Fig. 3. With the first layer of ResNet-50 having 64 filters, the researchers iterate through each filter in the same pipeline. The 64 filters are visualized in Fig. 4.

E. Evaluation of the Effect of Experiments

Finally, there are many ways to assess the effect of those experiments with image classification as the

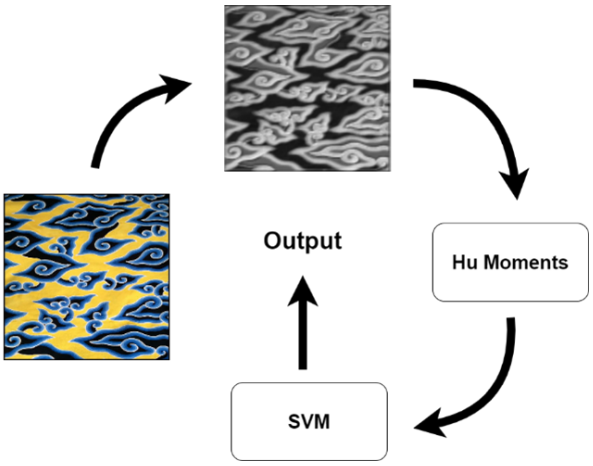

Fig. 3. Experiment with one filter of the first layer ResNet-50 pipeline.

TABLE I
DATASET COMPARISONS.

Dataset	Train Size	Image Size	Accuracy
MNIST Digits	60,000	28 × 28	57.83%
Batik Collections	408	256 × 256	49%
CIFAR-10	40,000	32 × 32	18.12%

problem. The researchers use accuracy as a standard metric for classification. It uses testing set to evaluate the effect of the experiments on those datasets.

III. RESULTS AND DISCUSSION

The researchers examine the accuracy of a classic Computer Vision model on three different datasets with varying complexities. It is deliberately directed to obtain comparisons and establish the result from the baseline model. According to Table I, for the baseline model, it is found that MNIST Digits have the highest accuracy at 57.83%, followed by batik collections at 49% and CIFAR-10 at 18.12%. It can be seen that the baseline model using the dataset with the highest level of complexity has the lowest accuracy. In this case, it is the CIFAR-10 dataset. Meanwhile, the dataset with the lowest level of complexity, the MNIST Digits, scores the highest. Table I finds little to no correlation between train size and image size with accuracy.

A. First Convolutional Layer of ResNet-50 Increases Accuracy on Complex Dataset

The researchers examine the performance of the model with the first convolutional layer of ResNet-50 as a feature extraction process. The effect of adding the first convolutional layer from ResNet-50 across the three different datasets is observed. As shown in Fig. 5, the most significant increase in accuracy is from the batik collections dataset, with a 3.00% increase, followed by a slight 0.68% increase in the CIFAR-10 dataset. However, there is a sharp decrease in accuracy for the MNIST Digits dataset by 21.57%.

In the experiment, it obtains the result for adding the CNN layer. It is surprising to find that the model trained with MNIST Digits scores the lowest in terms of the difference in accuracy, which in this case, it is a decrease in accuracy. Meanwhile, the more complex datasets, batik, and CIFAR-10, achieve higher accuracy compared to the baseline model. Applying the CNN layer to the baseline model helps the feature extraction process using complex datasets. However, it has an adverse effect on the accuracy if a dataset with a lower level of complexity is used, such as MNIST Digits.

Further experimentation on the MNIST Digits dataset is performed in which the researchers investigate the decrease in accuracy. Figure 6 shows that the MNIST Digits data is passed into the first CNN layer of ResNet-50. The observation shows that the image passed onto the CNN layer has lost some features, mainly in terms of image clarity, that the edges have become less defined.

B. Change in Accuracy Varies across Each Kernel from First Convolutional Layer in ResNet-50

From the result of the previous experiment, using the first convolutional layer of ResNet-50 affects the performance of the model. The next step is to continue to look for which kernel causes such an effect on the model. It is done by separating the 64 kernels in the first convolutional layer in ResNet-50 and applying them individually to the existing model.

The experiment shows that each kernel in the first convolutional layer of ResNet-50 used for image preprocessing gives result in varying accuracies, as shown in Fig. 7. It is found that for the CIFAR-10 dataset, the highest increase in accuracy is +4.02%. The lowest decrease in accuracy is -5.12%. Out of the 64 kernels in the first CNN layer of ResNet-50, 14 kernels result in an increase in accuracy for CIFAR-10.

Compared to the batik collections dataset, the highest increase is 8.00%, and the lowest decrease is -20.00%, with 6 out of 64 kernels increasing accuracy. However, for the MNIST Digits dataset, there is no increase in accuracy whatsoever, and the lowest decrease is -16.23%. The decrease in accuracy is found to be the result of less defined edges from the feature map of MNIST Digits, as shown in Fig. 8.

From the observation of experimenting with individual kernels from the first CNN layer of ResNet-50 as a feature extraction process, the largest improvement in

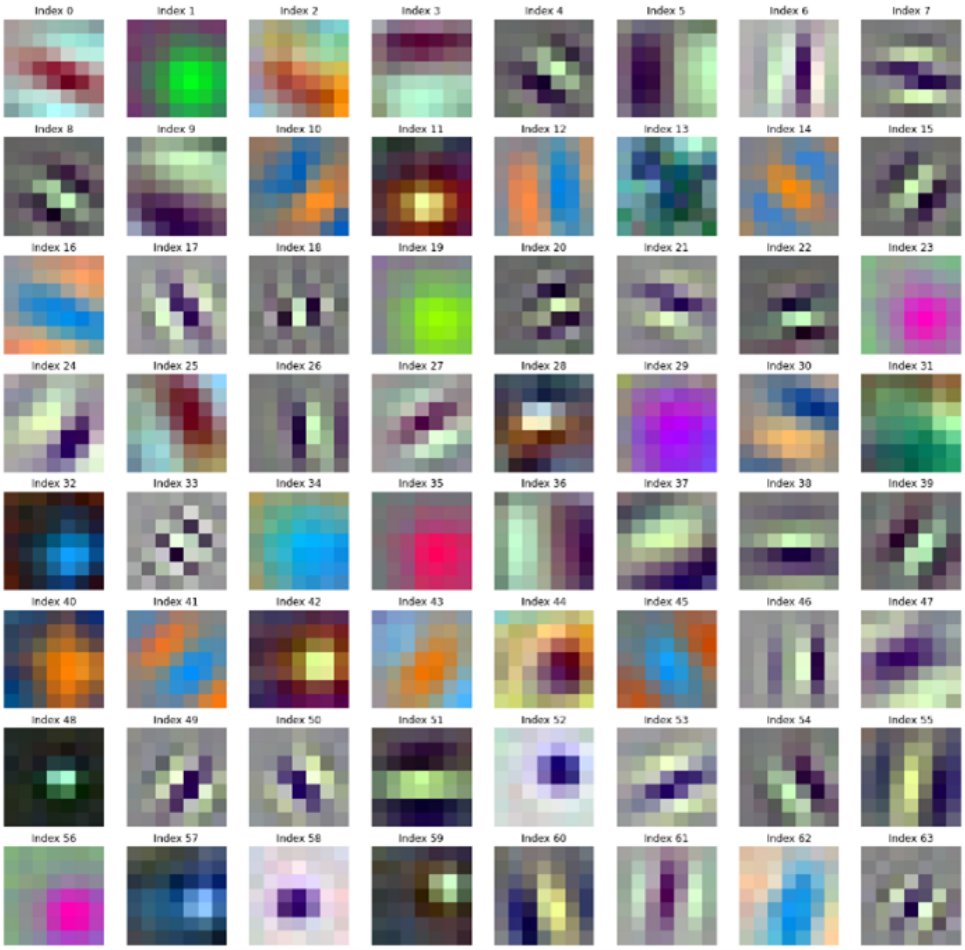

Fig. 4. The first layer of ResNet-50 filters.

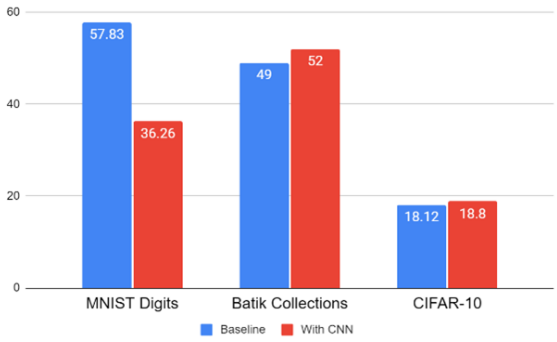

Fig. 5. Comparison of the result between baseline model and model by adding ResNet-50 first convolutional layer.

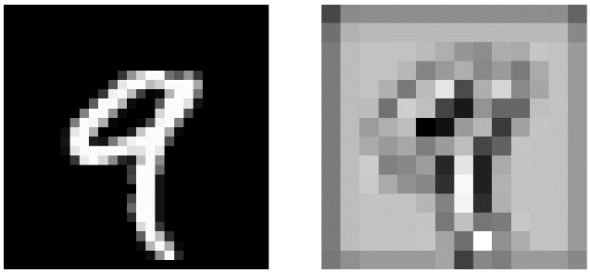

Fig. 6. MNIST Digits sample data before applying CNN filter (left) and after applying CNN filter (right).

accuracy comes from the Index-57 kernel. The kernel as feature extraction yields the highest increase in accuracy for the CIFAR-10 and the batik collections dataset. Figure 9 shows the sample images from the CIFAR-10 and batik collections dataset before and after passing the Index-57 kernel of the ResNet-50 CNN layer.

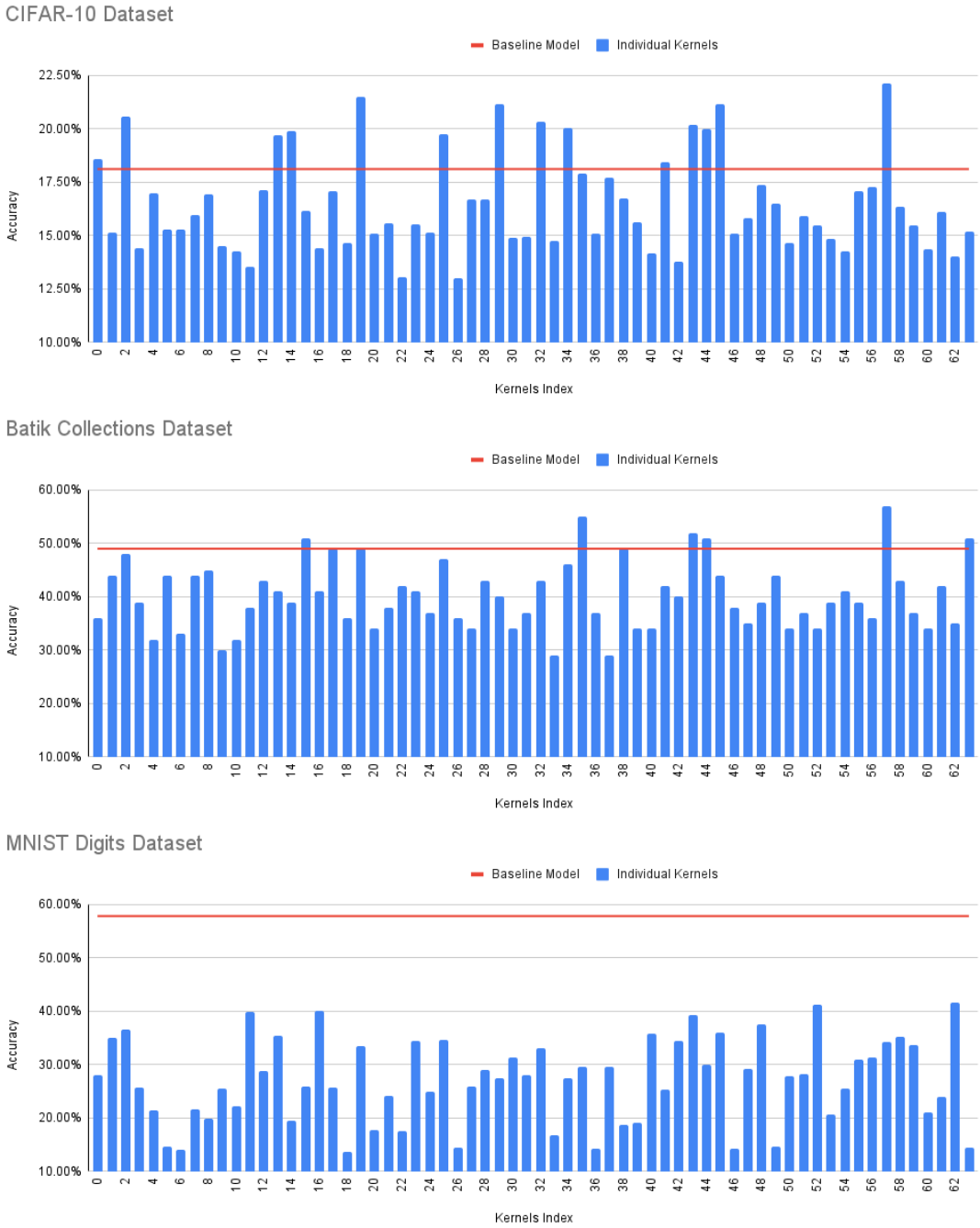

Fig. 7. Result from applying individual kernel from the first convolutional layer in ResNet-50.

C. Disparity between the Lowest and Highest Performing ResNet-50 Kernels

From the previous experiment, the Index-57 kernel from the first CNN layer of ResNet-50 yields an 8.00% increase for batik collections and a 4.02% increase for the CIFAR-10 dataset. Then, the researchers take a closer look by visualizing the kernel that produces the result. Figure 10 shows the representation of the kernel after being normalized.

The researchers observe the kernel and find that it resembles a Gaussian blur kernel. There is also a strong blue tint. It shows that the ratio of the blue channel is higher than the red and green channels. The kernel is also seen slightly offset from the center and leans

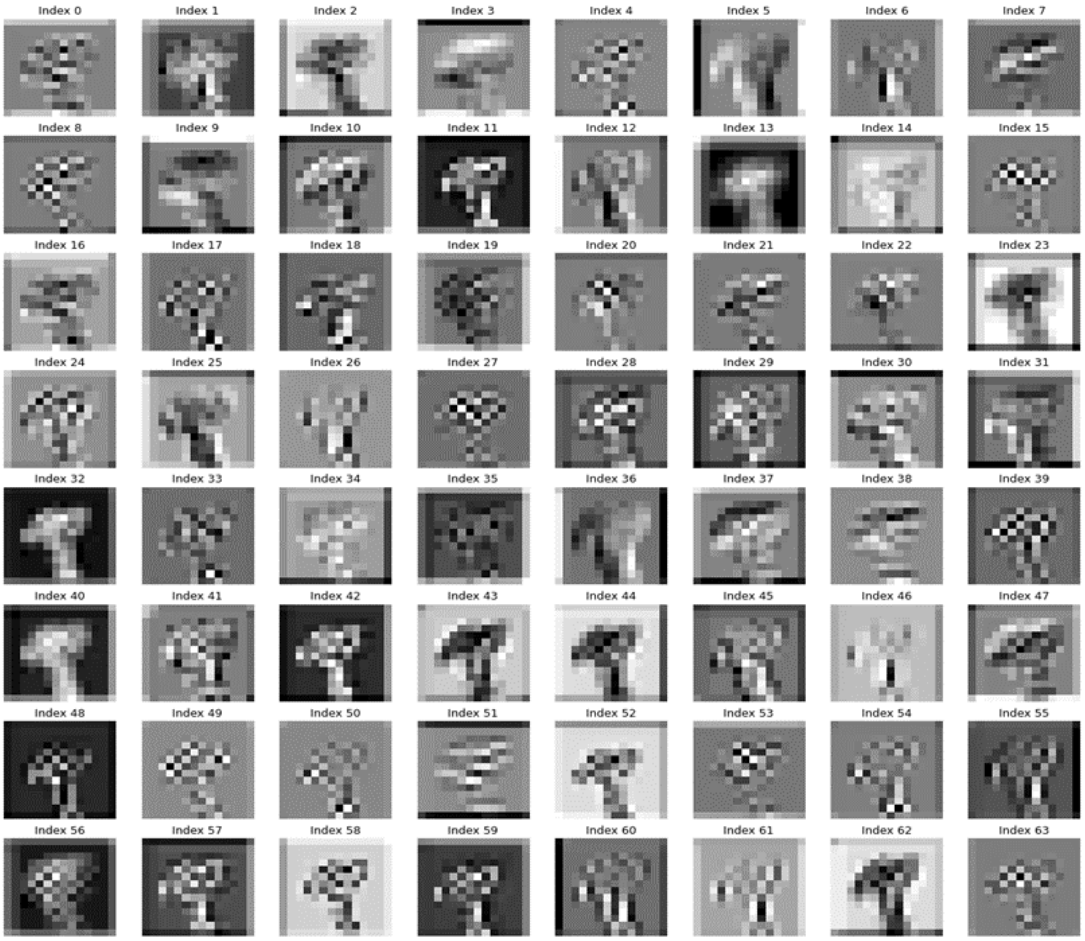

Fig. 8. Feature map of MNIST Digits sample data.

towards the right.

Compared to the lowest-performing kernels for the CIFAR-10 and batik collections, the Index-26 kernel and Index-33 resemble an edge detection kernel, as shown in Fig. 11. Index-26 kernel appears to be a vertical edge detector. Meanwhile, the Index-33 kernel is a diagonal edge detector.

Next, the researchers also observe which kernels are responsible for increasing accuracy in the CIFAR-10 and batik collections dataset. As shown in Fig. 12, the Index-57 kernel yields the highest increase among other kernels in the CIFAR-10 dataset. The second kernel, the Index-19 kernel, also resembles a Gaussian blur kernel with a green tint. The Index-45 kernel shows similarity to an edge detector kernel with a blue tint. Moreover, the Index-29 kernel resembles a Gaussian blur kernel with a purple tint. The Index-2 kernel shows a complex arrangement of the multi-colored kernel. Then, the Index-32 kernel also resembles a Gaussian blur kernel with a blue tint.

As shown in Fig. 13, the Index-57 kernel has the highest increase in accuracy for the batik collections dataset. The second kernel, the Index-35 kernel, shows a resemblance to a Gaussian blur kernel with a pink tint. Then, the Index-43 kernel resembles an edge detector kernel with a blue and orange tint. The Index-63 kernel seems to be in the spectrum of the Gaussian blur kernel and edge detector kernel. Index-44 resembles a Gaussian blur kernel with a purple tint. Meanwhile, the Index-15 kernel resembles an edge detector kernel.

D. Discussion

In the experiments, the researchers have observed the effects of the pre-trained CNN layer as a feature extractor. Out of the three datasets used, one interesting fact is that the dataset with the lowest complexity, which is MNIST Digits, does not always have better accuracy when used as an input for the study design. Initially, the researchers decide to experiment using three datasets with varying levels of complexities to obtain comparisons between them. However, it is found

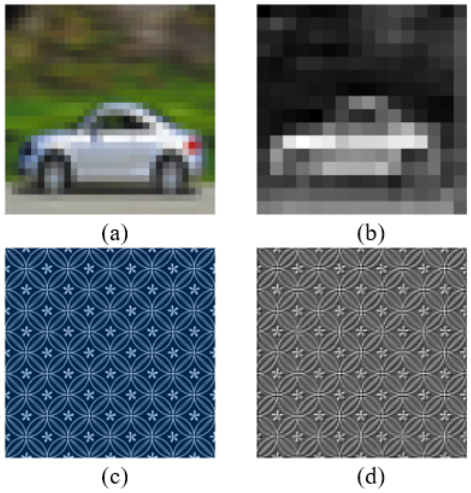

Fig. 9. Sample images from (a) CIFAR-10 before, (b) CIFAR-10 after, (c) Batik collections before, and (d) Batik collections after passing into the Index-57 kernel from the first CNN layer of ResNet-50.

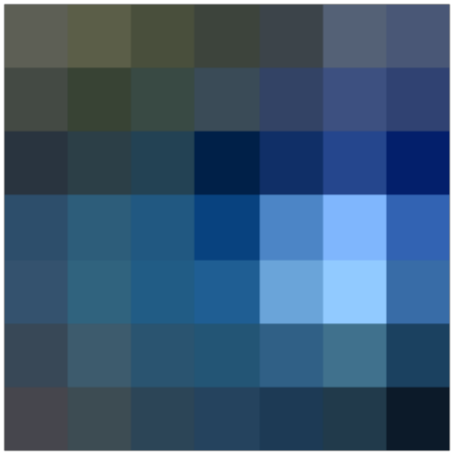

Fig. 10. Visualized Index-57 kernel.

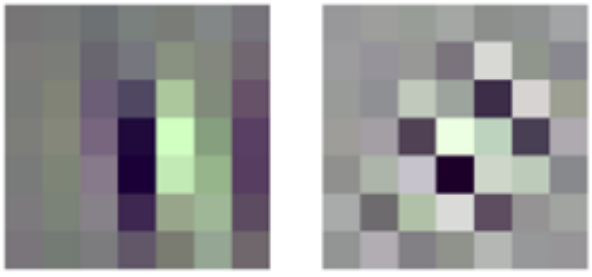

Fig. 11. Index-26 kernel (left) and Index-33 (right).

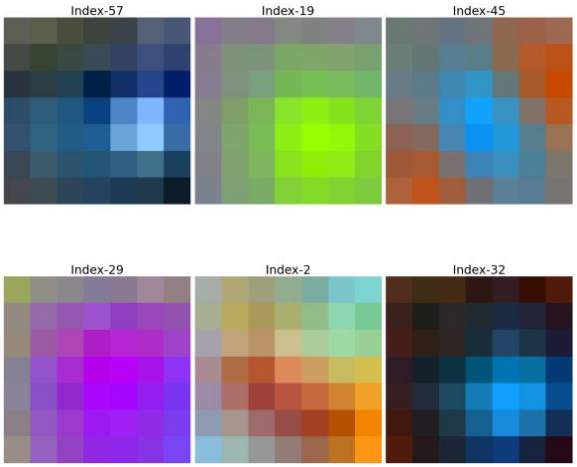

Fig. 12. Top six kernels with an increase in accuracy for the CIFAR-10 dataset.

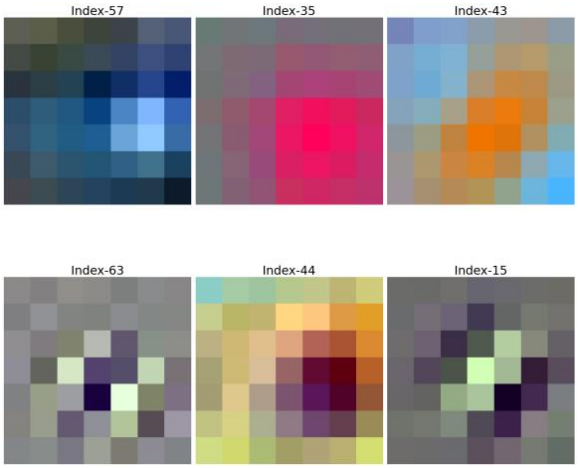

Fig. 13. Top six kernels with an increase in accuracy for the batik collections dataset.

that color is also an essential factor when working with a pre-trained CNN layer, specifically for ResNet-50.

They mostly resemble a Gaussian blur kernel looking at the kernels with the highest accuracy in the experiment when using individual kernels. The Gaussian blur has a property in which, when applied to an image, it will reduce noise while also reducing the detail of the original image. It goes in line with the fact that most of the top six kernels with the highest accuracy increase in CIFAR-10 and batik collections resemble a Gaussian blur kernel. The datasets consist of complex and noisy data. A Gaussian blur kernel can reduce the noise in the image so that the feature extraction process can extract meaningful features better. It complements the color of the image, which plays an important role when working with a pre-trained model.

On the other hand, applying the Gaussian blur kernel to the MNIST Digits dataset will in turn contribute to lowering the accuracy. Since the dataset is already low in complexity and has well-defined edges, using the

Gaussian blur kernel will reduce the detail of these edges. It is shown in the MNIST Digits feature map after passing it onto the CNN layer. Thus, it hinders the performance of the model.

The most notable thing found from experimenting with individual kernels using the CIFAR-10 dataset is the prevalence of kernels with a blue tint. It increases the accuracy of the baseline model. Since the CIFAR-10 dataset consists of data with a blue background, such as in ship or airplane class, it suggests that there can be a bias in the data. It shows why the model performs well when using biased data.

IV. Conclusion

The proposed study design aims to make sense of what happens inside a CNN by using a pre-trained CNN layer as a features extractor for detecting bias in image classification data. The research finds that color plays an essential role in CNN. Moreover, a bias in the dataset is discovered in the highest performing kernels in the pre-trained CNN layer when used as a features extractor. It is based on the color of the images in the dataset. Detecting this bias early on is important in helping developers to improve AI algorithms.

Further research should address the limitations of the study design. The most notable limitation is related to finding the image color in the dataset. In the study design, the researchers go with the approach of observation of the color, instead of quantifying the color. Further research should make this issue a primary consideration by also performing a more systematic approach and exploring different kinds of pre-trained CNN models.

References

[1] S. Leavy, "Gender bias in artificial intelligence: The need for diversity and gender theory in machine learning," in *Proceedings of the 1st International Workshop On Gender Equality in Software Engineering*, 2018, pp. 14–16.

[2] D. J. Fuchs, "The dangers of human-like bias in machine-learning algorithms," *Missouri S&T's Peer to Peer*, vol. 2, no. 1, pp. 1–14, 2018.

[3] J. Dastin, "Amazon scraps secret AI recruiting tool that showed bias against women," 2018. [Online]. Available: https://reut.rs/2UghQQS

[4] A. Chouldechova, D. Benavides-Prado, O. Fialko, and R. Vaithianathan, "A case study of algorithm-assisted decision making in child maltreatment hotline screening decisions," in *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*. PMLR, 2018, pp. 134–148.

[5] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[6] S. Alelyani, "Detection and evaluation of machine learning bias," *Applied Sciences*, vol. 11, no. 14, pp. 1–17, 2021.

[7] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," *arXiv Preprint arXiv:1901.04966*, 2019.

[8] W. Sun, O. Nasraoui, and P. Shafto, "Evolution and impact of bias in human and machine learning algorithm interaction," *PLOS ONE*, vol. 15, no. 8, pp. 1–39, 2020.

[9] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*. Antalya, Turkey: IEEE, Aug. 21–23, 2017, pp. 1–6.

[10] B. B. Traore, B. Kamsu-Foguem, and F. Tangara, "Deep convolution neural network for image recognition," *Ecological Informatics*, vol. 48, pp. 257–268, 2018.

[11] L. Shang, Q. Yang, J. Wang, S. Li, and W. Lei, "Detection of rail surface defects based on CNN image recognition and classification," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*. Chuncheon, South Korea: IEEE, Feb. 11–14, 2018, pp. 45–51.

[12] R. Chauhan, K. K. Ghanshala, and R. C. Joshi, "Convolutional Neural Network (CNN) for image detection and recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*. Jalandhar, India: IEEE, Dec. 15–17, 2018, pp. 278–282.

[13] B. P. Gyires-Tóth, M. Osváth, D. Papp, and G. Szűcs, "Deep learning for plant classification and content-based image retrieval," *Cybernetics and Information Technologies*, vol. 19, no. 1, pp. 88–100, 2019.

[14] Y. Wang, H. Liu, M. Guo, X. Shen, B. Han, and Y. Zhou, "Image recognition model based on deep learning for remaining oil recognition from visualization experiment," *Fuel*, vol. 291, pp. 1–14, 2021.

[15] X. Yang, Y. Zhang, W. Lv, and D. Wang, "Image recognition of wind turbine blade damage based on a deep learning model with transfer learning and an ensemble learning classifier," *Renewable Energy*, vol. 163, pp. 386–397, 2021.

[16] S. Hicks, M. Riegler, K. Pogorelov, K. V. Anonsen, T. de Lange, D. Johansen, M. Jeppsson, K. R.

Randel, S. L. Eskeland, and P. Halvorsen, "Dissecting deep neural networks for better medical image classification and classification understanding," in *2018 IEEE 31st International Symposium on Computer-Based Medical Systems (CBMS)*. Karlstad, Sweden: IEEE, June 18–21, 2018, pp. 363–368.

[17] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–42, 2019.

[18] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Science and Information Conference*. Las Vegas, USA: Springer, April 25–26, 2019, pp. 128–144.

[19] S. T. Krishna and H. K. Kalluri, "Deep learning and transfer learning approaches for image classification," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 7, no. 5S4, pp. 427–432, 2019.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[21] J. Wang, Y. Ma, L. Zhang, R. X. Gao, and D. Wu, "Deep learning for smart manufacturing: Methods and applications," *Journal of Manufacturing Systems*, vol. 48, pp. 144–156, 2018.

[22] A. Krizhevsky, *Learning multiple layers of features from tiny images*. University of Toronto, 2009.

[23] Y. LeCun, C. Cortes, and C. J. C. Burges, "The MNIST database of handwritten digits," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[24] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, 1962.

[25] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.