International Journal on Advanced Science Engineering Information Technology

Vol.13 (2023) No. 6
ISSN: 2088-5334

Fine Tuned of DenseNET121 to Classify NTT Weaving Motifs on Mobile Application

Yohanes Eudes Hugo Maur [a], Albertus Joko Santoso [a,*], Pranowo [a]

[a] *Magister Informatika, Universitas Atma Jaya, Yogyakarta, Indonesia*
Corresponding author: [*]albjoko@staff.uajy.ac.id

***Abstract*—The problem of classifying Woven Fabric Motifs through pattern recognition can be addressed using Convolutional Neural Networks (CNNs). Existing CNN architectures like VGG, ResNet, MobileNet, and DenseNet offer diverse propagation methods. These architectures, trained on datasets like imagenet, have demonstrated competence in solving large-scale classification tasks. The CNN model trained on the ImageNet dataset, hereinafter referred to as the pre-trained model, can be utilized to address the classification issue of NTT woven fabric motifs. This involves retraining the model using a new output layer and dataset, a method known as Transfer Learning. In addition to Transfer Learning, this research employs Fine Tuning, which entails retraining several classification layers. The pre-trained model used in this research is DenseNet121. This model was chosen because it does not require too much storage space and has good classification performance so that it can be embedded in smartphones. The results of this study indicate that of the three pre-trained models tested (DenseNet121, MobileNetV2, and ResNet50V2), the pre-trained Model DenseNet121 is the model that has the highest accuracy and the smallest loss, namely 92.58% accuracy and 29.62% Loss. Tests on mobile devices also show that from 130 test data, this model gets an accuracy of 99.23%. Overall, the classification model of NTT woven fabric motifs embedded in mobile devices can be used as an alternative to help the community or people who want to learn about NTT woven fabric motifs.**

***Keywords*— Transfer learning; fine tuning; mobile application; NTT woven fabrics**

Manuscript received 18 Sep. 2022; revised 22 Aug. 2023; accepted 6 Nov. 2023. Date of publication 31 Dec. 2023.

I. INTRODUCTION

Woven fabric is one of the artistic and cultural heritage of the Nusantara and the world community. Woven fabrics are usually used in formal events such as traditional ceremonies and wedding receptions. In Nusa Tenggara Timur (NTT) the tradition of weaving is usually done by a woman who fills their spare time in the dry season. The motifs of woven fabrics produced in NTT are generally abstract and have a philosophy inherited from their ancestors [1]. For ordinary people, distinguishing the motifs of woven fabrics is not easy, especially to know the meaning of these motifs. Information about woven fabric motifs in the NTT area is usually only known by older adults who used to make woven fabrics and people who are experts. For information about the motifs of NTT woven fabrics to be well documented, the information must be stored and processed using the Information Technology platform.

In this study, efforts to solve the pattern recognition problem of NTT Weaving motifs were carried out using an algorithm from Deep Learning. Deep Learning is one of the most effective areas of Artificial Intelligence for solving image classification problems [2]–[4]. This method can also be ascertained to perform image classification of woven fabric motifs effectively. In deep learning, neural networks commonly used to complete tasks related to image classification and pattern recognition are called convolutional neural networks (CNN) [4]. CNN works by imitating the Visual Cortex in the brain to recognize images [5]. The use of CNN is what makes Deep Learning more effective in solving Image classification problems when compared to traditional Machine Learning methods. This is because CNN already has a Feature Extractor, while in Traditional Machine Learning, the Feature Extractor technique must be designed from scratch. The main focus of this research is its application to the regional arts and culture field.

In the field of computer vision, CNN can be applied to various fields such as agriculture [6]–[9], health [10]–[14]. Rasyidi et al. [15] using the CNN algorithm to classify the types of batik cloth. The number of datasets used is 944 batik motifs which are categorized into six classes, namely Banji, Ceplok, Kawung, Mega Mendung, Parang, and Sekar Jagad.

Various CNN architectures, such as AlexNet, DenseNet, Resnet, SquezeNet, and VGG, were also compared to solve problems related to the classification of batik motifs. The results of this study indicate that the DenseNet architecture is the best in this study, with an accuracy of 94%. As for other studies related to the classification of batik motifs using the ResNet Architecture as conducted by Negara et al. [16], CNN architecture can also be formulated independently as done by Azhar et al. [17]. This research uses 3 Convolution Layers interspersed with Max Pooling Layer, 2 Dropout Layers, and 2 Dense Layers. The activation function used for the Convolution layer is ReLU. The results of this study indicate that the model that has been developed and trained on the Batik300 and Batik41K datasets has an accuracy of up to 100% in the testing process.

CNN will be influential if the data used for the training process is available in large quantities. Transfer Learning and Fine-tuning are solutions to overcome problems related to developing CNN models from scratch [18]. Transfer Learning is a technique for transferring features from the Convolutional Neural Network (CNN) architecture that has been engineered to solve image classification problems with many classes [19]. At the same time, Fine Tuning is a technique used to improve classification performance when using Transfer Learning by re-weighting the top few layers of the pre-trained model.

Sreenivasulu et al. [20] proved that the neural network model developed with the transfer learning approach (pre-trained inception model) has better accuracy than the CNN neural network model developed from the beginning, where the accuracy was obtained with the transfer learning approach (pre-trained inception model). -trained model Inception) has an accuracy of 95.23%, while the CNN neural network model which was developed from scratch only has an accuracy of 91.32%.

Research conducted by Hussain et al. [21] utilizing Transfer Learning to recognize patterns on woven fabrics which consist of 3540 image data and are grouped into 3 data classes. The pre-training models used for comparison are ResNet-50 and VGGNet. The results of this study indicate that the Pretrained ResNet-50 model (99.3% accuracy) has better-woven fabric pattern classification performance than the VGGNet model (92.4% accuracy).

In 2021, Rasyidi et al. [22] used Transfer Learning to identify Batik fabric-making techniques. The pre-trained models used in this study include ResNet, DenseNet, and VGG with Batch Normalization. This study uses 120 datasets, which are divided into four classes. Each existing data is then split into 30 smaller images. The original DenseNet161 dataset has the best performance with an accuracy of 79.71%, while the modified Pretrained Model VGG13 with Batch Normalization has the best accuracy of 87.61%, while for DenseNet161 the accuracy is only 82.78%.

As mentioned earlier, research related to the application of Deep Learning and Transfer Learning for image classification tasks has not yet reached the implementation stage on systems that can be accessed by the wider community, such as web-based or mobile applications. Harjoseputro et al. [23] utilized the Mobile Client-Server Architecture and Convolutional Neural Network to classify Javanese letters. This study utilizes a mobile client-server architecture so that the computational load of Java letter classification can be carried out on the server. The model developed by Harjoseputro et al. has a fairly large size and is only suitable to be applied to the server, so this model still requires an internet connection. The same technique has also been used by Lanjewar et al. [24] to classify diseases of tea leaves.

The model needs to be compressed for the developed Neural Network model to be embedded into mobile devices. If the Neural Network model developed uses the tensorflow library, then the model can be compressed into tensorflow lite form. Lee et al. [25] used this technique to classify arrhythmia. The model developed in this study can be embedded into a wearable device.

The research that the author himself will carry out is the image classification of woven fabric motifs from the NTT area using the Fine-Tuning approach from the CNN DenseNet121 architecture. The classification model that has been trained is then compressed to be embedded into the smartphone.

The main contributions of this research are as follows:

- We adapted the Transfer Learning technique which was improved with Fine Tuning to classify the motifs of NTT woven fabrics which were divided into 13 classes, Buna Ayotupas, Buna Insana, Krawang Nunkolo, Kaif Berantai Nunkolo, Futus Amarasi, Futus Biboki, Lotis Bebnisse, Lotis Biklusu, Naisa Pahat, Andungu, Biklusu, Mamuli, and Sotis.
- We will also compare several pre-trained models such as MobileNetV2, ResNet50V2, and DenseNet121 in solving problems related to image classification of woven fabric motifs.
- The application that has been developed in this research can be used by the World community and anyone who wants to learn about NTT Woven Fabrics.

II. Materials and Method

A. Transfer Learning and Fine Tuning

Transfer Learning is an enhancement of Machine Learning on a new task through the transfer of knowledge features of similar tasks that have been learned [19]. The application of transfer learning has also been proven to be more effective and efficient when compared to deep learning model learning from the start [26],[27],[28],[29]. Figure 1 illustrates the application of Transfer Learning to solve new problems.

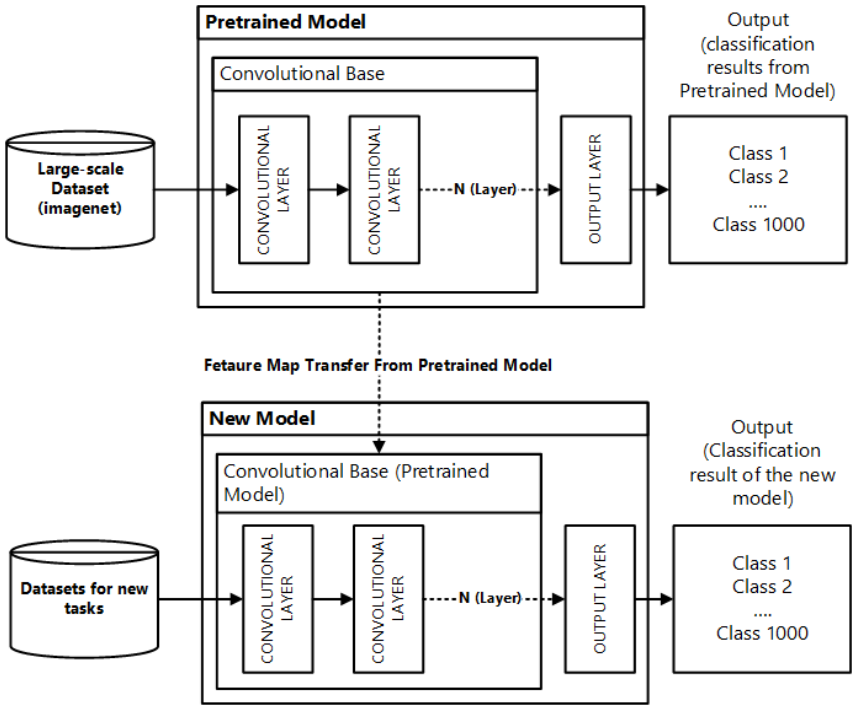

Fig. 1 Transfer Learning and Fine Tuning

Based on Figure 1, a learning model that has been trained using a large number of datasets, the pre-trained model extracts its knowledge or feature map to be used in solving new problems. In this study, the type of Transfer Learning used is Transductive Transfer Learning [30], [31], where the source and target domains are different but have the same task. Transfer Learning is usually used to overcome problems related to limited computing resources and the lack of datasets [22].

The pre-trained model consists of two parts, namely the Convolutional Base and Output Layer. When implementing Transductive Transfer Learning, the output layer is not extracted. This is because the previous layer output will show the classification results based on the classes in the large-scale dataset. For that, the new layer output needs to be re-engineered in the new model so that the resulting output can correspond to the existing classes in the new dataset. The performance of the Deep Learning model that uses the Transfer Learning approach can also be improved. This increase is done by fine-tuning or re-weighting some parts of the Feature Map from the Convolutional base pre-trained model.

Re-weighting or fine-tuning does not occur in all parts of the Convolutional base. The weighting only occurs in the top few layers. This is intended to reduce overfitting problems because in Deep Learning, if the number of convolutional layers being trained increases, then the trained Deep Learning model will study the dataset too deeply, so the engineered model will also study noise.

B. DenseNet

DenseNet is one of the discoveries in neural networks for visual object recognition [32]. DenseNet is very similar to ResNet [33] but has some basic differences. ResNet uses the additive (+) method which combines the previous layer (identity) with the future layer, whereas DenseNet combines the (.) output of the previous layer with the future layer.

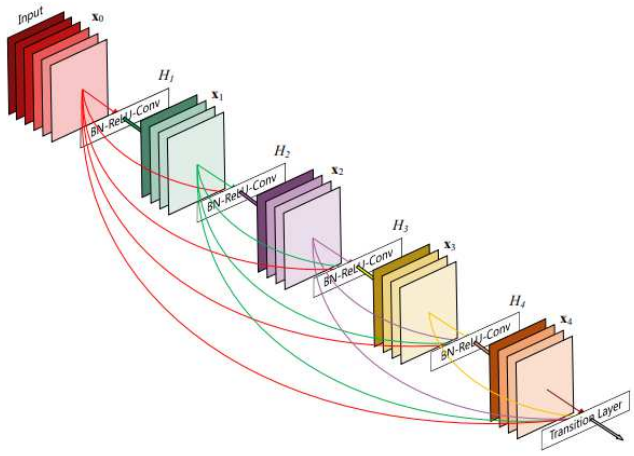

Fig. 2 DenseNet Connection [32]

The traditional convolutional feedforward network connects the output of the th-layer as input to the next layer ($\ell + 1$)th layer, where the layer transition becomes: $x\ell = H\ell(x\ell-1)$. In the ResNet CNN architecture, a skip connection is added, which passes a non-linear transformation with an identity function:

$$x\ell = H\ell(x\ell - 1) + x\ell - 1 \quad (1)$$

An advantage of ResNets is that the gradient can flow directly through the identity function from later to earlier layers. However, the identity function and the output of Hℓ are combined by summation, which may impede the information flow in the network.

In DenseNet Convolutional Neural Network Architecture, the connection patterns between layers are linked directly from any layer to all subsequent layers. Figure 2 illustrates the

schematically generated DenseNet layout. As a result, the th-layer receives the feature map of all previous layers, x0, . . . , xℓ−1, as input:

$$x\ell = H\ell([x0, x1, \ldots, x\ell - 1]), \quad (2)$$

where [x0,x1,.., xℓ−1] refers to the feature-map set produced in layer 0, …, 1. Because the connectivity between layers is quite dense, the CNN architecture is referred to as DenseNet. DenseNet Has various variants (DenseNet-121, DenseNet160, DenseNet-201), according to its number of layers. In this research, the DenseNet architecture used is DenseNet121. Details of the DenseNet121 is following: DenseNet121 = 5+(6+12+24+16) *2, where:
5 – Convolution and Pooling Layer
3 – Transition Layers (6, 12, 24)
1 – Classification Layer (16)
2- Dense Block (1*1 and 3*3 convolutional layer)

C. Data Collection and Processing

This study collected datasets from the Craft House belonging to the East Nusa Tenggara Regional National Craft Council, the Ina Ndao 2 Weaving Production House, and others sourced from the Internet. Images of each woven fabric motif are taken using a camera with a resolution of 25 MP. Figure 3 is an example of a sample dataset used.

Fig. 3 Dataset Sample

The dataset that has been collected for the training and testing process is 1729 with the following details in Table 1:

TABLE I
DATASET INFORMATION

No	Pattern	Total
1	Buna Ayotupas	154
2	Buna Insana	112
3	Krawang Nunkolo	143
4	Kaif Berantai Nunkolo	155
5	Futus Amarasi	139
6	Futus Biboki	146
7	Lotis Bebnisse	144
8	Lotis Biklusu	130
9	Naisa Pahat	130
10	Andungu	105
11	Kurangu	102
12	Mamuli	126
13	Sotis	143
	TOTAL	**1729**

For the developed neural network model to study the dataset optimally, we add a synthetic image generated through the data augmentation process. This technique is intended to increase the variety of training data. This method has also been proven effective in improving the performance of Deep Learning Models that require large-scale data [34]–[36]. The augmentation process is carried out by randomly applying rotation, flip, and crop and adding Gaussian noise to the data to be augmented. The data augmentation process is carried out using the Augmentor library [37].

D. CNN Model Development

The neural network model was developed using Transfer Learning techniques. The concept of developing this model can be seen in Figure 1. In this phase, the pre-trained model chosen is DenseNet121, which was chosen because its size is not large but has a high enough accuracy and computation time, which is relatively low. When using transfer learning to solve a new problem, the first step must be to freeze the convolutional base of the pre-trained model so that it is not re-weighted and to engineer a new output layer.

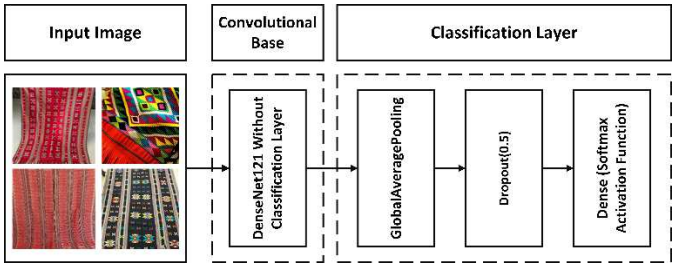

Fig. 4 Developed CNN model

The engineered output layer consists of 1 pooling layer, 1 dropout layer, and 1 dense or fully connected layer, with the activation function used being SoftMax. The SoftMax activation function on Dense Layer is intended to display a list of classification results with a probability level of confidence. Equation 3 is the formulation of SoftMax:

$$(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \quad (3)$$

$\sigma = Softmax,$
$\vec{z} = input,$
$e^{z_i} = standart\ exponential\ for\ input\ vector,$
$e^{z_j} = standart\ exponential\ for\ output\ vector,$
$K = num\ classes$

After the classification model is ready, the next step is to compile the model. At the compilation stage, the Optimizer used is Adam. The Adam optimizer is intended to make the learning rate-adaptive, along with changes in loss generated for each iteration [38]. This Optimizer has also been proven to provide optimal performance on neural network models [39]. Because the number of weaving motif classes is more than two, the Loss function used is Categorical Cross Entropy. The following is the formulation of the Categorical Cross Entropy loss function:

$$Loss = -\sum_{i=1}^{K} y_i \log(\hat{Y}_i) \quad (4)$$

$y_i = True\ Output, \hat{y}_i = Predicted\ Output$

E. Model Compression

This study uses Tensorflow as a library for developing Neural Network models so that the process of converting the Model into TFLite form (Tensorflow Lite) is also assisted by a library from Tensorflow. Figure 5 is a schematic of converting the Deep CNNs model into TFLite form.

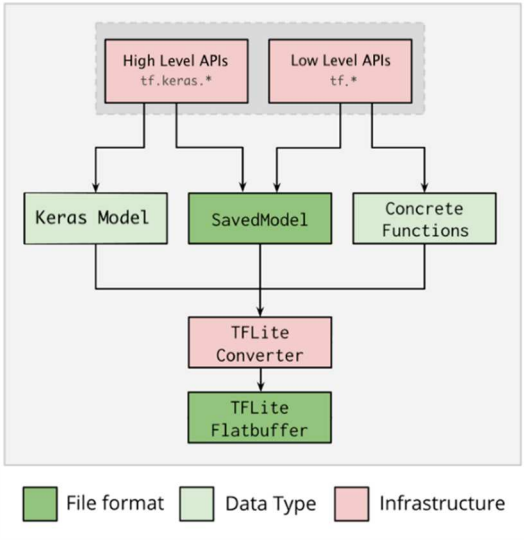

Fig. 5 Model Conversion Scheme into TFLite form [40]

When the Deep CNNs model has been trained and gets optimal accuracy, the model is first saved in the form of .pb or .h5. After the model is saved, the model will be converted using a TFLite Converter, and the output of the conversion of this model is a TFLite Flatbuffer, later known as.TFLite. The classification model that has been developed can be applied to mobile devices based on Android and IoS, or to microcontrollers because the Classification model developed has been converted into TFLite form, which is supported in several programming languages such as Java, Swift, Objective-C, C++, and Python. This conversion of the Model into TFLite form has several advantages, namely the absence of communication between the server and client so that the classification process can be carried out without internet connectivity, performance improvement with hardware acceleration, and model optimization.

F. Mobile Apps Development

For this study, we implemented the model on an Android-based mobile device. Android application coding uses the Java programming language, and the IDE is Android Studio. The architectural design of the application to be developed can be seen in Figure 6.

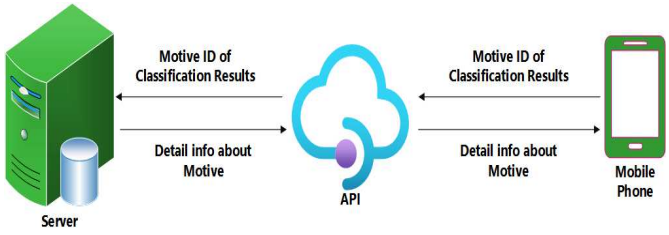

Fig. 6 Mobile Application Architecture

In conditions without an internet connection, this application can still classify because the classification process occurs on mobile devices, but detailed information about woven fabric motifs cannot be displayed because the information is on the server. Information stored on the server includes information about the motif's origin, manufacturing technique, purpose of use, and a description of the weaving motif. The last stage in developing this mobile application is to evaluate the accuracy and average computation time required during the classification process.

III. Result and Discussion

A. CNN Model Development

The development of the neural network model was carried out using a computer with the following specifications: Processor Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz, NVIDIA GeForce 930MX 2GB GPU, and 8192 MB RAM. The neural network model that has been constructed is then trained to learn all the weights and biases from the NTT woven fabric data. This training process aims to minimize the loss generated by the model, where loss is the distance between the actual data and the predicted results. The training is carried out in stages, starting from training the model constructed using transfer learning and following the fine-tuning process. The number of epochs used during the Transfer Learning training process is 50, and the Optimizer used is Adam, with a Learning Rate of 1 x 10^{-3}. To measure the effectiveness of the DenseNet121 pre-trained model in the Transfer Learning process, the same data will also be used as material for training the CNN model, which was developed from scratch, and several other pre-trained models:

TABLE II
TRANSFER LEARNING ACCURACY

Method	Train Acc	Val Acc	Test Acc
CNN from scratch	100	54.37	55.49
ResNet50V2	86.75	82.69	81.70
MobileNetV2	81.71	78.53	79.57
DenseNet121	72.6	81.55	78.45

TABLE III
TRANSFER LEARNING LOSS

Method	Train Loss	Val Loss	Test Loss
CNN from scratch	0.004	472	480
ResNet50V2	37.72	57.94	56.07
MobileNetV2	54.79	63.93	65.12
DenseNet121	80.02	60.69	64.62

Based on the data in Table II and Table III, it can be seen that the CNN model that was developed from scratch is overfitting. This can be seen from the accuracy and loss generated in the training process, which is much different from the results of the model evaluation in the model evaluation data on the validation data. The best classification performance is obtained from Transfer Learning with the pre-trained Model ResNet50V2 where this model has the highest accuracy value and the smallest loss. The model evaluation results on the test data also show that the model developed using the Transfer Learning approach with the pre-trained ResNet50V2 model has the highest accuracy and the lowest loss. Models that have been trained with the Transfer Learning approach can be improved with the Fine-Tuning

approach or the process of re-weighting the top few layers of the Pre-Trained Model. The process of applying Fine Tuning will provide more training time because there will be several CNN layers that will be retrained. Table IV is a detailed parameter that is trained and generated by the model developed from the pre-trained model to be tested:

TABLE IV
TRAINABLE PARAMETER

Method	Trainable TL Params	Trainable FT Params	Total Params
ResNet50V2	26,637	7,906,317	23,591,437
MobileNetV2	16,653	429,453	2,274,637
DenseNet121	13,325	181,453	3,442,701

At this stage, the Fine-Tuning process is carried out on the Top 10 Layers of the pre-trained Model. Because the propagation formula for each pre-trained model is different, the number of parameters being trained is also different. The retrained parameter is the number of parameters generated by the Convolution layer only. This is because the weighting process only occurs in the convolution layer. To determine the effectiveness of Fine Tuning, Table V, and Table VI provides information related to model training using Fine Tuning:

TABLE V
FINE-TUNING ACCURACY

Method	Train Acc	Val Acc	Test Acc
ResNet50V2	86.75	82.69	81.70
MobileNetV2	98.76	83.51	84.95
DenseNet121	98.62	92.41	92.58

TABLE VI
FINE-TUNING LOSS

Method	Train Loss	Val Loss	Test Loss
ResNet50V2	37.72	57.94	56.07
MobileNetV2	3.65	80.31	82.29
DenseNet121	4.02	28.91	29.62

TABLE VII
TRAINING DURATION

Method	Transfer Learning Time	Fine Tuning Time
ResNet50	6:26:27	7:58:48
MobileNetV2	2:00:30	2:09:11
DenseNet121	7:30:44	7:43:53

The model that has been applied to fine-tuning is then retrained using 50 epochs and a learning rate of 1 x 10^{-4}. The size of the learning rate is reduced by 10% to prevent the model from reaching convergence at the beginning of the training process. Based on the information in Table VI and Table VII, it can be seen that the application of Fine Tuning can improve the performance of neural network models that utilize Transfer Learning techniques. Fine Tuning on the Top 10 Layers of the pre-trained DenseNet121 model has the best classification performance. In the pre-trained Model of MobilenetV2, the accuracy produced is quite large, but the resulting loss is also quite large. In the pre-trained Model ResNet50V2 the resulting changes are not significant enough the resulting accuracy only increases by 3.25%, and the resulting loss remains the same.

In terms of training time (can be seen in Table VII) MobileNetV2 has the smallest computational time both during the Transfer Learning and Fine-Tuning processes, the number of layers contained in the pre-trained Model MobileNetV2 is also only 154, ResNet50V2 is 190, and DenseNet121 is 427. In the Transfer Learning stage, the number of layers will significantly affect the computation time, while during Fine Tuning, the number of parameters being trained will significantly impact the training time. The model that has been trained is then compressed into a TFLite form so that it can be tested on a mobile device.

B. Mobile Application Development

The neural network model that has been embedded in the mobile application will be tested on devices with 8 Core processor specifications with a speed of 449 - 1768 MHz and 4GB of RAM. When making data predictions, the engineered mobile application will display the top 3 classes of classification results and the percentage confidence level. The class with the highest level of confidence will display detailed information, as shown in Figure 7.

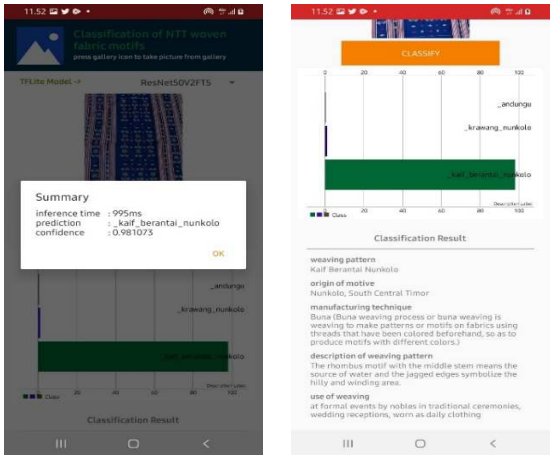

Fig. 7 Application Result in Mobile Phone

The model will be tested using test data to find out how well the neural network model performs classification on mobile devices. The test data used in this study amounted to 130 data, with each motif class having 10 images. Table VIII to Table X results from testing the model on mobile devices when classifying.

TABLE VIII
RESNET50V2 TESTING RESULT

Pattern	ResNet50V2		
	precision	recall	f1-score
_amarasi	1,00	1,00	1,00
_andungu	0,91	1,00	0,95
_ayotupas	1,00	0,90	0,95
_bebnisse	1,00	1,00	1,00
_biboki	1,00	0,90	0,95
_biklusu	1,00	1,00	1,00
_insana	1,00	1,00	1,00
_kaif_berantai _nunkolo	1,00	0,90	0,95
_krawang _nunkolo	0,91	1,00	0,95
_kurangu	0,82	0,90	0,86
_mamuli	1,00	0,90	0,95
_naisa	1,00	1,00	1,00
_sotis	0,91	1,00	0,95
Accuracy		**0,96**	

TABLE IX
MOBILENETV2 TESTING RESULT

Pattern	MobileNetV2		
	precision	recall	f1-score
_amarasi	0,91	1,00	0,95
_andungu	0,91	1,00	0,95
_ayotupas	0,82	0,90	0,86
_bebnisse	1,00	1,00	1,00
_biboki	1,00	0,70	0,82
_biklusu	1,00	1,00	1,00
_insana	0,83	1,00	0,91
_kaif_berantai _nunkolo	0,83	1,00	0,91
_krawang _nunkolo	1,00	0,90	0,95
_kurangu	1,00	0,90	0,95
_mamuli	1,00	1,00	1,00
_naisa	1,00	0,90	0,95
_sotis	1,00	0,90	0,95
Accuracy		**0,94**	

TABLE X
DENSE121 TESTING RESULT

Pattern	DenseNet121		
	precision	recall	f1-score
_amarasi	1,00	1,00	1,00
_andungu	1,00	1,00	1,00
_ayotupas	1,00	1,00	1,00
_bebnisse	1,00	1,00	1,00
_biboki	1,00	1,00	1,00
_biklusu	1,00	1,00	1,00
_insana	1,00	1,00	1,00
_kaif_berantai _nunkolo	0,91	1,00	0,95
_krawang _nunkolo	1,00	0,90	0,95
_kurangu	1,00	1,00	1,00
_mamuli	1,00	1,00	1,00
_naisa	1,00	1,00	1,00
_sotis	1,00	1,00	1,00
Accuracy		**0,99**	

Based on the data in Table VIII to Table X, it can be seen that the pre-trained DenseNet121 model has the most optimal accuracy, followed by ResNet50V2 and MobileNetV2. Of the 130 images tested, the DenseNet121 Model only has one prediction error, with one image of the Krawang Nunkolo motif, which the model predicts as a Kaif Berantai Nunkolo motif. Hence, the precision of the model prediction for the Kaif Berantai Nunkolo motif decreases and recalls the Krawang Nunkolo motif.

After looking at the model's performance for classifying, the next step is to check the resource usage required by the application when running on a mobile device. Table XI summarizes the resource usage the application requires when running on a mobile device.

TABLE XI
RESOURCES USAGE EVALUATION OF EACH PRE-TRAINED MODEL

Component	Pre-Trained Model		
	ResNet50V2	MobileNetV2	DenseNet121
CPU	21%	**19%**	23%
RAM	221MB	**115MB**	166MB
Energy Consumption	Light-Medium	**Light**	Light-Medium
Application Size After Compilation	111MB	**37,67MB**	44,95MB
Avg. Inference Time (ms)	898,67	**140,67**	800,11

Regarding CPU usage, RAM, Application Size, Energy Consumption, and Average computing time MobileNetV2 is superior to ResNet50V2 and DenseNet. This is because this model is specifically engineered for mobile devices, but even so, the DenseNet121 model has a lot of convoluted neural networks. Also, the size is not much different from the MobileNetV2 model because the number of parameters generated by the model is not large enough, in contrast to ResNet50V2, which has a large number of parameters (see Table IV).

IV. CONCLUSION

The training and testing results show that fine-tuning can improve the performance of the pre-trained model used during the Transfer Learning process. It can be seen that the implementation of fine-tuning can also effectively increase accuracy and reduce loss during the training, validation, and testing processes. DenseNet121 is the best pre-trained model (92.58% accuracy and 29.62% Loss) after the model applies fine-tuning for the top 10 Layers.

The results of this study also show that the developed NTT woven fabric motif classification model can be implemented into a smartphone. The application of the model on smartphones also shows that the model developed using DenseNet121 as the pre-trained model has the best accuracy (99.23% accuracy in 130 test data), but when viewed in terms of resource use efficiency on mobile devices, the model developed using MobileNetV2 as the pre-trained model is the most efficient model used in smartphones.

Although there are some shortcomings in the engineered model, especially regarding resource efficiency, the application that has been developed can be used as an alternative to help the world community or people who want to learn about NTT culture, especially woven fabrics from NTT. Further development needs to be done, such as adding motif classes and datasets, developing a classification model to make it more efficient for mobile devices, and utilizing a mobile application development framework that supports various operating systems.

REFERENCES

[1] M. Siombo, "Kearifan Lokal Dalam Proses Pembuatan Tenun Ikat Timor (Studi pada Kelompok Penenun di Atambua-NTT)," Bina Huk. Lingkung., vol. 4, Oct. 2019, doi:10.24970/bhl.v4i1.88.

[2] Y. Xu, Y. Zhou, P. Sekula, and L. Ding, "Machine learning in construction: From shallow to deep learning," Dev. Built Environ., vol. 6, p. 100045, 2021, doi:10.1016/j.dibe.2021.100045.

[3] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," Pattern Recognit. Lett., vol. 141, pp. 61–67, 2021, doi:10.1016/j.patrec.2020.07.042.

[4] X. Bai et al., "Explainable deep learning for efficient and robust pattern recognition: A survey of recent developments," Pattern Recognit., vol. 120, p. 108102, 2021, doi:10.1016/j.patcog.2021.108102.

[5] G. W. Lindsay, “Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future,” J. Cogn. Neurosci., vol. 33, no. 10, pp. 2017–2031, Sep. 2021, doi:10.1162/jocn_a_01544.
[6] B. Tugrul, E. Elfatimi, and R. Eryigit, “Convolutional Neural Networks in Detection of Plant Leaf Diseases: A Review,” Agriculture, vol. 12, no. 8. 2022. doi:10.3390/agriculture12081192.
[7] G. Sakkarvarthi, G. W. Sathianesan, V. S. Murugan, A. J. Reddy, P. Jayagopal, and M. Elsisi, “Detection and Classification of Tomato Crop Disease Using Convolutional Neural Network,” Electronics, vol. 11, no. 21. 2022. doi:10.3390/electronics11213618.
[8] M. Küçükdemirci, G. Landeschi, M. Ohlsson, and N. Dell’Unto, “Investigating ancient agricultural field systems in Sweden from airborne LIDAR data by using convolutional neural network,” Archaeol. Prospect., vol. 30, no. 2, pp. 209–219, Apr. 2023, doi:10.1002/arp.1886.
[9] A. M. Mishra, S. Harnal, V. Gautam, R. Tiwari, and S. Upadhyay, “Weed density estimation in soya bean crop using deep convolutional neural networks in smart agriculture,” J. Plant Dis. Prot., vol. 129, no. 3, pp. 593–604, 2022, doi:10.1007/s41348-022-00595-7.
[10] Y. Xie et al., “Convolutional Neural Network Techniques for Brain Tumor Classification (from 2015 to 2022): Review, Challenges, and Future Perspectives,” Diagnostics, vol. 12, no. 8. 2022. doi:10.3390/diagnostics12081850.
[11] P. Oza, P. Sharma, S. Patel, and P. Kumar, “Deep convolutional neural networks for computer-aided breast cancer diagnostic: a survey,” Neural Comput. Appl., vol. 34, no. 3, pp. 1815–1836, 2022, doi:10.1007/s00521-021-06804-y.
[12] A. K. Sharma et al., “Dermatologist-Level Classification of Skin Cancer Using Cascaded Ensembling of Convolutional Neural Network and Handcrafted Features Based Deep Neural Network,” IEEE Access, vol. 10, pp. 17920–17932, 2022, doi:10.1109/ACCESS.2022.3149824.
[13] A. Kumar, A. R. Tripathi, S. C. Satapathy, and Y.-D. Zhang, “SARS-Net: COVID-19 detection from chest x-rays by combining graph convolutional network and convolutional neural network,” Pattern Recognit., vol. 122, p. 108255, 2022, doi:10.1016/j.patcog.2021.108255.
[14] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” SN Comput. Sci., vol. 2, no. 3, pp. 1–21, 2021, doi:10.1007/s42979-021-00592-x.
[15] M. A. Rasyidi and T. Bariyah, “Batik pattern recognition using convolutional neural network,” vol. 9, no. 4, pp. 1430–1437, 2020, doi:10.11591/eei.v9i4.2385.
[16] B. S. Negara, E. Satria, S. Sanjaya, and D. R. Dwi Santoso, “ResNet-50 for Classifying Indonesian Batik with Data Augmentation,” in 2021 International Congress of Advanced Technology and Engineering (ICOTEN), 2021, pp. 1–4. doi:10.1109/ICOTEN52080.2021.9493488.
[17] Y. Azhar, M. C. Mustaqim, and A. E. Minarno, “Ensemble convolutional neural network for robust batik classification,” IOP Conf. Ser. Mater. Sci. Eng., vol. 1077, no. 1, p. 12053, 2021, doi:10.1088/1757-899x/1077/1/012053.
[18] L. Alzubaidi et al., “Novel Transfer Learning Approach for Medical Imaging with Limited Labeled Data,” Cancers, vol. 13, no. 7. 2021. doi:10.3390/cancers13071590.
[19] W. Ge and Y. Yu, Borrowing Treasures from the Wealthy: Deep Transfer Learning through Selective Joint Fine-Tuning. 2017. doi:10.1109/CVPR.2017.9.
[20] B. Sreenivasulu, A. Pasala, and G. Vasanth, “Adaptive Inception Based on Transfer Learning for Effective Visual Recognition,” Int. J. Intell. Eng. Syst., vol. 13, no. 6, pp. 1–10, 2020, doi:10.22266/ijies2020.1231.01.
[21] M. A. Iqbal Hussain, B. Khan, Z. Wang, and S. Ding, “Woven Fabric Pattern Recognition and Classification Based on Deep Convolutional Neural Networks,” Electronics, vol. 9, no. 6. 2020. doi:10.3390/electronics9061048.
[22] M. A. Rasyidi, R. Handayani, and F. Aziz, “Identification of batik making method from images using convolutional neural network with limited amount of data,” Bull. Electr. Eng. Informatics; Vol 10, No 3 June 2021, 2021, doi:10.11591/eei.v10i3.3035.
[23] Y. Harjoseputro, Y. D. Handarkho, H. Tresy, and R. Adie, “The Javanese Letters Classifier with Mobile Client- Server Architecture and Convolution Neural Network Method,” vol. 13, no. 12, pp. 67–80, 2019.
[24] M. G. Lanjewar and K. G. Panchbhai, “Convolutional neural network based tea leaf disease prediction system on smart phone using paas cloud,” Neural Comput. Appl., vol. 35, no. 3, pp. 2755–2771, 2023, doi:10.1007/s00521-022-07743-y.
[25] K.-S. Lee et al., “Compressed Deep Learning to Classify Arrhythmia in an Embedded Wearable Device,” Sensors, vol. 22, no. 5, 2022, doi:10.3390/s22051776.
[26] Z. Al-Halah, S. K. Ramakrishnan, and K. Grauman, “Zero Experience Required: Plug \& Play Modular Transfer Learning for Semantic Visual Navigation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2022, pp. 17031–17041.
[27] P. Kora et al., “Transfer learning techniques for medical image analysis: A review,” Biocybern. Biomed. Eng., vol. 42, no. 1, pp. 79–107, 2022, doi:10.1016/j.bbe.2021.11.004.
[28] G. He, P. Xue, and J. Meng, “Few-shot Thangka image classification based on improved DenseNet,” J. Phys. Conf. Ser., vol. 1678, p. 12087, Nov. 2020, doi:10.1088/1742-6596/1678/1/012087.
[29] V. Gupta et al., “Cross-property deep transfer learning framework for enhanced predictive analytics on small materials data,” Nat. Commun., vol. 12, no. 1, p. 6595, 2021, doi:10.1038/s41467-021-26921-5.
[30] S. Rezaei, J. Tahmoresnezhad, and V. Solouk, “A transductive transfer learning approach for image classification,” Int. J. Mach. Learn. Cybern., vol. 12, no. 3, pp. 747 – 762, 2021, doi:10.1007/s13042-020-01200-9.
[31] J. Kobylarz, J. J. Bird, D. R. Faria, E. P. Ribeiro, and A. Ekárt, “Thumbs up, thumbs down: non-verbal human-robot interaction through real-time EMG classification via inductive and supervised transductive transfer learning,” J. Ambient Intell. Humaniz. Comput., vol. 11, no. 12, pp. 6021 – 6031, 2020, doi:10.1007/s12652-020-01852-z.
[32] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” CoRR, vol. abs/1608.0, 2016, [Online]. Available: http://arxiv.org/abs/1608.06993
[33] K. He, X. Zhang, S. Ren, and J. Sun, “Identity Mappings in Deep Residual Networks.” arXiv, 2016. doi:10.48550/ARXIV.1603.05027.
[34] I. W. A. S. Darma, N. Suciati, and D. Siahaan, “Neural Style Transfer and Geometric Transformations for Data Augmentation on Balinese Carving Recognition using MobileNet,” Int. J. Intell. Eng. Syst., vol. 13, no. 6, pp. 349–363, 2020, doi:10.22266/ijies2020.1231.31.
[35] T. Anwar and S. Zakir, “Effect of Image Augmentation on ECG Image Classification using Deep Learning,” in 2021 International Conference on Artificial Intelligence, ICAI 2021, 2021, pp. 182–186. doi:10.1109/ICAI52203.2021.9445258.
[36] A. Rahman, Y. Lu, and H. Wang, “Performance evaluation of deep learning object detectors for weed detection for cotton,” Smart Agric. Technol., vol. 3, 2023, doi:10.1016/j.atech.2022.100126.
[37] M. D. Bloice, C. Stocker, and A. Holzinger, “Augmentor: An Image Augmentation Library for Machine Learning,” ArXiv, vol. abs/1708.0, 2017.
[38] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization.” 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[39] B. D. Satoto, M. I. Utoyo, R. Rulaningtyas, and E. B. Koendhori, “Custom convolutional neural network with data augmentation and bayesian optimization for gram-negative bacteria classification,” Int. J. Intell. Eng. Syst., vol. 13, no. 5, pp. 524–538, 2020, doi:10.22266/ijies2020.1031.46.
[40] TensorFlow Developers (2023) “TensorFlow”. Zenodo. doi:10.5281/zenodo.10126399.