Komputika: Jurnal Sistem Komputer
Volume 12, Nomor 1, April 2023, hlm. 1 - 11
Terakreditasi Peringkat 3, SK No. 148/M/KPT/2020
DOI: **10.34010/komputika.v12i1.8626**

ISSN: 2252-9039 (print)
ISSN: 2655-3198 (online)

A Comparative Analysis of Transfer Learning Architecture Performance on Convolutional Neural Network Models with Diverse Datasets

Muhammad Daffa Arviano Putra[1], Tawang Sahro Winanto[2], Retno Hendrowati[3], Aji Primajaya[4], Faisal Dharma Adhinata[5,*]

[1,3)] Department of Informatics, Universitas Paramadina, Jakarta, Indonesia
[2,4)] Department of Informatics, Universitas Singaperbangsa Karawang, Karawang, Indonesia
[5)] Department of Software Engineering, Institut Teknologi Telkom Purwokerto, Purwokerto, Indonesia
[1,2,5)] Bisa AI Academy, Bandung, Indonesia
**email: faisal@ittelkom-pwt.ac.id*

(Naskah masuk: *11 Desember 2022*; diterima untuk diterbitkan: *02 Mei 2023*)

ABSTRACT *– Deep learning is a branch of machine learning with many highly successful applications. One application of deep learning is image classification using the Convolutional Neural Network (CNN) algorithm. Large image data is required to classify images with CNN to obtain satisfactory training results. However, this can be overcome with transfer learning architectural models, even with small image data. Therefore, with transfer learning, the success rate of a model is likely to be higher. Since there are many transfer learning architecture models, comparing each model's performance results is necessary to find the best-performing architecture. In this study, we conducted three experiments on different datasets to train models with various transfer learning architectures. We then performed a comprehensive comparative analysis for each experiment. The result is that the DenseNet-121 architecture is the best transfer learning architecture model for various datasets. The DenseNet-121 transfer learning architecture is the best because it achieved the highest evaluation scores in the second and third experiments. Although MobileNet was superior in the first and second experiments, the evaluation value in the third experiment was very low.*

Key Words *– Deep learning, transfer learning, CNN, image classification*

1. INTRODUCTION

Deep learning is a research trend because of its wide application and relative success rate. It is because of how deep learning works, which mimics the workings of the human brain. As a result, processes such as feature extraction, which is the main focus of machine learning-based research, can be carried out automatically by deep learning algorithms [1]. Various examples of the application of deep learning include applications in medical image analysis [2], sentiment analysis on Twitter data [3], vehicle detection [4], plant disease classification [5], communication signal processing [6], and the stock market forecasting [7]. There are various algorithms in deep learning. One example of a well-known deep learning algorithm today is the Convolutional Neural Network (CNN) [8].

Convolutional Neural Network (CNN) is a type of neural network often used to detect and identify objects in an image. CNN, a deep neural network model, is designed to process two-dimensional data in image-processing tasks [9]. For training the CNN model, large datasets are often required for the model to get a satisfactory level of success. However, a transfer learning architecture on a small dataset can help models achieve better accuracy [10]. Transfer learning is a deep learning method in which a model trained on one problem is reused for another issue. Transfer learning enables deep learning model training to achieve high accuracy even when using small amounts of data [10]. However, many transfer learning architectures can be used on CNN models. As a result, comprehensive research is needed to find the best-performing transfer learning architecture.

Various studies have been conducted on the performance of transfer learning architecture on the CNN method for image classification. For example, a comparative study of several transfer learning architectures on the CNN model for detecting COVID-19 from CT scan images [11]. Another study has done experiments and comparative analysis between the classic CNN model and several transfer learning architectures to catch pneumonia from chest X-ray images [12]. A similar study can also be seen in [13], which implemented various transfer learning architectures to detect fatigue crack initiation sites and compare the results from different models. The study's drawback is that they only focus on using multiple transfers learning architectural models in specific problems with one dataset.

One other study examines the comparison of transfer learning architectures on different datasets [14]. However, this study only compared two datasets, namely small and large datasets. The research also does not provide many answers regarding the best transfer learning architecture. It is because, in large datasets, all models created with transfer learning architecture have a high probability of getting perfect accuracy. In addition, this study did not provide an in-depth analysis regarding differences in the accuracy of each model on small datasets. Therefore, our study will thoroughly experiment with various transfer learning models. The goal is to find the best transfer learning architecture that can be used on the CNN model.

For the experiments in this study, we will create several deep-learning models with CNN for image classification with several different datasets. The type and amount of data used will vary. Various transfer learning architectures will be applied to these models. The transfer learning architectures we implement include MobileNet, VGG-19, Resnet50V2, DenseNet-121, and NASNetMobile. Then, we will make a comprehensive analysis regarding the comparison of transfer learning architectures used to find the best transfer learning architectural model. The main contributions of our research are as follows:

- Analyze the performance of each transfer learning architecture on the CNN model on different datasets using well-known evaluation metrics
- Provides a performance comparison of all transfer learning architectures for all datasets to get the best-performing architecture

This study helps find the best transfer learning architecture for the CNN model. It is advantageous for small datasets to get maximum performance results. By finding the best transfer learning architecture from the results of our research, further research that utilizes the transfer learning architecture can immediately decide on the transfer learning architecture to be implemented. Thus, the steps for defining the transfer learning architecture can be omitted.

2. Dataset and Method

This comparative study on transfer learning architecture begins with data collection. There are three datasets used. Each dataset is loaded and goes through the preprocessing stage, i.e., resizing the image. Then the data will be trained by adding various transfer learning architectures to the CNN model separately. The transfer learning architectures tested were MobileNet, VGG-19, Resnet50V2, DenseNet-121, and NASNetMobile. In the training and validation process, each model will be evaluated by comparing the values of training loss, training accuracy, validation loss, and validation accuracy. After that, the testing process will be carried out using the test data, and the comparison of the results will be analyzed. After that, the same method will be applied to the second dataset and the third dataset. The results of the evaluation of all models in the three datasets will be studied and analyzed. This research results in the discovery of the best transfer learning architecture. Figure 1 shows the method proposed in this study.

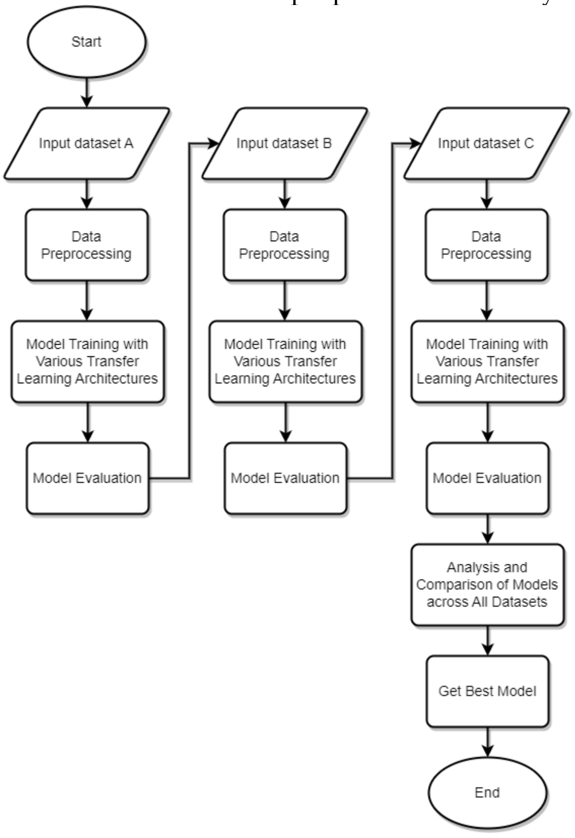

Start
Input dataset A
Data Preprocessing
Model Training with Various Transfer Learning Architectures
Model Evaluation
Input dataset B
Data Preprocessing
Model Training with Various Transfer Learning Architectures
Model Evaluation
Input dataset C
Data Preprocessing
Model Training with Various Transfer Learning Architectures
Model Evaluation
Analysis and Comparison of Models across All Datasets
Get Best Model
End

Figure 1 Flowchart of the comparative study of transfer learning architecture

2.1. Dataset

The main objective of this study is to compare the performance of various transfer learning architectures across a variety of datasets. To accomplish this, three distinct datasets have been selected for three separate experiments. All datasets used are publicly available from Kaggle, which guarantees accessibility and reproducibility of the study. The specific datasets were chosen based on their small data size, which is necessary for evaluating the effectiveness of the transfer learning architectures in adapting to limited new data. The chosen datasets are also diverse in terms of the number of classes and the number of images, presenting unique challenges and characteristics that can affect the performance of the models. Additionally, all images within the datasets are in RGB (Red, Green, Blue) format. The details of each dataset are as follows:

1. The first experiment used the Indonesian Wayang Types dataset [15]. In the dataset, 233 wayang images consist of 6 classes. The six classes are Wayang Beber, Wayang Gedog, Wayang Golek, Wayang Kulit and Wayang Suluh. Each class consists of about 32 to 45 image data. An example of an image in this dataset is shown in Figure 2.

Figure 2 Image example from the first dataset

2. The second experiment uses the Drone-Bird Classification dataset [16]. The dataset contains 330 images of 2 classes, namely the Bird and Drone classes. Each class consists of 165 image data. An example of an image in this dataset is shown in Figure 3.

Figure 3 Image example from the second dataset

3. The third experiment uses the Grapevine Leaves Image dataset [17]. The dataset has 500 images of vine leaves consisting of 5 classes. The five classes are Ak, Ala Idris, Buzgulu, Dimnit, Nazli. Each class consists of 100 image data. An example of an image in this dataset is shown in Figure 4.

Figure 4 Image example from the third dataset

2.2. Data Preprocessing

Data preprocessing is a step to prepare image data before creating an image classification model. In this study, the preprocessing stage was carried out, namely changing the image data size in each dataset to 224x224. Then each dataset is divided into 80% training data, 10% validation data, and 10% testing data.

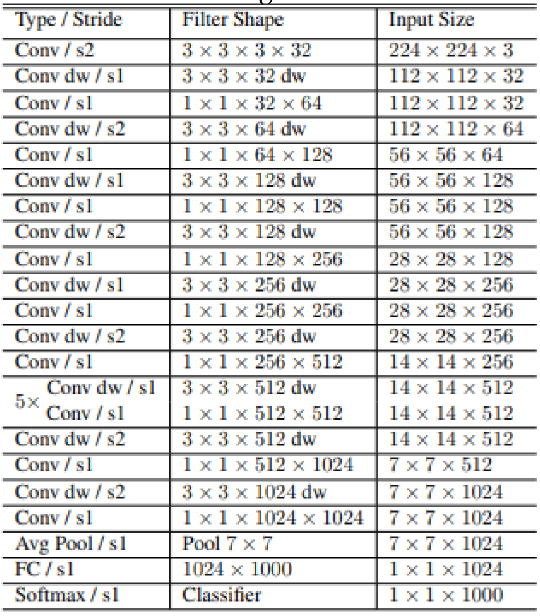
2.3. Convolutional Neural Network

Convolutional Neural Network (CNN) is a popular type of deep neural network, especially for image and video classification tasks. The advantages of CNN are a way for researchers to develop models to solve various complicated things in different fields [8]. It is reflected in CNN's ability to process data in three dimensions, each of which is useful for processing voice, image, and video data. In summary, CNN consists of several layers: convolution, pooling, and fully connected [18]. In the case of image classification, the fully connected layer is the layer that provides predictions from a classification [19]. However, the convolution layer is the most important because it creates a feature activation map to understand features in image or video data [8], [19].

Various pre-trained models have been built and trained with the CNN algorithm model for image classification tasks. Therefore, everyone can proceed from the pre-trained model to the next image classification model. The new model already has a lot of 'experience' and 'learning' when it is trained for another specific task. The technique of using pre-trained models is also known as transfer learning [20].

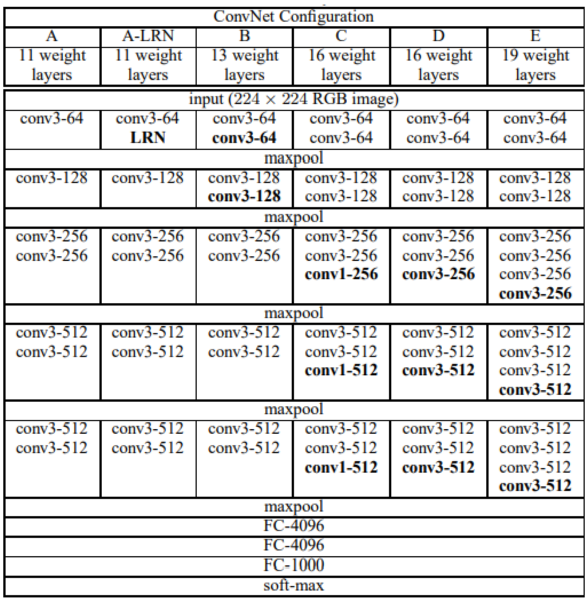
2.4. MobileNet

MobileNet is designed to address the need for excess computing resources. The MobileNet convolution layers are classified into ten blocks. The first uses standard convolution, which produces 32 features, while the next block uses DSC and down-sampling with max-pooling. The feature map increases with binary multiplication up to 1024 features in the last block [21]. The MobileNet architecture is shown in Figure 5.



Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1 Conv / s1	3 × 3 × 512 dw 1 × 1 × 512 × 512	14 × 14 × 512 14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

Figure 5 MobileNet architecture [22]



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 6 VGG-19 architecture [23]

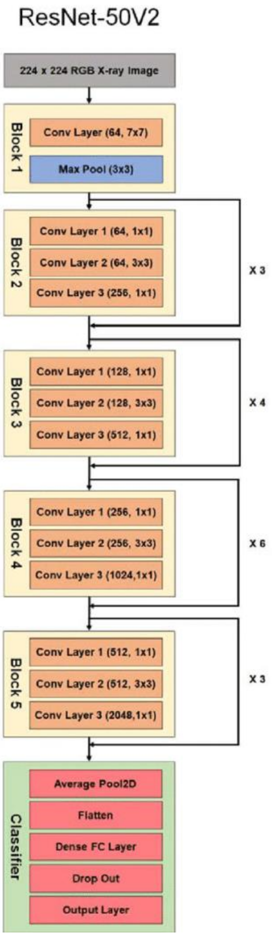


Figure 7 ResNet50V2 architecture [25]

2.5. VGG-19

VGG-19 is a network architecture created by the Visual Geometry Group (VGG) at the University of Oxford in 2014. The VGGNet architecture can perform with high accuracy. There are 19 layers in the VGG-19 architecture, of which 16 are convolution layers, and 3 are fully connected layers [24]. The VGG-19 architecture is shown in Figure 6.

2.6. ResNet50V2

Residual Network (ResNet) is a popular type of architecture that emphasizes the concept of residual blocks. This architecture uses two simple rules: each layer has the same number of filters for the same output feature map size. The second rule is that if the size of the output feature map is divided into two, the number of filters for each layer is doubled [26]. For this research, we will use ResNet50v2. The ResNet50V2 architecture has a total of 50 layers. This architecture is shown in Figure 7.

2.7. DenseNet-121

DenseNet-121 is a type of model of the Dense Convolutional Network (DenseNet) architecture [27]. DenseNet121 consists of four dense blocks in which there is a transition layer between the two blocks. There are three transition layers in DensetNet121, consisting of the convolution layer and the pooling layer. All convolution, transition, and classification layers total 121 layers [28]. The DenseNet-121 architecture is shown in Figure 8.

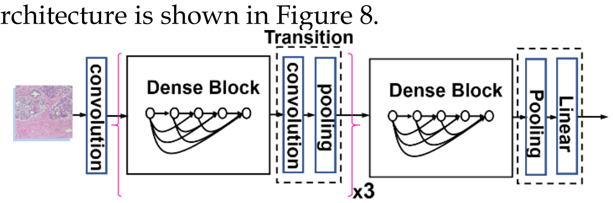


Figure 8 DenseNet-121 architecture [28]

2.8. NASNetMobile

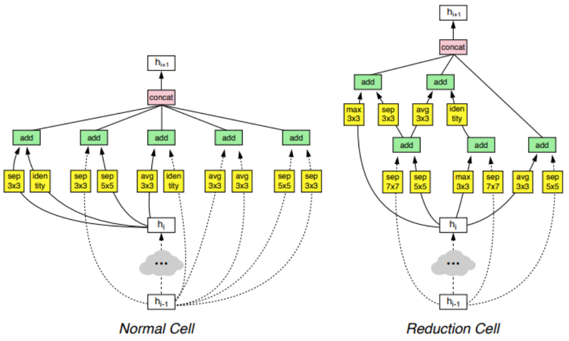


Figure 9 NASNet architecture [29]

Researchers developed the NASNet architecture from Google Brain [29]. The initial idea for this architecture stems from using the Neural Architecture Search (NAS) framework as a search method to find the best convolutional architecture on small datasets. Then with the contribution of a new search space design called NASNet search space, the architecture is transferred to a larger dataset. The best architecture was found in the NASNet search space, which was then named NASNet. In the NASNet architecture, there are two convolutional cells called Normal Cells and Reduction Cells. Normal Cell functions to return a feature map with the exact dimensions, while Reduction Cell functions to produce a feature map where the height and width of the feature map are reduced by a factor of two. The original NASNet architecture is shown in Figure 9. We use the NASNetMobile architecture for this research because the input image size used is 224x224 [30].

2.9. Model Evaluation and Comparison

In this comparative study, we look at the results of the training loss, training accuracy, validation loss, and validation accuracy values of each model in each dataset. All training and validation processes are carried out at the same number of epochs and batch sizes. We observe the performance of each transfer learning architecture for three different datasets for training and validation processes. After that, we compared the accuracy, precision, recall, and f1-score results of all models across all datasets in the testing process. Finally, we conducted an in-depth analysis to conclude the best transfer learning architecture from the three experiments.

3. Results and Discussion

This study utilizes several tools and technologies, including the Python programming language, the TensorFlow machine learning framework, and the Keras API. To accelerate the training and testing processes, we utilized Google Colaboratory and took advantage of the provided Tesla T4 GPU for conducting the experiments. The results of all the experiments in the training, validation, and testing processes are presented. Finally, a comparative analysis is performed to evaluate the performance of the transfer learning architecture models based on the experiments.

A. Training Process

We experiment with each dataset for the training process and train our models with different transfer learning architectures. We use the adam optimizer and categorical cross-entropy for each model as the loss function. We use the batch size of 8 for all models. Then, for the number of epochs, we use 50 epochs. However, we evaluate the number of epochs in multiples of 10, from 10 to 50. Table 1, 2, and 3 shows the training results of accuracy and loss values for each experiment with each different dataset.

Table 1 Training result on the first dataset

Model	Accuracy					Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	99.46%	100%	100%	100%	100%	10.91%	3.68%	1.96%	1.21%	0.85%
VGG19	74.59%	80.54%	82.70%	84.86%	89.19%	110%	85.76%	70.66%	60.34%	51.59%
ResNet50V2	99.46%	100%	100%	100%	100%	6.95%	2.39%	1.28%	0.79%	0.55%
DenseNet121	97.30%	99.46%	100%	100%	100%	21.30%	8.52%	4.89%	3.03%	2.05%
NASNetMobile	97.30%	99.46%	100%	100%	100%	20.18%	8.22%	4.51%	2.86%	1.97%

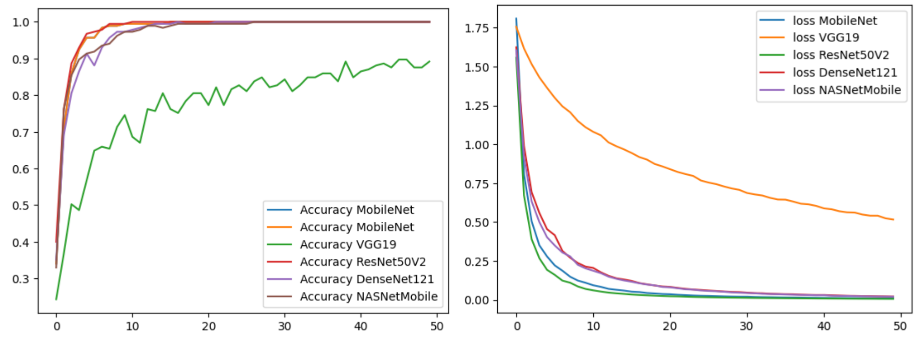


Figure 10 Training learning curve on the first dataset

Table 2 Training result on the second dataset

Model	Accuracy					Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	100%	100%	100%	100%	100%	1.13%	0.39%	0.21%	0.13%	0.08%
VGG19	93.08%	95.77%	97.69%	98.08%	98.85%	36.85%	25.77%	20.25%	17.04%	14.37%
ResNet50V2	100%	100%	100%	100%	100%	0.38%	0.14%	0.07%	0.04%	0.03%
DenseNet121	100%	100%	100%	100%	100%	1.6%	0.67%	0.36%	0.36%	0.16%
NASNetMobile	100%	100%	100%	100%	100%	3.14%	1.3%	0.61%	0.37%	0.26%

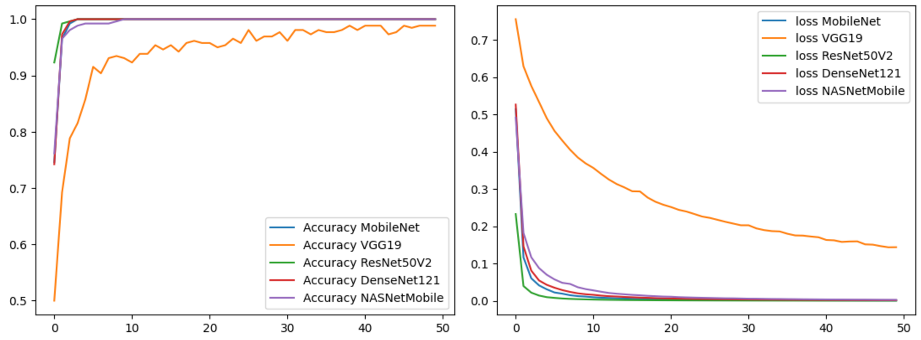


Figure 11 Training learning curve on second dataset

Table 3 Training result on third dataset

Model	Accuracy					Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	93.50%	98.25%	99.50%	100%	100%	29.72%	15.12%	8.58%	5.16%	3.51%
VGG19	66.25%	74.75%	78.75%	80.50%	80.50%	124%	101%	86.55%	76.53%	69.23%
ResNet50V2	93.25%	9850%	100%	100%	100%	35.30%	16.92%	9.57%	5.88%	3.78%
DenseNet121	90.50%	94.75%	97.25%	99%	98.75%	45.68%	27.11%	18.72%	13.85%	9.86%
NASNetMobile	85.75%	92.50%	97%	98%	98.75%	50.93%	31.52%	21.91%	15.65%	11.17%

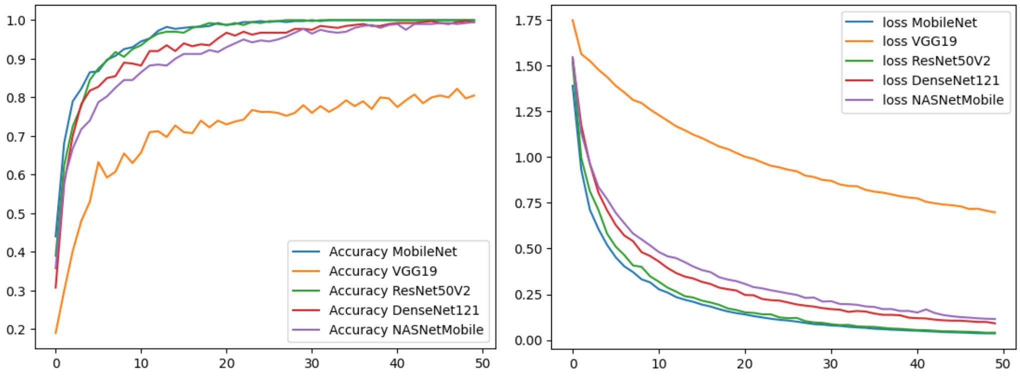


Figure 12 Training learning curve on the third dataset

In the first experiment, there are four transfer learning architecture models that have 100% accuracy at epoch 50. However, of the four models, ResNet50V2 has the lowest training loss value compared to the other models. VGG19 is the model with the worst performance because it has a low accuracy value compared to other models. The second experiment on the second dataset did not differ much from the first one. Four models have a training accuracy value of 100%, with ResNet50V2 having the lowest loss value. VGG19 is also the worst model in this experiment.

The difference began in the third experiment, where only two models achieved 100% accuracy in the training process. The transfer learning architecture models mentioned are MobileNet and ResNet50V2. However, DenseNet121 and NASNetMobile still have pretty good accuracy, with 98.75%.

B. Validation Process

In the following process, we try to analyze each model's performance in the three experiments on validation data. Here we look at validation accuracy and loss validation by evaluating the number of epochs in multiples of 10, which is 10 to 50. Tables 4, 5, and 6 show the results of validating the accuracy and loss values for each experiment on different datasets. The validation process results did not differ much from the first experiment, where ResNet50V2 achieved the highest accuracy validation with the lowest validation loss value. The four models obtained 100% validation accuracy in the second experiment, but the MobileNet architecture reached the lowest loss validation value. None of the models got 100% validation accuracy in the third experiment. In addition, all models tend to have a reasonably high loss validation value. However, the best architectural model here is achieved by DenseNet-121, with a validation accuracy value of 90%. MobileNet and NASNetMobile followed it with a validation accuracy value of 86%.

Table 4 Validation result on the first dataset

Model	Validation Accuracy					Validation Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	86.36%	95.45%	95.45%	100%	100%	25.73%	19.25%	15.87%	15.25%	13.50%
VGG19	63.64%	54.55%	59.09%	59.09%	68.18%	117%	101%	92.90%	87.54%	81.42%
ResNet50V2	95.45%	100%	100%	100%	100%	19.55%	13.84%	11.58%	10.13%	9.39%
DenseNet121	86.36%	95.45%	95.45%	100%	100%	40.38%	27.99%	24.90%	20.40%	19.50%
NASNetMobile	81.82%	90.91%	90.91%	90.91%	86.36%	47.96%	37.45%	36.47%	34.21%	33.96%

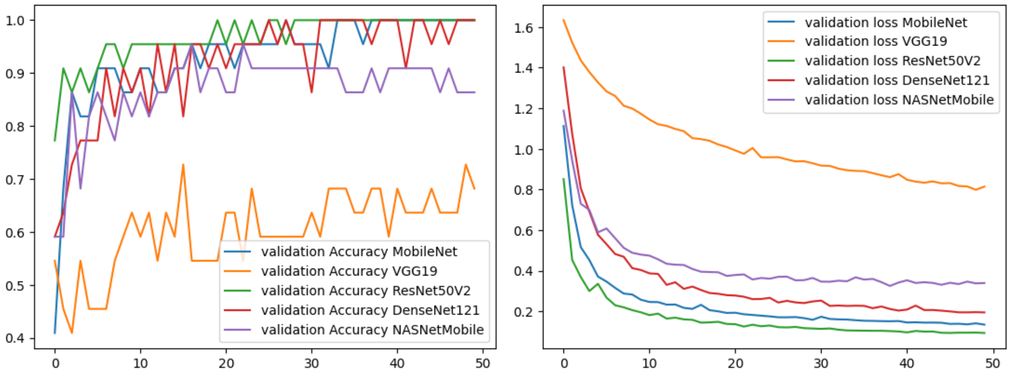


Figure 13 Validation learning curve on first dataset

Table 5 Validation result on second dataset

Model	Validation Accuracy					Validation Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	100%	100%	100%	100%	100%	1.62%	0.78%	0.51%	0.38%	0.30%
VGG19	90.62%	90.62%	90.62%	93.75%	93.75%	48.24%	39.04%	33.34%	29.72%	26.59%
ResNet50V2	100%	100%	100%	100%	100%	1.18%	0.75%	0.58%	0.48%	0.42%
DenseNet121	100%	100%	100%	100%	100%	2.98%	1.62%	1.17%	1.17%	0.75%
NASNetMobile	100%	100%	100%	100%	100%	8.92%	5.05%	3.71%	2.87%	2.38%

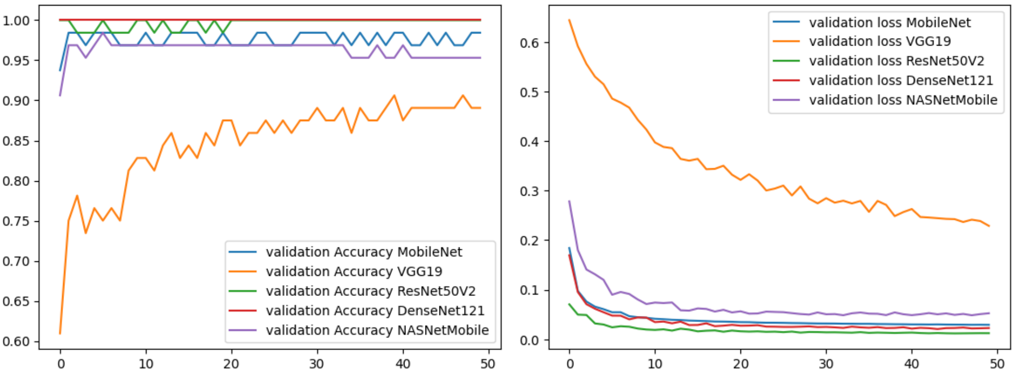


Figure 14 Validation learning curve on second dataset

Table 6 Validation result on third dataset

Model	Validation Accuracy					Validation Loss				
	Epoch					Epoch				
	10	20	30	40	50	10	20	30	40	50
MobileNet	86%	84%	86%	86%	86%	44.51%	37.92%	34.95%	34.32%	35.40%
VGG19	52%	78%	74%	78%	82%	122%	97.91%	83.89%	74.92%	68.03%
ResNet50V2	70%	80%	84%	86%	78%	63.70%	54.77%	51.39%	52.04%	53.37%
DenseNet121	86%	84%	88%	88%	90%	53.93%	43.68%	35.42%	36.68%	30.92%
NASNetMobile	74%	80%	80%	86%	86%	68.79%	58.41%	51.78%	48.82%	47.98%

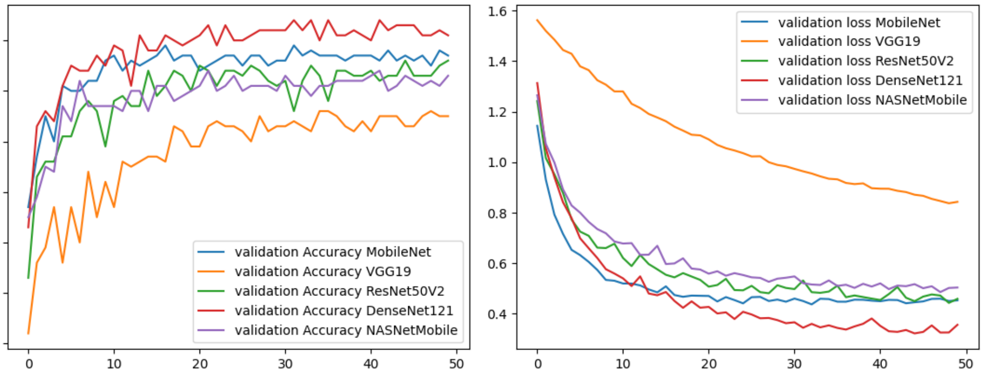


Figure 15 Validation learning curve on the third dataset

C. Testing Result and Analysis

After going through the training and validation process, we describe the testing results through several evaluation parameters that have been determined beforehand, namely Precision, Recall, F1-Score, and Accuracy. It is done to understand the classification performance of each transfer learning model in each dataset. The evaluation values of the three experiments are shown in Tables 7, 8, and 9.

Table 7 Testing result on the first dataset

Model	Precision	Recall	F1-Score	Accuracy
MobileNet	91.67%	92.50%	90.92%	91.67%
VGG19	81.94%	77.50%	74.70%	75%
ResNet50V2	86.35%	84.17%	82.62%	83.33%
DenseNet-121	84.72%	84.17%	82.40%	83.33%
NASNetMobile	87.50%	89.17%	86.57%	87.50%

Table 8 Testing result on the second dataset

Model	Precision	Recall	F1-Score	Accuracy
MobileNet	95.45%	96.15%	95.62%	95.65%
VGG19	91.67%	92.31%	91.29%	91.30%
ResNet50V2	95.45%	96.15%	95.62%	95.65%
DenseNet-121	95.45%	96.15%	95.62%	95.65%
NASNetMobile	95.45%	96.15%	95.62%	95.65%

Table 9 Testing result on the third dataset

Model	Precision	Recall	F1-Score	Accuracy
MobileNet	79.56%	50%	48.39%	50%
VGG19	56.05%	60%	51.85%	60%
ResNet50V2	55.44%	52%	47.99%	52%
DenseNet-121	84.61%	80%	79.34%	80%
NASNetMobile	52.87%	46%	38.35%	46%

From the three experiments conducted, it can be concluded that in each dataset, there is one transfer learning architecture that excels in the four evaluation parameters except for the second experiment. In the first dataset, the MobileNet architecture is the best transfer learning model with a precision value of 91.67%, a recall value of 92.50%, an F1-score value of 90.92%, and an accuracy value of 91.67%. Thus, the first experiment performed for the first dataset was won by the MobileNet architecture.

As for the classification performance in the second experiment, it was found that four architectural models had the same values for all evaluation metrics. These architectures are MobileNet, ResNet50V2, DenseNet-121, and NASNetMobile. The four architectures get a precision value of 95.45%, a recall value of 96.15%, an F1-score value of 95.62%, and an accuracy value of 95.65%. As a result, there are four best architectures for the second dataset in the experiment.

DenseNet-121 became the most superior transfer learning architecture model in the classification performance for the third experiment. It is proven by the gain of a precision value of 84.61%, a recall value of 80%, an F1-score value of 79.34%, and an accuracy value of 80%. DenseNet-121 wins this experiment as no transfer learning architecture comes close to the performance of DenseNet-121, and other architectures tend to have poor performance overall.

D. Comparative Analysis

From the experimental results, we conducted an in-depth analysis of the effects of each transfer learning architecture in the training, validation, and testing processes. From the first experiment, we found that the ResNet50V2 architecture achieves high accuracy in the training and validation process and with the lowest loss. However, after testing with test data, the MobileNet architecture achieved the highest results on all evaluation metrics.

Interesting results emerged in the second experiment. Because in this second dataset experiment, four models have 100% accuracy values in the training, validation, and testing processes. The architectures in question are MobileNet, ResNet50V2, DenseNet-121, and NASNetMobile. In addition, the four architectures also achieved the same results on all evaluation metrics after predicting the data testing. Thus, in this second dataset, we can conclude that only the VGG19 architecture gets low evaluation results and is different from other architectures.

In the third experiment, two architectures stand out, namely ResNet50V2 and DensetNet-121. ResNet50V2 excels in the training process with 100% accuracy. However, during the validation and testing process, ResNet50V2 received poor accuracy. On the other hand, DenseNet-121 was lost in the training process with 98.75% accuracy. However, DenseNet-121 got 90% accuracy in the validation process and achieved the highest evaluation score. The cause of DenseNet-121 being superior to ResNet50V2 could be due to the testing process, where the ResNet50V2 architecture experienced an overfit, which resulted in the validation process getting poor results.

After looking at and comparing all the experiments, we have come up with three architectures that have the potential to be the best architectures. We then try to provide the following conclusions:

a. The MobileNet architecture excels in test results for the first and second datasets but performs poorly in the third.
b. The DenseNet-121 architecture excels in test results for the second and third datasets. But in the third dataset, DenseNet was lost in the training process. In the first dataset, DenseNet still has a reasonably good evaluation value even though it fails to MobileNet.
c. The ResNet50V2 architecture excels in the results of the second dataset test. ResNet50V2 has a good value in the training and validation process for the first dataset. This architecture also excelled in the training process in the third dataset, but it experienced an overfit and received poor validation and testing scores.
d. The NASNetMobile architecture also excels in the second dataset test results. But other than that, NASNetMobile does not have any impressive results.
e. The VGG19 architecture is the worst architectural model of the three experiments conducted.

After analyzing the experimental results on the three datasets, we find that the three architectures have pretty good performance. These architectures are DenseNet-121, MobileNet, and ResNet50V2. However, looking at the conclusions we have stated previously, the DenseNet-121 transfer learning architecture can be said to be the best because it achieved the highest evaluation scores in the second and third experiments. DenseNet-121 also scored reasonably well on the first try. On the other hand, even though MobileNet was superior in the first and second experiments, the evaluation value in the third experiment was very low. In addition, ResNet50V2 has good results in the training and validation process, but not so good after the testing process. Thus, DenseNet-121 has an excellent overall performance in the three experiments conducted.

4. CONCLUSION

This study aims to provide a comparative analysis of various transfer learning architectures for the CNN algorithm. Previous research that made comparative studies has not provided many answers regarding the best transfer learning architecture. It is because the study only compares the value of accuracy and the length of time for learning. Therefore, we comprehensively conducted experiments with three different datasets with five transfer learning architectures in this study. Then we compare the performance results of each architecture on each dataset and perform a thorough analysis. Based on our experiments and analysis, we found that DenseNet-121 is the best performing transfer learning architecture. However, it is important to note that there is still room for future research regarding the comparison of performance of transfer learning architectures. One area for future research is to explore the performance of other various transfer learning architectures not used in this study. Future studies could also consider using the same architectures from this study but with additional distinct datasets to further validate their performance, especially the DenseNet-121 architecture. Additionally, future research could also explore the impact of using different data splits for training, validation, and testing models to enable a more comprehensive evaluation of the performance of each transfer learning architecture. Through these efforts, it is hoped that the transfer learning architecture with the best performance for multiple datasets can be found.

REFERENCES

[1] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J. Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00444-8.

[2] J. Ker, L. Wang, J. Rao, and T. Lim, “Deep Learning Applications in Medical Image Analysis,” *IEEE Access*, vol. 6, pp. 9375–9389, 2018, doi: 10.1109/ACCESS.2017.2788044.

[3] S. Kamiş and D. Goularas, “Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data,” in *Proceedings - 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications, Deep-ML 2019*, Aug. 2019, pp. 12–17. doi: 10.1109/Deep-ML.2019.00011.

[4] H. S. Dikbayir and H. Ibrahim Bulbul, "Deep Learning Based Vehicle Detection from Aerial Images," in *Proceedings - 19th IEEE International Conference on Machine Learning and Applications, ICMLA 2020*, Dec. 2020, pp. 956–960. doi: 10.1109/ICMLA51294.2020.00155.

[5] K. P. Akshai and J. Anitha, "Plant disease classification using deep learning," in *2021 3rd International Conference on Signal Processing and Communication, ICPSC 2021*, May 2021, pp. 407–411. doi: 10.1109/ICSPC51351.2021.9451696.

[6] Y. Liu, Y. Li, Y. Zhu, Y. Niu, and P. Jia, "A Brief Review on Deep Learning in Application of Communication Signal Processing," in *2020 IEEE 5th International Conference on Signal and Image Processing, ICSIP 2020*, Oct. 2020, pp. 51–54. doi: 10.1109/ICSIP49896.2020.9339345.

[7] M. Biswas, A. Shome, M. A. Islam, A. J. Nova, and S. Ahmed, "Predicting stock market price: A logical strategy using deep learning," in *ISCAIE 2021 - IEEE 11th Symposium on Computer Applications and Industrial Electronics*, Apr. 2021, pp. 218–223. doi: 10.1109/ISCAIE51753.2021.9431817.

[8] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[9] E. I. Haksoro and A. Setiawan, "pengenalan jamur yang dapat dikonsumsi menggunakan metode transfer learning pada convolutional neural network" *Jurnal ELTIKOM: Jurnal Teknik Elektro, Teknologi Informasi dan Komputer*, vol. 5, no. 2, pp. 81-91, 2021.

[10] S. Ahuja, B. K. Panigrahi, N. Dey, V. Rajinikanth, and T. K. Gandhi, "Deep transfer learning-based automated detection of COVID-19 from lung CT scan slices," *Appl. Intell.*, vol. 51, no. 1, pp. 571–585, Jan. 2021, doi: 10.1007/s10489-020-01826-w.

[11] A. Seum, A. H. Raj, S. Sakib, and T. Hossain, "A comparative study of CNN transfer learning classification algorithms with segmentation for COVID-19 detection from CT scan images," in *Proceedings of 2020 11th International Conference on Electrical and Computer Engineering, ICECE 2020*, Dec. 2020, pp. 234–237. doi: 10.1109/ICECE51571.2020.9393129.

[12] X. Wei, Y. Chen, and Z. Zhang, "Comparative Experiment of Convolutional Neural Network (CNN) Models Based on Pneumonia X-ray Images Detection," in *Proceedings - 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence, MLBDBI 2020*, Oct. 2020, pp. 449–454. doi: 10.1109/MLBDBI51377.2020.00095.

[13] S. Y. Wang and T. Guo, "Transfer Learning-Based Algorithms for the Detection of Fatigue Crack Initiation Sites: A Comparative Study," *Front. Mater.*, vol. 8, Nov. 2021, doi: 10.3389/fmats.2021.756798.

[14] M. K. Nalini and K. R. Radhika, "Comparative analysis of deep network models through transfer learning," in *Proceedings of the 4th International Conference on IoT in Social, Mobile, Analytics and Cloud, ISMAC 2020*, Oct. 2020, pp. 1007–1012. doi: 10.1109/I-SMAC49090.2020.9243469.

[15] G. Fadilla, "Indonesian Wayang Types," *Kaggle*, 2022. https://www.kaggle.com/datasets/gibranfadilla/indonesian-wayang-types (accessed Oct. 26, 2022).

[16] B. Saha, "Drone-Bird Classification," *Kaggle*, 2022. https://www.kaggle.com/datasets/imbikramsaha/drone-bird-classification (accessed Oct. 26, 2022).

[17] M. Koklu, M. F. Unlersen, I. A. Ozkan, M. F. Aslan, and K. Sabanci, "A CNN-SVM study based on selected deep features for grapevine leaves classification," *Measurement*, vol. 188, p. 110425, Jan. 2022, doi: 10.1016/j.measurement.2021.110425.

[18] R. Gandikota and D. Mishra, "How You See Me: Understanding Convolutional Neural Networks," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, Oct. 2019, pp. 2069–2073. doi: 10.1109/TENCON.2019.8929603.

[19] F. D. Adhinata, G. F. Fitriana, A. Wijayanto, and M. P. K. Putra, "Corn Disease Classification Using Transfer Learning and Convolutional Neural Network," *JUITA J. Inform.*, vol. 9, no. 2, p. 211, Nov. 2021, doi: 10.30595/juita.v9i2.11686.

[20] F. D. Adhinata, D. P. Rakhmadani, M. Wibowo, and A. Jayadi, "A Deep Learning Using DenseNet201 to Detect Masked or Non-masked Face," *JUITA J. Inform.*, vol. 9, no. 1, p. 115, May 2021, doi: 10.30595/juita.v9i1.9624.

[21] E. Prasetyo, R. Purbaningtyas, R. D. Adityo, N. Suciati, and C. Fatichah, "Combining MobileNetV1 and Depthwise Separable convolution bottleneck with Expansion for classifying the freshness of fish eyes," *Inf. Process. Agric.*, Jan. 2022, doi:

10.1016/j.inpa.2022.01.002.

[22] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017.

[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ArXiv*, Sep. 2014.

[24] D. Hindarto and H. Santoso, "Plat Nomor Kendaraan dengan Convolution Neural Network," *J. Inov. Inform.*, vol. 6, no. 2, pp. 1–12, Oct. 2021, doi: 10.51170/jii.v6i2.202.

[25] M. E. Karar, E. E.-D. Hemdan, and M. A. Shouman, "Cascaded deep learning classifiers for computer-aided diagnosis of COVID-19 and pneumonia diseases in X-ray scans," *Complex Intell. Syst.*, vol. 7, no. 1, pp. 235–247, Feb. 2021, doi: 10.1007/s40747-020-00199-4.

[26] M. Harahap, E. M. Laia, L. S. Sitanggang, M. Sinaga, D. F. Sihombing, and A. M. Husein, "Deteksi Penyakit Covid-19 Pada Citra X-Ray Dengan Pendekatan Convolutional Neural Network (CNN)," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 6, no. 1, pp. 70–77, Feb. 2022, doi: 10.29207/resti.v6i1.3373.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.

[28] X. Li, X. Shen, Y. Zhou, X. Wang, and T.-Q. Li, "Classification of breast cancer histopathological images using interleaved DenseNet with SENet (IDSNet)," *PLoS One*, vol. 15, no. 5, p. e0232127, May 2020, doi: 10.1371/journal.pone.0232127.

[29] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 8697–8710. doi: 10.1109/CVPR.2018.00907.

[30] S. K. Addagarla, "Real Time Multi-Scale Facial Mask Detection and Classification Using Deep Transfer Learning Techniques," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 4, pp. 4402–4408, Aug. 2020, doi: 10.30534/ijatcse/2020/33942020.