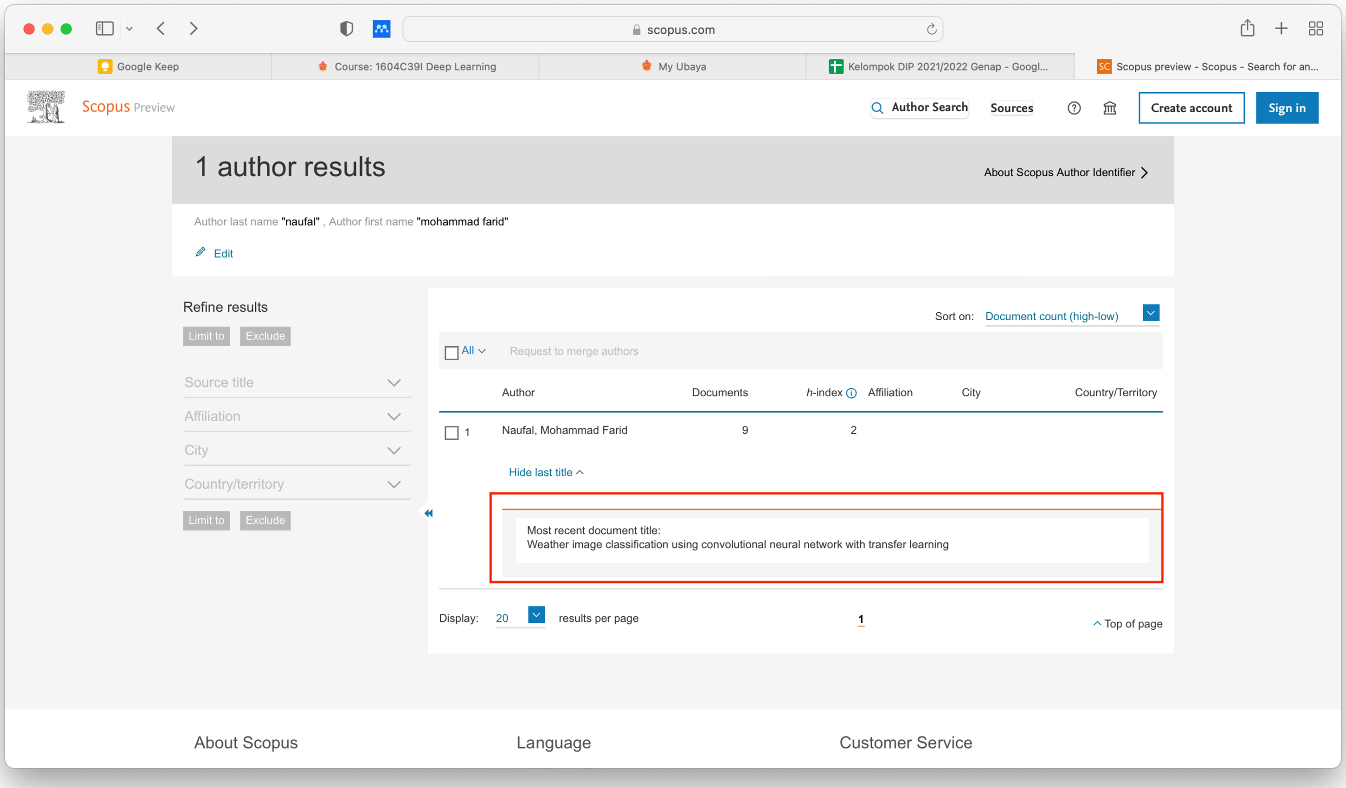


scopus.com

Google Keep | Course: 1604C39I Deep Learning | My Ubaya | Kelompok DIP 2021/2022 Genap - Googl... | Scopus preview - Scopus - Search for an...

Scopus Preview

Author Search | Sources | Create account | Sign in

1 author results

About Scopus Author Identifier

Author last name "naufal" , Author first name "mohammad farid"

Edit

Refine results

Limit to | Exclude

Source title

Affiliation

City

Country/territory

Limit to | Exclude

Sort on: Document count (high-low)

All | Request to merge authors

	Author	Documents	*h*-index	Affiliation	City	Country/Territory
1	Naufal, Mohammad Farid	9	2			

Hide last title

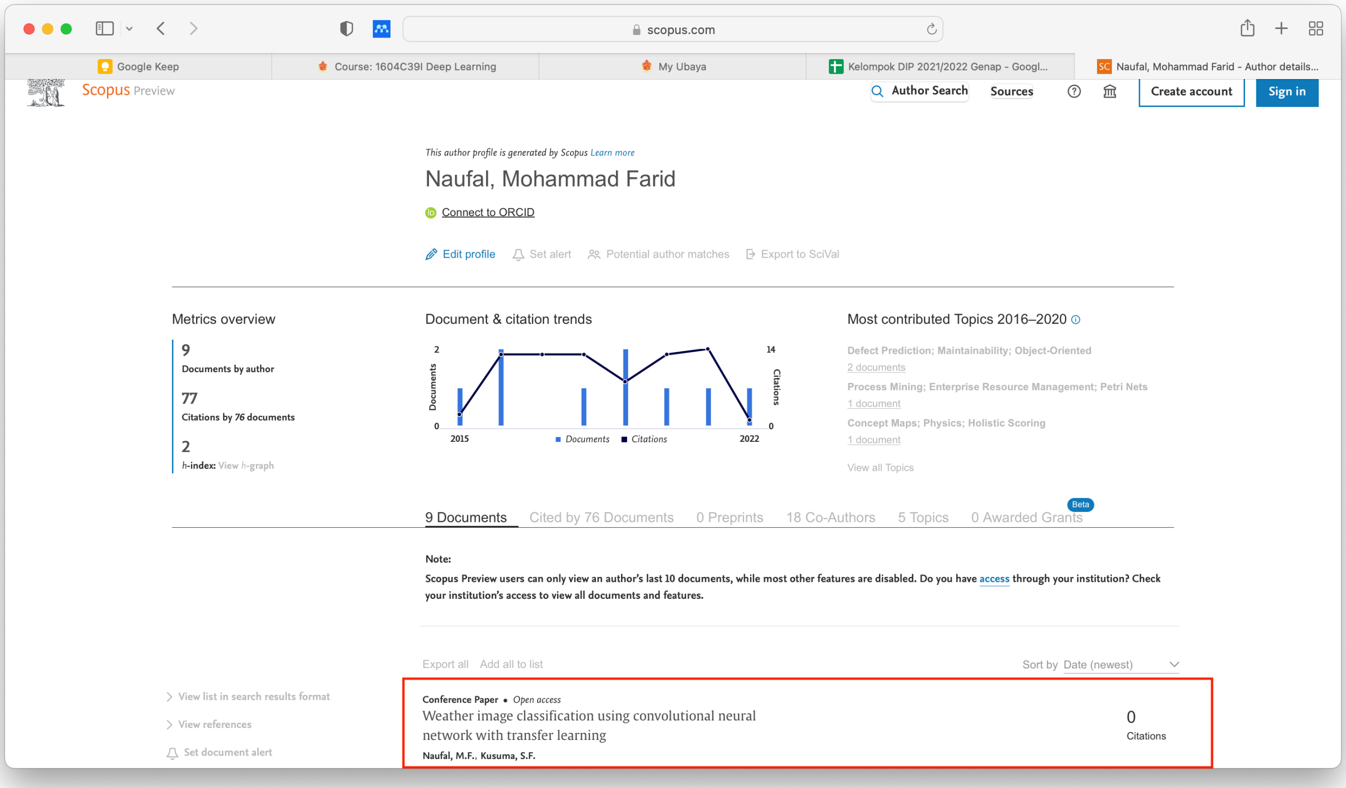
Most recent document title:
Weather image classification using convolutional neural network with transfer learning

Display: 20 results per page

1

Top of page

About Scopus | Language | Customer Service


scopus.com
Google Keep
Course: 1604C39I Deep Learning
My Ubaya
Kelompok DIP 2021/2022 Genap - Googl...
Naufal, Mohammad Farid - Author details...
Scopus Preview
Author Search
Sources
Create account
Sign in

This author profile is generated by Scopus *Learn more*

Naufal, Mohammad Farid

Connect to ORCID

Metrics overview

9
Documents by author

77
Citations by 76 documents

2
h-index: View *h*-graph

Document & citation trends

Most contributed Topics 2016–2020

Defect Prediction; Maintainability; Object-Oriented
2 documents

Process Mining; Enterprise Resource Management; Petri Nets
1 document

Concept Maps; Physics; Holistic Scoring
1 document

View all Topics

Note:
Scopus Preview users can only view an author's last 10 documents, while most other features are disabled. Do you have access through your institution? Check your institution's access to view all documents and features.

Conference Paper • *Open access*
Weather image classification using convolutional neural network with transfer learning
Naufal, M.F., Kusuma, S.F.

0
Citations

Weather image classification using convolutional neural network with transfer learning

Cite as: AIP Conference Proceedings **2470**, 050004 (2022); https://doi.org/10.1063/5.0080195
Published Online: 25 April 2022

Mohammad Farid Naufal and Selvia Ferdiana Kusuma

View Online Export Citation

Weather Image Classification using Convolutional Neural Network with Transfer Learning

Mohammad Farid Naufal[1, a)] and Selvia Ferdiana Kusuma[2, b)]

[1]*Faculty of Engineering, Universitas Surabaya, Surabaya 60293, Indonesia*
[2]*Manajemen Informatika, PSDKU Politeknik Negeri Malang, Kediri, Indonesia*

[a)] Corresponding author: faridnaufal@staff.ubaya.ac.id
[b)] selvia.ferdiana@polinema.ac.id

Abstract. Weather condition is an important factor that is considered for various decisions. In the industrial world, weather classification is very useful, such as in development of self-driving cars, smart transportation systems, and outdoor vision systems. Manual weather classification by humans is inconsistent and takes a long time. Weather forecast information obtained from the internet is not real time at a specific location. Weather image has unique characteristics because one type of weather can be like another. Computer vision is a branch of computer science to recognize or classify images that can assist in classifying weather images that do not depend on weather forecast information from the internet. This study aims to classify weather images using Convolutional Neural Network (CNN) with Transfer Learning. Four CNN architectures, MobileNetV2, VGG16, DenseNet201, and Xception were used to perform weather image classification. Transfer learning was used to speed up the process of training models to get better performance faster. The proposed method will be applied to the weather image which consists of six classes, cloudy, rainy, shine, sunrise, snowy, and foggy were classified in this study. The experiment result with 5-cross validation and 50 epochs showed that the Xception has the best average accuracy of 90.21% with 10,962 seconds of average training time and MobileNetV2 has the fastest average training time of 2,438 seconds with 83.51% of average accuracy.

Keywords: CNN, Transfer Learning, Weather Classification

INTRODUCTION

In recent years, computers have the ability to do various things. Computers can make observations of satellite images and determine the weather on that day and make forecasts for the next weather [1]. By using internet connection, all computers can access this information. However, the weather information can be different from one place to another. Even though the computer gets current weather information from the internet, it doesn't mean that the information is the same as the current location. In the industrial world the real time weather classification is very useful. Vision assisted transportation system such as self-driving car can use weather image for assistance and activate the wipers if in rainy situation. It can also inform the car of making decisions in real time based on current weather.

Weather image has unique characteristics. It is very depending on lighting, cloud conditions, rainwater, and snow. This is a challenge in classifying weather images that have similarities between one weather and another. For example, foggy is similar to snowy and cloudy is similar to rainy. Image classification is one of the latest technological areas that can replace human visual abilities [2]. By using image classification, the computer can see the weather only based on weather images in real time. The weather image classification application can assist in the development of an independent autonomous machine or Advance Driver Assistance System (ADAS) [3].

Kang et. al. [3] used CNN with GoogleNet and AlexNet architecture to classify the weather into 4 classes, hazy, rainy, and snowy or none of them. The accuracy produced by GoogleNet was 92.0% and AlexNet was 91.1%. However, there is no information related to where the dataset was obtained, the distribution of testing and training data was not explained. Xia et al. [4] conducted comparison of several CNN architectures, AlexNet, VGG16, and GoogleNet for classifying weather images into 4 classes, foggy, rainy, snowy, and sunny. AlexNet has the best accuracy of 86.47%. However, there is no information about computational time of each architecture. Elhoseiny et al. [5] using CNN with AlexNet architecture to classify weather image into two classes, sunny and cloudy. This study uses the ImageNet database to build the pre-trained model. However, due to Alexnet's simple architecture, the accuracy obtained was 82.2%. Notarangelo et al. [6] used CNN and transfer learning to classify weather image. However, it only used VGG16 architecture and binary label classification. The weather image was labelled as With Rain (WR) and No Rain (NR). The dataset used in this study was obtained from Image2Weather which is dash cam photo in Tokyo Metropolitan Area. Transfer learning dataset was obtained from the ImageNet [7]. The accuracy produced by this study was 85.28%.

Transfer learning is a method that solve basic problem of insufficient training data in Machine Learning [8]. It will transfer the knowledge from one domain to another target domain with assumption that the training and testing data are independent and identically distributed. In computer vision, transfer learning is used to speed up the training process to get better performance. The pre-trained model was trained in several image dataset, then it will retrain to source dataset.

From the previous studies that have been conducted, there is still no research that attempts to classify multiclass weather images using several CNN architectures with transfer learning. This study uses several CNN architectures, MobileNetV2, VGG16, DenseNet201, and Xception to classify weather images that have 6 weather types, cloudy, foggy, rain, shine, snow, and sunrise. The transfer learning is used to get a better performing classification model faster. The transfer learning dataset used in this study was ImageNet [7]. The dataset used in this study was obtained from Kaggle [9] and Camera as Weather Sensor (CWS) [10] that can be accessed publicly with the aim of this study being comparable with further research. The performance metrics calculated to validate this study are accuracy, precision, recall, F1 score, and computational time.

METHODOLOGY

The research methodology consists of collecting datasets, dataset augmentation, hyperparameter configuration, training the classification models, testing the classification models, and calculating the performance of classification models. The steps of research methodology can be seen in Fig. 1.

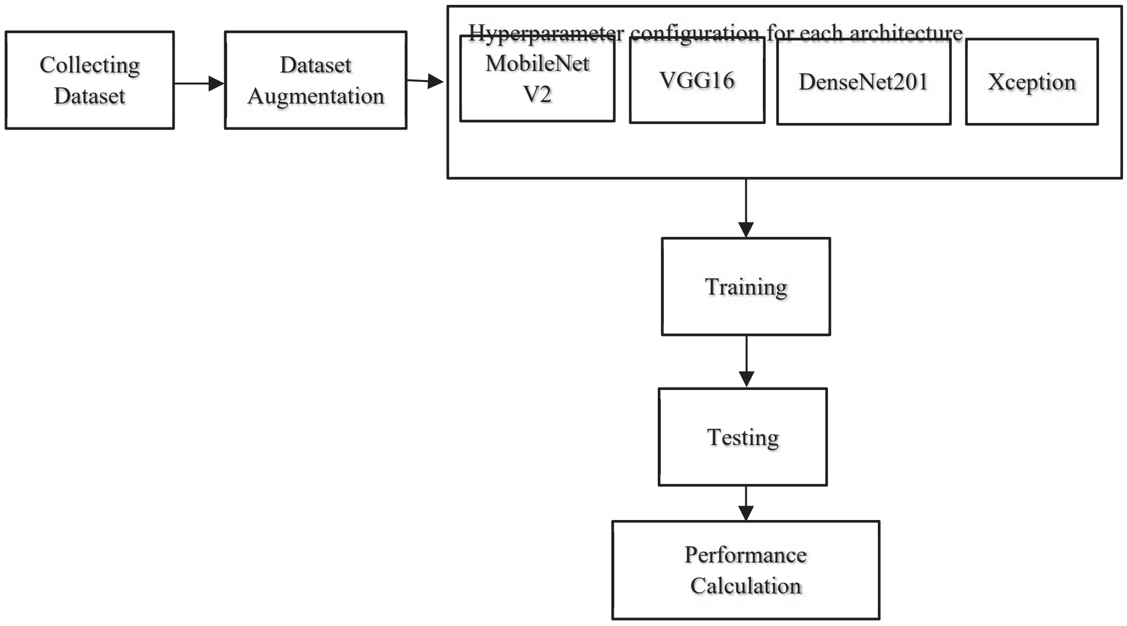


FIGURE 1. Research Methodology Steps

Collecting Dataset

This study used 1,816 weather images which were obtained from the Multi-class Weather dataset for image classification in the Kaggle repository [9] and Camera as Weather Sensor (CWS) repository [10]. In Kaggle repository, there are four weather classes, cloudy, rainy, shine, and sunrise. In CWS repository there are two weather classes, snowy and foggy. Table 1 shows the number of datasets of each weather class. Figure 2 shows an example dataset of each class. The pixel size is resized to 64x64 prior to training and testing process due to weather image has different pixel size from one another.

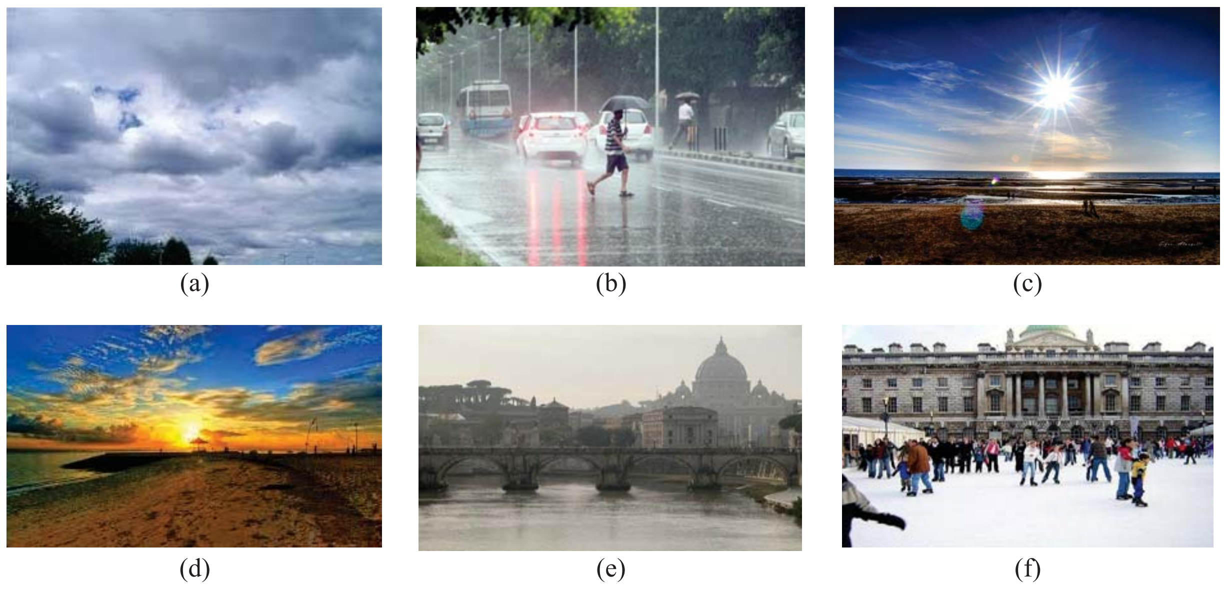


FIGURE 2. Example image of cloudy (a), rain (b), shine (c), sunrise (d), snowy (e), and foggy (f)

TABLE 1. Dataset Distribution

Class	Number of images	Source
Cloudy	300	Kaggle
Rain	215	Kaggle
Shine	253	Kaggle
Sunrise	357	Kaggle
Snowy	334	CWS
Foggy	357	CWS
Total	**1816**	

Data Augmentation

Data Augmentation enriches training data to avoid overfitting. Data augmentation uses libraries provided by Keras [11]. The data augmentation steps consist of horizontal flip, shear range with a value of 0.2 and zoom range with a value of 0.2. Horizontal flip is used to make the image more varied because the training data is added with an image that is rotated horizontally 90 degrees. Shear range uses shear transformation [12] to make an image more varied with a certain degree of rotation, and zoom range is used to enlarge the image by a certain percentage of the original image.

Hyperparameter Configuration

At this stage, several parameters related to the neural network are configured. The parameters configured in this study are the type of architecture, transfer learning database, optimizer, loss function, and number of epochs. Table 2 shows the hyperparameter configuration. MobileNetV2, VGG16, DenseNet201, and Xception were chosen as CNN architectures to classify the weather images for reasons that will be discussed in this study. Those architectures performance to classify the weather images will be compared.

MobileNetV2 still uses depthwise and pointwise convolution like MobileNetV1. MobileNetV2 adds two new features: 1) linear bottlenecks, and 2) shortcut connections between bottlenecks [13]. In the bottleneck section there are inputs and outputs between the models while the inner layer or layer encapsulates the model's ability to change the input from lower-level concepts (i.e. pixels) to higher level descriptors (i.e. image category). Ultimately, as with residual connections on traditional CNNs, shortcuts between bottlenecks allow for faster training and greater accuracy. In this study, MobileNetV2 is used to classify weather image because it has a fast-training process with good performance.

VGG16 is a CNN with 16 layers deep [14]. The model gets 92.7% of accuracy on top-5 test in ImageNet database which has over 14 million images belonging to 1000 classes. The model has 224x224 default input size with 3 channels of RGB image. The model has 3x3 filter of convolution layers with a stride 1 and 2x2 maxpool layer with stride 2. This study used VGG16 to classify weather image because it is commonly used architecture in various applications. VGG16 also commonly used architecture with pre-trained model but it is very slow if trained from scratch.

DenseNet201 connects each layer to every other layer in a feed-forward fashion [15]. In this study, DenseNet201 used to classify weather images because it has some advantages, it reduces vanishing-gradient problem, strengthen propagation of feature, uplift feature reuse, and reducing the number of parameters.

Xception stands for Extreme version of Inception [16]. In this study, Xception used to classify weather because it is even better than previous version, Inception-v3 [17]. Xception has the same number of model parameter as Inception-v3 but it outperforms Inception-v3 on ImageNet dataset with 17,000 classes.

ImageNet is large scale image database which is built upon WordNet structure backbone [18]. ImageNet very useful in object recognition, image classification, and image clustering. In term of transfer learning, ImageNet database is used as dataset to train CNN models to build the pre-trained model. In this study, the pre-trained model was obtained from ImageNet database in Keras library.

Adam Optimizer is the extension of Stochastic Gradient Descent (SGD) which has broader adoption in deep learning for image classification [19]. Adam optimizer is computationally efficient, low memory requirement, and easy to implement. This study uses Adam optimizer to change the attributes of neural networks in order to minimize the losses.

Categorical Cross Entropy is a Cross Entropy loss function for multiclass classification. This study uses Categorical Cross Entropy that will train the CNN probability output over six weather classes. Softmax activation function is used in the output layer of CNN architecture since it is the only activation function recommended for Categorical Cross Entropy.

The number of epochs used in this study were 50. The reason for determining the number of epochs is the consideration of training time and the specifications of the computer used to train the model. Based on the results that will be discussed in the results chapter, 50 epochs are sufficient to make the accuracy for training data convergent enough.

TABLE 2. Hyperparameter Configuration

Hyperparameter	Type
Architecture	MobileNetV2, VGG16, DenseNet201, Xception
Transfer Learning database	ImageNet
Optimizer	Adam
Loss Function	Categorical Cross Entropy
Number of epochs	50

Training

At this stage, training process is carried out on each architecture. The training was repeated and carried out 5 times. From each time of the training process is carried out a trial using 5 cross validations. The experiment was carried out on a computer with the specifications Intel® Core™ i3-3220 CPU @ 3.30GHz (4 CPUs), 8 GB of RAM, AMD Radeon 5500 Series, and 256 GB SSD. The distribution of dataset will be divided into 80% for training, 10% for validation, and 10% for testing. Details of the use of cross validation will be explained in more detail at the testing stage.

The pre-trained models for each architecture are obtained from the ImageNet database in Keras Library. The pre-trained model used for the training process will be modified by not loading the fully connected output layer. This is done because the number of classes on the weather image is different with the class on the pre-trained model. The pre-trained model is used as a feature extractor whose output will be max pooling first then become as input to the artificial neural network (ANN) l. The ANN consists of two hidden layers and one output layer. Each hidden layer has 128 and 64 neurons using the ReLu activation function. The output layer consists of 6 neurons using the Softmax activation function. Equation (1) shows the ReLu activation function and x is the activation function input. Equation (2) shows the calculation of the Softmax activation function with input x_i, the number of class labels n, and the j-th class labels.

$$R(x) = \max(0, x) \tag{i)}$$

$$S(x_i) = \frac{e^{x_i}}{\sum_j^n e^{x_j}} \tag{ii)}$$

This study also uses the data augmentation in order to avoid overfitting. Data augmentation using libraries provided by Keras [11]. The data augmentation used are horizontal flip, shear range with a value of 0.2, and zoom range with a value of 0.2. Horizontal flip is used to make the image more varied because the training data is added with an image that is rotated horizontally 90 degrees. Shear range uses shear transformation [12] to make an image more varied with a certain degree of rotation and zoom range is used to enlarge the image by a certain percentage of the original image.

Testing

In each epoch, a validation process derived from 10% of the dataset is carried out with validation data to see its accuracy. However, the accuracy obtained from the validation data is not used to update the weight on the neural network layer. In other hand, the accuracy of the model obtained from the training data is used to update the weight on the neural network layer.

Testing was performed using 10% of the weather dataset randomly. Testing is done using 5 cross validations and repeated 5 times. In each algorithm, the performance of each cross validation will be observed. This is done to see whether the resulting performance of each architecture is stable or not.

Performance Calculation

At this stage the performance calculations of MobileNetV2, VGG16, DenseNet201, and Xception architecture are carried out. The calculated performance is accuracy, precision, recall, F1 Score for multiclass classification and computational training time. Performance calculations are performed in each cross validation. Equation (3) (4) (5) (6) shows the calculation formula of accuracy, precision, recall, and F1 Score, respectively. TP is True Positive, TN is True Negative, FP is False Positive, and FN is False Negative, n is the number of classes. Computational training time is calculated when the model performs training of 50 epochs.

$$Accuracy = \sum_{i=1}^{n} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{iii)}$$

$$Precision = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i} \tag{iv)}$$

$$Recall = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i} \quad \text{v)}$$

$$F1\ Score = \sum_{i=1}^{n} \frac{Precision_i\ x\ Recall_i}{Precision_i + Recall_i} \quad \text{vi)}$$

RESULTS AND DISCUSSION

In this stage, the performance of each architecture is discussed. Table 3 shows the performance results of each architecture. Xception has the highest average accuracy of 0.9022 when compared to other architectures. The purpose of using average accuracy is performance calculations can be more generalized because the experiment was carried out using 5 cross validations. Xception also has a stable accuracy in every cross validation. The highest accuracy gap in the Xception is at 2nd and 3rd cross validation which is 0.0549. The DenseNet201 also has the best accuracy of 0.9285 at the 4th cross validation. However, the average accuracy of Xception is still the best. Xception's training time is also still better when compared to DenseNet201.

MobileNetV2 has the fastest average training time of 2,483 seconds. This faster training time is caused by shortcuts between bottlenecks allow for faster training. However, the average accuracy of MobileNetV2 is still far below Xception, which is 0.8351 although MobileNetV2 training time is 4.41 times faster than Xception. MobileNetV2 can be chosen as one of the architectures to classify weather images if the computer used for the training process has low specifications.

The VGG16 has the longest average training time of 15,361 seconds. In addition, VGG16 has an average accuracy of 0.6835 which is not good enough. Although VGG16 is the architecture commonly used for image classification, it is not suitable for weather image classification.

TABLE 3. Performance results of each Architecture

Architecture	Metric	CV1	CV2	CV3	CV4	CV5	Average
MobileNetV2	Accuracy	0.7912	0.8626	0.8626	0.8846	0.7747	0.8351
	Precision	0.7912	0.8626	0.8626	0.8846	0.7747	0.8351
	Recall	0.7912	0.8626	0.8626	0.8846	0.7747	0.8351
	F1 Score	0.7912	0.8626	0.8626	0.8846	0.7747	0.8351
	Train Time	2203	2204	2646	2637	2723	2483
VGG16	Accuracy	0.7967	0.8406	0.3516	0.7033	0.7252	0.6835
	Precision	0.7967	0.8406	0.3516	0.7033	0.7252	0.6835
	Recall	0.7967	0.8406	0.3516	0.7033	0.7252	0.6835
	F1 Score	0.7967	0.8406	0.3516	0.7033	0.7252	0.6835
	Train Time	17237	14669	14937	14998	14966	15361
DenseNet201	Accuracy	0.8626	0.8956	0.9011	0.9285	0.8791	0.8934
	Precision	0.8626	0.8956	0.9011	0.9285	0.8791	0.8934
	Recall	0.8626	0.8956	0.9011	0.9285	0.8791	0.8934
	F1 Score	0.8626	0.8956	0.9011	0.9285	0.8791	0.8934
	Train Time	10346	12370	12594	12793	12982	12217
Xception	Accuracy	0.8901	0.9395	0.8901	0.8846	0.9065	0.9022
	Precision	0.8901	0.9395	0.8901	0.8846	0.9065	0.9022
	Recall	0.8901	0.9395	0.8901	0.8846	0.9065	0.9022
	F1 Score	0.7912	0.8626	0.8626	0.8846	0.7747	0.8351
	Train Time	10352	11057	11051	11078	11270	10962

Figure 3 shows graphs of the accuracy of each epoch for the MobileNetV2 and VGG16 while Fig. 4 shows for DenseNet201 and Xception from 1 to 5 cross validation. The blue line shows the accuracy for the training set while the yellow dotted line for validation set.

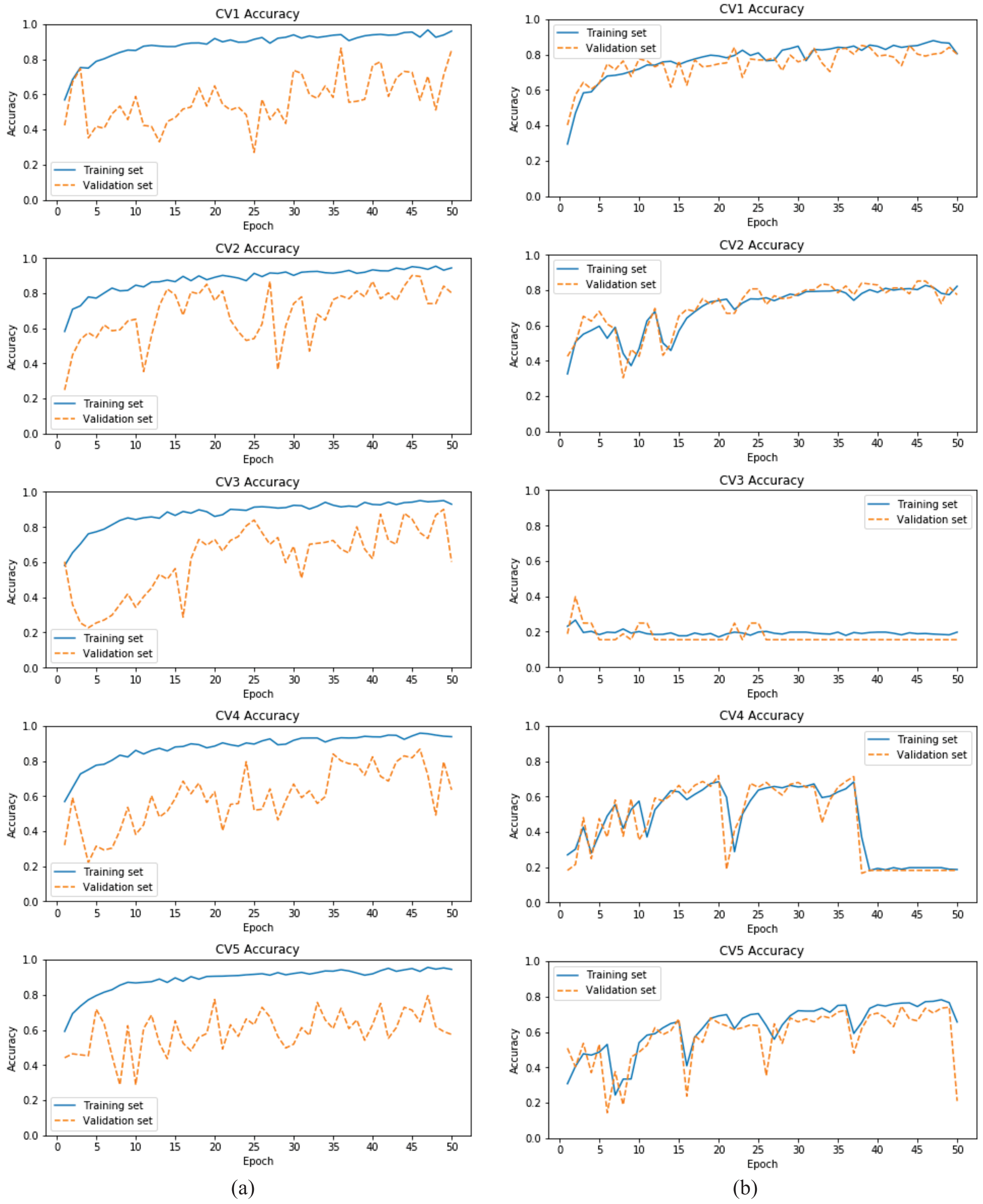


FIGURE 3. Graph of the accuracy of each epoch for the MobileNetV2 (a) and VGG16 (b) from 1 to 5 cross validation

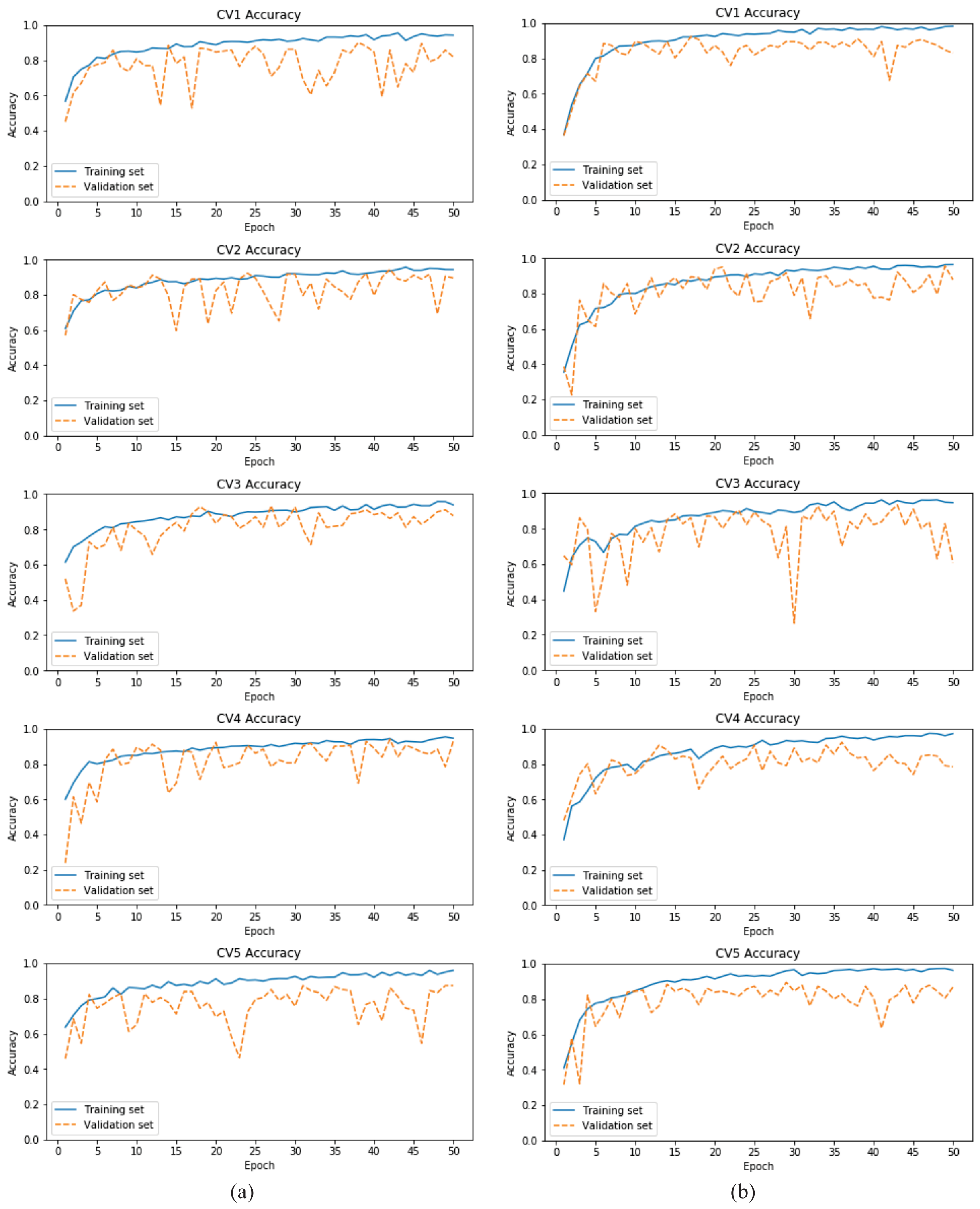


FIGURE 4. Graph of the accuracy of each epoch for the DenseNet201 (a) and Xception (b) from 1 to 5 cross validation

Based on the graph of training and validation set accuracy each cross validation, MobileNetV2, DenseNet201, and Xception have convergent accuracy as the epoch increases. However, the accuracy of the VGG16 is not convergent enough and this has an impact on the poor accuracy in the testing set. Xception has a narrow gap between the training and validation set accuracy in each epoch which has an impact on the good accuracy of the testing set.

Xception has the best accuracy when compared to other architectures for classifying the weather images. Xception is recommended for weather image classification. However, Xception requires long time to train the model, but it still better when compared to VGG16 and DenseNet201. MobileNetV2 has the fastest time to train the model. However, the accuracy is not the best, but it still above 80% which is 83.51%.

CONCLUSION

Weather image classification problems in 6 types of weather, cloudy, rainy, shine, sunrise, snowy, and foggy can be solved using CNN with transfer learning. Based on the experiment using 5 cross validations and 50 epochs, Xception using transfer learning from the ImageNet database can classify weather images with average accuracy and training time of 90.21% and 10.962 seconds. MobileNetV2 with transfer learning can classify weather images with an average accuracy of 83.51% and 2.438 seconds, respectively.

In the industrial world, CNN model with transfer learning is expected to be implemented directly for weather image classification which is useful for developing self-driving cars, smart transportation systems, and outdoor vision systems. Xception can be chosen as the architecture that has the best accuracy but requires a longer training time. If a computer with high specifications is available for model training, then Xception is the right choice. However, if the computer has low specifications, then MobileNetV2 can be chosen because it has a faster training time. For further research, there is a chance to apply weather classification with their intensity using CNN and transfer learning. The experiment for choosing the architecture of ANN which has best performance also need to be conducted.

REFERENCES

1. "Automotive Revolution & Perspective Towards 2030," 2016. doi: 10.1365/s40112-016-1117-8.
2. B. Javidi, *Image Recognition and Classification*. CRC Press, 2002.
3. L. W. Kang, K. L. Chou, and R. H. Fu, "Deep learning-based weather image recognition," *Proc. - 2018 Int. Symp. Comput. Consum. Control. IS3C 2018*, pp. 384–387, 2019, doi: 10.1109/IS3C.2018.00103.
4. J. Xia, D. Xuan, L. Tan, and L. Xing, "ResNet15: Weather Recognition on Traffic Road with Deep Convolutional Neural Network," *Adv. Meteorol.*, vol. 2020, 2020, doi: 10.1155/2020/6972826.
5. M. Elhoseiny, S. Huang, and A. Elgammal, "Weather classification with deep convolutional neural networks," *Proc. - Int. Conf. Image Process. ICIP*, vol. 2015-Decem, no. September, pp. 3349–3353, 2015, doi: 10.1109/ICIP.2015.7351424.
6. N. M. Notarangelo, K. Hirano, R. Albano, and A. Sole, "Transfer learning with convolutional neural networks for rainfall detection in single images," *Water (Switzerland)*, vol. 13, no. 5, pp. 1–17, 2021, doi: 10.3390/w13050588.
7. "ImageNet." https://www.image-net.org/index.php (accessed May 27, 2021).
8. C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11141 LNCS, pp. 270–279, 2018, doi: 10.1007/978-3-030-01424-7_27.
9. "Multi-Class Images for Weather Classification | Kaggle." https://www.kaggle.com/somesh24/multiclass-images-for-weather-classification (accessed May 27, 2021).
10. W. T. Chu, X. Y. Zheng, and D. S. Ding, "Camera as weather sensor: Estimating weather information from single images," *J. Vis. Commun. Image Represent.*, vol. 46, pp. 233–249, Jul. 2017, doi: 10.1016/j.jvcir.2017.04.002.
11. F. Chollet and & O., "Keras: the Python deep learning API," *Keras: the Python deep learning API*, 2020. https://keras.io/ (accessed Dec. 18, 2020).
12. R. N. Goldman, "More matrices and transformations: Shear and pseudo-perspective," in *Graphics Gems II*, Elsevier Inc., 1991, pp. 338–341.

13. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4510–4520, Jan. 2018, Accessed: May 27, 2021. [Online]. Available: http://arxiv.org/abs/1801.04381.
14. K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” Sep. 2015, Accessed: May 27, 2021. [Online]. Available: http://www.robots.ox.ac.uk/.
15. G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 2261–2269, Aug. 2016, Accessed: May 27, 2021. [Online]. Available: http://arxiv.org/abs/1608.06993.
16. F. Chollet, “XCeption: Deep Learning with Depthwise Separable Convolutions,” *Comput. Vis. Found.*, 2016, doi: 10.4271/2014-01-0975.
17. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 2818–2826, 2016, doi: 10.1109/CVPR.2016.308.
18. L. Fei-Fei, J. Deng, and K. Li, “ImageNet: Constructing a large-scale image database,” *J. Vis.*, vol. 9, no. 8, pp. 1037–1037, 2010, doi: 10.1167/9.8.1037.
19. D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.