An Xception Based Convolutional Neural Network for Scene Image Classification with Transfer Learning

Xizhi Wu
Department of Intelligence and Computing
Tianjin University
Tianjin, China
cswxz@tju.edu.cn

Rongzhe Liu
Department of Cyber Engineering
Dalian University of Technology
Dalian, China
lllrz@mail.dlut.edu.cn

Hanqing Yang
Department of Computer Science and Engineering
Washington University in St. Louis
St. Louis, United States
alberty@wustl.edu

Zizhao Chen
Department of Mathematics
University of Chinese Academy of Sciences
Beijing, China
chenzizhao15@mails.ucas.ac.cn

***Abstract*—Over the past decade, image classification, which can provide assistance to address complex tasks such as planetary exploration and unmanned driving, has become a hot topic. As a subproblem of image classification, scene image classification has received increasing attention. Based on previous studies, the Xception model achieved superior performance on image classification tasks in comparison with the original Inception model. The Xception model is advantageous at processing image classification, yet it has not been used for scene image classification. To tackle this issue, this paper proposed an Xception based transfer learning, and analyzed the model performance by comparing it with the Inception-V3 model. We found that the Xception based transfer learning significantly outperforms other methods such as Inception-V3, which is nicely demonstrated by the experimental results on the Intel Image Classification Challenge dataset. Furthermore, the Xception has shown greater robustness and ability in generalization with less overfitting problems.**

Keywords- Xception model; Inception-V3 model; Convolutional neural network; Transfer learning; Scene image classification Introduction

I. INTRODUCTION

Deep-learning based image classification has received an intensive level of attention from researchers thanks to the available image databases such as ImageNet [1]. This paper focuses on scene classification in photographs. Since a scene is often composed of several entities organized in an unpredictable layout, scene classification differs from the conventional object classification.

In 2017, François Chollet proposed the Xception model, developed from Inception-V3 model, which replaced the Inception modules with depth wise separable convolutions to use parameters more efficiently [2]. However, the classifications of scenes remain challenging if only based on ImageNet data.

This paper describes a novel scene classification method using transfer learning on the pre-trained Xception model based on ImageNet database. The model is proved to be capable of classifying scenes such as glaciers and mountains that are not included in the pre-trained dataset. The experimental results show that the accuracy of transfer learning on Inception-V3 classification reached 91.81%, whereas transfer learning on Xception reached 91.20%. By investigating the influence of the source datasets, we found that the transfer learning method was able to incorporate both relevant and seemingly irrelevant source datasets for pretraining, and the relevant source dataset brought better classification accuracy than that of the seemingly irrelevant source dataset. This study demonstrates that the transfer learning technique has great potential in effective identification of random image data sets when the number of image data is limited. The main contributions of this work can be listed as follows:

1. The performance of Xception based transfer learning in scene image classification is analyzed.
2. The generalization performance of Xception and Inception-V3 model is verified and compared comprehensively.
3. We prove the effectiveness of transfer learning on the Xception model to classify specific datasets.

The rest of this paper is organized as follows. In Section 2, background information is introduced to explain the structure of Xception and Inception-V3. Section 3 introduces the related methods and materials in detail. In Section 4, the performance of the proposed model is verified and analyzed. Finally, in Section 5, a summary of the performance of the model is given and suggestions for future work are provided.

Identify applicable sponsor/s here. *(sponsors)*

II. BACKGROUND INFORMATION

The architectures of Inception and Xception are explained in detail in this Section. In addition, we compare and analyze the performance of the two models.

A. *Inception-V3*

Inception was originally proposed by Szegedy et al., containing 42 layers [3]. Inception-V3, the third generation of the original model, was proposed by Google Brain, containing 159 layers [4].

Specifically, Inception-V3 can be regarded as three parts: convolution layers, Inception modules, and classifiers (as shown in Figure 1). The Inception module is designed based on Network-In-Network, in which multiple convolution layers are performed in parallel to scale up the network, and the convolution results of each branch are then concatenated (as illustrated in Figure 2) [5]; Inception-V3 achieves superior performance in object recognition compared with the original model, including classifications in flowers [6], apparels [7], fast foods [8], and traffic signs [5]. Owing to the ability and advantage of Inception-V3 model, this study adopts it as the baseline model for comparison.

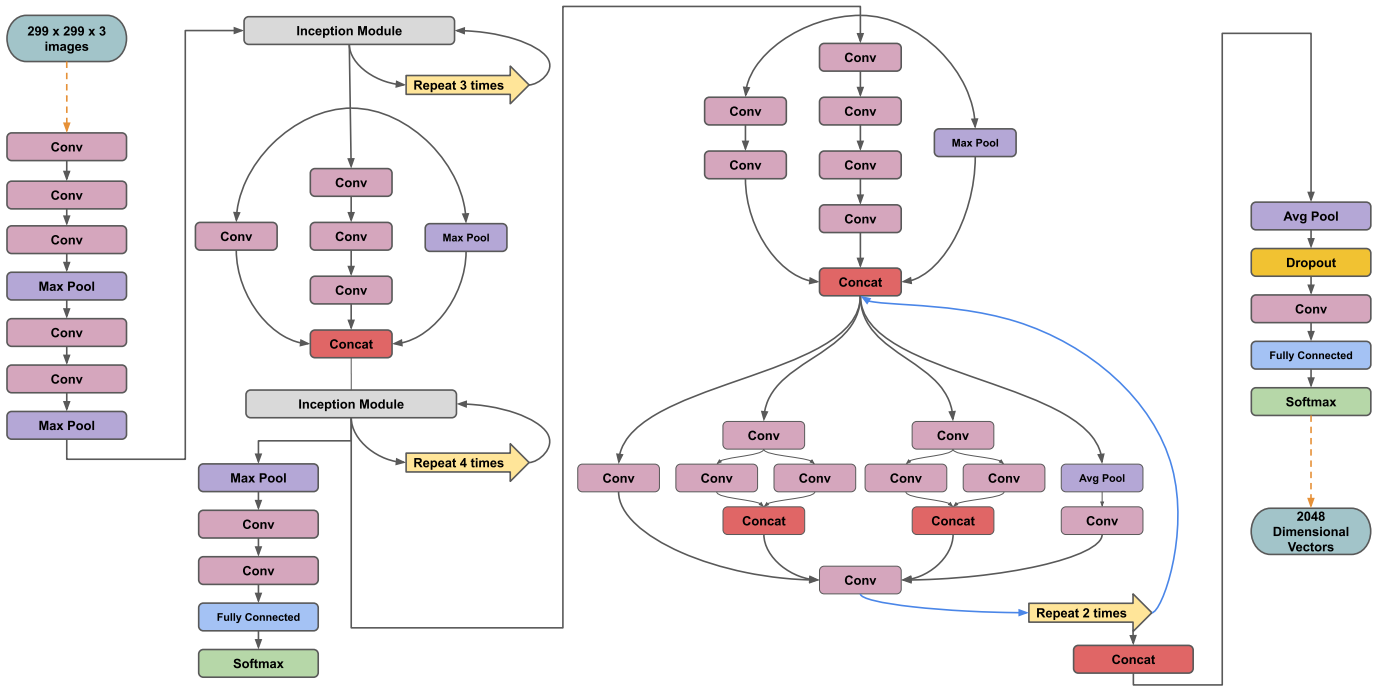

Figure 1. Architecture of Inception-V3

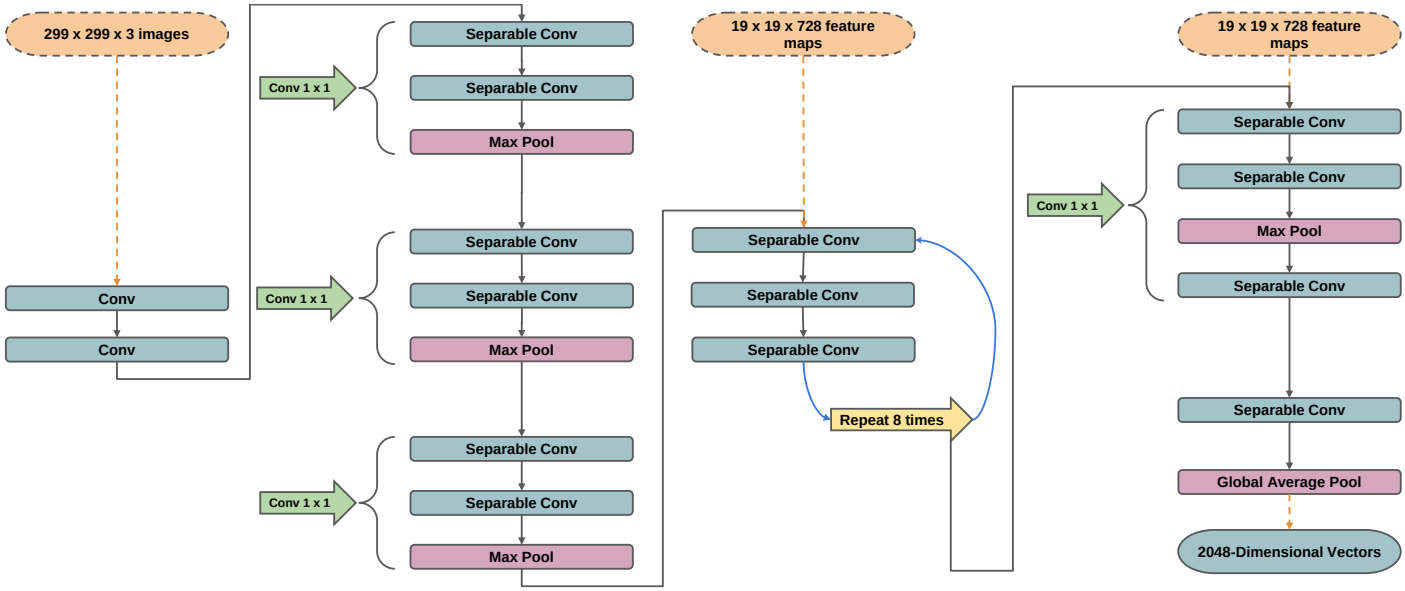

Figure 3. Architecture of Xception

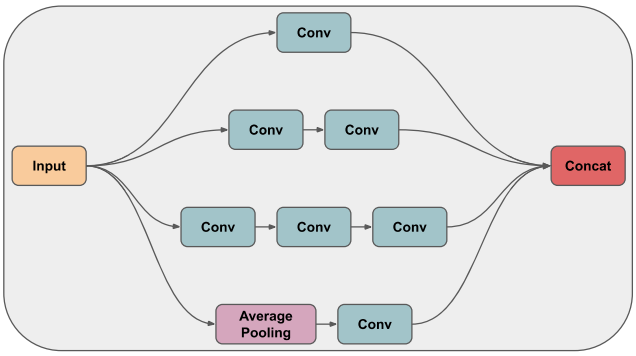

Figure 2. Inception module

B. *Xception*

Xception is based on Inception-V3, using a linear stack of depth wise separable convolution layers with residual connections to reduce time and space complexity, with more details shown in Figure 3 [2].

The depth wise separable convolution in Xception separates the learning of channel-wise and space-wise features. Moreover, the residual connection is used to solve the problem of vanishing gradients and representational bottlenecks by creating a shortcut in the sequential network [9].

C. *Xception vs Inception*

A typical Inception module is illustrated in Figure 2. The Inception structure is similar to using a 1x1 convolution layer to learn the association of features between channels from the input feature map, and then segment the output feature map and process the following 3x3 convolution layer to deal with the association of spatial elements [10]. In addition, the depth wise separable convolution uses a corresponding 3x3 convolution layer to handle the associations of spatial elements on each channel separately. As shown in Figure 4, the process produces a single convolution only. The extreme form of this Inception module is almost the same as the depth wise separable convolution, which has been used in neural network design as early as 2014 [2].

Based on these factors, the depth wise separable convolution is able to replace the Inception module, which could be used to improve the structure of the Inception series by constructing a model of stacked depth wise separable convolutions.

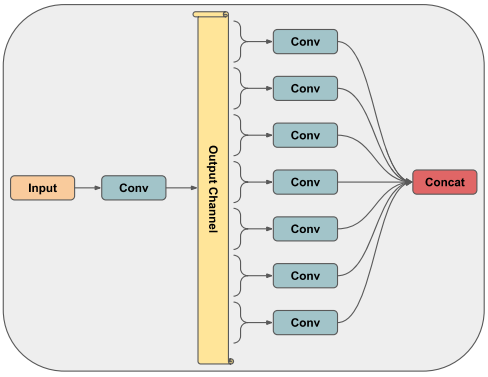

Figure 4. Extreme Inception module

III. Materials and Methods

The materials and methods used in this paper are discussed in this section. Firstly, the experimental environment is introduced. Then, the dataset used in creating the model, Intel Image Classification dataset, is examined. Moreover, the data processing and transfer learning techniques are introduced in Sections 3.3 and 3.4. Finally, the experimental method is described in detail in Section 3.5.

A. *Experimental Environment*

The experiment is carried out on a HASEE Z8-KP7S2 running Windows 10, which has an I7-7700HQ processor and an Nvidia GTX 1070 graphics card. Python 3.7.0 was used. The libraries used to create the test models are TensorFlow 2.1.6 and Keras 2.1.1. Furthermore, Python Imaging Library (PIL) is used to read data; Matplotlib, Seaborn, and Skearn are used to help visualize the models.

B. *Data*

The Intel Image Classification dataset is used for the transfer learning [11]. This dataset contains 25000 images, 150 x 150 pixels in size, distributed over six different natural scenes, including forests, mountains, oceans, glaciers, buildings, and streets. The original dataset is divided into three sets: training set and validation set, with 14034, 3000, and 7301 images, respectively (shown in Figure 5).

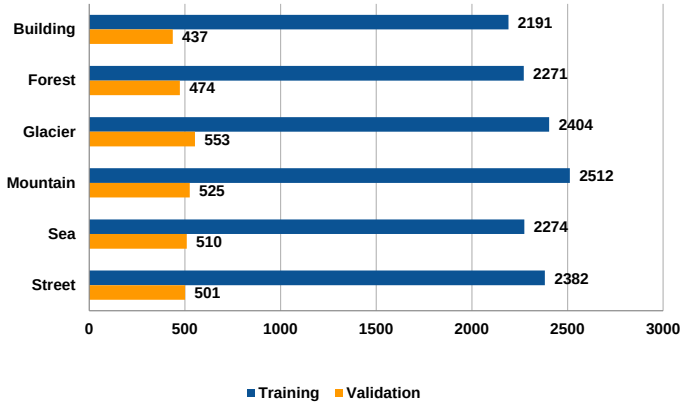

Figure 5. Distribution of images in Intel Images Classification challenge datasets

C. *Data Preprocessing*

Data preprocessing is important because it prepares a dataset for further analysis. Figure 6 shows an overview of how we deal with the dataset. Image data is first transformed from image data to matrix form using vectorization. Vectorization is used to convert an image from three channels to a matrix by scaling each value from 0 to 255 to 0 to 1, which is essential for further processing of the data.

The differences between images in a dataset are extracted and learned by a model as features. From Figure 5, it is clearly shown that the image data are distributed unevenly across different categories. When the training dataset is small, the limited features would hinder the performance of the model [12]. As a result, data augmentation was applied to the dataset. Data augmentation creates new data by applying techniques such as mirroring, random cropping, rotation, local warping, and adding

noise to the original data. Due to the limited data, data augmentation is crucial to improve the accuracy of the model because it can provide more data for the model to learn. With data augmentation, the model becomes more robust and has stronger generalization ability.

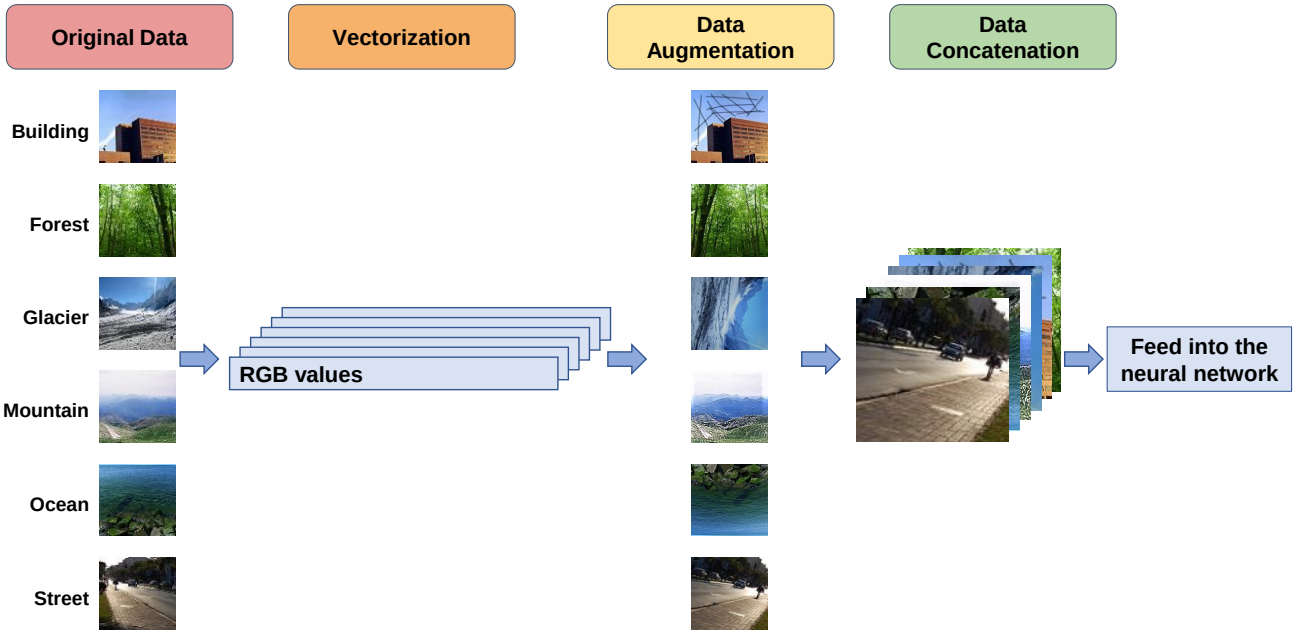

Figure 6. Data preprocessing for Inception based transfer learning and Xception based transfer learning

D. *Transfer Learning*

Transfer learning refers to the process of learning new tasks based on the knowledge learned from related tasks [13]. For example, knowledge about tree trunks may help in understanding forests. In the case of sufficient-source but limited-target domain data, transfer learning can significantly improve the performance of the model. According to Jason Yosinski, a machine learning researcher at Cornell, using weights transferred from a less relevant network is better than using random weights [14]. Further, even if substantial fine-tunings are performed on new tasks, initializing the network with transferred features can improve the generalization of the model.

E. *Methods*

This paper discusses two transfer learnings, one based on Xception and the other based on Inception-V3. In the experiment, models are created under each transfer learning to verify the transferability of the pre-trained models based on the ImageNet dataset. Two control factors play a role in the experiment: 1) whether pre-trained layers are trainable; 2) whether weights are transferred from a pre-trained model or randomized. Varying these two factors lead to the construction of four models (as shown in Table 1): the base model and three independent models.

More specifically, the base model is with the randomly initialized weights and untrainable layers. The first independent model is with randomly initialized weights and trainable layers. The second independent model is with pre-trained weights and untrainable weights. And finally, the third independent model, is with the pre-trained weights and trainable layers. Before outputting results, each model has to go through a fully connected layer with ReLU as the activation function, and a dropout layer. The process is repeated twice, then followed by another fully connected layer. Finally, the Softmax function is used as the uppermost layer. A schematic diagram can be found in Figure 7.

A ReLU function can be written as the following form:

$$R(\hat{z}) = max(0, \hat{z}) \tag{1}$$

A Softmax function can be expressed as follows:

$$\sigma(\hat{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2}$$

Model	Base Model	Independent Model 1	Independent Model 2	Independent Model 3
Weights	Random	Random	ImageNet	ImageNet
Trainable	False	True	False	True

Table 1. Inception based model and Xception based model with specified parameters

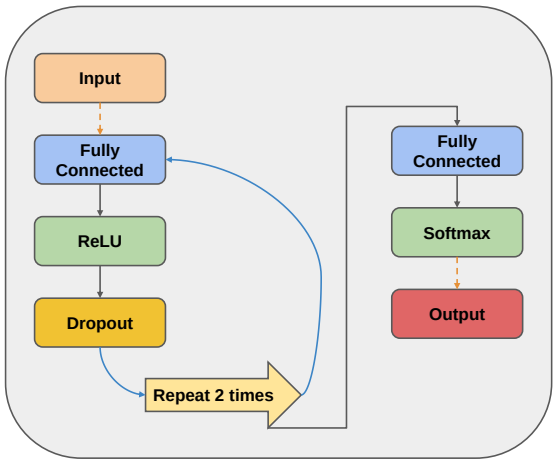

Figure 7. Exit flow for Xception based transfer learning and Inception based transfer learning

	Inception				Xception			
	Base	*Independent*			*Base*	*Independent*		
	Model	*Model 1*	*Model 2*	*Model 3*	*Model*	*Model 1*	*Model 2*	*Model 3*
Average Train Accuracy	56.03	62.74	90.2	**93.95**	69.04	81.36	87.21	**92.37**
Average Validation Accuracy	16.38	67.2	86.97	**91.81**	21.11	78.69	88.04	**91.20**

Table 2. Average accuracy (%) of models on Intel Image Classification Challenge dataset; values are obtained by finding average accuracy of 10 runs of each model with 3 epochs
Note: Best performance is in bold

Epoch	Inception-V3 Model 3				Xception Model 3			
	Average Acc		*Average Loss*		*Average Acc*		*Average Loss*	
	Train	*Val*	*Train*	*Val*	*Train*	*Val*	*Train*	*Val*
1/3	80.00	90.60	58.70	26.29	81.83	89.70	60.84	33.44
2/3	90.94	91.51	26.00	23.67	89.86	91.17	30.93	28.41
3/3	93.95	91.81	17.52	23.22	92.32	91.20	23.02	29.39

Table 3. Average accuracy and loss (%) of Model 3 on *Intel Image Classification Challenge* dataset; values are obtained by finding average values of 10 runs of each epoch

IV. Results and Discussion

The base model was expected to have the worst performance. The third independent model was expected to have the best performance. In addition, it was also expected that the Xception based transfer learning outperforms Inception based transfer learning.

From our results, it can be observed that the Xception based transfer learning has better overall performance than the Inception based transfer learning, with higher accuracy and less errors (illustrated in Table 2). However, it is notable that the Inception based transfer learning surprisingly achieved better performance than the Xception based transfer learning. Figures 8 and 9 show the detailed experimental results of both Model 3. The Inception based model has a more severe overfitting issue as shown in Table 2, the training accuracy of the Xception model is 1.12% higher than its validation accuracy, whereas that of the Inception model is 2.14%. Therefore, the performance of the Inception model is questionable, even with slightly higher validation accuracy.

Another factor worth-noting is that by allowing the pre-trained layers to be trainable, the performance of both models improve. Since fine-tuning does not necessarily improve the performance of the model, it can be seen that our Xception based transfer learning could generalize well on scene images.

Our initial test of using 10 epochs showed that most models converged as early as the third epoch. Therefore, we applied early stopping in the training process to use 3 epochs for each model. Table 3 shows the average training and validation accuracy of 10 runs of each Model 3. The average validation loss of both Model 3 reaches the minimum at the second epoch. The fact that both models converge at a quite early stage could be mainly explained by their structures as both models incorporate multiple fully connected layers.

Figure 8. Confusion matrix for Inception Model 3

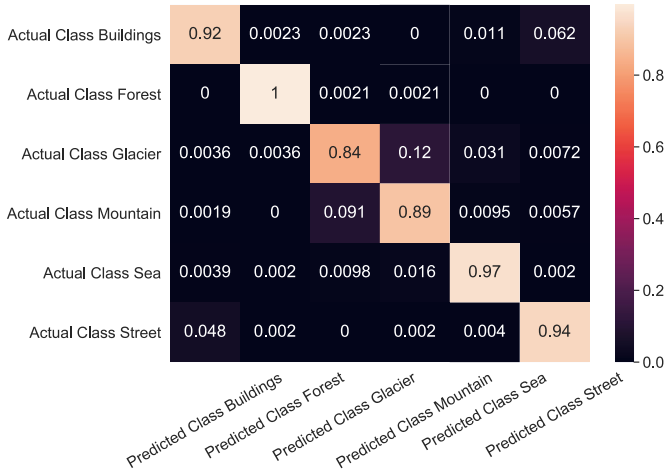

Figure 9. Confusion matrix for Xception Model 3

V. CONCLUSION

In this paper, a transfer learning on Xception based CNN for scene image classification is proposed. The performance of the model is conducted and evaluated on the Intel Image Classification Challenge dataset. The experimental results suggest that the model could achieve the highest training and validation accuracy on the scene classification task in comparison with the Inception-V3 when using trainable weights transferred from ImageNet. It is clear that Xception outperforms the comparison methods. Therefore, we can argue that transfer learning via Xception is good at processing scene image classification.

In the future, it will be interesting to test how the resolutions of an image affect the classification of scenes using the model. Furthermore, new transfer learning by adopting weights from the convolution layers of our model may produce important findings.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L. Li, K. Li, & F. Li, "ImageNet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248-255, 2009.

[2] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251-1258, 2017.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan,V. Vanhoucke, & A. Rabinovich, "Going deeper with convolutions ," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, & Z. Wojna, "Rethinking the Inception architecture for computer vision," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818-2826, 2016.

[5] C. Lin, L. Li, W. Luo, K. Wang, & J. Guo, "Transfer learning based traffic sign recognition using Inception-v3 model," *Periodica Polytechnica Transportation Engineering*, 2018.

[6] X. Xia, C. Xu, & B. Nan, "Inception-v3 for flower classification," *International Conference on Image, Vision and Computing*, pp. 783-787, 2017.

[7] E. S G, G. Prabhu J, A. Rishikesh, C. N A, & U. V, "Apparel classification using Convolutional Neural Networks," *International Conference on ICT in Business Industry & Government*, pp. 1-5. 2016.

[8] N. Hnoohom & S. Yuenyong, "Thai fast food image classification using deep learning," International ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering, pp. 116-119, 2018.

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

[10] V. Sze, Y. Chen, T. Yang, & J. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295-2329, 2017.

[11] P. Bansal (2019, January 30). Intel Image Classification. Retrieved December 11, 2020, from https://www.kaggle.com/puneet6060/intel-image-classification

[12] X. Li, W. Zhang, Q. Ding, & J. Sun, "Intelligent rotating machinery fault diagnosis based on deep learning using data augmentation," *Journal of Intelligent Manufacturing*, vol. 31, no. 2, pp. 433–452, 2020.

[13] L. Torrey & J. Shavlik, "Transfer learning," *Handbook of Research on Machine Learning Applications*, 2009.

[14] J. Yosinski, J. Clune, Y. Bengio, & H. Lipson, "How transferable are features in deep neural networks," *Advances in Neural Information Processing Systems*, vol. 27, pp. 3320-3328, 2014.