Systematic Configuration for Hyperparameters Optimization in Transferring of CNN Model to Disaster Events Classification from UAV Images

Supaporn Bunrit *, Nittaya Kerdprasop, and Kittisak Kerdprasop

School of Computer Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand; Email: nittaya@sut.ac.th (N.K.), kerdpras@sut.ac.th (K.K.)
*Correspondence: sbunrit@sut.ac.th (S.B.)

***Abstract*—Deep learning and computer vision-based approaches incorporated with the evolution of the relevant technologies of Unmanned Aerial Vehicles (UAVs) and drones have significantly motivated the advancements of disaster management applications. This research studied a classification method for disaster event identification from UAV images that is suitable for disaster monitoring. A Convolution Neural Network (CNN) of GoogleNet models that were pretrained from ImageNet and Place365 datasets was explored to find the appropriate one for fine-tuning to classify the disaster events. In order to get the optimal performance, a systematic configuration for searching the hyperparameters in fine-tuning the CNN model was proposed. The top three hyperparameters that affect the performance, which are the initial learning rate, the number of epochs, and the minibatch size, were systematically set and tuned for each configuration. The proposed approach consists of five stages, during which three types of trials were used to monitor different sets of the hyperparameters. The experimental result revealed that by applying the proposed approach the model performance can increase up to 5%. The optimal performance achieved was 98.77 percent accuracy. For UAV/drone applications, where a small on-board model is preferred, GoogleNet that is quite small in model size and has a good structure for further fine tuning is suitable to deploy.**

***Keywords*—hyperparameter optimization, transfer learning, fine-tuning, convolution neural network, deep learning, disaster events classification, Unmanned Aerial Vehicles (UAVs), Drones**

I. Introduction

Unmanned Aerial Vehicles (UAVs), also known as drones, have obtained significant consideration in different directions. Currently, drone systems with sufficient relevant technologies development could be employed in a diversity of applications, including security monitoring and surveillance, disaster management, remote sensing, search and rescue, construction and infrastructure inspection, precision agriculture, and many others. Over the last five years, there is evidence of a significant increase in the knowledge, awareness, and popularity of drone applications in disaster situations. Drones appeared to provide significant value in disaster management when compared to conventional methods in terms of cost and efficiency [1]. They reduce the time required to locate victims and the time required for subsequent intervention by searching a large area in a short period of time [2]. The capability of them that have visual sensor tools for acquiring aerial images or videos, increase their ability to provide detailed information about the surrounding environments. Such rich information can be processed to both offline and real-time applications. As a result, computer vision-based approaches incorporated with the evolution of the relevant technologies of UAVs/Drones have significantly motivated the advancements. However, the accuracy of the employed approaches depends on different factors, such as image resolution, capturing time, viewing angle, illumination, different structures of aerial images and reference data [3].

For disaster management applications, disaster event classification from the monitored images or videos is required. Monitoring different surrounding scenes is needed to be able to recognize whether any type of disaster is emerging, for example, fire, flood, or collapsed buildings. This presents a problem for the classification approach in computer vision and deep learning applications. Consequently, the prominent progression of the well-known scheme; the transfer learning of the pretrained Convolution Neural Network (CNN) model is investigated in this work. Since the data collection process is time-consuming and expensive, despite the existence of a dedicated dataset size [4], it is not sufficient to train the created CNN layers by starting from scratch; moreover, huge amounts of computer resources are required. To merit the best performance, the selected transfer learning scheme is suitable for analyzing in detail the fine-tuning process from the pertained selected models to our specific task. Most existing research works adopt the hyperparameter tuning of the pretrained CNN by referring to previous works. But hyperparameter tuning is both model and dataset-specific, thus it is

Manuscript received March 31, 2023; revised April 20, 2023; accepted May 30, 2023.

interesting to explore deeply the effects of each choice in hyperparameter tuning.

The proposed work aimed at looking for a suitable pretrained model to gain the best performance for the studied dataset. In this research, GoogleNet [5] pretrained models from the source task of ImageNet [6] and Place365 [7] datasets were all explored as the fine-tuning method of transfer learning. To search for the optimal performance from the fine-tuning, the process of hyperparameters optimization (or tuning) is investigated and proposed as a systematic approach. The systematic configuration has revealed many interesting byproducts from our empirical study. The contributions of this work are as follows:

(1) Explore a classification method for disaster events classification that is suitable for further study in disaster monitoring application.
(2) Obtain a network that is pretrained from different source datasets to select the superior one. We explore GoogleNet models that are pretrained from ImageNet and Place365 datasets to find the appropriate one for fine-tuning the disaster events dataset.
(3) Propose a systematic configuration in searching for the optimal hyperparameter by a fine-tuning scheme based on empirical study.

II. Literature Reviews

A. Diaster Event Classification from UAV/Drone Data

A disaster is an event caused by a natural or man-made hazard that occurs over a short or extended period. It results in significant physical harm or destruction, as well as mortality or a significant alteration in the environment [8]. Recently, UAVs/Drones have been increasingly popular in gathering images or videos since they are faster and more accurate than satellite imagery and allow for more prompt assessment. In the last decade, disaster events classification, identification, or disaster monitoring embedded by machine learning and deep learning techniques have been most frequently employed. Most previous works focus on the disaster from only one event such as fire or flood. Forest fire monitoring from aerial images using deep learning techniques was studied by Kim *et al.* [9], where transfer learning also applied in wildfire identification [10]. Flooding observed from UAV images is explored as flood detection method by Munawar *et al.* [11]. Recently, Kyrkou and Theocharides [12] proposed an emergency response application using UAV images to monitor and classify four disaster events using deep learning.

B. CNN Pretrained Models

Since the success of AlexNet [13] in the 2012 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC), deep learning networks based on CNN have been tentatively expanded as a major approach for a variety of applications. Year by year, deeper and/or more efficient architectures have been proposed and published as the pretrained models. It is known in the community that fully trains the CNN network to a specific task needs numerous computer resources and takes time. Most importantly, dataset size affects the performance. For these reasons, transfer of learning approach has been increasingly studied and many well-known pre-trained architectures are made public. Fig. 1 represents published models from ImageNet source task dataset that were studied by Bianco *et al.* [14]. The research reported by comparing the performance in terms of top-5 accuracy to model size and number of operations. Many works [15–17] have been selected from some of these models to further explore and apply to their applications. For GoogleNet, the selected model used in our study is at the location marked by red rectangle in Fig. 1. Its model size is quite small compared to others. Although its performance is not the best, it is a good structure for further fine tuning. Moreover, model size is a concern in UAV/drone applications, where a small on-board model is preferred.

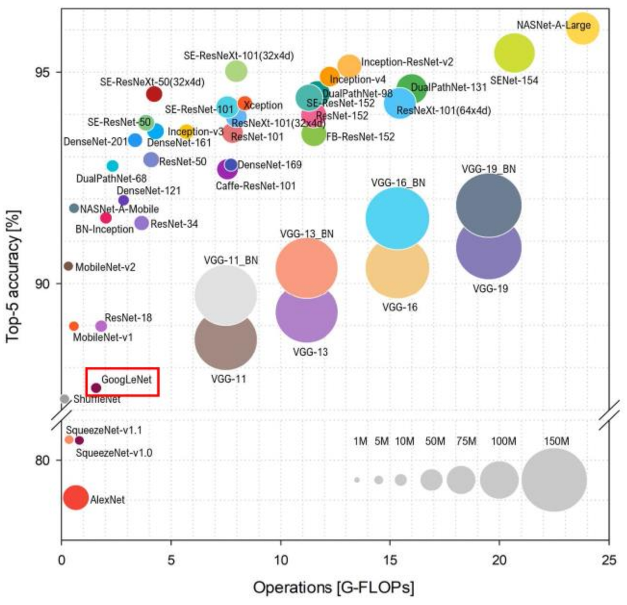


Figure 1. Pretrained Models compared in terms of top-5 accuracy to model size and number of operations [14].

C. GoogleNet (InceptionV1)

GoogleNet or InceptionV1 was proposed by Szegedy *et al.* [5]. The architecture consisted of 22 weighted layers. It was proposed under an improvement of the calculation resources from previous works. The efficiency of the network came from wider and deeper layers by incorporating nine modules of inception module on some parts of the network. Only small filter sizes of 1×1, 3×3, and 5×5 was used in the module. Each block in a module can be parallel and the results from all blocks are concatenated to the output and send to the next module or next layer. It has auxiliary classifiers that are connected on top of the output of inception modules. Each auxiliary classifier has a 5×5 average pooling layer, a 1×1 convolutional layer, two fully connected layers, and a softmax layer. Detailed layers of GoogleNet can be referred to in [5].

In this research, we explore GoogleNet pretrained model from both source datasets which are ImageNet and Place365.

D. Transfer Learning and Fine-Tuning

The term transfer learning means the transferring of knowledge (in terms of weight and bias values) from some of the pretrained architectures trained by some large dataset (source task). Such knowledge from pretrained model is transferred to the task specific dataset. Transfer learning can apply by two schemes namely fixed feature extractor and fine-tuning. Fixed feature extractor directly uses the pretrained weights and biases transferred to a task specific with no need to retrain the network. The opposite is of the fine-tuning, whereby the network must be retrained on some parts of a network using a task specific dataset with weights and biases initialized from pretrained values. By the transfer learning method, it can help us create a high-performance model in a timesaving way rather than training a model from scratch.

E. Hyperparameter in CNN Models and Tuning Methods

There are two types of parameters that exist in neural networks-based algorithm. These are model parameters and hyperparameters. Model parameters are weights and biases that can be initialized and updated through the data learning process. Where hyperparameters cannot be directly estimated from data learning and must be set before training a model. They are used to either configure a model or to specify the algorithm used to minimize the loss function [18]. Therefore, these hyperparameters are needed to be fixed before running or training a model. As a result, methods for tuning or searching require the set of hyperparameters, in addition, their ranges need to be considered [19]. Based on the knowledge [18, 19] and experience, the top three hyperparameters affected the performance of fine-tuning CNN model are the initial learning rate, the number of epochs, and the minibatch size.

The initial learning rate is the size of the step to determine how fast or slowly the algorithm descends the error curve. The too large or too small value can cause a model to never descend or to converge too slowly. It is a small positive value, often in the range between 0.0 and 1.0 [18]. Traditionally, neural networks are trained using the stochastic gradient descent optimization algorithm. Then, the error gradient is used to update the model weights, and the process is repeated. Therefore, one training epoch means the algorithm has made one pass through the entire training data. For a model being trained from scratch, hundreds or thousands of epochs may be needed, but in fine-tuning a pretrained model, only 5 to 100 are always suitable. The number of epochs that are too high always lead to overfitting. The batch size will define how often to update weights and biases, the parameters of a model. In the case of stochastic gradient descent, a group of samples called a “minibatch” is used in a single iteration. The minibatch size is a fixed number of training samples that is less than the entire dataset size. Thereby, in each iteration, the network is trained on a different group of samples until all samples in the dataset are used. The minibatch size is typically between 1 and a few hundred, where a value equal to 32 is a good default value [19]. In modern machine supported parallel tasks with Graphics Processing Unit (GPU) included, the maximum minibatch size will fix by the GPU specification.

The performance of most machine learning algorithms depends on their hyperparameter settings [20, 21], including the fine tuning of the pretrained CNN model. To the best of our knowledge, by not using software packages or tools for optimization algorithms, none of the systematic procedures in the search for optimal hyperparameters is proposed as an academic empirical study. Only a small number of scenarios in the deep learning or machine learning communities were discussed and suggested to the researchers [18, 19].

III. Materials and Methods

A. Dataset

This research uses the dedicated Aerial Image Dataset for Emergency Response Applications (AIDER) dataset [4], which is the same dataset used in EmergencyNet [22]. The dataset construction involved manually collecting all images for four disaster events, namely Fire/Smoke, Flood, Collapsed Building/Rubble, and Traffic Accidents, as well as one class for the Normal case. These aerial images for the disaster events were collected through various online sources. During the data collection process, the various disaster events were captured with different resolutions and under various condition with regards to illumination and viewpoint. Finally, to replicate real world scenarios the dataset is imbalanced in the sense that it contains more images from the Normal class. Table I shows summary of our studied dataset that consists of five classes of which there are four for disaster events and one normal class. In the case of the normal class, we did not use all the images from the original dataset, only 1200 was enough and suitable. The dataset is divided into train, validate, and test set in the ratio of 70%, 10%, and 20%, respectively.

TABLE I. Studied Dataset from AIDER [4] (Train: 70%, Validate: 10%, Test: 20%)

Class	Number of Samples			
	Train	Validate	Test	Total (Per Class)
Collapsed Building	358	51	102	511
Fire/Smoke	365	52	104	521
Flood	368	53	105	526
Normal	840	120	240	1200
Traffic Accident	340	49	97	486
Total (Per Set)	2271	325	648	3244 (All Class)

B. Fine-Tuning Process

Procedures involve a fine-tuning using GoogleNet Pretrained model which is depicted in Fig. 2. The process mainly consists of six consecutive procedures as follows.

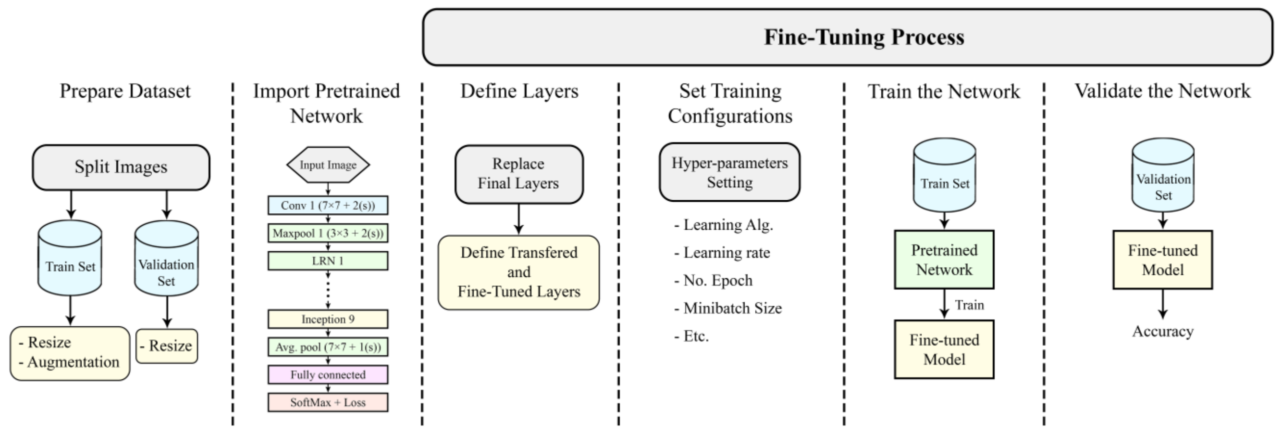


Figure 2. Fine-tuning process of CNN pretrained network.

1) Prepare dataset

In this research, 80% of the studied dataset out of the 20% for testing is split into 70% for train set and 10% for validate set. Each image from both train and validate set is resized to be the same size and used as the input image of the pretrained model. To prevent from overfitting in the fine tuning/training, augmentation techniques are employed to the train set.

2) Import pretrained network

GoogleNet pretrained networks from both ImageNet and Place365 source datasets are imported to Matlab. In this research, we conduct the empirical experiment with Matlab R2021a. Computer laptop of 12th Gen Intel(R) Core (TM) i7-12700H with 2.30 GHz processor is used. The installed RAM is 16 GB with 64-bit Windows 11 and NVIDIA GeForce RTX 3070 Laptop GPU is employed.

3) Define layers

The fine-tuning process starts from replacing the final layers of the pretrained network according to the numbers of class of a task specific dataset. As a result, we replaced the classification layer for five classes instead of 1000 classes and 365 classes for ImageNet and Place365 pretrained model. Then, the transferred (frozen) layers and fine-tuned (train) layers are defined. To focus on the dataset-specific features, we force most of the lower layers of GoogleNet as the transferring and just some higher layers are fine-tuning.

4) Set training configurations

Before training or fine-tuning, the training configurations are needed to be set. This means some hyperparameters are needed to be specified. In this research, we investigated on the Stochastic Gradient Descent with Momentum (SGDM) learning algorithm with the default momentum value of 0.9. Then, the procedure for systematic configurations in searching for the optimal values of the top three hyperparameters most affecting the performance is proposed based on an empirical study that will be explained in the next sub-section. The top three hyperparameters are the initial learning rate, the number of epochs, and the minibatch size.

5) Train the network

After setting all hyperparameters in the previous step, the selected pretrained model is then trained further (fine-tuned) with the training set of the studied dataset. Therefore, as training, the model is gradually fine-tuning the parameters of a network by using disaster event images according to the setting of the hyperparameters.

6) Validate the network

To compare the results from many cases of hyperparameters setting, when each training configuration is finished, the fine-tuned model is then validated with the validation set. In this study, we use accuracy as a measurement merit. After that, the test set is applied to the fine-tuned model to compare the performance of the model for each configuration.

C. Proposed Systematic Configuration of Hyperparameter Tuning

The systematic configuration approach is proposed to optimize the hyperparameters. Each trial in the fine-tuning process is defined as one time to fine-tune or train a model according to a configuration to be explored. Such a configuration consists of three hyperparameters setting values to be optimized: the initial learning rate, the number of epochs, and the minibatch size. All stages of the proposed approach are applied to the relevant models of each source dataset separately (ImageNet and Place365). Three types of trials used for the empirical experiments are defined as follows:

(1) Primary/Exploration trial: 5 trials for each configuration
(2) Secondary trial: 10 trials for each configuration
(3) Confirmatory trial: 30 trials for each configuration

Stage 1*: Monitoring focus on the initial learning rate.*

Using the primary trial to first inquire the effects of the initial learning rate (by fixing the minibatch size to be 32).
Response: The best two initial learning rates.

- Define the initial learning rate as 0.01, 0.001, 0.0001, 0.00001, and 0.000001.
- Define the number of epochs as 10, 20, and 30.
- Set all combination configurations from the defined values and name them as A01, A02, …

- Select the best two initial learning rate for exploring the next stage from the test set result.

***Stage 2**: Monitoring focus on the number of epochs.*

Using the primary trial to inquire the effect of number of epochs by fixing each of the best two initial learning rates responded in Stage 1.

Response: The best three configurations and conclusion about the effect of the number of epochs for each fixed initial learning rate.

- Fix each initial learning rate from the best two responded from Stage 1.
- Define the number of epochs as 5, 15, 25, 35, 40, 60, 80, and 100.
- Set all combination configurations from the defined values and name them as B01, B02, …
- Compare the test results and record the best three configurations (consider including configurations A from Stage 1).
- Analyze the effect of the number of epochs in response to the overall accuracy.

***Stage 3**: Monitoring focus on the effect of minibatch size (MB) to each of the best two initial learning rates.*

Use exploration trial to inquire the effect of minibatch size where only the best three configurations (by using maximum accuracy in an exploration trial) from Stage 2 are further explored.

Response: To decide which value of initial learning rate should not be considered for next stage.

- Define minibatch size as 16, and 64 for each of the best three configurations from Stage 2 and define its initial learning rate as another one from the best two selected. The number of epochs is fixed the same as at Stage 2.
- Set all combination configurations from the defined values and name them as C01, C02, …
- Analyze the effect of different minibatch size and eliminate one initial learning rate.

***Stage 4**: Monitoring the effect of all minibatch sizes.*

Use secondary trial to inquire the effect of all minibatch sizes (which are 8, 16, 32 and 64) to the selected configurations from Stage 3.

Response: The best configuration (by considering maximum accuracy in a secondary trial).

***Stage 5**: Confirm of the best configuration.*

Use Confirmatory trial to search for the maximum performance of the best configuration selected from Stage 4. Compare maximum performance from this stage with the previous stage and chose the best one.

Response: The maximum performance.

IV. Experimental Resultss and Discussion

A. Experimental Result

When the proposed systematic configuration method is applied in the fine-tuning process, the accuracy results from the test set are employed to compare the performance. Fig. 3 shows the mean values of the accuracy for each primary trial of all combination configurations in Stage 1, where E10, E20, and E30 are the number of epochs that were used as 10, 20, and 30, respectively. Each pair of bar graphs also compares between pretrained source datasets from ImageNet and Place365. It can be clearly inferred from the figure that the initial learning rates of 0.001 and 0.0001 are the best two values that will be further explored in the next stage.

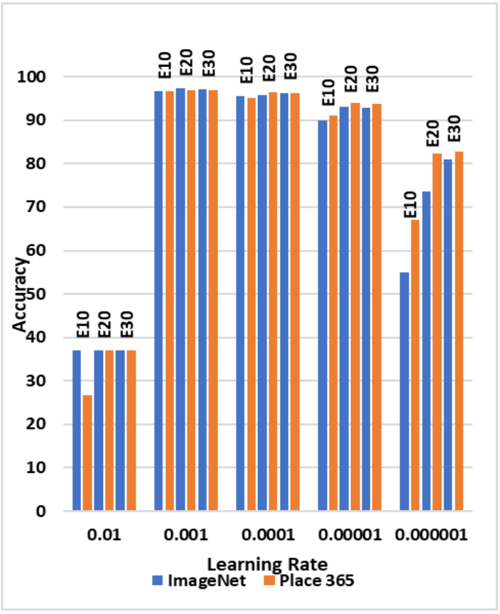


Figure 3. Mean of accuracy from each primary trial in Stage 1.

Table II shows the result from each primary trial in Stage 2 when inquiring about the effect of the number of epochs. The mean values of the accuracy from all configurations are presented, and the three best configurations are marked as the maximum three values in the Max column. The result from the pretrained source datasets of ImageNet and Place365 is demonstrated separately. In the case of ImageNet, the mean values of both Learning Rates (LR) are quite stable since the number of epochs equals 30, whereas Place365 is stable since the number of epochs equals 40. Overall, the accuracy from LR at 0.0001 is less than LR at 0.001, around 1% from ImageNet, and around 0.5% from Place365. As shown in the Max column, the best three configurations from both sources are those with LR equal to 0.001. Such three configurations of each source will be further explored in Stage 3.

TABLE II. Mean and Max Accuracy from Stage 2 When the Best Three Configurations are Selected (MB = 32)

	ImageNet			**Place365**		
No. Epochs	LR=0.001		LR=0.0001	LR=0.001		LR=0.0001
	Mean	Max	Mean	Mean	Max	Mean
100	97.41		96.45	97.63		97.24
80	97.50		96.60	97.45		97.17
60	97.59	98.30	96.98	97.84	98.30	97.22
40	97.92		96.94	97.57	97.99	97.07
35	97.07		96.91	96.63		96.08
30	97.14		96.30	96.94		96.30
25	96.91	98.46	96.45	96.18		95.97
20	97.28		95.86	96.82	98.19	96.45
15	96.45	98.15	95.93	96.73		96.32
10	96.70		95.56	96.67		95.15
5	96.51		94.10	95.52		93.82

To decide which value of the initial learning rate should be discarded, we must concentrate on the effect between the selected learning rates and the minibatch sizes in Stage 3. Table III shows the comparison results when the minibatch sizes of 16 and 64 of the three best configurations from Stage 2 are further explored with LR equal to 0.0001. The table report in terms of the maximum accuracy in a trail and the values when minibatch size equals 32 from Stage 2 are also compared in a table. From the result, it can be concluded that with different minibatch sizes of 16 and 64, when LR equals 0.0001, the performance is still less than when LR equals 0.001 for both source datasets. Therefore, we can eliminate the value of 0.0001 when exploring in Stage 4.

TABLE III. MAXIMUM ACCURACY FROM EACH TRIAL IN STAGE 3 WITH DIFFERENCE MINIBATCH SIZE (LR = 0.0001 FOR MB=16 AND 64)

ImageNet				**Place365**			
No. Epochs	Minibatch Size			No. Epochs	Minibatch Size		
	16	(32)	64		16	(32)	64
60	97.38	98.30	97.46	60	97.52	98.30	97.66
25	96.95	98.46	97.19	40	97.17	97.99	96.84
15	96.45	98.15	96.78	20	96.98	98.19	96.22

When Stage 4 is applied, the three best configurations from Table III. are further explored in order to select the best configuration. In this stage, all defined minibatch sizes are explored using the secondary trial. For the ImageNet source model, we acquired the maximum performance, which equals 98.77, from two configurations: MB equals 8 of 20 epochs and MB equals 32 of 25 epochs. We selected a configuration in which MB equals 32, which can be confirmed by the confirmatory trail in Stage 5. The response for maximum accuracy from 30 trials is still equal to 98.77. In the case of the Place365 source model, the same best performance was achieved, which equals 98.77 in the configurations where MB equals 64 of 60 epochs.

Figs. 4 and 5 depict the confusion matrix from the test set result, of which the maximum performance was 98.77 from both ImageNet and Place365 source models. Only 8 images out of 648 are incorrectly classified from both models. For ImageNet source model in Fig. 4, two images of Collapsed Building class are classified as Fire class and another two are classified as Traffic Incident class. Whereas all images in Fire class are correctly classified. For Place365 in Fig. 5, the model predicts one of each misclassified Collapsed Building class to be one of the other classes and all images in Normal class are correctly classified.

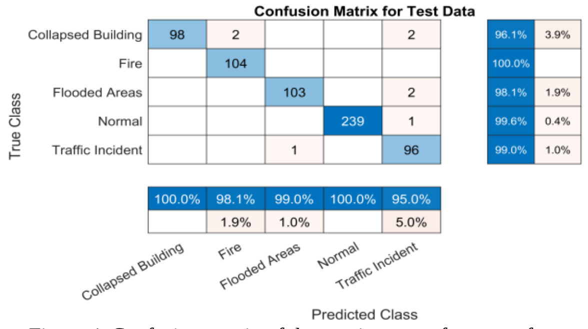


Figure 4. Confusion matrix of the maximum performance from ImageNet source model.

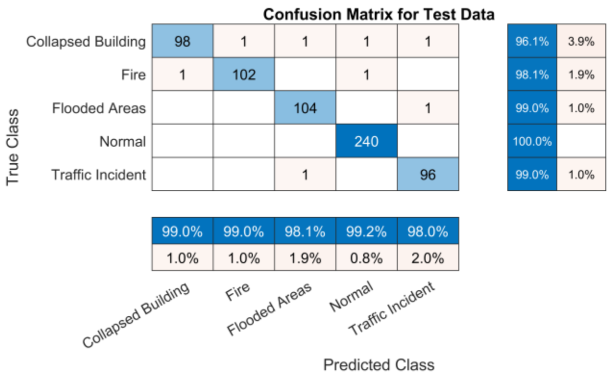


Figure 5. Confusion matrix of the maximum performance from Place365 source model.

Fig. 6 shows some images from the true class of Collapsed Building that incorrectly classified to be the other class. The top row are the images that classified to be a class of Traffic Incident (left) and a class of Fire (right) from ImageNet pretrained source. Whereas the bottom row are the images that classified to be a class of Traffic Incident (left) and a class of Fire (right) from Place365 pretrained source.

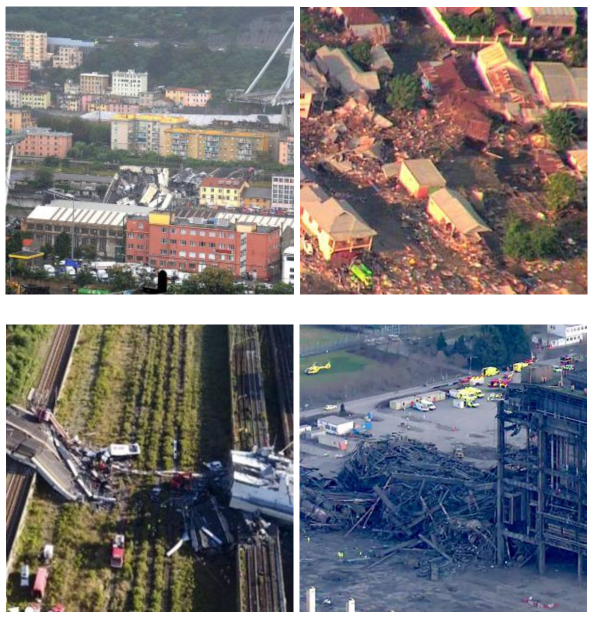


Figure 6. Example of images in the true class of Collapsed Building that are incorrectly classified.

Fig. 7 represents the images from the class of Flood Areas and Traffic Incident that incorrectly classified to be the opposite class. The top row are the images that classified from a class of Flood Areas to be Traffic Incident (left) and from a class of Traffic Incident to be Flood Areas (right) from ImageNet pretrained source. Whereas the bottom row are another two images that are classified in the same way from Place365 pretrained source. Overall, from both models, the classes of Collapsed Building, Flood Areas and Traffic Incident are the most ambiguous among the classes and are always incorrectly classified.

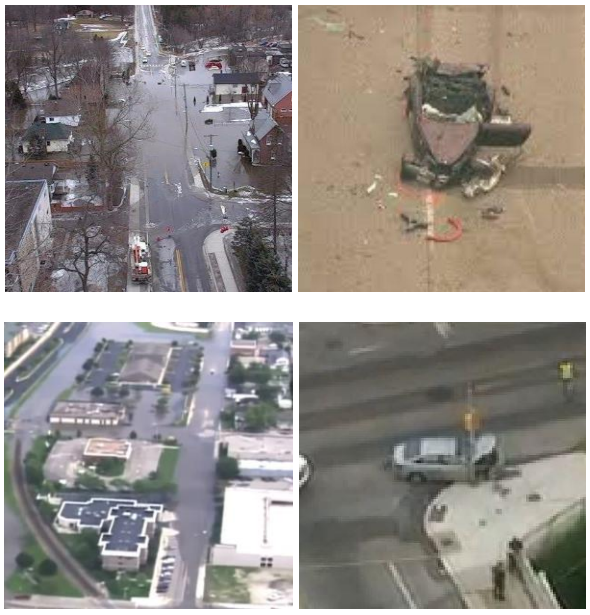
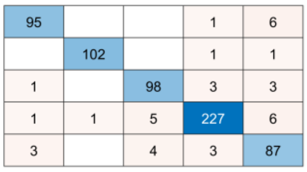


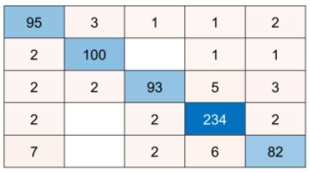
Figure 7. Example of images in the true class of Flood Areas and Traffic Incident that are incorrectly classified.

Fig. 8 shows the confusion matrix of the worst results (class orders are the same as in Fig. 5) from the confirmatory trial in Stage 5 when 30 trails are employed for the best selected configuration. Fig. 8(a) is the performance obtained from ImageNet which equal to 93.98 and Fig. 8(b) is the performance obtained from Place365 which equal to 93.21. It can be inferred from the results that the maximum performance yield from the proposed method can increase the accuracy around 5%.



95			1	6
	102		1	1
1		98	3	3
1	1	5	227	6
3		4	3	87

(a) ImageNet



95	3	1	1	2
2	100		1	1
2	2	93	5	3
2		2	234	2
7		2	6	82

(b) Place365

Figure 8. Confusion matrix of the worst results from the confirmatory trial of both source models.

B. *Discussion*

Based on what is well-known in the deep learning community about the fine-tuning that the initial learning rate is the most effected hyperparameter to the performance of a model and the best default value of the minibatch size is 32, the proposed method, therefore, at Stage 1 focus on monitoring the ranges of the learning rate by fixing the minibatch size to be 32. The method gradually explores by using three types of trials to eliminate the insignificant values of each hyperparameter and select the best one. The initial learning rate is the first optimal hyperparameter that can be selected in Stage 3. The number of epochs is the second selection, and the last is the minibatch size, which can be decided in Stage 3 and Stage 4, where Stage 5 is used to confirm the best selected set of hyperparameters.

For the selected GoogleNet model, the source pretrained dataset has no effect on the studied disaster events dataset. It can be inferred that both source datasets are similar in image properties and related to the disaster events dataset. As a result, the transfer learning scheme that employed in this research is suitable for disaster event classification. In terms of model complexity and performance compared to the other pretrained networks, including our experimental results, the architecture of GoogleNet-based layers is suitable and interesting for further use in disaster monitoring applications where a small on-board model is needed. For the model operated on-board the UAV, environmental concerns in terms of low power embedded in platforms with a minimum memory requirement and sufficient performance to run in real-time are all that need to be addressed.

The optimal hyperparameters determined from the proposed systematic configurations can be studied further by confirming the results with some tools or software packages for hyperparameter optimization methods.

V. Conclusion

This research proposed a transfer learning approach for classifying disaster events from UAV/Drone images. Both ImageNet and Place365 are explored as sources of pretrained GoogleNet. It can be concluded that these different source datasets are similar in image properties, related to the studied dataset, and suitable to be applied as the fine-tuning method for disaster events classification. The search for the optimal hyperparameters is proposed as a systematic configuration procedure. The accuracy result from an empirical study of the proposed systematic approach revealed an optimal performance of 98.77. The obtained results were higher than a compare-based performance of up to 5%.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Supaporn Bunrit conducted the research, analyzed the data, and wrote the paper; Nittaya Kerdprasop and Kittisak Kerdprasop guide for the direction of analyzing the data; all authors had approved the final version.

References

[1] S. M. S. M. Daud, M. Y. P. M. Yusof, C. C. Heo, L. S. Khoo, M. K. C. Singh, M. S. Mahmood, and H. Nawawi, "Applications of drone in disaster management: A scoping review," *Sci. Justice*, vol. 62, no. 1, pp. 30–42, Nov. 2021.

[2] C. V. Tilburg, "First report of using portable Unmanned Aircraft Systems (Drones) for search and rescue," *Wilderness Environ Med*, vol. 28, no. 2, pp. 116–118, June 2017.

[3] A. Al-Kaff, D. Martín, F. García, A. de la Escalera, and J. M. Armingol, "Survey of computer vision algorithms and applications for unmanned aerial vehicles," *Expert Syst. Appl*, vol. 92, pp. 447–463, Feb. 2018.

[4] AIDER (Aerial Image Dataset for Emergency Response Applications). [Online]. Available: https://zenodo.org/record/3888300#.Ys_0h3ZByUk

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[6] ImageNet Large Scale Visual Recognition Challenge (ILSVRC). [Online]. Available: https://www.image-net.org/challenges/LSVRC/index.php

[7] Places. [Online]. Available: http://places2.csail.mit.edu/

[8] M. Sivakumar and N. M. TYJ, "A literature survey of unmanned aerial vehicle usage for civil applications," *J. Aerosp. Technol. Manag*, vol. 13, 2021.

[9] S. Kim, W. Lee, Y. S. Park, H. W. Lee, and Y. T. Lee, "Forest fire monitoring system based on aerial image," in *Proc. 2016 3rd International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*, Dec 2016, pp. 1–6.

[10] H. Wu, H. Li, A. Shamsoshoara, A. Razi, and F. Afghah, "Transfer learning for wildfire identification in UAV imagery," in *Proc. 2020 54th Annual Conference on Information Sciences and Systems (CISS)*, 2020, pp. 1–6.

[11] H. S. Munawar, F. Ullah, S. Qayyum, S. I. Khan, and M. Mojtahedi, "Uavs in disaster management: Application of integrated aerial imagery and convolutional neural network for flood detection," *Sustainability*, vol. 13 no. 14, p. 7547, 2021

[12] C. Kyrkou and T. Theocharides, "Deep-learning-based aerial image classification for emergency response applications using unmanned aerial vehicles," in *Proc. the CVPR Workshops*, Long Beach, CA, USA, 16–20 June 2019, pp. 517–525.

[13] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS 2012*, 2012, pp. 1106–1114.

[14] S. Bianco, R. Cadene, L. Celona and P. Napoletano, "Benchmark analysis of representative deep neural network Architectures," *IEEE Access*, vol. 6, no. 1, pp. 64270–64277, 2018.

[15] J. Jennifer, J. Krislynd, S. Aprianto, and D. Suhartono "A comparative study of various convolutional neural network architectures for eye tracking system," *JOIG*, vol. 10, no. 4, Dec. 2022.

[16] N. Y. R. Wang and N. Li, "A novel active object detection network based on historical scenes and movements," *Int. J. Comput. Sci. Eng.*, vol. 13, no. 3, pp. 79–83, 2021.

[17] P. H. Kashika and B. V. Rekha, "Deep learning technique for object detection from panoramic video frames," *Int. J. Comput. Sci. Eng.*, vol. 14, no. 1, pp. 20–26, 2022.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016, ch. 11.

[19] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," *Neural Networks*, pp. 8–11, 2012.

[20] X. X. H. Li, and F. Hu, "The flocs target detection algorithm based on the three frame difference and enhanced method of the Otsu," *Int. J. Comput. Sci. ng.*, vol. 7, no. 3, pp. 197–200, 2015.

[21] X. Lui and C. Wang, "An empirical study on hyperparameters optimization for fine-tuning pre-trained language models," in *Proc. 11th Int. Joint Conf. on NLP*, Aug. 2021, vol. 1, pp. 2286–2300.

[22] C. Kyrkou and T. Theocharides, "EmergencyNet: Efficient aerial image classification for drone-based emergency monitoring using atrous convolutional feature fusion," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens*, vol. 13, pp. 1687–1699, 2020.

Copyright © 2023 by the authors. This is an open access article distributed under the Creative Commons Attribution License (CC BY-NC-ND 4.0), which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.