Image Recognition Tasks by the VGG 16 Model
– Classification Between Cat and Dog –

Name: Ran Arino

Student ID: 153073200

Email: rarino@myseneca.ca

Course: Machine Learning

Course ID: BDA500NAA.05380.2237

Professor: Dr. Amir Moslemi

Abstract

This paper explores the application of the VGG16 model, a pre-eminent convolutional neural network (CNN), for the classification of images of dogs and cats. The original data is a MATLAB-based dataset; during the preprocessing phases, the data is normalized, resized, and converted to the RGB scale to suit the VGG16 input conditions. Employing transfer learning techniques, the VGG16 model was adapted to the dataset with its convolutional base pre-trained on ImageNet.

The methodology of this project encompasses data splitting for training and validation sets, model configuration with Keras, and evaluation using metrics like accuracy, precision, recall, F1-score, and the Receiver Operating Characteristic (ROC) curve. Model training was monitored for overfitting, with hyperparameters and regularization techniques adjusted accordingly.

Results demonstrated the VGG16's robust classification capabilities with high precision (0.93-1.00) and recall (0.92-1.00) across classes, an F1-score of 0.96, and an AUC of 1.00, indicating excellent recognition of images. The project also focused on potential overfitting, as evidenced by fluctuations in validation loss and divergence from training loss, emphasizing the need for further research on hyperparameter tuning and regularization techniques.

The findings confirm that the VGG16 model is a potent tool for image classification tasks, offering high accuracy and generalization with reduced computational demand. The project contributes to the understanding of CNNs in image recognition and proposes directions for further enhancing model performance in practical applications.

Introduction

The Virtual Geometry Group 16 (VGG16) model is known as one of the classic deep learning approaches on convolutional neural network (Gu et al., 2021). The VGG16 models are widely utilized for various fields of image classification tasks; degree of ripeness in fruits (Hermana et al., 2021), leaf species (Gu, et al., 2021), and malware analysis in cybersecurity (Shah et al., 2022). Therefore, the VGG16 is the generalized pre-trained model based on the large and diverse dataset. In the VGG16 model, Convolutional Neural Networks (CNNs), Multi-Layer Perceptron (MLP), and Transfer Learning (TL) have pivotal roles; they are also critical in broad machine learning and deep learning techniques, especially in image classification and recognition.

Firstly, CNNs are widely utilized deep learning techniques in image recognition and classification tasks. As Simonyan and Zisserman (2015) implemented, the CNNs generally have three components; convolutional layers, pooling layers, and fully connected layers. More specifically, as Defferrard et al (2017) argued that “CNNs extract the local stationarity property of the input data or signals by revealing local features that are shared across the data domain”. In

other words, since CNNs define the size of scopes in advance and slide this area over the entire image, its network can specify the common points among all datasets, which enables the machine to understand those common features regardless of where they are located or how different the background is. In the case of the VGG16 model, the size of scopes – "each small receptive field" - is set 3 by 3 in each convolutional layer (Simonyan & Zisserman, 2015).

Secondly, the MLP is one of the feed-forward neural network methods (Gardner, 1988 & Valls et al., 2021). In the MLP architecture, one or more hidden layers are implemented in between a single input and output layer, also each node passes to every node in the subsequent layer after being processed with applied weights (Valls et al., 2021). After these dot product operations between corresponding nodes and weights, an activation function would be applied to those results. In this phase, the following non-linear transportations are typically employed as its activation function; sigmoid, tanh, and relu (Valls et al., 2021). Since the hidden layers conduct those processes, the neural network can capture complex patterns in the data. Considering that three fully connected layers are implemented after the convolutional and pooling blocks (Simonyan & Zisserman, 2015), the MLPs are incorporated into those last parts of the VGG16 model.

Lastly, the TL is the machine learning technique that reuses the knowledge acquired in the training phase, such as weights and biases, in order to predict or classify new targets (Hermana et al., 2021). Thus, TL can reduce the expensive time for recollecting the data and rebuilding the model, although most machine learning and statistical models are necessary to recreate the models when the newly added data is retrieved from different distributions (Pan & Yang, 2010). In other words, TL could help to create a more generalized model, which enables the computer to handle new tasks, based on its current knowledge. Considering that the VGG 16 model from Keras package provides the pre-trained on the large dataset of ImageNet (Akther et al., 2021), so the VGG16 is one of the models led by TL and can applied to the different image recognition tasks.

Methodologies

(1) Dataset: CatDog.mat

In this project, the original dataset is the binary MATLAB file about the images of cats and dogs. The original data, dictionary structure, is composed of for 4 pairs of keys and values;

1. Key: "X", Value: a nested list of floats with dimensions of (4096, 242); each column-wise vector shows each image information.
2. Key: "G", Value: a list of binary data: the label 0 shows the "cats", the label 1 shows the "dogs".
3. Keys: "nx" and "ny", Value: both values show the size of images as integer; they correspond to the width and height of each image, respectively.

(2) Preprocessing

In this phase, the feature matrix "X" would be modified by following data preprocessing tasks show below (*Figure 1* shows images of a cat and a dog after those processes);

1. Transposing the feature matrix "X": the original dimensions changed to (242, 4096), which means that each row-wise vector has all features of each image.
2. Normalizing "X": applying the minimum and maximum normalization to the entire data and changing all values into the range of [0., 1.].
3. Reshaping "X": based on the image sizes from the values of "nx" and "ny", the feature matrix will be changed to the multidimensional shape of (242, 64, 64, 1), which indicates there are 242 images in total, the pixel size of each image is 64 by 64, and all images are gray scale.
4. Resizing "X": each image size should be 224 by 224 pixel to match the input size with the VGG16 model.
5. Converting to the RGB scale: In the VGG16, the images are processed by the RGB scale, so the matrix "X" will be reshaped to the three-channel format – dimensions of (242, 224, 224, 3).
6. Transpose the images: Swapping the height and width dimensions to rotate images by 90 degrees clockwise.

(3) Train and Test data Split

In this case, the preprocessed data will be split into the training and test sets while maintaining the ratio of the original target data. Therefore, the number of data whose target labels are 0 and 1 is roughly equivalent in both the training and test sets.

(4) Model Creation: VGG16

Aforementioned, the VGG16 model is applied to the dataset in this project, and all the layers in this model are structured by Keras from the Tensorflow library. The detailed CNN configurations are mainly based on the research from Simonyan and Zisserman (2015);

- Input is the 224 by 224 RGB image; the image size matches the input size of the pre-trained VGG16 model.
- There are five convolutional blocks; each block has two or three convolutional layers in 2D pattern – the number of filters in these layers doubles with each subsequent block, starting from 64 in the first block and increasing to a maximum of 512 in the final two blocks – and one max pooling operation for 2D spatial data.
- The parameters of all convolutional layers are frozen. According to Shah et al (2022), this process allows the training time to be significantly reduced.
- After those convolutional blocks, three fully connected layers are defined. The first and second layers have 128 and 64 units, respectively, and ReLU(Rectified Linear Unit) is implemented as activation function in both layers; the last layer has a single unit and sigmoid activation function for the binary classification.

- The model is complied with the Adam optimizer, binary cross entropy as the loss function, and accuracy as the evaluation metric.

(5) Model Training

The defined model in the previous step starts learning the training data using the "fit" method. There are several hyperparameters and other crucial settings;

- The model will go through the entire training dataset fifteen times; "epochs = 15".
- Data is divided into batches of 32 samples each; "batch_size = 32".
- The model's performance is evaluated at each time by validation sets, which are unfamiliar data for the model; "validation_data=(X_test, y_test)". It helps monitor the signs of overfitting.

(6) Model Evaluation

After completing the model training phases, the model will be evaluated its performance with various metrics;

- Confusion Matrix
- Accuracy, Precision, Recall, and F1-score
- Plotting Receiver Operating Characteristic (ROC) Curve.

Discussion

To begin, *Figure 2* shows the performance of the model at each training phase, and *Figure 3 shows* its visualizations. Training accuracy became stable after the 6th epoch, which could show the model reached sufficiently capture the image patterns. On the other hand, the validation accuracy began to fluctuate after being topped at the 4th epoch, which could indicate the model was overfitting. In addition, the training loss decreases sharply at the initial phase and does steadily at the later part, which shows the model minimizes its error on the training set. For the validation losses, some fluctuations are observed after the 7th epoch although the trend is down continuously. Also, the divergence between the training and validation losses would be one of the challenges for the future model training. Overall, the model successfully learns the training set, as evidenced by higher accuracies and lower losses. However, the model might not be generalized against the unfamiliar data, which means overfitting on the training data. For the future model training to improve its performance and prevent overfitting, it is expected to use the regularization techniques, such as adding dropout and applying l2 or l1 norm; also, the early stopping would be one of the options.

Figure 4, which is the confusion matrix, indicates a high level of accuracy in the classification tasks between dogs and cats. There are only three misclassifications among the

whole validation samples of 73. These performances are reflected on the precision, recall, and f1-score, shown in *Figure 5*. Both precision and recall showed higher scores, but the model is likely to be sensitive to predicting cats, considering that all errors occurred when the model incorrectly predicted the actual dog images as cats. Nevertheless, since all four metrics showed high scores, the model shows relatively good performance.

Lastly, *Figure 6* is the ROC curve, which demonstrates outstanding performance in classifying the images of dogs and cats. The notable behavior is a steep rise of the ROC curve toward the top left corner, which means a higher true positive rate and lower false positive rate. Considering that the ideal ROC curve is drawn perpendicularly, our model's ROC curve approaches this ideal shape closely. Also, if we consider that the area under the ROC curve (AUC) of the model that perfectly predicts all labels shows 1, our model's AUC of 0.999 indicates quite close to the ideal classification model.

Conclusion

This project has tested the efficacy of the VGG16 model in performing binary classification tasks, especially classifying images of dogs and cats. The model's performance was evaluated, considering the potential signs of overfitting simultaneously, under various metrics – confusion matrix, accuracy, precision, recall, F1-score, and the ROC curve. The VGG16 model can achieve high scores for all metrics, which could be evidence that the VGG16 model encouraged building a robust classification model while reducing computational costs and extensive training time.

However, during the training phases, the validation losses slightly fluctuated and observed a divergence from training losses. These behaviors showed the potential risks of overfitting on the training data. As one of the challenges for further research, it should focus on what kinds of strategies are suitable to mitigate the overfitting risks, when we should stop the training iterations, and how we should tune hyperparameters of regularizations, such as the threshold of dropout and degree of l2 or l1 norm.

Appendices

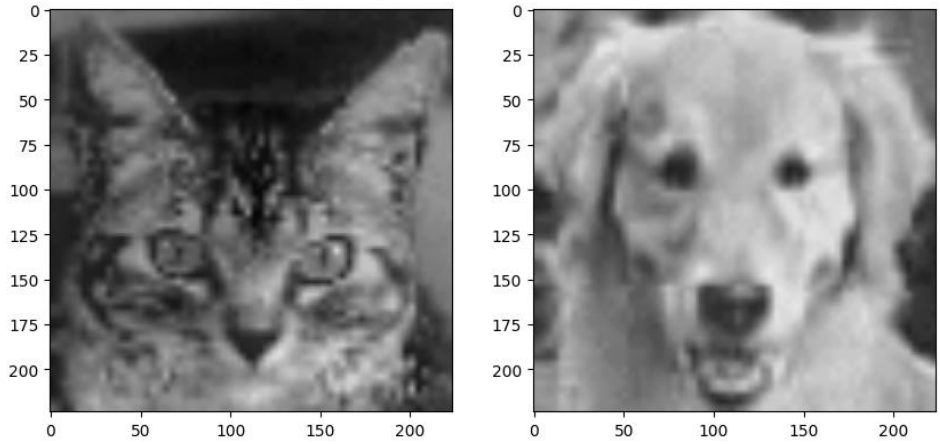

***Figure 1**: Images of a cat from the first row of data and a dog from the last row of data*

	Epoch	Loss	Accuracy	Val_Loss	Val_Accuracy
0	1	1.1325	0.4675	0.5488	0.6438
1	2	0.4398	0.7870	0.3924	0.7808
2	3	0.2141	0.9290	0.3015	0.8767
3	4	0.1132	0.9763	0.1576	0.9726
4	5	0.0977	0.9822	0.1319	0.9452
5	6	0.0726	0.9941	0.1040	0.9726
6	7	0.0365	0.9941	0.1564	0.9452
7	8	0.0247	0.9941	0.0853	0.9726
8	9	0.0154	1.0000	0.1030	0.9589
9	10	0.0095	1.0000	0.1413	0.9452
10	11	0.0080	1.0000	0.0851	0.9726
11	12	0.0068	1.0000	0.0671	0.9726
12	13	0.0060	1.0000	0.0861	0.9726
13	14	0.0042	1.0000	0.0953	0.9589
14	15	0.0037	1.0000	0.0904	0.9589

***Figure 2**: Results of each training steps from the VGG16 model*

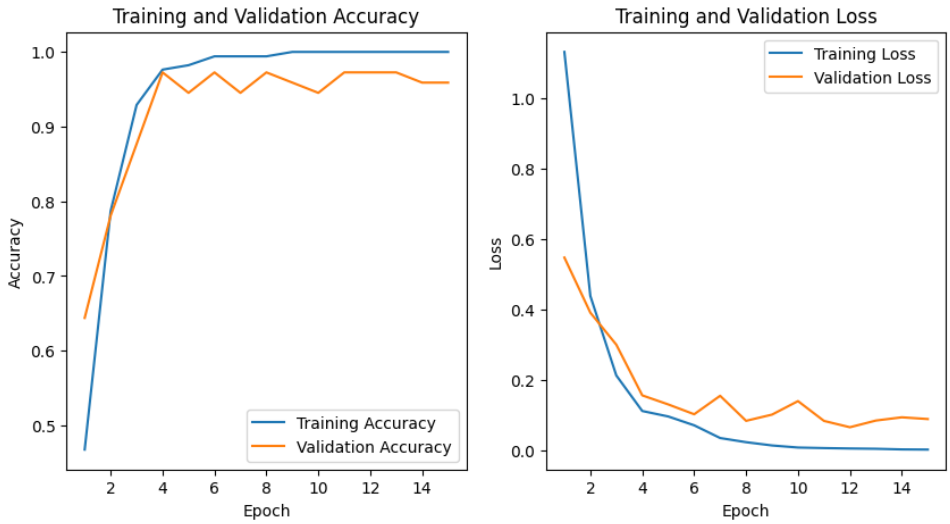

***Figure 3**: Accuracy and loss in training and test sets at each iteration*

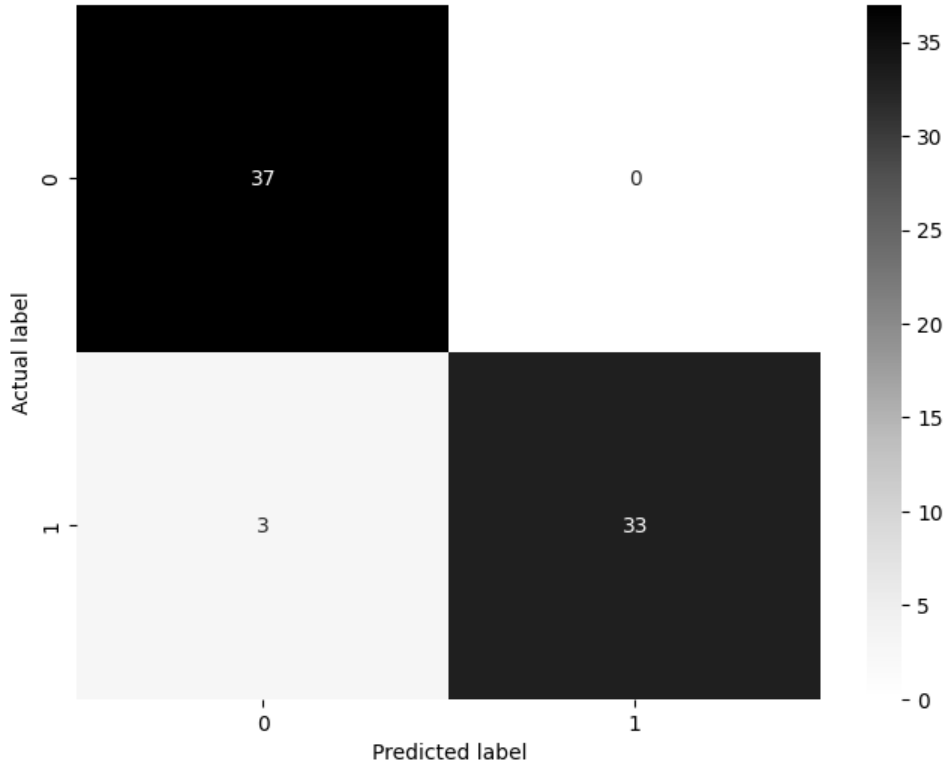

***Figure 4**: Confusion matrix*

	precision	recall	f1-score	support
0	0.93	1.00	0.96	37
1	1.00	0.92	0.96	36
accuracy			0.96	73
macro avg	0.96	0.96	0.96	73
weighted avg	0.96	0.96	0.96	73

***Figure 5**: Classification reports*

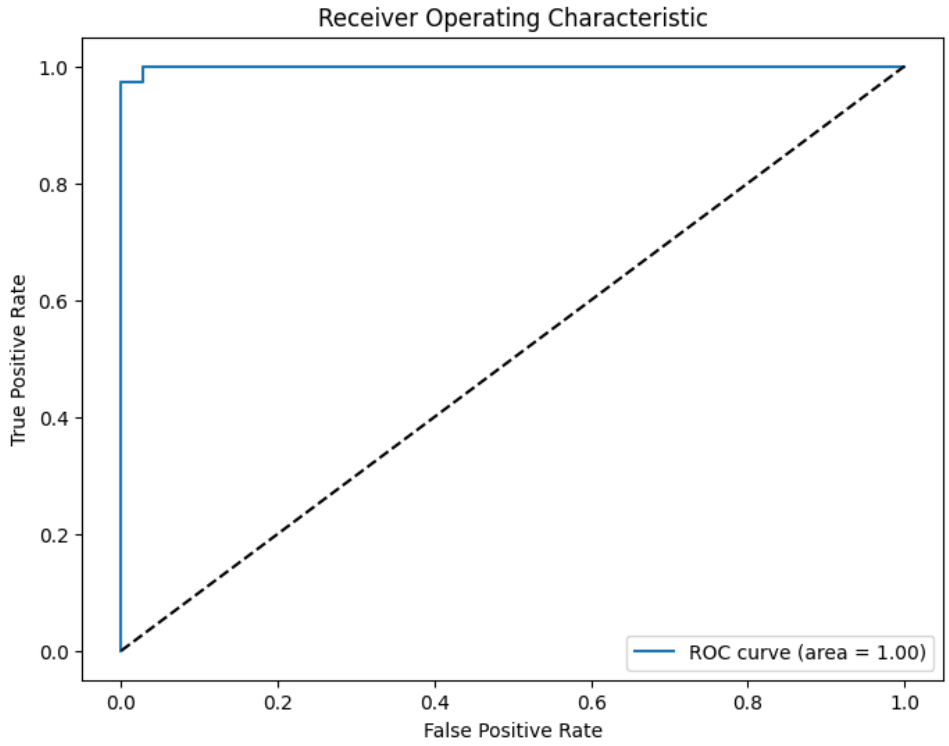

***Figure 6**: ROC curve*

References

Akther, J., Harun-Or-Roshid, M., Nayan, A. A., & Kibria, M. G. (2121). Transfer learning on VGG16 for the Classification of Potato Leaves Infected by Blight Diseases. *Emerging Technology in Computing, Communication and Electronics (ETCCE)*. https://doi.org/10.1109/ETCCE54784.2021.9689792

Defferrard, M., Bresson, X., & Vandergheynst, P. (2017). Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *Advances in Neural Information Processing Systems 29*. https://doi.org/10.48550/arxiv.1606.09375

Gardner, M. W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. Atmospheric Environment (1994), 32(14), 2627–2636. https://doi.org/10.1016/S1352-2310(97)00447-0

Gu, J., Yu, P., & Ding, W. (2021). Leaf species recognition based on VGG16 networks and transfer learning. *IEEE Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. pp. 2189-2193, doi: 10.1109/IAEAC50856.2021.9390789.

Hermana, A. N., Rosmala, D., & Husada, M. G. (2021). Transfer Learning for Classification of Fruit Ripeness Using VGG16. *Association for Computing Machinery*. https://doi.org/10.1145/3450588.3450943

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

Shah, S. S. H., Jamil, N., Khan, A. U. R., et al. (2022). Performance comparison of visualization-based malware detection and classification techniques. *2022 17th International Conference on Emerging Technologies (ICET)*, 200–205. https://doi.org/10.1109/ICET56601.2022.10004652

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations (ICLR 2015)*, 1–14. https://doi.org/10.48550/arxiv.1409.1556

Valls, J.M., Aler, R., Galván, I.M., & Camacho, D. (2021). Supervised data transformation and dimensionality reduction with a 3-layer multi-layer perceptron for classification problems. *Journal of Ambient Intelligence and Humanized Computing 12*, 10515–10527. https://link.springer.com/article/10.1007/s12652-020-02841-y