University of Milan

Machine Learning Project

Neural Network The Binary Classification Of Cats And Dogs

Dao Yen Hoa
988008

Prof. Nicolò Cesa-Bianchi

Abstract

The project at hand represents the experience from Machine Learning course that has provided me with a comprehensive understanding of the theory and practice of building and applying Neural Networks. Specifically, this final report focuses on the task of classifying a complex dataset of images depicting cats and dogs. This task required to utilize a range of advanced machine learning techniques, including data pre-processing, model regularization, and cross-validation. To facilitate our exploration of these concepts, I developed an experimental framework that was divided into four distinct stages. The first stage involved an overview of theories to build and train neural networks, while the second stage focused on data pre-processing techniques designed to improve the quality and accuracy of training data. The third stage of our project explored various model architectures that were used to optimize the task result. Finally, cross-validation techniques was applied to evaluate the risk and select the best model for classification task. Overall, this project represents a significant accomplishment in journey to become skilled machine learning practitioners and has provided valuable insights into the complex and exciting world of deep learning.

Table of Contents

Chapter 1
Introduction

Convolutional Neural Networks (CNNs) have emerged as one of the most successful techniques to image classification problems because of its exceptional capacity to extract and learn complex characteristics directly from raw picture data. The task of image classification is a crucial aspect of the field of computer vision, with far-reaching applications across a diverse range of fields, from the development of self-driving cars to the diagnosis of complex medical conditions. In this report, I undertake a detailed study of the use of neural networks for classifying images of cats and dogs, using advanced cross-validation techniques to evaluate the accuracy and efficiency of different CNN architectures. The overarching goal is to determine the most effective and efficient approach to image classification using neural networks and to develop new insights into how to optimize the accuracy and efficiency of this powerful algorithm.

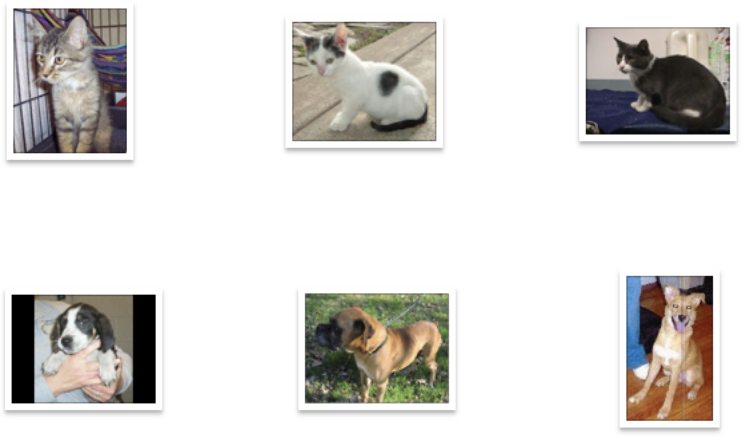

Figure 1.1: Example of images of cats and dogs in dataset

1.1 Neural Network

Convolutional Neural Networks (CNNs) are particularly well-suited for image processing tasks, due to its unique architecture, which consists of three key components: convolutional layers, pooling layers, and fully connected layers. At the heart of a CNN are a series of hidden layers, each of which plays a critical role in extracting meaningful features from the image data. For example, the convolutional layer applies a set of convolution filters to the input image, resulting in a set of feature maps that highlight different aspects of the image. The pooling layer, on the other hand, downsamples the feature maps, reducing the dimensionality of the data and improving the computational efficiency of the network. Finally, the fully connected layer aggregates the features learned by the previous layers and applies them to the task of object recognition, identifying the specific object present in the input image. Together, these three essential components work in harmony to enable CNNs to achieve exceptional performance in a wide variety of image classification tasks, making them an invaluable tool for researchers and practitioners alike.

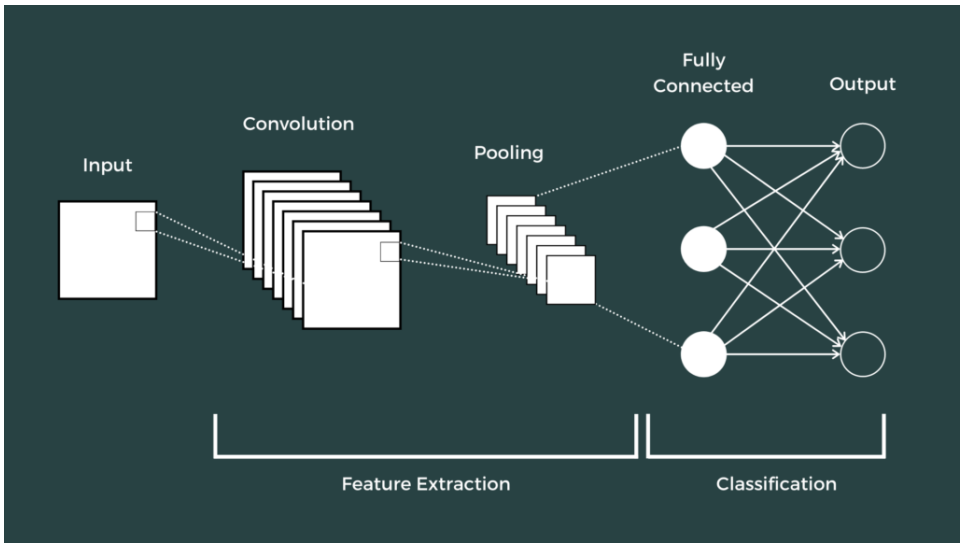

Figure 1.2: A sample architecture of CNN

Convolutional Layer: This is the first step which extract features from image.

- Number of Filter: Also known as the depth or channels, is a critical parameter in determining how many unique feature maps or patterns a convolutional layer can learn..

- Kernel Filter: The kernel size determines the size of the local receptive field of the convolutional operation, which is the area of the input data that is considered at each convolutional step.

- Padding: The purpose of padding is to preserve the spatial dimensions (width and height) of

the input data throughout the convolutional layers."same" padding, where the output feature map has the same spatial dimensions as the input data.

- Activation: Refers to the application of an activation function to the output of a convolutional operation, for introducing non-linearity into the model. In deep learning, ReLU and Sigmoid are two commonly used activation functions.

Pooling Layer: Pooling is a technique used to down-sample feature maps, reducing their dimensionality. The commonly used method is max pooling, which selects the highest value from a local neighborhood. It reduces the spatial dimensions of the feature maps by selecting the maximum value in each local region.

Flatten: To prepare the feature maps for further processing, a flattening layer is typically added to convert the multi-dimensional feature maps into a one-dimensional vector, which can be easily fed into a fully connected layer.

Fully Connected Layer: It connects every neuron in one layer to others

Loss Function: The binarycrossentropy function computes the cross-entropy loss between true labels and predicted labels.

Optimizer: RMSProp is a popular optimization algorithm used in deep learning to update the parameters of a neural network during training to balance the step size, or momentum, and uses an adaptive learning rate for each weight based on the magnitude of the gradients observed for that weight in the recent past. In addition to RMSProp, another widely used optimizer is Adam. It is an extension of the stochastic gradient descent (SGD) algorithm, which is one of the most popular optimization algorithms used in deep learning. The combination of momentum term and bias correction terms has made Adam a popular choice to further improve the optimization performance.

Metrics: 'Accuracy' calculates how often predictions equal labels. This metric creates two local variables, total and count that are used to compute the frequency with which predictions matches the true values.

1.2 Cross-Validation

Cross-validation is a popular technique used in machine learning to assess the performance of models. The process involves dividing the dataset into multiple folds and training the model on various combinations of training and validation sets. By doing so, cross-validation provides a more accurate estimate of the model's performance, as it is tested on several independent subsets of the data. There are different types of cross-validation techniques, such as k-fold cross-validation and stratified k-fold cross-validation. K-fold cross-validation divides the data into k equal-sized folds, and each fold is used once for validation, while the others are used for training.

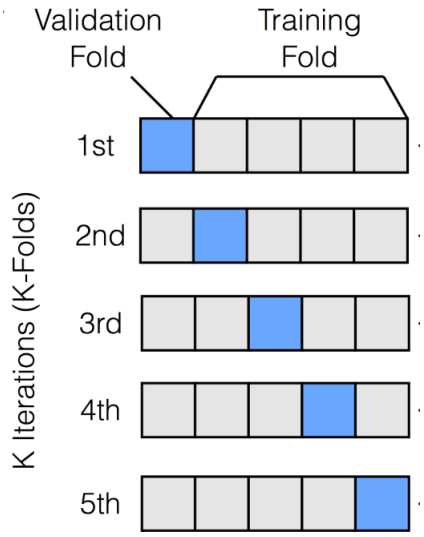

Figure 1.3: 5-fold Cross Validation

Moreover, zero-one loss, also referred to as the misclassification loss, is an evaluation metric frequently employed in machine learning for assessing the accuracy of classification models. This metric is particularly valuable when working with binary classification tasks. By taking the average loss across all k iterations, the zero-one loss provides an overall estimation of the model's effectiveness.

$$\ell(y, \widehat{y}) = \begin{cases} 0 & \text{if } y = \widehat{y}, \\ 1 & \text{otherwise.} \end{cases}$$

Figure 1.4: Zero-one loss

1.3 VGGNet

VGGNet, developed by the Visual Geometry Group (VGG) at the University of Oxford, is a renowned convolutional neural network (CNN) architecture. It stands out for its simplicity and effectiveness in image classification. At the core of VGGNet is its deep architecture, comprising multiple stacked convolutional layers. Typically, the network employs small 3x3 filters with a stride of 1, followed by a max pooling layer with a 2x2 window and a stride of 2. This design enables the network to learn hierarchical representations of the input images, capturing important features in the process.

Chapter 2
Data Processing

2.1 Datasets and Transformation

This dataset includes 25000 images shared by dogs and cats as jpg with variety of sizes. There is 2 error images while processing which are 666 and 11702 so they have been removed. 24998 images then was transformed into RGB with reduced size.

2.2 Data splitting

Finally, the dataframe has been splitted into three sets: 0.2 for test set, the rest is for train set and 0.2 of it is for validation. Moreover, ImageDataGenerator was used and rescale all three datasets into [0,1] range by 1/255.

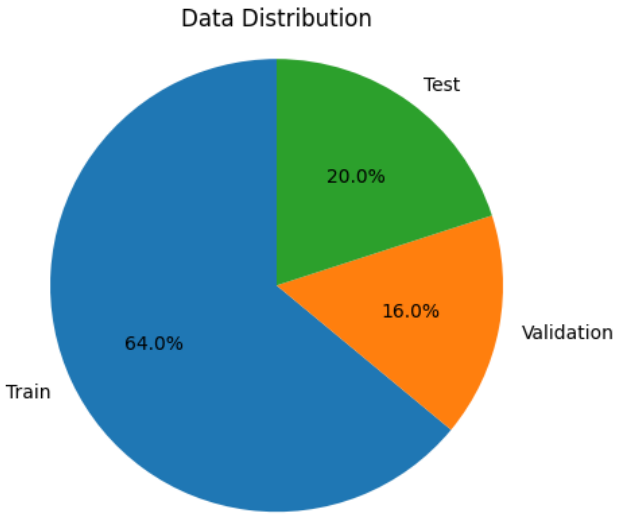

Figure 2.1: Data chart

Chapter 3
Models and Results

There are 4 models inspired by VGGNet which is a Convolutional Neural Network architecture proposed by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014, and cross-validation have been used in this project as described in the next part. The same things are they all have the kernel size 3,3 with padding 'same', activation 'ReLU', and maxpooling 2,2. Binaryloss is used for model compile loss. Finally, 10 epochs to fit the model using train and validation data. This is the table of the differences between their architecture.

	Model 1	**Model 2**	**Model 3**	**Model 4**
Convolutional Layers	32, 64	32, 64, 128	32, 64, 128, 256	64, 128, 256, 512
Padding	same	same	Same - Valid	Same - Valid
Dropout			0.2	0.4
Fully Connected	128 - ReLU 1 - Sigmoid	128 - ReLU 1 - Sigmoid	128 - ReLU 1 - Sigmoid	128 - ReLU 1 - Sigmoid
Optimizer	rmsprop	adam	adam	adam

Figure 3.1: Hyperparameters of each model

3.1 Model 1

When examining the performance of model 1, the findings show that the training process demonstrates a significantly higher rate of development, but the comparable progress in the validation phase is just slightly perceptible. Furthermore, a deeper look indicates that, while the training results fluctuate, the validation progress follows a more stable and linear course. Because the improvement does not translate considerably to the validation set, these findings imply that the model is slightly overfitting to the training data.

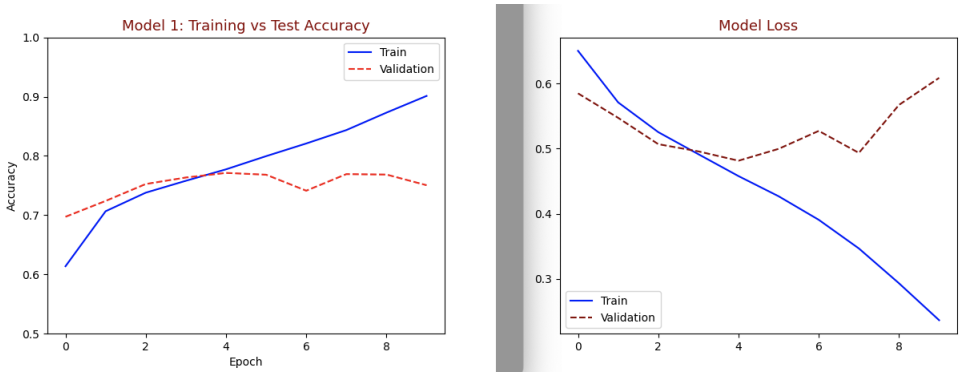

Figure 3.2: Accuracy and Loss of Model 1

3.2 Model 2

In an attempt to remedy this issue, model 2 received a revision by changing the optimizer to 'adam'. This change resulted in a slightly improved outcome; nonetheless, the improvement seen in the test results was not significant. Furthermore, the convergence of training and validation progress in model 2 is noteworthy. The tighter alignment of these two measures suggests that the optimizer tweak resulted in a more balanced optimization process in which the model's performance on unknown data (validation set) better matches its performance on the training set.

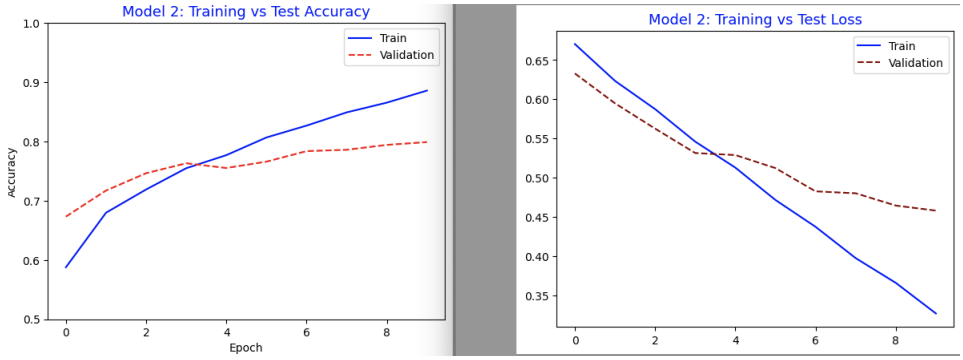

Figure 3.3: Accuracy and Loss of Model 2

3.3 Model 3 - Model 4

To further mitigate the risk of overfitting and improve the generalization capabilities of the models, Dropout regularization was introduced in both model 3 and model 4. Dropout randomly deactivates a certain percentage of the neural network's units during training, which helps prevent the model

from relying too heavily on specific features or patterns in the data. This regularization technique aims to enhance the model's ability to generalize to unseen examples.

Upon evaluating the impact of Dropout on the models, it is worth noting that while the inclusion of Dropout did not significantly improve the accuracy of the predictions, it did yield a pattern among the training, validation, and test results. This indicates that the models are now better aligned in terms of their performance across different datasets. It also suggests that the models are not overly biased towards the training data and are more likely to make predictions that are consistent across various datasets.

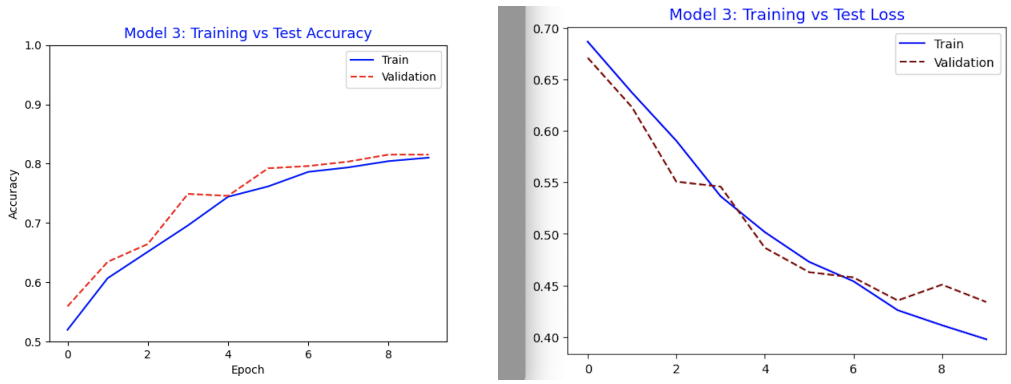

Figure 3.4: Accuracy and Loss of Model 3

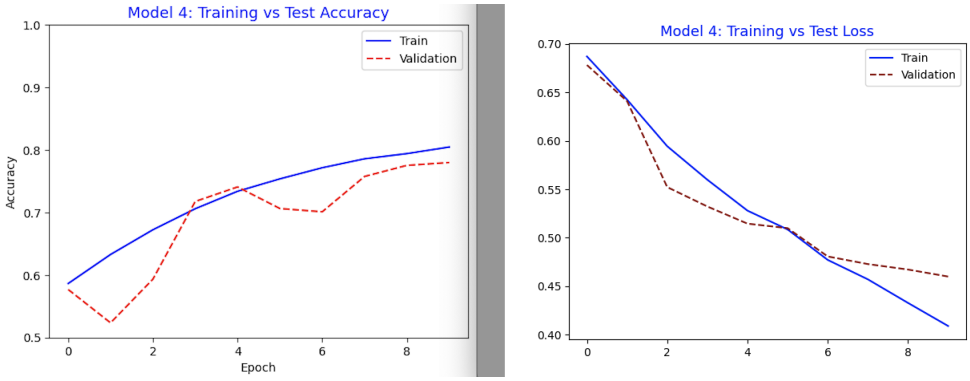

Figure 3.5: Accuracy and Loss of Model 4

Next, a comparison between model 3 and model 4 reveals interesting insights. Model 4, which incorporated a higher dropout rate and convolutional layers, did not demonstrate superior performance compared to model 3. In fact, the introduction of these additional layers and higher dropout rates resulted in a relatively worse outcome. This outcome emphasizes the importance of carefully fine-tuning model architectures and hyperparameters, as indiscriminately increasing complexity or regularization might not always yield improved results.

Test	Model 1	Model 2	Model 3	Model 4
Loss	0.4897	0.4529	0.4240	0.4474
Accuracy	0.7960	0.8068	0.7982	0.7888

Figure 3.6: Test results of 4 models

Finally, the performance evaluation of the models demonstrated the efficiency of certain improvements, such as modifying the optimizer and incorporating Dropout regularization, in increasing the performance of training, validation, and test outcomes. However, while adding architectural modifications, it is critical to take caution to ensure that the complexity and regularization processes are correctly balanced for best performance.

3.4 Cross-Validation

In this section, I choose model 1 and model 4 to apply k-fold (which is 5) to gain insights the model and estimate risk and evaluate the models' abilities to classify instances correctly using the zero-one loss metric. Model 1 represents the most basic architecture an model 4 instead more complicated and higher values of convolutional layers.

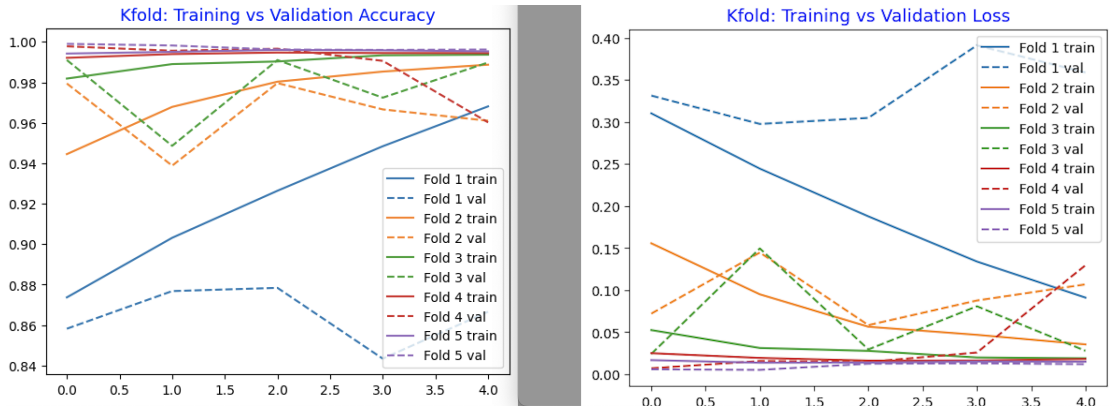

Figure 3.7: 5-fold Cross Validation Results of Model 1

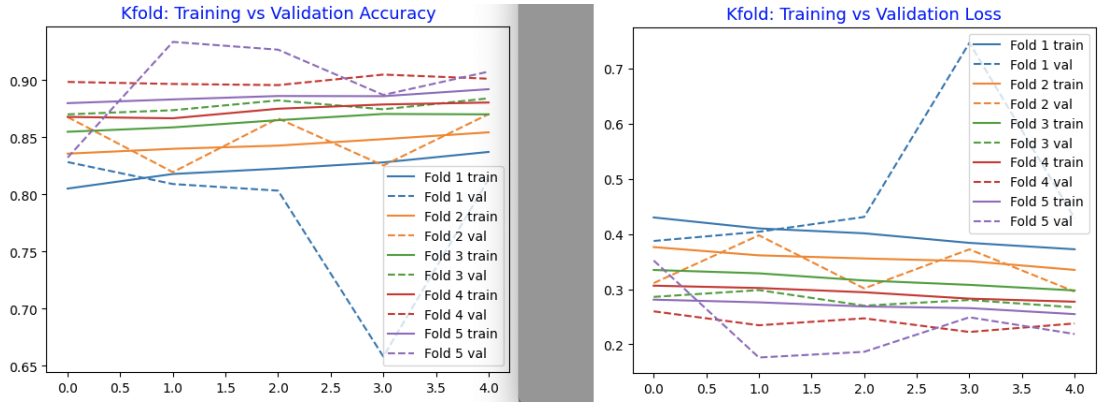

Figure 3.8: 5-fold Cross Validation Results of Model 4

Furthermore, I attempted to construct a new model architecture (called model5) by experiencing 5 trials using the Randomsearch tuning which involves randomly sampling from a predefined search space of hyperparameters and evaluating the model's performance using these randoms. Then, applying kfold to it to see if it may produce a better outcome. The table below displays the outcomes of three models using test data.

Test	**Model 1**	**Model 4**	**Model 5**
Loss	0.12	0.28	0.22
Accuracy	95.4%	87.54%	91.2%

Figure 3.9: Test Results using Cross-Validation

Chapter 4
Conclusion

Finally, the performance examination of the models revealed numerous elements of Neural Networks. They demonstrate that optimizer adam can produce better results than rmsprop, and that incorporating Dropout can assist decrease overfitting and make the model slightly more stable during training, validation, and testing. Model 1, even the most simple model, does not have the best accuracy as model 2, but it improves significantly by using cross validation. Model 3 and 4 on the same side include Dropout and more convolutional layers, as well as a different sort of padding. However, their final results did not exhibit a significant difference or improvement. Moving to next step with cross validation, model 1 has shown the lowest value of loss as well as highest accuracy compared to other two which are model 4 and 5 (the one adding by using tuning technique).

However, there are still opportunities for advancement, such as researching the reasons of overfitting and experimenting with hyperparameters, dataset variety, and architectures. Running might also be considered an issue, finding ways to shorten the runtime would enable more extensive exploration of various model designs to identify the optimal hyperparameter settings. Not only that, but we can also use PCA and the optimizer SGD to extract more meaningful features and refine the training process.

References

Rastogi, Sajal. “K-Fold — K-Fold Averaging on Deep Learning Classifier.” Analytics Vidhya, 16 Sept. 2021, www.analyticsvidhya.com/blog/2021/09/how-to-apply-k-fold-averaging-on-deep-learning-classifier/.
Mishra, Mayank. “Convolutional Neural Networks, Explained.” Medium, 2 Sept. 2020, towards-datascience.com/convolutional -neural-networks-explained-9cc5188c4939.
https://github.com/christianversloot/machine-learning-articles/blob/main/how-to-use-k-fold-cross-validation-with-keras.md
Simonyan, Karen, and Andrew Zisserman. VERY DEEP CONVOLUTIONAL NETWORKS for LARGE-SCALE IMAGE RECOGNITION. 2015

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.