Dog breed multi-class classification with transfer learning

Michael Ajao-olarinoye

10118047

7088CEM

[olarinoyem@uni.coventry.ac.uk](mailto:olarinoyem@uni.coventry.ac.uk)

data science and computational intelligence

I acknowledge Zainab Umar Bugaje as my team member

Abstract – this paper shows the use of transfer learning to solve a multiclass classification problem. Transfer learning is been widely used now in the data science community. Mobilenetv2, inceptionv3 and inception\_resnet\_v3, were compared in this paper and the best for the image classification problem was found to be the inception\_resnet\_v3. The result got Shows accuracy of 96% a loss function of 0.122 And an epoch of 17. I used the test data for prediction on the model.

Keywords – CNN, models, functions, neural networks, algorithm, transfer learning, convolutional.

# Chapter 1 - Introduction

In this report, I will be discussing neural network techniques that I used to solve a real-world image classification problem. Artificial neural network Are computational structures slightly inspired by animal brain biological neural networks, these systems learn to execute tasks by considering scenarios, usually without being configured under task-specific rules. For example, in image recognition, they can learn to recognize images that include rabbit by examining examples of images that have been manually labelled as "rabbit" or "no rabbit" and using the findings to recognize rabbit in other images (Yung-Yao et al. , 2019).

The use of an artificial neural network to solve problems in supervised and unsupervised learning has been improved upon over time and the different algorithm has been created to help in solving the problem. Real-world application of neural network algorithm has made the community to generate interest in developing and improve on a better algorithm.

The problem I want to solve in the project is a multi-class classification problem. The techniques proposed in this paper involve developing a solution based on using deep convolutional neural network and also implementing transfer learning with various pretrained algorithm, which is a predictive modelling technique for a specific but somehow related situation, which can then be reused partly or wholly to enhance the training and boost the performance of a model on an interesting issue. In machine learning transfer learning means It involves reusing weights from a pre-trained network model in one or more layers in a new model and then holding the weights constant, fine-tuning them or changing the weights altogether while training the model.

I will explain further a bit of image classification used in computer vision. Classification of images, which can be defined as the task of categorizing images into one of several predefined classes, is a fundamental problem in the field of computer vision. This is the basis for other computer vision functions, such as location, recognition, and segmentation (Karpathy, 2016). In recent years, image recognition has been commonly used as one method used to identify the target object of the image (Simonyan and Zisserman,, 2015). According to the paper written by (Simonyan and Zisserman,, 2015) which used an architecture that is made up of networks of convolution filters and developing a model. Some of the most common approaches are the approach used for Neural Networks. This approach relies on the study of the decision-boundary surface separating adult images from non-adult images utilizing the computer-based classification law, called perceptron (Rosenblatt, 1958).

Convolutional Neural Networks are used the most frequently for image classification tasks. They are a kind of Neural Network and were first introduced by (Le Cun et al. n.d.) Where the LeNet-5 model demonstrated greater results in the dataset of handwritten digits than supporting vector machines and K-nearest neighbour algorithms. using a convolutional neural network with different methods changed and implemented in different algorithm makes it that go-to for multiclass image classification. This report would have various chapters that would discuss the steps taken from preprocessing to selecting the models and experimentation:

Chapter 1 the introduction which would briefly explain what artificial neural network is, and briefly talk about image classifications, multi-class classifications problems and discuss other works that have been done in the domain of image multi-class classification detection.

Chapter 2 will discuss the problem we are trying to solve and the description of the dataset. We will discuss the method and various neural network algorithm that will be used in the experiment.

Chapter 3 this is where the experimental process that was involved in preprocessing the dataset, creating and training and validating the models and comparing in between them.

Chapter 4 this chapter will focus on the results gotten from the experiments and conclusion.

this paper is written using the dataset of dog’s pictures found on Kaggle competitions, the dataset comes with a multiclass classification problem that is trying to predict that the pictures of each dog.

Image classification according to ( Lu and Weng, 2007) Image classification is a dynamic method that can be influenced by several variables and has been attempting to solve the problem by highlighting the overview of major advanced classification strategies and techniques used to improve classification accuracy

## Related works

There are various papers written on image classification using neural network, I will discuss a few in this paper.

The work reveals that the framework for multi-class classification of airborne sensor data is tested by a single SVM study against a collection of classifiers commonly used in remote sensing, in particular the effect of the scale of training on classification accuracy. In addition to the SVM, the same datasets were classified using differential analysis, decision tree, and multilayer neural perception networks. (Foody and Mathur, , 2004).

In another paper, it was proposed that using a hybrid method by mixing a few established algorithms like nearest neighbour method, decision tree model and others might be effective in a multiclass classification problem, and decided to focus mainly on creating a modern, efficient and Effective multi-class image recognition algorithm for adults using MPEG-7 descriptors. Generic MPEG-7 descriptors derive model attributes from the picture and use it as input values for the neural network being proposed (Kim et al., 2005)

In a paper on deep convolutionary networks, it was suggested that a deep convolutionary neural network architecture codenamed Inception that achieves the current state-of-the-art classification and detection in the 2014 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC14) be developed. (C. Szegedy et al. , 2015).

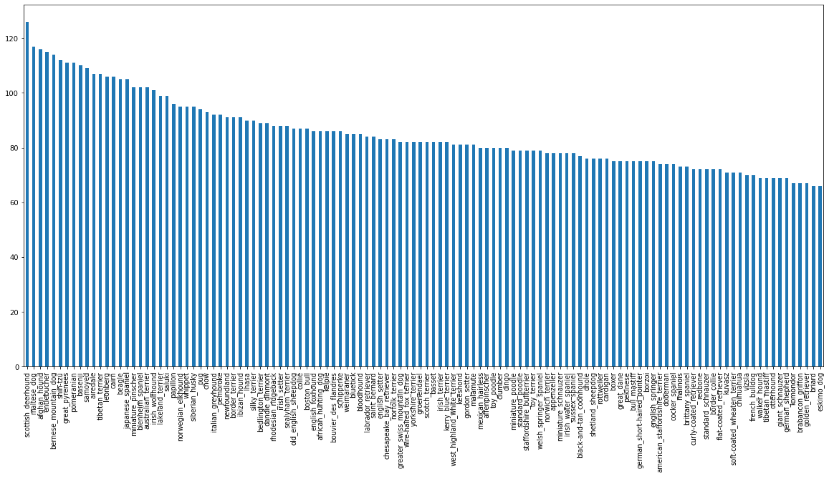
A paper written on Image Classification Algorithm Based on Deep Learning-Kernel Function Introduces the concept of sparse representation in the architecture of the deep learning network and makes good use of the sparse representation of well-multidimensional data linear decomposition capabilities and the deep computational advantages of multi-layer nonlinear simulation to complete the complex function approximation in the deep learning model ( Jun-E Liu and Feng-Ping An , , 2020).

A paper Implemented multi-class image classification using the VGG-19 Deep Convolutional Network as a Transfer Learning System where VGGNet is pre-trained in the ImageNet dataset (Dr. Vaibhav Kumar 2020).

# Chapter 2 – the dataset and models

The problem we would like to solve is the Multiclass classification of an image dataset using various neural network methods. we are trying to classify multiple different breeds of dog. Multi-class image classification is an important issue because it is the same type of technology that Tesla uses in its self-driving cars or Airbnb uses to automatically add information to their listings.

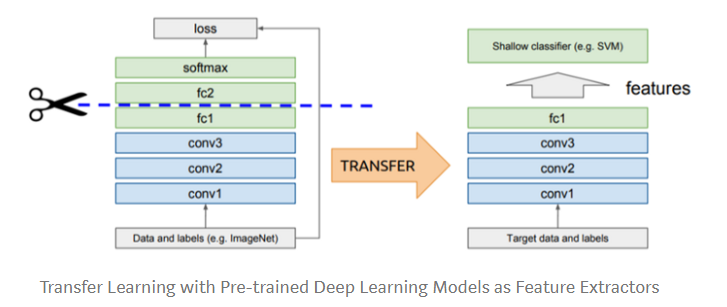
The dataset consists of a set of 10,000 + pictures of 120 distinct breeds of dogs. The dog data set consists of 120 breeds of dogs. I discovered that the labels are different amount of pictures for each breed of dogs do the 10,222 different id’s and wanting a 20 different breeds days an average of 60 plus images for each dog breed Google recommends to have a minimum of 10 images per class to start to solve it classification problem with neural network Because my data was in a zip file I had to unzip it then cheque the labels to the images to cheque if it's correct and also to a cheque for missing values.



1. A chart showing the 120 class (dog breed) with its respective values.

In this paper I implemented the principle of transfer learning, Transfer learning is the concept of challenging the independent leaning model by using learned knowledge for one problem to tackle similar ones.

Deep learning systems and models are layered architectures which at different levels learn different features. Such layers are then eventually connected to the last layer (usually a completely connected layer for supervised learning) to achieve the final output. This hierarchical model helps one to use a pre-trained network as a fixed function extractor for certain functions, without its final layer, the main concept here is to use the weighted layers of the pre-trained model to remove features but not change the weights of the layers of the model when experimenting with new data for the new challenge (Dipanjan (DJ) Sarkar 2018).



1. A diagram showing a transfer learning algorithm with pretrained deep learning models as feature Extractors.

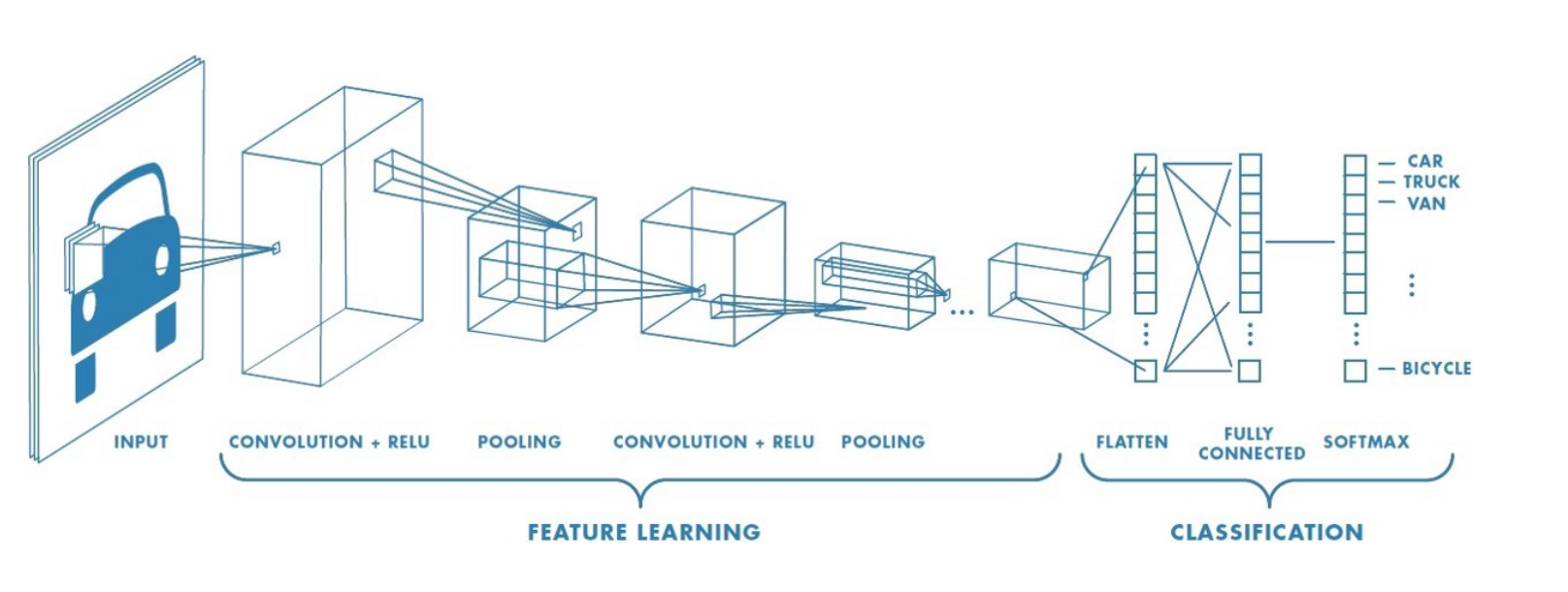
Transfer learning has long been practiced as a method in computer learning to address the various problems in visual categorization. In recent years, demands for high accuracies and technical efficiencies are growing due to the abundance of knowledge such as images, audios, and videos over the Internet. For these factors, the transition of information has drawn significant interest in the areas of machine learning and computer vision.

One of the key requirements for learning transfer is the existence of templates which function well in source tasks. Their respective departments have publicly discussed many of the cutting-edge, deep learning techniques. Some expand for deep learning into the most common domain. Deep learning has been used widely utilizing different CNN systems to solve numerous computer vision challenges, such as image recognition and predictions the report, How transferable are featured in deep neural networks, present their findings on how the lower layers function as typical extractors of computer vision features, such as edge detectors, while the final layers operate for different tasks. (Yosinski et al. 2014).

In this paper I will discuss creating a simple convolutional neural network, comparing three different pre-trained models which are mobilenet\_v2, inception\_v3 and inception\_resnet\_v2.

Convolutional neural network

A convolutional layer neural network is a deep learning algorithm capable of processing an input image, attributing properties such as learning weights and preferences to specific aspects/objects in the picture and distinguishing them from one another. Pre-processing within ConvNet is even smaller than other classification algorithms.



1. A diagram of the process a fully convolutional neural network model takes.

CNN are feedforward networks. A ConvNet can capture the Spatial and Transient dependencies in an image by adding appropriate filters. The architecture makes the image dataset better suited because of the decrease in the number of parameters involved and the reusability of weights (Sumit Saha 2018). The task of the ConvNet is to reduce the picture to a form that is easier to interpret, without missing the features that are crucial to make successful predictions.

MobileNet\_v2

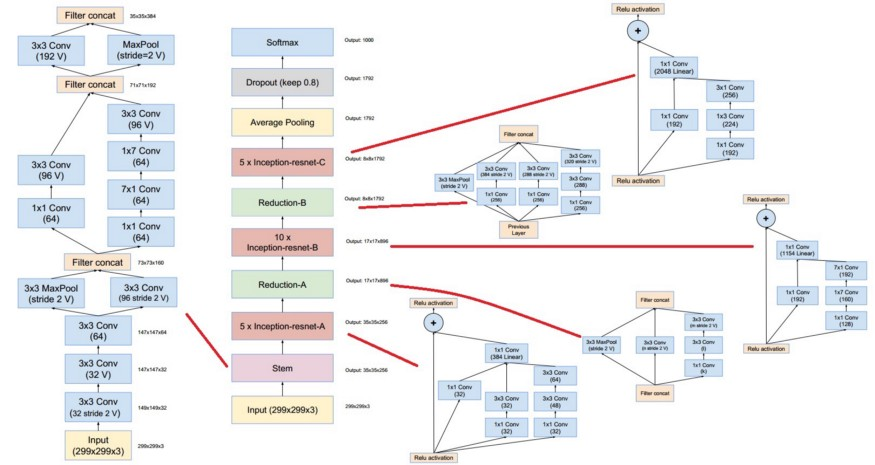
Mobilenet\_v2 improves the state-of-the-art efficiency of smartphone devices on various assignments and tests, as well as over a few different device sizes. We also explain how these mobile models can be extended to object detection in a novel system called SSD Lite. The MobileNetV2 design is based on an inverted residual framework where the input and output of the residual block are thin layers of bottleneck relative to traditional residual architectures that use expanded data representations. The method followed enables the input/output realms to be decoupled from the expressivity of the transition, thereby offering a simple structure for further study. We assess our success in classifying ImageNet, detecting COCO artefacts, segmenting VOC images (Sandler et al. 2018).

Inception\_v3

The Inception v3 paper explores how to scale up networks in a way that aims to allow the most efficient use of distributed computing by adequately factored advances and rigorous regularisation. Benchmarking methods for the 2012 ILSVRC Classification Software Testing Kit reveal substantial state-of-the-art gains: 21.2 per cent top-1 and 5.6 per cent top-5 errors for single frame measurement utilising a network with a computational expense of 5 billion multiple-adds per estimation and using less than 25 million parameters.. For a range of 4 models and multi-crop assessment, we record the validation set for 3.5% top-5 error, 3.6% test sample error and the validation set with 17.3% top-1 error (Szegedy et al. 2015). Factorization was implemented in the convolution layer to further minimize dimensionality, to mitigate the issue of overfitting.

inception\_resnet\_v2

The implementation of residual links in combination with a more conventional architecture culminated in The state-of-the-art performance of a competition. its results were close to the new Inception-v3 network level. In the paper on inceptionv4, it was provided strong scientific evidence that residual link preparation greatly accelerates the creation of Inception Networks. There is also some evidence that residual start-up networks outperform equally costly start-up networks by a slim margin without residual links. They also propose some new simplified architectures for residual and non-residual networks of initiation.



1. A diagram of the schema and algorithm of the inception\_resnet\_v2 model.

They also demonstrate how proper activation scaling stabilizes the training of very large residual networks. We reach 3.08% top-5 error on the ImageNet classification (CLS) test set with a total of three residuals and one Inception-v4. It has the computational costs of Inception-v4. Inception-ResNet-v2 was working even quicker and achieved marginally higher final performance than Inception-v4 (Sik-Ho Tsang 2018).

# Chapter 3 – experimentation

In the experimentation of this paper, I will be using google colab as a platform, because the use of a GPU is important to build and train our model. The process taken in this paper will involve getting the data ready by downloading it from Kaggle, preprocessing and creating train, test and validation split, Choosing, fit / training the model using TensorFlow Hub, tf.keras.applications, Tensor Board, Early Stopping, Evaluating a model, by looking at the ground through truth label, improve the model through experimentation.

After downloading the dataset and creating a workbook on colab and importing the required library. Google drive was were my dataset was kept and connected Google colab notebook to utilize the use of a GPU for computation in doing preliminary data exploration, I checked the number of unique breeds in the data, the true labels against the false labels.

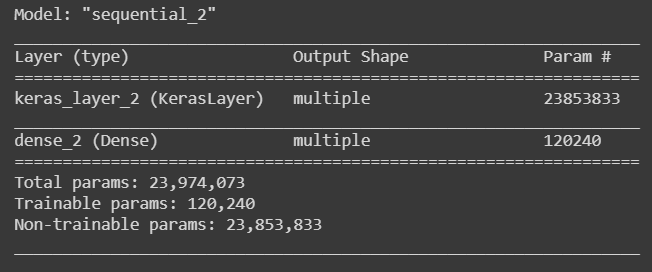
After converting the data to Boolean, it is important to convert it to numbers before passing it into a machine learning model. Since we have over 10,000 images, I will start the experimentation with just 2000 images as an increase as need be split my data into train and tests 80% training at 20% testing. the dataset turned from images into tensors and resized into different shapes for different models to be used.

I created Various functions like the function to process each image, then I created data batches in deep learning, I read that instead of finding patterns in your entire data set at the same time we often used it to find patterns in batches because the files may take up memory on the GPU when trying to compute them at the results, which might lead it to get an error. which might lead to the model not training very well he created another chapter to get the image label then I created a function To turn data into batches of size 32, By visualising the data batches I created the function to show 25 images using matplotlib to make competition efficient that is when we use the batch which is tightly wound collection of tensors.

The next process involves creating and training a model well it was three different models we created our chain Since our data is ready I use TensorFlow hub and choose image problem to men and chose image classification Doing this I got a different set of pre-trained models And checking the image shape of each model to determine what models to use for mobileNet\_V2\_132\_224 model, says the image shape is 224 for the height and 224 for the width. Both Inception\_v3 and Inception\_resnet\_v2 both of image sizes of 299 for height and width.

Before a model can be built there a few things that need to be defined, the input shape the output shape and URL of the model want to use both the input and output shape must be in form of tensors. using Keras a function was created to take in the input shape the output shape and the model was chosen URLs, the function defines the layers in the Keras model in a sequential way, compiles the model, build the model.

In setting up the models layer I went to read the Keras documentation that talked about functional and sequential API. since we are using the sequential API which is just a line of a stack of layers. a functional API is defined for building complex models. For the first layer, we used in our model which was from TensorFlow hub models URL. The first layer is an actual model that has many more layers the input layer takes in the images and find patterns in them based on the patterns found in the models we have chosen. the next layer which is the dense layer function Of Keras is the output layer of the model it brings all the information that was found in the input layer, together with the output to the shape which was chosen for the 120 labels, the method activation Parameter was set to be SoftMax which means we'd like to assign a probability value for each of the 120 labels between zero and one. I compiled the model using the model compile function with loss, optimiser, and metrics, the optimiser was set to Adam, metrics were set to accuracy and the loss function was set to categorical cross-entropy function. After running the mobileNet model I ran the summary function to check the values in it. The parameters which are the patterns led by the mobileNet model and the added parameters are the ones of the dense layer, which shows that the bulk of information in our model has already been run and we are going to take that and adapt to the problem we're trying to solve. I created the callback function Which are helper functions for a model that is used during training to save model progress, check the progress and stop running if the model is not improving its accuracy. an early stopping callback was created to prevent it from overfitting by stopping a model when the certain chosen evaluation metric stops improving. it can also do so well at finding patterns when A model trains for too long in certain datasets that have not been used to find the patterns.



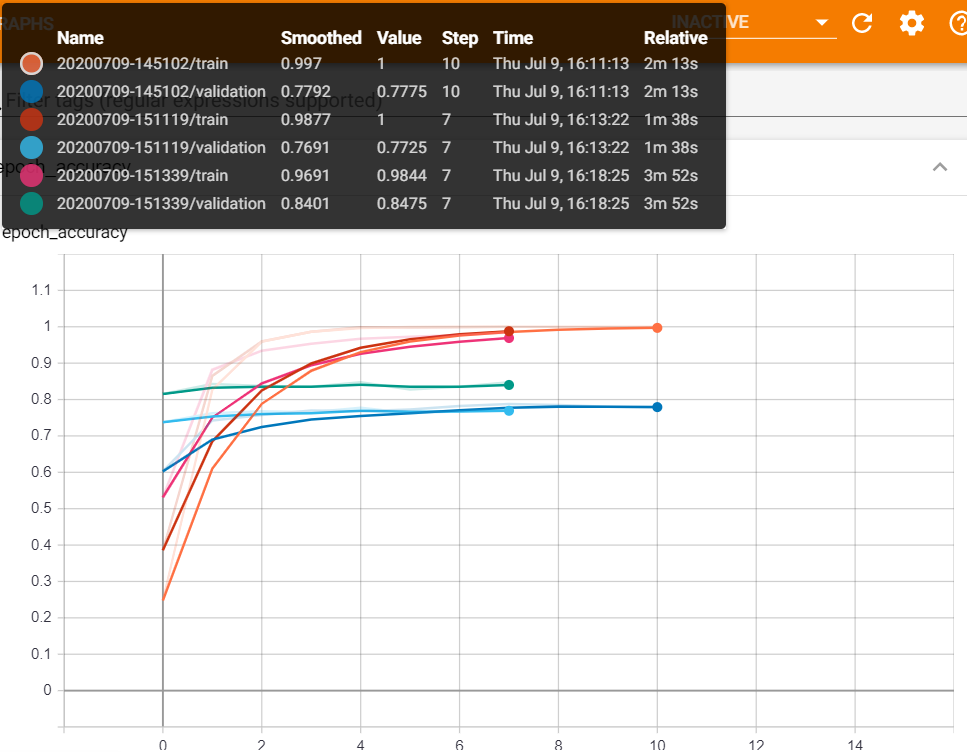
1. The result got from running the model summary function of the mobileNet\_v3 model.

The three models I am going to be testing to see if they are working by training them only 2000 images, which is using 1600 images as the training data and 400 images as the testing data. Setting The parameter for the number of epochs to be used which defines how many passes of the data would we like our model to do, which is our model trying to find patterns in the images that are related to the label. The number of epochs function to be used are not specific because during my research I found out that it is better to run a model, close to 30 epochs, so I created a slide from 10 to 100 and also using the Early stopping callback function Which means that the models will stop after improving.

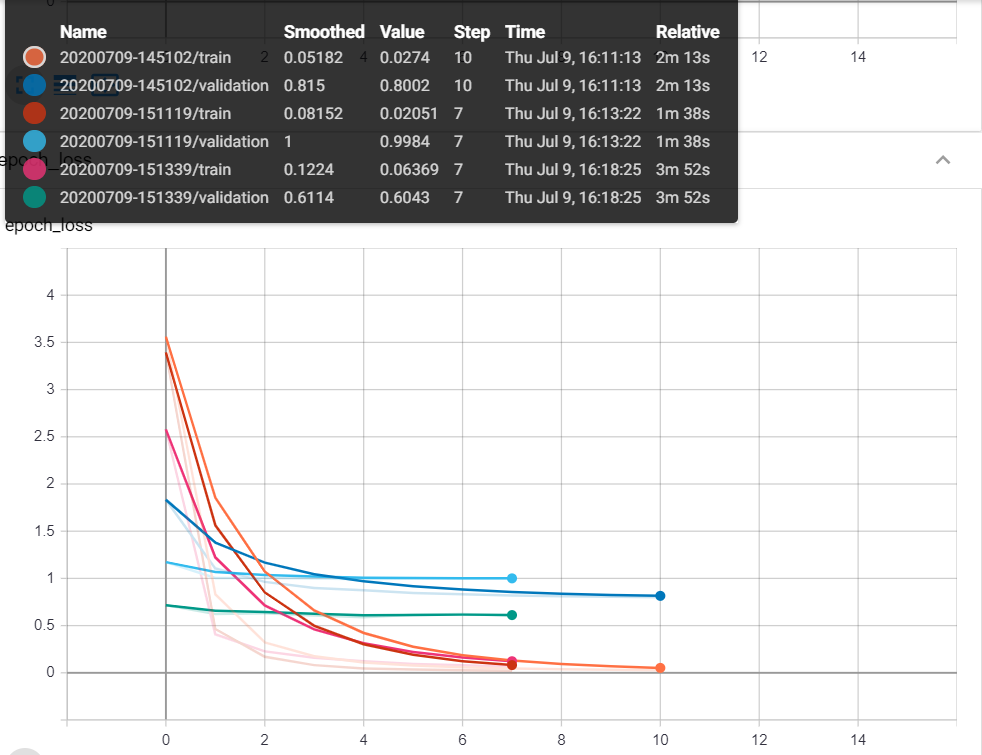
A model function was created, and tensor board callback was implemented in function and the fit function was also added to the function and returns the more given models’ accuracy. The three models showed signs that are overfitting was present. from the result gotten the three models, I found out that the inception\_ResNet\_v2 model performed better than the other two in terms of the validation accuracy. After training of the models, I compared it on the validation datasets and check the prediction probability, which is also the confidence level, and checking How well the model ran on the validation data. A function was created to visualize the results of the models on the validation of data and to see how well it predicts with its probability.

# Chapter 4 – results and conclusion

The figure below shows the epoch accuracy and epoch loss between the three models I need choose the number of steps it takes for each model to get to the point where the epoch stops working so for Models, mobileNet\_v2 it took 12 steps and for model inception\_v3 took 11 steps of a model inception\_resnet\_v3 took 5 steps. The epoch test diagram and chart was done on the training data sets of 1600 images. The aim is to minimise the epoch loss and to increase the epoch accuracy as we can see from the chart below the training datasets Seems to outperform the validation data sets which tells me there is evidence of overfitting in the model

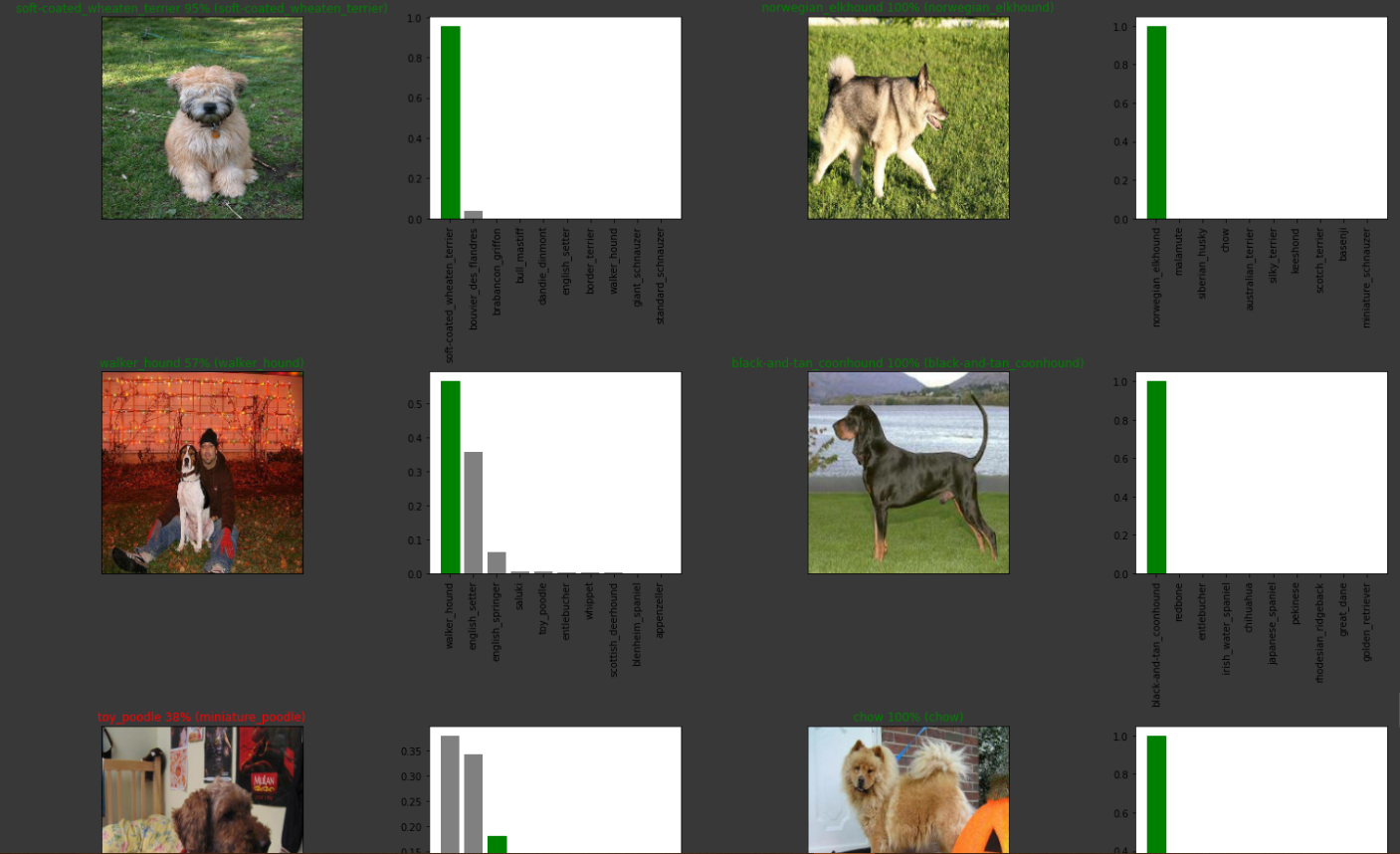


1. The chart of the epoch accuracy three models on the training and validation dataset.



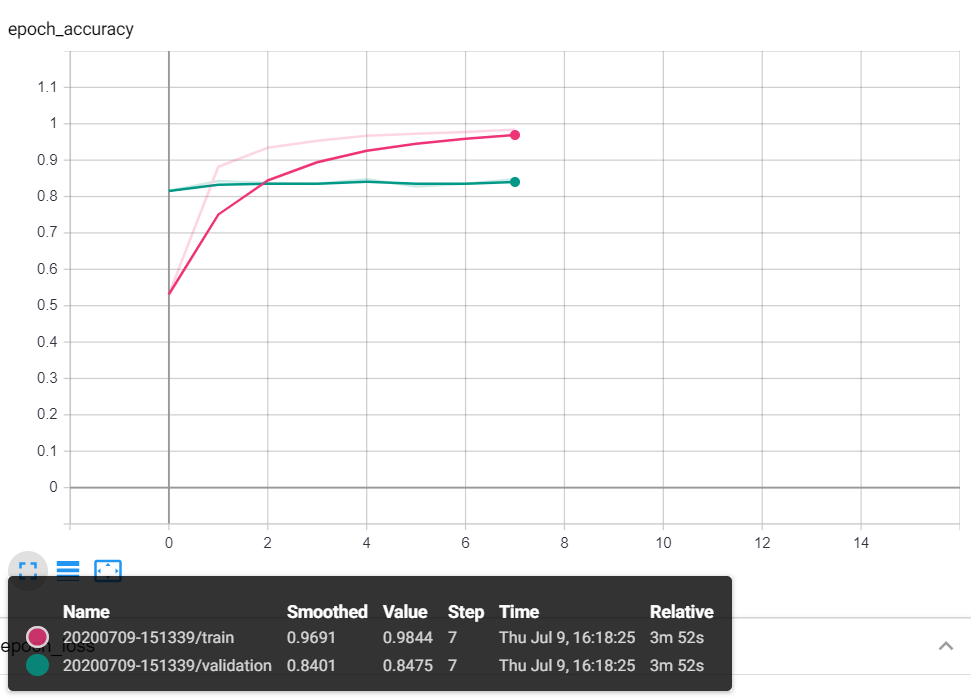
1. The chart of the epoch loss three models on the training and validation dataset.

One way I will try to see to solve the issue of overfitting is by running the chosen model on the full datasets. Selecting the inception Resnet model and running the validation dataset on it to predict a function Created to visualize the prediction and evaluate the model was created. In the diagram below we can see the percentage of the accuracy of the model on predicting the type of dog that is present in the image.

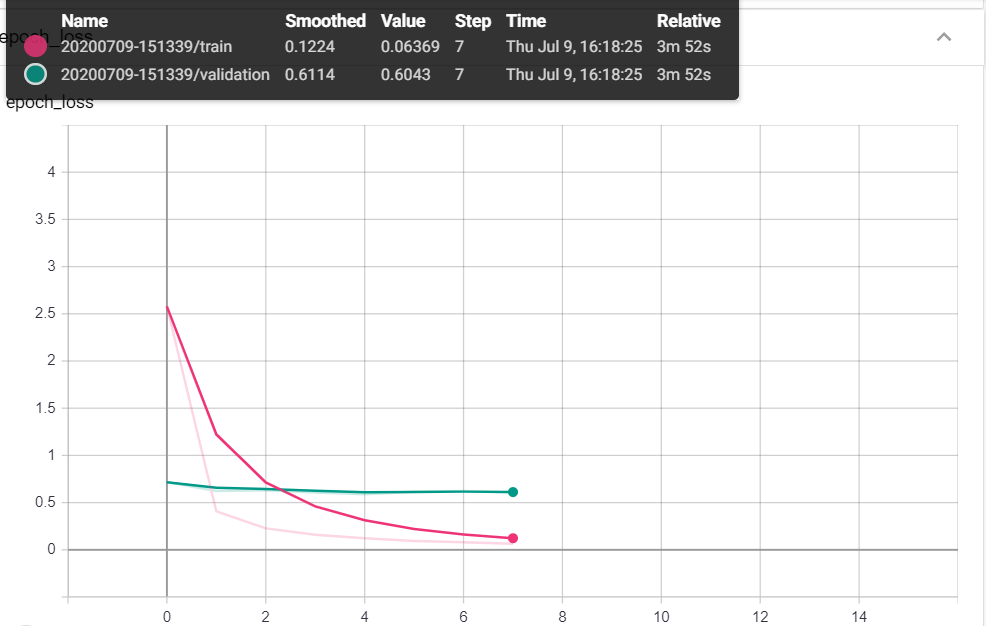


1. The diagram showing the performance of the inception\_resnet\_v2 when trained with 2000 images

From the image above if the Prediction is right the title will Green And if the position is wrong the title will be red The top 10 predicted labels as shown in the bar chart beside the picture where the first bar is the model's prediction and the rest are the top predictions. the predicted level is on the right while the true liberal is on the left from the results, we can see not the model got a couple of the predictions right. From the third picture below, we can see at the model got the prediction wrong. The model can be seen to perform well in predicting the names of the dogs in the pictures then the model is saved, and the entire model was run on the full dataset. the model that was trained on the 2000 images has an accuracy of 84% With a loss function of 0.06.



1. The chart of the epoch accuracy of the inception\_resnet\_v2 model on the training and validation data



1. The chart of the epoch loss of the inception\_resnet\_v2 model on the training and validation data

Both images below show the epoch accuracy and loss of the inception\_resnetv2 model that was run on the training dataset. It took a couple of hours for the final model to render.

# Conclusion

In this paper, we considered working with three Difference convolutional neural network by applying transfer learning to show how we can train And improve on a pre-train model that has already been trained on a dataset. The model improved after being trained on the full data sets on shows that are the inception\_ResNet\_v2 model. predicts far better than the other convolutional neural networks that were chosen. The model when ran on the full dataset with an accuracy of 96% and a loss of 0.1217 while it took 17 epochs to get to the required accuracy. Which shows that the architecture of the inception\_resnetv2 algorithm which utilizes residual connections instead of filter concatenations that is just the ordinary inception model implements in its algorithm. There are other ways to improve on a transfer learning model, like by fine-tuning it.

# Appendix (code)

[https://github.com/michaelajao/dog-image classification/blob/master/dog\_breed\_identification%20(1).ipynb](https://github.com/michaelajao/dog-image%20classification/blob/master/dog_breed_identification%20(1).ipynb)

# <https://colab.research.google.com/drive/1XfHIjmFZD6dYcRnNJluyKdjt0yl3gaKV?usp=sharing>

# References

Yung-Yao, C., Yu-Hsiu, L., Ming-Han, C., and I-Hsuan, Y. (2019) 'Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes'*. Sensors* 19 (9)

Simonyan, K. and Zisserman, A. (2015) 'Very Deep Convolutional Networks for Large-Scale Image Recognition'. arXiv.Org

Shiva Verma (2019) Multi-Label Image Classification with Neural Network | Keras [online] Available from <https://towardsdatascience.com/multi-label-image-classification-with-neural-network-keras-ddc1ab1afede> [7 July 2020]

Rohan Saha (2018) Transfer Learning - A Comparative Analysis [online] Available from <https://www.researchgate.net/publication/329786975\_Transfer\_Learning\_-\_A\_Comparative\_Analysis> [7 July 2020]

Le Cun, Y., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (n.d.) Handwritten Digit Recognition with a Back-Propagation Network [online] Available from <https://www.cs.rit.edu/~mpv/course/ai/lecun-90c.pdf> [10 July 2020]

Foody, G. M. and Mathur, A. (2004) 'A Relative Evaluation of Multiclass Image Classification by Support Vector Machines'. IEEE Transactions on Geoscience and Remote Sensing 42 (6), 1335-1343

Rosenblatt, F. (1958) 'The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain'. Psychological Review 65 (6), 386-408

Kim, W., Lee, H., Park, J., and Yoon, K. (eds.) (2005) Advances in Artificial. 'MultiClass Adult Image Classification using Neural Networks'. held Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg

C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (eds.) (2015) 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 'Going Deeper with Convolutions'

Jun-E Liu and Feng-Ping An (2020) 'Image Classification Algorithm Based on Deep Learning-Kernel Function'. Scientific Programming 2020

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014) How Transferable Are Features in Deep Neural Networks? [online] Available from <https://arxiv.org/abs/1411.1792> [8 July 2020]

Dr. Vaibhav Kumar (2020) Transfer Learning For Multi-Class Image Classification Using Deep Convolutional Neural Network [online] available from <https://analyticsindiamag.com/transfer-learning-for-multi-class-image-classification-using-deep-convolutional-neural-network/> [10 July 2020]

Sumit Saha (2018) A Comprehensive Guide to Convolutional Neural Networks — the ELI5 Way [online] Available from <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> [8 July 2020]

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018) MobileNetV2: Inverted Residuals and Linear Bottlenecks [online] Available from <https://arxiv.org/abs/1801.04381> [8 July 2020]

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2015) Rethinking the Inception Architecture for Computer Vision [online] Available from <https://arxiv.org/abs/1512.00567> [8 July 2020]

Sik-Ho Tsang (2018) Review: Inception-v4 — Evolved from GoogLeNet, Merged with ResNet Idea (Image Classification) [online] available from <https://towardsdatascience.com/review-inception-v4-evolved-from-googlenet-merged-with-resnet-idea-image-classification-5e8c339d18bc> [9 July 2020]

‌

‌

‌

‌

‌

‌