**Deep Learning with Keras & TensorFlow - Capstone Project**

**Skill**

Deep Learning with K...

**Sub Skill**

TensorFlow and Keras

Image Classification

Training a Model

1

**Instruction**

**Stanford Dogs Classification**

Welcome to your capstone project – Stanford Dogs Classification. You are required to build and train a model that classifies dogs from the Stanford Dogs dataset. This dataset is included in Tensorflow Datasets, and provides a challenging classifier that will test what you have learnt in this course.

In previous datasets, the images have been pre-processed to a certain degree, and can be used as-is. However, with this dataset, the dogs are not always centred in the image, there can be other objects such as people in the images, the images are not all the same size or orientation, and the dogs are not always orientated towards the camera. This makes this particular dataset tricky when it comes to building a classifier.

In addition to this, you will want to predict on images that are similar to the training images, for example with other objects in the picture, without the dog being centred and orientated towards the camera, etc.

**Core Features to build**

* A data pipeline (or generator) that loads the Stanford Dogs dataset and performs and pre-processing necessary to feed into a model
* A deep Convolutional Neural Network
* Training functionality, including tracking training performance
* Model evaluation functionality

**Your Task**

Your task is to build this model based on the details in this document and submit it. Please read the details carefully before attempting this project.

**The Dataset**

Although the dataset is provided in Tensorflow you have some decisions to make around it. First, it’s important to familiarise yourself with the dataset:

<https://www.tensorflow.org/datasets/catalog/stanford_dogs><http://vision.stanford.edu/aditya86/ImageNetDogs/main.html>

You will note that there are:

1. There are 12000 training images, complete with labels and bounding boxes
2. There are 8580 test images

You will need to decide the following:

1. Do you want to use the dataset provided by Tensorflow or the one on the Stanford website (both links above)
2. What is the split between train, validation and test?
3. Do you want to crop images to the bounding boxes provided, or not?
4. How to handle images with multiple dogs in them
5. Do you want to resize the images, and if so to what size?
6. Do you want to zero-pad images to the same size, for example this could be done where some images are orientated in landscape and others in portrait?
7. What augmentation is required
8. Do you need to normalise the pixel values?
9. Do you want to perform pre-processing on the images during training, or before the model is trained?
10. Do you want to weight dog breeds based on the number of training samples in the dataset?

**The Model**

When building a model, you will need to decide:

1. Are What is the architecture of the model, including how many convolutional and dense layers, what activation function(s) do you want to use. Or do you want to use a pre-trained model from Keras Applications?
2. What does the final output layer of the model look like?
3. What loss function is most appropriate?
4. What optimizer is most appropriate?
5. Do you want to use a pre-trained model, and if so:
   1. Which one will you use?
   2. Will you allow the base model or part of the base model to be trained, or set to non-trainable?
6. Will you train in a single pass, or vary the data over different training runs (for example using bounding boxes initially, and then using the original images in a 2nd pass)

**Training**

When training the model, you will need to decide:

1. How many epochs will you train for?
2. What is the learning rate?
3. What is the batch size?
4. Do you vary learning rate during training?
5. Which callbacks you wish to use, if any?
6. What training metrics will you capture during training to understand how your model is learning?
7. Are you going to save your model regularly during training, once at the end, or not at all?

**Evaluation**

How will you evaluate your model performance? This can be statistical as well as visualising the results of predictions where the model is both accurate and inaccurate.

It is advisable to use a confusion matrix to understand how well the model is predicting for each class. A good model will have a confusion matrix that looks similar to this with a clear line showing correct predictions from top left to bottom right:

**Submission**

Please submit the Jupyter Notebook ONLY. This should clearly show:

1. The code you have written
2. The output of the code
3. A comprehensive description of each code block, together with the decisions you’ve made and the rationale for those decisions.
4. A comprehensive description of what you also tried that did not work, and what lessons you have learnt from this capstone.

**Important Points**

Please keep the following essential points in mind when building and training this model.

* Use Python 3.7 – 3.10 (officially supported by Tensorflow)
* Use the latest official release of Tensorflow, at the time of writing this is v2.9.1
* Use an appropriate hardware platform. Google Colab is sufficient for this purpose, or a powerful PC (preferably with a GPU) if you have access to one. A standard laptop is unlikely to be powerful enough to train this model in a reasonable timeframe and prolonged training can lead to hardware damage due to heat build-up.
  1. If you need to install Tensorflow on your local PC, you can use the official instructions here <https://www.tensorflow.org/install>
* This is a challenging dataset to predict accurately on, so iterate your approach over time making note of what works and what doesn’t. In the long term, this is as useful to you as getting a high model accuracy.
* Plagiarism will lead to disqualification of the submitted project. The objective of this exercise is to give you a real-world project to build and we hope you will use this opportunity wisely to your benefit.