PGDip Applied Data Analytics Project Interim Report

Terry Dunne

D00147664

24/02/2025

1. **Background/Introduction**

The problem to be tackled is to build a model that can take the 10 breeds of informational data of images provided in the dataset and be able to take user input and make predictions on their provided image.

Some questions we are going to touch on is what the best tools/framework are to build and distribute the model, what effect does image quality of the training images have on the outcomes of the model and if the model can be accessed in a PWA rather than a traditional app.

Image classification is a hugely growing field for data analytics as it allows for a way of interpreting visual data and allows for pattern to be follow and trends to be seen. Improvements in this can be a huge benefit to public whether it is in medical diagnosis, identifying poisonous animals/plants.

The project will touch on computer vision and machine learning to build the model. As it is not a massive dataset, I have been looking into TensorFlow as in the future this kind of lightweight model would be easier to transfer to another environment where any potential users can easily access and make use of the model.

My solution would take the form of building and training of the model and then making use of the compatibility of tensor and certain web environments to make the model available to all users through a web portal of some kind.

2. **Literature Review**

Building an Image Classification Dataset

In classification tasks, a key quality indicator of a dataset is the separability of the data points. When data points can be sectioned off with similar points, the model can easily learn to classify them together and identify them. However, if the data points are not well-separated and are scattered instead, the model will struggle to distinguish between classes and fail at classifying them. For the imagewoof dataset that was given the ten different dog breeds are already separated into different folders, so no further action needs to be taken.

In most cases, having a large training dataset improves model performance by enabling it to learn more robust representations for the computer vision task. For this dataset, the lowest number of images for a breed is around 1300 and the highest are around 1750. Some guides for building a model refer to having at least one hundred images per classification you want, which is a positive this set is above. Also, there is guidance regarding image size and most to recommend downsizing their images to 224x224 or lower while building the model and ingesting input from potential users.

There are functioning apps that to identify dog breeds from a photo taken/uploaded. There are also webpages that allow for the same thing. A potential limitation of these is people might not want to have a dedicated app on there phone for this one specific task. If it was a site, you could just visit quickly on your phone as needed it might see more use.

The Rise of Progressive Web Apps (PWAs)

Progressive Web Apps (PWAs) are web-based app designed to function similarly to native mobile apps, them a to provide a seamless and friendly experience to the user. Unlike traditional apps that require a download and installation, PWAs allow the user access to the content/app right away through the browser. This eliminates the need for app store, making the app more accessible to users and avoiding the need to service different mobile operating systems.

Google specifies three main features of PWAs:

- **Reliability** PWAs should load instantly, regardless of whether a user is online or offline.
- **Speed**: PWAs should load in fewer than three seconds, and the content should be immediately available without any delays or scrolling issues.
- **Engagement**: PWAs should have engaging content and give users the power to add the app to their home screens. PWAs can also send push notifications to users without the user having to download a specific app to receive them.

Ultimately, a PWA aims to mimics the behaviour and features of a native phone app but operates entirely within a web browser. This approach reduces any potential storage burden on users' devices while still aiming to deliver a high-quality, app-like experience.

Training and Validation Accuracy

When building an image classification model, tracking the training and validation accuracy and loss is essential for assessing performance and ensuring the model generalizes well to unseen data. These metrics provide valuable insights into whether the model is learning correctly or facing issues like under/overfitting.

Accuracy refers to how many images the model correctly classifies out of the total number of images. While accuracy is a useful metric it does not indicate the types of errors the model could be making, such as false positives or false negatives. Loss refers to the model's errors by measuring how far its predictions deviate from the given labels. A lower loss indicates that the model is making better predictions on the data, a higher loss suggests poor learning from the model. These metrics are calculated separately for training data in which the model learns from, and then validation data which is used to assess its generalisation ability.

Monitoring accuracy and loss is crucial for detecting any potential under/overfitting. Underfitting occurs when the model is too simple to capture patterns in the training data, leading to low training and validation accuracy with high loss. To address this, one can use a different architecture, increase training time, or add more data to the training.

Conversely, overfitting happens when the model memorizes training data instead of learning and identifying the patterns present. This is shown when training accuracy is high, but validation accuracy remains low. Techniques like data augmentation, dropout, and regularisation can help address overfitting and improve generalisation.

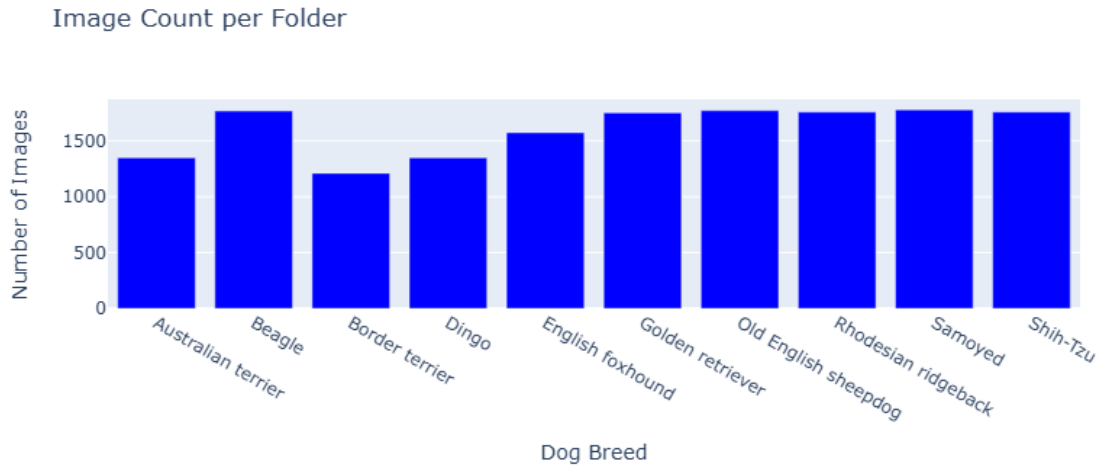
For optimal performance, training and validation accuracy should both increase while loss should decrease over time. If validation loss rises while training loss drops, it suggests the model may be overfitting. Tracking accuracy and loss enables fine-tuning of the parameters and model architecture, ensuring a well-performing image classification model that generalizes effectively while avoiding common pitfalls.

3. Exploration of Data and Methods

The dataset provided by and was made available through an extractable file through imagenette. It is a publicly available dataset, and no other data collection is needed. The dataset came already split into two separate folders for training and validation. Inside each folder has a lot of pictures for every breed.

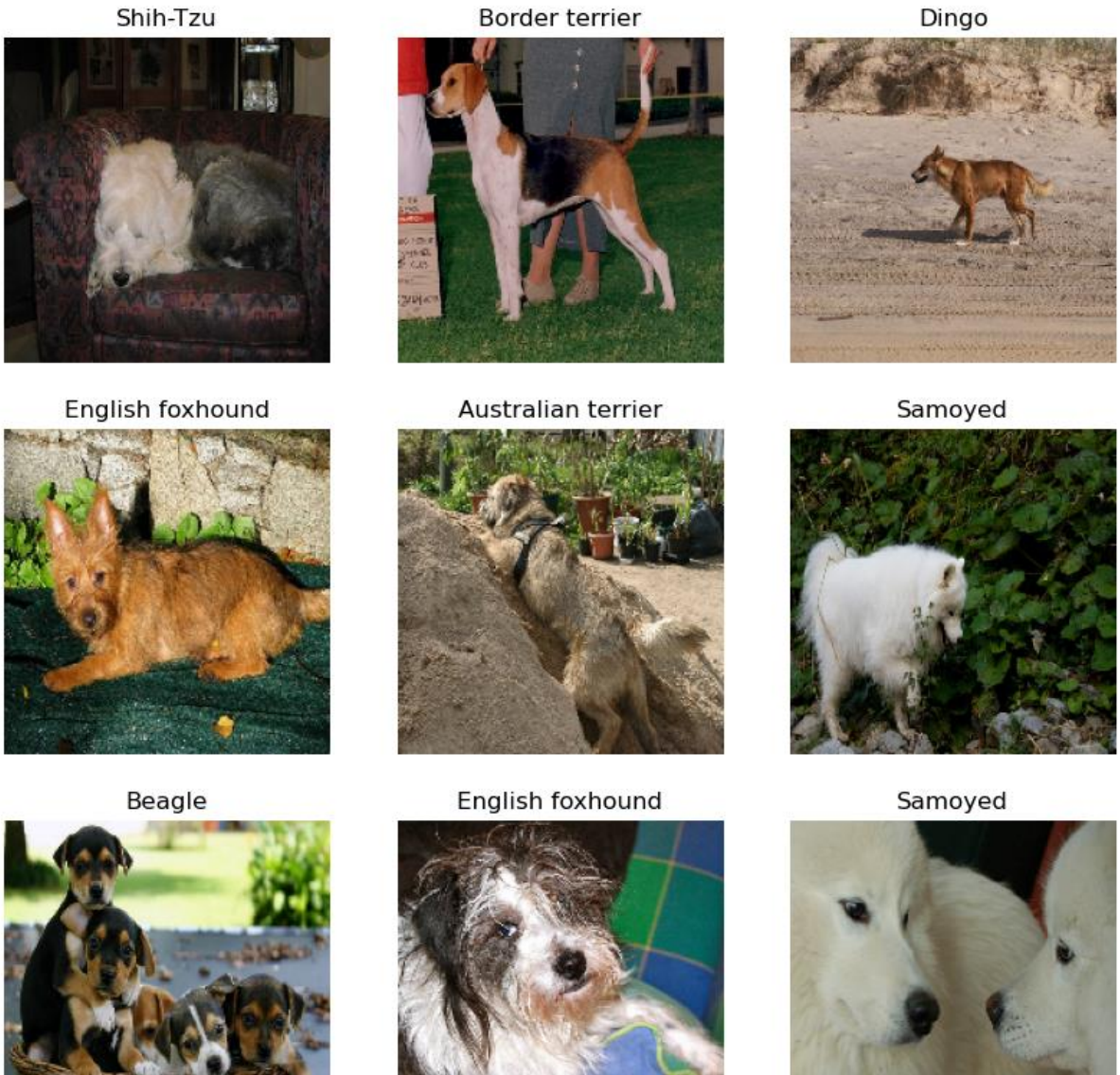
Looking at how the data was extracted each breed was already split into separate training and validation folders. As the model is going to be built from the ground up, it is best to consolidate these into one folder, so any rate or percentage wanted for validation is not predetermined and configurable as the model is being built. Looking at the images in the training and validation folders there is no issue in merging images into one folder.

The breeds are in folders with non-descript name like 'n0345345'. For readability and any future issues encountered I am going to rename these to their actual names.



6 breeds have 1750+ images, the other 4 may have potential issues/not be as accurate as the others, good thing to earmark and investigate further after the model is fully built if these breeds will be as dependable as the others.

For the initial exploration of the data, Keras has an image classification from scratch guide that will be followed. The guide example is a binary example but looking at documentation changes will be made so that it is working for more than 2 inputs, increasing to the 10 dog breeds present in the dataset.



A sample of the random images from the imagewoof dataset highlighting the different breeds

The guide adds in some layers of data augmentation which should help the model be exposed to different aspects of the same training.

Moving on to the resulting output of building this model and training.


Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Training and Validation Loss
Training Loss
Validation Loss


Border terrier (42.03%)


Dingo (47.11%)


Border terrier (75.45%)


Dingo (60.78%)


Dingo (52.93%)

Although the accuracy and loss are trending in right direction, the graphs of these are very atypical and would suggest that possibly something out of the ordinary is occurring. With the results shown from some test images the model is not correctly identifying the breeds. The simple tweak from the Keras tutorial was not enough to transfer over to make the model complete and working. The model needs to be built in a different way to achieve the goal of this project.

A different approach is needed to better tackle the ten breeds in the dataset, more option needs to be accessed and considered in building and training the model.

4. Proposed Future Analysis

The goal would be to have an interface available to users in a web environment where they can upload an image easily from their device and have the model run on the image and give them the result at a quick pace. The model would be hosted on a service and allowed to be accessed from a portal of some kind. The result should show the amount of match the user's picture has to the breeds available in the set.

First step would be to have the model completed and accurately identifying the breeds. Changes are needed to allow the model to function correctly, whether it is altering learning rates or adding dropout. Once the model is functioning at a high percentage the next step would be to find a way to host this model somewhere it can be accessed outside of the dev process. From the documentation Keras and TensorFlow provide, it is very compatible to being hosted accessed through other framework JS frameworks and should be fit for the end goal of the project. The last step being able to have someone interface with the model from this new location and making sure it is possible to have the model run remotely.

References

levity.ai. (n.d.). *How to build a dataset for image classification*. [online] Available at: https://levity.ai/blog/create-image-classification-dataset. [Accessed 20 Feb. 2025].

Acharya, A. (2023). *How to Choose the Right Data for Your Computer Vision Project*. [online] Encord.com. Available at: https://encord.com/blog/choose-the-best-data-guide-computer-vision/ [Accessed 20 Feb. 2025].

Smith, C. (2024). *The Rise of Progressive Web Apps (PWAs) - OuterBox*. [online] OuterBox. Available at: https://www.outerboxdesign.com/digital-marketing/progressive-web-apps-pwas [Accessed 20 Feb. 2025].

Yu, Y. (2022). Deep Learning Approaches for Image Classification. doi:https://doi.org/10.1145/3573428.3573691.

vl, F. (2023). *Interpreting Training/Validation Accuracy and Loss*. [online] Medium. Available at: https://medium.com/@frederik.vl/interpreting-training-validation-accuracy-and-loss-cf16f0d5329f.

Team, K. (n.d.). *Keras documentation: Image classification from scratch*. [online] keras.io. Available at: https://keras.io/examples/vision/image_classification_from_scratch/.