Research Project Report

Classifying the coat of cats using both categorical features and visual features

Devon Hope – 101038344

Yichen Dou – 101003897

Prof. Alan Tsang

COMP 4107 B, Winter 2021

Apr 23, 2021

Repository: https://github.com/Yichen-Dou/W21-COMP4107-Project-Grootslang

1 Introduction

1.1 Research Question

How neural networks help predict the coat of cats using both categorical features and visual features.

1.2 Background

Classification problems are a popular topic in areas of machine learning nowadays. The neural network is feed with several images and outputs a specific label or labels for classification. However, what if we have extra information about the labels? For example, if we want to predict the price of a house based on an image and numerical values such as the number of rooms, levels, and location, there is a problem that numerical, categorical, and image features are trained well on different neural network architectures respectively. In this research, we will make a neural network with two input layers, one is for categorical features, and another is for images when predicting the coat of cats.

1.3 Datasets

Cat Breeds Dataset from Kaggle is a dataset which contains a csv data file and a bunch of images. The csv data file includes categorical features including age, gender, breed, coat and size of a cat, where each instance belongs to a single cat with related images.

1.4 Machine Learning Libraries Used In The Research

1. Tensorflow: A well-known open-source library for machine learning.

2. Keras: An open-source library that provides extra features to TensorFlow. It offers interface users can access to make artificial neural networks with many toolkits.

3. Scikit-learn: A free machine learning library that offers many algorithms in data analysis and data processing.

2 Methodology

2.1 Data Analysis

All data formatting can be found in ‘Analysis.ipynb’.

2.1.1 Missing Values

Several rows of cats were missing breeds, and the number of images per breed was severely uneven. The number of images per class type is important, if the number is uneven, or misrepresented, the produced results could be skewed and the accuracy of the results would be false.

2.1.2 Correlation

We used correlation graphs to determine which classes (i.e., coat, breed, age and gender) shared cats and for what reasons. It is commonly found that size and coat are related as well as gender and size, this makes sense on the surface as the larger a cat will appear has direct correlation with the size of its coat, as well as its gender.

2.1.3 Uneven Categories

Some classes contained very few categories, such as coat, where the total number of coat types was four, in most cases however, there were only three coats, thus the number of cats was limited to three types of coats. The same can be said with breeds, some breeds contained fewer than 500 image samples, our baseline for accurate testing, therefore they needed to be cut from the sample size, limiting our input again.

2.2 Data Preprocessing

2.2.1 Image Resizing

When we load images from paths stored in the data file, we need to make sure all images should have the same dimension. Moreover, for a neural network model can successfully predict the coat of cats, the network should be present enough detail of images, such that patterns like hair edges can be detected. After many trials on a standard convolutional neural network, we found that resolutions above 128 pixels of width and 128 pixels of height give us an optimal result at an acceptable loss.

2.2.2 Remove Instances That Contain Null Values

Since the features we will train for the neural network are all categorical features, unlike numerical features, we may not fill empty spaces with mean values of a feature in this dataset. It is better for us to drop instances that contain null values in this case.

2.2.3 Remove Labels That Contain Few Instances

In our data analysis, we investigated several instances in each category of the coat. Then we determined that there are only 60 instances in hairless cats, comparing to more than thousands of instances in other coat categories. With a lack of data, the neural network may not learn its associated features well, so instances about hairless cats are removed.

2.2.4 One-hot Encoding

Our labels were represented as text, but the neural network needs them to be encoded before it can understand. Since our research problem is about making a single-label and multi-class classification network, we use the one-hot encoding as our technique to encode the text of coats.

2.3 Neural Network Model

Model diagrams can be found in the Models folder under ‘final_model_1’ and ‘final_model_2’ for the complex structures.

2.3.1 Basic Model Structure

The first model that was implemented was the basic CNN structure using the RMSprop optimizer from Keras. This model produced inaccuracies and high loss values, but it also showed that the data was being skewed in our initial testing of the breed and coat classes. The preprocessing that had been done was for the total number of breeds and coat types, unfortunately we had to cut some of these breeds and coats as the number of images for specific categories in each class was less than desirable, leaving us with 26 breeds and 3 coats.

2.3.2 Complex CNN Structure

The complex deep learning CNN was made with dual input, from two separate models that are combined together. The first defines the density of each layer, and the second model defines the type of convolution with a series of convolutional layers, maxpooling layers, a flatten layer and a dropout layer. The dual input CNN produced the best accuracies and losses, nearing 70% accuracy in most tests, and losses below 0.7. There are two versions of this structure, the main difference is the optimizer, the first, model_1, uses the standard gradient descent optimizer, or SGD, and the second uses the Adam optimizer. The first model measures loss with binary cross entropy and the second with categorical cross entropy, the difference is notable in the tuning that is needed to stop overfitting with the learning rate and decay for Adam optimizer.

2.4 Training, Testing and Validation

2.4.1 Early Testing

Initial testing began with the standard CNN at 20 epochs with a learning rate between 0.01 and 0.05 and the RMSprop optimizer. The results showed inconsistencies in our data formatting methods as well as the dataset itself. This was useful for developing the final combined neural network structure, if these errors in our data had not been found, the final network would not train as well, and the losses would be much higher. As you can see below, the first attempt at training with our standard CNN shows heavy overfitting.

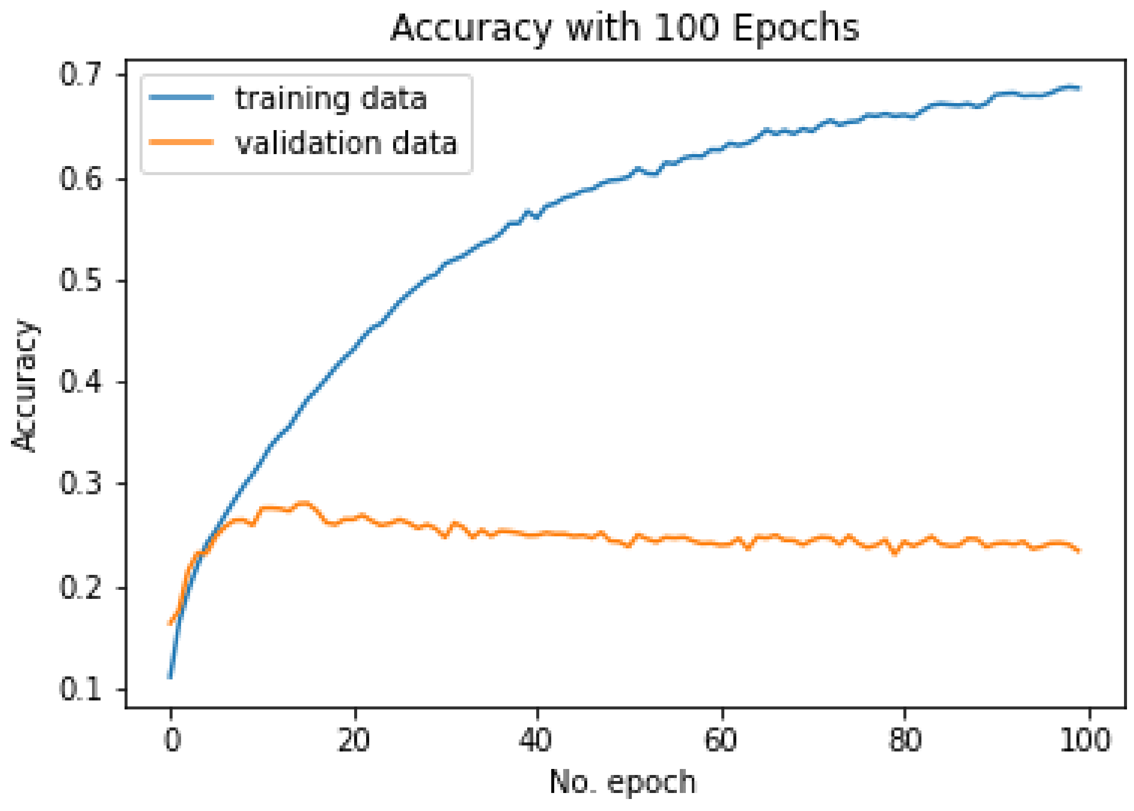


Figure 1/Standard CNN

2.4.2 Testing

Testing took place with the final combined CNN, the more complex one, over several iterations of adjusting dropouts and layers to get the CNN to train properly. Testing the CNN encapsulates training and fitting properly for our desired accuracy and loss values. When testing the complex CNN's, we encountered a few issues, overfitting, low accuracies or stagnant accuracies and high loss values. Over several iterations of the complex CNN model, we determined that testing the single input version of the complex CNN was not fitting or training properly with the breed class alone. When switching to the coat class, determining a cat by their coat value, we obtained more accurate results, but with fewer categories in this class, three, training the model would only be very inaccurate for real world use. This is where the idea of the dual input layer complex CNN came from, by training the network with more than one class type, we could identify a cat, their breed, age, gender and coat with more precision.

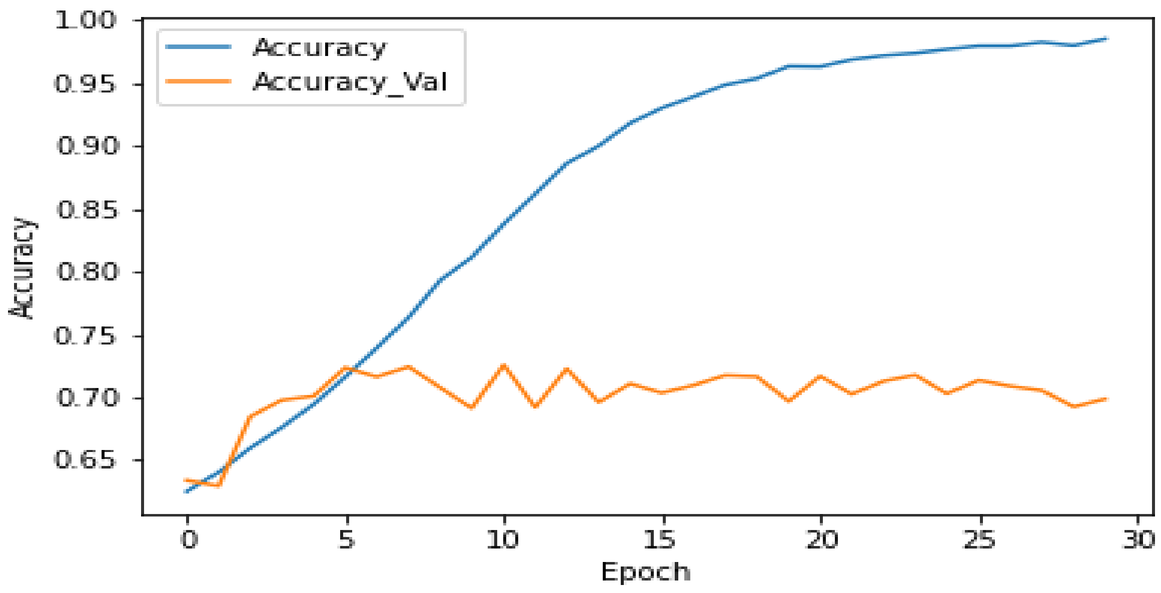


Figure 2/Single Input Complex CNN with coat class & untuned

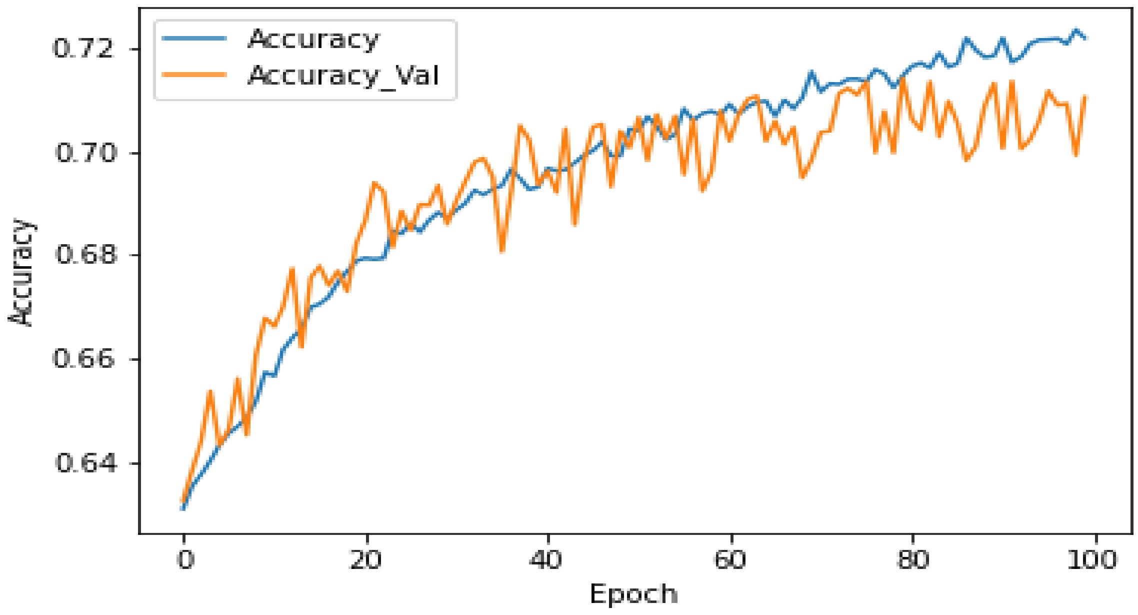


Figure 3/ Single Input Complex CNN with Coat Class & tuning

2.4.3 Training

Training model_1 of the complex dual input CNN's resulted low but stagnant accuracies and low losses, it was clear that more tuning was needed to train the network properly. When training model_2, of the complex dual input CNN's, the accuracies were steadily increasing but would stagnate after they reached 70%. This showed us that the learning rate could be adjusted further and the dropout values of both CNN's were too high. The follow up can be found in the next section, hyperparameter tuning. The range of epochs for training was limited by our combined compute power of our GPU's, from 10, at testing, to 100 for training. Each step would approximately take up to a minute depending on the batch size and resolution of our formatted images.

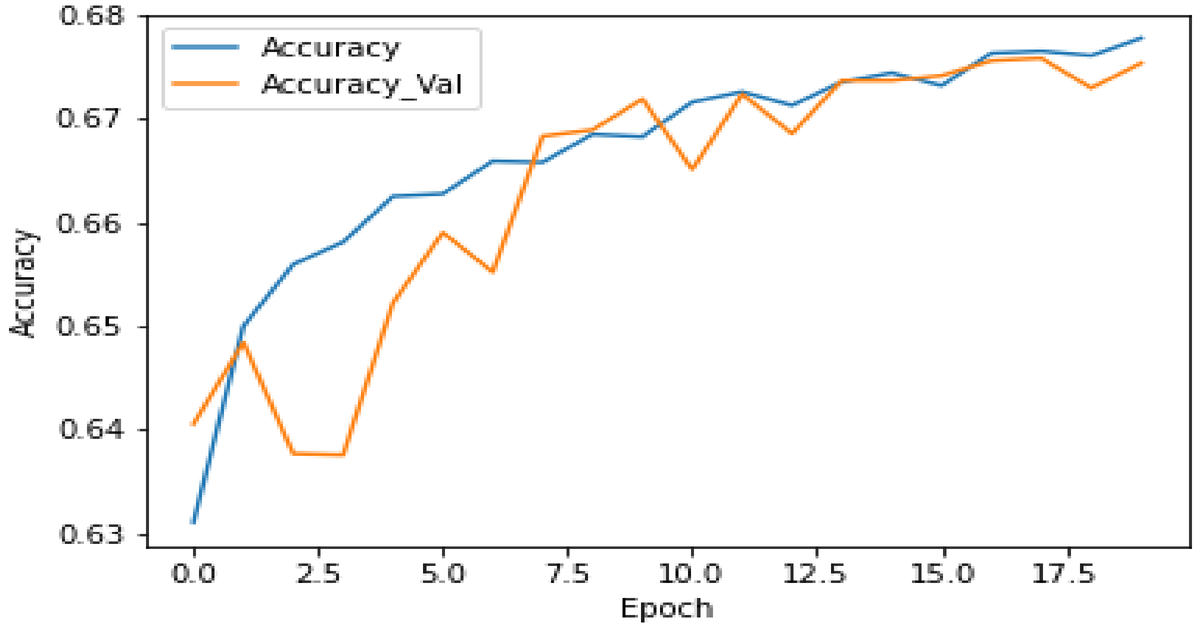


Figure 4/ Model_1 after 20 epochs

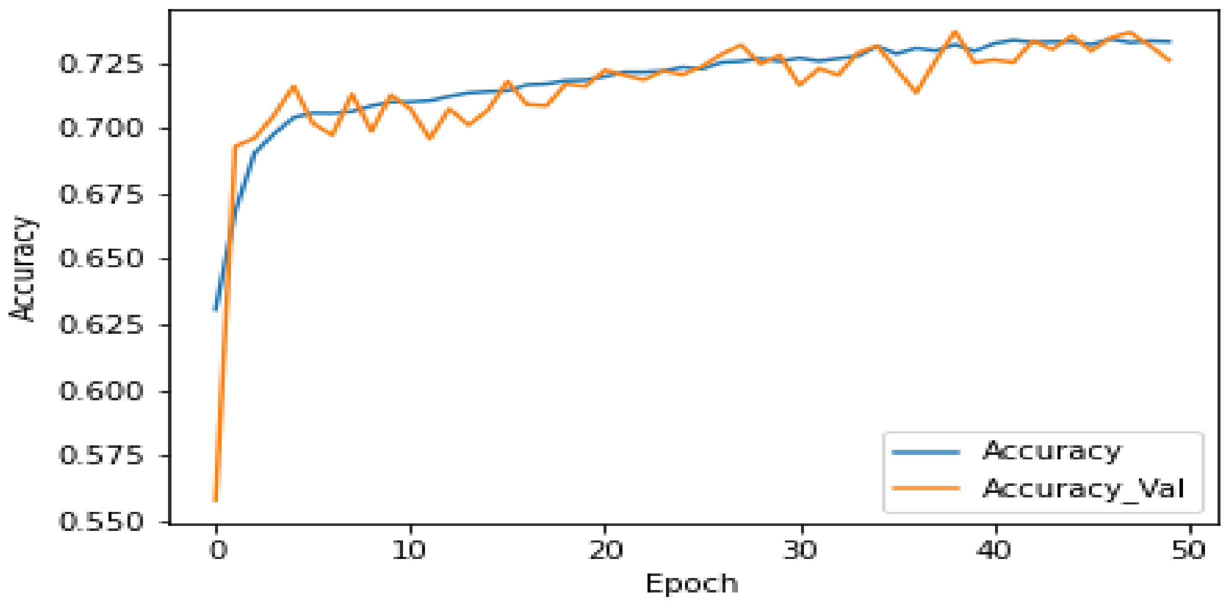


Figure 5/ Model_2 after 50 epochs

2.5 Hyperparameter Tuning

For research purposes, we firstly make some networks individually for either categorical features or visual features rather than making a combined network directly. After we find models giving good results, a combined neural network will be made.

2.4.1 An ANN Network For Categorical Features

In early experiments, we only add few layers to the network, usually one to two hidden layers containing few neurons, then we get a result at a high loss after many epochs. After that, we increase the number of neurons and hidden layers to test if the network can contain enough information about features. The Adam, a stochastic gradient descent method based on adaptive learning, can generate an optimal result without overfitting or underfitting for categorical features. Including the number of neurons, the number of layers, activation functions, learning rate, and momentum, we try tuning many hyperparameters to get a better result as much as possible.

2.4.2 A CNN Network For Visual Features

Similar to what we do in 2.4.1, we start from a small convolutional neural network to a more complex convolutional neural network that contains a rich number of parameters. During the research, we also confirm which resolution of the images should be picked. In early experiments, losses range from nearly 0.9 to 0.6 after many epochs. We tuned the number of neurons, number of layers, learning rate, activation functions, weights initializers, kernel sizes for convolutional layers, dropout ratios, and max pooling sizes. At the end, we choose the stochastic gradient descent (SGD) as our optimizer method with tuned hyperparameters.

2.4.3 Combined Network For Both Categorical Features And Visual Features

When it comes to a combined network taking both networks from previous research, we face many challenges. In this case, we firstly experiment many optimizers with their hyperparameters including batch size, epochs per step, momentum and decay rate. Then we look back to tune the two connected sub-networks like what we do in 2.4.1 and 2.4.2 to try to get a better accuracy and loss for the validation data.

<u>3 Results</u>

- Model_0 (categorical features only) after 50 epochs with a learning rate of 0.001

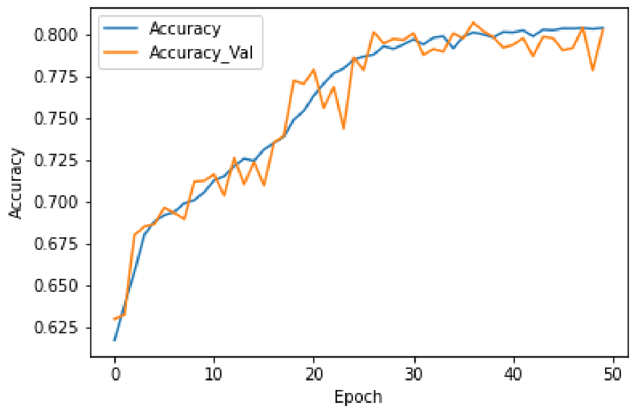


Figure 12/ Model_0 ANN Accuracy

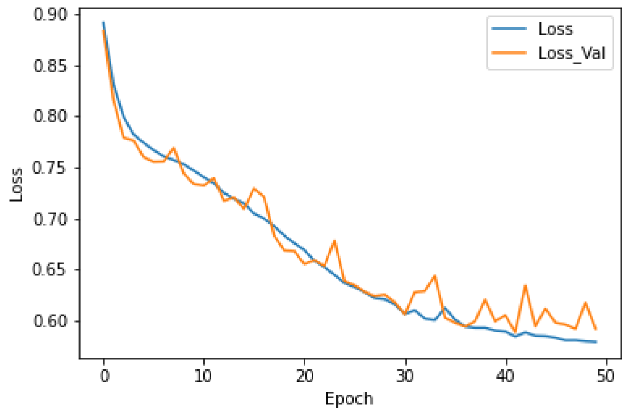


Figure 13/ Model_0 ANN Loss

- Model_0 (combined net) after 25 epochs with a learning rate of 0.025

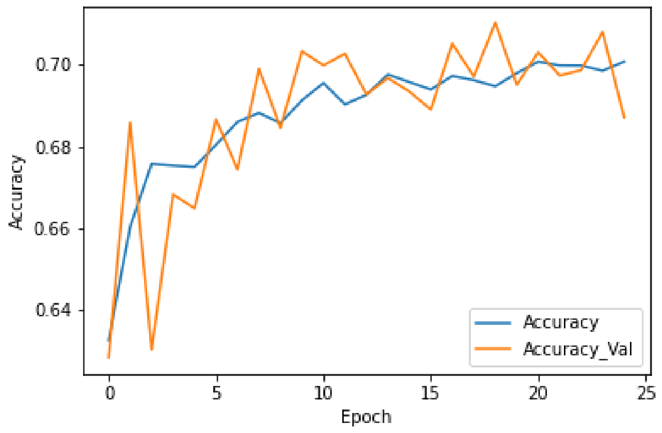


Figure 14/ Model_0 Combine Accuracy

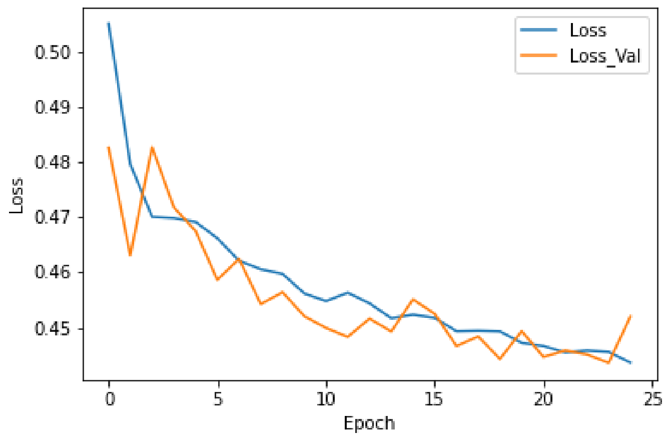


Figure 15/ Model_0 Combine Loss

- Model_2 after 50 epochs with a learning rate of 0.004

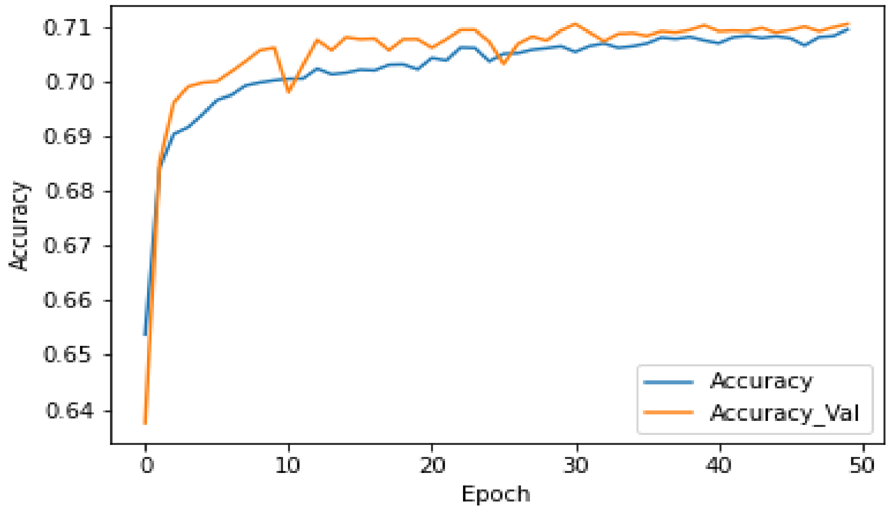


Figure 6/ Model_2 Accuracy

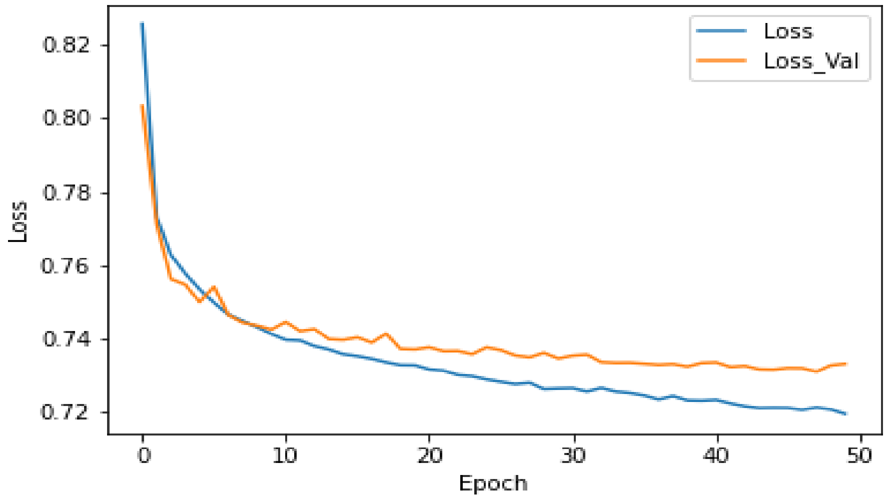


Figure 7/ Model_2 Loss

- Model_2 after 50 epochs with a learning rate of 0.015

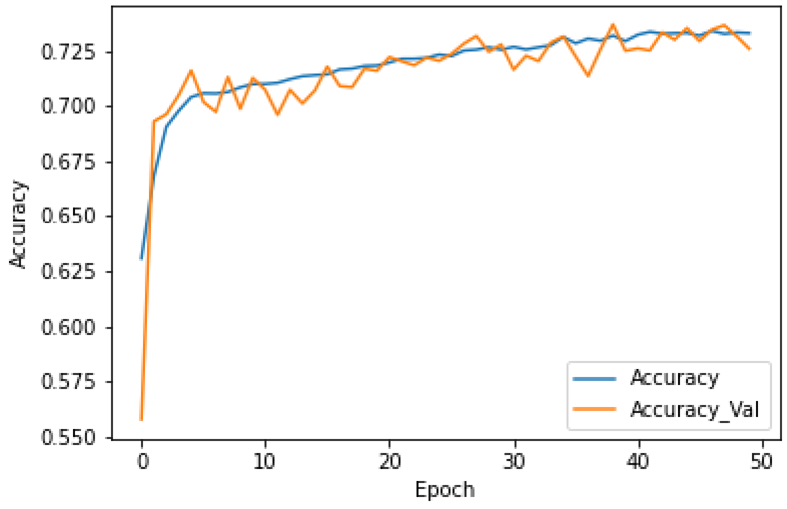


Figure 8/ Model_2 Accuracy

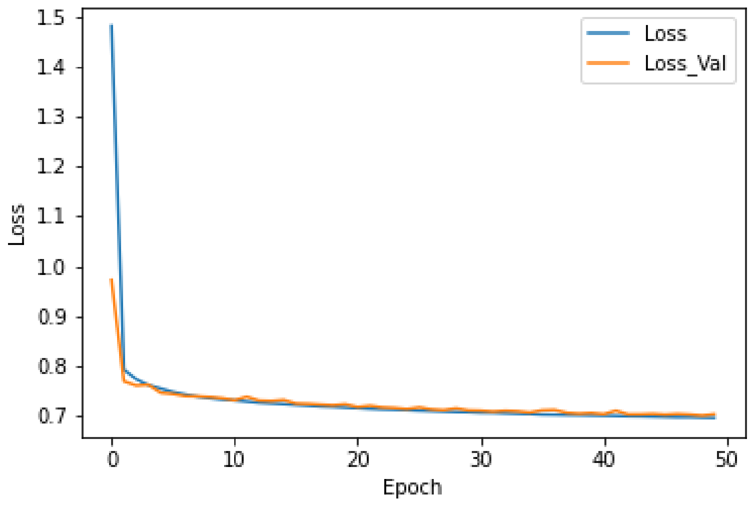


Figure 9/ Model_2 Loss

- Model_2 after 50 epochs with learning rate of 0.005

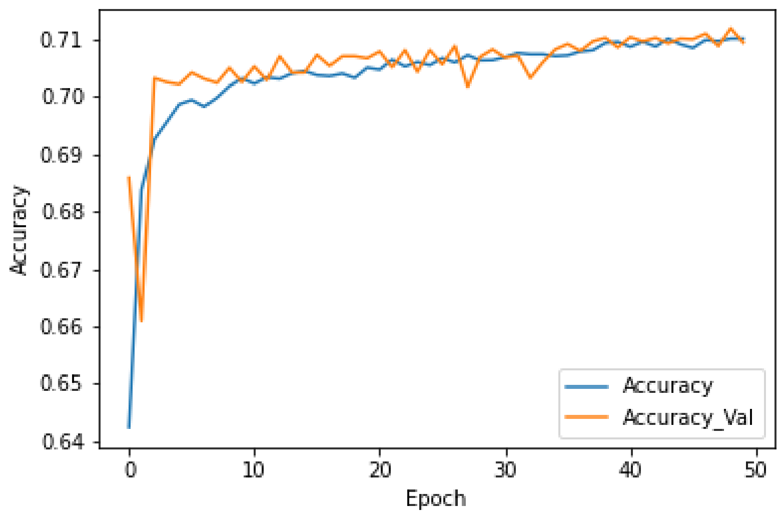


Figure 10/ Model_2 Accuracy

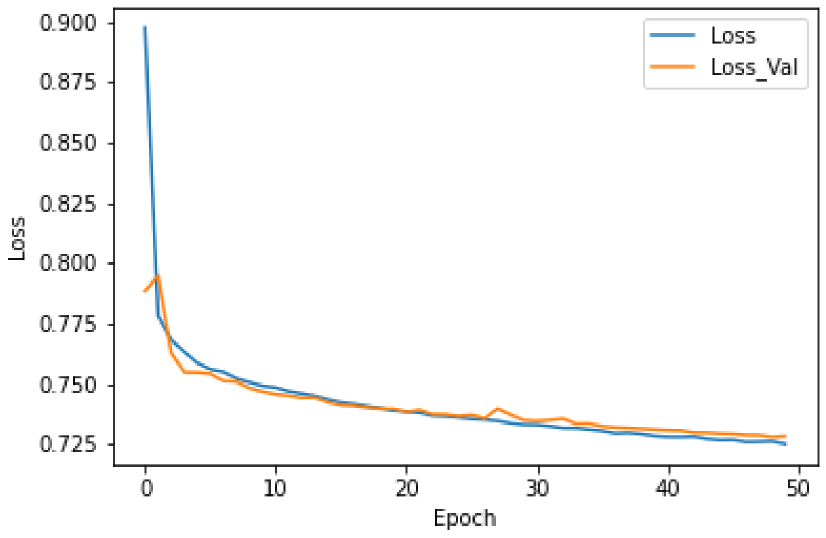


Figure 11/ Model_2 Loss

- Model_final_0 after 20 epochs with learning rate of 0.1

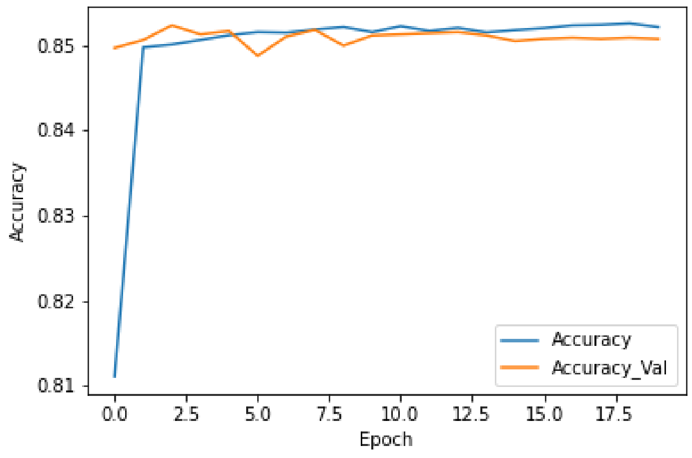


Figure 17/model final combine acc

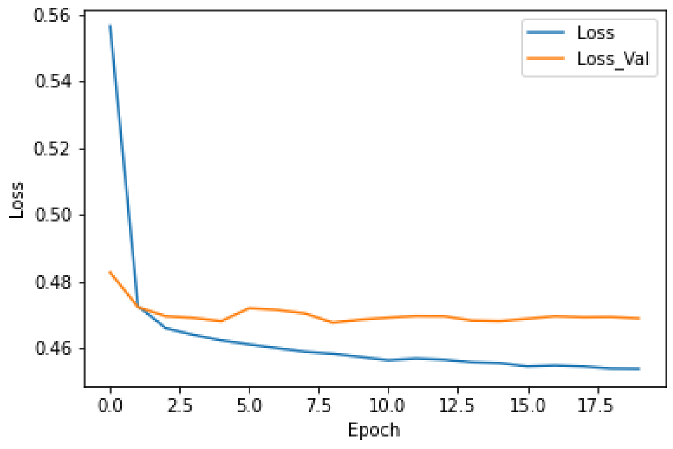


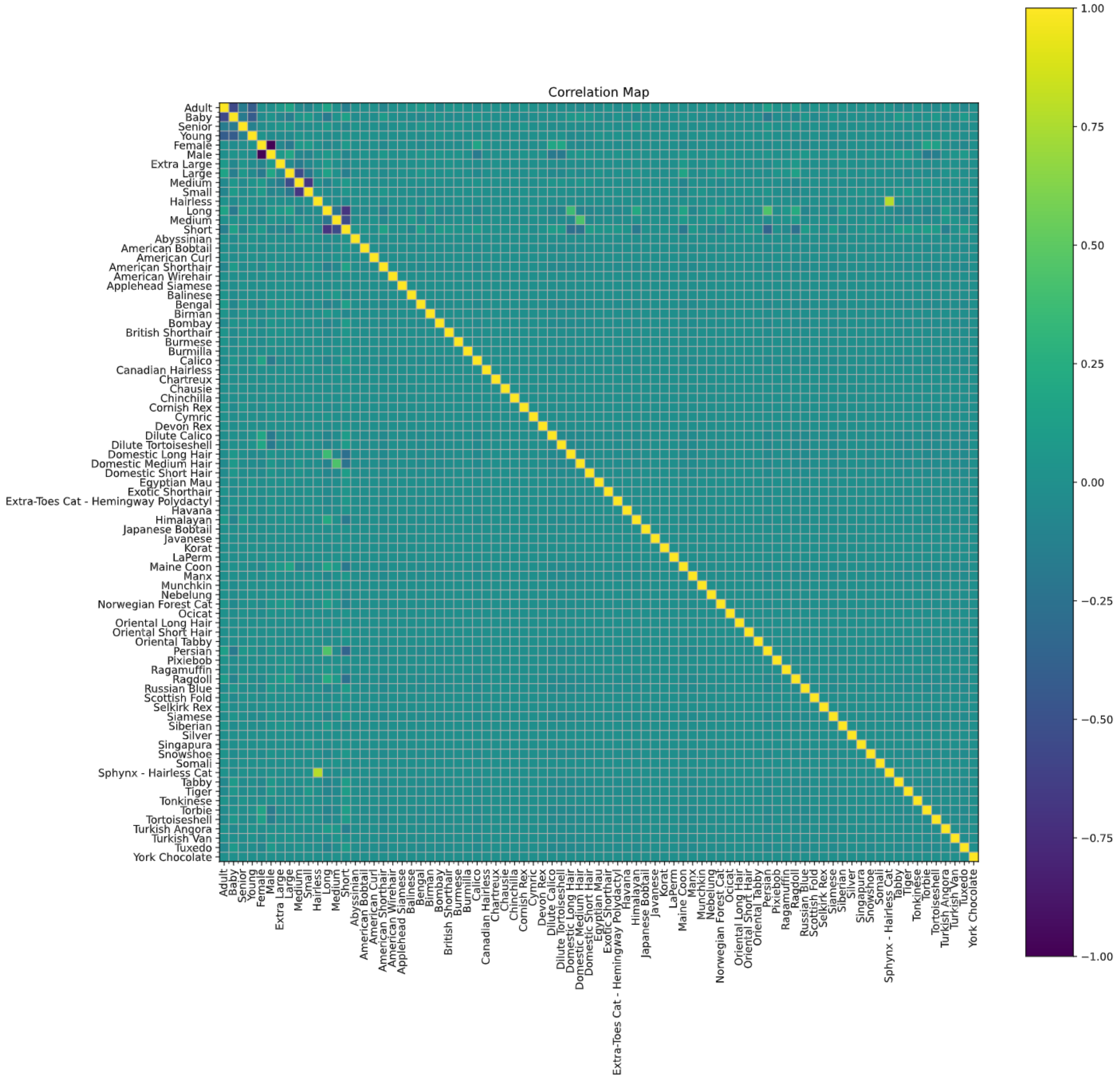
Figure 18/model final combine loss

4 Discussion

Based on our results from the several models that were tested, the more complex dual input CNN's seemed to work the best, however, they were limited still as their optimizer and structure could have been better suited to produce higher accuracies. Our personal computer hardware we used to train the networks was also a key part in the end product, if we had more time the final combination network could be better optimized for our personal hardware. The results as they are, seem fitting for our attempt, reaching above 70% accuracy for detecting four distinct features of a cat. Many other methods could have been used to better predict a cats features but this was the closest we came to producing a viable effort.

5 Conclusion

When we only look at the correlation map of every categorical feature in a one-hot encoded format, we notice that there truly exist correlations between some features, for example, the breed "Persian" has a positive correlation with the Coat "Long", and the age "baby" has a negative correlation with the Size "Large". If we only serve the neural network with images, this information in the data file may not be captured to increase the chances of predicting the coat of cats. Similarly, if we only serve the neural network with categorical data, the model may not extract and patterns from images of each cat, such as hair density. In our models trained on the fully one-hot encoded dataset, we get an accuracy of above 80% and a loss of 0.4 for the validation dataset without overfitting or underfitting. This has a better result than that of models trained with the non-one-hot encoded dataset, as well as that of ANN models trained for categorical features only. From our research, the stochastic gradient descent with momentum and decay on a combined neural network structure that ends in softmax activation can supply an optimal result for this single-label multi-class classification problem. It means the combined neural network taking both categorical features and visual features does increase the performance when evaluating the coat of cats.



References