Pet Image Recognition with Keras

Evan Hollier and Paul Homuth

Problem Statement

- Can a Image Recognition Model be used to classify 37 different pet breeds from photos?
  - What is the best pretrained model to use?
  - What are the best hyperparameters to add?

Problem Setup

- Input: Image from Oxford IIIT Pet Dataset
- Early Layers: Large Pretrained Model from Keras
- Final Layers: Custom Built Layer tuned to our data
- Output: Classification of Pet

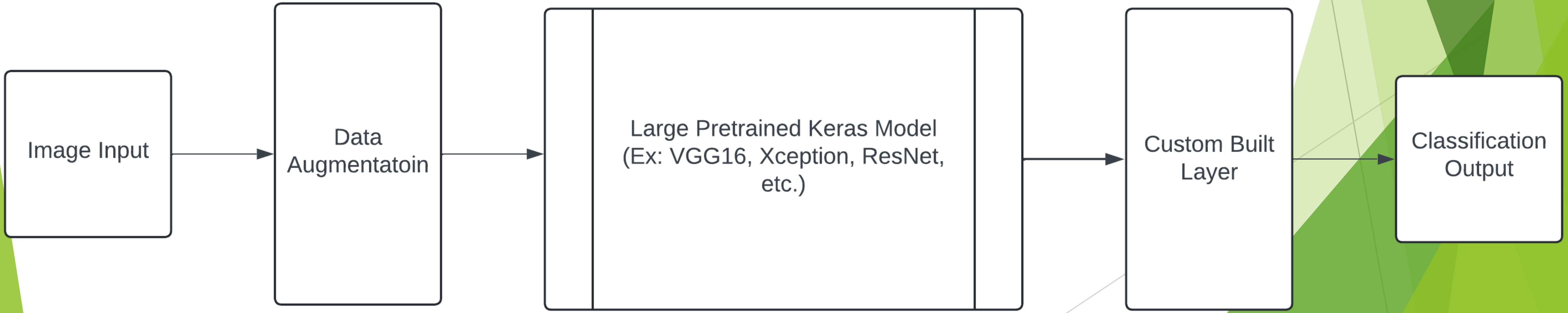


Image Input

Data Augmentatoin

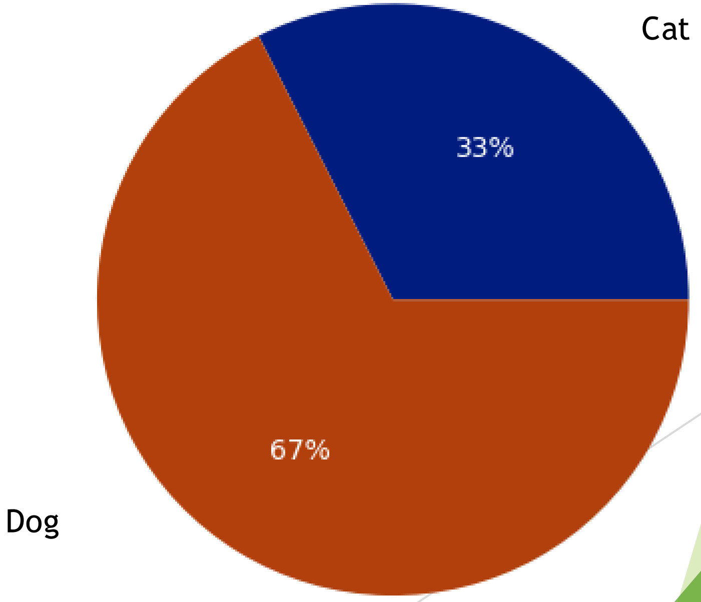
Large Pretrained Keras Model (Ex: VGG16, Xception, ResNet, etc.)

Custom Built Layer

Classification Output

Dataset Description

- Oxford IIIT Pet Dataset
  - A image set of 37 different breeds of pets. Each image has varying quality of light, scale, pose, and background making it a good random sampling of images
- Total of 7393 Images
  - 25 Dog breeds
  - 12 Cat breeds
  - Roughly 200 images per group



Data Issues

- Damaged or corrupted images
  - Found a lot local to one class (Egyptian Mau)
  - Corrupted Images have missing/warped sections
- Confirming Total images
  - Reported 7349 images, but we found 7393 images
  - Caused a lot of indexing issues requiring cleanup

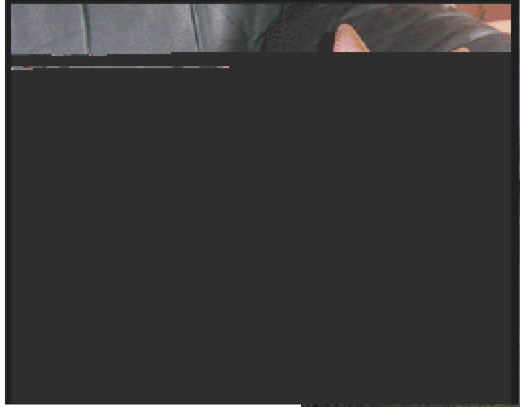


Image Quality Overview

- Non standardize dataset
  - Varying backgrounds, light levels, distance to subject, and angle of photo
  - All ages of subject can be found. Puppies and Kittens included
- Below are examples from Abyssinian cat category

Light Variations

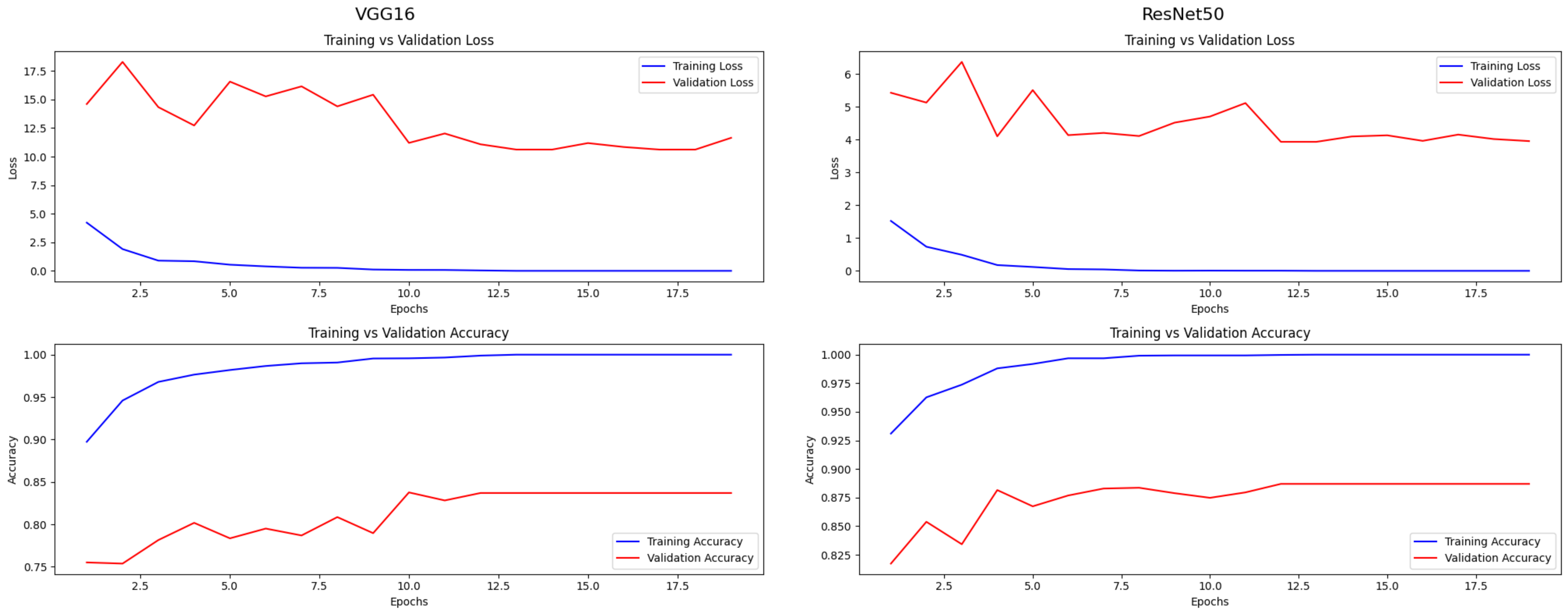
Background Variations

Age Variations

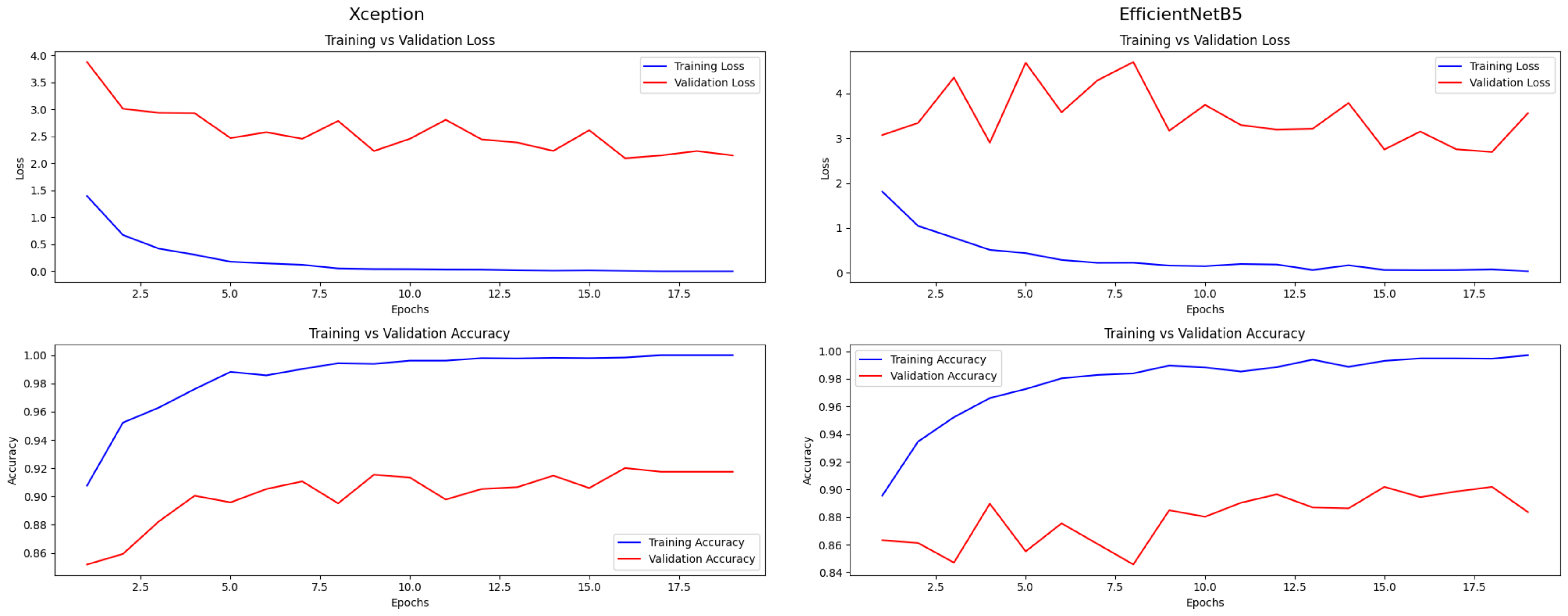
Pretrained Model Selection

- Four selected pretrained models were tested to serve as a baseline
  - Xception, VGG_16, ResNet, and EfficentNet
  - Selected due to wide range in complexity
- These baseline models were frozen and fit on our dataset
  - 37 Neurons, Softmax
- Baseline models were ran for 20 epochs
- Tracked the Loss and Accuracy metrics on both training and validation sets
  - Best model was selected out based on these metrics

Pretrained Model Selection Cont.



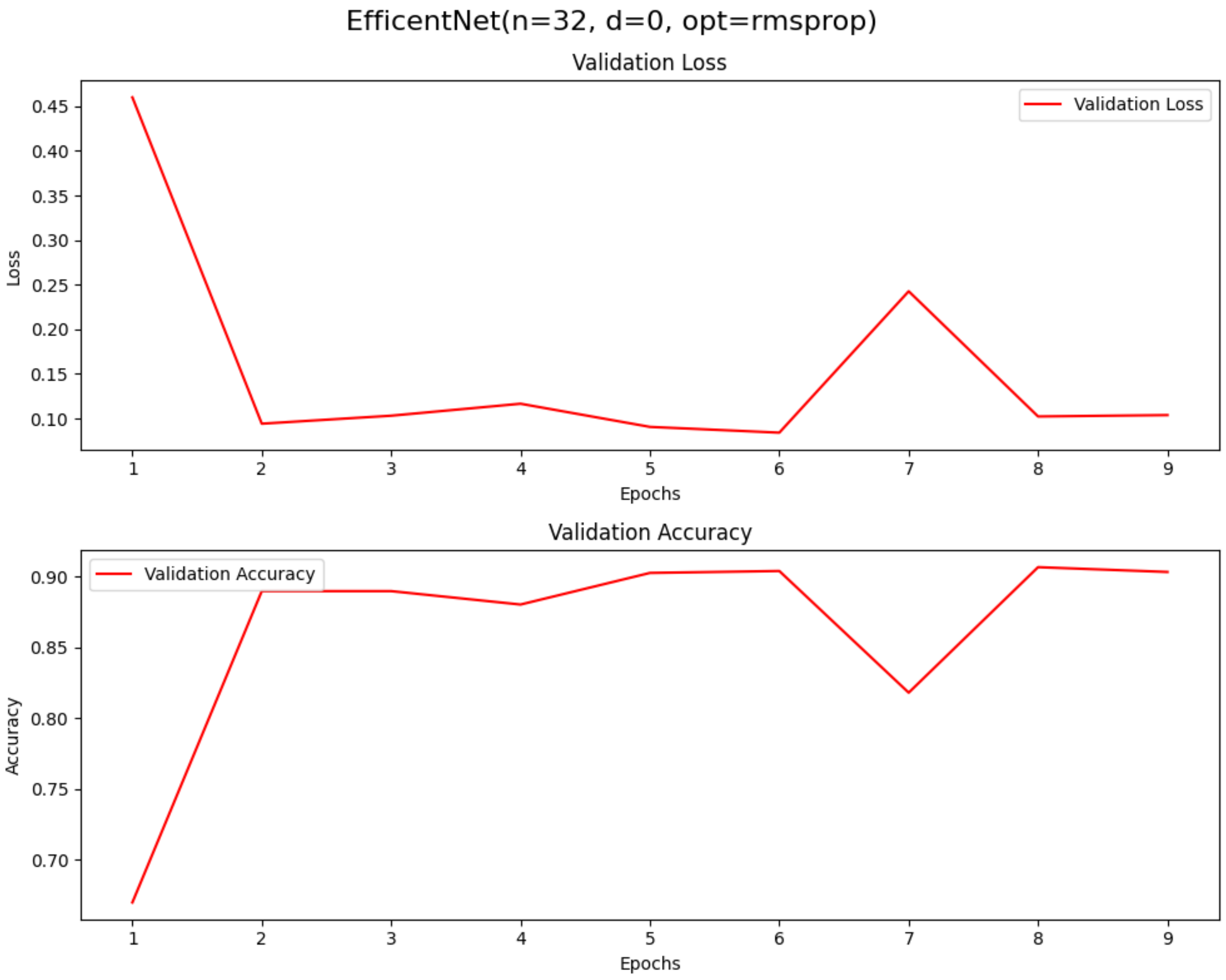
Pretrained Model Selection Cont.



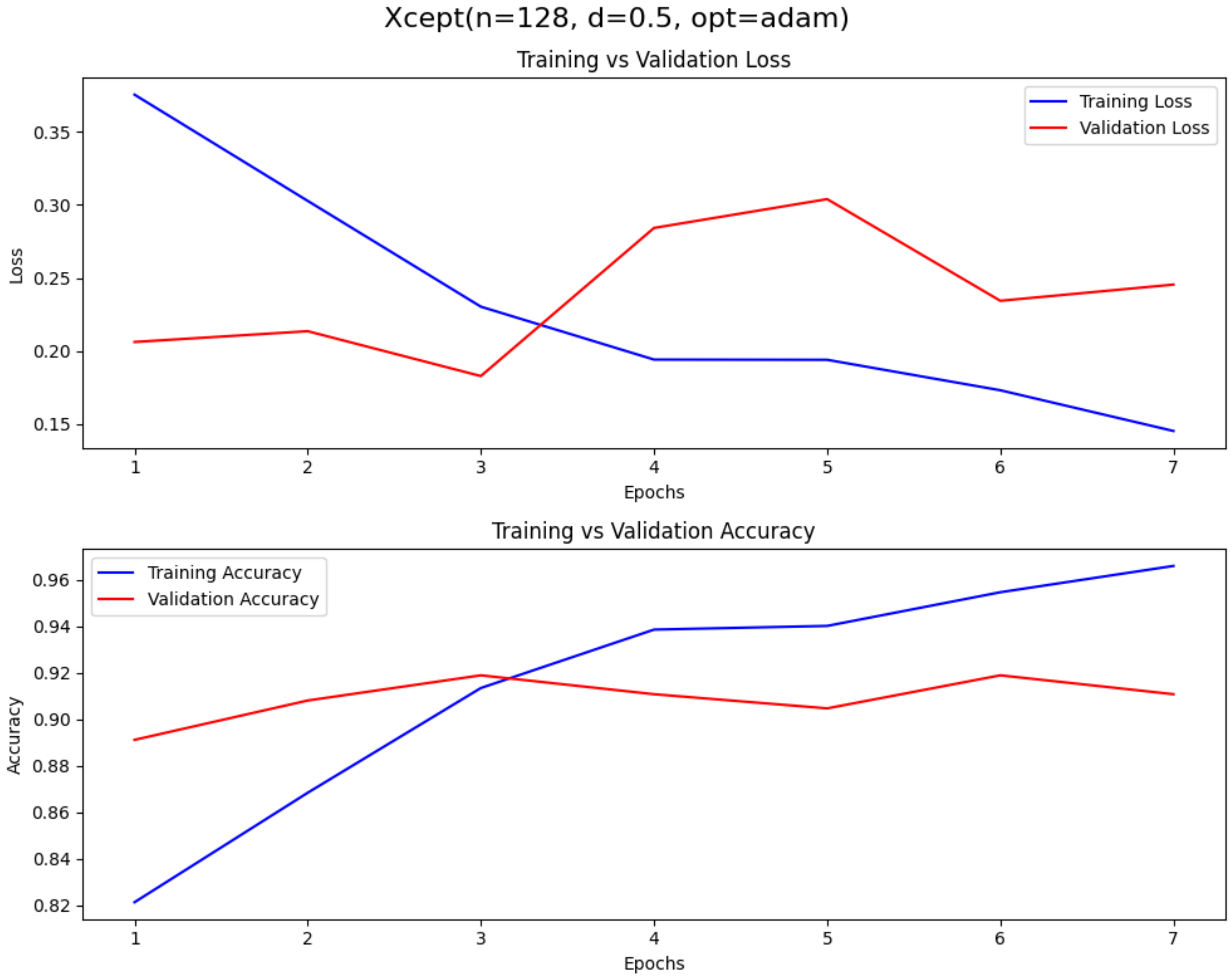
Grid Search

- Xception and EffcientNet baseline models needed a tie breaker
- Grid Searched across multiple parameters to determine winner in our custom layer
- Adjust Optimizer, Total number of Neurons, and Dropout Layer
  - Optimizers: RMSProp, Adam
  - Total Neurons: 16, 32, 64, 128
  - Dropout Layer: 0, .25, .5
- 10 epochs with early stopping patience of 4
- Best validation loss models were saved for comparison

Best Xception vs Best EfficentNet



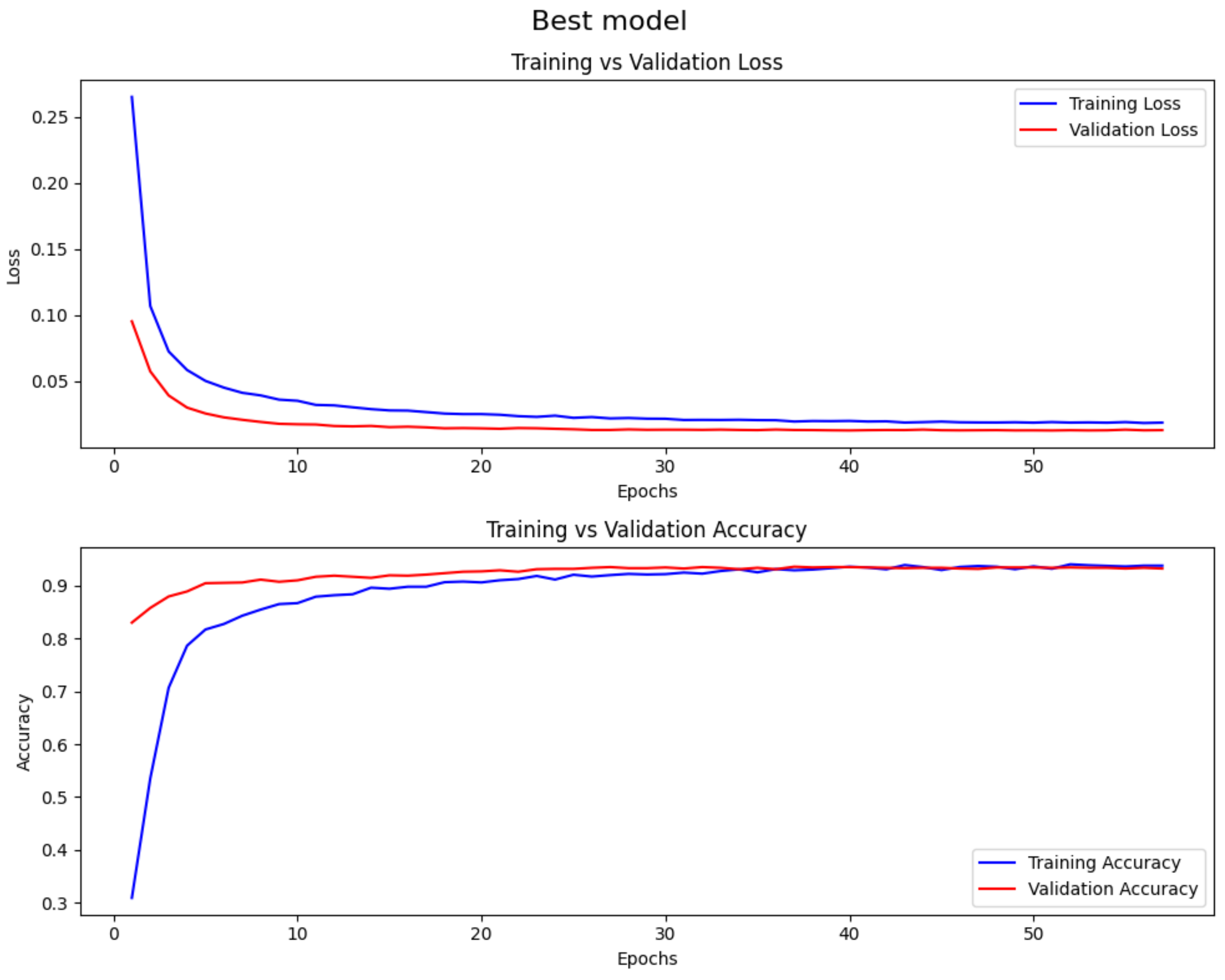
EfficentNet(n=32, d=0, opt=rmsprop)



Xcept(n=128, d=0.5, opt=adam)

Further Custom Layer Tuning

- Found Xception performed best
  - Adam optimizer, 128 neurons, and .5 dropout
- Increased total training time
- Added more dense layers before output
  - Up to two additional layers
  - Tried more neurons (256, 384, 512)
  - .5 dropout
- Tried L1, L2, and L1_L2 regularization
- Inserted Data Augmentation to increase training set
- Batch Normalization

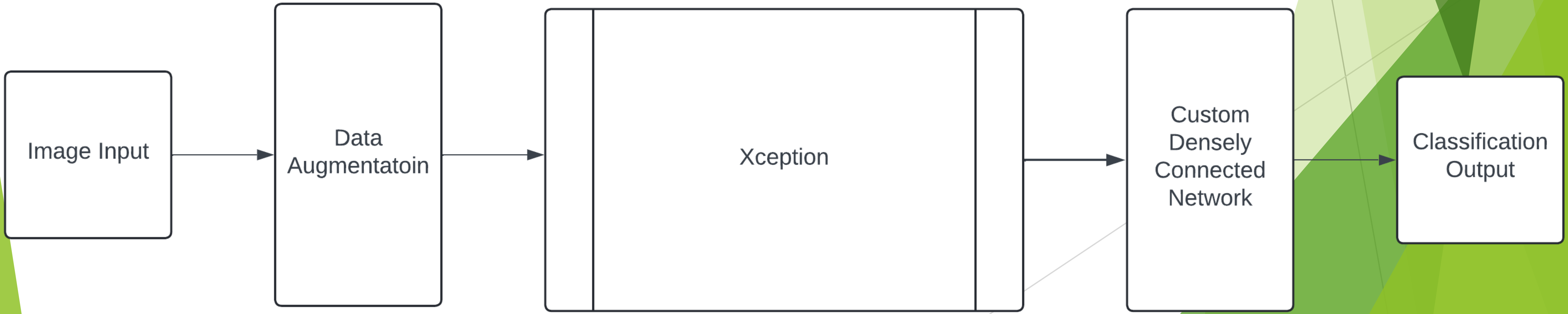

Best model

Training vs Validation Loss

Training vs Validation Accuracy

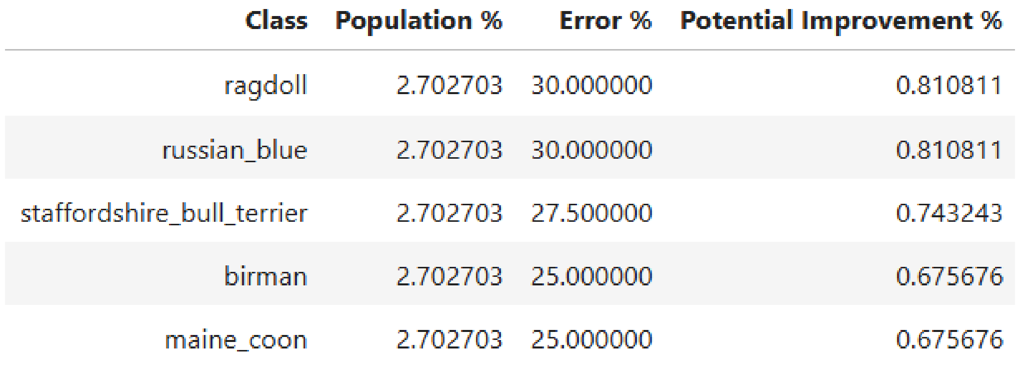
Final Model

- Input: Pet Image
- Data Augmentation: 1 out of 10,250 variations
- Early Layers: Xception with ImageNet weights
- Custom Layers: Two Densely Connected Layers
  - 256, 128 Neurons, Batch Normalized, 0.5 Dropouts
- Validation accuracy of 93.64%

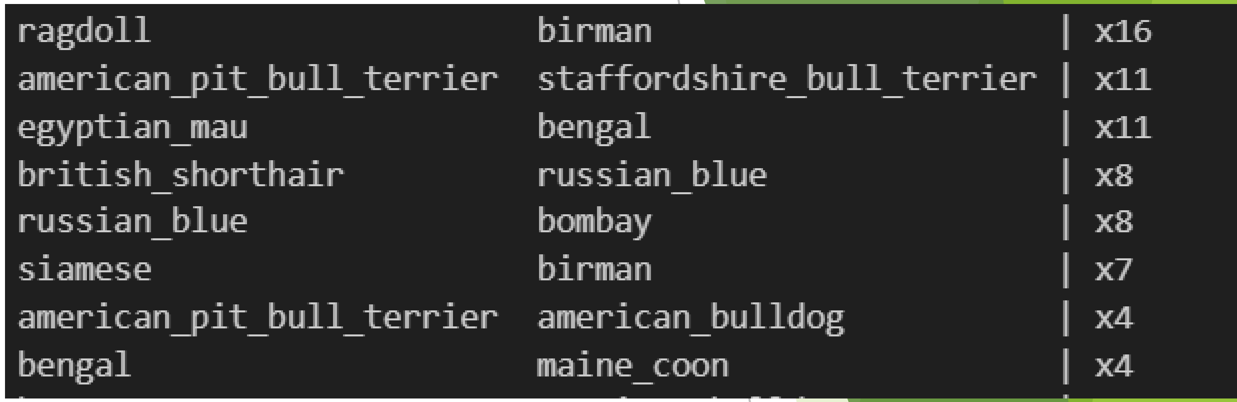


Results & Testing

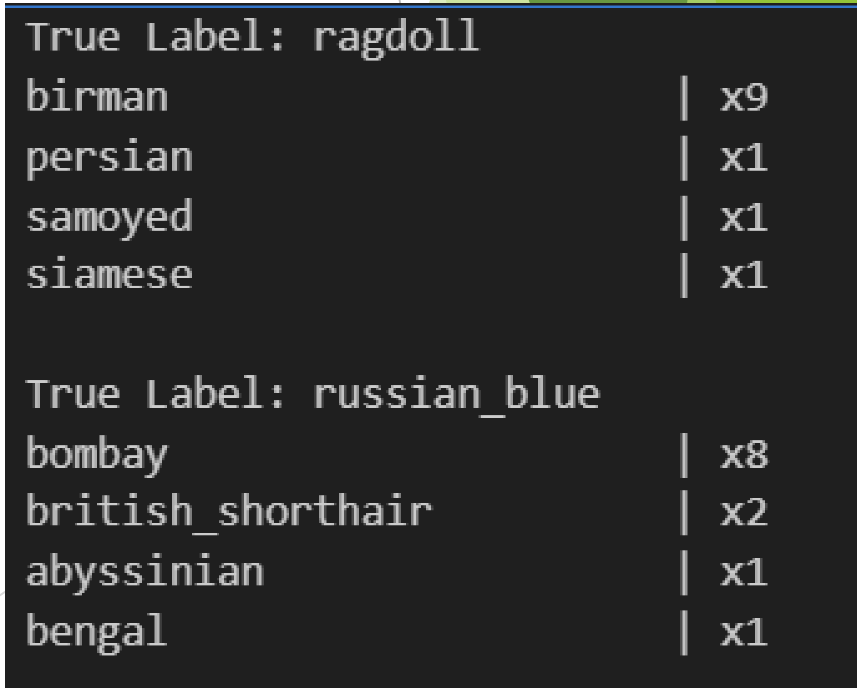
- Test Accuracy: 90.3 % = 1337 / 1480
- Test Loss: 0.0174
- 37 Classes Tested with 40 samples each
  - 27 Found to have at least one mistake
  - Ragdoll and Russian Blue Cats had the most misclassifications
  - Dogs misclassified as cats: 1
  - Cats misclassified as dogs: 5
  - Three pairs that were confused 10+ times
- Potential Improvement calculated for each class



Class	Population %	Error %	Potential Improvement %
ragdoll	2.702703	30.000000	0.810811
russian_blue	2.702703	30.000000	0.810811
staffordshire_bull_terrier	2.702703	27.500000	0.743243
birman	2.702703	25.000000	0.675676
maine_coon	2.702703	25.000000	0.675676



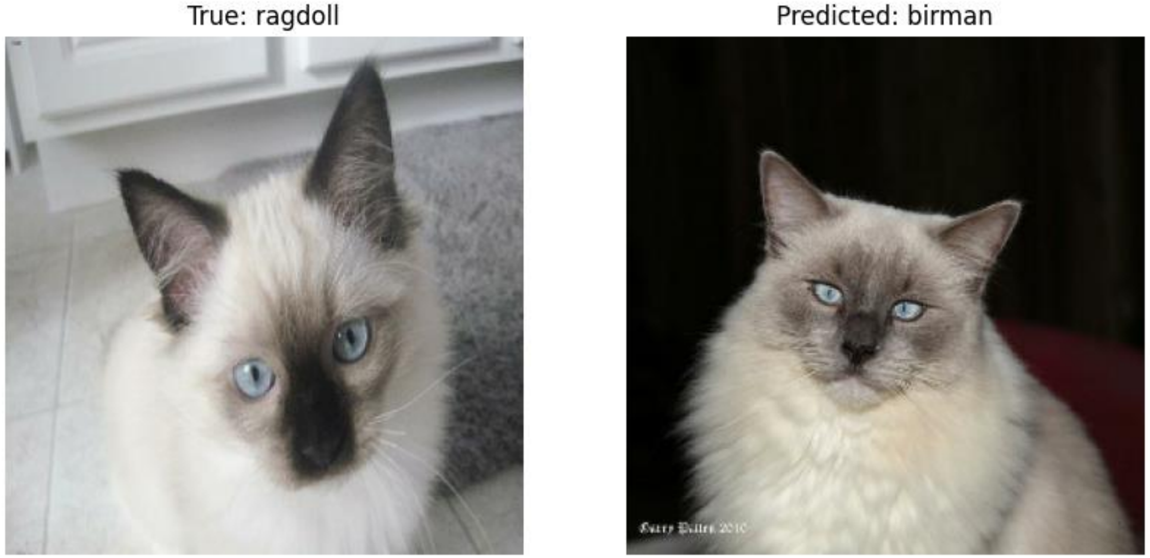
```
ragdoll                     birman                     | x16
american_pit_bull_terrier   staffordshire_bull_terrier | x11
egyptian_mau                bengal                     | x11
british_shorthair           russian_blue               | x8
russian_blue                bombay                     | x8
siamese                     birman                     | x7
american_pit_bull_terrier   american_bulldog           | x4
bengal                      maine_coon                 | x4
```

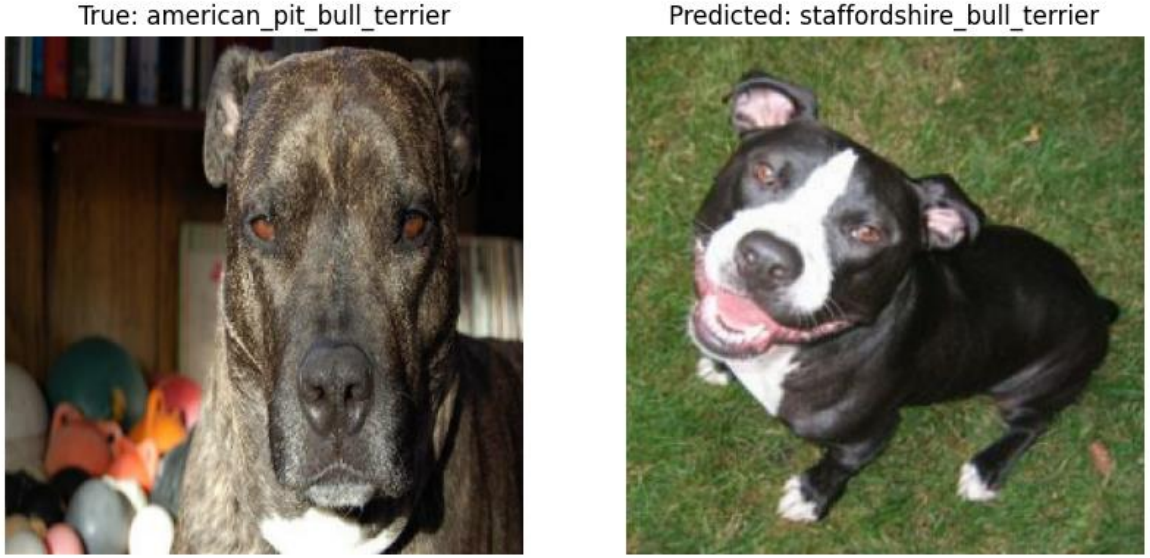


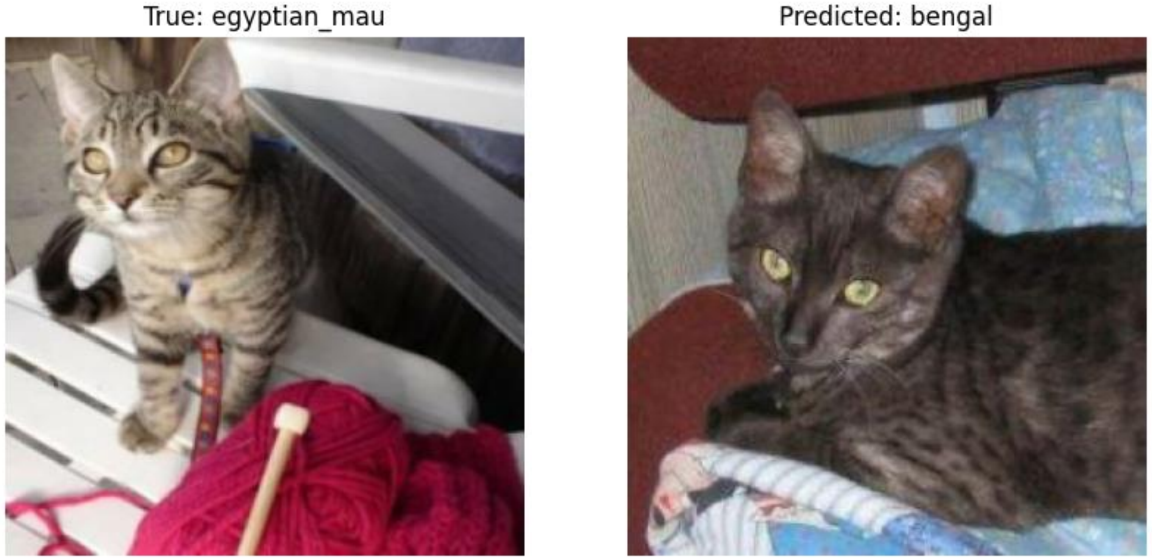
```
True Label: ragdoll
birman                | x9
persian               | x1
samoyed               | x1
siamese               | x1

True Label: russian_blue
bombay                | x8
british_shorthair     | x2
abyssinian            | x1
bengal                | x1
```

Most Confused Classes




True: american_pit_bull_terrier
Predicted: staffordshire_bull_terrier



Testing on Our Own Pets

Theo: Maine coon / Domestic Long Hair mix

Pred: maine_coon

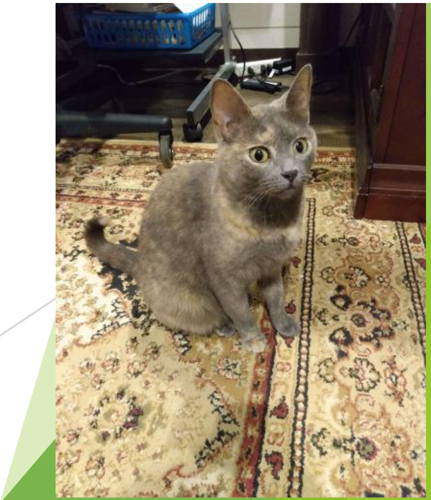
Pred: maine_coon

Pred: maine_coon

Pepper: Russian blue / Tortoiseshell mix

Pred: russian_blue

Pred: russian_blue

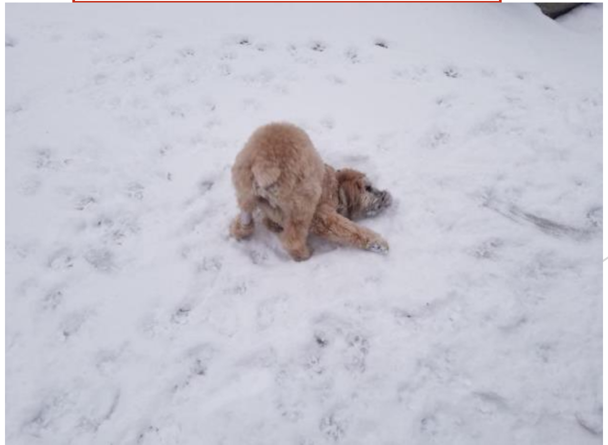


Dude: Wheaten terrier

Pred: wheaten_terrier

Pred: wheaten_terrier

Pred: abyssinian



Conclusion

- A image recognition model was trained and built:
  - Used Data Augmentation during training
  - Xception with ImageNet weights
  - Two Densely Connected Layers (256, 128)
  - Batch Normalization
  - Dropouts and variable learning rate
- The model produced a validation accuracy of 93.64 and a testing accuracy of 90.34
- Using these techniques is feasible to build an image recognition model to distinguish different pets