Cats & Dogs
Image Classification

Alexandra Schoenberg

Springboard Data Science Career Track, 2022-23

The Problem

- Can I create a model that will recognize and classify images of fourteen different types of shark species?
- Can I create a model that will do the same with images of cats and dogs?

Image Recognition & Classification

- Image recognition and classification is a function of deep machine learning with wide-reaching use cases, including:
  - Healthcare imaging
  - Scam detection and protection
  - eCommerce
  - Security
  - Many more

What Factors Might Impact Model?

- The initial dataset contains 14 different species of sharks, but many look similar to each other and do not have specific defining characteristics
- In both the sharks and cats/dogs datasets, the number of datapoints is relatively limited
- In both the sharks and cats/dogs datasets, animals are viewed from different angles and distances and are present in different numbers
- In the cats/dogs dataset specifically, animals are frequently different colors, sizes, and even shapes depending upon their breeds

Data Information

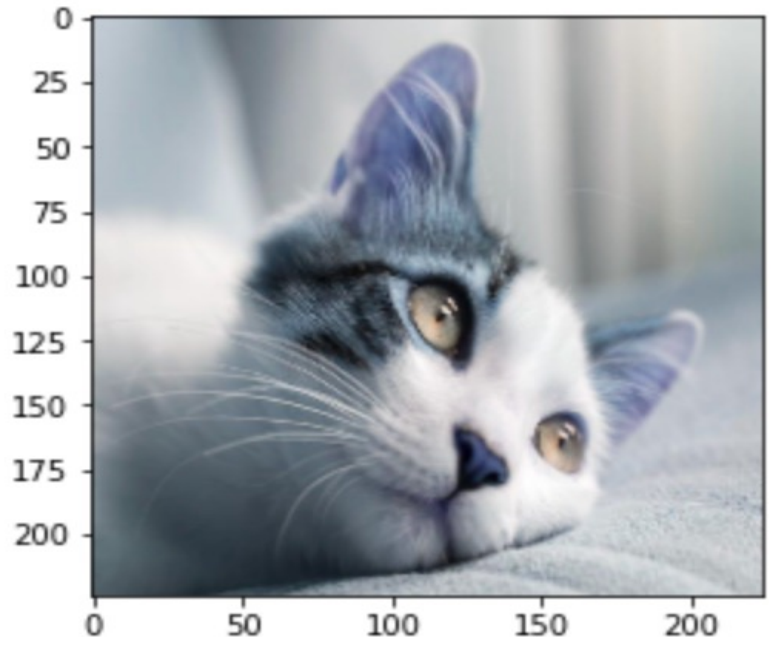
- The first dataset I attempted to use for the model included a total of 1549 image files of 14 different species of sharks
- The second and final dataset I used for the model included a total of 697 images of cats and dogs
  - Images were sourced from the internet and labeled by hand
  - Images were in jpeg format
  - Images were split into training and testing sets:
    - Training: 279 cats, 278 dogs
    - Testing: 70 cats, 70 dogs

Data Wrangling

- Images were downloaded with the naming convention “(dog/cat)_(number)”.jpg
- When images were imported into the notebook and appended into the training and testing datasets to be used for the model, I used this naming convention to create a label of ”cat” or “dog” for each image
- The training set labels were one-hot encoded
  - Cats were labeled with [1 0]
  - Dogs were labeled with [0 1]
- The training data was further split into training and validation data
  - 417 training images
  - 140 validation images

Data Exploration

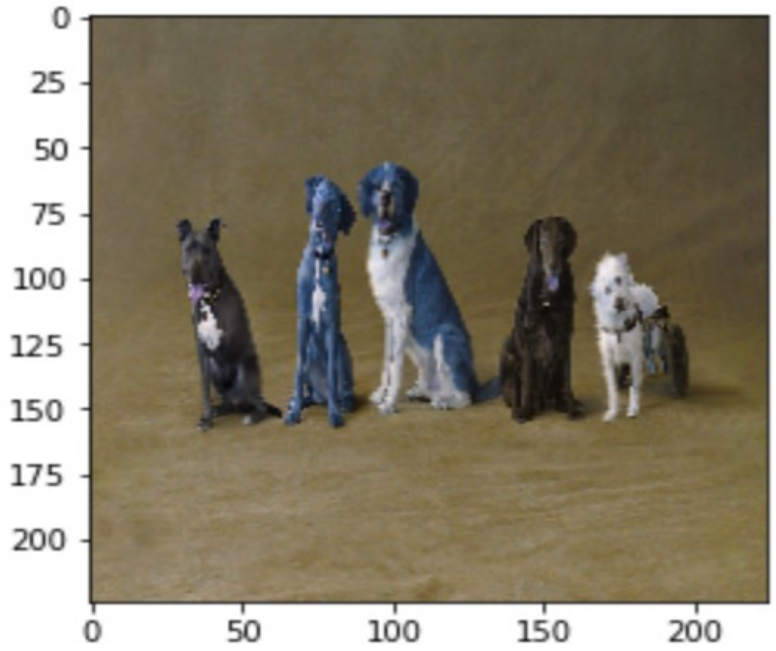
```
[1 0]
<matplotlib.image.AxesImage at 0x7f08662744f0>
```



I performed random checks on the training and testing datasets to ensure that each image was properly imported and labeled. And example of the training set labeling is above.

Data Visualization

```
dog
<matplotlib.image.AxesImage at 0x7f0861db8c10>
```



The testing data image labels were not initially one-hot encoded. An example of their output during spot-checks is above.

Modeling

- I tested and modified many models for this project
- The most successful two models were a basic Sequential Neural Network model and a MobileNetV2 model

Modeling

- The Sequential model was built manually, using layer types including:
  - 2D Convolutional
  - 2D Max Pooling
  - Batch Normalization
  - Flatten
  - Dense
- With the exception of the final Dense layer, which used ‘softmax’ activation to output the 2 categorical results, each layer’s activation type was ‘relu’
- The model was compiled using the ‘Adam’ optimizer and assessed for performance based on accuracy
- The model had a total of 9,837,090 trainable parameters

Modeling

- The MobileNetV2 model is a pre-built model available in Keras, and was applied as a form of transfer learning
- MobileNetV2 is a convolutional neural network that is 53 layers deep and has been trained on more than 1 million images from the ImageNet database
- The MobileNetV2 model was used to supplement a Sequential model of layers:
  - 2D Global Average Pooling
  - Batch Normalization
  - Dropout
  - Dense
- The model was compiled using the 'Adam' optimizer and assessed for performance based on accuracy
- The model had a total of 726,754 trainable parameters

Modeling

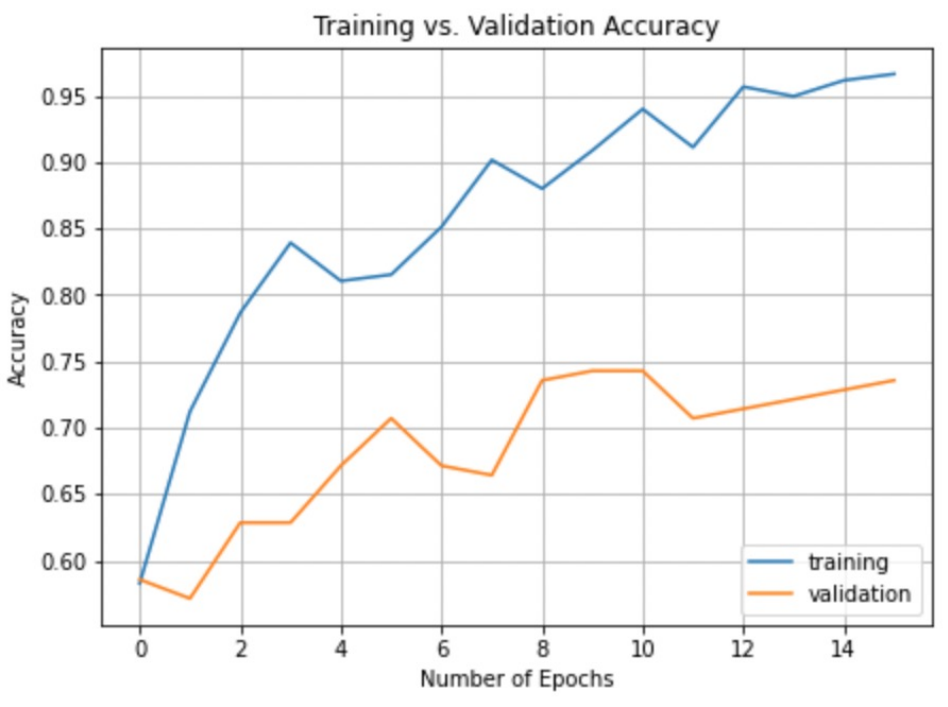
- Both models were set to respond and adjust to callbacks, which I defined, during their training
- Callbacks included:
  - Early Stop function if the model was not demonstrating improvement for more than 10 epochs
  - Learning Rate Reduction function if the model's validation set accuracy plateaued for more than 2 epochs

Model Results

- Model 1 executed for 23 epochs before reaching its early stop threshold, achieving accuracies of:
  - 0.65 for cats
  - 0.68 for dogs
- The more successful model was Model 2, the MobileNetV2 model, which executed for 16 epochs before reaching its early stop threshold. It achieved accuracies of:
  - 0.76 for cats
  - 0.71 for dogs

Model Results

Model 2 Training vs. Validation Accuracy



Model 2 Classification Report

	precision	recall	f1-score	support
cats	0.61	0.69	0.64	70
dogs	0.64	0.56	0.60	70
micro avg	0.62	0.62	0.62	140
macro avg	0.62	0.62	0.62	140
weighted avg	0.62	0.62	0.62	140
samples avg	0.62	0.62	0.62	140

Takeaways and Future Steps

- There is room for improvement in the model to achieve higher accuracy. Some methods to further improve the model might include:
  - Data augmentation
  - Further training/parameter tweaking of existing models
  - Transfer learning from more complex image recognition models
  - Etc.
- This project was a solid introduction to the capabilities and obstacles of image classification in machine learning
- Although the sharks model was not very successful, I gained many valuable skills that I could apply to the cats and dogs version as well as future projects
- In the future, I will build upon this foundation to ask better questions and build better models