IMAGE CLASSIFICATION : DOG V CAT

MIA BROWN

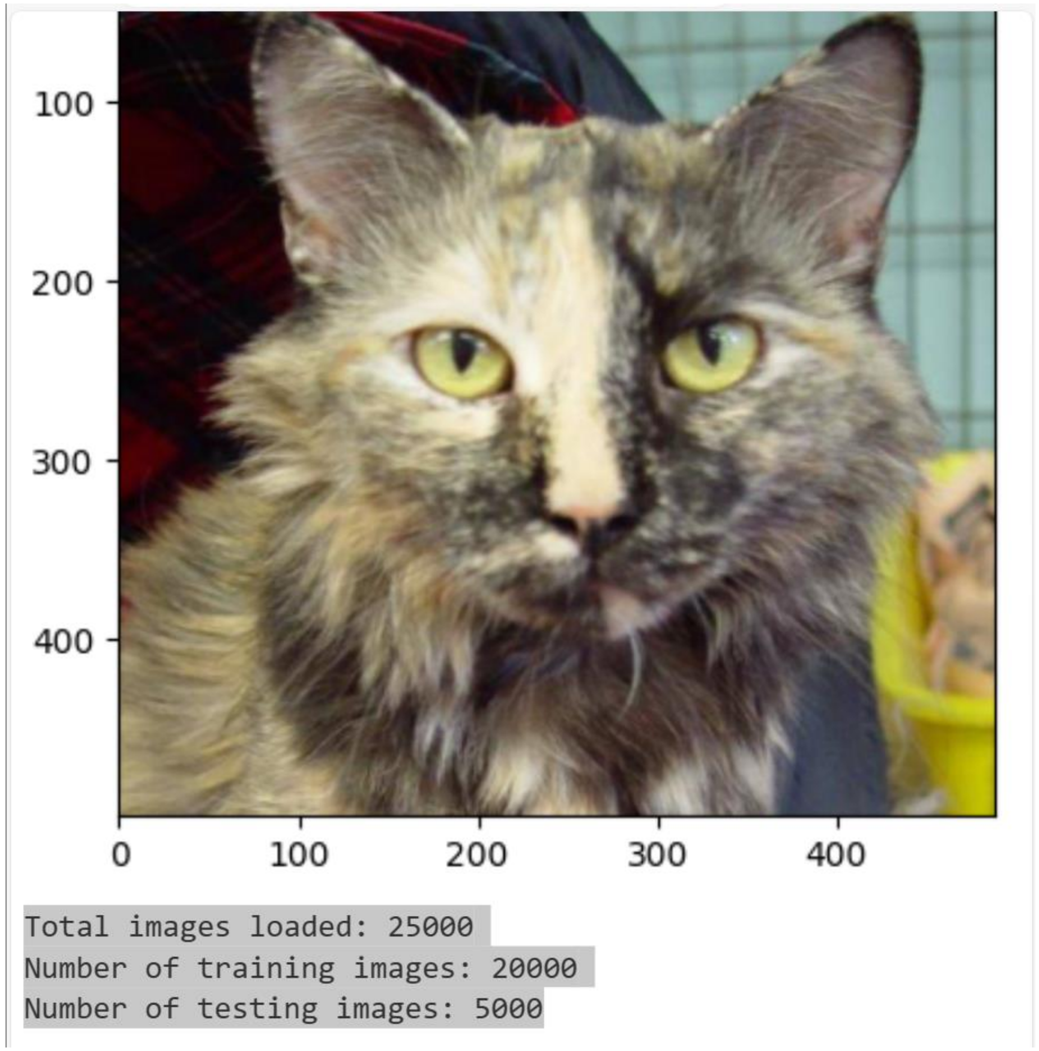
AGENDA

- Introduction
- Types of libraries
- Steps for classification
- Data Extraction
- Example Test for accuracy
- Actual results
- Challenges
- Possible improvements

PROJECT GOAL

The Project that I decided to do is image classification with an image of a dog or a cat.

- The first step in this project is to gather a dataset of images containing both dogs and cats. Fortunately, there are many publicly available datasets for this purpose, such as the Kaggle Dogs vs. Cats dataset.



```
Total images loaded: 25000
Number of training images: 20000
Number of testing images: 5000
```

Supervised Image Classification

Uses labeled training data to train a model to classify images into predefined categories or classes.

- **Most used**

In supervised image classification, each image in the training dataset is associated with a label or category, and the algorithm learns to map input images to their corresponding labels during training.

Unsupervised Image Classification

Involves clustering or grouping images based on their intrinsic properties without labeled training data.

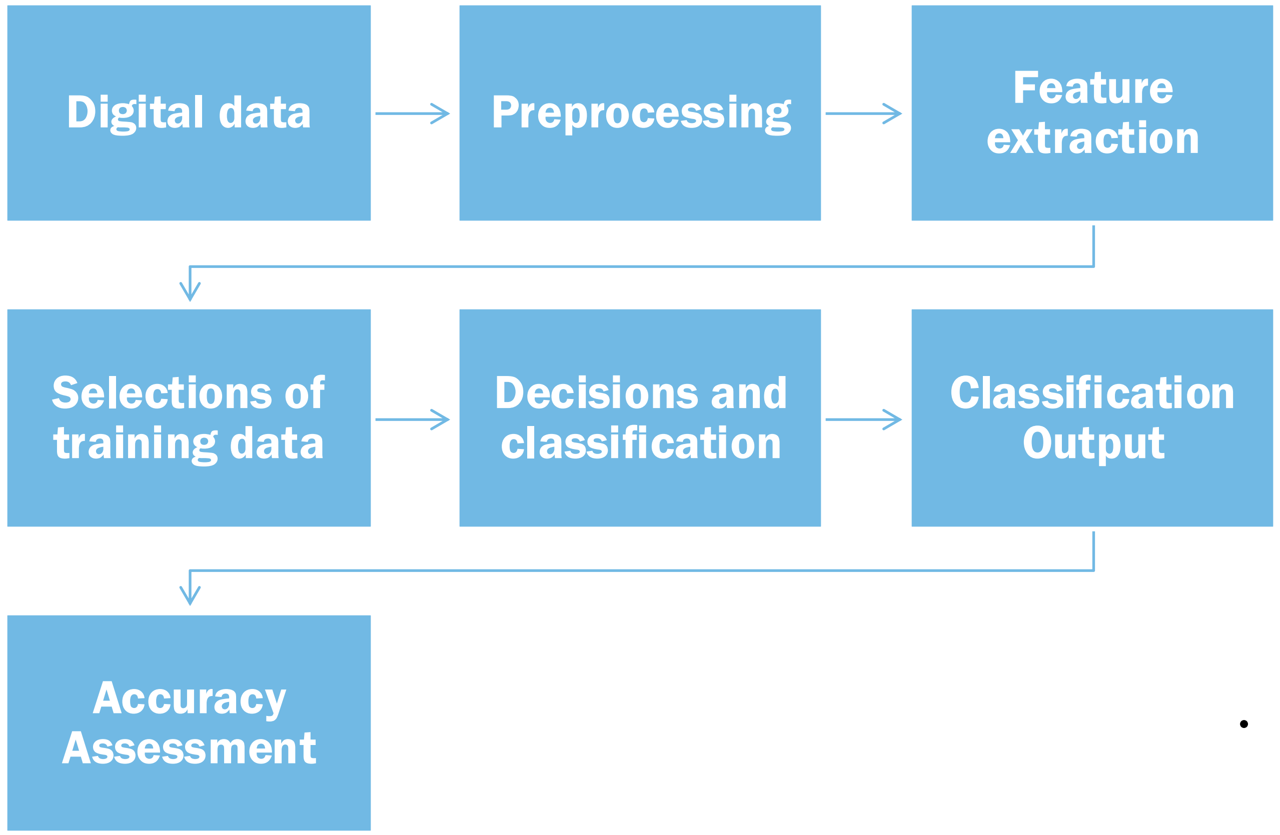
This approach is less common in image classification tasks compared to supervised learning because it relies on finding patterns or similarities in the data without explicit guidance from labeled examples.

Unsupervised clustering can be used to segment images into different regions based on similarity in pixel values, colors, textures, or other visual features.

LIBRARIES

```
import os
import zipfile
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
from sklearn.model_selection import
train_test_split
from sklearn.metrics import accuracy_score
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D,
Flatten, Dense
```

The project required the use of various libraries and frameworks tailored for machine learning and deep learning tasks. Key libraries included os, zipfile, matplotlib, numpy, PIL, scikit-learn, and Keras. Data preprocessing, model architecture design, and optimization were central to the model training process. Convolutional Neural Network (CNN) architecture was employed, featuring Conv2D, MaxPooling2D, Flatten, and Dense layers for feature extraction, dimensionality reduction, and classification.



Digital Data:

- Digital data refers to information that is stored and transmitted in a binary format, represented by combinations of 0s and 1s.

Preprocessing:

- Preprocessing involves cleaning, transforming, and organizing raw digital data to prepare it for analysis or modeling.

Feature Extraction:

- Feature extraction involves identifying and selecting relevant features from the preprocessed data. In image processing, features may include edges, textures, colors, or other visual patterns.

- Accuracy Assessment:
  - Accuracy assessment involves evaluating the performance of the classification model by comparing its predictions to the ground truth labels or known outcomes.

Selection of Training Data:

- The selection of training data involves choosing a subset of the preprocessed data to train a machine learning model.

Decisions and Classification:

- Decisions and classification refer to the process of using a trained model to make predictions or decisions about new, unseen data.

Classification Output:

- The classification output is the result of the classification process, indicating the predicted labels or categories assigned to the input data by the model.

WHY CNN?

CNN's have many advantages that makes is great for image classification tasks:

- Spatial Hierarchies

CNNs excel at learning intricate features hierarchically. Initially, the lower layers focus on detecting elementary features such as edges, whereas as the network delves deeper, it gradually assimilates more intricate patterns and representations of objects.

- Parameter Sharing

It refers to the practice of using the same set of weights (parameters) for multiple computations within the network.

Parameter sharing helps reduce the number of trainable parameters in the network

DURING CONVOLUTION

During a convolution, the filters (matrices the same size as the tile size) effectively slide over the input feature map's grid horizontally and vertically, one pixel at a time, extracting each corresponding tile (see Figure 3).

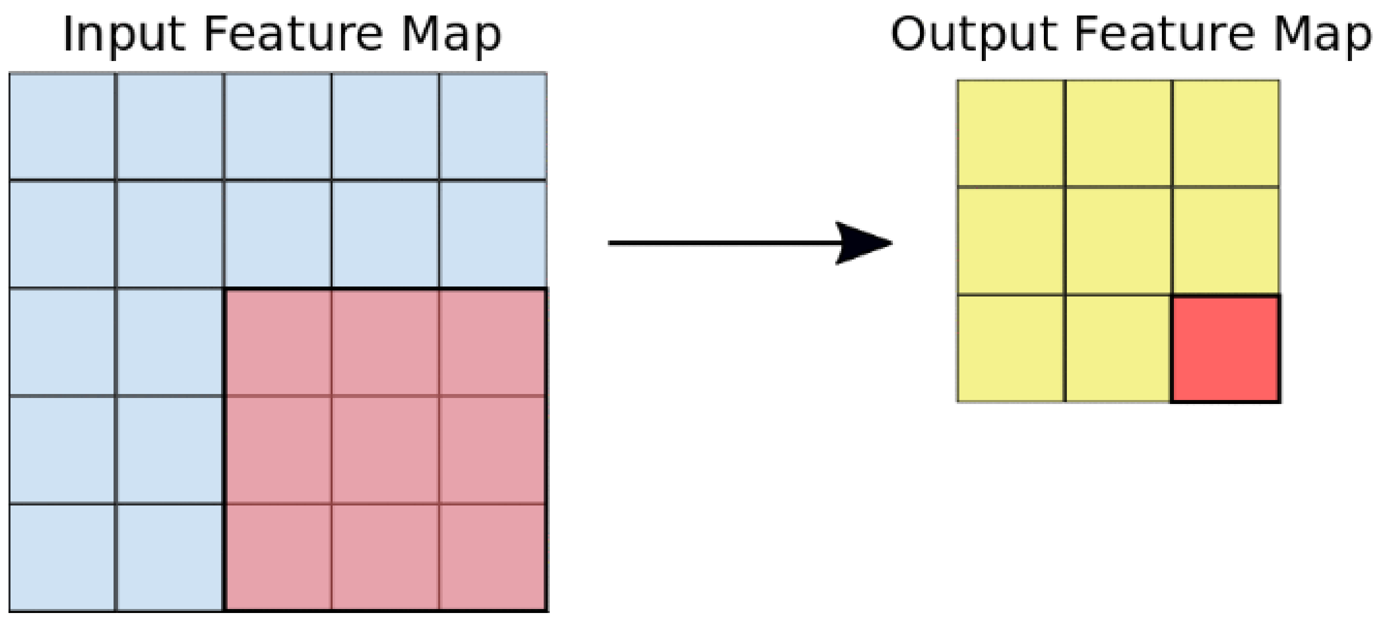
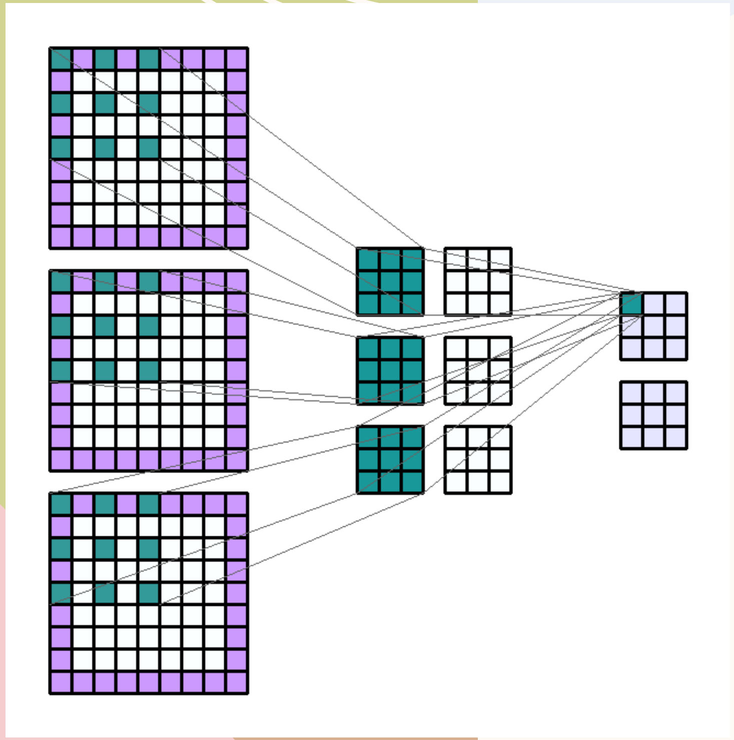


Figure 3. A 3x3 convolution of depth 1 performed over a 5x5 input feature map, also of depth 1. There are nine possible 3x3 locations to extract tiles from the 5x5 feature map, so this convolution produces a 3x3 output feature map.

ML Practicum: Image Classification | Machine Learning | Google for Developers

EXTRACTING FEATURES



Conv2D serves as the convolutional layer for extracting features from the input images. It's one of the fundamental building blocks of CNNs used for tasks such as image recognition, classification, and object detection.

This layer applies a convolution operation to the input, passing the result to the next layer. In image processing, this operation involves sliding a filter/kernel over the input image

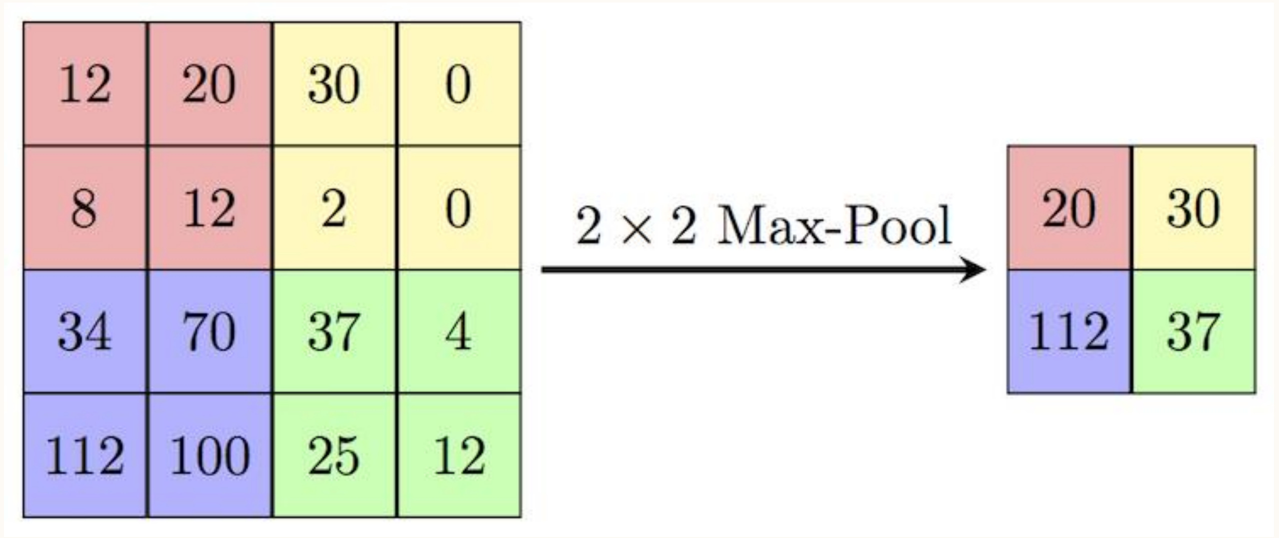
It involves sliding a matrix (also known as a kernel or filter) over a 2D data input and performing element-wise multiplication with the data that falls under the kernel.

MAXPOOLING2D LAYER

MaxPooling2D is a type of pooling layer commonly used in convolutional neural networks (CNNs) for image processing tasks. Pooling layers are used to reduce the spatial dimensions of the input volume, which helps in controlling overfitting and reducing computational complexity.

We have a 4x4 matrix for the initial input and a 2x2 that will run over the input

For each of the regions represented by the filter, we will take the **max** of that region and create a new, output matrix where each element is the max of a region in the original input.



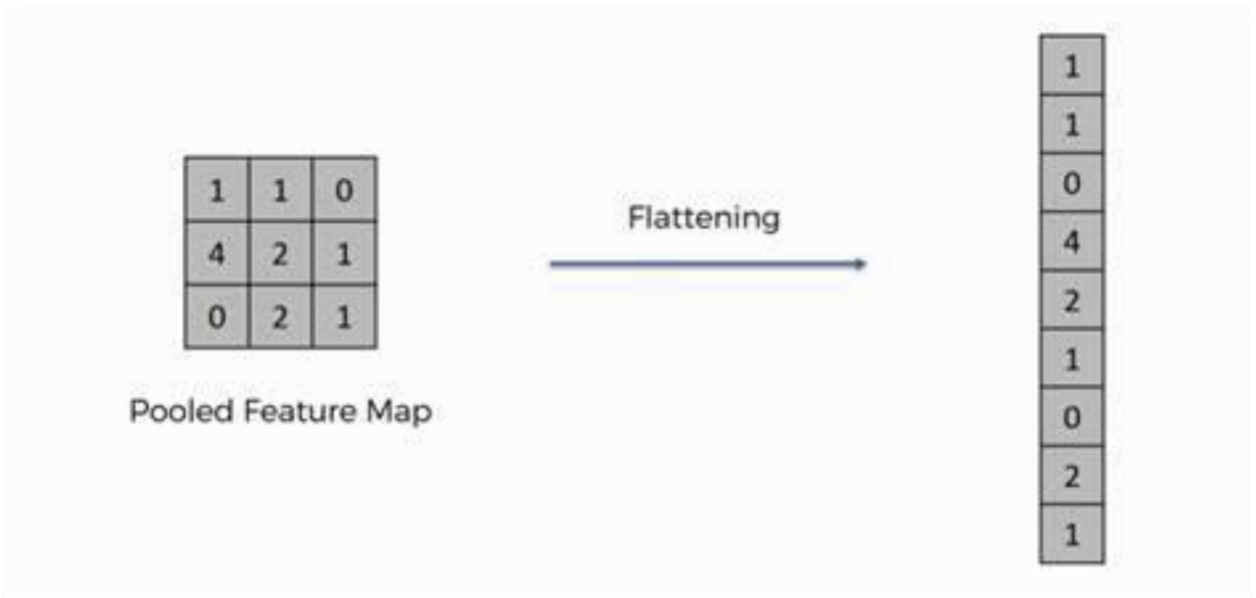
FLATTEN AND DENSE

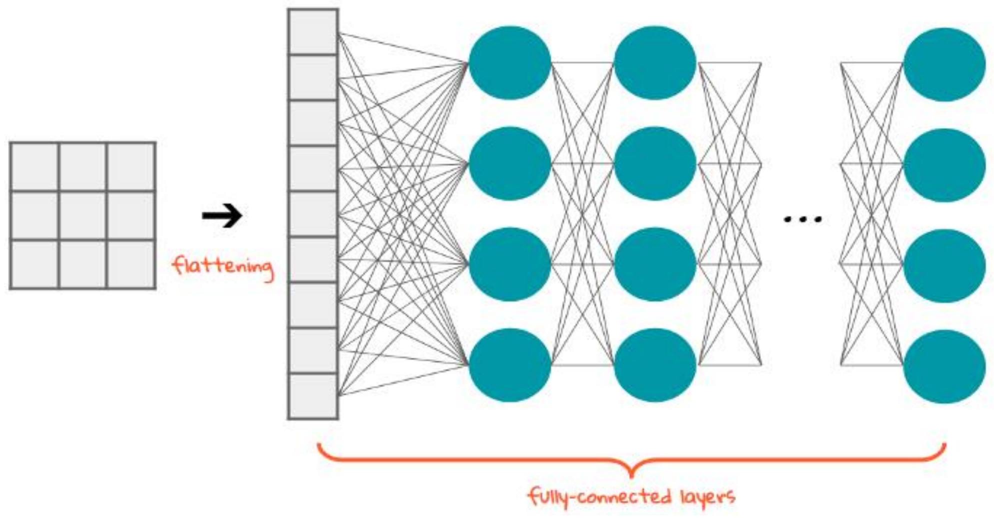
Flatten

- Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing
- After flattening, the flattened feature map is passed through a neural network. This step is made up of the input layer, the fully connected layer, and the output layer.

Dense

- It is a fully connected layer. Each node in this layer is connected to the previous layer i.e densely connected. This layer is used at the final stage of CNN to perform classification.



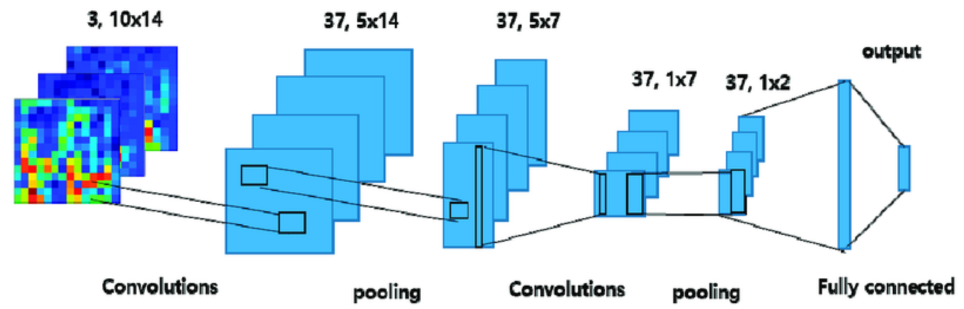


RESIZING IMAGES

Use a library like PIL to resize all the photos used to the same size.

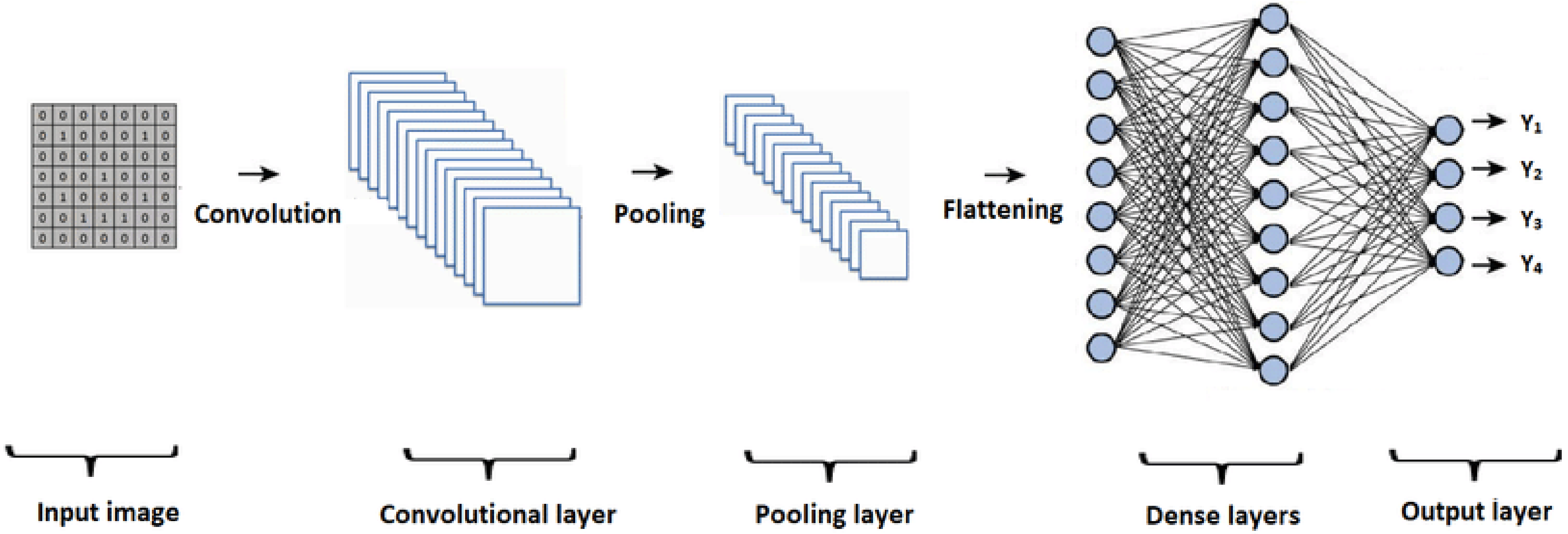
- Images in a dataset may vary in size, which can affect model performance. Resizing images to a uniform size ensures consistency in input dimensions.

- Resizing can be done using various interpolation methods such as nearest neighbor, bilinear, or bicubic interpolation. Common sizes for resized images include 224x224 or 256x256 pixels



- Why is pooling after convolution?
- Dimensionality Reduction: Pooling reduces the spatial dimensions (width and height) of the input volume, while retaining important information.

5x14, 5x7, 1x7, 1x2


Convolution
Pooling
Flattening
Y_1
Y_2
Y_3
Y_4
Input image
Convolutional layer
Pooling layer
Dense layers
Output layer

DATA EXTRACTION

DATA STORAGE

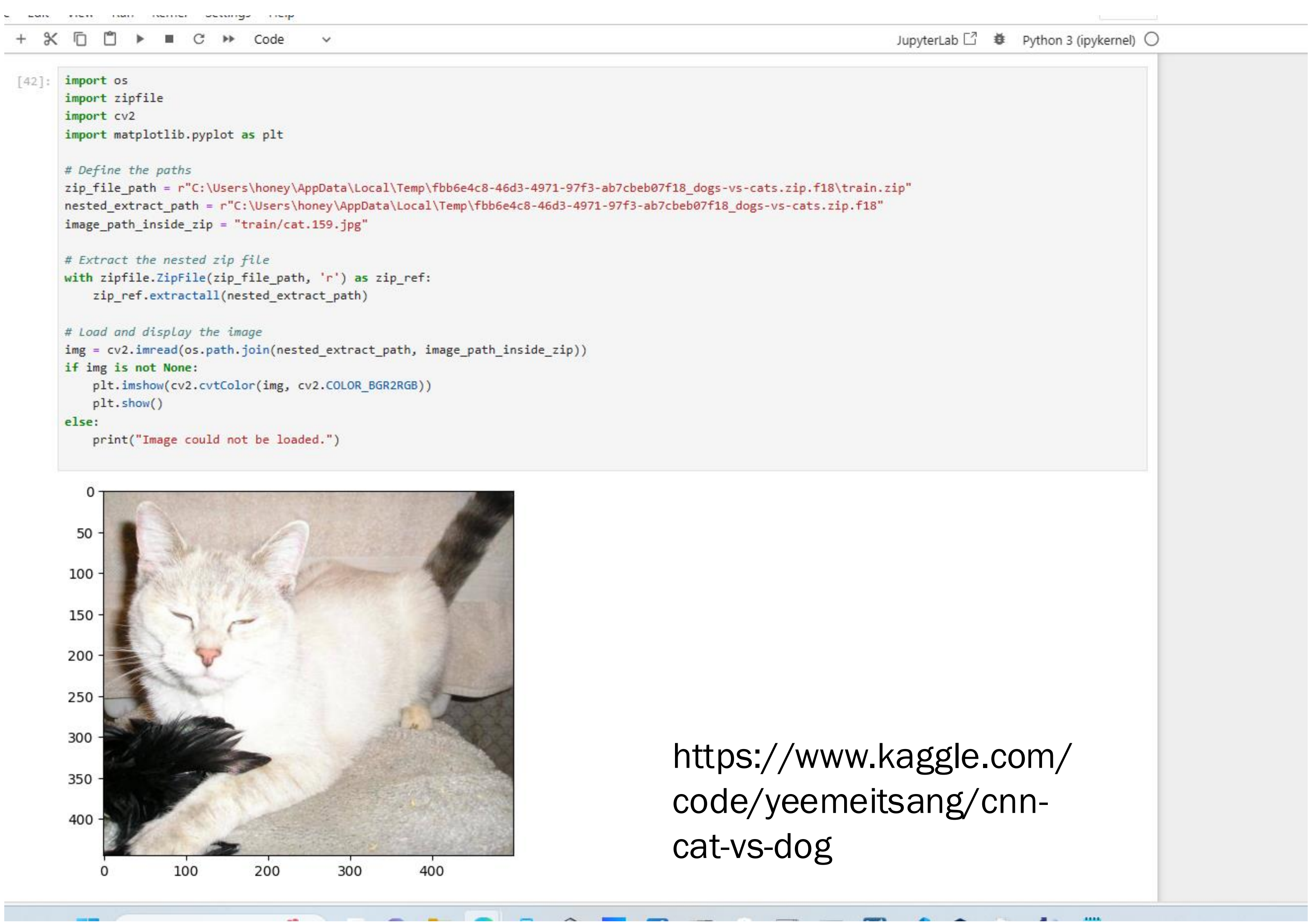
I will be putting the data into a zip file that has all the training data necessary for this task.

- These folders will have the images needed for the training data.
- Kaggle dog vs cat

The code will lead to the path of the zip files containing the necessary data for extraction.

12,499 images of cats

12,499 images of dogs

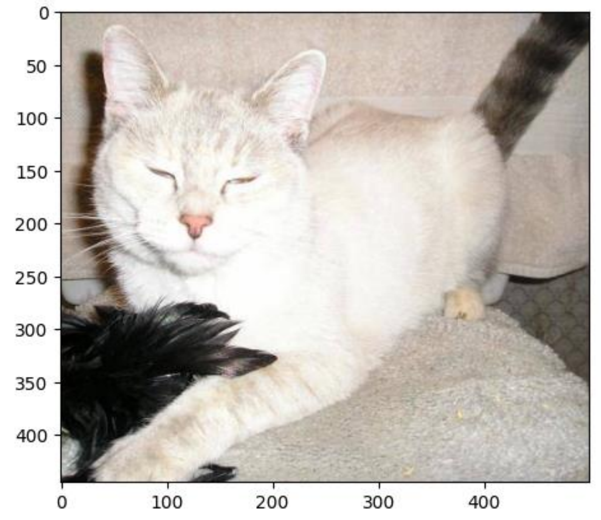


```python
import os
import zipfile
import cv2
import matplotlib.pyplot as plt

# Define the paths
zip_file_path = r"C:\Users\honey\AppData\Local\Temp\fbb6e4c8-46d3-4971-97f3-ab7cbeb07f18_dogs-vs-cats.zip.f18\train.zip"
nested_extract_path = r"C:\Users\honey\AppData\Local\Temp\fbb6e4c8-46d3-4971-97f3-ab7cbeb07f18_dogs-vs-cats.zip.f18"
image_path_inside_zip = "train/cat.159.jpg"

# Extract the nested zip file
with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
    zip_ref.extractall(nested_extract_path)

# Load and display the image
img = cv2.imread(os.path.join(nested_extract_path, image_path_inside_zip))
if img is not None:
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.show()
else:
    print("Image could not be loaded.")
```



https://www.kaggle.com/code/yeemeitsang/cnn-cat-vs-dog

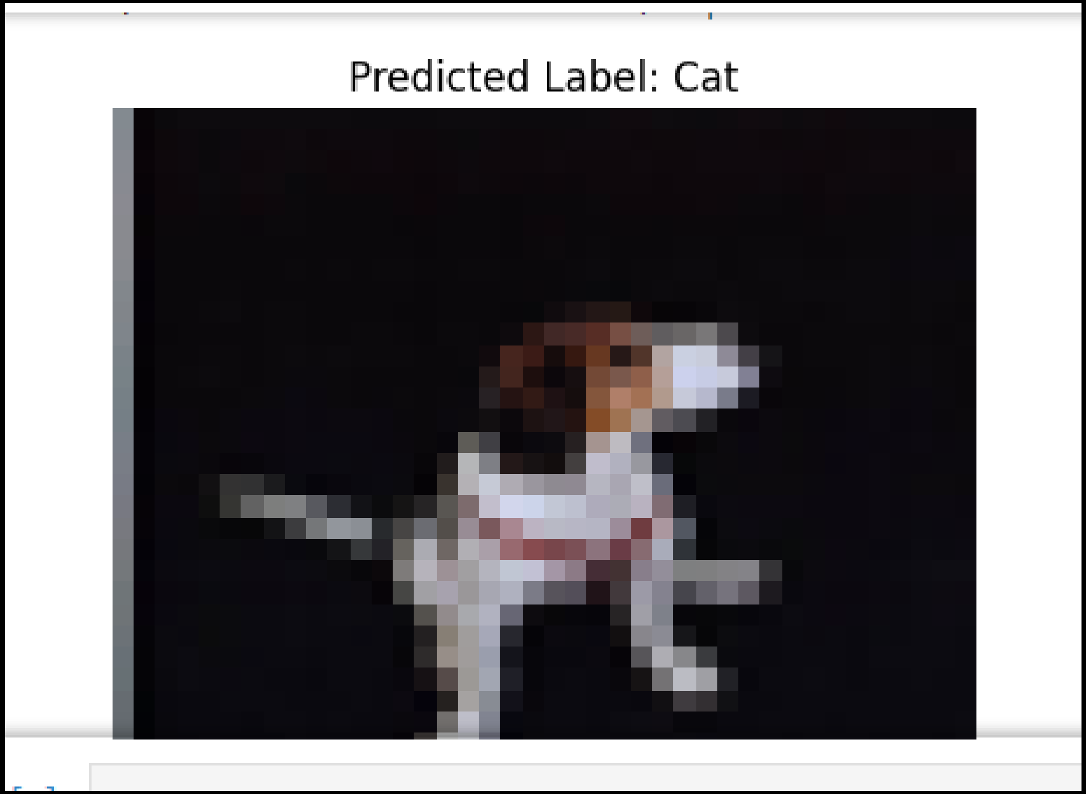
EVALUATION… SO FAR

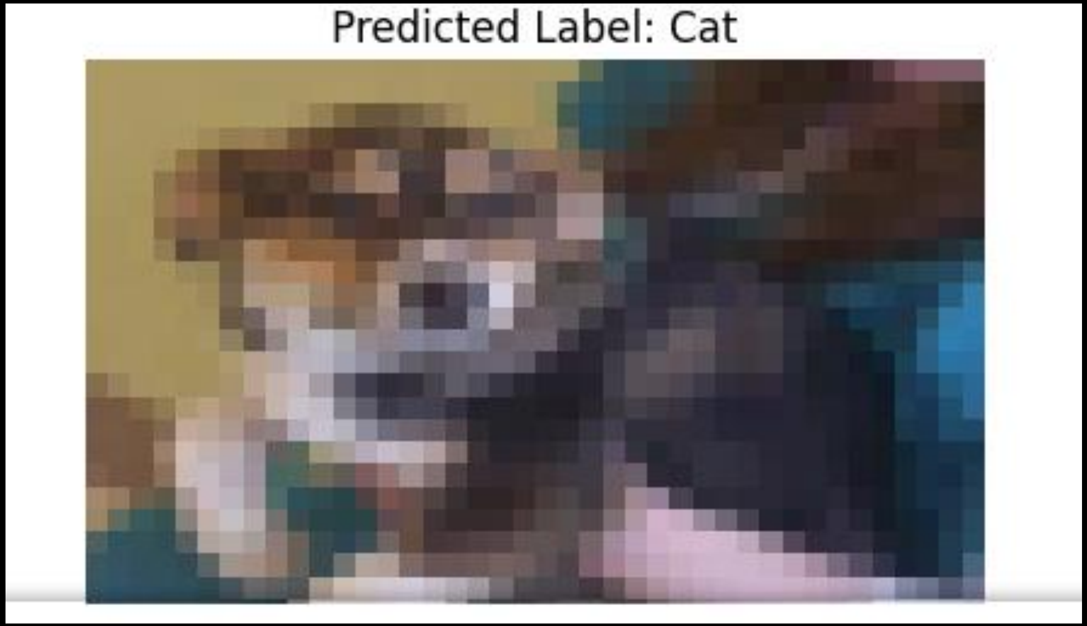
This output is from training and evaluating a Convolutional Neural Network (CNN) model for binary classification of images, specifically for identifying whether an image contains a cat or not.

25000 - This indicates the total number of images loaded from the dataset.

The test accuracy is approximately 76.03%, indicating that the model performs reasonably well on unseen data.

```
Total images loaded: 25000
Epoch 1/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 5s 9ms/step - accuracy: 0.5762 - loss: 3.0521 - val_accuracy: 0.6745 - val_loss: 0.6099
Epoch 2/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.6943 - loss: 0.5823 - val_accuracy: 0.7075 - val_loss: 0.5670
Epoch 3/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 5s 10ms/step - accuracy: 0.7285 - loss: 0.5316 - val_accuracy: 0.7195 - val_loss: 0.5636
Epoch 4/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 5s 9ms/step - accuracy: 0.7569 - loss: 0.4968 - val_accuracy: 0.7333 - val_loss: 0.5476
Epoch 5/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 5s 9ms/step - accuracy: 0.7778 - loss: 0.4613 - val_accuracy: 0.7458 - val_loss: 0.5325
Epoch 6/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.7986 - loss: 0.4265 - val_accuracy: 0.7590 - val_loss: 0.5174
Epoch 7/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.8198 - loss: 0.3970 - val_accuracy: 0.7563 - val_loss: 0.5241
Epoch 8/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.8312 - loss: 0.3642 - val_accuracy: 0.7667 - val_loss: 0.5151
Epoch 9/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.8565 - loss: 0.3353 - val_accuracy: 0.7640 - val_loss: 0.5857
Epoch 10/10
500/500 ━━━━━━━━━━━━━━━━━━━━ 4s 9ms/step - accuracy: 0.8554 - loss: 0.3212 - val_accuracy: 0.7638 - val_loss: 0.5935
157/157 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step - accuracy: 0.7603 - loss: 0.5839
Test accuracy: 0.7603999972343445
```


Predicted Label: Cat


Predicted Label: Cat

CHALLENGES

There are many challenges that can be at play when using image classification for a dog versus at cat.

- Breeds
- Lighting
- Poses

Overfitting and underfitting

AREAS OF IMPROVEMENT

- Increase Data Quality

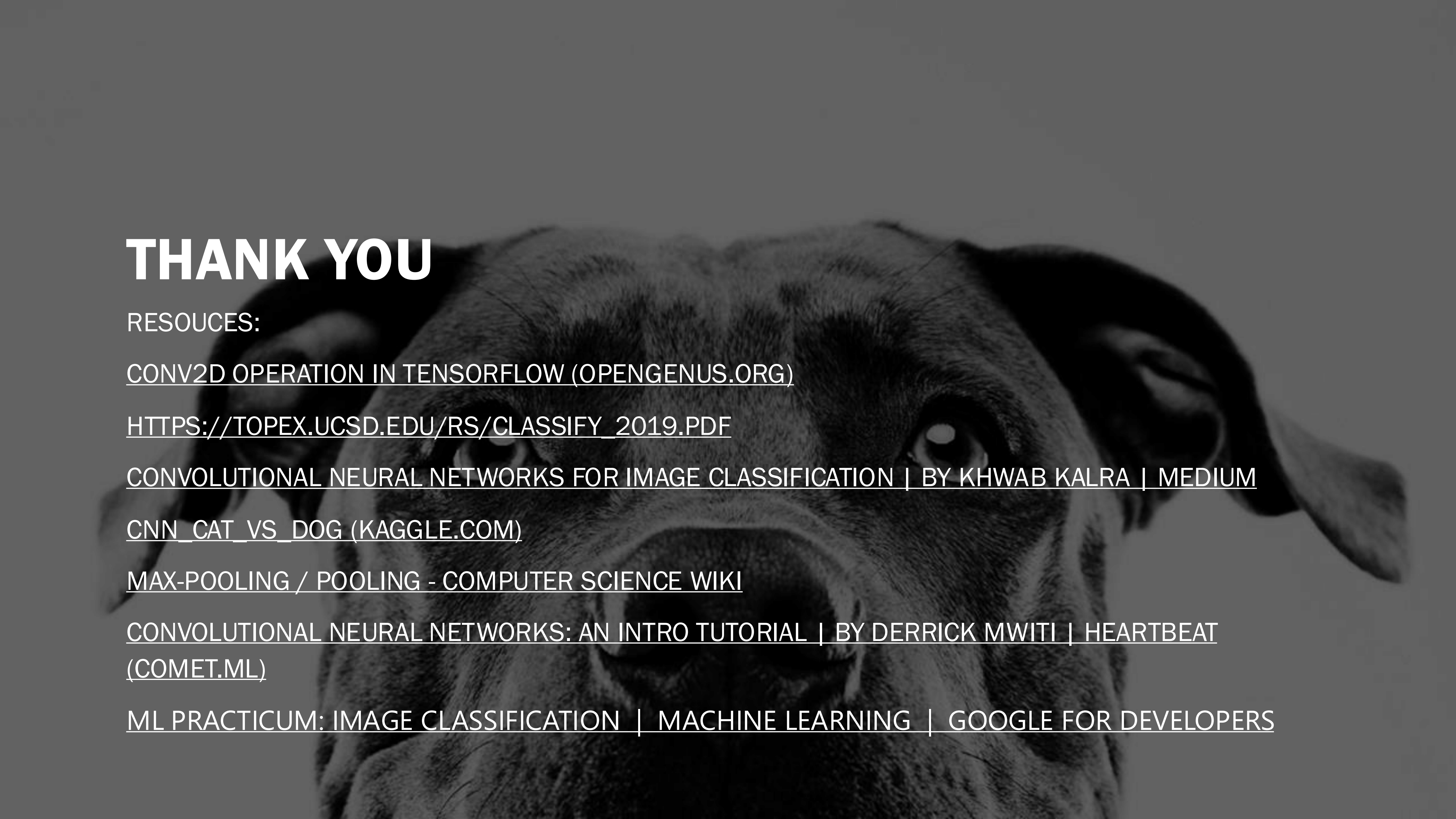
Ensure dataset is diverse, well-labeled, and representative of the target population.

- Regularization Techniques

Dropout is a regularization technique used in deep neural networks to prevent overfitting

- ADD MORE LAYERS
- Sizing of image
- Increase Epochs

Epochs are basically how many times you pass the entire dataset through the neural network.



THANK YOU

RESOUCES:

CONV2D OPERATION IN TENSORFLOW (OPENGENUS.ORG)

HTTPS://TOPEX.UCSD.EDU/RS/CLASSIFY_2019.PDF

CONVOLUTIONAL NEURAL NETWORKS FOR IMAGE CLASSIFICATION | BY KHWAB KALRA | MEDIUM

CNN_CAT_VS_DOG (KAGGLE.COM)

MAX-POOLING / POOLING - COMPUTER SCIENCE WIKI

CONVOLUTIONAL NEURAL NETWORKS: AN INTRO TUTORIAL | BY DERRICK MWITI | HEARTBEAT (COMET.ML)

ML PRACTICUM: IMAGE CLASSIFICATION | MACHINE LEARNING | GOOGLE FOR DEVELOPERS