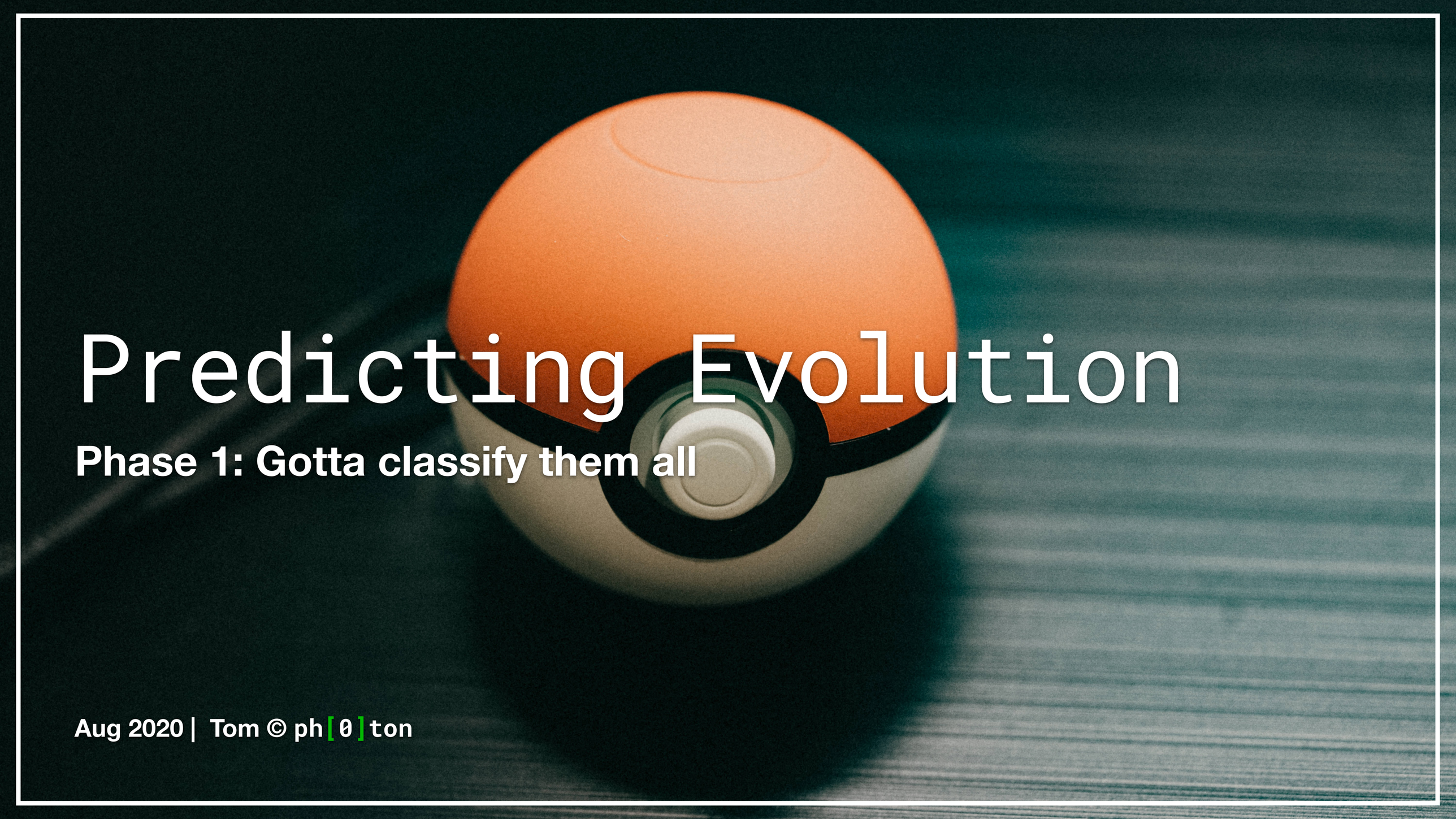


Predicting Evolution

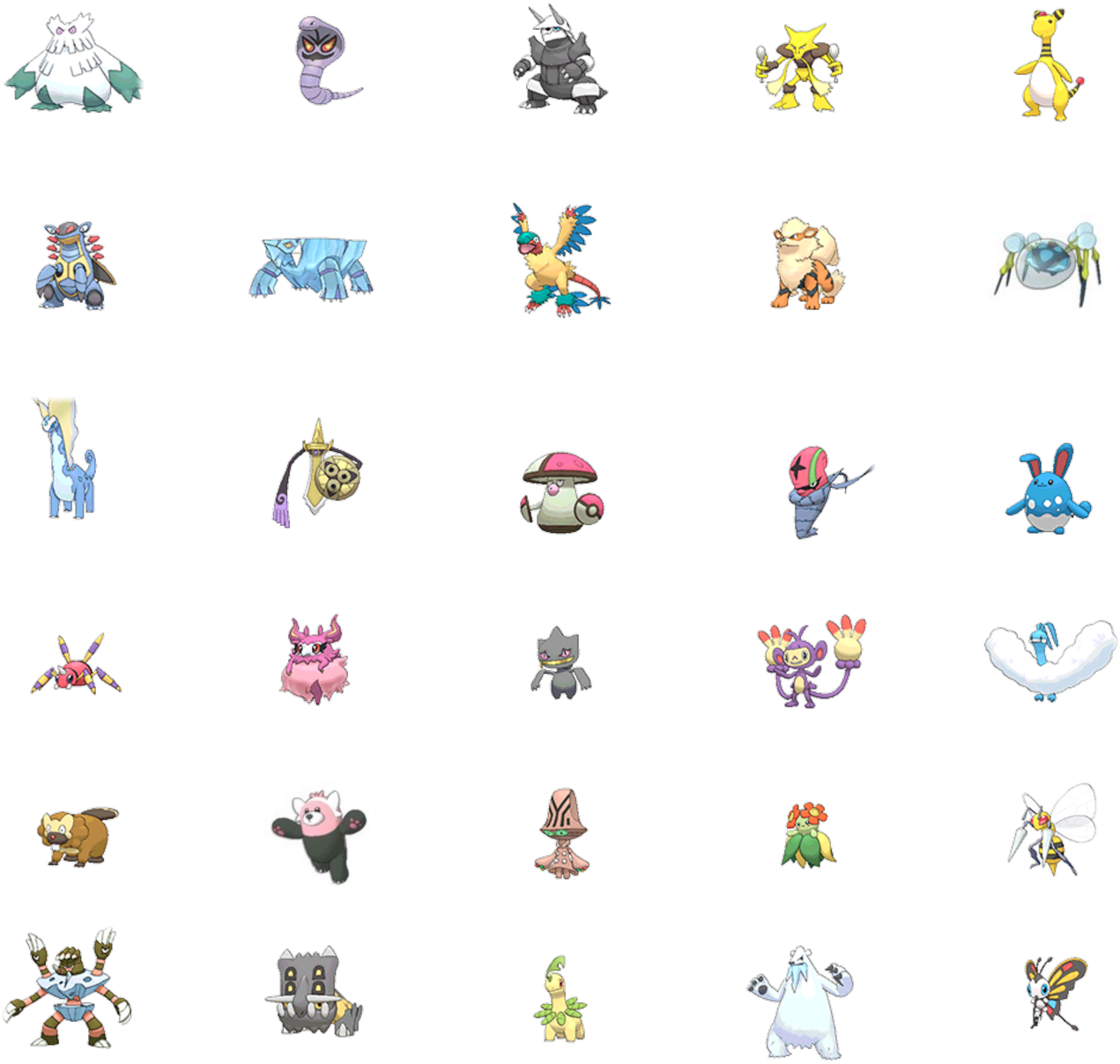
Phase 1: Gotta classify them all

Aug 2020 | Tom © ph[0]ton

Contents:

Purpose:

- Design and innovation firms are asked to come up with mock ups and pitches in a fast moving environment.
- This requires in-house or freelance resources that are taken away from current paying work.
- It’s possible that these inefficiencies could be addressed with machine learning.
- Can generating Pokémon be used as a proof of concept for rapid ideation and character prototyping?
- **This presentation is Phase 1 of this project - Classifying if a given Pokémon has evolved or not.**



NumPy

pandas

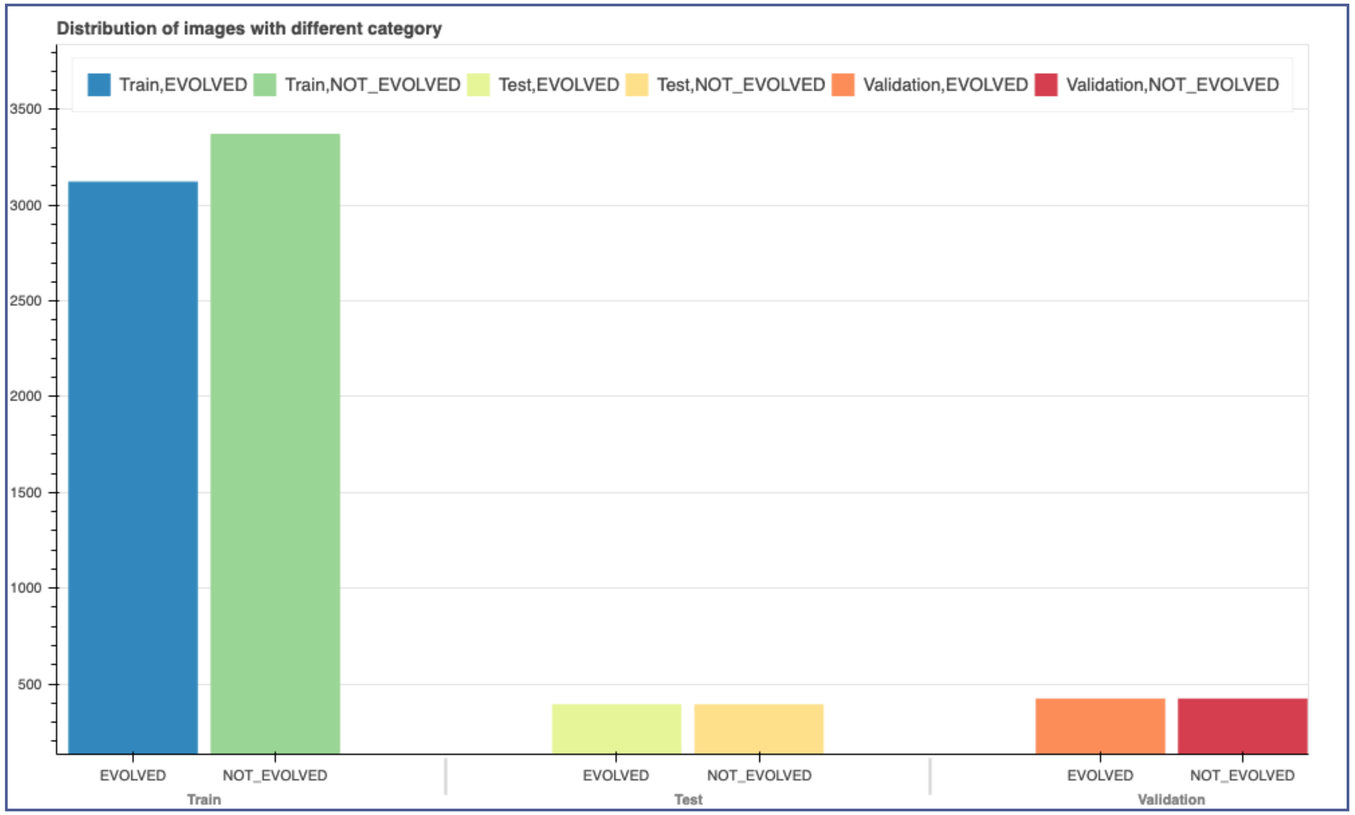
OpenCV

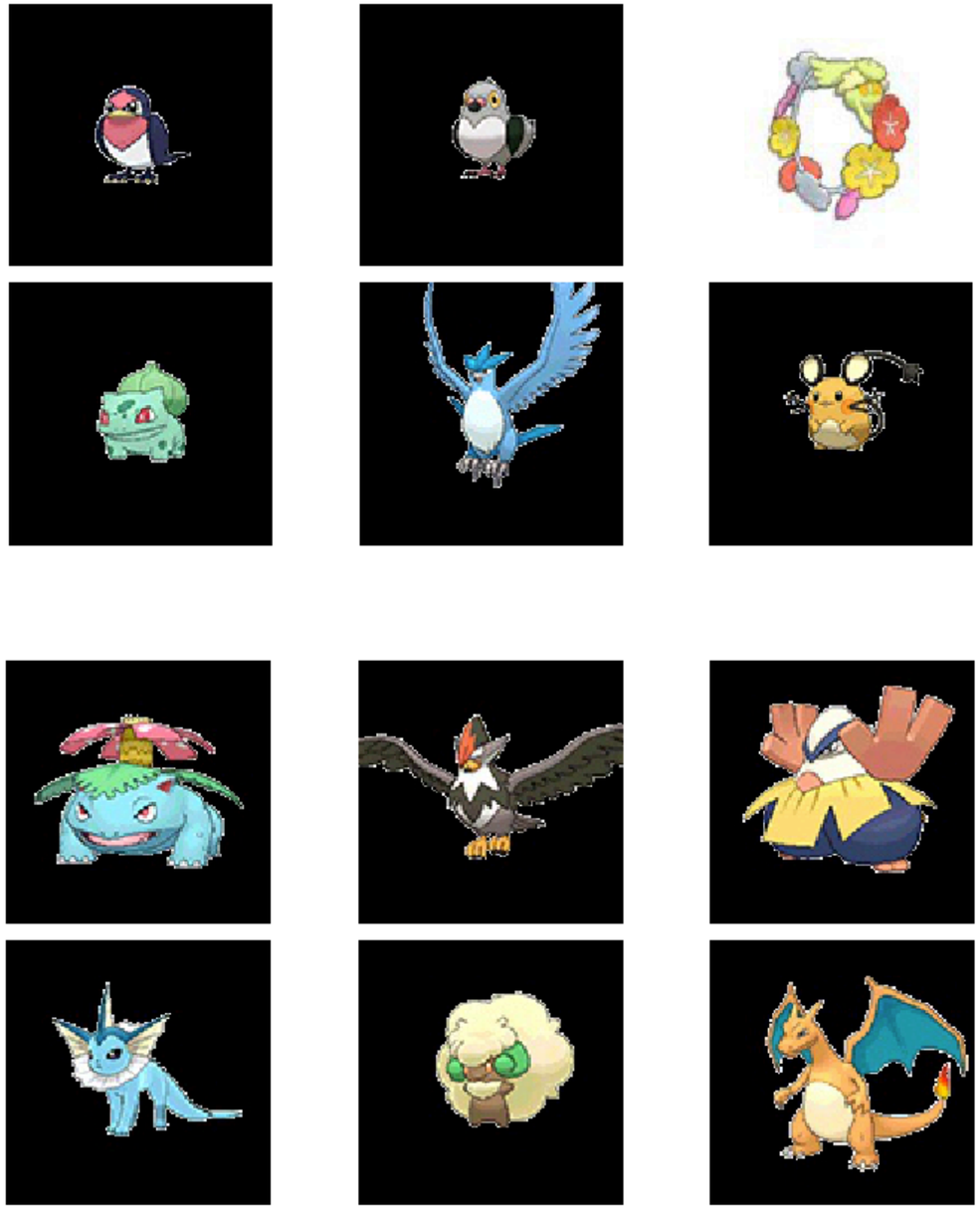
Data:

- Data obtained from Kaggle and can be found here:
  - https://www.kaggle.com/vishalsubbiah/pokemon-images-and-types
- Images of all Pokémon from generation 1 to generation 7
- Total images: 809
- Libraries used for modeling:
  - Numpy
  - Pandas
  - Matplotib
  - Bokeh
  - SKlearn
  - OpenCV
  - Tensorflow
  - Keras

Data (cont):

- Datasets were built by hand - sorting evolved and not evolved into two separate folders
- Images were reproduced 7 times in order to create the training data
- Low class imbalance retained as a result of this approach
- Images were converted from PNG to JPG to address issue with the transparency layer
- All images 120x120



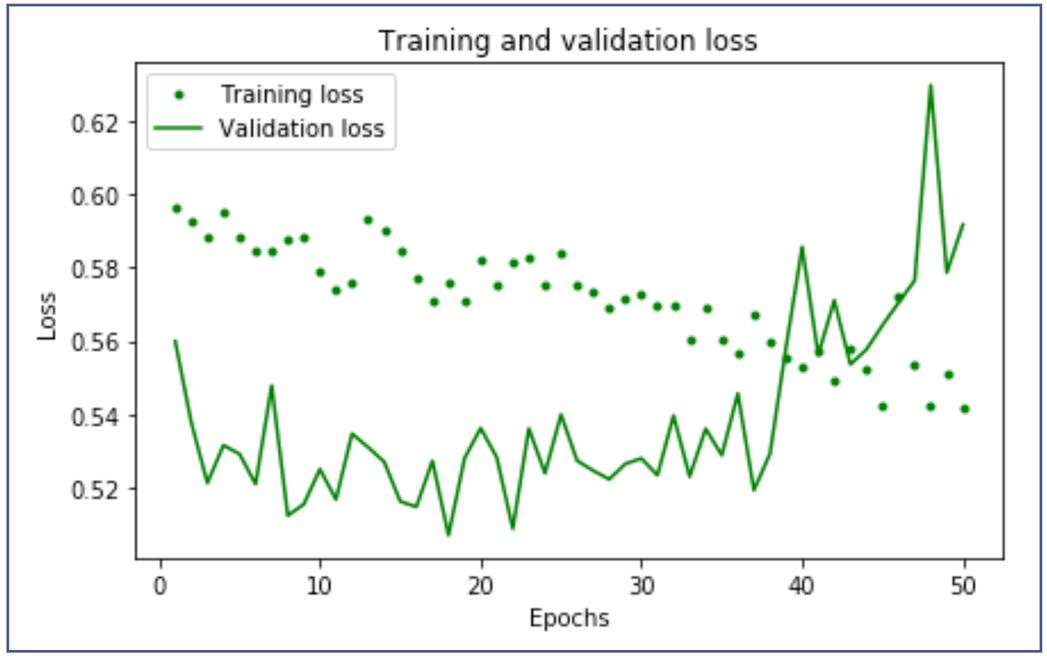


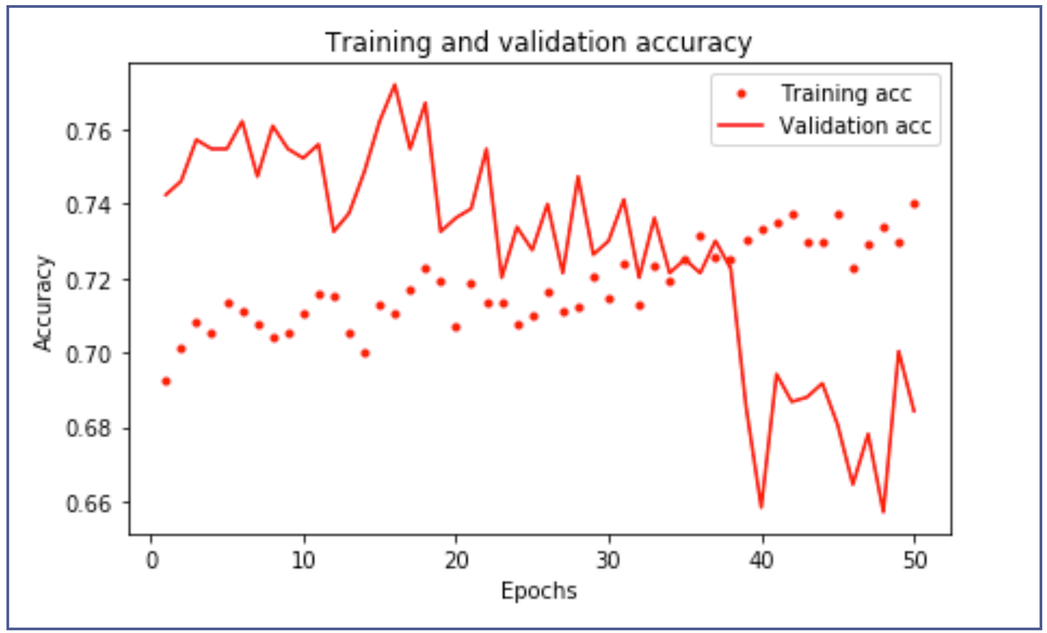
Data (cont):

- Pokémon that have not evolved typically carry youthful qualities and are generally smaller in size.
- Evolved Pokémon are typically mature in appearance and are larger.
- If using a Dummy Classifier, the dominant class (Not Evolved) will be predicted 52% of the time.
- This will serve as our benchmark for assessing model performance.

Base Model:

- Standard CNN.
- 2 Convolutional layers, 2 Pooling layers, Flatten layer, 2 Dense layers, 2 Dropout layers, and Output layer.
- Total parameters: 3,007,361
- Number of epochs: 50
- Heartbeat graphs show model not robust enough.





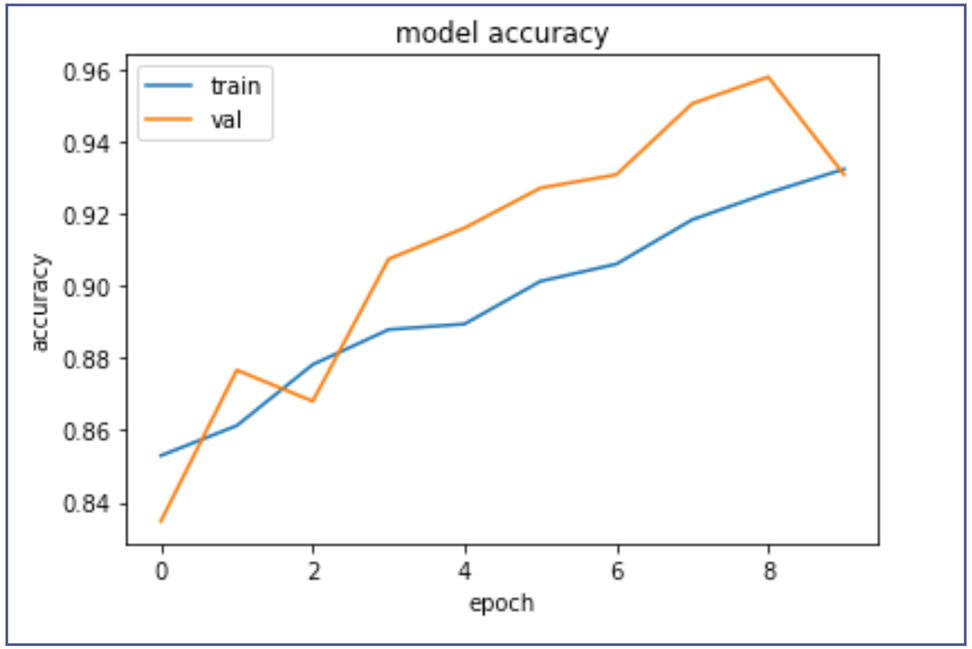
```
Here are the actual classes for each image
['images/TEST/NOT_EVOLVED/kricketot.jpg', 'images/TEST/EVOLVED/mienshao.jpg', 'images/TEST/NOT_EVOLVED/shinx.jpg', 'i
mages/TEST/EVOLVED/swanna.jpg', 'images/TEST/NOT_EVOLVED/nincada.jpg', 'images/TEST/EVOLVED/seaking.jpg', 'images/TES
T/NOT_EVOLVED/rotom.jpg', 'images/TEST/EVOLVED/sharpedo.jpg', 'images/TEST/NOT_EVOLVED/crabrawler.jpg']
Below are the predictions
```

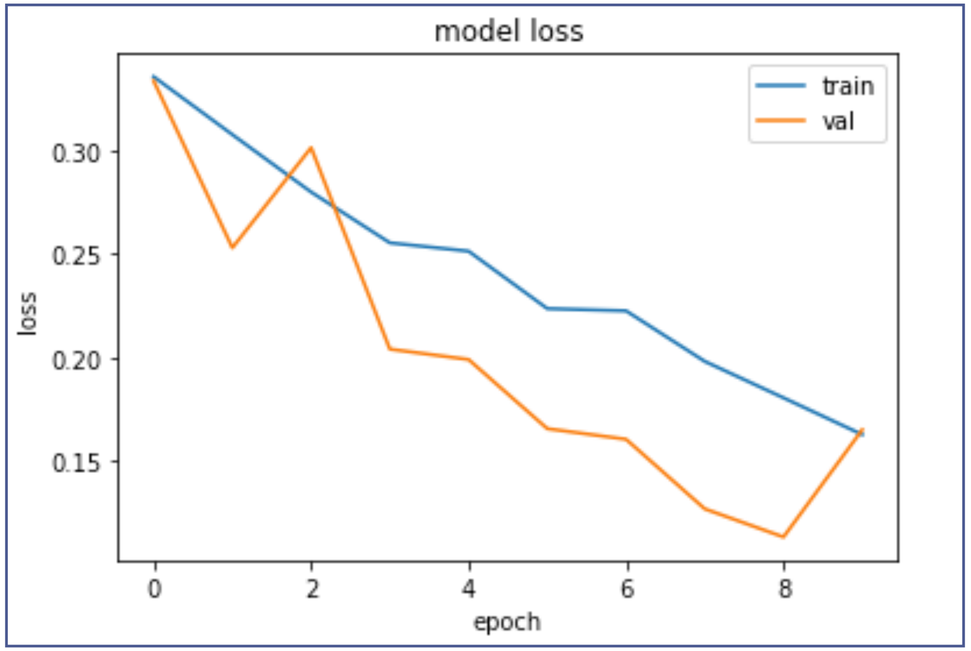
Base Model (cont):

- Accuracy: 74%
- Loss: 0.5407
- Optimizer: Adam
- Learning Rate: 0.001
- Steps per epoch: 300
- Better than the baseline metric (52%)

VGG16 Model:

- For the second Model, I used transfer learning to implement the VGG16 architecture and weights.
- The model was created by K. Simonyan, A. Zisserman and this version consists of 13 convolutional layers, and 5 max pooling layers.
- Total non-trainable parameters: 14,714,688
- I added four additional trainable layers consisting of 1 global average pooling layer and 3 dense layers.
- Total trainable parameters: 1,050,625
- Number of epochs: 10





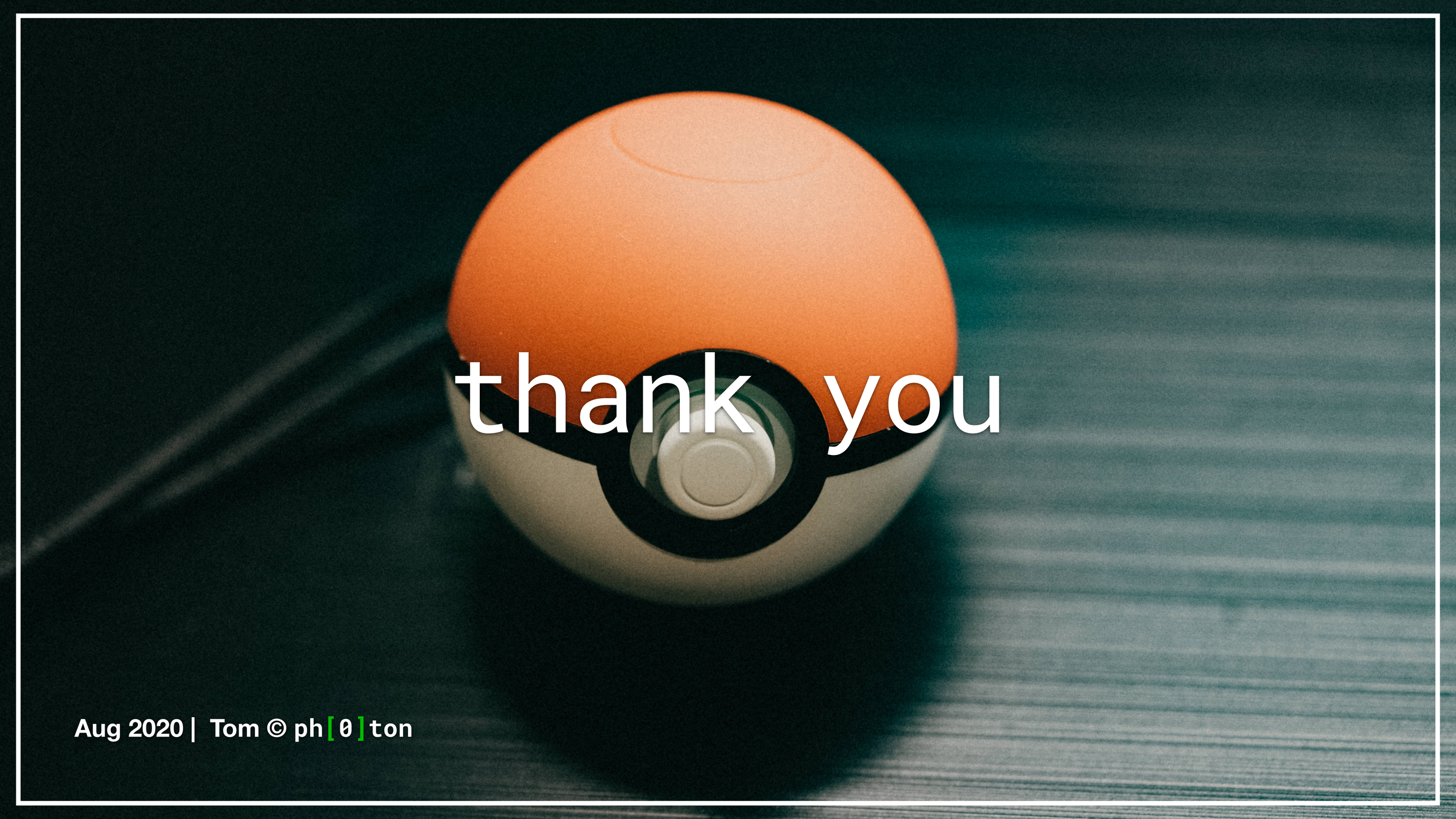
```
Here are the actual classes for each image
['images/TEST/NOT_EVOLVED/snivy.jpg', 'images/TEST/NOT_EVOLVED/delibird.jpg', 'images/TEST/NOT_EVOLVED/goomy.jpg', 'i
mages/TEST/EVOLVED/slaking.jpg', 'images/TEST/NOT_EVOLVED/furfrou.jpg', 'images/TEST/EVOLVED/tentacruel.jpg', 'image
s/TEST/EVOLVED/tranquill.jpg', 'images/TEST/EVOLVED/gloom.jpg', 'images/TEST/NOT_EVOLVED/doduo.jpg']
Below are the predictions
```

VGG16 Model (cont):

- Accuracy: 93%
- Loss: 0.1650
- Optimizer: Adam
- Learning Rate: 0.001
- Steps per epoch: 300
- Significantly better than the baseline metric (52%)

Next Steps:

- Take this dataset and build a GAN.
- Set up cloud computing for GPU powered training and generative output.
- Create custom tuning parameters according to Pokémon class.
- Build GUI that allows user to catch their very own Pokémon.
- Tell everyone I've ever met that I made it.



Aug 2020 | Tom © ph[0]ton