Identifying Military Aircraft

Half-Baked Holland Computing

Jaelle Kondohoma
Cameron Collingham
Joshua Carini
Kevin Nguyen

Motivation

- Other datasets have shown they can identify aircraft by name, if there potentially was a new aircraft in the skies which wasn't named yet it may be difficult to combat it or understand it's intentions.

- To mediate this we would create a Neural Network of transfer learning based off of the datasets to train the network to identify if a given picture of a military aircraft was either a fighter, bomber, or utility.

Given the small dataset of military aircrafts found on Kaggle: https://www.kaggle.com/a2015003713/militaryaircraftdetectiondataset , create a benchmark model for classification as well as a model that utilizes transfer learning to identify if a given picture of a military aircraft was either a fighter, bomber or utility aircraft. Utilize image augmentation to increase the size of training set.

Our Solution

Possible Applications

- Early warning systems for tense foreign relations
- Identifying experimental aircraft that have yet to have a designation

Experimental Setup (Benchmark)

- Settled on three aircraft classes
  - Fighter (takes priority detection)
  - Bomber
  - Utility

- Split dataset into train_ds, val_ds, and test_ds using image_dataset_from_directory with batch size 32

- Augment our aircraft dataset from approximately 2800 images to a total of roughly 32,000 using random flip and random rotation from keras sequential.

- Use the neural network found in this tutorial: https://keras.io/examples/vision/image_classification_from_scratch/ to set a benchmark for the error so we can compare that with the error we get from the pre-trained model Xception (trained on ImageNet).

- Fit our model using learning rate of 0.01 (1e-2) for 10 epochs

Experimental Setup (Transfer Learning)

- Settled on three aircraft classes
  - Fighter (takes priority detection)
  - Bomber
  - Utility

- Split dataset into train_ds, val_ds, and test_ds using image_dataset_from_directory with batch size 32

- Augment our aircraft dataset from approximately 2800 images to a total of roughly 32,000 using random flip random rotation from keras sequential.

- Use the neural network found in this tutorial: https://keras.io/guides/transfer_learning/#build-a-model which uses the pre-trained model Xception (trained on ImageNet).

- Freeze all the layers of Xception, and replace the top layer with a few of our own.

- Train this model with learning rate 1e-2

- Fine-tune by un-freezing Xception and using a small learning rate (1e-5).

- Use “training=False” to freeze the BatchNormalization layers. Otherwise, our training will destroy what the model has already learned.

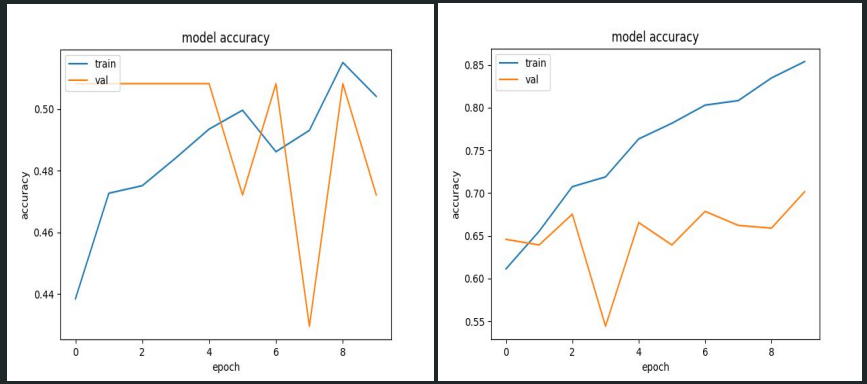
Expectations

- expect our results to be roughly between 40%-70% for accuracy.
- We expect transfer learning to be more accurate then any benchmark by a reasonable margin (20% - 30%).
- for the transfer learning, we used our benchmark model as the starting point to accurately compare our results.
- Slight overfitting due to small amount of images in dataset (hopefully solved by data augmentation)

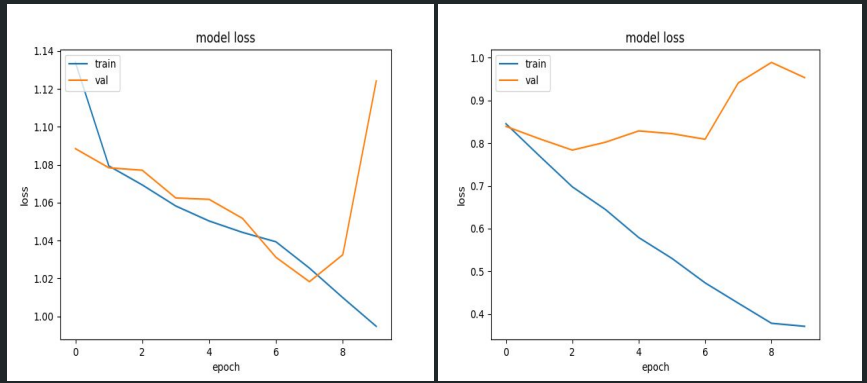
Experimental Results

The results we gathered were exactly what we expected.

During our training through transfer learning we managed to achieve up to 85% accuracy on recent tests.



Graph representing Accuracy. Left = benchmark, Right = Xception



Graph representing Measured Loss. Left = benchmark, Right = Xception

Discoveries

1. Our first experiments would not be accurate given the allotted epochs used during our benchmark and also because it stands as just a benchmark

    a. the benchmark was only supposed to provide a general idea of where our accuracies and losses would fall.

    b. Part of this is due to the rather small size of 2800 images (in training) in the dataset.

2. An excellent solution to these issues was using an image generator such as ImageDataGenerator provided by keras' image preprocessing suite

    a. The larger we can inflate our dataset the more accurate our results will be.

    b. Once out of benchmarking we will use Xception which is already trained and should produce more accuracy.

3. For further experiments we will increase the training epochs as well. The benchmark testing may have been good for a start but further testing should yield better accuracy and loss values.

Further Steps

- Solve overfitting within our transfer learning model
  - Early stopping
  - Adjusting dropout layer
  - Decreasing network size
- Images differ in perspective:
  - Some images are of a plane thousands of feet above the camera, others have multiple planes, at least one was the reflection on a pilot's face shield, and others were pictures of newspaper/magazine articles that had an image of a plane.
- Object detection and bounding boxes could focus the model on the important pixels.
  - If object detection is successful, use narrower classes (e.g. SR-71, F-16, U2)

Conclusion

- After comparing loss and accuracy between the two models, the results are clear:
    - Transfer learning is ideal for more compact datasets ,such as ours, while not diminishing the power of the image classification.
    - Although the accuracy for our transfer learning model was higher, due to overfitting, it does not mean our model was efficient.

- Starting with a pretrained initial model, we can make sure general features of the image are identified before moving on to more complex and specific features.

- Since Xception is already pre trained with many images, we were able to use their pretrained weights to supplement classifying our aircraft dataset.