Transfer learning on Dog Breed Classification

Dian Yu, Zhaokai Xu

Background

In recent years, deep learning has become increasingly popular in the fields of artificial intelligence. Among the considerably rapid improvement of deep learning technology, **fine-grained classification** such as **animal breed classification** and facial recognition is highly focused.

We decided to work on **dog breed classification** because dog is the most popular animal in the U.S.

Our goal is to help recognizing the lost dogs and return them to their homes. Our solution can also be applicable to other fine-grained classification problems, such as recognizing plant and other marine animal species.

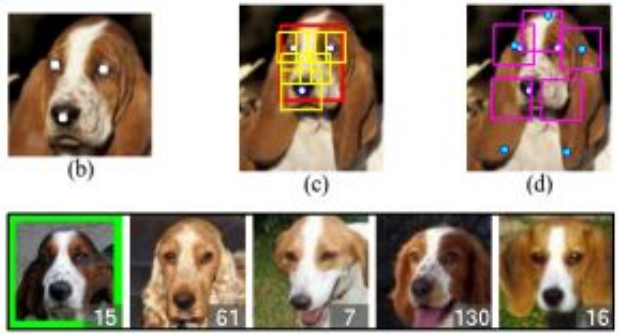
How machine learning help solving this problem

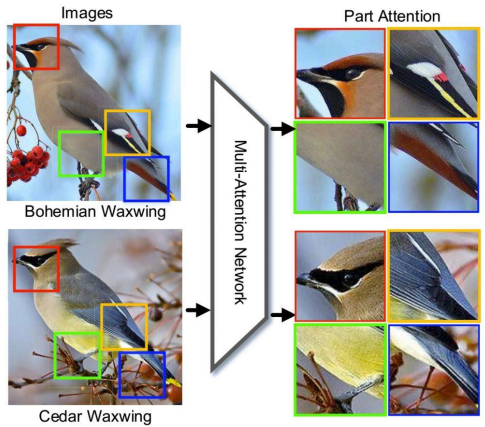
fine-grained image classification problem => not easy to solve with traditional ML methods.

Why? Image representation

Little difference between each category and sometimes large difference within the same category.

Neural network can extract the detailed features from images and capture the distinguishable features to identify particular species.

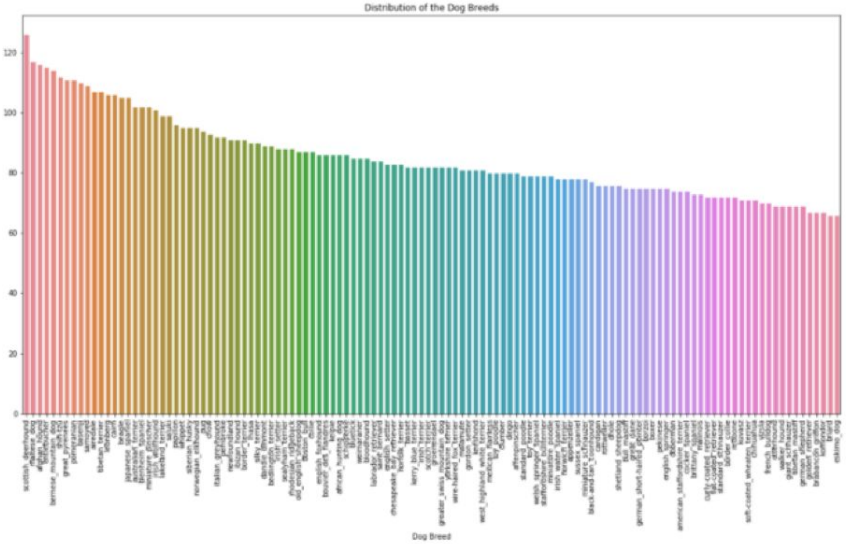


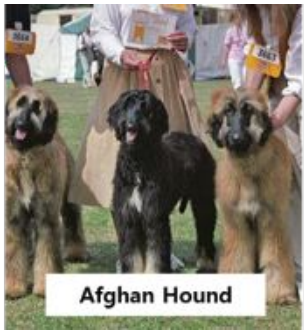


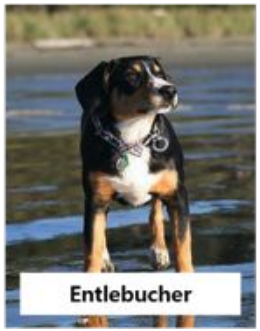
Dataset

Stanford dog dataset has **20580** images with **120** species. Dataset contains both basic pictures of dogs, as well as some variations of them.

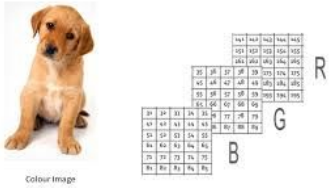
Dataset is resized to three partitions: 60/40/20 images per class.







Data Preprocessing




```
mean_nums = [0.485, 0.456, 0.406]
std_nums = [0.229, 0.224, 0.225]

chosen_transforms = {'train': transforms.Compose([
        transforms.RandomResizedCrop(size=256),
        transforms.RandomRotation(degrees=15),
        transforms.RandomResizedCrop(224),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        transforms.Normalize(mean_nums, std_nums)
]), 'val': transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize(mean_nums, std_nums)
]), 'test': transforms.Compose([
        transforms.Resize(256),
        transforms.CenterCrop(224),
        transforms.ToTensor(),
        transforms.Normalize(mean_nums, std_nums)
])
}
```

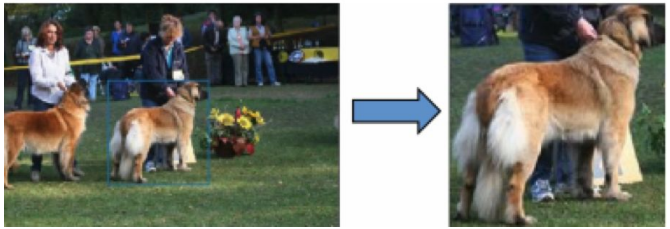
1 **One hot encoding:** each breed image falls into the a value from 0 to 119

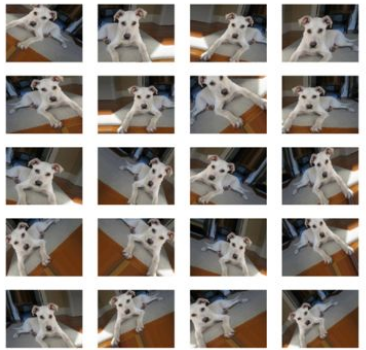
2 **Training : Validation : Testing = 4:1:1**

3 **Data augmentation**: randomly transform original images, such as cropping and rotating

4 **Data normalization:** normalize each RGB value with means and stds; resize to align with each model's input size

5 **Convention to tensor for GPU computation:** accelerate training process





Data Augmentation
- Rotating
- Scaling/Zooms
- Brightness
- Color Shift
- etc.

Model 1 --- Baseline CNN

For all the models, we trained 20 epochs for each of the three dataset 120*20, 120*40 and 120*60.

Firstly, we train a naive CNN model with 5 convolution layers with Batch Normalization, followed by max pooling and dropout layer to prevent overfitting. In the end, we applied a softmax layer to classify the dog breed with its label.

Filter size: 3x3

Loss function: CrossEntropyLoss

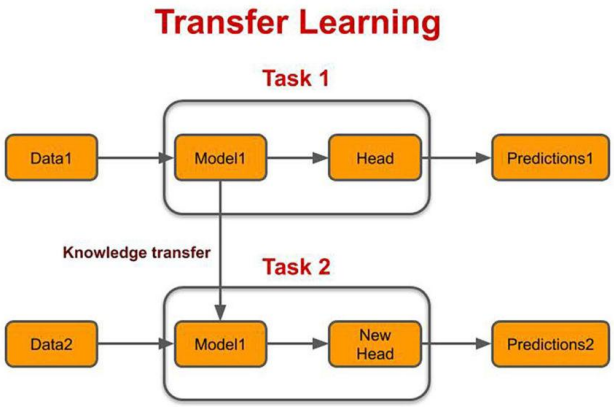
Optimizer: Statistical Gradient Descent

Transfer learning Models

Transfer learning focuses on **storing knowledge gained while solving one problem and applying it to a different but related problem.** EX Dot => cat

Weights and biases from previously trained on a dataset. Learned features are often transferable to different data. EX edges or horizontal lines

Save time and GPU resources




```
resnet = models.resnet34(pretrained=True)
num_ftrs = resnet.fc.in_features
resnet.fc = nn.Linear(num_ftrs, 120)

resnet = resnet.to(device)
criterion = nn.CrossEntropyLoss().to(device)
optimizer_ft = optim.SGD(resnet.parameters(), lr=0.00001, momentum=0.9)

resnet_best = train_model(resnet, criterion, optimizer_ft,
                          num_epochs=20, log_file_name='resnet_60_0dot00001')

print('-' * 10)
evaluate_model(resnet_best, log_file_name='resnet_60_0dot00001')
```

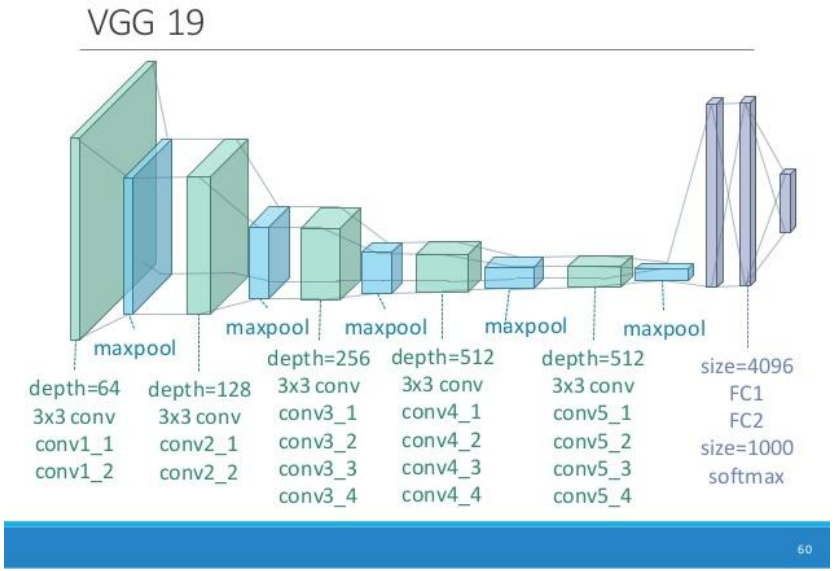
```
Epoch 0/19
----------
train Loss: 0.6193 Acc: 0.0079
val Loss: 0.6146 Acc: 0.0125
```

Model 2 --- VGG19

19 convolution layers, 5 max pooling layers and 3 fully connected layers.

Replace large kernel size (7, 11 from AlexNet) with multiple small 3*3 kernel for each layer

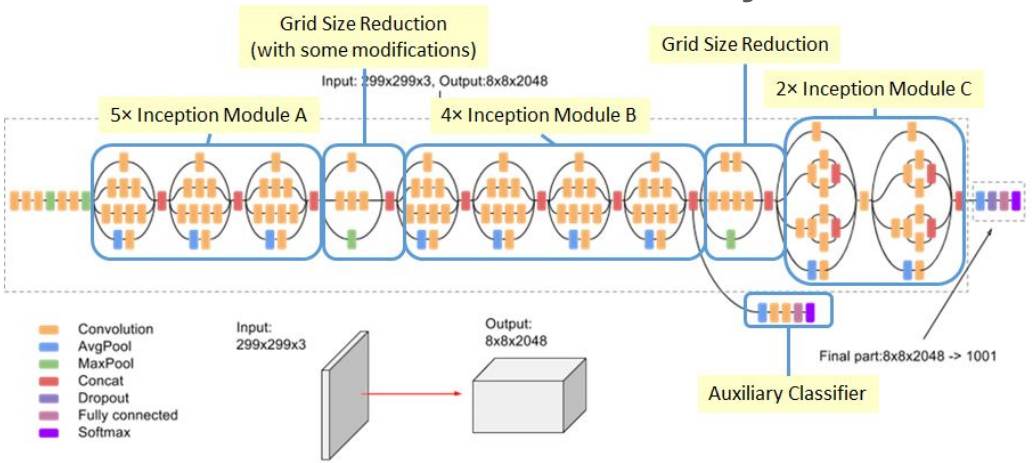
Better accuracy than CNN but longer time



Model 3 -- Inception V3

Originated from GoogleNet, but computation cost is only about 2.5 higher than that of GoogLeNet, and faster than VGG. A deep convolutional neural network combined 42 layers with **Auxiliary Classifier, Label-Smoothing Regularization, and Factorizing Convolutions techniques.**

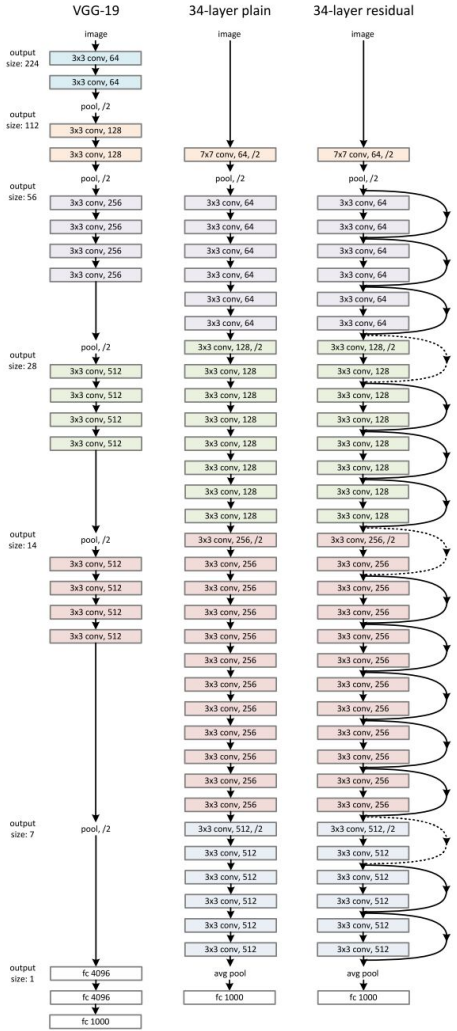
Feature extraction with a CNN + classification with both fully connected and softmax layers



Model 4 --- ResNet34

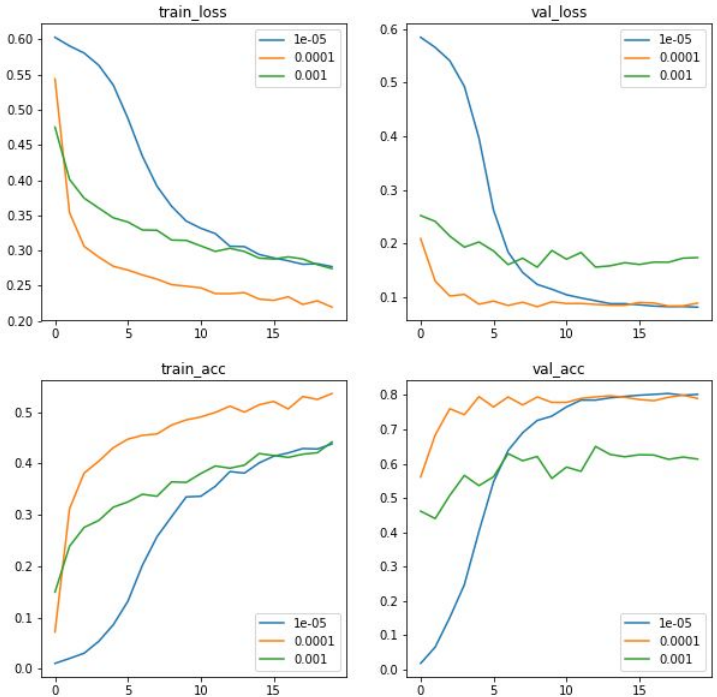
This model was the winner of ImageNet challenge in 2015. The fundamental breakthrough with ResNet was it allowed us to train extremely deep neural networks with 150+layers successfully.

Prior to ResNet training very deep neural networks was difficult due to the problem of **vanishing gradients**.

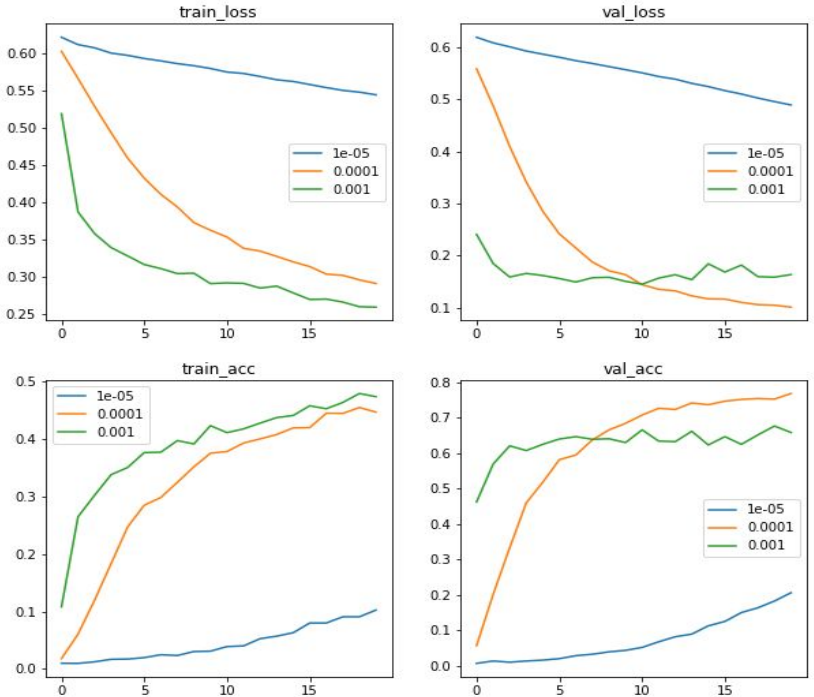


Hyperparameter Tuning - Learning Rate

VGG19

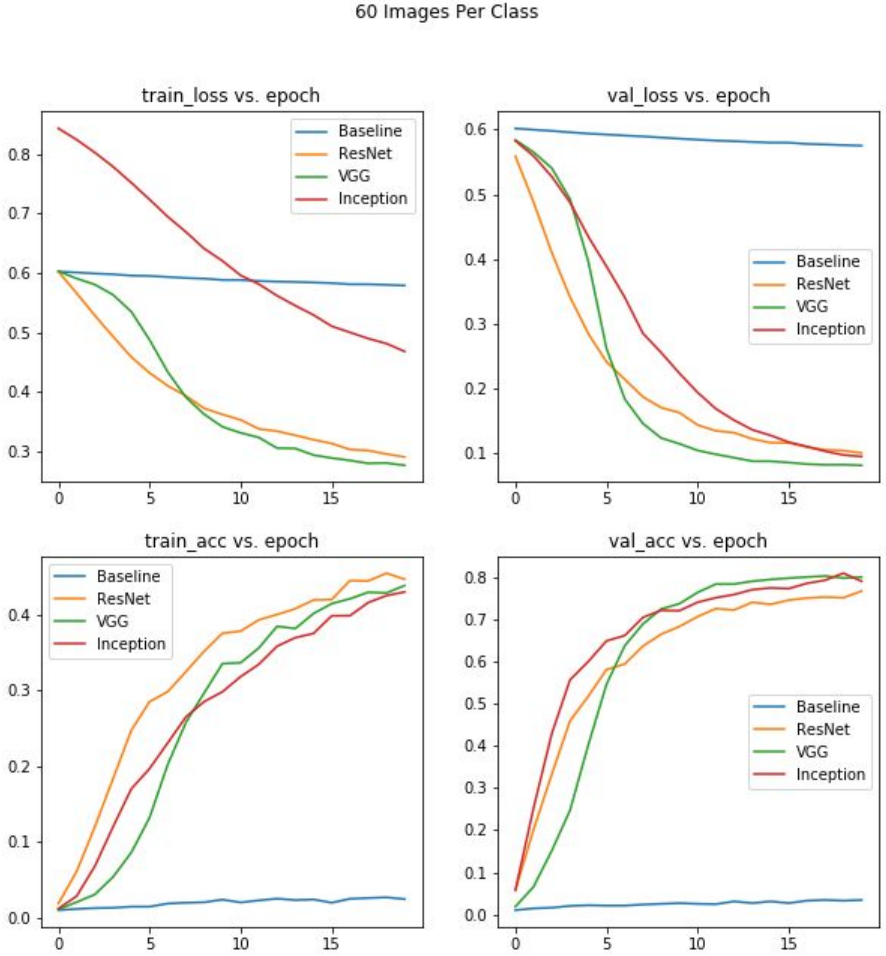


Resnet34



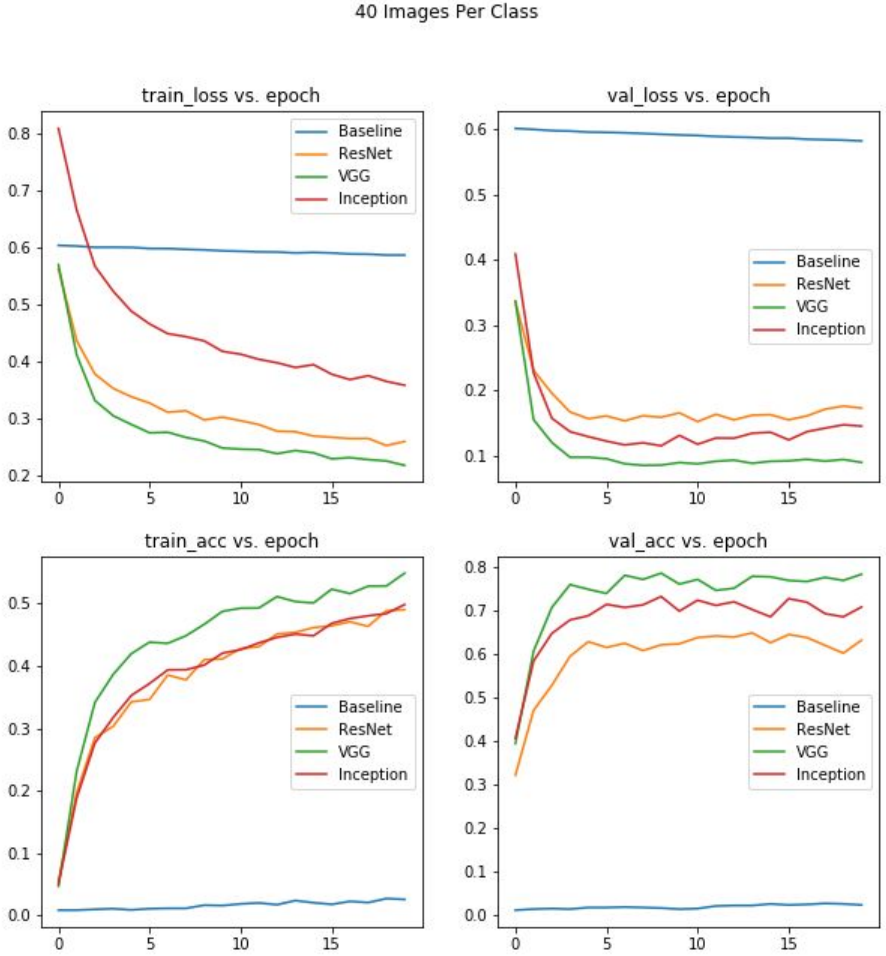
Results --

Models Performance on 120*60



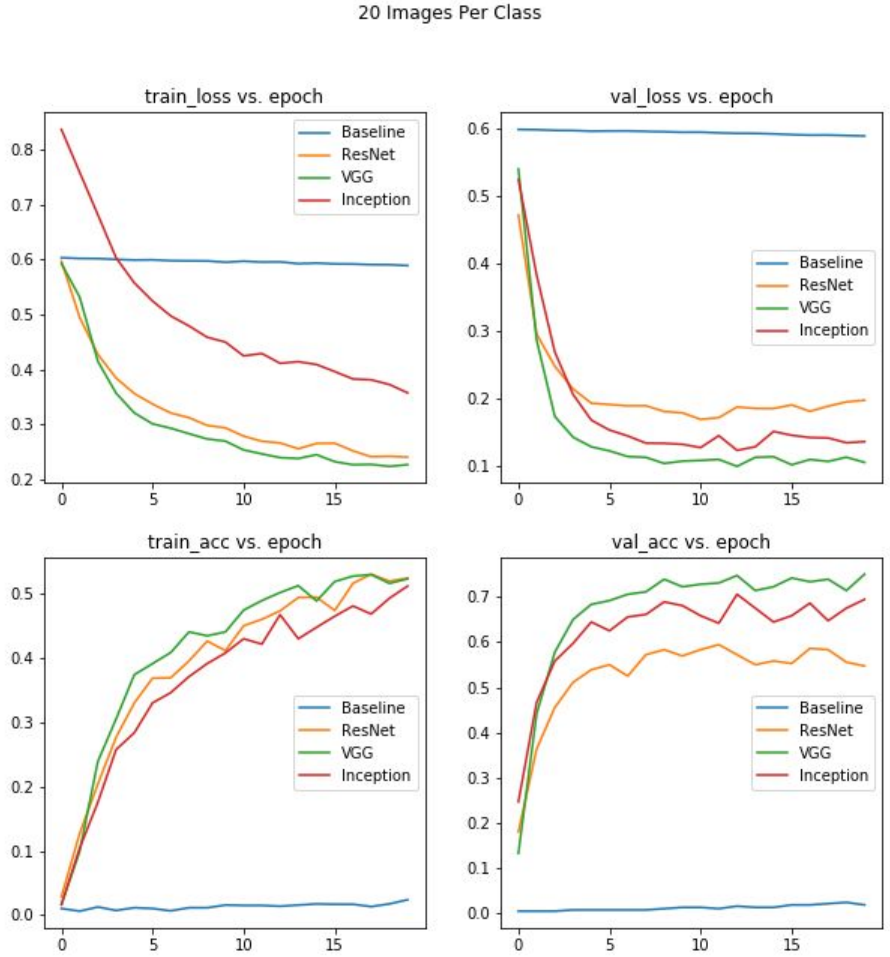
Results --

Models Performance on 120*40



Results --

Models Performance on 120*20



Results -- Test Accuracy

	Baseline CNN	ResNet34	Vgg19	Inception V3
120*20	0.02	0.61	**0.78**	0.71
120*40	0.03	0.69	**0.75**	0.73
120*60	0.03	0.75	0.78	**0.80**

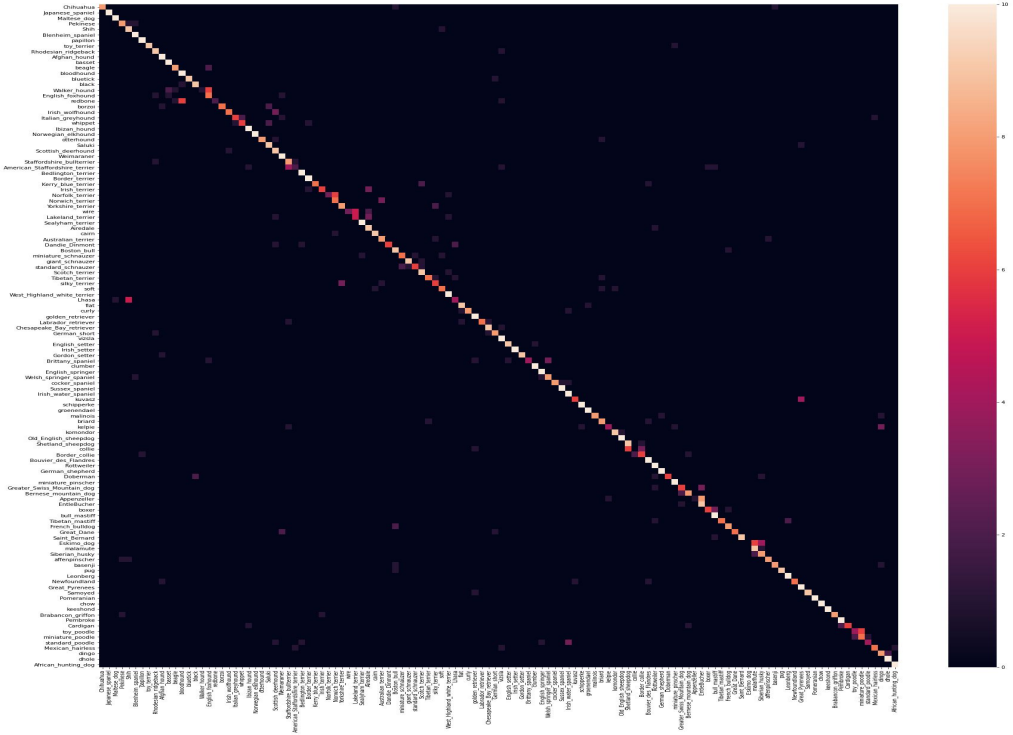
Inception V3 has the best performance on the largest dataset. All better than baseline

Vgg19 is more robust to smaller datasets.

ResNet34 takes the least time to train and has not-too-bad performance.

Number of image per class does not significantly matter in transfer learning due to pre-trained models

Results -- Confusion Matrix



Some breeds are harder to classify e.g. Lhasa(拉萨犬) vs. Shih-Tzu(西施犬)

Applications

1 categorize lost dogs for a better chance to be returned to their owners

2 help detect dog breed fraud

3 match dogs with people

Predict as --- beagle

Predict as --- German_shepherd

Just For Fun

Predict as --- pug

Predict as --- miniature_poodle

Predict as --- Afghan_hound

QA