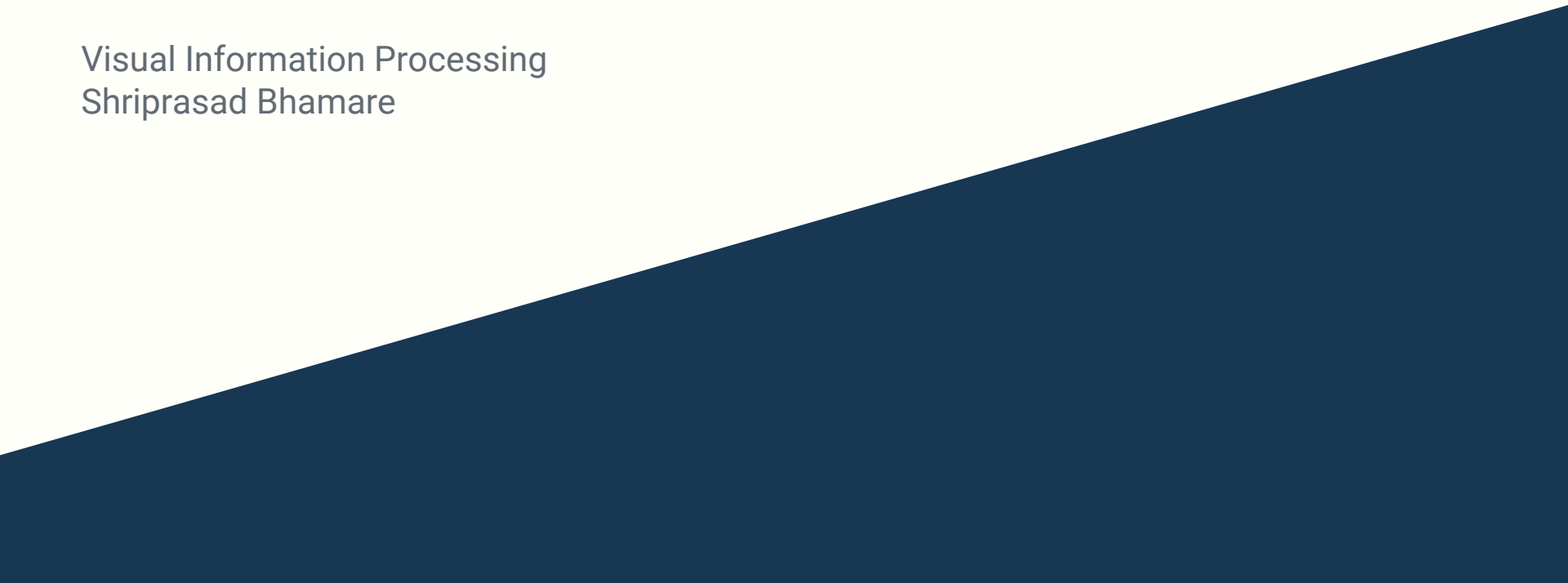


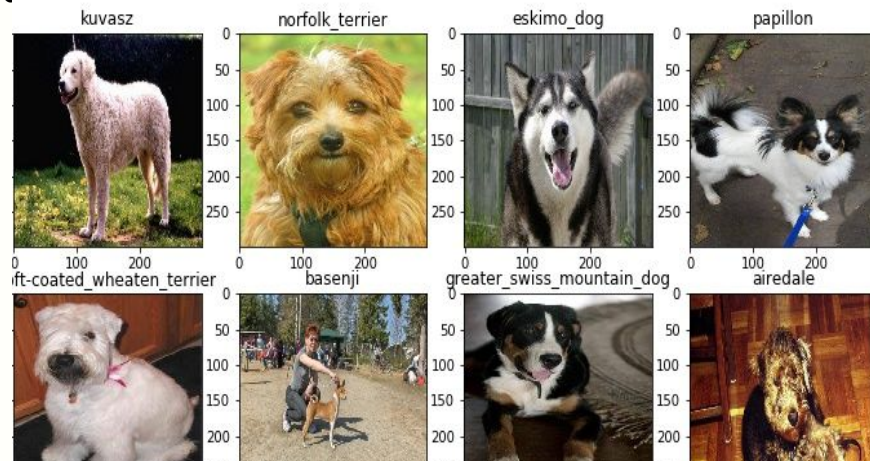
Dog Breed Classification using Convolutional Neural Network

Visual Information Processing
Shriprasad Bhamare

A dark blue diagonal gradient bar that starts from the bottom left and extends towards the top right, covering the lower half of the slide.

Project Overview

- Goal is to build a convolutional neural network (CNN)
- Classify the breed of dog from any user-supplied image.
- If the image is of a human and not a dog, the algorithm will provide an estimate of the dog breed that is most resembling.



Project Overview

- Human and Dog face detector
- CNN from scratch (Architecture +training + testing)
- CNN using Transfer Learning
 - VGG16
 - Resnet50
 - Inception
- Best accuracy giving model
- predictions

Dataset Description

- **Dog images:**

There are 133 total dog categories

8351 dog images having 6680 training dog images. 835, validation dog images, 836 test dog images.

<https://s3-us-west-1.amazonaws.com/udacity-ai/dog-project/dogImages.zip>

- **Human Images:**

The data set contains more than 13,000 images of faces collected from the web.

<http://vis-www.cs.umass.edu/lfw/lfw.tgz>

Technical Approach/Challenges:

- Python 3 and Keras with Tensorflow backend
- OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in images. OpenCV provides many pre-trained face detectors, stored as XML files
- Pre-trained ResNet-50, VGG-16, Inception model to detect dogs in images along with weights that have been trained on ImageNet, a very large, very popular dataset used for image classification and other vision tasks. ImageNet contains over 10 million URLs, each linking to an image containing an object from one of 1000 categories. Given an image, this pre-trained ResNet-50 model returns a prediction (derived from the available categories in ImageNet) for the object that is contained in the image.
- AWS EC2 p1.xlarge() instance is used considering the computational efficiency

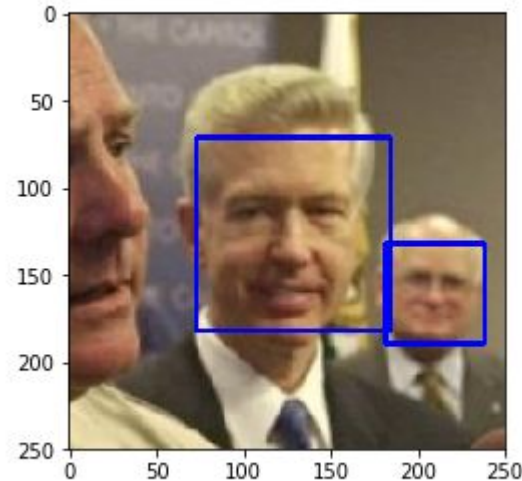
Human and Dog face Detection

Human Face Detection:

- Haar feature-based cascade classifiers
- OpenCV provides many pre-trained face detectors, stored as XML files.
- True if a human face is detected in an image and False otherwise

```
face_cascade =  
cv2.CascadeClassifier('haarcascades/haarcascade_  
frontalface_alt.xml')
```

Number of faces detected: 2



Dog face Detection

Detect Dogs

- pre-trained ResNet-50 model to detect dogs in images.
- weights trained on ImageNet
- Data Preprocessing
 - Keras CNNs require a 4D array
 - load the image and resize it to a square image that is 224×224 pixels
 - Image to an array, which is then resized to a 4D tensor.

```
img = image.load_img(img_path, target_size=(224, 224))
```

```
x = image.img_to_array(img) #3D tensor
```

```
return np.expand_dims(x, axis=0) #4D tensor
```

Data Preprocessing

- rescale the images by **dividing every pixel in every image by 255**.
- `train_tensors = paths_to_tensor(train_files).astype('float32')/255`
- `valid_tensors = paths_to_tensor(valid_files).astype('float32')/255`
- `test_tensors = paths_to_tensor(test_files).astype('float32')/255`

CNN Architecture

- `model.add(Conv2D(filters=16,kernel_size=(2, 2),strides=(1,1),padding='same',activation='relu',input_shape=(224, 224, 3)))`
- `model.add(MaxPooling2D(pool_size=(2, 2),strides=(1, 1),padding='same'))`
- `model.add(Conv2D(filters=32,kernel_size=(2, 2), strides=(1, 1), padding='same',activation='relu'))`
- `model.add(MaxPooling2D(pool_size=(2, 2), strides=(1, 1),padding='same'))`
- `model.add(Conv2D(filters=64, kernel_size=(2, 2),strides=(1, 1),padding='same',activation='relu'))`
- `model.add(MaxPooling2D(pool_size=(2, 2),strides=(1, 1), padding='same'))`
- `model.add(Dropout(0.3))` #to reduce the overfitting
- `model.add(GlobalAveragePooling2D())` #dimensionality reduction to generalize the model well
- `model.add(Dense(133, activation='softmax'))`

CNN model

Train

- Reduced Epochs to 5 (since it was taking too long)
- model checkpointing to save the model that attains the best validation loss.
- Lower the validation loss higher the accuracy
- Takes too much time to train the model

Test:

- Got very poor accuracy

CNN using Transfer Learning

- To reduce training time without sacrificing accuracy, I trained a CNN using transfer learning
- VGG16 model
- Optimizer :rmsprop ,sgd ,adam
- Train: 20 epochs ,batch size: 20

Accuracy: Improved

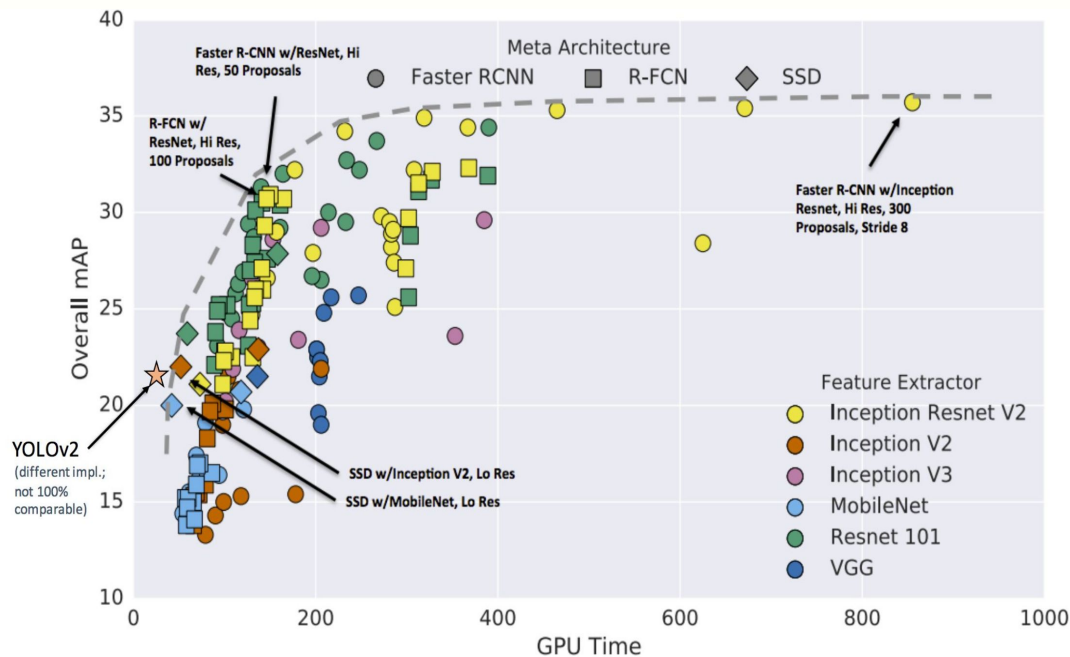
- Rmsprop : 46.77%
- Sgd : 50.48%
- Adam : 51.67%

CNN using Transfer Learning

- Resnet50 model
- Optimizer :rmsprop
- Train: 20 epochs ,batch size: 20

Accuracy: Improved

- Rmsprop : 79%



CNN using Transfer Learning

- Inception model
- Optimizer :rmsprop ,sgd ,adam
- Train: 20 epochs ,batch size: 20

Accuracy: Improved

- Rmsprop : 79.42%
- **Sgd** : **82.90%**
- Adam : 78.58%

Results

```
: make_prediction('myImages/Golden_retriever_05258.jpg')
```

```
Loading image...  
Extracting bottleneck features...  
Feeding bottleneck features into top model...  
Predicting breed...  
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!  
You look like a Golden retriever.
```



```
make_prediction('myImages/Bulldog_02845.jpg')
```

```
Loading image...  
Extracting bottleneck features...  
Feeding bottleneck features into top model...  
Predicting breed...  
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!  
You look like a Bulldog.
```



Results

```
make_prediction('myImages/alaskan.jpeg')
```

```
Loading image...  
Extracting bottleneck features...  
Feeding bottleneck features into top model...  
Predicting breed...  
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!  
You look like a Alaskan malamute.
```



```
make_prediction('myImages/Screenshot from 2018-11-
```

```
Loading image...  
Extracting bottleneck features...  
Feeding bottleneck features into top model...  
Predicting breed...  
Hello human!..Hello from the other side,I must have  
If you were a dog, you'd be a Chinese crested.
```



Results

```
make_prediction('myImages/12748073_1100499126647020_3213429309359239526_o.jpg')
```

Loading image...

Extracting bottleneck features...

Feeding bottleneck features into top model...

Predicting breed...

Hello human!..Hello from the other side,I must have called a thousand times..To tell you
If you were a dog, you'd be a American eskimo dog.



Results

- Could not detect dogs or humans in image.
- It doesn't predict resemblance to a dog breed for all human images



Sources

- Hsu, David, et. al, "Using Convolutional Neural Networks to Classify Dog Breeds"
- Wenting Shi, et. al, "Dog Breed Identification"
- Serge, et. al, "A Deep Convolutional Neural Network for Lung Cancer Diagnostic"