

**ISTANBUL TECHNICAL  
UNIVERSITY  
COMPUTER ENGINEERING DEPARTMENT**

**BLG 527E MACHINE LEARNING**

**CRN: 13817**

**Instructor: Zehra Çataltepe**

**Term Project  
December 20, 2017**

**Tuğrul Yatağan  
504161551**

**StudentName:** Tuğrul Yatağan

**StudentID:** 504161551

**Competition Name:** Dog Breed Identification

**kaggleID:** tugrulyatagan

**kaggleScore:** 4.46075

**kaggleScoreDate:** December 20, 2017

**methodNames:** Keras CNN, OpenCV DNN

**youtube videolink:**

## Running Code

dog.py file can be run with Python 2.7 interpreter. dog.py can be run directly without any command line arguments. tests.zip, train.zip, labels.csv.zip and saple\_submission.csv.zip files must be extracted into same directory with dog.py

```
python dog.py
```

It took about 5 minutes to run. It is tested on Ubuntu 16.04 environment. Following Python modules needs to be installed to run dog.py;

```
pip install numpy
```

```
pip install pandas
```

```
pip install cv2
```

```
pip install sklearn
```

```
pip install opencv-contrib-python
```

```
pip install imutils
```

```
pip install tqdm
```

```
pip install keras
```

## Dataset Description

Dataset contains images of 120 breeds of dogs. Training set has 10222 images and test set has 10357 images. Images has not only dogs but also surrounding environment and other objects like humans. Some images have multiple dogs with same breed in it. All images have different surrounding and different zoom and angle degree to dogs. All images have different resolution, brightness and dog scale. Most common and least common dog breeds in training set are:

Most common breeds:

scottish_deerhound	126
maltese_dog	117
afghan_hound	116
entlebucher	115
bernese_mountain_dog	114

Least common breeds:

golden_retriever	67
brabancon_griffon	67
komondor	67
briard	66
eskimo_dog	66

Dataset has similar dog breeds that classifying is challenging.

Brittany



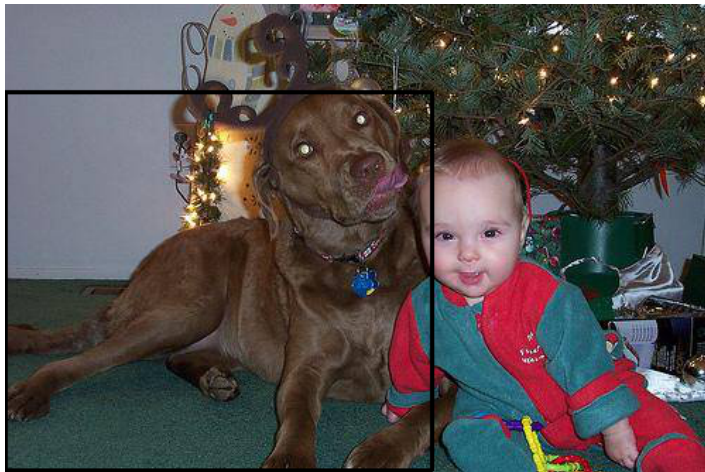
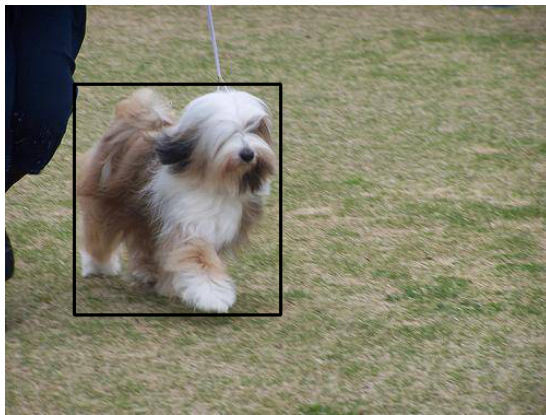
Welsh Springer Spaniel



## Methods Used

Dataset has images that have dogs in some portion of the image. So, I focused on detecting dog position in the image first because training on whole image can decrease classification accuracy since only a portion of the images has valuable data for dog breed classification. For this purpose I use OpenCV 3.3's deep neural network (dnn) library to detect dog positions on images. OpenCV dnn module has pretrained data(MobileNet SSD) to classify various objects like vehicles, humans, dogs and cats. This module use depthwise separable convolution to classify objects. The general idea behind depthwise separable convolution is to split convolution into a  $3 \times 3$  depthwise convolution followed by a  $1 \times 1$  pointwise convolution.

Example results of using OpenCV dnn MobileNet SSD to detect dogs:



To classify dog breeds, I use Keras CNN library on detected dog images from first step. CNN filters were halved in each conv layer. There were max pooling layers between them. There were also dropout layers to prevent overfitting. Summary of the model:

Layer (type)	Output Shape	Param #	INPUT
=====			
conv2d_1 (Conv2D)	(None, 150, 150, 16)	208	CONV
<hr/>			
max_pooling2d_1 (MaxPooling2D)	(None, 75, 75, 16)	0	POOL
<hr/>			
conv2d_2 (Conv2D)	(None, 75, 75, 32)	2080	CONV
<hr/>			
max_pooling2d_2 (MaxPooling2D)	(None, 37, 37, 32)	0	POOL
<hr/>			
conv2d_3 (Conv2D)	(None, 37, 37, 64)	8256	CONV
<hr/>			
max_pooling2d_3 (MaxPooling2D)	(None, 18, 18, 64)	0	POOL
<hr/>			
dropout_1 (Dropout)	(None, 18, 18, 64)	0	DROPOUT
<hr/>			
flatten_1 (Flatten)	(None, 20736)	0	FLATTEN
<hr/>			
dense_1 (Dense)	(None, 500)	10368500	DENSE
<hr/>			
dropout_2 (Dropout)	(None, 500)	0	DROPOUT
<hr/>			
dense_2 (Dense)	(None, 120)	60120	DENSE
=====			
Total params: 10,439,164			OUTPUT
Trainable params: 10,439,164			
Non-trainable params: 0			

I use 150 x 150 as a input image size for CNN since larger than this requires too much RAM.

## **Experiment Results**

Detection of dog positions on images with OpenCV dnn gives very accurate dog positions on images. There is no data to verify dog positions but I randomly choose 20 images and OpenCV dnn can detect dog positions on 19 images.

I split training dataset into 70%-30% for training and training validation set.

Training accuracy for CNN without dog detection is 10.2% which is very low. I got Kaggle score of 4.78474 with this method.

Training accuracy for CNN with dog detection preprocess is 13.7% which is not a quiet good achievement. I got Kaggle score of 4.46076 with this method. I assume (from Kaggle score) test accuracy for this method is a little better than first one.

## **Discussion**

Result shows that dog breed classification with simple CNN method is not sufficient enough to get high classification accuracy. Dog detection before processing CNN has little effect on increasing CNN classification accuracy. Only simple CNN is not adequate to this kind of challenging image classification.