# Dog Breed Classifier Proposal

by Bobby Mander

Feb 14, 2020

## Domain Background

Computer vision in machine learning is one of the original machine learning problem domains. Early experiments (all the way back to 1959) in the field involved images and their effect on the eye. These experiments established how the eye processes images, including edge detection and simplification. As technology and the AI field in general matured into the 1980's, new techniques were developed and tested including powerful software neural networks. Soon afterwards a specialized neural network, called the convolutional neural network (CNN) was developed that sought to replicate the processes that occur in the human eye to process images. This involved creating a hierarchical representation of the image from the simple to the complex so that images could be understood and filtered for the brain. CNNs quickly gained power thanks to Moore's Law and access to exponential computing power at a fraction of the cost with the advent of the Cloud. With machine learning competitions sponsored by Kaggle and others, many research teams got involved to develop better and better computer vision systems. One subfield in the computer vision space is image classification, trying to detect what an image contains with some level of confidence and granular detail. That is what we will be focusing on in this project.

## Problem Statement

We will be classifying images using machine learning in this project. Specifically we will be attempting to classify whether an image contains a dog and if it does, what breed of dog is present. If no dog is present, we will also attempt to detect if a human face is present and if it is, what kind of dog that face most resembles. We will solve this problem using convolutional neural networks (CNNs) trained for the purpose.

## Datasets and Inputs

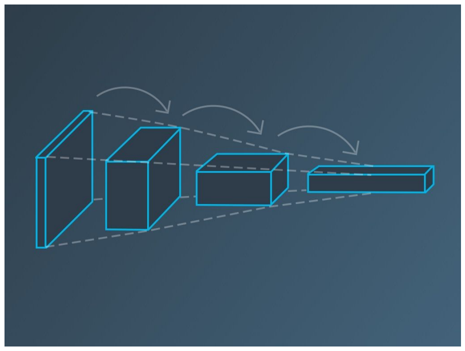
This project will use data obtained from Udacity's course database. The two datasets are a set of ~8,000 dog images and a set of ~10,000 human face images. These datasets can be obtained via the links in the references. These images are of various sizes and resolutions. We will need to normalize this data before it can be used for training or testing. The datasets are split between training, validation, and test sets. The datasets are also labeled with the appropriate breed since this is a supervised learning problem. We will train with the training set only. While training we will use the validation data set to confirm the applicability of the model and find the best performing model to use later against the test set. The main data is the dog dataset used to train the CNN. The human dataset will be tested and used with a standard OpenCV face detector so no training will be required with that dataset.

## Solution Statement

In order to solve the classification problem and identify the dog breed amongst a set of 133 different breeds, we will construct a CNN from scratch. We will first collect and normalize the input images so that they can be fed into the CNN for training. We will then construct a model for the CNN including the number of convolutional layers, maxpooling layers, and linear layers. The size of each layer will also be determined. Finally a rich set of hyperparameters will be configured to setup the training of the network. Once training is complete using the training set to adjust the network's weights and the validation set to confirm applicability of the network and save the optimal performing network, we will use the test data to determine how well the CNN predicts the dog breed.

In addition, we will use another solution technique to get even better results. We will employ transfer learning using the well established VGG16 CNN for image classification. This CNN was developed by the VGG group at Oxford University and contains 16 convolutional layers and 3 fully connected layers. It was trained extensively with the ImageNet database, a popular online database of thousands of everyday images with classification labels. With the pre-trained VGG16 CNN, we will use transfer learning to tweak it and apply it to our classification problem with the goal of getting even better results than our CNN from scratch.

The diagram below shows how a CNN is structured going from an image input at the left, through several convolutional layers in the middle that deepen and flatten the data till the linear classifiers at the right.


It's a Siberian Husky!

## Benchmark Model

In terms of benchmarks, random guessing will result in 0.8% chance of getting the dog breed correct. There are a variety of other CNNs available but these would have to be applied to the dog breed classification problem. The VGG16 CNN will be our benchmark model even though it has not been applied to this problem. We will adapt it using transfer learning to apply it and achieve superior results. We will attempt to achieve 10% or greater accuracy using the CNN from scratch and 60% or greater accuracy using the VGG16 with transfer learning.

## Evaluation Metrics

The evaluation metric we will use is the accuracy of the dog breed prediction. Simply this is the number of dog breeds predicted accurately versus the total predictions. Again we will aim for 10%+ accuracy for CNN from scratch and 60%+ for the VGG16 with transfer learning.

## Project Design

We will be using PyTorch in AWS SageMaker Jupyter notebooks to implement the project. The project involves several phases:

1. Data preparation. We will load the data and normalize it. PyTorch offers many normalization techniques including resizing, cropping, flipping, and scaling pixel values. We will experiment with many of these transformations to find the best one. We will visualize the data after the transforms to ensure that the data being fed into the model looks appropriate.
2. Model design. We will try to determine the best way to structure the CNN including the number of layers and the size of each layer. Some experimentation and trial and error will be necessary as part of the next phase.
3. Training. Training the CNN will take the most time as several hyperparameters will need to be chosen and configured. Hyperparameters including learning rate, loss function, optimization function, number of epochs, data transforms, layer sizes, etc… Hyperparameter optimization will be key for obtaining the best results.
4. Testing. This involves running the finally trained model against the test data and evaluating the results.

Finally we will put the project's pieces together to operate on any image and determine if the image has a dog, then what breed of dog or if the image has a human face, then what breed of dog that face resembles.

## References

1. Demush, Rostyslav "A Brief History of Computer Vision (and Convolutional Neural Networks)" https://hackernoon.com/a-brief-history-of-computer-vision-and-convolutional-neural-networks-8fe8aacc79f3
2. Dog image dataset: https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
3. Human face image dataset: https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip
4. Popular Networks "VGG16 – Convolutional Network for Classification and Detection" https://neurohive.io/en/popular-networks/vgg16/
5. ImageNet: http://www.image-net.org/
6. Base Jupyter Notebook: https://github.com/udacity/deep-learning-v2-pytorch/blob/master/project-dog-classification/dog_app.ipynb
7. Free Dog Images: https://www.pexels.com/search/dog/
8. Shams S "Understanding TensorFlow" https://machinelearningmedium.com/2018/01/02/understanding-tensorflow/