Use Convolutional Neural Networks to Identify Dog Breeds Proposal

**Domain Background:**

This project is one of Udacity's specific projects for completing Machine Learning Engineer nanodegree. It is proposed to build a pipeline to process supplied images to identify the breed of the dog in the image using Convolutional Neural Network. If a human is detected in the image, the code will identify the resembling dog breed.

**Problem Statement:**

Overall, the project is to develop an algorithm for a dog breed identification app.

To be specific, the following tasks need to be finished:

1. Write a human detector function using OpenCV's implementation of Haar feature-based cascade classifier.
2. Write a dog detector function using a pre-trained VGG16 model.
3. Create a CNN to classify dog breeds from scratch and attain a test accuracy of at least 10%.
4. Create a CNN to classify dog breeds using transfer learning and attain a test accuracy of at least 60%.
5. Write an algorithm that accepts a file path to an image and first determine whether the image contains a human, dog or neither. Then use the model develop in step4 to predict the dog breed.

**Datasets and Inputs:**

The dataset is provided. It includes human images and dog images. The human images can be downloaded from https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip. The dog images can be downloaded from https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip.

The dog dataset is used to train the CNN model. Both human and dog images are used to test the performance of the dog detector and the human detector. In addition, the train dataset, valid dataset and test dataset is split in advance. Train dataset contains about 80% of the total population and both valid and test dataset contain about 10% of the total population. All three datasets contain images for all 133 dog breeds.

**Solution Statement:**

The CNN developed from scratch has an architecture listed below:

Conv2d -> MaxPool -> Conv2d ->Maxpool -> Conv2d -> Maxpool -> Conv2d -> MaxPool -> Conv2d -> MaxPool -> Flattened -> FC -> FC

To utilize transfer learning, a pre-trained ResNet 50 model is used and a fully connected layer is added as the final classifier.

**Benchmark Model & Evaluation Metrics:**

For a CNN developed from scratch, an accuracy of at least 10% is requited.

For a CNN utilizing transfer learning, an accuracy of at least 60% is required.

**Project Design:**

The project is designed already. The features of some classical CNN models ( including ResNets, Inception Network, 1x1 Convolution) are learned before implementing the project.