Capstone Proposal

The project's domain background — the field of research where the project is derived:

The projects domain background is in image classification. The field of deep learning developed as a branch of machine learning with the goal of mimicking the human brain. Convolutional neural networks are commonly used in deep learning to solve a whole range of problems, with image classification being a common example where it excels vs other techniques. The applications of deep learning are growing as more and more research is conducted and more powerful GPUs are developed. Image classification in deep learning has progressed greatly over the past few years. Some transfer learning models offer super human performance in image classification these days, meaning, a deep learning CNN can classify an image with greater accuracy than a human.

A problem statement — a problem being investigated for which a solution will be defined:

Create a CNN to classify dog breeds:
Create an image classifier that can accurately classify if an image is a human or a dog, and accurately determine what breed that dog is, or what dog breed the human most looks like.

The datasets and inputs — data or inputs being used for the problem:

Data sets:

- This problem uses dog and human image data sets.
- Dog data set:
  - The dog data set has classified images of dog breeds and comes pre-split into training (6680 images), validation (835 images), and test (836 images) sets. The dog images are randomly sized with dogs in various environments and settings and positions.

- Human data set:
  - The human dataset is composed of 13233 250x250 named human images in 5750 folders. Each image has humans in various environments and settings and positions.

- Inputs:
  - Pre trained CNNS (Resnet 18 and VGG16):
    - These pre trained neural networks will be used as transfer learning inputs. Both these inputs were developed using very large image data sets containing many different images.
  - OpenCV pre-trained face detector.
    - This will be used to detect faces in images to assist in the classifier.

A solution statement — the solution proposed for the problem given:

The proposed solution for this project will be to develop both a convolutional neural network from scratch and a transfer learning convolutional neural network. The transfer learning CNN should have the best performance and will be used in creating the human/dog breed detection algorithm.

A benchmark model — some simple or historical model or result to compare the defined solution to:

There are two benchmark models, one for the CNN from scratch and one for the transfer learning CNN. The CNN from scratch must perform with an accuracy equal to or greater than 10%. This will prove that the model is working because it exceeds the probability of a random guess which would be around 1% accuracy. The transfer learning model must have an accuracy of equal to or greater than 60%.

A set of evaluation metrics — functional representations for how the solution can be measured:

A loss function will be used to evaluate the model. The loss function will measure any mistakes between the predicted and true class. Backpropagation will then quantify how bad a specific weight is in making a mistake. A training vs validation loss and a model vs test loss evaluation metric will be used to measure the models performance. This loss metric will measure the CNNs performance by accounting for uncertainty of the predicted value based on how much it varies from the actual label. This will be used to access how the model is performing after each training epoch by comparing the training loss to the validation loss. If validation loss is decreasing then model will save, if the training and validation diverge, it will not save, this will prevent overfitting. The model will then go through the simlar steps with the model and test set, finally giving a test loss and test accuracy percentage.

An outline of the project design — how the solution will be developed and results obtained:

Project Outline:

The project will involve data gathering, data transformation, and the choice of either a CNN from scratch or a transfer learning CNN, then training and validating and testing the models accuracy. Finally, the model will be used in an algorithm to determine dog breeds and human dog breed lookalike's against any input images.

1. Gather data:
    a. Classified dog and human data set zip files

2. Create a human face detector and a dog detector:
    a. Use the openCV face detector to create a function for finding faces.
    b. Use VGG16 transfer learning to classify dogs in images
    c. Create a dog detector algorithm

3. Training, validation, Test Split:
    a. If the data is not already separated, split it into its training, validation, and test data sets.

4. Transform to Tensor Data:
    a. Convert all the data sets to tensor data to be used by pyTorch.
        i. Data augmentation
        ii. Data normalization
        iii. Data resizing
    b. Create tensor data sets for training, validation, test split.

5. Convolutional Neural Network (Two Methods):

    a. Convolutional Neural Network from Scratch:
        i. Create a CNN from scratch using pyTorch and the converted tensor data
    b. Transfer learning Neural Network:
        i. Load a ResNet18 pre trained model
        ii. Replace the final layer of the frozen pre trained model with the new image tensor data.
        iii. This should have the greatest accuracy and will be used in the final algorithm.

6. Train and Validate the model:
    a. Save the model if the validation loss is less than the training loss through each epoch.

7. Test the model:
    a. Test to see if the model accuracy is acceptable against the test set.

8. Dog/Human classifier algorithm:
    a. Test the trained model against random images to determine if the dog breed classifier is working as expected. First input an image, transform it to tensor data, pass through the model and print out the dog breed.
        i. If a human is detected, print the image and classify closest resembling dog breed.
        ii. If a dog is detected, print the image and classify closest resembling dog breed.
        iii. If neither is detected, print error.

References:

1. PyTorch Documentation https://pytorch.org/
2. PyTorch Torchvision https://pytorch.org/docs/stable/torchvision/models.html
3. Project repo: https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification
4. Udacity: https://www.udacity.com/
5. FastAI Practical Deep Learning for Coders: https://course.fast.ai/
6. Log Loss Description: http://wiki.fast.ai/index.php/Log_Loss