Machine Learning Engineer Nanodegree
Capstone Proposal
M.Sc. Juan Carlos Kuri Pinto

Domain Background

The Dog Breed Classification project uses convolutional neural networks to classify photos of dogs, humans, and other animals into one of the 133 dog breeds that are contained in the dataset. Convolutional neural networks (LeCun and Bengio, 1995) are a special type of neural network that are translation-invariant and can find important invariant patterns in spatio-temporal signals. Convolutional neural networks are a powerful Deep Learning technique (Goodfellow, Bengio, and Courville, 2016). Deep Learning is a branch of Machine Learning (Mitchell, 1997) that uses neural networks to make machines learn salient and invariant patterns from data. Machine Learning is a branch of Artificial Intelligence (Russell and Norvig, 2020) whose algorithms are capable of learning from experience. And Artificial Intelligence is a branch of Computer Science whose intelligent algorithms try to make machine smarter.

The Dog Breed Classification project can be solved because it is just a classification problem. Photos of dogs, humans, and other animals are shown to convolutional neural networks and then the most similar dog breed is activated. The abstract problem, classification of images into labels, is already solved. There is nothing new. However, why should we solve this specific project? One goal could be to demonstrate that scientific research can be funny and can attract young researchers into the field of artificial intelligence. Another goal could be to extract important statistics about dog breeds, i.e. salient and invariant visual patterns for each dog breed. However, those statistics are hidden in the somewhat black-box nature of neural networks. It is a little bit difficult to visualize such multidimensional patterns. But it is possible.

My personal motivation to select this project is that I love animals. I could observe the behavior of animals endlessly. While, some scientists prefer to look at the stars, planets, black holes, and the universe, I'm rather passionate about the inner universe of our minds and the minds of animals. I've been studying neuroscience and artificial intelligence for many years. And such experience gave me the tools to understand animal behavior in a very deep way. I always learn new things from my observations and experiences. For example, those beautiful photos of dogs show a lot about the personalities of dogs. I wonder if the convolutional neural networks of this project can extract such personality traits. I guess the answer is yes. However, analyzing such artificial insights goes beyond this simple classification project. But this project is a great opportunity to think about those deep questions.

Problem Statement

Basically, we have 2 problems here. First, classifying a photo into 3 categories: Human, dog, and other. Second, classifying a photo into one of 133 dog breeds. The Dog App should tell if a photo has a human, a dog, or another entity. Then Dog App should tell the dog breed that mostly resembles the entity in such photo.

The problem of classifying photos into categories like dog breeds, or "dog and human", is quantifiable, measurable, and replicable. The problem is quantifiable and measurable because classifiers can be trained and the fitness of classifiers can be measured by a loss function, the metric. Moreover, some classifiers can return a number that measures of how similar photos and dog breeds are. Finally, the problem is replicable because the metric, the loss function, is not only applicable to the patterns in the training dataset; but the metric is also applicable to the patterns in the validation dataset and in the test dataset. Once trained, the convnet can also give a number that measures how similar a photo is to some particular dog breeds. And the photo does not need to have a dog in it. It could be a photo of a human or a photo of another animal.

Datasets and Inputs

The dataset of dogs was provided by Udacity and is already divided in 3 folders: train, valid, and test. The folder **train** has 133 subfolders (1 folder for each dog breed) that contain 6,680 images of dogs. The folder **valid** has 133 subfolders that contain 835 images of dogs. And the folder **test** has 133 subfolders that contain 836 images of dogs. In the dog project, the inputs are the photos and the outputs are the dog breeds. This

dataset will serve to train the benchmark (a convnet trained from scratch) and the ResNet-50 with pretrained weights and transfer learning. This dataset will also serve to test the accuracy of the human face detector and the dog detector.

Dataset of Dogs:
https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip

The dataset of humans was provided by Udacity and has 5,749 folders containing 13,233 images. In the dog project, it is used only for testing purposes, given that nothing is trained with this dataset. It is only useful to test the accuracy of the human face detector and the dog detector.

Dataset of Humans:
https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip

Solution Statement

Basically, we have 2 problems and their solutions. First, classifying a photo into 3 categories: Human, dog, and other. Second, classifying a photo into one of 133 dog breeds. Both classification problems are solved by using convolutional neural networks. And the human face detector is solved by using Haar cascades and boosting (Viola and Jones, 2001). The Dog App tells if a photo has a human, a dog, or another entity. Then it tells the dog breed that mostly resembles the entity in such photo.

The algorithm of Haar cascades and boosting uses Haar features to create weak classifiers capable of detecting human faces in groups of pixels. In each iteration of the cascade, weak classifiers do a vague job at classifying patterns. However, this vague job is compensated by the boosting algorithm, which is capable of creating a strong classifier out of weak classifiers like Haar features. How so? Each weak classifier do its lazy job and its mistakes are passed to the next level in the cascade. In this way, difficult-to-learn patterns are constantly retrained until the Haar features create a strong classifier.

Convolutional neural networks, or convnets, are a special type of neural network that are translation-invariant and can find important invariant patterns in spatio-temporal signals. Convnets have 2 types of layers: Convolutional layers and fully-connected layers. Convolutional layers are capable of finding translation-invariant patterns or features throughout the visual field. They represent visual patterns in a very efficient way because much less parameters are needed. Whereas, fully-connected layers have much more parameters that are location-specific, not location-invariant.

The problem of classifying photos into dog breeds and its solution using convnets are quantifiable, measurable, and replicable. The solution is quantifiable because the parameters or synapses that represent patterns in the convnet can be trained through the gradient descent technique applied to the loss function. Neurons are connected in a special neural architecture and generate outputs. Such outputs or predictions are compared to the ground-truth labels in order to compute a loss function, which is a metric of how adapted the convnet is. Hence, this solution is also measurable. Moreover, the solution is replicable because the metric is not only applicable to the patterns in the training dataset; but the metric is also applicable to the patterns in the validation dataset and in the test dataset. Once trained, the convnet can also give a probability distribution based on softmax that tells how similar a photo is to some particular dog breeds. And the photo does not necessarily have a dog in it. It could be a photo of a human or a photo of another animal. The convnet is capable of extrapolating patterns and making visual analogies, which is absolutely amazing.

Benchmark Model

The benchmark model is a convnet trained from scratch, whose neural architecture is:

```
Net(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1))
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1))
  (conv3): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1))
  (conv4): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1))
```

```
  (drop): Dropout(p=0.25, inplace=False)
  (fc1): Linear(in_features=9216, out_features=2304, bias=True)
  (fc2): Linear(in_features=2304, out_features=576, bias=True)
  (fc3): Linear(in_features=576, out_features=133, bias=True)
)
```

And its forward method is:

```
    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = self.pool(F.relu(self.conv3(x)))
        x = self.pool(F.relu(self.conv4(x)))
        x = x.view(x.size(0), -1)
        x = self.drop(F.relu(self.fc1(x)))
        x = self.drop(F.relu(self.fc2(x)))
        x = self.fc3(x)
        return x
```

Evaluation Metrics

Both the loss and the accuracy are metrics to compare the benchmark model with the final solution, the ResNet-50 with pretrained weights and transfer learning.

The benchmark model produced its best results at the epoch 23:
train_loss=0.005845, train_acc=89.31%
valid_loss=0.104890, valid_acc=14.97%

The best benchmark model was saved and produced the following results in the test dataset:
test_loss=0.120632, test_acc=15.07% (126/836)

The final solution, the ResNet-50 with pretrained weights and transfer learning, produced its best results at the epoch 24:
train_loss=0.000749, train_acc=99.19%
valid_loss=0.006908, valid_acc=87.54%

The best ResNet-50 model was saved and produced the following results in the test dataset:
test_loss=0.009719, test_acc=85.05% (711/836)

It is a really humbling experience to try to do your best at creating the best convnet possible with the best hyperparameters and yet the results pale in comparison to the results of the ResNet-50 with pretrained weights and transfer learning.

Project Design

The Dog Breed Classification project has the following steps:

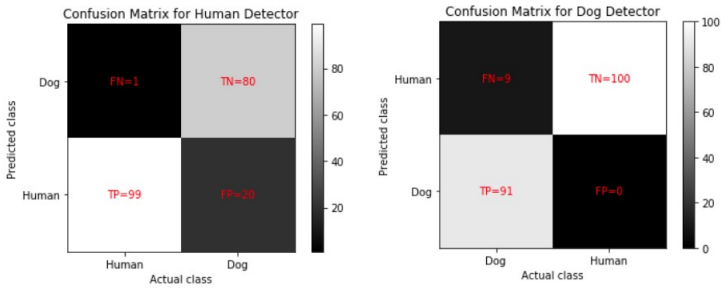
1. Importing the Human Dataset.
2. Importing the Dog Dataset.
3. Testing the human face detector.
4. Testing the dog detector.
5. Bechmark CNN (from scratch).
6. ResNet-50 (transfer learning).
7. Creating and testing the Dog App.

Part 1. Importing the Human Dataset.
Part 2. Importing the Dog Dataset.

In these 2 parts, the human dataset and the dog dataset are downloaded only if they don't exist in a local directory. Then, both datasets are loaded into memory as lists of images or dataloaders in PyTorch.

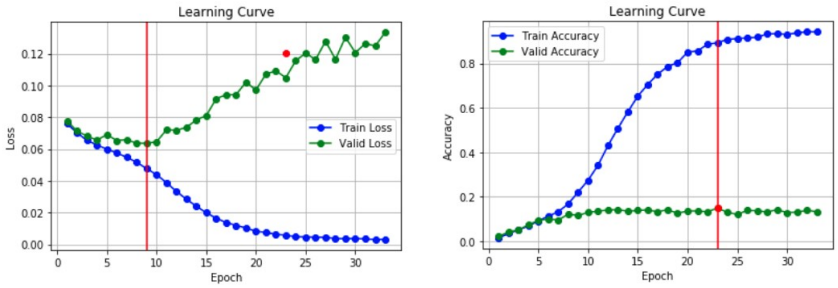
Part 3. Testing the human face detector.
Part 4. Testing the dog detector.

In these 2 parts, the human face detector and the dog detector are tested with the 100 first samples of the human dataset and the dog dataset. Confusion matrices and accuracies are analyzed.



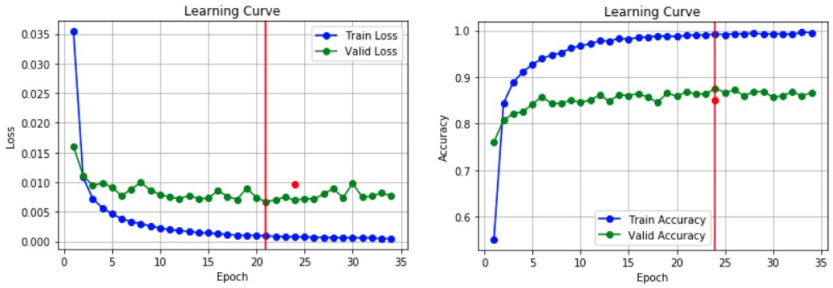
Part 5. Bechmark CNN (from scratch).

The Benchmark CNN to classify dog breeds is created, trained from scratch, and tested. The learning curves of loss and accuracy are analyzed.



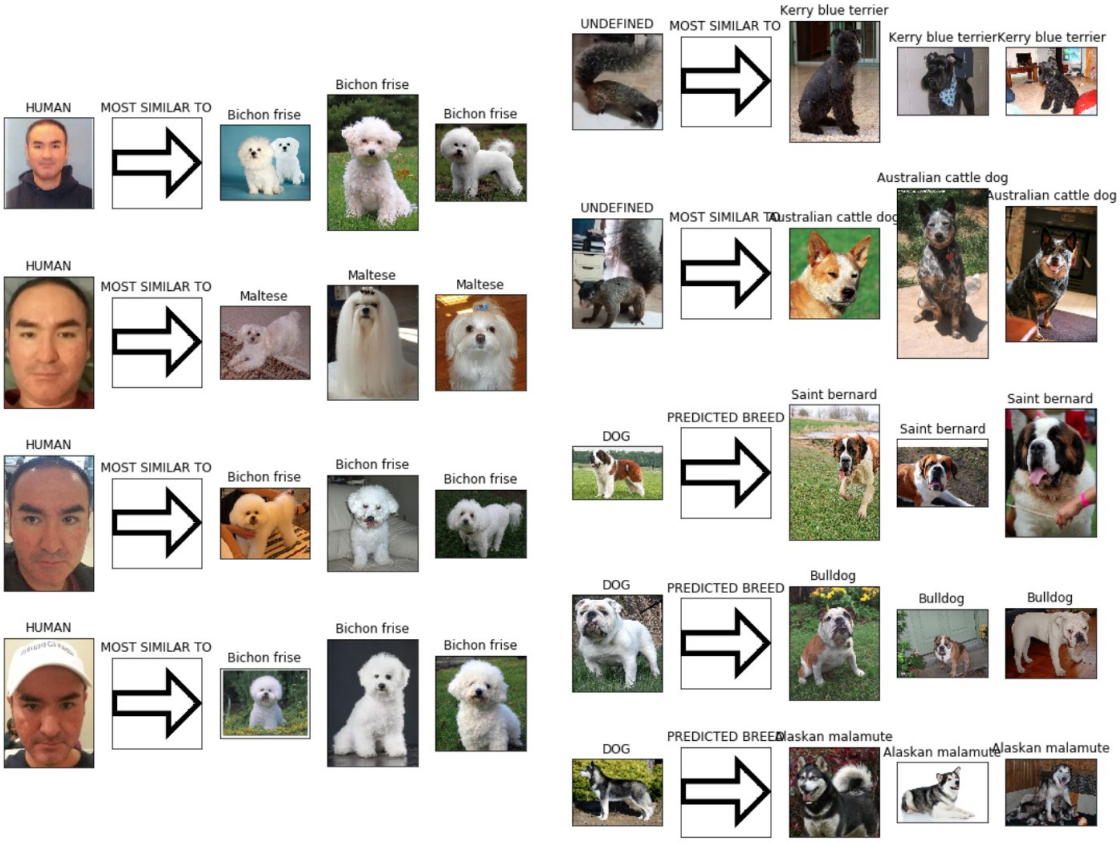
Part 6. ResNet-50 (transfer learning).

The ResNet-50 to classify dog breeds is created, trained, and tested. Training is done by using pretrained features and transfer learning. The learning curves of loss and accuracy are analyzed.



Part 7. Creating and testing the Dog App.

The human face detector, the dog detector, and the ResNet-50 are combined to create the Dog App. Each photo is classified into 3 categories: Human, dog, and other. Then, each photo is classified into one of 133 dog breeds. The results are amazing!



Bibliography

LeCun, Y. and Bengio, Y. (1995). Convolutional networks for images, speech, and time series. In Michael A. Arbib (ed.), Handbook of Brain Theory and Neural Networks. MIT Press. pp. 3361

Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning (Adaptive Computation and Machine Learning series). MIT Press.

Mitchell, T. (1997). Machine Learning (McGraw-Hill International Editions Computer Science Series). McGraw-Hill.

Russell, S. and Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th Edition). Pearson.

Viola, P. and Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features.