Dog Breed Identification - Milestone

Kaitlyn Mulligan

Marist College - DATA 440L

April 10, 2019

Abstract

1 Introduction

(Describes the motivation of this work and outlines the rest of the paper.)

In the field of Machine Learning, we have the ability to take data, learn the data through different methods, and then can predict on a test data set based on what was learned. This process can be done on datasets of images such as dog breeds. In regards to images of dogs, we can pass in a training dataset, learn them using different methods, and then we can test how well the program knows the images by passing in a testing dataset and see how accurately the program predicts the dog breed in the new image. There are different methods that can be implemented to achieve this goal, but the focus of this paper will be implementing a Convolutional Neural Network. As stated in [1], Convolutional Neural Networks are known "for their ability to recognize patterns present in images."

While exploring Convolutional Neural Networks we will see how the method works, analyze other implementations of this method, and finally implement it as a classifier in order to indentify different dog breeds. We will then analyze the results to see how well the program is classifying the dog breeds.

2 Background and/or Related Work

(Describes what other researchers in the same area have done, and how they perhaps could be improved.)

3 Methodology

(Describes what is the approach taken in this paper.)

The method used to attempt to predict the breed of dog in the images is a Convolutional Neural Network. The program that was implemented was built off of a program learned in the class as well as some additions from outside research.

Notes on the program:

The first step is to download all of the necessary packages. Next I downloaded the file with the labels of the breeds dogs and changed the strings to integers as well as identifying how many unique breeds of dogs there are in this dataset. With this, we can now begin part one which is downloading the dataset. The code for part one is as follows.

```
from skimage.transform import resize

# Part 1

# if os.path.isfile('content/train.tar.gz'):
if os.path.isfile('train.tar.gz'):
  !ls la*
else:
  !gdown https://drive.google.com/open?id=1kSc5EtnpgZAUJIhV-fEh
    XwVt-48HjfHj
  !tar -xvf train.tar.gz

# get pic count from dataframe
numPics = len(df.index)
#numPics = 8000
#print(numPics)

X = np.empty((numPics,64,64,3))
cnt = 0
labels = []

# iterate over dataframe
for idx, row in df.iterrows():
  #print('train/'+row['id']+'.jpg')
  img = imread('train/'+row['id']+'.jpg')
  #print(img.shape)
  X[cnt] = resize(img,(64,64,3))
  cnt += 1
  print(cnt)
  #print('label is: ', row['breed'])
  labels.append(row['breed'])
#    if(cnt >= 8000):
#       break
```

Next we need to verify the dataset is ok and prepare the data for the Convolutional Neural Network which we can do with the following two pieces of code. First this is how we can verify the data is ok.

```
print(X.shape)

x = np.reshape(X[4,:,:,1],(64,64))
print(x.shape)
plt.imshow(x)

#print(labels)
```

Next is how we can prepare the data for the Convolutional Neural Network.

```
x = X/255.0
print(np.max(x))

lbls = np.zeros((X.shape[0],120)) # we have 120 dog breeds

for i in range(X.shape[0]):
  #print(i)
  lbls[i][labels[i]-1] = 1

```

```
# split dataset into training and validation (test set)
from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=10)
skf.get_n_splits(x, labels)

for train_index, test_index in skf.split(X, labels):
  X_train, X_test =    x[train_index],    x[test_index]
  y_train, y_test = lbls[train_index], lbls[test_index]

print("X_train:")
print(X_train.shape)
print("X_test")
print(X_test.shape)
print("y_train:")
print(y_train.shape)
print("y_test")
print(y_test.shape)

#print(labels[0])
#print(lbls[0])

# X = x
# labels = lbls
```

Once all of this is complete, we can move on to the Convolutional Neural Network. Part two is to build the Convolutional Neural Network and Part three is to fit the CNN to the images. Both of these parts will be included in the chunk of code as follows.

```
# PART 2

# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import Dropout

# Initialising the CNN
classifier = Sequential()

# Step 1 - Convolution
classifier.add(Conv2D(32, (5, 5), input_shape = (64, 64, 3),
                      activation = 'relu'))

# Step 2 - Pooling
```

```
classifier.add(MaxPooling2D(pool_size = (4, 4)))

# Adding a second convolutional layer
classifier.add(Conv2D(32, (3, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (4, 4)))

# Step 3 - Flattening
classifier.add(Flatten())

# Step 4 - Full connection
classifier.add(Dense(units = 512, activation = 'relu'))
classifier.add(Dropout(0.2))
classifier.add(Dense(units = 256, activation = 'relu'))
classifier.add(Dropout(0.2))
classifier.add(Dense(units = 120, activation = 'sigmoid'))

# Compiling the CNN
classifier.compile(optimizer = 'adam',
                   loss = 'binary_crossentropy',
                   metrics = ['accuracy'])

# PART 3

classifier.fit(X_train, y_train,
               batch_size = 128,
               epochs = 500,
               validation_data = (X_test,y_test))
```

Now that we have downloaded the data, built the Convolutional Neural Network, and fit the CNN to the the images, we can begin predicting the breeds of the dogs in the testing dataset. To predict the breeds, we can implement the following code.

```
sample = 481
prediction = classifier.predict(np.reshape(X_test[sample], (1,64,
    64,3)))
print(prediction)
print(np.argmax(prediction)+1)
print(y_test[sample])
print(np.argmax(y_test[sample])+1)



x = np.reshape(X_test[sample,:,:,1],(64,64))
io.imshow(x)
```

(This code is still a rough version and needs some editing)

4 Experiments

(Describes the experiments performed, including details on the data used.)

The data for this experiment comes from a Kaggle competition in which the task was to determine the breed of a dog based on an image. The data can be accessed here: Dog Breed Identification. The data set has both a training and a testing set of images of different dogs. All of the images have a filename that corresponds to a unique id. A file is included which contains the unique id of the image along with the breed of the dog. The dataset is comprised of 120 different breeds of dogs.

While implementing the program I have written, it took a lot of work to figure out how many samples I could use, and what size I needed to resize the images to in order to not crash the program. I initially resized the images to 256 by 256 and used the entire dataset, but this quickly crashed the program. After a lot of fiddling with the numbers, I was able to use the entire dataset with resizing the images to 64 by 64. This all occurred before getting to the Convolutional Neural Network. The program was crashing while just in the downloading and resizing phase of the images. Once this worked I was able to start preparing the data for the Convolutional Neural Network. This included normalizing it to [0, 1] and converting the labels to one hot encoding as well as splitting the data into training and validation sets. Once this is done, I was able to build the Convolutional Neural Network and fit the CNN to the images. In order to avoid crashing the program again I used a batch size of 128. When attempting to predict the breeds of dogs, I found the the CNN was overfitting the data. Therefore, I increased the number of epochs. I started at 25 epochs, increased to 50, and further increased to 500 epochs. I found that with 50 epochs the program was still overfitting the data. When using 500 epochs, I did not see a problem of overfitting. While the program was predicting different breeds of dogs, it still does not seem to be predicting very well. **(Need to edit this paragraph and split up)**

5 Discussion and/or Analysis

(Examines the results of the experiments and draws some conclusions about their significance.)
(Discuss problems ran into and analyze accuracy of predictions)

6 Conclusion

(Summarizes the paper and its findings.)

References

[1] A. Escontrela, “Convolutional Neural Networks from the ground up,” *Towards Data Science*, 16-Jun-2018. [Online]. Available: https://towardsdatascience.com/convolutional-neural-networks-from-the-ground-up-c67bb41454e1. [Accessed: 24-Mar-2019].

[2] "Dog Breed Identification," *Kaggle*. [Online]. Available: https://www.kaggle.com/c/dog-breed-identification. [Accessed: 13-Feb-2019].

[3] H. Bendemra, "Build Your First Deep Learning Classifier using TensorFlow: Dog Breed Example," *Towards Data Science*, 26-Apr-2018. [Online]. Available: https://towardsdatascience.com/build-your-first-deep-learning-classifier-using-tensorflow-dog-breed-example-964ed0689430. [Accessed: 24-Mar-2019].