Dog Breed Classification using Machine Learning

Rodney Jones (PVAM), Nghiep Chinh Luu (HCC), and Stephanie Suniga (TSU)

Mentors: Drs. Nalinda Kulathunga, Daniel Vrinceanu and Yunjiao Wang

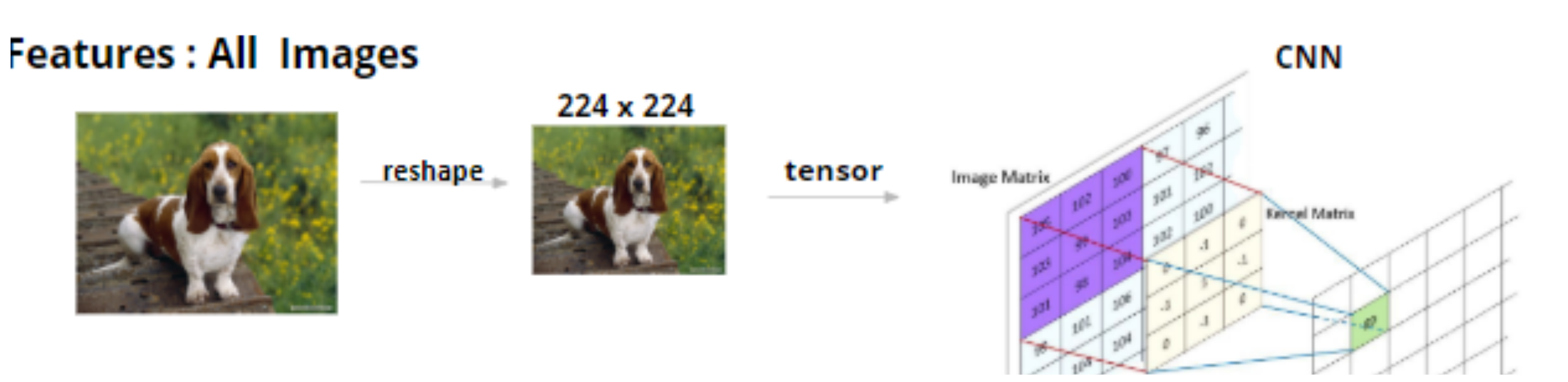
Summer Program in data Science – 2019, Department of Mathematics, Texas Southern University, Houston, TX 77004.

1. Abstract

In data science, machine learning tools have been successfully used in many disciplines, especially in image recognition and classification. In this work, we show the results from an image classification model trained using Machine Learning. For this particular example, we used a dataset with images of dogs from 133 different breeds and a model was built to identify the breed of a given dog image. During the implementations, we researched about how to use unprocessed image datasets to train a model and classify dog breeds. Model construction was done with the help of transfer learning using the ResNet50[4] pre-trained model. We applied basic image pre-processing techniques to clean the data and Convolutional Neural Networks (CNN) to extract features. Finally a fully connected neural network was used for the classification. Results show that we were able to obtain a validation accuracy of 83%. Possible over-fitting of the model was monitored using the learning curves. Accuracy score was used to present the confusion matrix.

2. Method (Steps in brief)

Pre-Processing and training: First, each image was re-shaped into a 224 X 224 X 3 tensor. Then, the tensor was then processed using a CNN with the help of TensorFlow in Keras [5] to extract the low dimensional features. Finally the 3D tensor was flattened into a 1D array and classified using a fully connected neural network.



Model Selection: For the best results we used ResNet50[4] to obtain the bottleneck features. ResNet50 is a CNN that is trained on more than a million images from the ImageNet[6] database. The network is 50 layers deep and can classify images into 1000 object categories.

3. Data Set

Our data set[7] consists of 133 classes of dog breeds. The total number of images: 8,351 RGB images.

Training- the dataset we used to train the model. The model learns from this data and weights and biases were get updated accordingly.

Validation - The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters.

Testing - The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

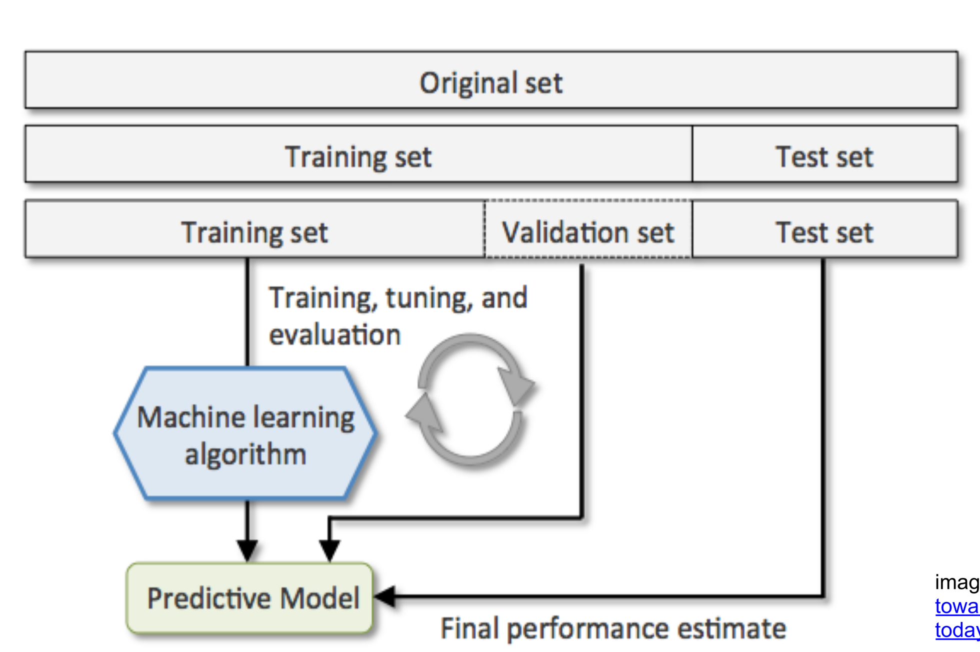
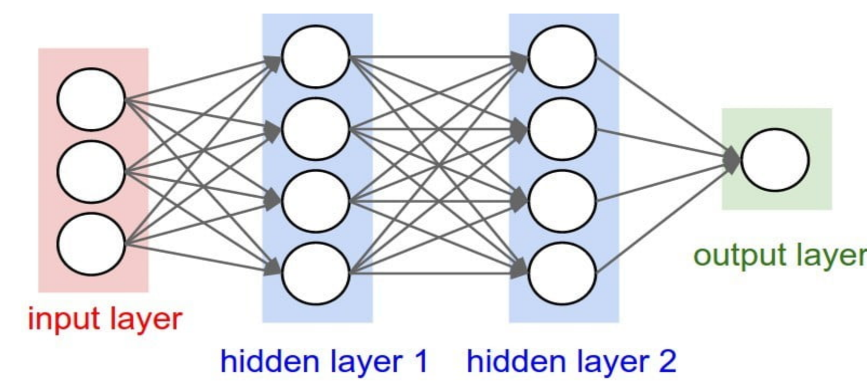


image ref: https://towardsdatascience.com/whos-quitting-today-e1b0ca2fa90f

3. Neural Networks

Neural networks are a collection of functions and operations that are modeled after the human brain, in order to recognize patterns. Neural networks begin with a neuron which is the basic unit of the brain. Neural networks work by clustering and classifying data by using features and their labels. They help to group similar data by using different types of layers, activation functions, loss functions and optimizers.



https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/

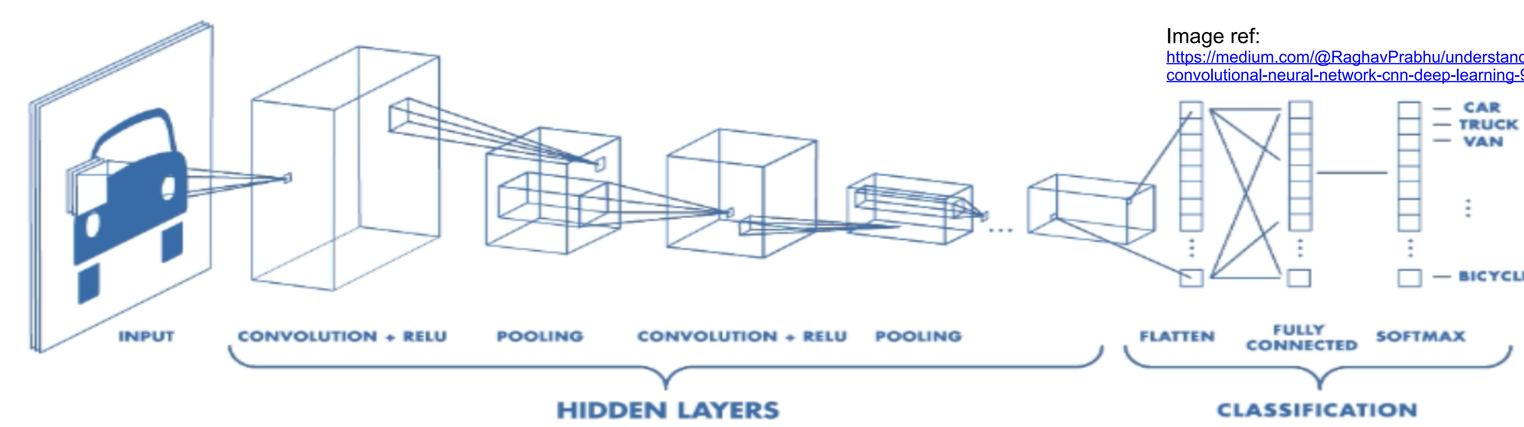
Components of Neural Networks:

- **Layers** - input, hidden, output
- **Loss Functions** - Mean Squared Error, Sum of Least Squares, Cross Entropy
- **Feedforward** - information travels forward through input nodes, then through hidden layers, then finally through output nodes.
- **Backpropagation** - is an process in supervised learning where the weights in an artificial neural networks get updated in order to minimize the loss using an optimizer.

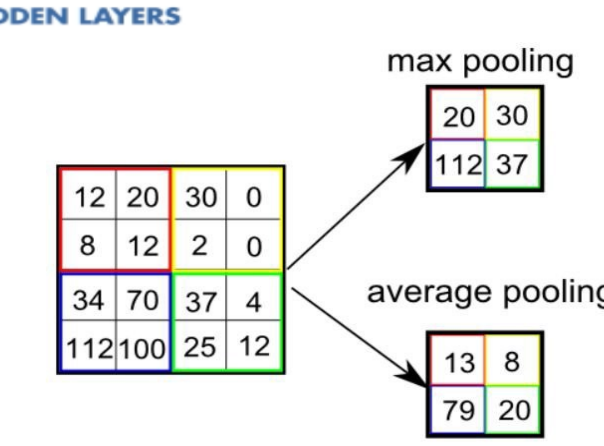
4. Convolutional Neural Networks.

Convolutional Neural Network: is a Neural Network used in image recognition and it is specifically designed to process pixel data. The word convolutional refers to the filtering process that helps to extract the important features of an image. The image is converted into numbers which depends on the strength of the pixel color. In each stride it calculates the dot product between the filer and image. ReLU activation function changes the negative numbers to 0 and positive numbers stay the same.

Image ref: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148



Pooling: reduces the sample size of the data set. Eg: Maximum pooling , Average Pooling and Channel Pooling.

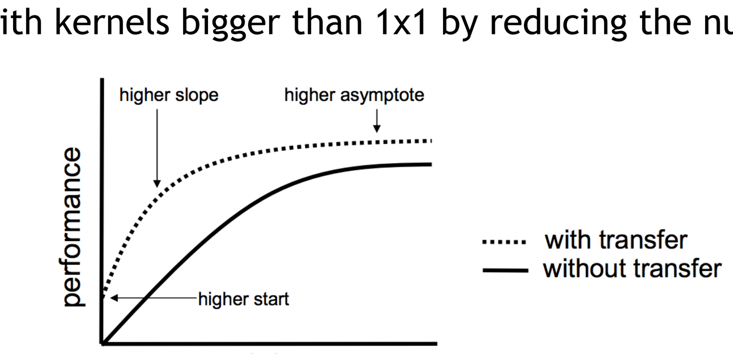


Dense: in the fully connected network, first the multidimensional arrays are re-shaped in to a one dimensional array. This process is called flattening. After flattening, this 1D array will be the input to a fully connected neural network which we called as dense layers. Each node in the output layer gives the probability of a given image being in a certain class.

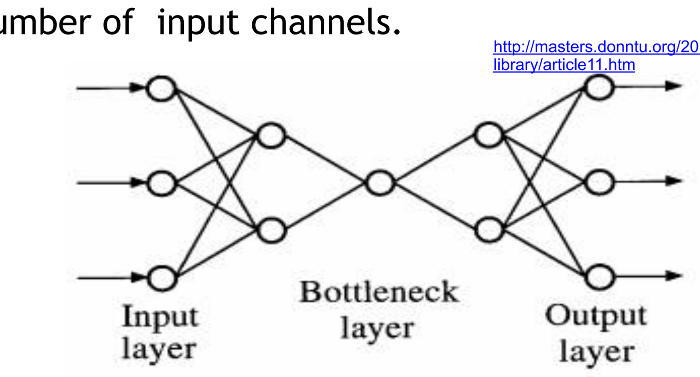
4. Transfer Learning and Bottleneck Features.

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. It is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different problem.

Bottleneck features are viewed as a layer that reduces the size of the input tensor in a convolutional layer with kernels bigger than 1x1 by reducing the number of input channels.



http://orzo.union.edu/Archives/SeniorProjects/2018/CS.2018/files/kashyapa/kashyapa-499-report.pdf



http://masters.donntu.org/2010/fknt/sivyakov/library/article11.htm

4. Results - II (Learning Curves/Confusion Matrix)

Model summary

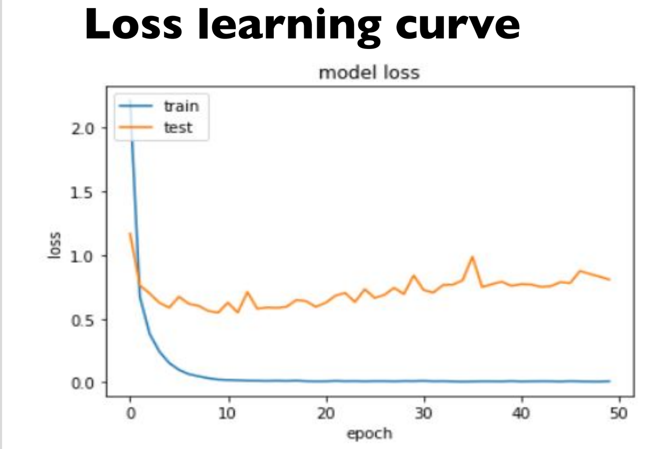
With 2 fully-connected layers being connected to the layer from pretrained-model ResNet50 - bottleneck features.

```
Layer (type)                 Output Shape              Param #
=================================================================
global_average_pooling2d_2 ( (None, 2048)              0
_________________________________________________________________
dense_2 (Dense)              (None, 800)               1639200
_________________________________________________________________
dense_3 (Dense)              (None, 133)               106533
=================================================================
Total params: 1,745,733
Trainable params: 1,745,733
Non-trainable params: 0
_________________________________________________________________
```

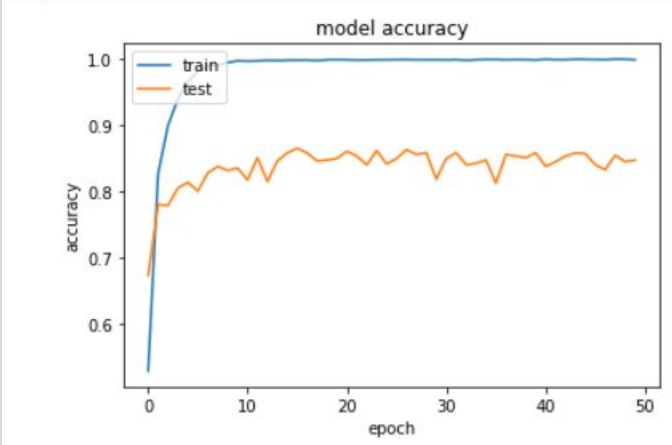
```
history = ResNet50_model_2.fit(train_ResNet50, train_targets,
          validation_data=(valid_ResNet50, valid_targets),
          epochs=50, batch_size=64, callbacks=[checkpointer_2], verbose=1)
```

```
Test accuracy: 83.0144%
```

Loss learning curve

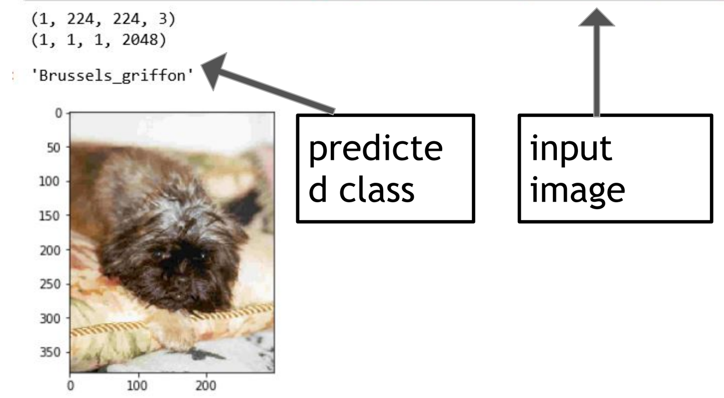


Accuracy learning curve



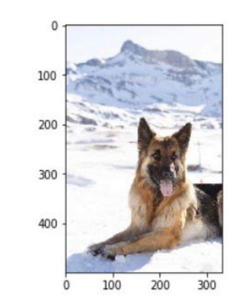
Incorrectly predicted sample

```
ResNet50_predict_breed("Data/dogImages/test/001.Affenpinscher/Affenpinscher_00036.jpg")
(1, 224, 224, 3)
(1, 1, 1, 2048)
'Brussels_griffon'
```

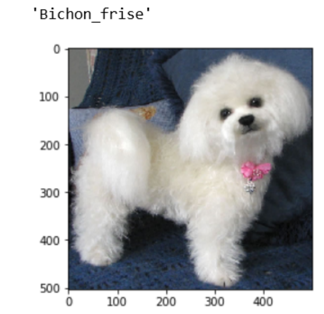


Correctly predicted samples

```
ResNet50_predict_breed("Data/dogImages/test/071.German_shepherd_dog/German_shepherd_dog_04886.jpg")
(1, 224, 224, 3)
(1, 1, 1, 2048)
'German_shepherd_dog'
```



```
ResNet50_predict_breed("Data/dogImages/test/024.Bichon_frise/Bichon_frise_01707.jpg")
(1, 224, 224, 3)
(1, 1, 1, 2048)
'Bichon_frise'
```



6. Discussion

We demonstrate an easier way to classify dog breeds by using computer vision rather than the human eye. This is convenient to use because there are hundreds of different dog breeds that we may not know immediately. We used total of 8,351 images with 133 classes to classify dog breeds using the concept of deep learning. We extracted the bottleneck features from different pre-trained models and the Resnet50 showed better performance in this project. We used a fully connected layer on top of the bottleneck features and the best model was obtained when 2 hidden layers were used in the classification process. We found that more hidden layers may cause over-fitting. The optimizer used was rms-prop[7] and we used the *sigmoid* in the hidden layers and the *softmax* in the output layer as our activation functions to obtain the best model with accuracy of 83.01%.

Reference :

1. Neural Networks (2019), https://skymind.ai/wiki/neural-network
2. Convolutional Neural Networks http://cs231n.github.io/convolutional-networks/
3. Transfer Learning (2019), https://machinelearningmastery.com/transfer-learning-for-deep-learning/
4. ResNet50 (2019), https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33
5. TensorFlow http://ruder.io/transfer-learning/
6. ImageNet https://towardsdatascience.com/how-to-scrape-the-imagenet-f309e02de1f4
7. Rmsprop http://ruder.io/optimizing-gradient-descent/index.html#rmsprop

Acknowledgment :

This work has been supported by National Science Foundation through an Excellence in Research award (CNS-1831980).

Contact email : rjones102@student.pvamu.edu