Machine Learning Engineer NanoDegree

Capstone Project

Dogs Vs Cats

Neeraja Akula

05/24/2018

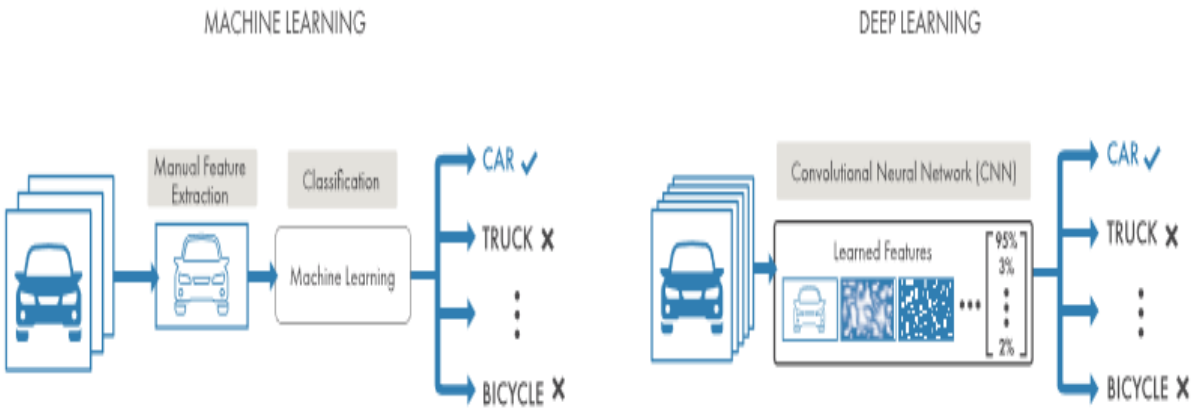
Project Overview

I have selected to work on old kaggle competition called “Dogs Vs Cats”. Which is big challenge to distinguish the images are either of dog or cat.

https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition

The challenge is here to automatically distinguish images of dogs from images of cats. To distinguish dog and cat images is easy task for the humans. But for computers very difficult because of complicated features such as variety of breeds, shapes colors, photos composition so on in the images. Before it used to CAPTCHA challenge (Completely Automated Public Turning Test to tell Computers and Humans Apart) to distinguish human user of a system from computers. But the CAPTCHA accuracy only 82.7% now the modern computer vision technology using Convolution Neural Network should able to improve the accuracy. CNN works by extracting features directly from images, while the network trains on collections of images. This automated feature extraction makes deep learning models highly accurate for computer mission tasks such as object classification. The project consists of deploying a deep learning model trying to distinguish the given images of dogs and cats.

A key advantage of deep learning networks is that they often continue to improve as the size of your data increase. You can see and observe the below picture of comparing a machine learning approach to categorizing vehicles with deep learning.



Problem Statement

The problem here is to us Convolutional Neural Networks (CNNs) and deep learning techniques to train a computer to distinguish between images of dogs and cats. We have a train dataset it contains 25,000 images of dogs and cats with labels to achieve this. We will use this to train and validate our model. We have a test dataset, it contain 12,500 mixed images. It has numeric ID. We need to provide label (cat or dog) to these images. The problem is a classic computer vision challenge. The data set images are not in same size. The images should be resized. My approach would be to train a deep convolution neural network (CNN) implemented using TensorFlow to tell the computer to analyze the size of image. My object is to start with simple implementation of CNN and adding several layers of convolution, max pooling and slowly improve on this simple model by adding dropout.

Metrics

The metric proposed to evaluate the classifier performance is simple accuracy and log loss.

Log loss main goal of the convolution optimizer will be to minimum the cross entropy loss which is the same as log loss. Accuracy score is defined as correctly predicted class labels. This accuracy score will be determining on the training dataset and the validation set. This is secondary metric that will be determined and shared but the selected optimal model will be based on minimizing log loss. The model with the lowest log loss will have one of the good accuracy score.

Analysis

Data Exploration and Exploratory Visualization

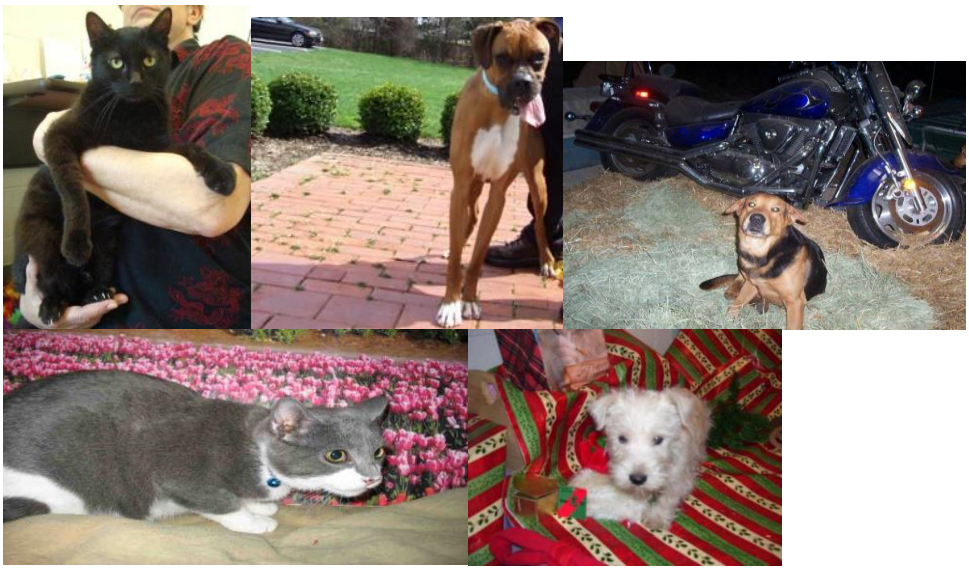
The data for this Dogs_Vs_Cats project can be found on kaggle website. The train zip file contains 25,000 images for training and the images have cat or dog label. The test zip file contains 12,500 images for testing with no label.

https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data

Each image in train.zip file has the format “cat.250.jpg” which includes a label for cat or dog and an image sequence number.

Some images shared below:

Dogs and Cats



The majority of the images are low resolution, poor quality and different height and width these characteristics of the images makes this classification exercise challenging in identifyig them.

We can observe here, these images have different width and height. So this is the first challenge to solve. Images to be reduced to a uniform width and height. These images are big, for process efficient, images should be small. So each pixel will be an input into a neural network so big images should require major computing power. Example each image 50x50 gray image would be converted to an array of 50x50x1 pixels = 2500 pixels. If we go with 128x128 colored image that

would be converted to an array of 128x128x3 pixels = 49152 this 25k images will be come a massive dataset. So I would like to resize all 25000 images to 50x50 pixels and reduced to gray scale.

Algorithms and Techniques

I feel a deep convolutional network is the bes algorithm to choose for this problem. The deep neural networks very effective at classifying images is their ability to automatically learn multiple levels of abstaction that simply characterize each class in a given classification task. Then can recognize pattersn with extreme variability, and with robustness to distortions and geometric transformations.

Cnvolutional neural networks are biologically inspired variants of multilayer perceptorns, designed to copy the behaviour of a visual cortex. These models mitigate the challenges posed by the multilayer perceptorns architecture by exploiting the strorng spatially local correlation present in natural images. All neurons in a given convolutional layer share the same weights and detect exactly the samething. Replicating units in this way allows for features to be detected regardless of their position in the visual field, thus constituting the property of translation invariance. Also sharing weights between neurons makes it computationally efficient.

Here is my approch: Started with simple convoution neural network architecture with a convolutional layer with relu activation function followed by one max pool layer Adam used to optimize this model, got accuracy 49% after simple convolution neural nework is set up, then I decided we need a bigger convolutional neural network architecture to improve the accuracy.

The following will be done to futher tune simple model.

1. Adding more convolution and fully connected layers.
2. Tuning the No. of neurons in the all different layers.
3. Tuning the learning rate in the optimization function
4. Increasing No. of training epochs.
5. Using grayscale images instead of color images.

Benchmark

A basic benchmark for this project would be wether it can give better results than could be obtained by chance. I have run simple CNN model architecture with a convolutional layer and one max pooling layer with 4 epochs, I got 49% accuracy. I will take this is as my benchmark. I will try to beat this benchmark using this model.

Methodology

Data Preprocessing

1. Resize all the images to same height and width and reduce the size. All the images were resized to 50x50 cv2 was used to do the resizing images and read as grayscale images. Sample code below.

   img = cv2.resize(img, IMG_SIZE,IMG_SIZE)

2. Create a dataset for labels. The kaggle dataset has a jpeg labelled as either cat or dog.

   Def label_img(img):
   Word label = img.split(‘.”) [-3]

   Conversion to one-hot array
   If word_label == ‘cat’: return [1,0]
   elif word_label == ‘dog’: return [0,1]

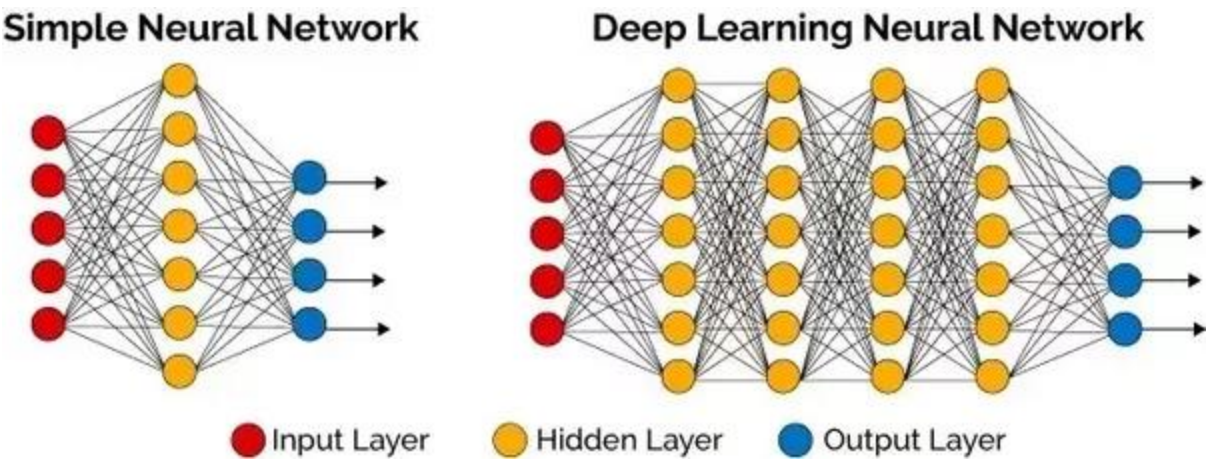
3. Shuffle the cat and dog images with in the dataset since the images were read as first all cat images and then all dog images. Shuffling of the training data to preserve the random state of our data.

4. Reshap the dataset and labels into the proper format before feeding into Tensorflow.

No further preprocessing was done for images since convolution neural networks work best when in images are kept close to natural state and the model is allowed in extract all rlevant features.

Implementation

I started implementing a basic simple convolution neural network in tflearn.



Resources: http://tflearn.org/getting_started/

In this simple model used convolution layer with relu activation function followed by one max pool layer run this with 4 epochs, got accuracy 49%. During this process I had a problem with model process time it took longer time then I did resolve by reducing image size to 50x50 with grayscale images. I know the output image will not be very clear since all the images are reduce to 50x50 for machine to process fast though the tradeoff between speed and loss.

Refinement:

Spending lot of time in this project on this stage decided to increase parameter to tune the simple model. The following adjustments were tired to further refine the model.

1. Change in layer structure of neural network. Adding more convolution and fully connected layers and max pool layers.
2. Tuning of neural network parameters.
3. Selecting the best optimization function – Adam optimizer.

4. Tuning the learning rate in the optimization.
5. Increasing No. of training epochs.

I went through lot of resources and research papers to get knowledge and apply the techniques to this project. The above techniques influenced the model output. However trying different combinations showed larger batch sizes resulting in faster running times and also causes a small increase in accuracy.

Results

Model Evaluation and Validation

I tried all the techniques the final output seems reasonable in terms of the parameters and layers structure. I got accuracy 89.81% this is good for this project compared with the basic model accuracy 49% I reached this accuracy with 15 epochs using 24500 training samples and 500 validation samples. I have tested unlabeled unseen images and the result was quite good 89% Those images are from test dataset of the Kaggle competition. The result is trusted but can be improved.
Robustness: Input images in both training and validation sets have little to no consistency in terms of images quality and position so on. Nevertheless, it is able to consistently classify them with high accuracy.

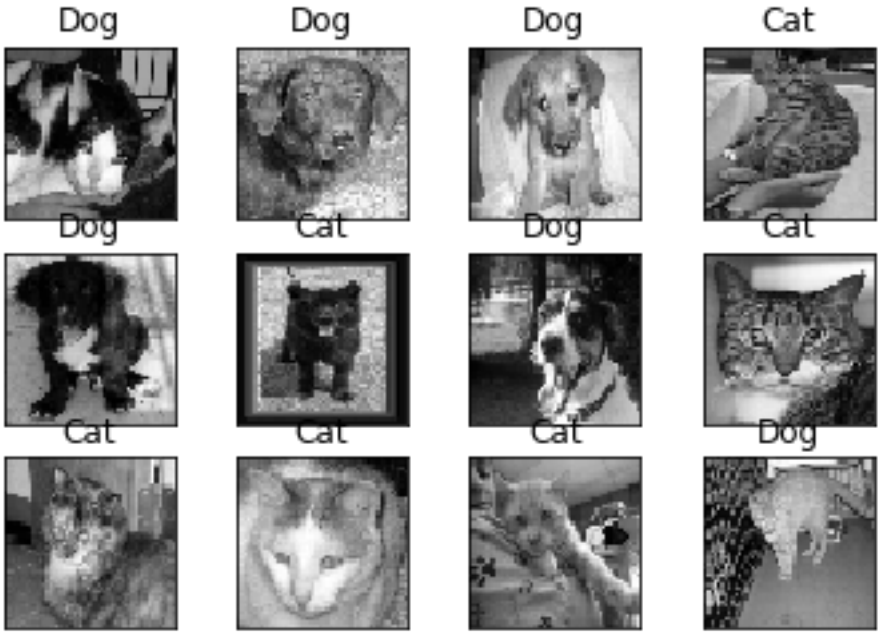
Justification

The final model for this problem is a deep convolution neural network that using convolution layers, Max pool layers, fully connected layers on grayscale images to distinguish Dogs vs Cats. I have tried different combination of network parameters, optimization function to tune the model. I have seen very small amount of benefit increasing of continuing this optimization further and I felt comfortable stopped at this point. The benchmark for this project were to beat 49% accuracy, now the final model got 89.81% accuracy the result beat the benchmark. This accuracy tells the effectiveness of convolutional neural networks for image classification tasks.

Conclusion

Free-Form Visualization

With accuracy 89.81% it’s very interesting to examine the images that were incorrectly classified to see that made them different.



I know the images are not clear but I had to choose the image size 50x50 to avoid longer data process time. You can see above couple of images were predicated as dog instead of cat. This mistake understandable because the images could be unclear, images of dog breeds that looks like cats and images of cats in poses more associated with dogs. If there were more dog breeds in the training data, it could help to identify images as cats and improve visualizing and accuracy.

Reflection

I enjoyed working on this project. While working, I gained more knowledge of deep learning, convolution neural networks and TensorFlow through courses and books. During preprocessing dataset learned about image processing like resize and grayscale. Writing the code for the model took time as I learnt how to code with TensorFlow and TFLearn. The simple convolutional neural network structure was the basic structure to reach the success for this project.

Optimization and Refinement was the best phase for me. This phase make me to learn about more convolution neural network. I have gone through too many resources. The final part is documenting all my experiences and finding in this report.

The whole project was a very interesting exercise for me. Now I have a confidence to solve this kind of image processing and prediction problems.

Improvement

A bigger network figures things out better, and quicker, but tends to also over fit the training data. The chosen model got a high accuracy score it's noticeable that the training set accuracy is a few percent higher than the validation set accuracy. This tells that the model is likely over fitting to the training data. We can rectify slightly this increasing dropout, but there does seem to be a limit or use of other modern techniques that can help reduce overfitting. Using external data (other than 25K images) which data is available publically can be used for this project to train the model. Better input preprocessing also help the model such as if the images has two or three animals, remove all keep one. If there are humans with their animals in the image, remove human keep animal, mainly keep the animal face that has most valuable feature in distinguishing between animals.

Referances

1. https://www.kdnuggets.com/2017/11/understanding-deep-convolutional-neural-networks-tensorflow-keras.html
2. http://deeplearning.net/tutorial/
3. https://www.r-bloggers.com/making-sense-of-logarithmic-loss/
4. https://www.mathworks.com/discovery/deep-learning.html
5. http://ml-cheatsheet.readthedocs.io/en/latest/optimizers.html#adam
6. https://www.analyticsvidhya.com/blog/2016/04/deep-learning-computer-vision-introduction-convolution-neural-networks/
7. http://cv-tricks.com/tensorflow-tutorial/training-convolutional-neural-network-for-image-classification/
8. http://neuralnetworksanddeeplearning.com/chap3.html
9. https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/