Dogs vs. Cats Dataset

Ranaa Ansari, Daniel Arenson, Yining Feng, Han Nguyen

Northwestern University

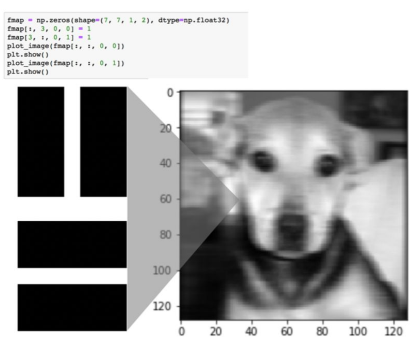
MSDS-422 Assignment 7: Image Processing with a CNN

Summary and problem definition for management

We are providing advice to a website provider who is looking for tools to automatically label images provided by end users. As we look across the factors in the study to ultimately make a recommendation about image classification, we are most focused on a model that achieves the highest possible accuracy in image classification. Our models will take into consideration the tradeoff between time and accuracy. Machine learning algorithms will be used to classify images. The image input which is provided from a Cats and Dogs database, will be analyzed and the predicted result will be given as output.

Research Design

The data set for this assignment is 2,000 color pictures of dogs and cats and is more complex than the linear images of handwritten digits. To prepare the data for analysis, we need to turn each picture into a Numpy array which we can easily concatenate, reshape and manipulate. The complexity of each array will be a factor of image size -128 pixels square vs. 64- and the number of layers or channels – color has 3 layers vs. 1 for grayscale images. These decisions will have an impact on processing time, so the objective is to minimize complexity while maximizing differentiation for classification.



An example of how a 'convolutional' filter is applied to each 4x4 pixel group – enhancing vertical & horizontal changes

Once the data is ready, we take the following steps:

(1) Scale the data and split it into training & test data sets

(2) Test two models using assignment 6 'DNN' techniques

(3) Apply 'CNN 'method for filtering and processing images

(4) Compare the model timing and accuracy

In the reading for this assignment, the research points to the fact that in 'real' neural networks there is a filtering process prior to the data being input to the brain. To simulate this natural process, we will apply a technique to filter the image and enhance specific vertical or horizontal attributes. This will help the neural network soften the inputs prior to applying a DNN similar to the one used on the MNIST data set. The technique is referred to as a 'Convolutional Neural Network' (CNN). Although this step adds complexity to the model, we will see that the CNN model performs much better than simply using a DNN.

Programming Work

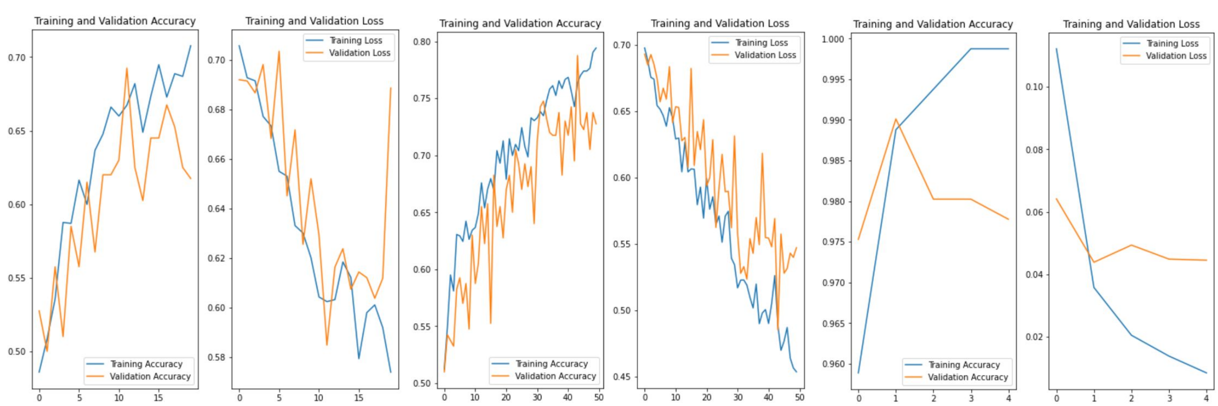
The programming dataset of cats and dogs images was extracted from this week's module on Canvas. The file contains 1000 images of cats and 1000 images of dogs. The entire dataset was used and split into a train and validation regime and utilized 3 Convolutional Neural

Network models within Keras and Tensorflow to create a binary classifier to identify which images are actually cats and which images are actually dogs.

After the data was split into train and validation, it was passed through into Keras *ImageDataGenerator*, which is used to perform augmentation and preprocessing of each train and validation image, while configuring and normalizing the data. Once this was complete, each model had various combinations of Epochs, Convolutional Layers (*keras.layers.Conv2D*), Pooling Layers(*keras.layers.MaxPooling2D*), Dense Layers(*keras.layers.Dense*), Dropout (*keras.layers.Dropout*) and optimizers fine tuned accordingly to each model. Each Convolutional Neural Network model was then further evaluated with an accuracy score and validation score (applied with *.compile(), .summary() and .fit_generator()*) .

Results and Recommendation

A total of three experiments were conducted to test the performance of convolutional neural networks on this dataset, with varying optimizers, layers, and epoch cycles. The first model utilized the RMSprop optimization method which yielded a validation accuracy of 68.8%. The next two models utilized the Adam optimizer to update network weights, testing with up to fourteen convolutional layers, four pooling layers, and two dense layers. In order to evaluate the performance of each model, the training/validation accuracy and loss were calculated and plotted as shown below:



	Optimizer	ConvolutionalLayers	PoolingLayers	DenseLayers	Epochs	TrainingAccuracy	ValidationAccuracy
Model 1	RMSProp	3	4	2	20	0.708	0.618
Model 2	Adam	4	4	2	50	0.794	0.728
Model 3	Adam	14	1	1	5	0.999	0.978

As the criticality of model accuracy outweighs run time in this scenario, it is apparent that Model 3 (utilizing the Adam optimizer with fourteen convolutional layers, one pooling layer, and one dense layer) yielded the best performance of the three models with an accuracy of 100% for the training data and 98.5% for testing data. Convolutional neural networks are highly recommended as a classification method for this dataset, especially when used with the Adam optimizer.

References

Brownlee, J. (2019, April 12). How to Configure Image Data Augmentation in Keras. Retrieved May 24, 2020, from https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Pokharna, H. (2016, July 28). The best explanation of Convolutional Neural Networks on the Internet! Retrieved May 21, 2020, from https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8

Pradhan, M. (2019, December 1). Image Classification using MobileNet in the browser. Retrieved May 21, 2020, from https://medium.com/analytics-vidhya/image-classification-using-mobilenet-in-the-browser-b69f2f57abf

Turmukhambetov, D. (2018, April 25). Mobile Nets: Efficient Convolutional Neural. Networks for Mobile Vision Applications. Retrieved May 21, 2020, from http://visual.cs.ucl.ac.uk/vrg/slides/2018-04-25.pdf