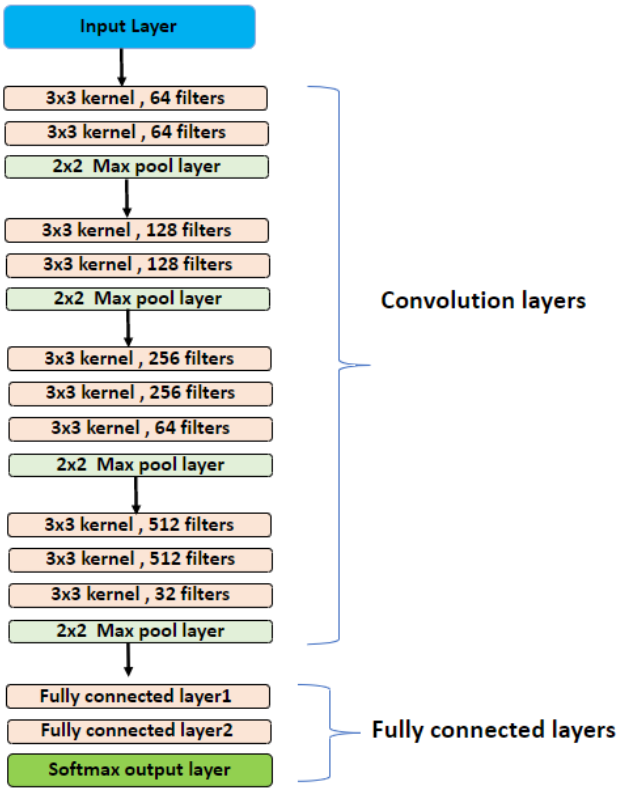
Building a CNN model using Pytorch to classify cow teats images

Lakshmikar Reddy Polamreddy Yeshiva University
lakshmikarpolamreddy@gmail.com

Abstract

I have built a CNN model using Pytorch that will classify cow teats images into four different categories. For this, I built my model with 10 convolution layers, 3 pooling layers, 2 fully connected layers and 1 output layer. When I trained this model for 300 epochs, the training loss has gradually decreased from 1.36 to 0.4 and achieved a training accuracy of 84 percent. I tested my model on 380 unseen images but I achieved 43 percent accuracy. Though I built various models with several convolution layers, i have faced the problem of over-fitting and test accuracy was way below 40 percent. To avoid the problem of over fitting, data augmentation techniques like rotation, horizontal flip, dropout were utilized but could not improve on test accuracy. As I have used less number of convolution layers and low kernel size of 3x3, this is computationally faster than the VGG model. However my model performance is far less than that of the VGG model in terms of accuracy metric. In future, I would like to improve my model performance by further parameter tuning and building the appropriate architecture.



1. Introduction

Earlier, when I tried to classify these cow teats images using a classic NN model, I could achieve an accuracy around 43 percent, which is significantly low. To improve on this, we can build a CNN model using Pytorch library. But several CNN models were built earlier by many authors for image classification. To quote for instance, the VGG16 model was built and it could achieve an accuracy of 66.8 percent. My objective is to build a CNN model that achieves better accuracy than VGG model with as many less layers as possible.

My CNN model architecture is as shown in the below picture. It has just 10 convolution layers which is less that that of VGG16 model. Unlike VGG16 model, I have used zero padding instead of same padding. But I have used same input size of 224x224x3 as higher size takes loner time for training

2. Related Work

Separable Confident Transductive Learning [1] model model for Dairy Cows Teat-End Condition Classification was developed by the members of Cornell University. An accuracy of 77.6 percent was achieved with this model.They have proposed a separation loss to enlarge the dissimilarity between different categories. They performed experiments with seventeen benchmark ImageNet models by optimizing the loss functions.

3. Methods

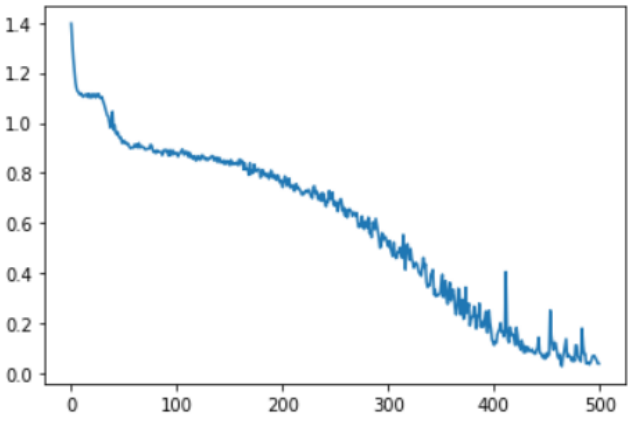
We calculate a separate loss for each class label per observation and sum the result

$$\text{Loss} = \sum_{c=1}^{M} (y_{o,c} log(p_{o,c}), \quad (1)$$

where M is number of classes, y is binary indicator (0 or 1) if class label c is the correct classification for observation o and p is the predicted probability observation o is of class c

4. Results

Below graph shows loss value for training images versus number of epochs for one of my models with 7 convolutions that was run for 500 epochs. This has been plotted with number of epochs on X-axis and loss values on Y-axis. Though the training loss for this model has reduced from 1.36 to 0.05 after 500 epochs and seems to have converged, it has just given 39 percent accuracy despite the train accuracy being 99 percent.

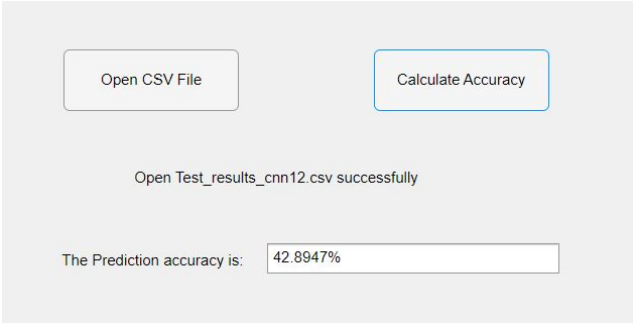


For this reason, I have built another model with 10 convolution layers. after training this model for 300 epochs, it has given better accuracy on the test images.Below are its results.

Train accuracy : 84 percent

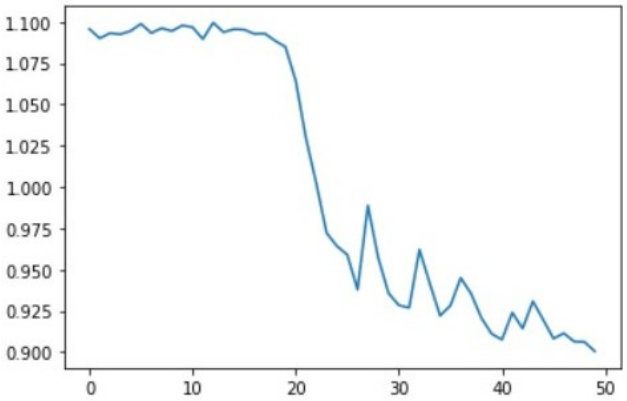
Validation accuracy : 59 percent

After testing this convolution model on 380 unseen images, it has shown an accuracy of 43 percent.



Data set	Train data set	Validation data set	Test data set
Accuracy in percentage	84	59	43

The train losses in this case have not converged even after running for 300 epochs. I trained this model for every 50 epochs. Below graph shows train losses after 100 epochs from 51st epoch to 100th epoch.



Since the losses have not converged, I trained this model for another 100 epochs, then train accuracy has increased to 98 percent but validation accuracy has reduced to 50 percent indicating over-fitting.So, I have compromised on number of epochs due to this problem.

Below table shows the comparison of test accuracy's obtained using classical NN model and CNN model. It indicates that performance of CNN model has not improved better than the classical NN model.

Model	Classical NN model	CNN model
Test accuracy in percentage	43	43

4.1. Datasets

I have considered 1149 cow teats images categorized into 4 classes for training my CNN model. 1000 images have been randomly selected for train data set and 149 images for validation data set. A batch size of 32 images have been selected for running each epoch. The trained model was applied on 380 unseen images to predict their classes.

Data set	Train data set	Validation data set	Test data set
Number of images	1000	149	380

Using data loader of Pytorch, I have transformed and normalized these data sets.

5. Discussion

Regarding number of convolution layers, my CNN model was initially built with 16 convolution layers with kernel size of 5x5. As training of this took a long time even for 10 number of epochs, I could not tune the model for reduced losses. As a result, I decided to use the less number of layers and less kernel size. Then, I began with 16 convolution layers but the losses remained at 1.0 and were not converging after 10 epochs though the model was run for 300 epochs. Finally, when I used 10 convolution layers with 3x3 filter, losses reduced and converged after 350 epochs.

Regarding activation functions, I have used Relu for all the layers except for the output layer where the softmax function has been applied to get the values of probability between 0 and 1.

Regarding kernel size, I preferred 3x3 because of the advantages it has over other sizes. For instance, a filter of 1x1 will not extract good number of features. At the same time, a filter size of 5x5 and above will take substantially longer time for training.

Regarding loss, I have used Cross-Entropy loss function as it measures the performance of a classification model whose output is a probability value between 0 and 1

Regarding optimizer, I preferred to use Stochastic gradient descent to Adam optimizer as the former has converged faster than the latter in this case. a learning rate of 0.002 and momentum of 0.9 has been used for faster convergence over losses.

Data augmentation techniques like horizontal flip, rotation. dropout layers have been applied to avoid over-fitting but could not overcome this problem. Further work is necessary in this aspect.

To make the training faster, I have normalized the images to have a mean and standard deviation of 0.5 each.

6. Conclusion

This project has been started to achieve better accuracy than classical NN model and VGG 16 model in terms of classification of cow teats images. I have tried this using multiple models with various convolution layers than ran for epochs from 100 to 500. Out of all the models that were built, one model with 10 convolution layers that was run for 300 epochs has given me 43 percent accuracy, higher than the any other model. However, there is a lot of scope for improving the performance. By performing parameter tuning, by data augmentation techniques, by adding more convolution layers etc, we may achieve better accuracy. Due to less computational power of my computer and time constraint, I have compromised with 43 accuracy for now. With this experience and knowledge, I would like to build another better model in future which will give better accuracy than the VGG16 model.

References

[1] Y. Zhang, I. R. Porter, M. Wieland, and P. S. Basran. Separable confident transductive learning for dairy cows teat-end condition classification. *Animals*, 12(7):886, 2022. 1