David Vaughan

Machine learning

Juntao Huang

4/30/2023

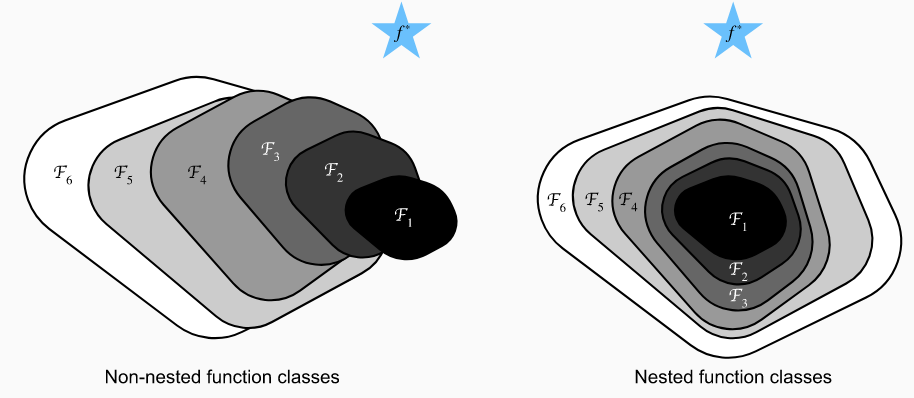
Final Paper

Project 2 Residual Network

The goal of project two was to apply a Residual Network to the image classification problem on the CIFAR-10 dataset and to compare the performance with AlexNet. “The CIFAR-10 dataset contains 60,000 32x32 color images in 10 different classes.[[3]](https://en.wikipedia.org/wiki/CIFAR-10#cite_note-HopeResheff2017-3) The 10 different classes represent airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.” (Wikipedia CIFAR-10) Using an AlexNet approximation for the midterm project I was able to achieve an accuracy of around 70%. AlexNet was a convolutional neural network (CNN) used in the 2012 ImageNet Large Scale Visual Recognition Challenge and the AlexNet network achieved a top-5 error of 15.3% (Wikipedia AlexNet). AlexNet helped start the trend of bigger and bigger neural networks. As CNNs have gotten bigger and the layer counts increase one new method used to keep improving their accuracy is Residual Networks. Residual Networks are modern CNN that make use of massive data sets, GPUs, and new techniques to help make CNN’s even more accurate for solving the image classification problem.

To solve the image classification problem a CNN is usually used. CNNs use a convolutional layer, which performs a convolution (element-wise matrix multiplication, summation of all the elements, and add bias) on the input data to extract features = {W - K + 2P}/{S} + 1, O: output size, W: input size, K: kernel size, P: same padding (non-zero), S: stride. The convolution operation involves sliding a small filter, or kernel, over the input data and computing a dot product between the filter and each patch of the input. The product then will probably go through a ReLU activation function (Wikipedia Contributors, “Convolutional Neural Network”). The kernel is changed and adjusted through backpropagation to get optimal values. A CNN can have padding: adding zeros around the matrix (for example 4x4 pad becomes 6x6). A CNN can have a Pooling layer, which reduces the dimensions of the hidden layer by combining the outputs of neuron clusters at the previous layer into a single neuron in the next layer. O = {W - K}/{S} + 1. W: input size, K: kernel size, S: stride size = kernel size.

Older versions of CNNs like LeNet used to use sigmoid as an activation function. This was not ideal because non-linear functions can take too long to compute for large neural networks. (Wikipedia “Convolutional Neural Network”) “In order to use stochastic gradient descent with backpropagation of errors to train deep neural networks, an activation function is needed that looks and acts like a linear function, but is, in fact, a nonlinear function allowing complex relationships in the data to be learned.” (Jason Brownlee) ReLU was chosen to fulfill these criteria because is simple to compute (if input > 0: return input else: return 0). Because of ReLU speed and the fact that is somewhat linear, it allows for deeper CNN to be made.

As our Neural network gets larger, and as the function class gets bigger, we want the function to be more accurate and to move closer to the correct function for the neural network. But with non-nested functions, we can slowly drift away from the correct function, and we will end up with an incorrect answer. 

Diagram, schematic

Description automatically generatedAlso, large CNNs suffer from the vanishing gradient problem. The vanishing gradient problem is encountered when neural networks use backpropagation, which tries to find the minimum of a function with derivatives and weights. As the Network gets deeper and deeper the weights get smaller and smaller until backpropagation no longer works effectively. To solve the vanishing gradient problem and the non-nested classes, we make our neural network have a nested function class. To make this we have a function F(x) = G(x) + x. G(x) is a residual block which can consist of a convolution, batch normalization, activation function then another convolution and batch normalization. Batch normalization is used to solve internal covariate shift which is “the distribution of each layer’s inputs changes during training, as the parameters of the previous layers change”. (Ioffe) Internal covariate shift slows down the neural network, so batch normalization is used throughout the network to help speed it up. Batch normalization in Pytorch is a function y= β + (γ \*(x – E[x]))/sqrt (Var[x]+ϵ​). (BatchNorm2d — PyTorch). The x input is the input layer that travels through a residual connection past the residual block to be added with the result of that residual box g(x). (*D21*). To add x correctly no matter the size the Residual class uses “… two types of networks: one where we add the input to the output before applying the ReLU nonlinearity whenever use\_1x1conv=False, and one where we adjust channels and resolution by means of a convolution before adding.” (*D2l*). After F(x) is found it goes through the activation function RELU and then goes on to the next block. Because x is added to F(x) it helps to solve the vanishing gradient problem. With this, we can achieve nested function classes which help make the neural network more accurate.

A problem encountered when implementing the Res-net the d21 website wants to define the class ResNet (d2l.Classifier). When trying to implement this, there were errors, and it did not work properly. It may have something to do with Google Collab or maybe it was not implementing it correctly. Using a GPU/cuda was useful to test quickly whether a change to the neural network was going to increase accuracy or not. Increasing the neuron count to above 4000 caused an error where there was not enough GPU RAM. Using multiple blocks above 1000 neurons caused the accuracy to plummet to 10%. Increasing the block count from 5 to 10 increased the accuracy from about 73% to 76%. Adding another convolution and batch normalization to the residual block g(x) did not improve accuracy and it caused a significant slowdown to the overall performance. The accuracy of the network stopped increasing by around 5000 iterations. There was no discussion with other team members because I was the only team member.

The result of the project was an accuracy of around 75%. It is around 5% better than my version of AlexNet but a Residual Network would probably be better as you scaled up the network more and more. With bigger machines more code blocks could be added which would add some accuracy to the final output.

Sources

Works Cited

“8.6. Residual Networks (ResNet) and ResNeXt — Dive into Deep Learning 1.0.0-Beta0 Documentation.” *D2l.ai*, d2l.ai/chapter\_convolutional-modern/resnet.html#resnet-model. Accessed 29 Apr. 2023.

“BatchNorm2d — PyTorch 2.0 Documentation.” *Pytorch.org*, pytorch.org/docs/stable/generated/torch.nn.BatchNorm2d.html#torch.nn.BatchNorm2d. Accessed 30 Apr. 2023.

“BatchNorm2d: How to Use the BatchNorm2d Module in PyTorch.” *Www.datascienceweekly.org*, www.datascienceweekly.org/tutorials/batchnorm2d-how-to-use-the-batchnorm2d-module-in-pytorch. Accessed 30 Apr. 2023.

“CIFAR-10.” *Wikipedia*, 29 Mar. 2021, en.wikipedia.org/wiki/CIFAR-10.

Ioffe, Sergey. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015.

Jason Brownlee. “A Gentle Introduction to the Rectified Linear Unit (ReLU) for Deep Learning Neural Networks.” *Machine Learning Mastery*, 20 Apr. 2019, machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.

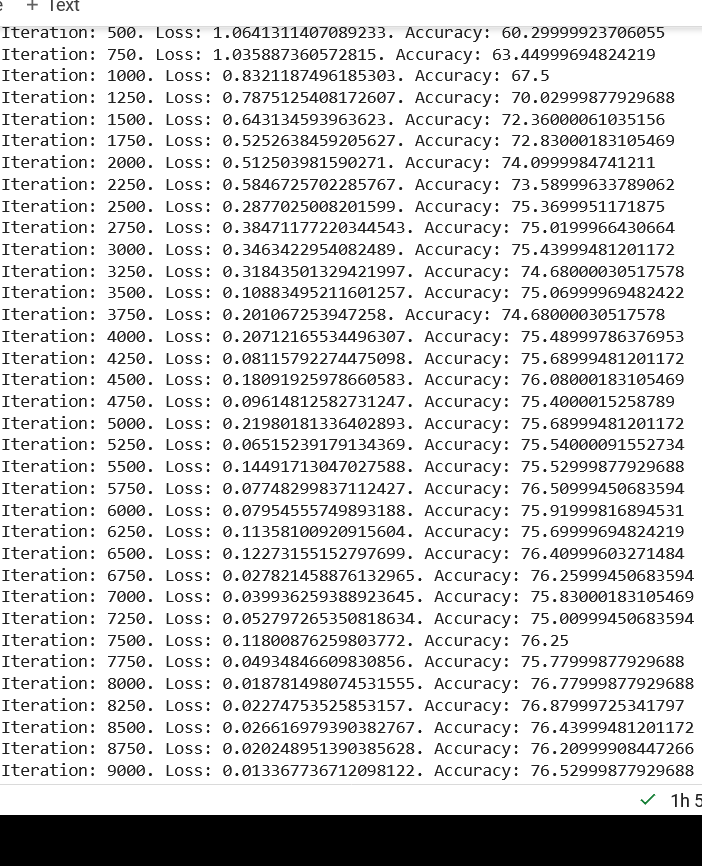
Mayur. “Image Classification Using Deep Learning.” *GitHub*, 25 Apr. 2023, github.com/Mayurji/Image-Classification-PyTorch/blob/main/ResNet.py. Accessed 29 Apr. 2023.

Wikipedia Contributors. “AlexNet.” *Wikipedia*, Wikimedia Foundation, 18 Oct. 2019, en.wikipedia.org/wiki/AlexNet.

---. “Convolutional Neural Network.” *Wikipedia*, Wikimedia Foundation, 27 Feb. 2019, en.wikipedia.org/wiki/Convolutional\_neural\_network.

Other screenshots.

‌



‌‌

Graphical user interface, text, application

Description automatically generated

