Application of VGG and Residual Neural Network to an Image Classification Problem

Boluwatife E. Awoyemi
Department of Mathematics and Statistics, Texas Tech University, 2500 Broadway, Lubbock, TX 79409, US

Description of Problem

In this project, we will be applying the Visual Geometry Group (VGG) neural network and the Residual Neural Network (ResNet) to image classification of the CIFAR-10 data set [1]. I chose the VGGNet and ResNet models used in this project by considering the ones with reasonably comparable depths. For example, it is apparently unreasonable to make comparisons between VGG-16 or VGG-19 networks and ResNet-18. A better match for VGG networks with such depths would be ResNet-50 or ResNet-101.

The CIFAR-10 data set consists of 60000, 3 X 32 X 32 RGB images with 10 non-overlapping classes: **airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck**.
Of the 60000 images, 50000 are the training set and the remaining 10000 are test set which was selected as random 1000 images per class. This data set was collected by Alex Krizhevsky and his group.

The aim of this project is to use these models that are considered deeper than the usual convolutional neural network (CNN) to classify the data set above more accurately.
We also hope to compare both networks to see how close or far apart the losses and accuracy values are after computation.

Neural Networks Architecture

VGG Neural Network Model

VGG-Visual Geometry Group is a model designed with block of layers with convolutions. The VGG-11 which is used in this project is a convolutional neural network architecture that consists of 11 layers, including 8 convolutional and max-pooling (5) layers, and 3 fully connected layers. In terms of blocks, this network has five convolutional blocks. Before the fully connected layers, the first two blocks have one convolution and max-pooling layer each and the latter three have two convolution and one max-pooling layers each. The number of output channel doubles after each block till the model has produced 512 feature maps, with the first block having 64 output channels. My application defines the convolution layers in the features module and the fully connected layers in the classifier module.

VGG-11 Architecture Framework:
<u>Convolutional (Conv) and Max-pooling (MaxPool) layers</u>

- Block 1
  Conv1: 64 features, kernel size 3x3, padding 1, stride 1
  MaxPool1: 2x2 max pooling, stride 2
- Block 2
  Conv2: 128 features, kernel size 3x3, padding 1, stride 1
  MaxPool2: 2x2 max pooling, stride 2
- Block 3
  Conv3: 256 features, kernel size 3x3, padding 1, stride 1
  Conv4: 256 features, kernel size 3x3, padding 1, stride 1
  MaxPool3: 2x2 max pooling, stride 2
- Block 4
  Conv5: 512 features, kernel size 3x3, padding 1, stride 1
  Conv6: 512 features, kernel size 3x3, padding 1, stride 1
  MaxPool4: 2x2 max pooling, stride 2
- Block 5
  Conv7: 512 features, kernel size 3x3, padding 1, stride 1
  Conv8: 512 features, kernel size 3x3, padding 1, stride 1
  MaxPool5: 2x2 max pooling, stride 2

Fully connected layers (FCL)

- FCL1: 4096 units
- FCL2: 4096 units
- FCL3: 10 units (output layer for CIFAR-10, which has 10 classes)

As explained above, convolutional layer 8 (Conv8) has 512 output channels where each channel represents a different feature the network learned and detected in the input data set. I learned that these features increase in abstractness and complexity as you go deeper into the network, and it is the fully connected layers after these convolutional layers that take these features and combine them to make predictions based on the learned representations.

We should also note that the 512 output channels in the last convolutional layer can vary in different network architectures. The choice of 512 channels in VGG-11 is based on empirical observations and considerations of model capacity for the specific tasks it was designed for.

Residual Neural Network Model

The lesson from the Dive into Deep Learning Website [3] says "ResNet uses four modules made up of residual blocks, each of which uses several residual blocks with the same number of output channels". After the first residual layer, the number of input features doubles whereas the height and width are halved for the next three layers. Then to complete the layers, we add a global average pooling layer, followed by the fully connected layer output. The python code I have also generates networks where we adjust channels and resolution by means of a 1 X 1 convolution before adding.

Excluding the convolution adjustment layer and the first layer, there are four convolutional layers in each module. Together with the first 7 X 7 convolutional layer, the global average pooling layer and the final fully connected layer, there are 18 layers in total. This is why this is called the ResNet-18 model. If we change the numbers of channels and residual blocks in the module, we can create deeper ResNet models.

Also, the ResNet-18 model has skip connections which give room for the network to learn residual mappings. This is what helps in avoiding the vanishing gradient problem and facilitates the training of deeper networks because the gradient gets to flow undisturbed.

ResNet-18 Architecture Framework:
<u>Convolutional (Conv) layers.</u>

Please note that each block consists of a batch normalization and activation function lines.

- Initial Convolution
  Conv0: 3 features, kernel size 3x3, padding 1, stride 1
  Output: 64 channels, kernel size 7x7, padding 3, stride 2

- Residual Layer (Block) 1
  Conv1: 64 features, kernel size 3x3, padding 1
  Conv2: 64 features, kernel size 3x3, padding 1
  Conv3: 64 features, kernel size 3x3, padding 1
  Adj1: 1 X 1 convolution to adjust dimensions
  Output: 64 channels (stays unchanged)

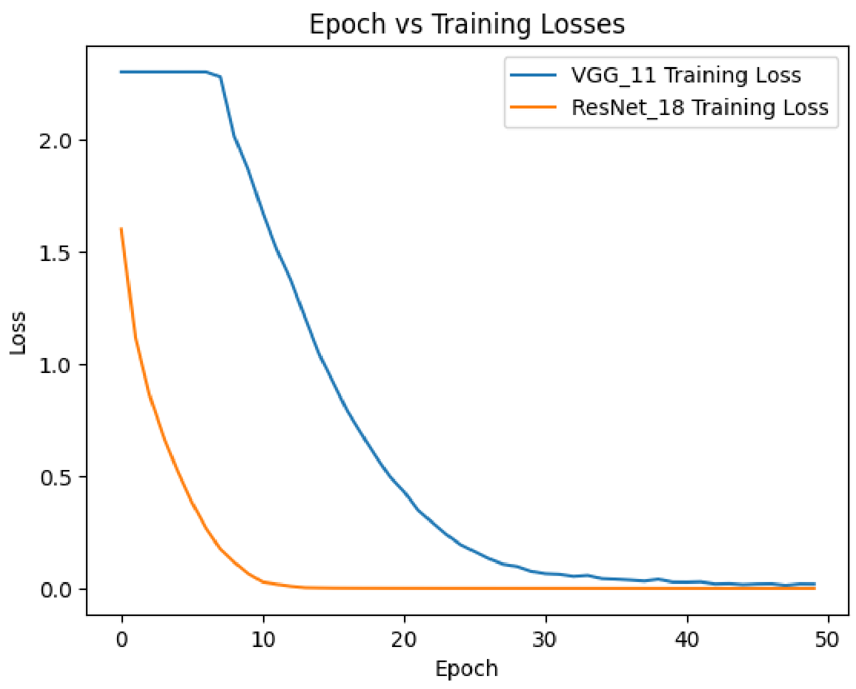
- Residual Layer (Block) 2
  Conv4: 64 features, kernel size 3x3, padding 1
  Conv5: 64 features, kernel size 3x3, padding 1
  Conv6: 64 features, kernel size 3x3, padding 1
  Conv7: 64 features, kernel size 3x3, padding 1
  Adj2: 1 X 1 convolution to adjust dimensions
  Output: 128 channels, stride 2 (due to the first convolutional layer in this block)

- Residual Layer (Block) 3
  Conv8: 128 features, kernel size 3x3, padding 1
  Conv9: 128 features, kernel size 3x3, padding 1
  Conv10: 128 features, kernel size 3x3, padding 1
  Conv11: 128 features, kernel size 3x3, padding 1
  Adj3: 1 X 1 convolution to adjust dimensions
  Output: 256 channels, stride 2 (due to the first convolutional layer in this block)

- Residual Layer (Block) 4
  Conv12: 256 features, kernel size 3x3, padding 1
  Conv13: 256 features, kernel size 3x3, padding 1
  Conv14: 256 features, kernel size 3x3, padding 1
  Conv15: 256 features, kernel size 3x3, padding 1
  Adj4: 1 X 1 convolution to adjust dimensions
  Output: 512 channels, stride 2 (due to the first convolutional layer in this block)
- Global Average Pooling
  Input: 512 channels
  Output: 512 channels
  Global average pooling layer to reduce spatial dimensions to 1 X 1
- Fully Connected Layer
  Input: 512 channels Output: Number of classes (10 for CIFAR-10)

This framework focuses on the convolutional layers and their impact on the number of channels. The stride in the first convolutional layer of each residual block affects the spatial dimensions and the number of channels in the output. This is why I added the convolutional layer adjustment at the end of each residual layer so as not to get the error of size mismatch when the layers are connecting to each other. The global average pooling layer also collapses the spatial dimensions to 1 X 1, and the fully connected layer produces the final output based on the learned features.

Results



Loss Comparison for both networks over 50 epochs

The figure above shows the decrease in losses as the number of epochs increase for the VGG-11 network and the ResNet-18 network.

Highest Accuracy results for each convolutional neural network

Neural Network	Batch Size	Epoch Size	Learning Rate	Highest Accuracy Epoch	Highest Accuracy Loss	Highest Accuracy
VGG11	128 (2)	50 (2)	0.01	47	78.00	0.0201
ResNet18	128 (2)	50 (2)	0.01	23	79.74	0.0007

Discussion

It is evident that these two neural network models are very good in training large datasets due to their structures. For the CIFAR-10 data set, the VGG model and the ResNet model are said to have accuarcy of over 90% especially in the models with much more layers. However, applying the two models to the same problem shows some stark differences between them.

The diagram above shows us that the ResNet model converges faster than the VGG model. After a little over 10 epochs, the ResNet model almost converged to 0 whereas it took the VGG model over 40 epochs. We can also see that the VGG training loss started and remained in the higher loss region than the ResNet training loss and it also took the VGG model about 10 epochs for its accuracy to start increasing (with corresponding drop in loss).

The code results shows that the accuracy for the ResNet started from a much higher value on the first iteration than the VGG's. The ResNet also had the overall highest accuracy (79.74%) at epoch 23 with a loss much lower than the VGG's loss. All these points point to ResNet being the better solution for the problem in this case.

This accuracy in ResNet is can be attributed to the fact that it deals with the vanishing gradient problem and its overall structure. Still, this does not mean that the VGG-11 model is a bad model. The VGG is a a network designed to take on more layers so higher number of layers are liable to show better performance. One of the things I also noted about the VGG model was that it took a shorter time to arrive at 50 epochs that in took the ResNet model.

In conclusion, for the CIFAR-10 data set used in this project, the Residual Neural Network had a higher accuracy right from the bat, and also had the higher accuracy overall.

References

1. Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

2. Networks Using Blocks (VGG) - Dive into Deep Learning https://d2l.ai/chapter_convolutional-modern/vgg.html

3. Networks Using Blocks (VGG) - Dive into Deep Learning https://d2l.ai/chapter_convolutional-modern/resnet.html