Image Classification Architectures

Khalid Hasan (id: 17-34538-2)[a], Aninda Kumar Sharma (id: 17-35513-3)[a], Syed Fahim Intisar (id:18-36355-1)[a], Nur-A-Marzan Dipro (id: 19-39529-1)[a]

[a]*Department of Computer Sciences, American International University-Bangladesh*

Abstract

Computer vision is a collection of scientific methods that deals with training of computers to interpret the visual world. It is an AI technology. Deep learning is an artificial intelligence (AI) function that tries to follow the workings of the human brain in processing data and developing patterns for making decisions. In deep learning, a convolutional neural network or CNN is a class of deep neural network. It is generally applied to examine visual imageries. Object detection is a computer technology that detects instances of semantic objects of a certain class. Object detection combines image classification and object localization. Combined, all of these problems are called object recognition.

1. Introduction

AlexNet is the name of a convolutional neural network (CNN) architecture, designed by Alex Krizhevsky in collaboration with Ilya Sutskever and Geoffrey Hinton, who was Krizhevsky's Ph.D. advisor. AlexNet competed in the ImageNet Large Scale Visual Recognition Challenge on September 30, 2012. The network achieved a top-5 error of 15.3%, more than 10.8 percentage points lower than that of the runner-up. The original paper's primary result was that the depth of the model was essential for its high performance, which was computationally expensive, but made feasible due to the utilization of graphics processing units (GPUs) during training.

DenseNet or **Densely Connected Convolutional Networks** are convolutional network architectures which are the next step on the journey to keep increasing the depth of deep convolutional networks. Densely Connected Convolutional Networks was introduced by Gao Huang, Zhuang Liu et al in 2016. Their research paper was published in 2017 CVPR which got the 'Best Paper Award' with over 2000 citations. DenseNet simplify the connectivity pattern between layers introduced in other architectures like Highway Networks, VGG16, Fractal Networks.

residual neural network (**ResNet**) is an artificial neural network (ANN) of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. Residual neural networks do this by utilizing *skip connections*, or *shortcuts* to jump over some layers. Typical *ResNet* models are implemented with double- or triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between. An additional weight matrix may be used to learn the skip weights; these models are known as HighwayNets. Models with several parallel skips are referred to as *DenseNet*. In the context of residual neural networks, a non-residual network may be described as a *plain network*.

VGG16 is a network model based on convolutional neural network proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The "16" and "19" stand for the number of weight layers in the network. ImageNet, a dataset of over 14 million images belonging to 1000 classes, is a dataset in which VGG 16: achieves top-5 test accuracy of 92.7%. The model was one of the most famous submissions at the ILSVRC-2014 conference. With this scheme, large kernel-sized filters are replaced with 33 kernel-sized filters one after another, thereby improving over AlexNet. NVIDIA Titan Black GPU's were used to train VGG16 for weeks.

2. Literature Review

Let's talk about **Alexnet** first:

To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task means that this problem cannot be specified even by a dataset as large as ImageNet, so our model should also have lots of prior knowledge to compensate for all the data we don't have. Convolutional neural networks (CNNs) constitute one such class of models. Their capacity can be con-trolled by varying their depth and breadth, and they also make strong and mostly correct assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies). Thus, compared to standard feedforward neural networks with similarly-sized layers, CNNs have much fewer connections and parameters, and so they are easier to train, while their theoretically-best performance is likely to be only slightly worse. Despite the attractive qualities of CNNs, and despite the relative efficiency of their local architecture, they have still been prohibitively expensive to apply in large scale to high-resolution images. Luckily, current GPUs, paired with a highly-optimized implementation of 2D convolution, are powerful enough to facilitate the training of interestingly-large CNNs, and recent datasets such as ImageNet contain enough labeled examples to train such models without severe overfitting.

Then comes the term of **Densenet**:


```
tf.keras.applications.DenseNet121(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
)
```

'include top' defines if it can include the fully-connected layer at the top of the network or not. Also 'weights' can be one of None, pre-trained 'ImageNet' or the path to the weights file to be loaded. In input tensor, optional Keras tensor which is output of layers. Input () is provided to use as the image input for the model. input shape can be optional shape tuple if include top is False. If not, the input shape must be (224, 224, 3) with 'channels last' data format or (3, 224, 224) with 'channels first' data format. It must have exactly 3 inputs channels, and width and height would be no less than 32. Optional pooling mode for feature extraction when include top is 'False'. 'None' will signal that the output of the model will be the 4D tensor output of the previous convolutional block. Another option 'avg' means that global average pooling will be applied to the output of the previous convolutional block. In this case, the output of the model will be a 2D tensor in this case. Lastly 'max' signals the application of global max pooling. In the case of classes, optional number of classes are here to classify images into something. It is specified only if include top is True, and no weights argument is given.

Now let's look into **Resnet**:

We build extra layers in the Deep Neural Networks, which result in enhanced precision and performance, to resolve a complex problem mainly. The insight that these layers will gradually learn more complicated aspects after adding more levels. For example, if images are detected in the first layer, edges may be detected, textures can be recognized in the second layer, and the third layer can similarly learn to detect things. However, it was revealed that with the standard Convolutional neural network model, there is a maximum depth threshold. This is a figure that describes error for training and testing data for a network of 20 layers and a network of 56 layers. We can see that error% for 56-layer is more than a 20-layer network in both cases of training data as well as testing data. This suggests that with adding more layers on top of a network, its performance degrades. This could be blamed on the optimization function, initialization of the network and more importantly vanishing gradient problem. You might be thinking that it could be a result of overfitting too, but here the error% of the 56-layer network is worst on both training as well as testing data which does not happen when the model is overfitting.

Finally, we will glance at **VGGnet**:

A 224 x 224 RGB picture is the input for the layer cov1. In this computation, the image goes through a stack of convolutional layers whose receptive fields are extremely small: 3*3 which is the smallest size in which left/right, up/down, and center can be detected. There is also a configuration which uses 1*1 convolution filters as well, which is essentially a linear transformation of the input channels followed by nonlinearity. A convolution stride of 1 pixel has been determined. When using a layer with a 3*3 convolution, the

input is set up so the spatial resolution will be preserved after convolution, where the padding will be 1 pixel. A few of the convolutions are followed by five max-pooling layers for spatial pooling. Maximization is combined with stride 2 over a window of 2*2 pixels. Following a stack of convolutional layers (which are different in depth in different architectures), there are three fully connected (FC) layers. The first two have 4096 channels each, the third uses ILSVRC-1000 classification and has 1000 channels. Lastly, there is the soft-max layer. Every network has the same configuration of the layers that are fully connected.

3. Discussion

Alexnet:

The architecture of Alexnet network is summarized. It contains eight learned layers —five convolutional and three fully-connected.

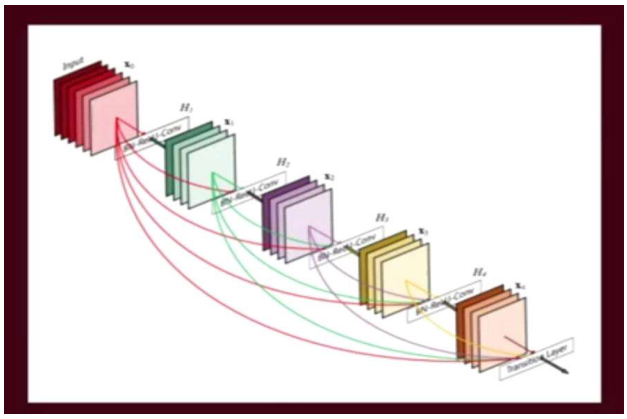
- **ReLU Nonlinearity**: AlexNet uses Rectified Linear Units (ReLU) instead of the tanh function, which was standard at the time. ReLU's advantage is in training time; a CNN using ReLU was able to reach a 25% error on the CIFAR-10 dataset six times faster than a CNN using tanh.
- **Multiple GPUs:** Back in the day, GPUs were still rolling around with 3 gigabytes of memory (nowadays those kinds of memory would be rookie numbers). This was especially bad because the training set had 1.2 million images. AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time.
- **Overlapping Pooling:** CNNs traditionally "pool" outputs of neighboring groups of neurons with no overlapping. However, when the authors introduced overlap, they saw a reduction in error by about 0.5% and found that models with overlapping pooling generally find it harder to overfit.

The Overfitting Problem: AlexNet had 60 million parameters, a major issue in terms of overfitting. Two methods were employed to reduce overfitting:

- **Data Augmentation:** The authors used label-preserving transformation to make their data more varied. Specifically, they generated image translations and horizontal reflections, which increased the training set by a factor of 2048. They also performed Principle Component Analysis (PCA) on the RGB pixel values to change the intensities of RGB channels, which reduced the top-1 error rate by more than 1%.
- **Dropout:** This technique consists of "turning off" neurons with a predetermined probability (e.g. 50%). This means that every iteration uses a different sample of the model's parameters, which forces each neuron to have more robust features that can be used with other random neurons. However, dropout also increases the training time needed for the model's convergence.

Densenet:

What it does here is that the input to the next layer is the concatenation of all the previous layer inputs.



From the figure, we see that, in layer x3 the input is the sum of output from x2, output from x1, output from x0 and the original input all concatenate it to make one really deep feature map with the same spatial resolution. So, adding dense blocks, like 10 dense blocks all in a row connected to each other in this way get really deep. So, what they do in the dense net model is they break up the dense blocks in such a way so that they can use a 'one by one convolution'. And one by one convolution preserves the spatial resolution but it shrinks the depth of the feature map and then they have max pooling to reduce the feature map size. For this 'back to the connectivity' pattern, one of the big claims it has is that this is more parameter efficient. In some models you have some of the parameters there just to preserve certain information from the previous layer. In dense Nets since the information is concatenated ahead, one doesn't need to have information preservation functions. Big advantage for DenseNet is that it can easily propagate the error signal to earlier layers more directly.

Resnet:
Say we have a shallow network and a deep network that maps an input 'x' to output 'y' by using the function H(x). We want the deep network to perform at least as good as the

shallow network and not degrade the performance as we saw in case of plain neural networks (without residual blocks). One way of achieving so is if the additional layers in a deep network learn the identity function and thus their output equals inputs which do not allow them to degrade the performance even with extra layers. It has been seen that residual blocks make it exceptionally easy for layers to learn identity functions. It is evident from the formulas above. In plain networks the output is

H (X) =f (X)

So, to learn an identity function, f(x) must be equal to x which is grader to attain whereas in case of ResNet, which has output:

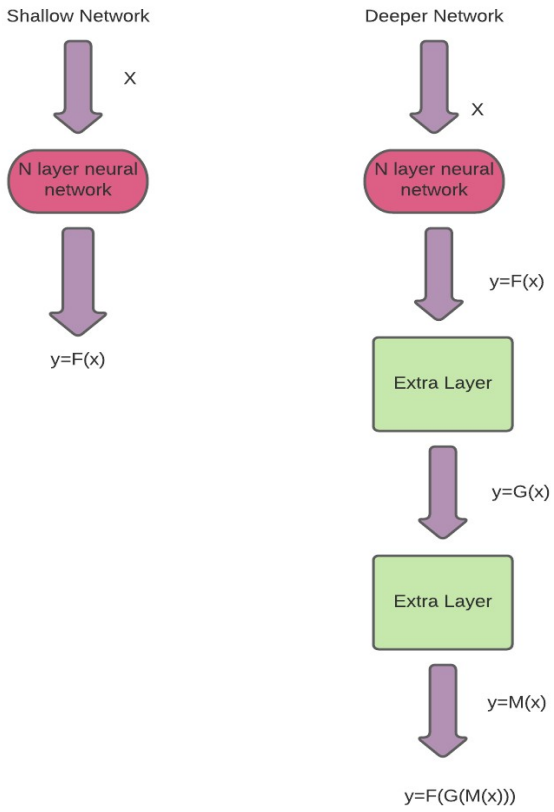
H (X) = f (X)+X,

f(X) =0

H(X) =X

All we need is to make f(x)=0 which is easier and we will get x as output which is also our input.

In the best-case scenario, additional layers of the deep neural network can better approximate the mapping of 'x' to output 'y' than it's the shallower counterpart and reduces the error by a significant margin. And thus, we expect ResNet to perform equally or better than the plain deep neural networks.



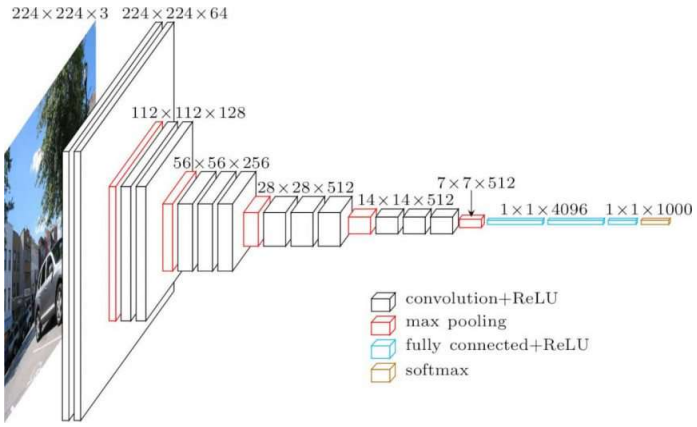
Using ResNet has significantly enhanced the performance of neural networks with more layers and here is the plot of error% when comparing it with neural networks with plain layers.

Keras is an open-source neural network library written in Python which is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. It is designed to enable fast experimentation with deep neural networks. Keras Applications include the following ResNet implementations and provide ResNet V1 and ResNet V2 with 50, 101, or 152 layers.

ResNet50
ResNet101
ResNet152
ResNet50V2
ResNet101V2
ResNet152V2

The primary difference between ResNetV2 and the original (V1) is that V2 uses batch normalization before each weight layer.

VGGnet:
The VGG16 model outperformed previous generations by a considerable margin in ILSVRC2012 and ILSVRC2013. Likewise, the VGG16 result is in direct competition with GoogleNet (a 6.7% error result) for the classification task winner and overtakes Clarifai's winning ILSVRC-2013 submission, which achieved 11.2% with external training data and 11.7% without this. When compared to a single-net architecture, the VGG16 architecture performs better (7.0% test error), outperforming Google.Net by 0.9%. In this study, it is demonstrated that the depth of the representation greatly enhances the classification accuracy, and a conventional ConvNet architecture with a substantially increased depth is capable of achieving the best classification accuracy on the ImageNet dataset. Some Drawbacks are;
1. Trains take an excruciatingly long time.
2. Among the network configuration weights the disk/bandwidth requirements are quite high.



4. Conclusion

AlexNet is an incredibly powerful model capable of achieving high accuracies on very challenging datasets. However, removing any of the convolutional layers will drastically degrade AlexNet's performance. AlexNet is a leading architecture for any object-detection task and may have huge applications in the computer vision sector of artificial intelligence problems. In the future, AlexNet may be adopted more than CNNs for image tasks.
As a milestone in making deep learning more widely-applicable, AlexNet can also be credited with bringing deep learning to adjacent fields such as natural language processing and medical image analysis.

The DenseNet model solve the problem of lack of depth in convolutional networks by guaranteeing the flow of maximum information. In here every layer is simply connected directly with each other. The problem with just stacking on more convolutional layers is that when you back propagate the gradient from a higher layer to a lower layer, you must take the partial derivative of each hidden unit which can potentially get really small and thus the network hardly updates its weights as it rains. These convolutional network architectures called densely connected convolutional networks try to improve the convolutional network system by increasing the depth of the convolutional neural

network.
ResNets are being implemented in almost all of AI's new tech to create state-of-the-art systems. The principle on which ResNets work is to build a deeper network compared to other plain networks and simultaneously find an optimized number of layers to negate the vanishing gradient problem. ResNeXt have also been deployed on CIFAR-10 dataset, the results are remarkable and its architecture gave a top-5 error rate with 3.30% thus winning second position in ILSVRC competition.
VGG16 is over 533MB due to its depth and number of fully-connected nodes. The problem makes it difficult to deploy VGG.VGG16 is used in many deep learning image classification problems. However, it would be better to use lower-level network architectures such as SqueezeNet, GoogleNet, and so on.

References

[1] https://arxiv.org/abs/1505.06798

[2] https://keras.io/api/applications/vgg/

[3] Very Deep Convolutional Networks for Large-Scale Image Recognition (ICLR 2015)

[4] https://www.kaggle.com/saptarsi/using-pre-trained-vgg-model

[5] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[7] G. Huang, Z. Liu and L. van der Maaten, "Densely Connected Convolutional Networks," 2017. (https://arxiv.org/abs/1608.06993)

[8] R. Kumar, K. Gredd and J. Schmidhuber, "Highway Networks," 2015.

[9] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", 2014

[10] G. Larsson, M. Maire and G. Shakhnarovich, “FractalNet: Ultra-Deep Neural Networks without Residuals,” 2017.

[11] DenseNet (https://keras.io/api/applications/densenet/)

[12] R.M. Bell and Y. Koren. Lessons from the netflix prize challenge.ACM SIGKDD Explorations Newsletter,9(2):75–79, 2007.

[13] A. Berg, J. Deng, and L. Fei-Fei.Large scale visual recognition challenge 2010. www.image-net.org/challenges. 2010.

[14] L. Breiman. Random forests.Machine learning, 45(1):5–32, 2001.

[15] D. Cire¸san, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification.Arxiv preprint arXiv:1202.2745, 2012.

[16] D.C. Cire¸san, U. Meier, J. Masci, L.M. Gambardella, and J. Schmidhuber. High-performance neuralnetworks for visual object classification.Arxiv preprint arXiv:1102.0183, 2011.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale HierarchicalImage Database. InCVPR09, 2009.

[18] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei.ILSVRC-2012, 2012.URLhttp://www.image-net.org/challenges/LSVRC/2012/.

[18] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: Anincremental bayesian approach tested on 101 object categories.Computer Vision and Image Understand-ing, 106(1):59–70, 2007.

[19] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, Cali-fornia Institute of Technology, 2007. URLhttp://authors.library.caltech.edu/7694.

[20] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov. Improving neural net-works by preventing co-adaptation of feature detectors.arXiv preprint arXiv:1207.0580, 2010