DATA 586: Advanced Machine Learning

2023W2

Shan Du

Convolutional Neural Networks (LeNet)

- Before CNN, we used an MLP where we flattened each image from a 28×28 matrix (MNIST image) into a fixed-length 784-dimensional vector, and thereafter processed them in fully connected layers.
- With convolutional layers, we can retain the spatial structure in our images.
- As an additional benefit of replacing fully connected layers with convolutional layers, we will enjoy more parsimonious models that require far fewer parameters.

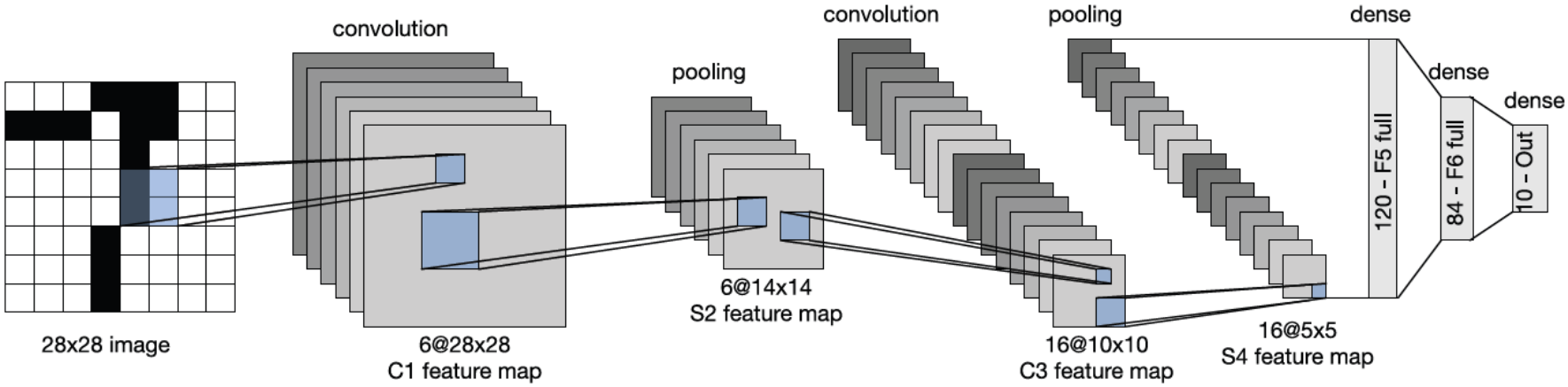
Convolutional Neural Networks (LeNet)

- *LeNet*, among the first published CNNs, has captured wide attention for its performance on computer vision tasks.
- The model was introduced by (and named for) Yann LeCun for the purpose of recognizing handwritten digits in images (LeCun et al., 1998). His team published the first study to successfully train CNNs via backpropagation.
- At the time LeNet achieved outstanding results beating the performance of support vector machines, then a dominant approach in supervised learning, achieving an error rate of less than 1% per digit. LeNet was eventually adapted to recognize digits for processing deposits in ATM machines.

Convolutional Neural Networks (LeNet)

- At a high level, LeNet (LeNet-5) consists of two parts: (i) a convolutional encoder consisting of two convolutional layers; and (ii) a dense block consisting of three fully connected layers.

Convolutional Neural Networks (LeNet)



Data flow in LeNet. The input is a handwritten digit, the output a probability over 10 possible outcomes.

Convolutional Neural Networks (LeNet)

- The basic units in each convolutional block are a convolutional layer, a sigmoid activation function, and a subsequent average pooling operation.
- Note that while ReLUs and maxpooling work better, these discoveries had not yet been made at the time. Each convolutional layer uses a 5×5 kernel and a sigmoid activation function.

Convolutional Neural Networks (LeNet)

- These layers map spatially arranged inputs to a number of two-dimensional feature maps, typically increasing the number of channels.
- The first convolutional layer has 6 output channels, while the second has 16.
- Each 2×2 pooling operation (stride 2) reduces dimensionality by a factor of 4 via spatial downsampling.
- The convolutional block emits an output with shape given by (batch size, number of channel, height, width).

Convolutional Neural Networks (LeNet)

- In order to pass output from the convolutional block to the dense block, we must flatten each example in the minibatch.
- LeNet's dense block has three fully connected layers, with 120, 84, and 10 outputs, respectively. Because we are still performing classification, the 10-dimensional output layer corresponds to the number of possible output classes.

Modern Convolutional Neural Networks

- AlexNet
- VGG
- NiN
- GoogLeNet
- ResNet
- DenseNet

Deep Convolutional Neural Networks (AlexNet)

- Although CNNs were well known in the computer vision and machine learning communities following the introduction of LeNet (LeCun et al., 1995), they did not immediately dominate the field.
- Although LeNet achieved good results on early small datasets, the performance and feasibility of training CNNs on larger, more realistic datasets had yet to be established.

Representation Learning

- Different from conventional machine learning methods, the pioneers of NNs believed that features themselves ought to be learned.
- Moreover, they believed that to be reasonably complex, the features ought to be hierarchically composed with multiple jointly learned layers, each with learnable parameters.
- In the case of an image, the lowest layers might come to detect edges, colors, and textures, in analogy to how the visual system in animals processes its input.

Representation Learning

- The first modern CNN (Krizhevsky et al., 2012), named AlexNet after one of its inventors, Alex Krizhevsky, is largely an evolutionary improvement over LeNet.
- It achieved excellent performance in the 2012 ImageNet challenge.
- Interestingly in the lowest layers of the network, the model learned feature extractors that resembled some traditional filters.

Representation Learning

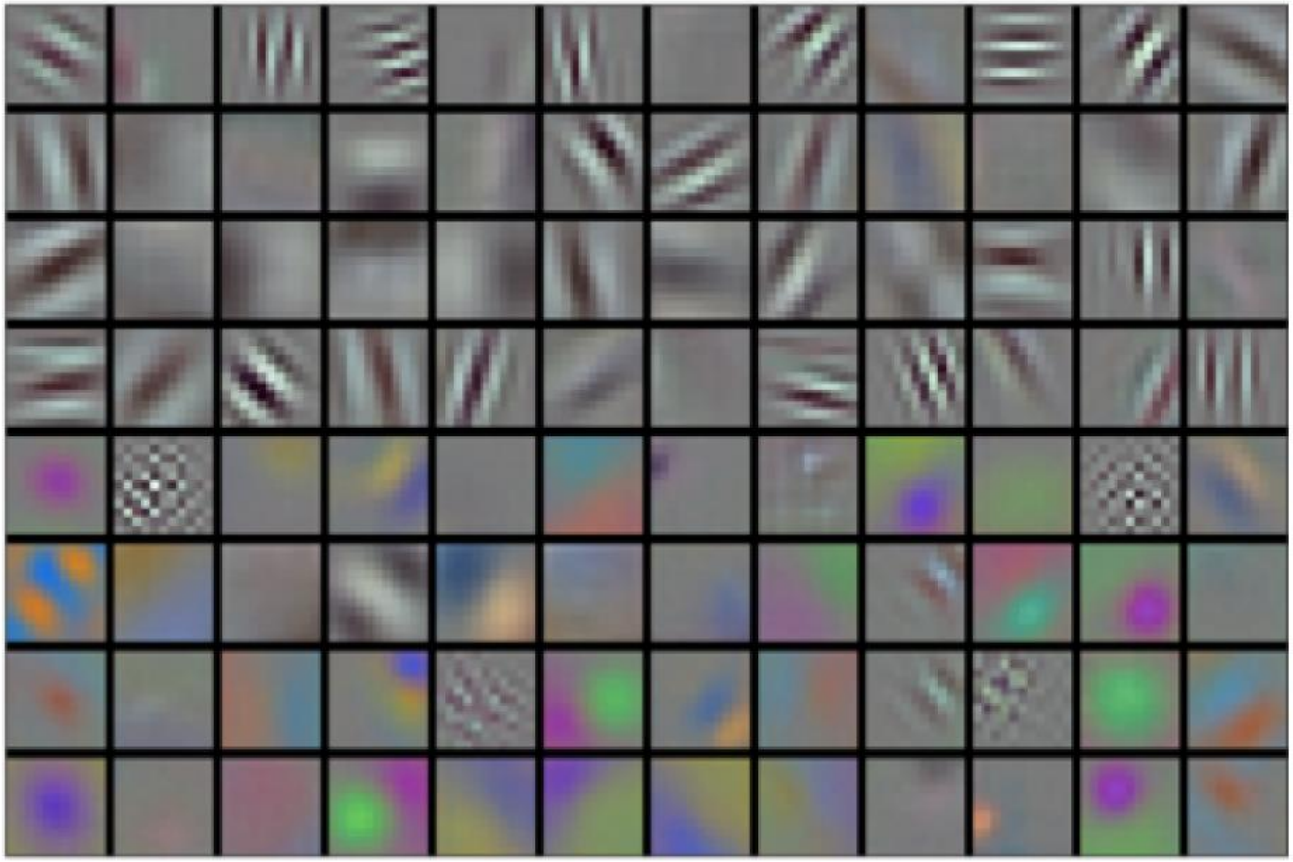


Image filters learned by the first layer of AlexNet. Reproduction courtesy of Krizhevsky et al. (2012).

Representation Learning

- Higher layers in the network might build upon these representations to represent larger structures, like eyes, noses, blades of grass, and so on.
- Even higher layers might represent whole objects like people, airplanes, dogs, or frisbees.
- Ultimately, the final hidden state learns a compact representation of the image that summarizes its contents such that data belonging to different categories can be easily separated.

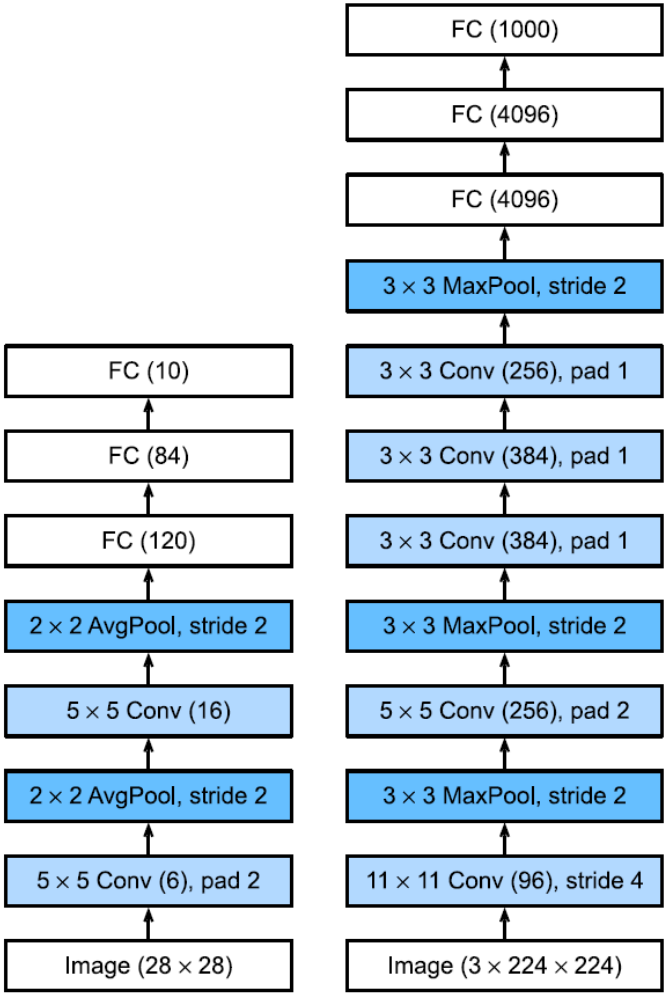
Deep Convolutional Neural Networks (AlexNet)

- AlexNet (2012) and its precursor LeNet (1995) share many architectural elements. This begs the question: why did it take so long?
- A key difference is that over the past two decades, the amount of data and computing power available had increased significantly.
- As such AlexNet was much larger: it was trained on much more data, and on much faster GPUs, compared to the CPUs available in 1995.

Deep Convolutional Neural Networks (AlexNet)

- AlexNet, which employed an 8-layer CNN, won the ImageNet Large Scale Visual Recognition Challenge 2012 by a large margin (Russakovsky et al., 2013).
- This network showed, for the first time, that the features obtained by learning can transcend manually-designed features, breaking the previous paradigm in computer vision.

Deep Convolutional Neural Networks (AlexNet)



From LeNet (left) to AlexNet (right).

Deep Convolutional Neural Networks (AlexNet)

- The architectures of AlexNet and LeNet are strikingly similar.
- There are also significant differences between AlexNet and LeNet. First, AlexNet is much deeper than the comparatively small LeNet5. AlexNet consists of eight layers: five convolutional layers, two fully connected hidden layers, and one fully connected output layer. Second, AlexNet used the ReLU instead of the sigmoid as its activation function.

Networks Using Blocks (VGG)

- The design of neural network architectures has grown progressively more abstract, with researchers moving from thinking in terms of individual neurons to whole layers, and now to blocks, repeating patterns of layers.
- The idea of using blocks first emerged from the Visual Geometry Group (VGG) at Oxford University, in their eponymously-named VGG network (Simonyan
- and Zisserman, 2014). It is easy to implement these repeated structures in code with any modern deep learning framework by using loops and subroutines.

VGG Blocks

- The basic building block of CNNs is a sequence of the following: (i) a convolutional layer with padding to maintain the resolution, (ii) a nonlinearity such as a ReLU, (iii) a pooling layer such as max-pooling to reduce the resolution.
- One of the problems with this approach is that the spatial resolution decreases quite rapidly. In particular, this imposes a hard limit of $\log_2 d$ convolutional layers on the network before all dimensions (d) are used up. For instance, in the case of ImageNet, it would be impossible to have more than 8 convolutional layers in this way.

VGG Blocks

- The key idea of Simonyan and Zisserman (2014) was to use multiple convolutions in between downsampling via max-pooling in the form of a block.
- They were primarily interested in whether deep or wide networks perform better. For instance, the successive application of two 3×3 convolutions touches the same pixels as a single 5×5 convolution does.

VGG Blocks

- In a rather detailed analysis they showed that deep and narrow networks significantly outperform their shallow counterparts.
- Stacking 3×3 convolutions has become a gold standard in later deep networks.
- Consequently, fast implementations for small convolutions have become a staple on GPUs.

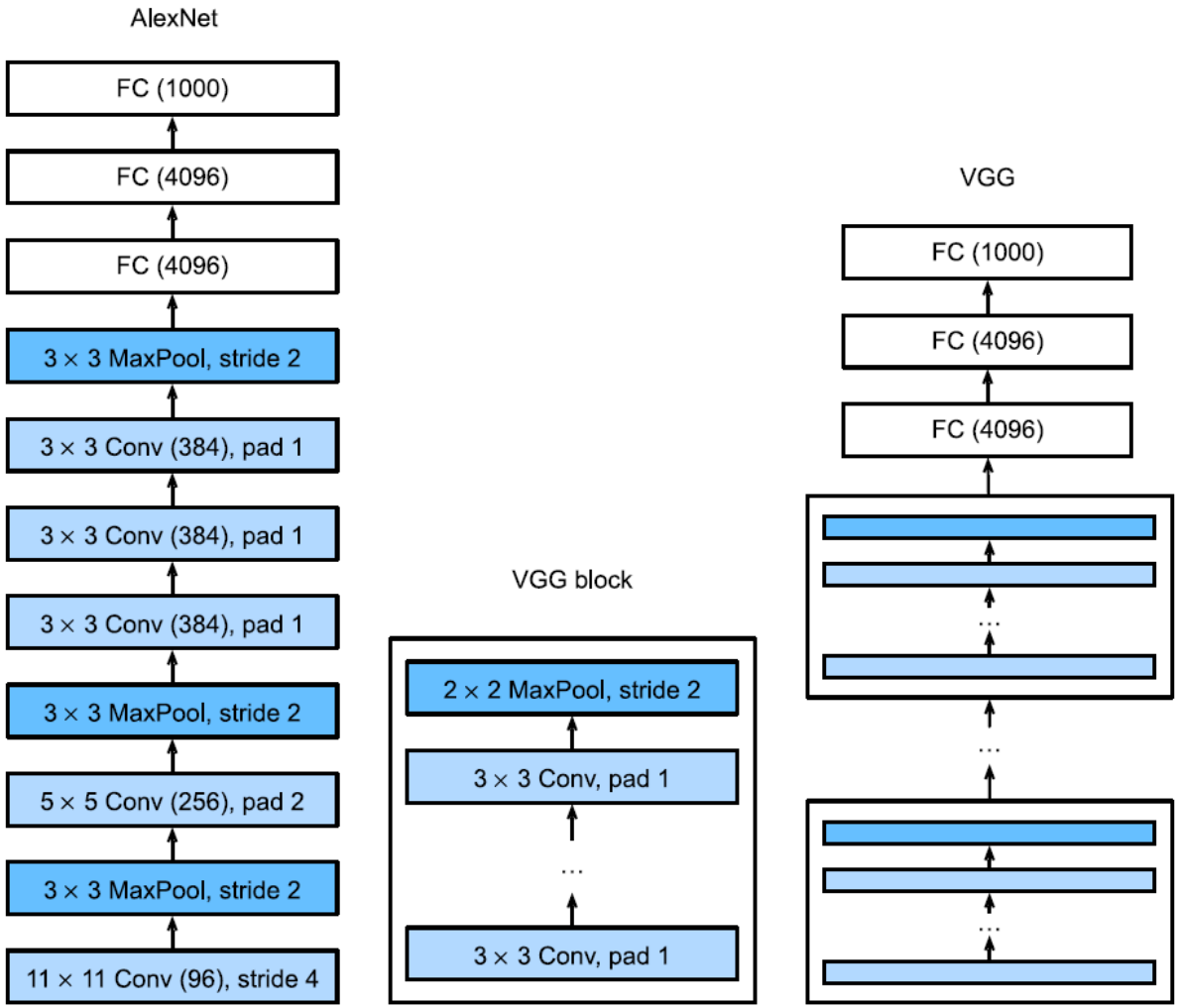
VGG Blocks

- A VGG block consists of a sequence of convolutions with 3×3 kernels with padding of 1 (keeping height and width) followed by a 2×2 max-pooling layer with stride of 2 (halving height and width after each block).

VGG Network

- Like AlexNet and LeNet, the VGG Network can be partitioned into two parts: the first consisting mostly of convolutional and pooling layers and the second consisting of fully connected layers that are identical to those in AlexNet.
- The key difference is that the convolutional layers are grouped in nonlinear transformations that leave the dimensonality unchanged, followed by a resolution-reduction step.

VGG Network



From AlexNet to VGG. The key difference is that VGG consists of blocks of layers, whereas AlexNets layers are all designed individually.

VGG Network

- The original VGG network had 5 convolutional blocks, among which the first two have one convolutional layer each and the latter three contain two convolutional layers each.
- The first block has 64 output channels and each subsequent block doubles the number of output channels, until that number reaches 512. Since this network uses 8 convolutional layers and 3 fully connected layers, it is often called VGG-11.

Network in Network (NiN)

- LeNet, AlexNet, and VGG all share a common design pattern: extract features exploiting spatial structure via a sequence of convolutions and pooling layers and post-process the representations via fully connected layers.
- This design poses two major challenges. First, the fully connected layers at the end of the architecture consume tremendous numbers of parameters. For instance, even a simple model such as VGG-11 requires a monstrous 25088 × 4096 matrix, occupying almost 400MB of RAM in single precision (FP32). This is a significant impediment to computation, in particular on mobile and embedded devices.

Network in Network (NiN)

- Second, it is equally impossible to add fully connected layers earlier in the network to increase the degree of nonlinearity: doing so would destroy the spatial structure and require potentially even more memory.

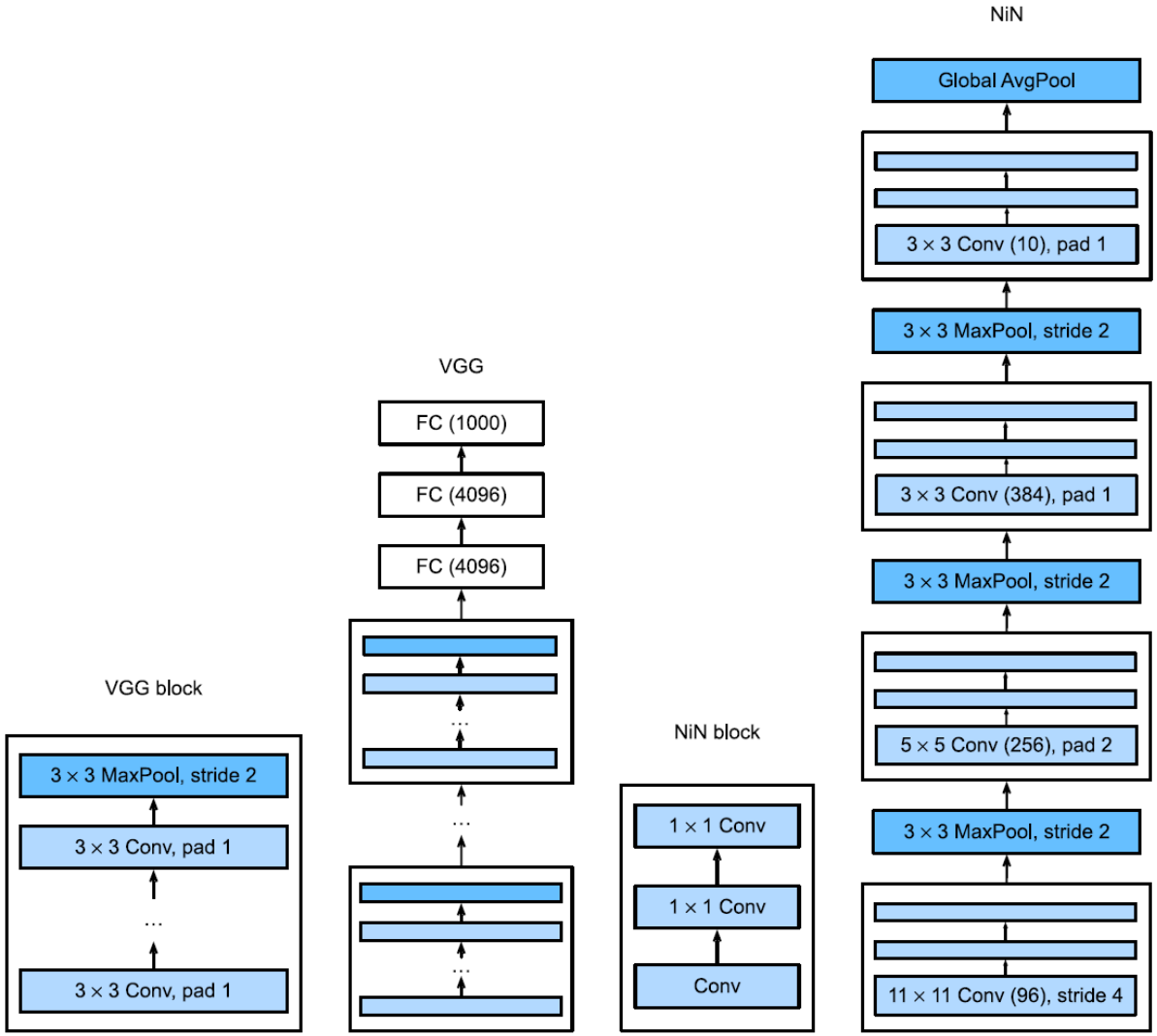
NiN Blocks

- The network in network (NiN) blocks (Lin et al., 2013) offer an alternative, capable of solving both problems in one simple strategy.
- They were proposed based on a very simple insight: (i) use 1×1 convolutions to add local nonlinearities across the channel activations and (ii) use global average pooling to integrate across all locations in the last representation layer.

NiN Blocks

- The idea behind NiN is to apply a fully connected layer at each pixel location (for each height and width). The resulting 1×1 convolution can be thought as a fully connected layer acting independently on each pixel location.
- The initial convolution is followed by 1×1 convolutions, (whereas VGG retains 3×3 convolutions) and in the end where we no longer require a giant fully connected layer.

NiN Model



Comparing the architectures of VGG and NiN, and of their blocks.

NiN Model

- NiN avoids fully connected layers altogether. Instead, NiN uses a NiN block with a number of output channels equal to the number of label classes, followed by a global average pooling layer, yielding a vector of logits.
- This design significantly reduces the number of required model parameters, albeit at the expense of a potential increase in training time.

Multi-Branch Networks (GoogLeNet)

- In 2014, GoogLeNet won the ImageNet Challenge (Szegedy et al., 2015), using a structure that combined the strengths of NiN (Lin et al., 2013), repeated blocks (Simonyan and Zisserman, 2014), and a cocktail of convolution kernels.
- It is arguably also the first network that exhibits a clear distinction among the stem (data ingest), body (data processing), and head (prediction) in a CNN. This design pattern has persisted ever since in the design of deep networks: the stem is given by the first 2–3 convolutions that operate on the image. They extract low-level features from the underlying images. This is followed by a body of convolutional blocks. Finally, the head maps the features obtained so far to the required classification, segmentation, detection, or tracking problem at hand.

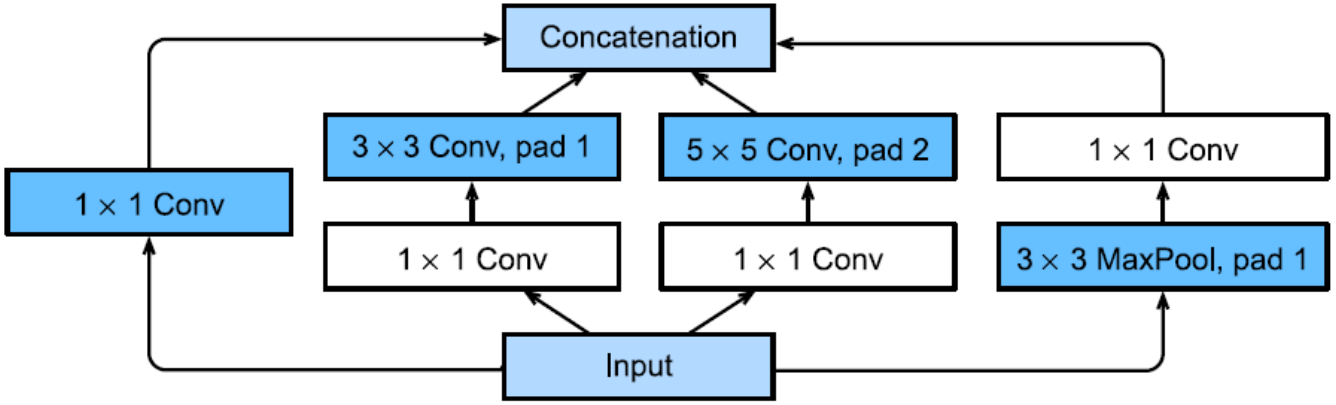
Multi-Branch Networks (GoogLeNet)

- The key contribution in GoogLeNet was the design of the network body. It solved the problem of selecting convolution kernels in an ingenious way.
- While other works tried to identify which convolution, ranging from 1×1 to 11×11 would be best, it simply concatenated multi-branch convolutions.

Inception Blocks

- The basic convolutional block in GoogLeNet is called an Inception block, stemming from the meme “we need to go deeper” of the movie Inception.
- The inception block consists of four parallel branches.

Inception Blocks

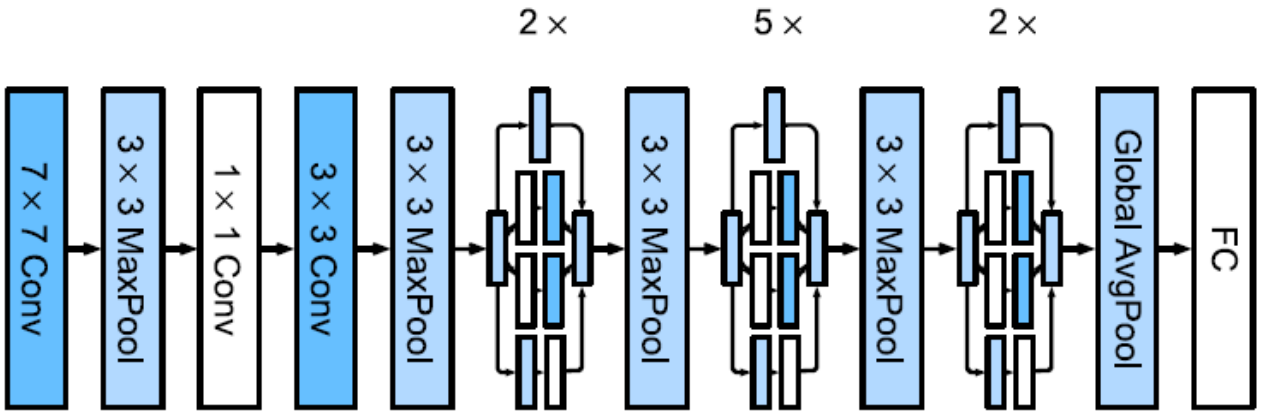


Structure of the Inception block.

The first three branches use convolutional layers with window sizes of 1×1, 3×3, and 5×5 to extract information from different spatial sizes. The middle two branches also add a 1×1convolution of the input to reduce the number of channels, reducing the model's complexity. The fourth branch uses a 3 × 3 max-pooling layer, followed by a 1×1 convolutional layer to change the number of channels.

The four branches all use appropriate padding to give the input and output the same height and width. Finally, the outputs along each branch are concatenated along the channel dimension and comprise the block's output.

GoogLeNet Model



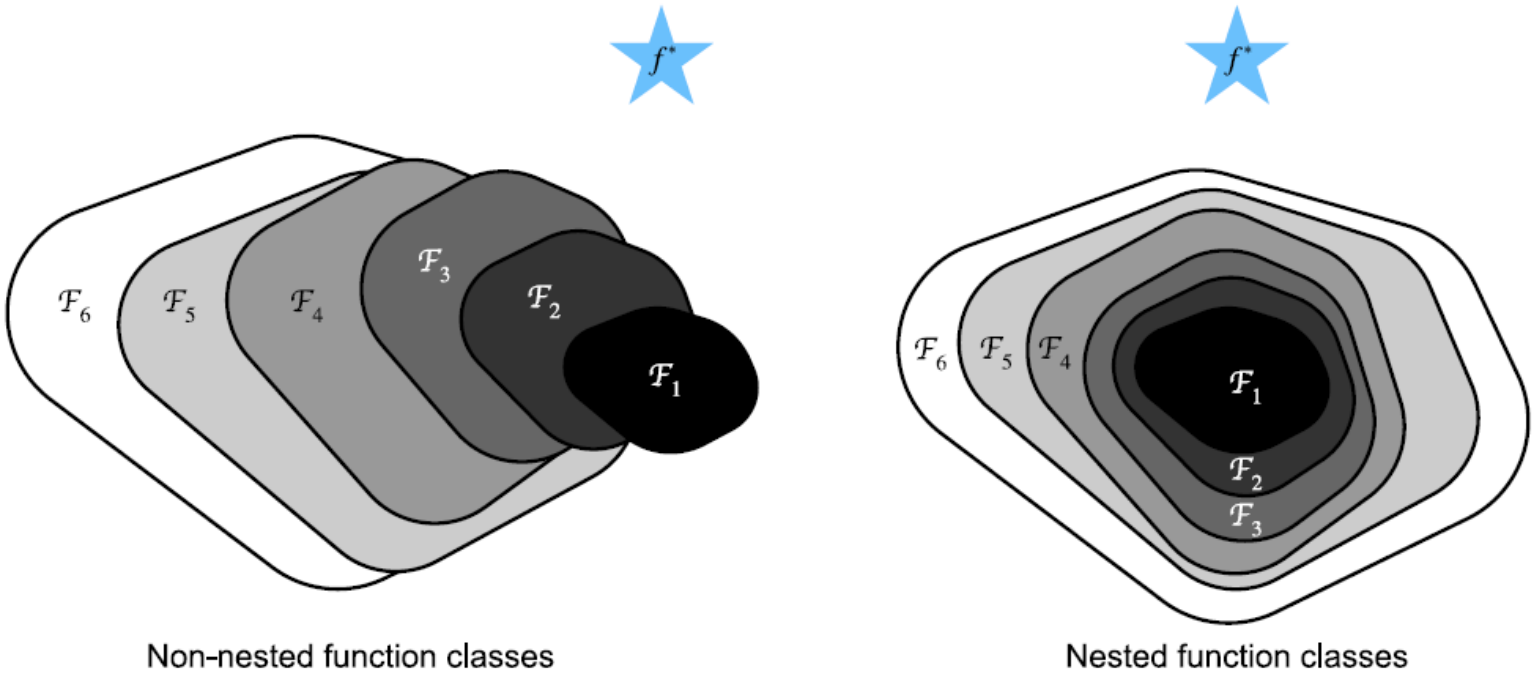
The GoogLeNet architecture.

This GoogLeNet uses a stack of a total of 9 inception blocks, arranged into 3 groups with max-pooling in between, and global average pooling in its head to generate its estimates. Max-pooling between inception blocks reduces the dimensionality. At its stem, the first module is similar to AlexNet and LeNet.

Residual Networks (ResNet)

- As we design increasingly deeper networks it becomes imperative to understand how adding layers can increase the complexity and expressiveness of the network.
- Even more important is the ability to design networks where adding layers makes networks strictly more expressive rather than just different.

Function Classes



Non-nested function classes

Nested function classes

For non-nested function classes, a larger (indicated by area) function class does not guarantee to get closer to the truth function (f*). This does not happen in nested function classes.

Function Classes

- Thus, only if larger function classes contain the smaller ones are we guaranteed that increasing them strictly increases the expressive power of the network.
- For deep neural networks, if we can train the newly-added layer into an identity function $f(x) = x$, the new model will be as effective as the original model. As the new model may get a better solution to fit the training dataset, the added layer might make it easier to reduce training errors.

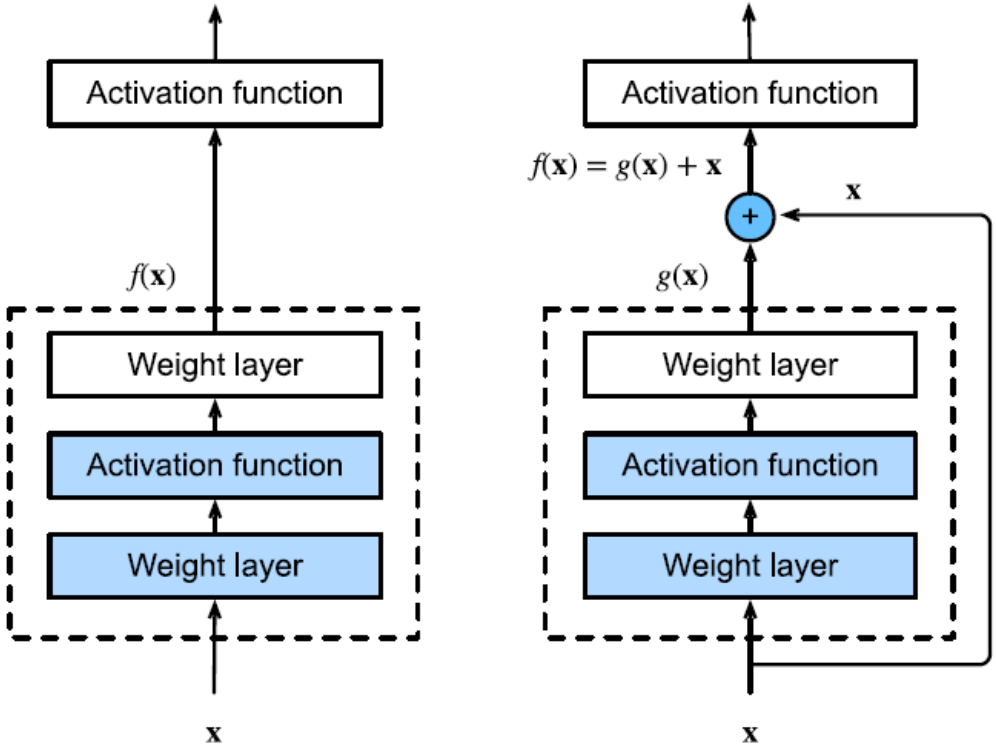
Residual Blocks

- This is the question that (He et al., 2016) considered when working on very deep computer vision models.
- At the heart of their proposed residual network (ResNet) is the idea that every additional layer should more easily contain the identity function as one of its elements. These considerations are rather profound but they led to a surprisingly simple solution, a residual block.
- With it, ResNet won the ImageNet Large Scale Visual Recognition Challenge in 2015.

Residual Blocks

- The design had a profound influence on how to build deep neural networks. For instance, residual blocks have been added to recurrent networks (Kim et al., 2017, Prakash et al., 2016). Likewise, Transformers (Vaswani et al., 2017) use them to stack many layers of networks efficiently. It is also used in graph neural networks (Kipf and Welling, 2016) and, as a basic concept, it has been used extensively in computer vision (Redmon and Farhadi, 2018, Ren et 294 Modern Convolutional Neural Networks al., 2015).

Residual Blocks

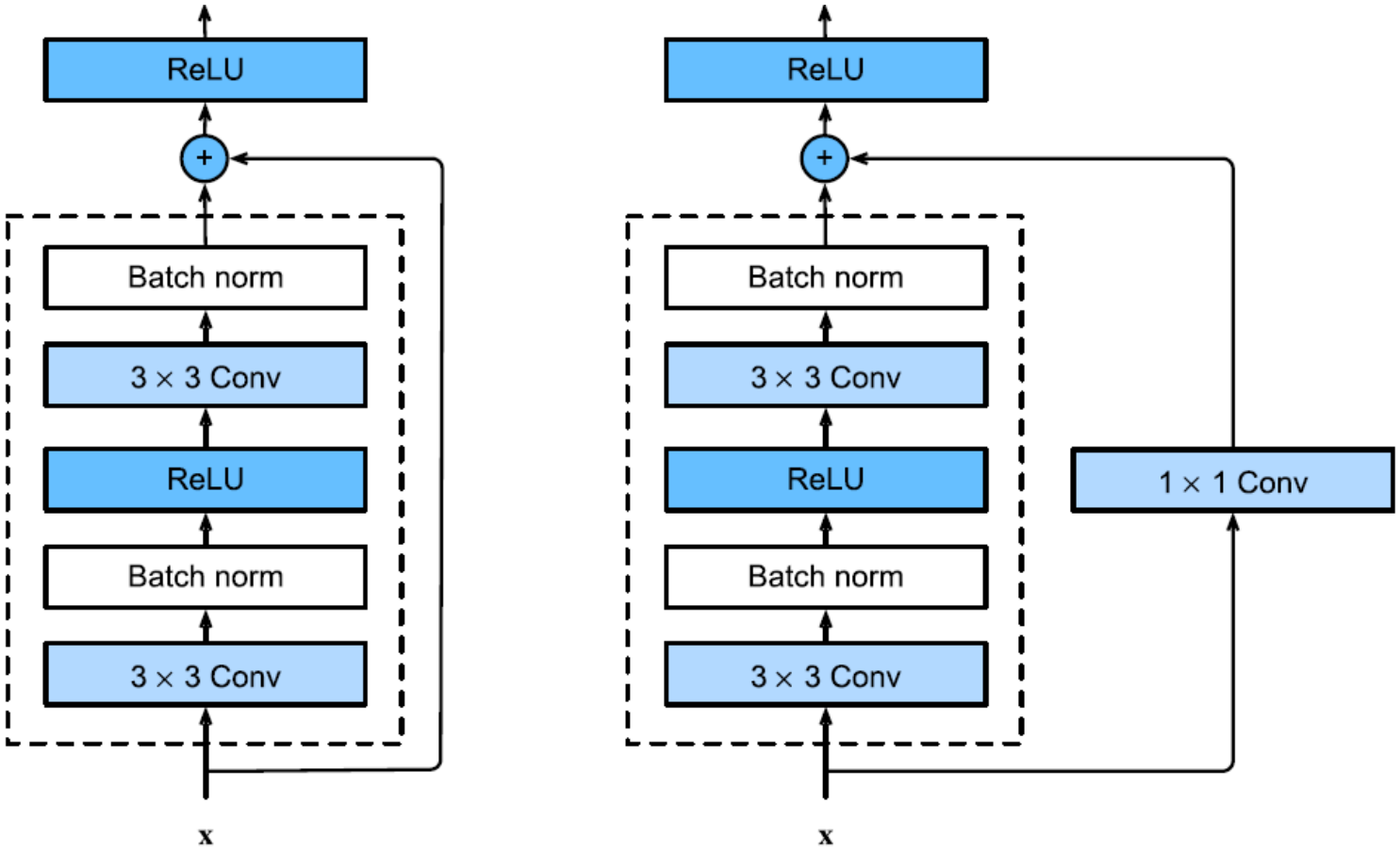


In a regular block (left), the portion within the dotted-line box must directly learn the mapping $\mathrm{f}(\mathbf{x})$. In a residual block (right), the portion within the dotted-line box needs to learn the residual mapping $\mathrm{g}(\mathbf{x}) = \mathrm{f}(\mathbf{x}) - \mathbf{x}$, making the identity mapping $\mathrm{f}(\mathbf{x}) = \mathbf{x}$ easier to learn.

Residual Blocks

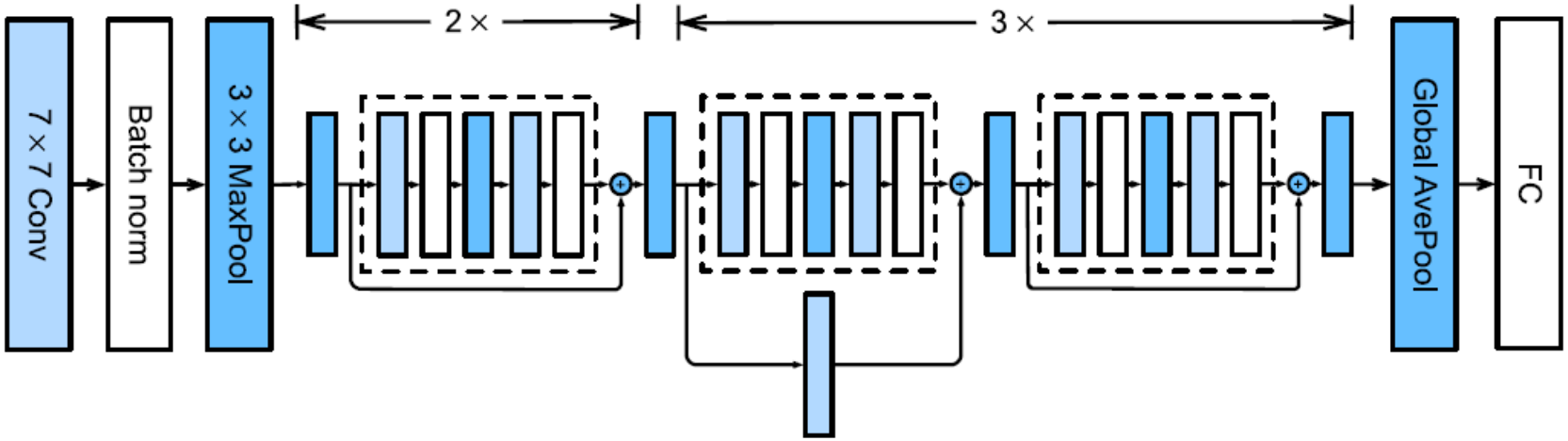
- ResNet follows VGG's full 3×3 convolutional layer design. The residual block has two 3×3 convolutional layers with the same number of output channels. Each convolutional layer is followed by a batch normalization layer and a ReLU activation function.
- Then, we skip these two convolution operations and add the input directly before the final ReLU activation function. This kind of design requires that the output of the two convolutional layers has to be of the same shape as the input, so that they can be added together. If we want to change the number of channels, we need to introduce an additional 1×1 convolutional layer to transform the input into the desired shape for the addition operation.

Residual Blocks



ResNet block with and without 1×1 convolution, which transforms the input into the desired shape for the addition operation.

ResNet Model



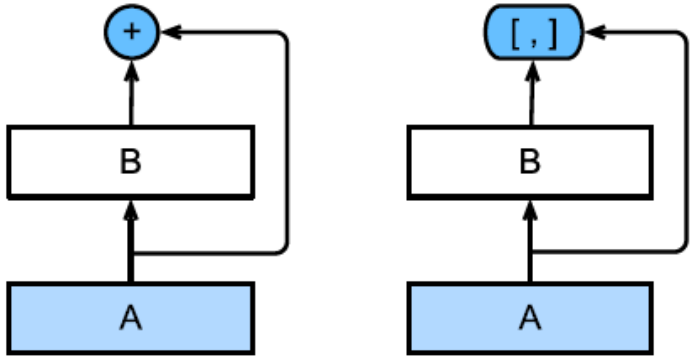
The ResNet-18 architecture.

The first two layers of ResNet are the same as those of the GoogLeNet we described before: the 7×7 convolutional layer with 64 output channels and a stride of 2 is followed by the 3×3 max-pooling layer with a stride of 2. The difference is the batch normalization layer added after each convolutional layer in ResNet.

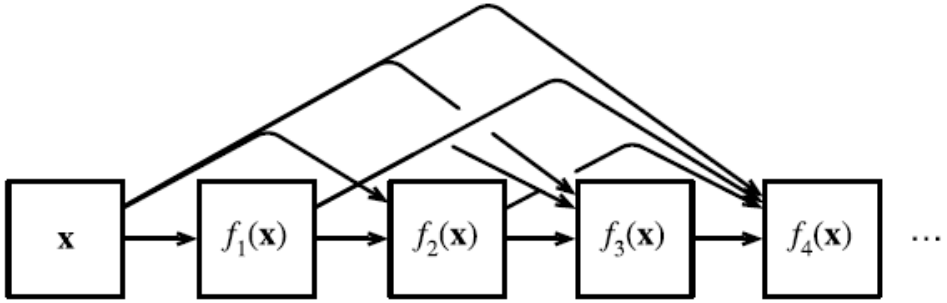
Densely Connected Networks (DenseNet)

- ResNet significantly changed the view of how to parametrize the functions in deep networks.
- DenseNet (dense convolutional network) is to some extent the logical extension of this (Huang et al., 2017).
- DenseNet is characterized by both the connectivity pattern where each layer connects to all the preceding layers and the concatenation operation (rather than the addition operator in ResNet) to preserve and reuse features from earlier layers.

From ResNet to DenseNet



The main difference between ResNet (left) and DenseNet (right) in cross-layer connections: use of addition and use of concatenation.



Dense connections in DenseNet. Note how the dimensionality increases with depth.

Densely Connected Networks (DenseNet)

- The name DenseNet arises from the fact that the dependency graph between variables becomes quite dense. The last layer of such a chain is densely connected to all previous layers.
- The main components that compose a DenseNet are dense blocks and transition layers. The former define how the inputs and outputs are concatenated, while the latter control the number of channels so that it is not too large

Transition Layers

- Since each dense block will increase the number of channels, adding too many of them will lead to an excessively complex model.
- A transition layer is used to control the complexity of the model. It reduces the number of channels by using a 1×1 convolution. Moreover, it halves the height and width via average pooling with a stride of 2.