STOR566: Introduction to Deep Learning

Lecture 8: Convolutional Neural Networks

Yao Li
UNC Chapel Hill

Sep 17, 2024

Materials are from *Learning from data (Caltech)* and *Deep Learning (UCLA)*

Output Feature Map Shape

- The shape of a output feature map depends on: shape of the input feature map, kernel size, stride, padding, etc.

Shape:

- Input: $(N, C_{in}, H_{in}, W_{in})$ or (C_{in}, H_{in}, W_{in})
- Output: $(N, C_{out}, H_{out}, W_{out})$ or $(C_{out}, H_{out}, W_{out})$, where

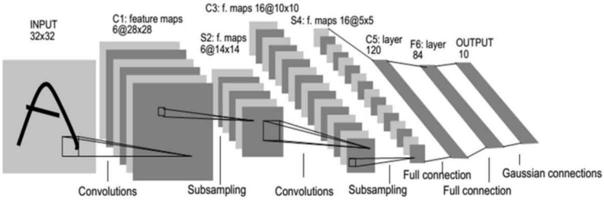
$$H_{out} = \left\lfloor \frac{H_{in} + 2 \times \text{padding}[0] - \text{dilation}[0] \times (\text{kernel\_size}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2 \times \text{padding}[1] - \text{dilation}[1] \times (\text{kernel\_size}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor$$

(The formula is taken from pytorch conv2d document)

Classical CNNs

LeNet5

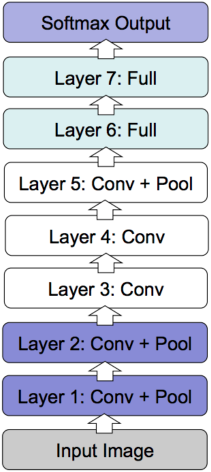


- Input: 32×32 images (MNIST)
- Convolution 1: 6 5×5 filters, stride 1
  - Output: 6 28×28 maps
- Pooling 1: 2×2 max pooling, stride 2
  - Output: 6 14×14 maps
- Convolution 2: 16 5×5 filters, stride 1
  - Output: 16 10×10 maps
- Pooling 2: 2×2 max pooling with stride 2
  - Output: 16 5×5 maps (total 400 values)
- 3 fully connected layers: $120 \Rightarrow 84 \Rightarrow 10$ neurons

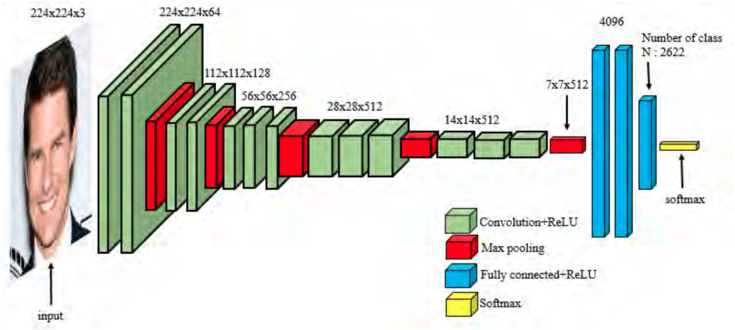
AlexNet

- 8 layers in total, about 60 million parameters and 650,000 neurons.
- Trained on ImageNet dataset

  "ImageNet Classification with Deep Convolutional Neural Networks", by Krizhevsky, Sustskever and Hinton, NIPS 2012.



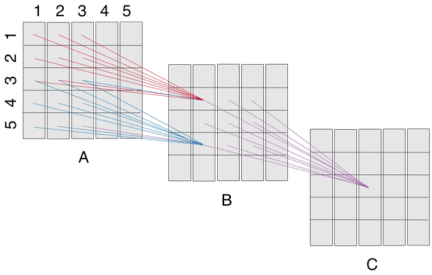
VGG Network



224x224x3
224x224x64
112x112x128
56x56x256
28x28x512
14x14x512
7x7x512
4096
Number of class
N : 2622
softmax
input
Convolution+ReLU
Max pooling
Fully connected+ReLU
Softmax

What do the kernels learn?

- The receptive field of a neuron is the input region that can affect the neuron's output
- The receptive field for a first layer neuron is its neighbors (depending on kernel size) $\Rightarrow$ capturing very local patterns
- For higher layer neurons, the receptive field can be much larger $\Rightarrow$ capturing global patterns



Data Augmentation

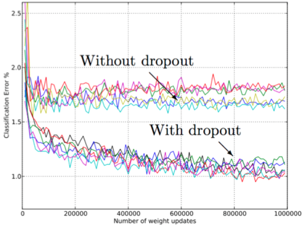
- Increase the size of data by
  - Rotation: random angle between $-\pi$ and π
  - Shift: 4 directions
  - Rescaling: random scaling up/down
  - Flipping
  - Gaussian noise
  - Many others
- Can be combined perfectly with SGD (augmentation when forming each batch)

Dropout: Regularization for neural network training

- One of the most effective regularization for deep neural networks!

Method	CIFAR-10	CIFAR-100
Conv Net + max pooling (hand tuned)	15.60	43.48
Conv Net + stochastic pooling (Zeiler and Fergus, 2013)	15.13	42.51
Conv Net + max pooling (Snoek et al., 2012)	14.98	-
Conv Net + max pooling + dropout fully connected layers	14.32	41.26
Conv Net + max pooling + dropout in all layers	12.61	**37.20**
Conv Net + maxout (Goodfellow et al., 2013)	**11.68**	38.57

Table 4: Error rates on CIFAR-10 and CIFAR-100.

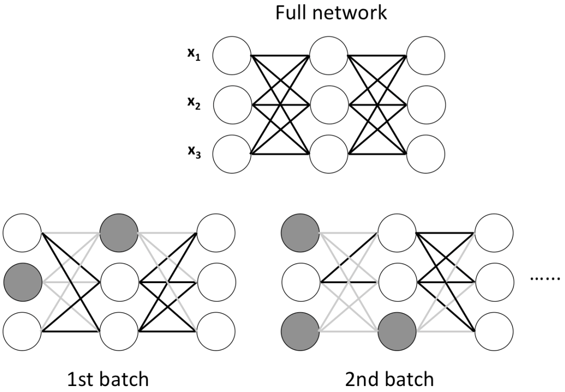


Srivastava et al, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, 2014.

Dropout (training)

Dropout in the **training** phase:

- For each batch, turn off each neuron (including inputs) with a probability $1 - \alpha$
- Zero out the removed nodes/edges and do backpropagation.



Dropout (test time)

- Training: Each neuron computes

$$x_i^{(l)} = B\sigma(\sum_j W_{ij}^{(l)} x_j^{(l-1)} + b_i^{(l)})$$

  where B is a Bernoulli variable that takes 1 with probability α
- The expected output of the neuron:

$$E[x_i^{(l)}] = \alpha\sigma(\sum_j W_{ij}^{l} x_j^{l-1} + b_i^{l})$$

- Use the expected output at test time

  $\Rightarrow$ multiply all the weights by α

Explanations of dropout

- For a network with n neurons, there are 2^n possible sub-networks
- Dropout: randomly sample over all 2^n possibilities
- Can be viewed as a way to learn Ensemble of 2^n models

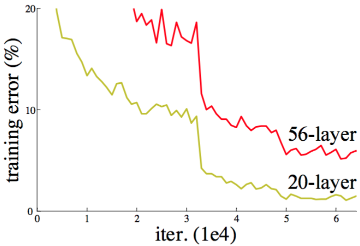
Revisit Alexnet

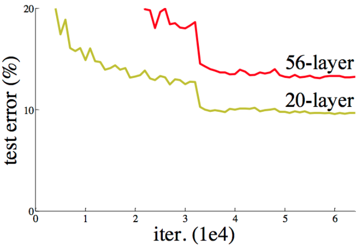
- Dropout: 0.5 (in FC layers)
- A lot of data augmentation
- Momentum SGD with batch size 128, momentum factor 0.9
- L2 weight decay (L2 regularization)
- Learning rate: 0.01, decreased by 10 every time when reaching a stable validation accuracy

Residual Networks

- Very deep convnets do not train well

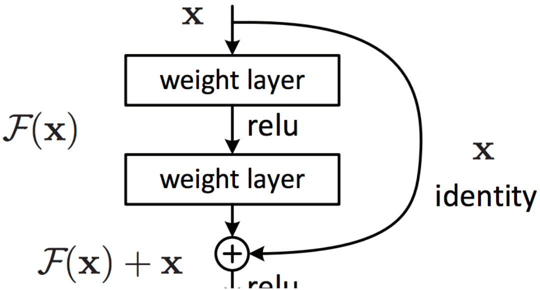
  vanishing gradient problem





Residual Networks

- Key idea: introduce "pass through" into each layer

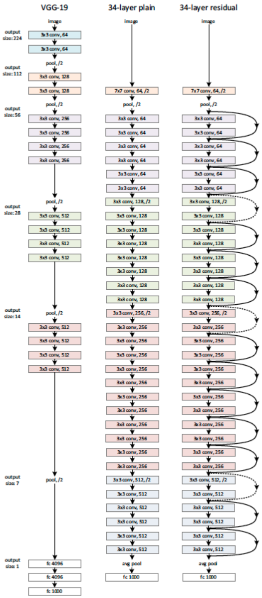


- Thus, only residual needs to be learned

Residual Networks

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).



Conclusions

- Classical Architectures
- Training Techniques

Questions?