Introduction to Machine Learning

Convolutional Neural Networks

Ashley Gao

William & Mary

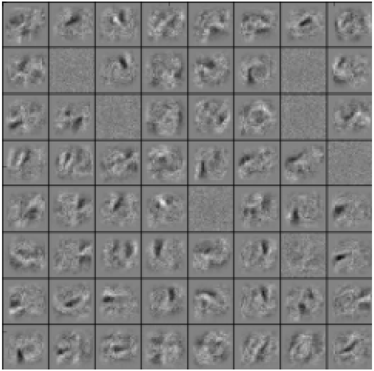
November 13, 2023

Overview

- What makes vision hard?
  - Vision needs to be robust to a lot of transformations or distortions
    - Change in pose or viewpoint
    - Change in illumination
    - Deformation
    - Occlusion (some objects are hidden behind others)
  - Many object categories can vary wildly in appearance (e.g. chairs)
  - Geoff Hinton: “Imaging a medical database in which the age of the patient sometimes hops to the input dimension which normally codes for weight!”

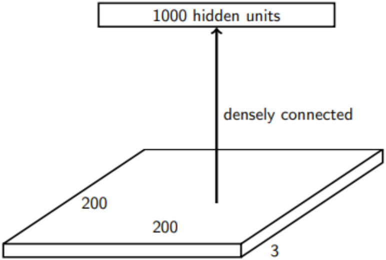
Overview

- Recall we looked at some hidden layer features for classifying handwritten digits:



Overview

- Suppose we want to train a network that takes a 200×200 RGB image as the input.



- What is the problem with having this as the first layer?
  - Too many parameters! Input size = $200 \times 200 \times 3 = 120$K. Parameters = $120\text{K} \times 1000 = 120$ million.
  - What happens if the object in the image shifts a little?

Overview

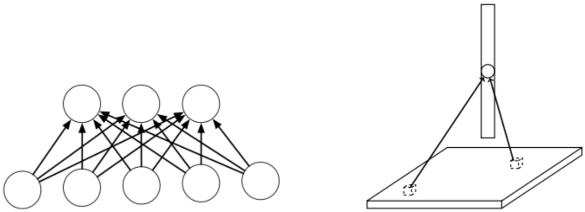
- The same sorts of features that are useful in analyzing one part of the image will probably be useful for analyzing other parts as well
  - E.g. edges, corners, contours, object parts
- We want a neural net architecture that lets us learn a set of feature detectors that are applied at all image locations

Overview

- So far, we have seen this type of layers:
  - Fully connected layers
- Different layers could be stacked together to build powerful models
- Let's add another layer type: the convolution layer

Fully Connected Layers

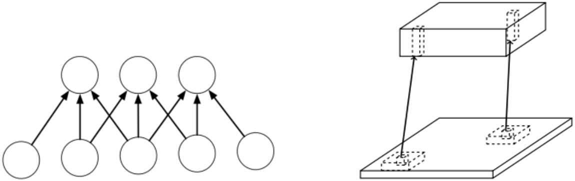
- Fully connected layers:



- Each hidden unit looks at the entire image

Locally Connected Layers

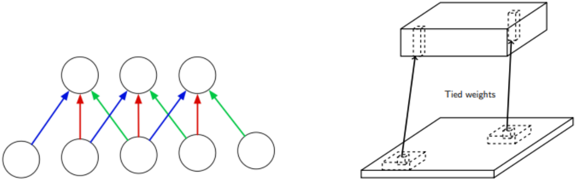
- Locally connected layers:



- Each column or set of hidden units looks at a small region of the image

Convolution Layers

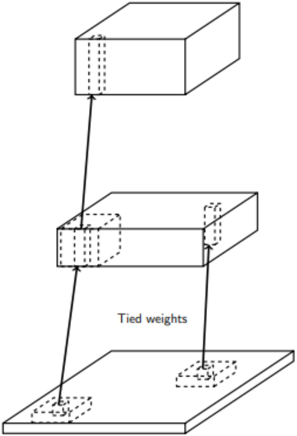
- Convolution layers:



- Each column or set of hidden units looks at a small region of the image, and the weights are shared between all image locations

Going Deeply Convolutional

- Convolution layers can be stacked:

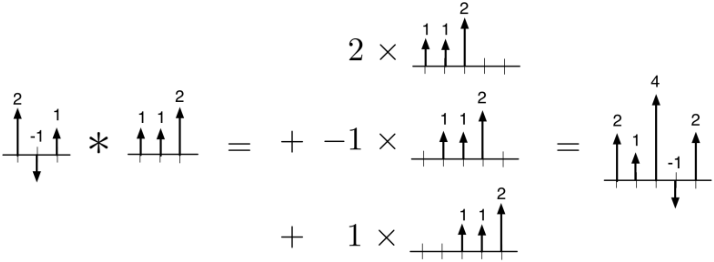


Convolution

- We have been vectorizing our computations by expressing them in terms of matrix and vector operations for computational efficiency
- Now we introduce a high-level operation, convolution.
  - The motivation is not computational efficiency
  - Rather, the motivation is to get some understanding of what convolution layers can do

Convolution

- Let's look at the 1-D case first.



Convolution

- Convolution can also be viewed as matrix multiplication

$$(2, -1, 1) * (1, 1, 2) = \begin{pmatrix} 1 & & \\ 1 & 1 & \\ 2 & 1 & 1 \\ & 2 & 1 \\ & & 2 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}$$

Convolution

- Some properties of convolution
  - Commutativity

$$a * b = b * a \tag{1}$$

  - Linearity

$$a * (\lambda_1 b + \lambda_2 c) = \lambda_1 a * b + \lambda_2 a * c \tag{2}$$

2-D Convolution

- Here is an example of 2-D convolution, which we will use in computer vision

$$
\begin{array}{|c|c|c|}\hline 1 & 3 & 1 \\ \hline 0 & -1 & 1 \\ \hline 2 & 2 & -1 \\ \hline \end{array}
\ast
\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 0 & -1 \\ \hline \end{array}
=
\begin{array}{l}
1 \times \begin{array}{|c|c|c|c|}\hline 1 & 3 & 1 & \\ \hline 0 & -1 & 1 & \\ \hline 2 & 2 & -1 & \\ \hline & & & \\ \hline \end{array} \\
+ \; 2 \times \begin{array}{|c|c|c|c|}\hline & 1 & 3 & 1 \\ \hline & 0 & -1 & 1 \\ \hline & 2 & 2 & -1 \\ \hline & & & \\ \hline \end{array} \\
+ -1 \times \begin{array}{|c|c|c|c|}\hline & & & \\ \hline & 1 & 3 & 1 \\ \hline & 0 & -1 & 1 \\ \hline & 2 & 2 & -1 \\ \hline \end{array}
\end{array}
=
\begin{array}{|c|c|c|c|}\hline 1 & 5 & 7 & 2 \\ \hline 0 & -2 & -4 & 1 \\ \hline 2 & 6 & 4 & -3 \\ \hline 0 & -2 & -2 & 1 \\ \hline \end{array}
$$

2-D Convolution

- The thing we convolve by is called a kernel, or filter
- What does this convolution kernel do?

$*$

0	1	0
1	4	1
0	1	0

2-D Convolution

- What does this convolution kernel do?

$*$

0	-1	0
-1	8	-1
0	-1	0

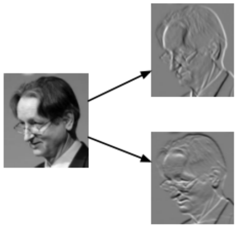
2-D Convolution

- What does this convolution kernel do?

$$* \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

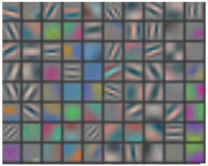
Convolutional Networks

- Let's finally turn to convolutional networks.
  - Detection layers (or convolutional layers)
  - Pooling layers
- The convolution layer has a set of filters.
  - Its output is a set of feature maps, each one obtained by convolving the image with a filter.



convolution

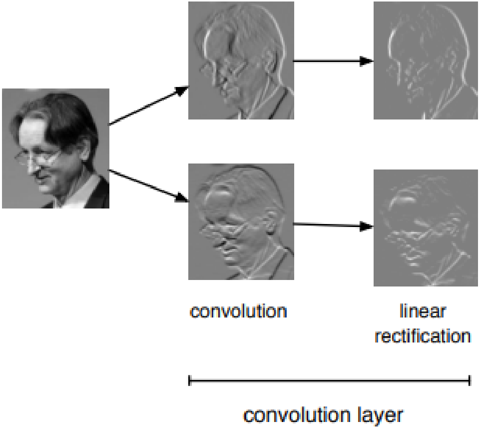
Example first-layer filters



(Zeiler and Fergus, 2013, Visualizing and understanding convolutional networks)

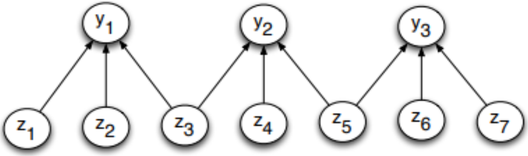
Convolutional Networks

- It's common to apply a linear rectification nonlinearity: $y_i = \max(z_i, 0)$
  -
  - Two edges in opposite directions shouldn't cancel



Pooling Layers

- The other type of layer is the pooling layer.
  - These layers reduce the size of the representation and build invariance to small transformations

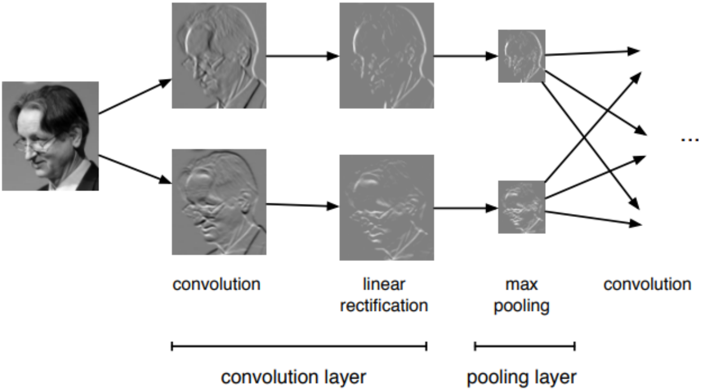


- Most commonly, we use max-pooling
  - Which computes the maximum value of the units in a pooling group

$$y_i = \max_{j \in J} z_j \tag{3}$$

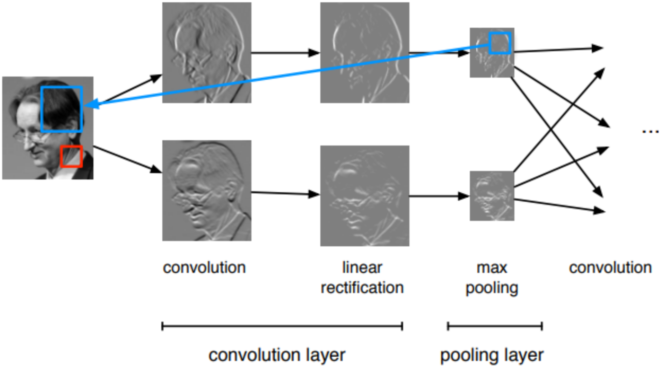
Convolutional Networks

- A more holistic view on the CNN(s):



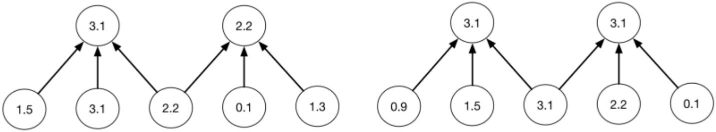
Convolutional Networks

- Because of pooling, higher-layer filters can cover a larger region of the input than equal-sized filters in the lower layers



Convolutional Networks

- We said the network's responses should be robust to the translation of the input. But this can mean two different things
  - Convolution layers are equivariant: it you translate the inputs, the outputs are translated by the same amount.
  - We'd like the network to be invariant: if you translate the inputs, the prediction should not change
  - Pooling layers provide invariance to small translation



Convolution Layers

- Each layer consists of several feature maps, each of which is an array
    - For the input layer, the feature maps are usually called channels
    - If the input layer represents a grayscale image, it has 1 channel.
- Each unit is connected to each unit within its receptive field in the previous layers.
    - This includes all of t he previous layer's feature maps.

Image Classification

Overview

- Object recognition is the task of identifying which object category is present in an image
- It is challenging because objects can differ widely in position, size, shape, appearance, etc., and we have to deal with occlusions, lighting changes, etc.
- Why we care about it
  - Direct application to image search
  - Closely related to object detection, the task of locating all instances of an object in an image
    - e.g. a self-driving car detecting pedestrians or stop signs
- For the past 5-10 years, all the best object recognizers have been various kinds of conv nets and an architecture we called the Transformer
  - We will cover Transformers later in this class

Recognition Datasets

- In order to train and evaluate a machine learning system, we need to collect a dataset
  - The design of the dataset can have major implications
- Some questions to consider:
  - Which classes to include?
  - Where should the image come from?
  - How many images to collect?
  - How to normalize (preprocess) the images?

Image Classification

- Conv nets are just one of many possible approaches to image classification
  - However, they have been successful and now become a classical approach in the deep learning era
- Biggest image classification "advances" of the lasst two decades
  - Datasets have gotten much larger (because of digital cameras and the Internet)
  - Computers got much faster
    - Graph processing units (GPUs) turned out to be really good at training big neural nets
    - Generally, about 30 times faster than CPUs
  - As a result, we could fit bigger and bigger neural nets

MNIST Dataset

- MNIST dataset of handwritten digits
  - Categories: 10 digit classes
  - Source: Scans of handwritten zip codes from envelopes
  - Size: 60,000 training images and 10,000 test images, grayscale, of size 28×28
  - Normalization: centered within the image, scaled to a consistent size
    - The assumption is that the digit recognizer would be part of a larger pipeline that segments and normalizes images
- In 1998, Yann LeCun and colleagues built a conv net called LeNet which was able to classify digits with 98.9% test accuracy
  - It was good enough to be used in a system for automatically reading numbers on checks

Calltech101

- Caltech101 was the first major object recognition dataset, collected in 2003
- Design decisions:
  - Categories: 101 object categories; open the dictionary to random pages and select from nouns that were associated with images
  - Source: find candidates with Google Image Search, hand-select the ones that actually represent the object category
  - Number of examples: as many as possible per category
    - Most machine learning benchmarking is done using a fixed number of training examples per category (usually between 1 and 20)
  - Normalization:
    - Scale to be 300 pixels wide
    - Flip so that the object is facing the same direction
    - Rotate certain object categories because their proposed algorithm couldn't handle vertical objects

Input Vectors







Calltech101

- Beware of dataset biases
  - These are idiosyncrasies of a dataset resulting from how it was collected or normalized
- An algorithm can appear to have good training and testing error, but fail to generalize if the training data doesn't resemble the real world
- E.g. in Caltech101, the sizes and locations of the images are a lot m ore regular than you would expect in the "real world"

Calltech101

- There were lots of works on Caltech101 for 5 years or so
  - It quickly became clear that dataset biases made it too gameable
- By contrast, MNIST is still a productive source of insights after 20 years of its introduction!

ImageNet

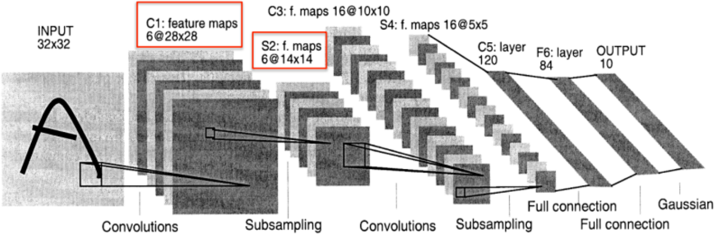
- ImageNet is the modern object recognition benchmark dataset
  - It was introduced in 2009 and has led to amazing progress in object recognition since then

ImageNet

- Used for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). an annual benchmark competition for object recognition algorithms
- Design decisions:
  - Categories: Taken from a lexical database called WordNet
    - WordNet consists of "synsets" or sets of synonymous words
    - They tried to use as many of these as possible; almost 22,000 as of 2010
    - Of these, they chose the 1000 most common for the ILSVRC
    - The categories are really specific, e.g. hundreds of kinds of dogs
  - Size: 1.2 million full-sized images for the ILSVRC
  - Source: Results from image search engines, hand-labeled
    - Labeling such specific categories was challenging; annotators had to be given the WordNet hierarchy, Wikipedia, etc.
  - Normalization: none, although the contestants are free to do preprocessing

LeNet

- Here's the LeNet architecture, which was applied to handwritten digit recognition on MNIST in 1998
  - In the image, subsampling equates to pooling



Size of CNNs

	LeNet (1989)	**LeNet (1998)**	**AlexNet (2012)**
classification task	digits	digits	objects
categories	10	10	1,000
image size	16×16	28×28	$256 \times 256 \times 3$
training examples	7,291	60,000	1.2 million
units	1,256	8,084	658,000
parameters	9,760	60,000	60 million
connections	65,000	344,000	652 million
total operations	11 billion	412 billion	200 quadrillion (est.)

AlexNet

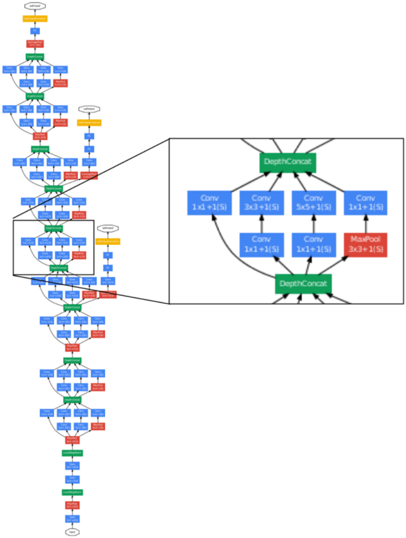
- AlexNet, 2012. 9 Weight layers. 16.4% top-5 error (i.e. the network gets 5 tries to guess the right category)

	LeNet (1989)	**LeNet (1998)**	**AlexNet (2012)**
classification task	digits	digits	objects
categories	10	10	1,000
image size	16×16	28×28	$256 \times 256 \times 3$
training examples	7,291	60,000	1.2 million
units	1,256	8,084	658,000
parameters	9,760	60,000	60 million
connections	65,000	344,000	652 million
total operations	11 billion	412 billion	200 quadrillion (est.)

- They used lots of tricks to train NN
  - ReLU units, weight decay, data augmentation, SGD, dropout
- AlexNet's stunning performance on the ILSVRC is what set off the deep learning boom of the last 5-10 years

GoogLeNet

- GoogLeNet, 2014
- 22 weight layers
- Fully convolutional (no fully connected layers)
- Convolutions are broken down into a bunch of smaller convolutions
- 6.6% test error on ImageNet



Classification

- ImageNet results over the years. Note that errors are top-5 errors

Year	Model	Top-5 error
2010	Hand-designed descriptors + SVM	28.2%
2011	Compressed Fisher Vectors + SVM	25.8%
2012	AlexNet	16.4%
2013	a variant of AlexNet	11.7%
2014	GoogLeNet	6.6%
2015	deep residual nets	4.5%

- Human performance is around 5.1%
- They stopped running the object recognition competition because the performance is already so good