
The Ancient
Secrets
Computer Vision

Logistics:

- Homework 4 is out!
  - Due tonight BUT everyone has 3 more late days
- Final Project Proposals
  - 1-2 paragraphs
  - Due next Tuesday
  - Very open ended, whatever you find interesting
  - Should be about 2 homeworks
  - Default project: train a classifier on a dataset, compete on Kaggle, will post link soon


Previously
On
the
Ancient Secrets
of Computer Vision

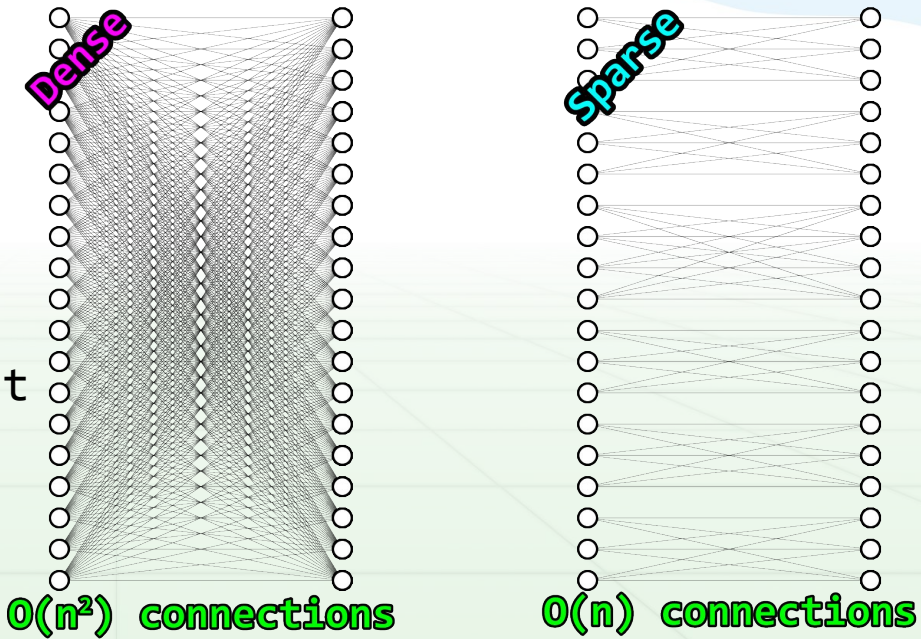
Too many weights!

Would rather have sparse connections

- Fewer weights
- Nearby regions - related
- Far apart - not related

Convolutions!

- Just weighted sums of small areas in image
- Weight sharing in different locations in image



Convolutional Layer

Input: an image

Processing: convolution with multiple filters

Output: an image, # channels = # filters

Output still weighted sum of input (w/ activation)

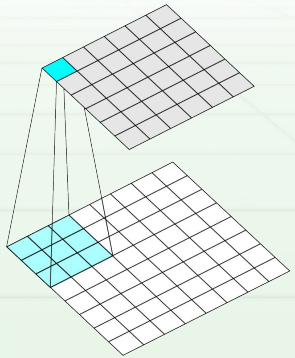
Kernel size

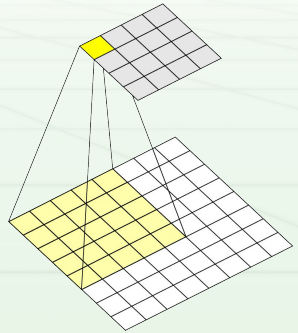
How big the filter for a layer is

Typically 1 x 1 <-> 11 x 11

1 x 1 is just linear combination of channels in previous image (no spatial processing)

Filters usually have same number of channels as input image.



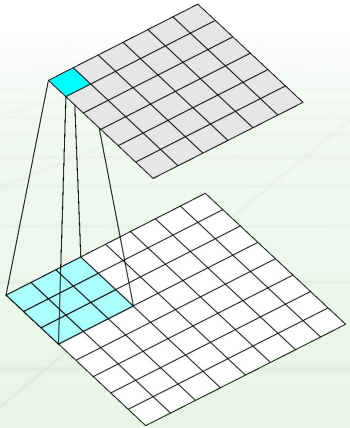


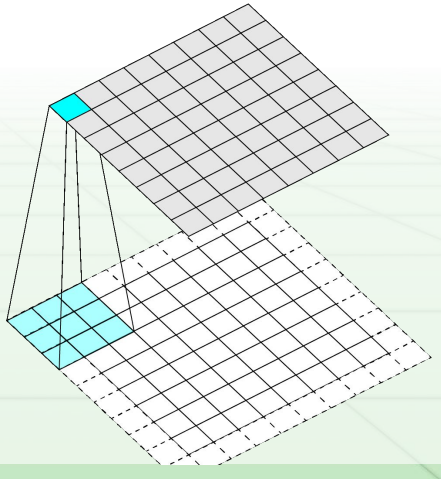
Padding

Convolutions have problems on edges

Do nothing: output a little smaller than input

Pad: add extra pixels on edge



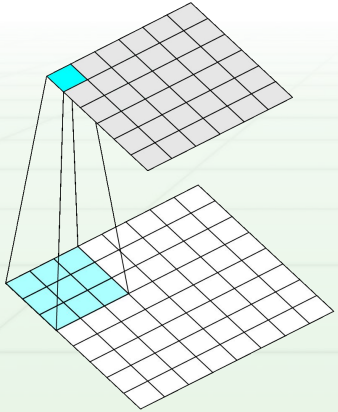


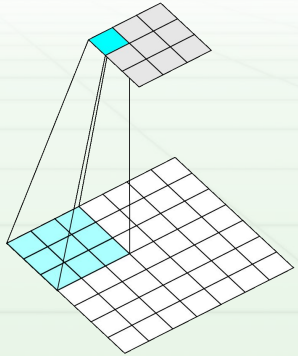
Stride

How far to move filter between applications

We've done stride 1 convolutions up until now, approximately preserves image size

Could move filter further, downsample image





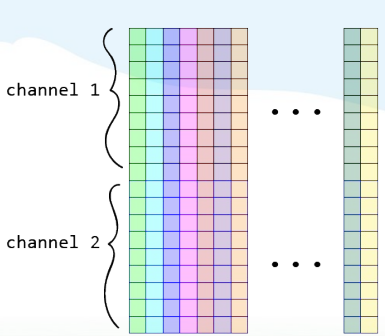
Im2col: rearrange image

Take spatial blocks of image and put them into columns of a matrix.

Im2col handles kernel size, stride, padding, etc.

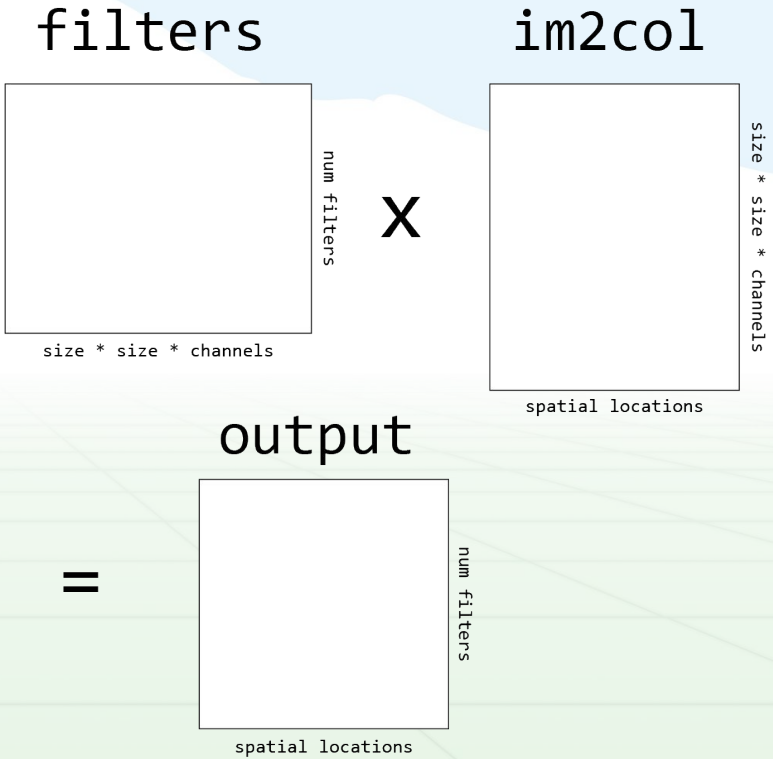
Makes matrix with all relevant pixels in the image.

Multiple channel image stacked by channel.



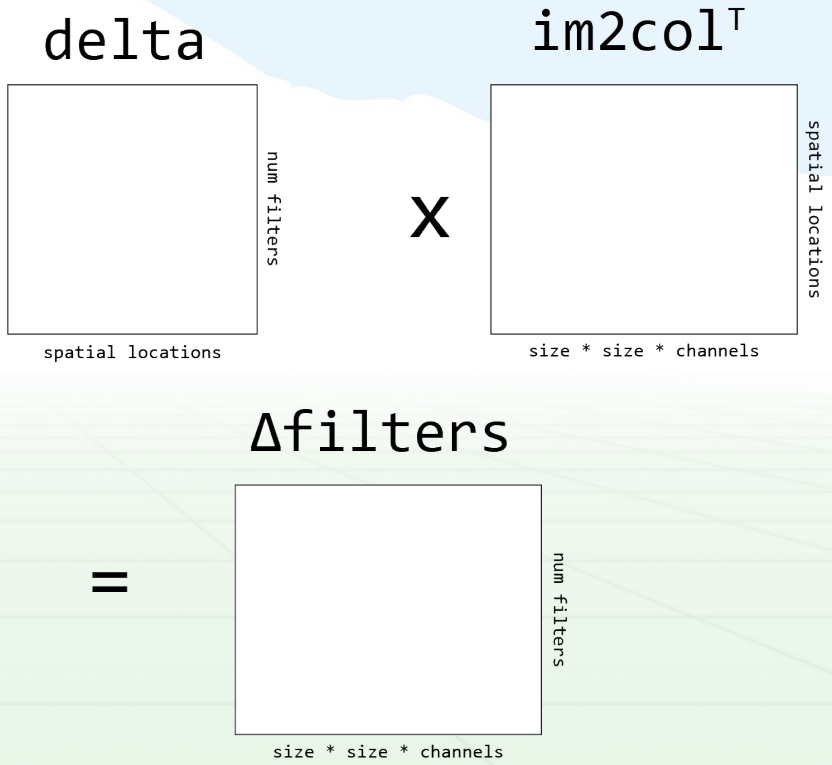
Im2col: rearrange image

Now we just multiply by our filter matrix to do convolutions.



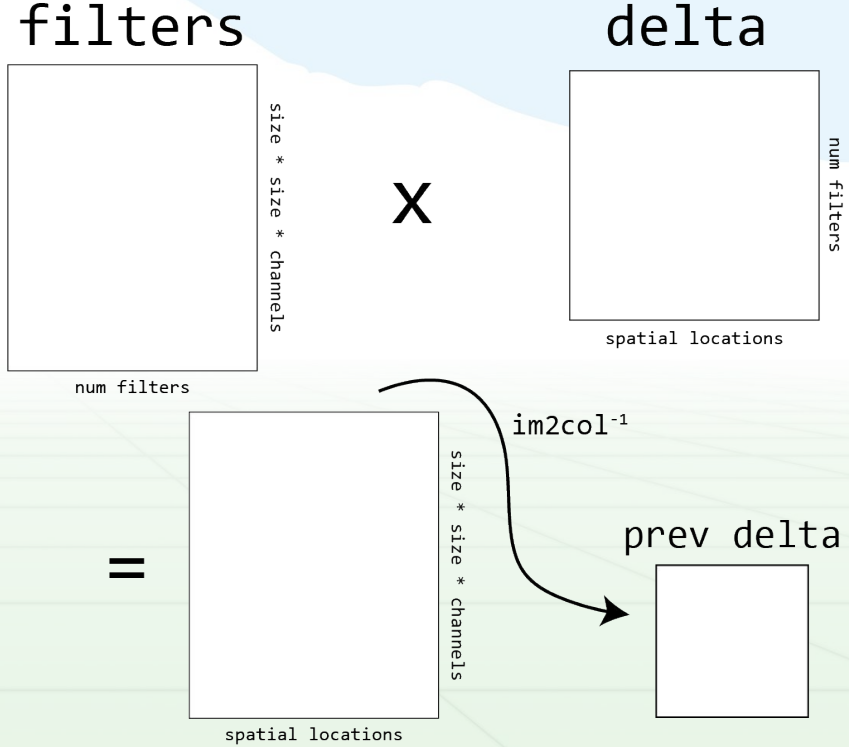
Im2col: rearrange image

Can calculate our weight updates as well by multiplying the delta of a layer by the input.



Im2col: rearrange image

Finally, can calculate backpropagated delta by multiplying the current delta by weights and doing the reverse of im2col.



Pooling Layer

Input: an image

Processing: pool pixel values over region

Output: an image, shrunk by a factor of the stride

Hyperparameters:

- What kind of pooling? Average, mean, max, min
- How big of stride? Controls downsampling
- How big of region? Usually not much bigger than stride

Most common: 2x2 or 3x3 maxpooling, stride of 2

(Fully) Connected Layer

The standard neural network layer where every input neuron connects to every output neuron

Often used to go from image feature map -> final output or map image features to a single vector

Eliminates spatial information

Convnet Building Blocks

Convolutional layers:

- Connections are convolutions
- Used to extract features

Pooling layers:

- Used to downsample feature maps, make processing more efficient
- Most common: maxpool, avgpool sometimes used at end

Connected layers:

- Often used as last layer, to map image features -> prediction
- No spatial information
- Inefficient: lots of weights, no weight sharing


Chapter
Fourteen
Network Architectures

MNIST: Handwriting recognition

50,000 images of handwriting

28 x 28 x 1 (grayscale)

Numbers 0-9

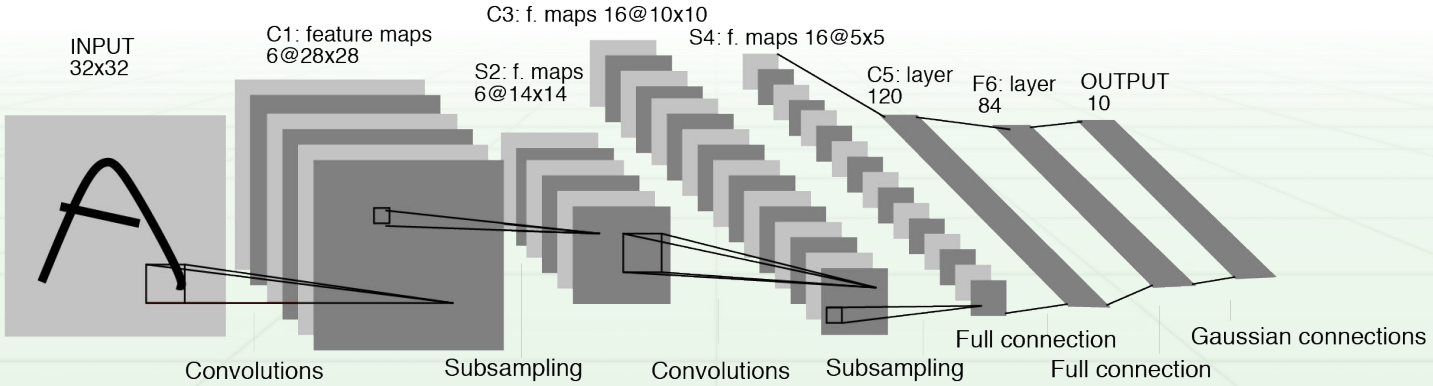
You're working with this in your homework!

LeNet: First Convnet for Images*

99% accuracy on MNIST (Yann LeCun 1998)

Has all elements of modern convnet

- Convolutions, maxpooling, fully connected layers
- Logistic activations after pooling layers (nowadays use RELU)
- Weight updates through backpropagation



*probably? Maybe neocognitron but not trained w/ backprop Fukushima, Kunihiko (1980). "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position" (PDF). Biological Cybernetics. 36 (4): 193–202.

ImageNet: Really big image dataset

Dataset of images and labels from Princeton

14 million images, 22k categories

Challenge subset (most used):

1.2 million images, 1000 categories

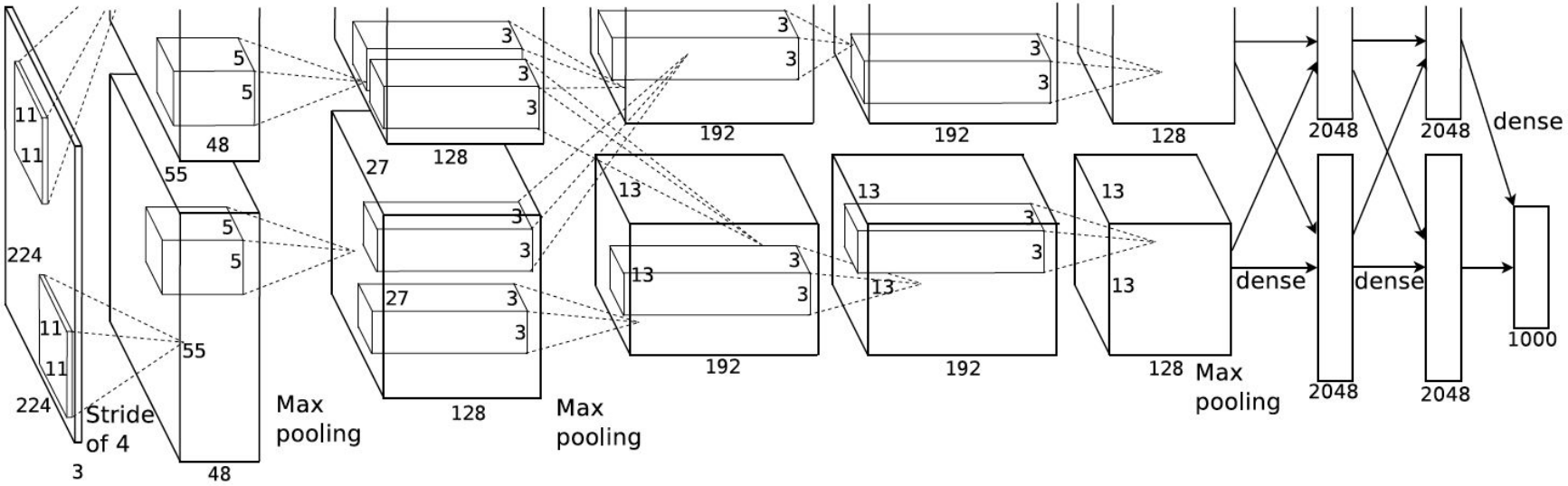
ImageNet: Really big image dataset

Labels taken from WordNet, a lexical database of english words and how they relate

Paid workers on AMT to label images with WordNet label or *synset* (group of words)

AlexNet: first good network

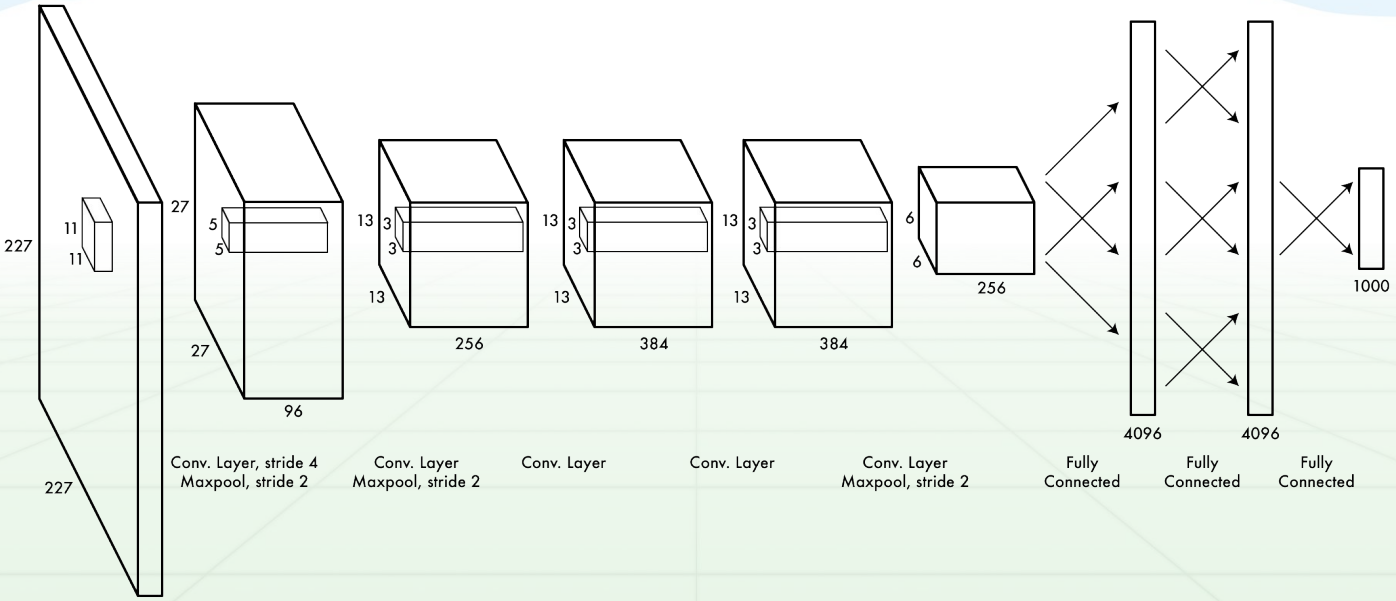
Entry for ImageNet challenge



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet: first good network

Entry for ImageNet challenge



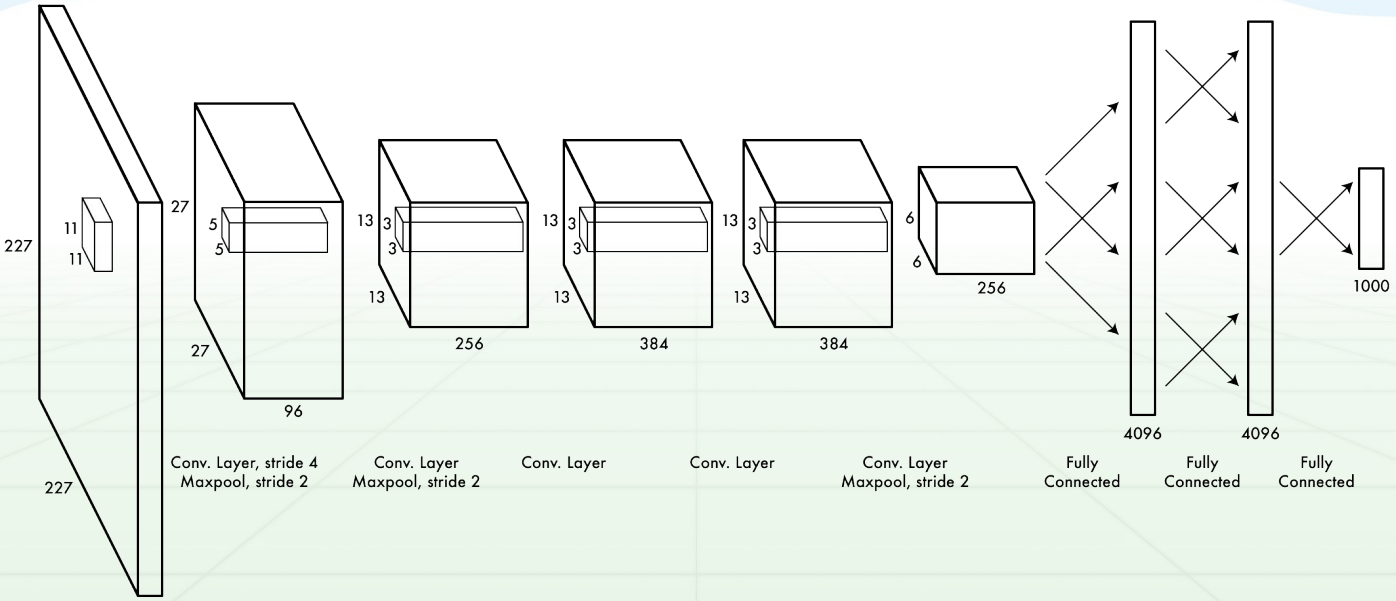
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

AlexNet: first good network

Entry for ImageNet challenge

Much higher accuracy than “traditional” methods (SVM on SIFT)

Why did it take so long?



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

GPUs: How modern CNNs are possible

Alex spent a long time implementing the components of a CNN on a GPU, his software Cudaconvnet was the first “modern” neural network framework

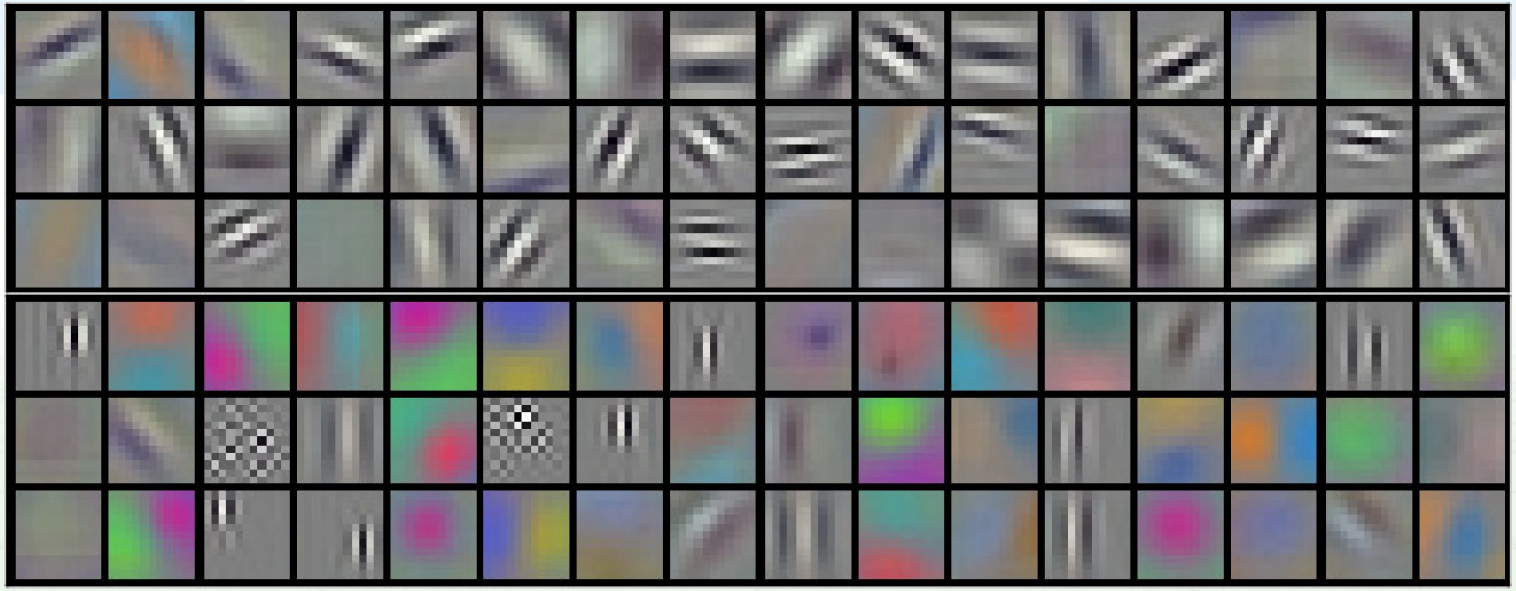
GPUs allow >100x speedup compared to CPU, training still took weeks on ImageNet

This idea, CNNs + GPUs started revolution in computer vision just 5 years ago

(check NVIDIAs stock in last 5 years)

What are these networks learning?

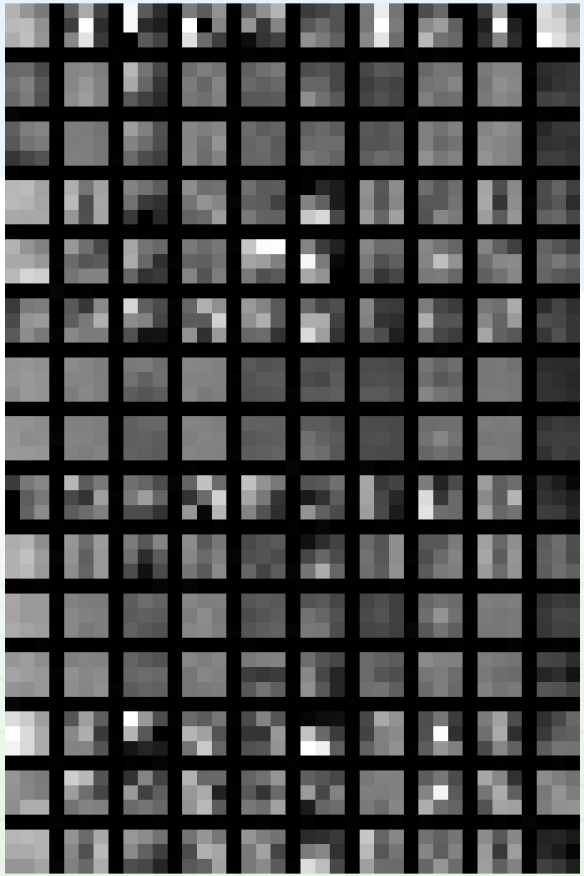
Features! Here’s 1st layer of AlexNet



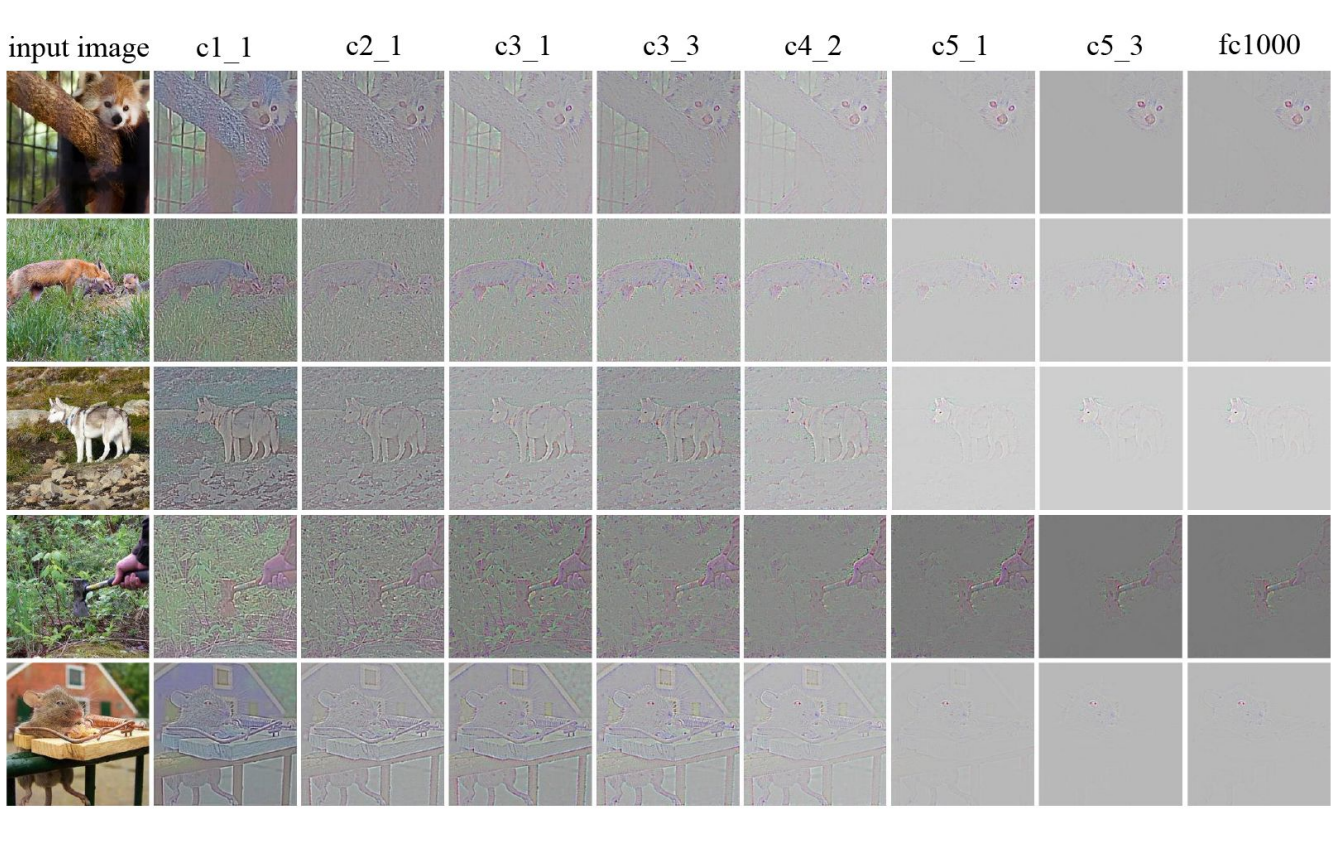
Later layers harder to visualize

Combinations of lower-level features, not much meaning by themselves

We can try other methods to visualize them, information flow



What info goes through the network



https://icmlviz.github.io/icmlviz2016/assets/papers/4.pdf

Idea: find interesting images

We can feed images into our network and see which ones activate certain neurons

(a) Unit sensitive to white flowers.

(c) Unit senstive to round, spiky flowers.

(d) Unit senstive to round green or yellow objects.

https://arxiv.org/pdf/1312.6199.pdf

Idea: *make* interesting images

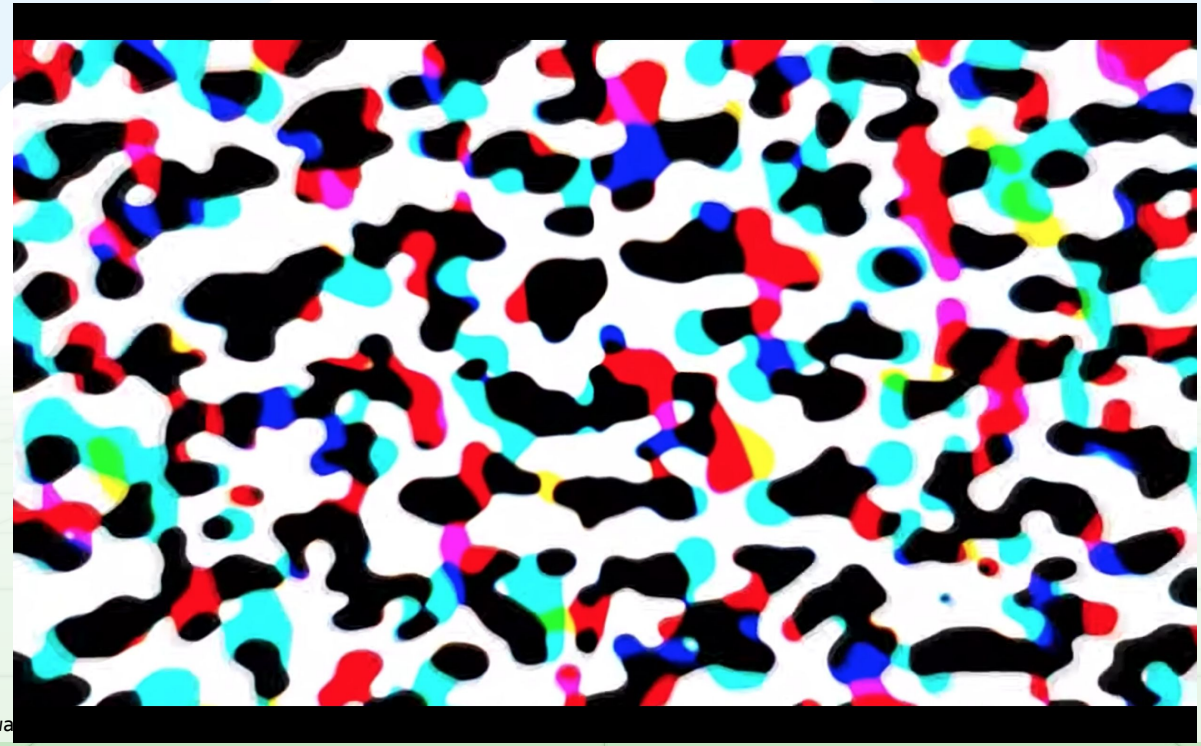
Instead of optimizing network, we can do gradient descent on the image too

Optimize the image to activate certain neurons or certain layers, then we can learn what those layers or neurons do!

You may have seen this under the name *deep dream*

Early layers: blobs

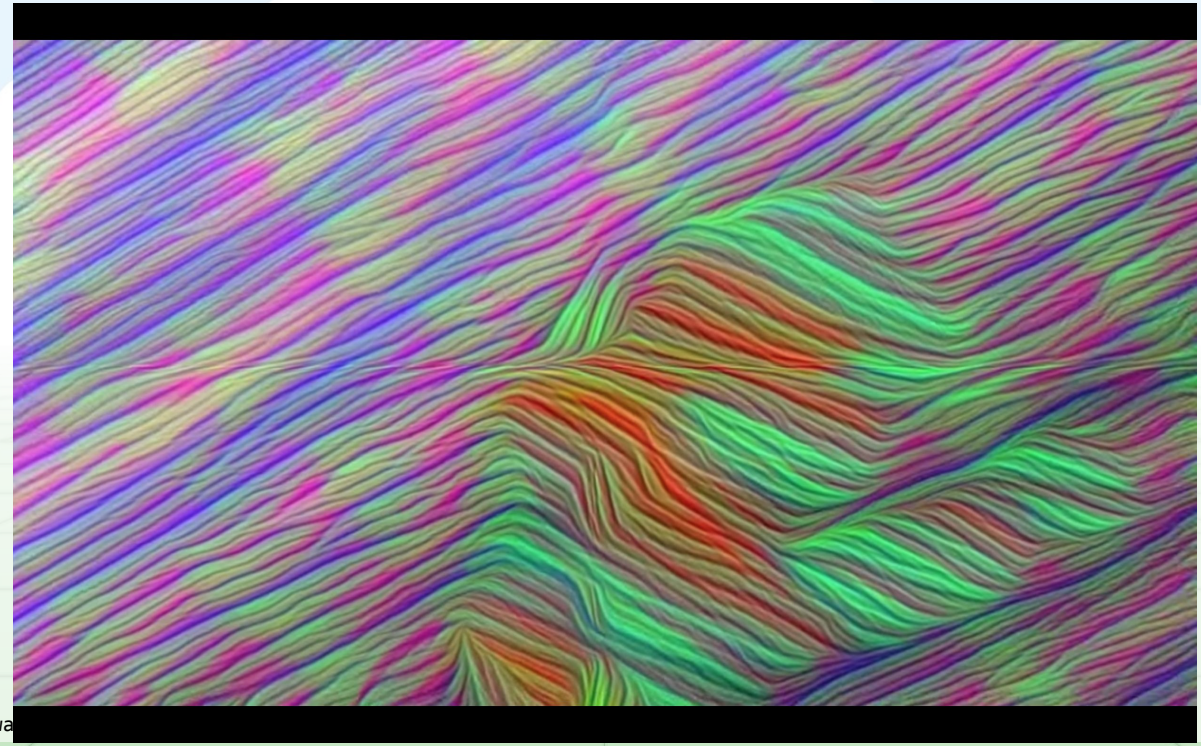
Neurons respond strongly to high contrast



https://www.youtube.com/wa

Middle layers: edges, curves, eyes

Neurons respond to simple features and shapes



https://www.youtube.com/wa

Middle layers: edges, curves, eyes

Neurons respond to simple features and shapes

https://www.youtube.com/wa

Middle layers: edges, curves, eyes

Neurons respond to simple features and shapes

https://www.youtube.com/wa

Later layers: eyes, objects, dogs

Neurons respond to arrangements of features

https://www.youtube.com/wa

Later layers: eyes, objects, dogs

Neurons respond to arrangements of features

https://www.youtube.com/wa

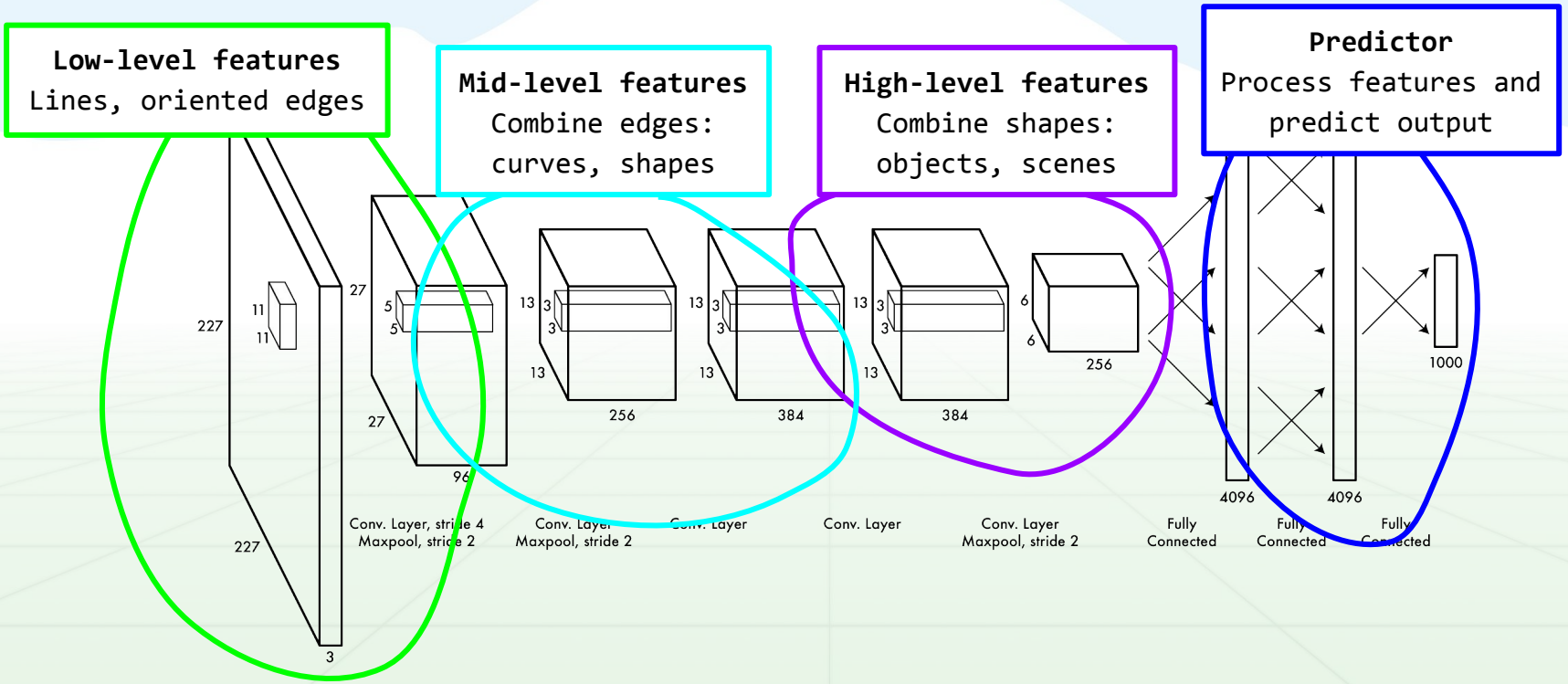
Later layers: eyes, objects, dogs

Neurons respond to arrangements of features

https://www.youtube.com/wa

Neural networks work!

At least, seem to do what we want them to



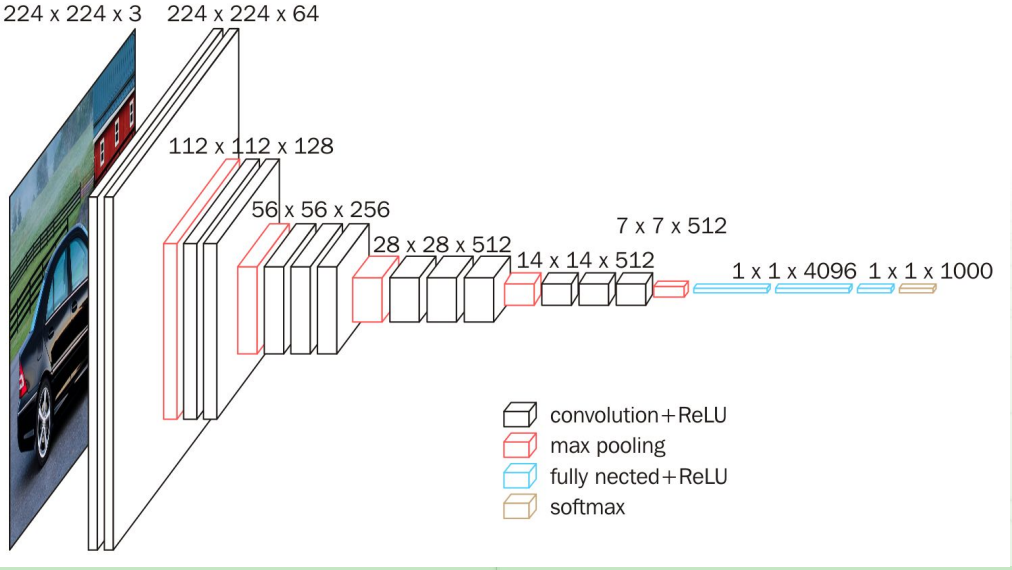
VGG: networks getting bigger

From the Visual Geometry Group at Oxford

Considered “very deep” at the time, 16 - 19 layers

VGG-16 is still commonly used as a feature extractor

Although really it shouldn’t be, much better alternatives



VGG: networks getting bigger

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

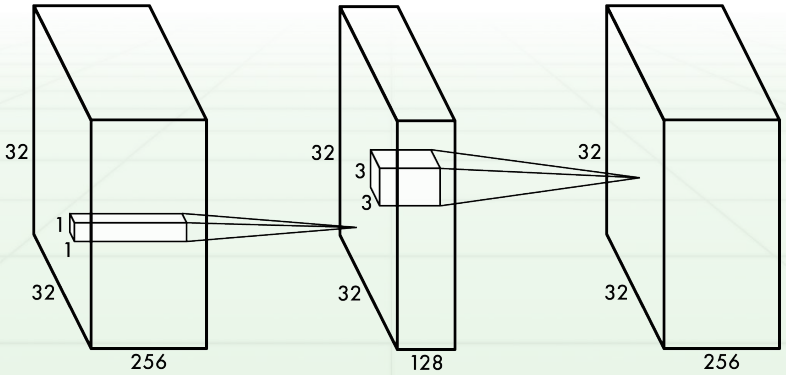
VGG is inefficient

Just stringing together a bunch of 3x3 convolutions in general, is pretty inefficient.

Network in network idea from Lin et al.:

- Use 3x3 convolutions to process spatial information
- Use 1x1 convolutions to reduce # of channels

E.g.



https://arxiv.org/pdf/1312.4400.pdf

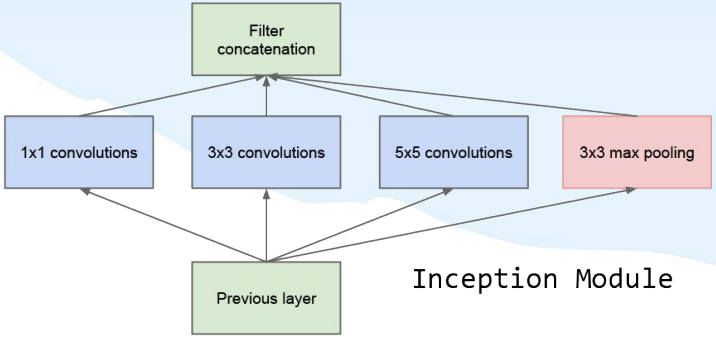
GoogleNet: networks getting weird

Another way to efficiency:

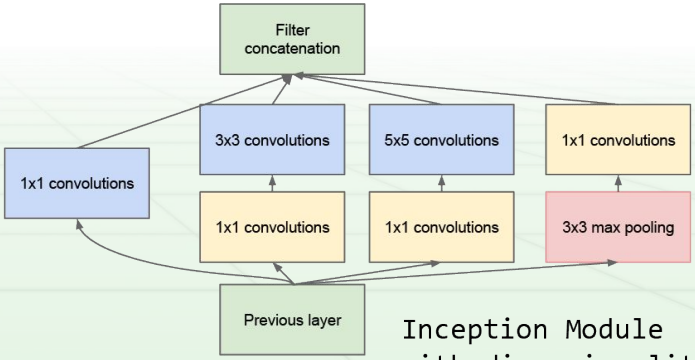
- Split layers
- Not just one size, but many
- 1x1, 3x3, 5x5

Also use 1x1 convs to compress feature maps

Can make large networks that are still efficient, better than VGG yet smaller and faster.



Inception Module



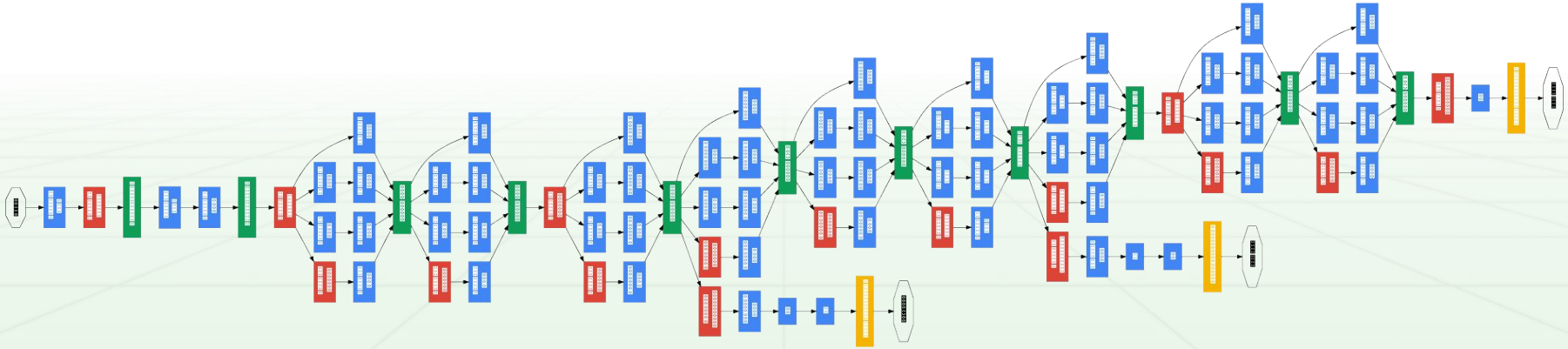
Inception Module with dimensionality reduction

https://www.cs.unc.edu/~wliu/papers/GoogLeNet.pdf

GoogleNet: networks getting weird

Like… really big

And this is sort of all we do now, try to make networks bigger without making them too expensive

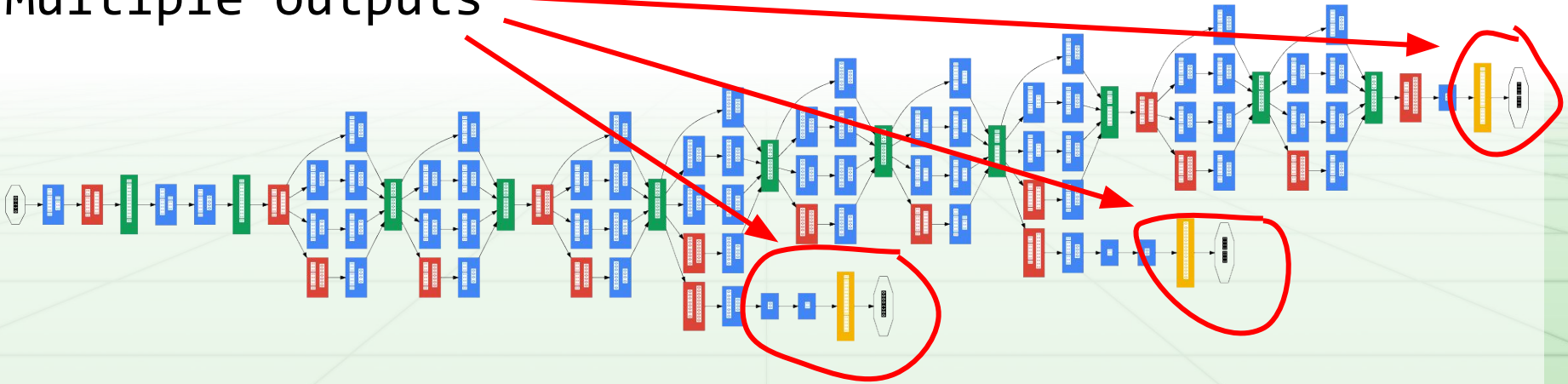


GoogleNet: networks getting weird

Like… really big

And this is sort of all we do now, try to make networks bigger without making them too expensive

Multiple outputs



Gradient explosion / vanishing

With very deep networks, the gradients flow through many layers of weights on their way back

With saturating activation functions like logistic or with small weights gradients can “vanish”

With non-saturating activations or large weights gradients can EXPLODE!

Learning doesn’t scale, what works at 2 levels doesn’t at 20

Brief intro to grad school

Grad school is awesome!

No rules

- Get to work on your own projects
- Everything is self-directed
- Just need an advisor who trusts you (and gives you money!)

Tons of smart people

- All of your colleagues are experts
- Get to learn from the people who did the actual research

Grad school is also...

Long

- 6+ years of putting life on hold

Tedious

- Research is slow
- Many projects fail, can invest 6 months with no results

Challenging

- Work is self-directed
- Success is not guaranteed
- HUGE component of randomness, luck
- Grad students 6x more likely to experience depression or anxiety than general public*

https://www.insidehighered.com/news/2018/03/06/new-study-says-graduate-students-mental-health-crisis

Grad school is *not school*

I’ve been here for 5 years, taken 10 classes

Grad students are researchers and TAs

- CSE has lots of money for research
- Students are expected to produce research and papers
- Other departments, less money
- Students TA *and* do research for their dissertation

Grad school is **WORK**

Advisor is your mentor/boss/manager

Each advisor has multiple grad students, Ali has ~10, spend time meeting with them and guiding research on high level

Grad students do **all** the actual programming, etc. and often do much of the paper writing.

Yet we are only paid for 20 hours / week (expected to be learning the other half)

Negotiations with UW

UW's initial offer:

0% raise

$1000+ more in fees

Union's asks:

Living wage

Better mental-health coverage

Trans-affirmative health care

Hourly employees raises (like ugrad TAs)

UW's response: society isn't there yet :-(

Made a lot of progress...

Not there yet

Union going on **strike**

First, one day strike on Tuesday

Ali will teach class, no office hours, probably no responses from TAs online

Want to help?

- Call, email Hank or Ana Mari Cauce
- Show up and hang out on picket lines