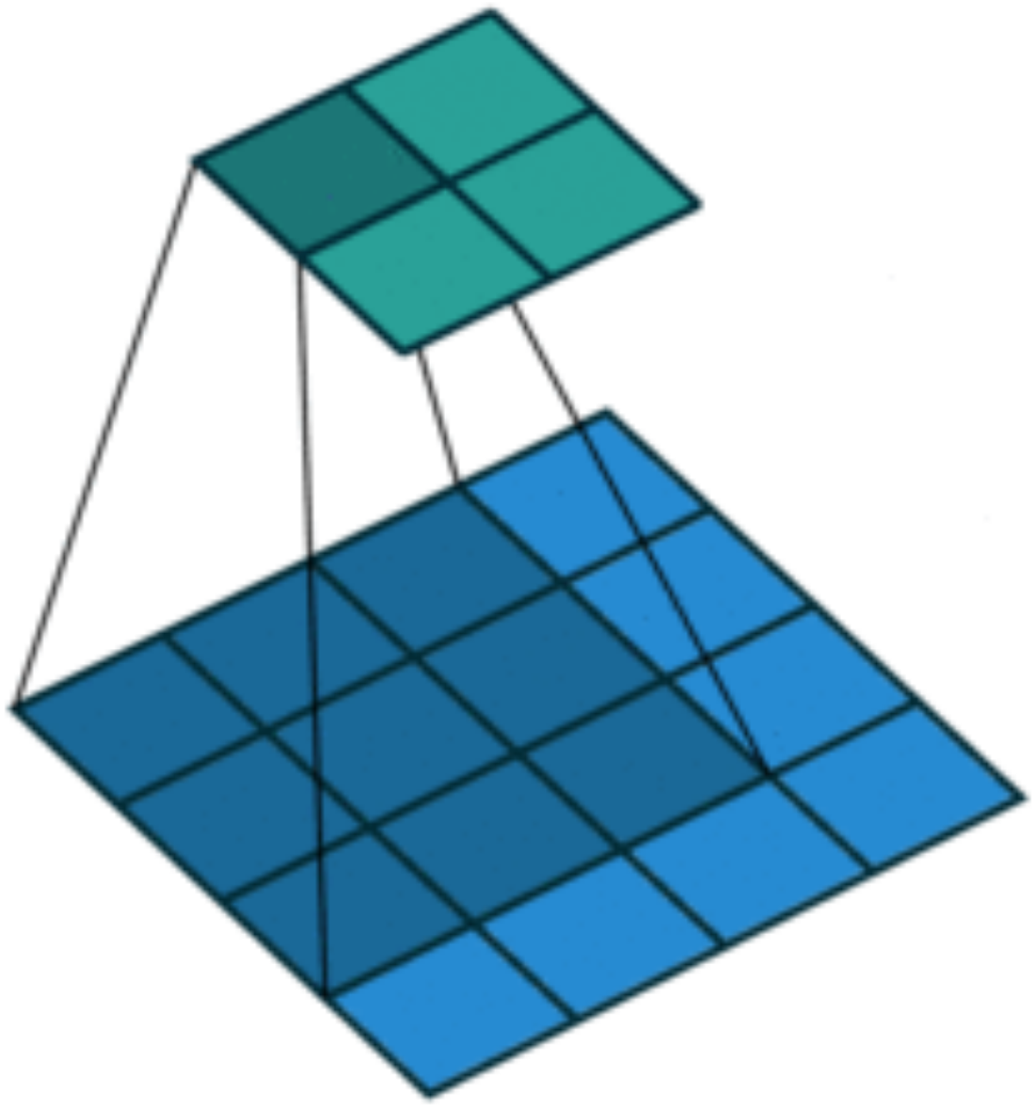
Convolutions-II

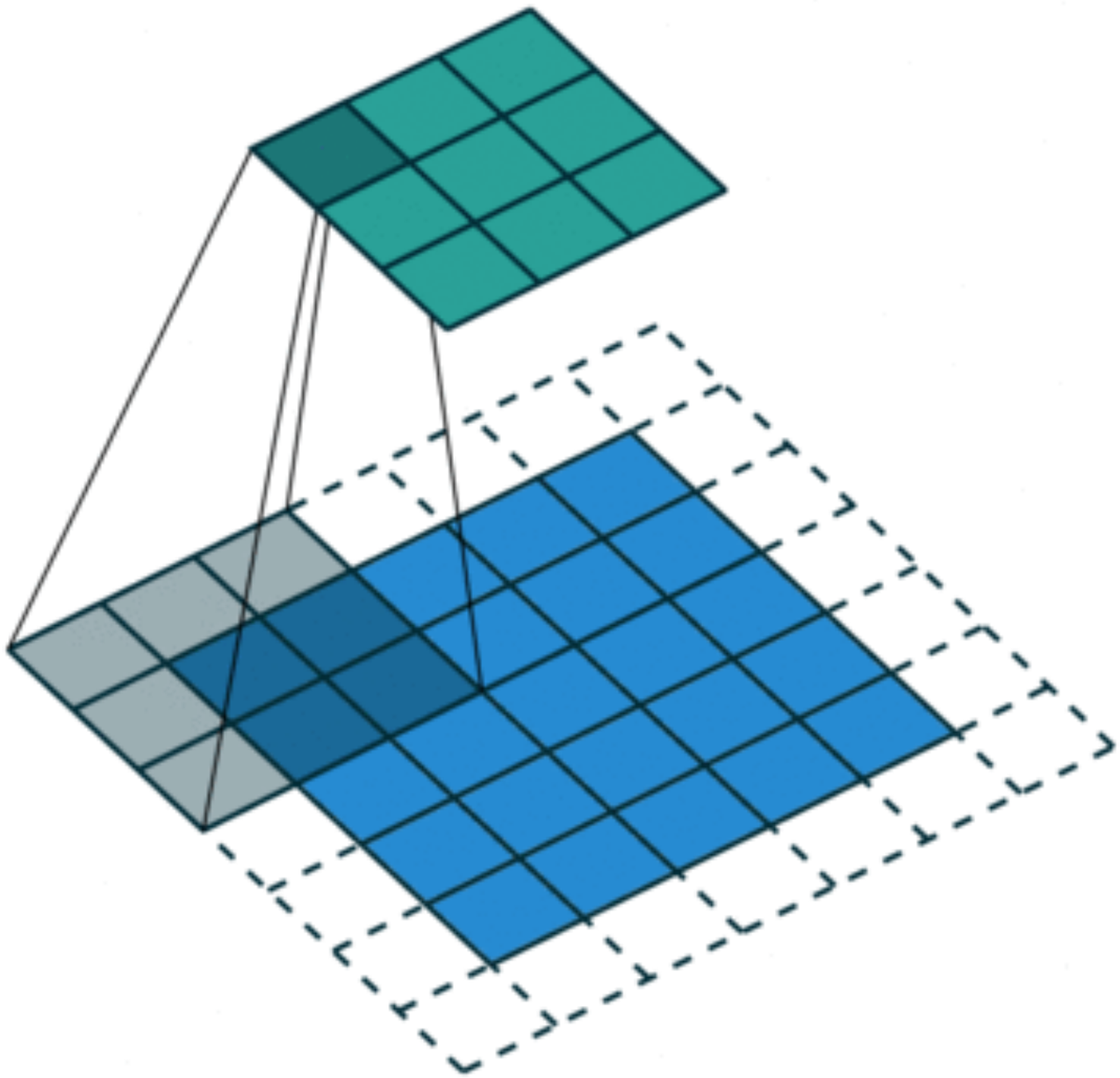
Recap

- Fully connected layer - too many parameters
- New building block:
  - Convolution operator
  - Locality
  - Parameter sharing

Convolution Animation



Padding + Strides Animation



Above GIFs From

- https://github.com/vdumoulin/conv_arithmetic

Code

2D Convolution layer:

- Keras:

```
Conv2D(
    filter_size=24,
    kernel_size=(2,2),
    strides=(1,1),
    padding='valid',
    input_shape=(128, 128, 3),
    channels_last=True
)
```

What Does This Mean

- 24 Filters
- Each filter is 2x2 - (patch size, FOV)
- Stride 1x1 - The adjacent patches are one over in both directions.
- Valid padding - don’t add any padding (tensorflow bs)
- input_shape - 128x128 color image (R, G ,B channels)
- Channels_last - tensor shape - see explanation

Parameters

- 24 Filters
- 2x2 filter
  - So patch has 4 pixels
  - Linear combination so **4 weights**
- **24 * 4 = 96**

Typically

- Conv2D is followed by:
  - Pooling layers
    - MaxPool
    - AvgPool
    - . . .

Pooling

- Also Image -> Image
- Operates at patch level as well
- Patches **Don’t Overlap**

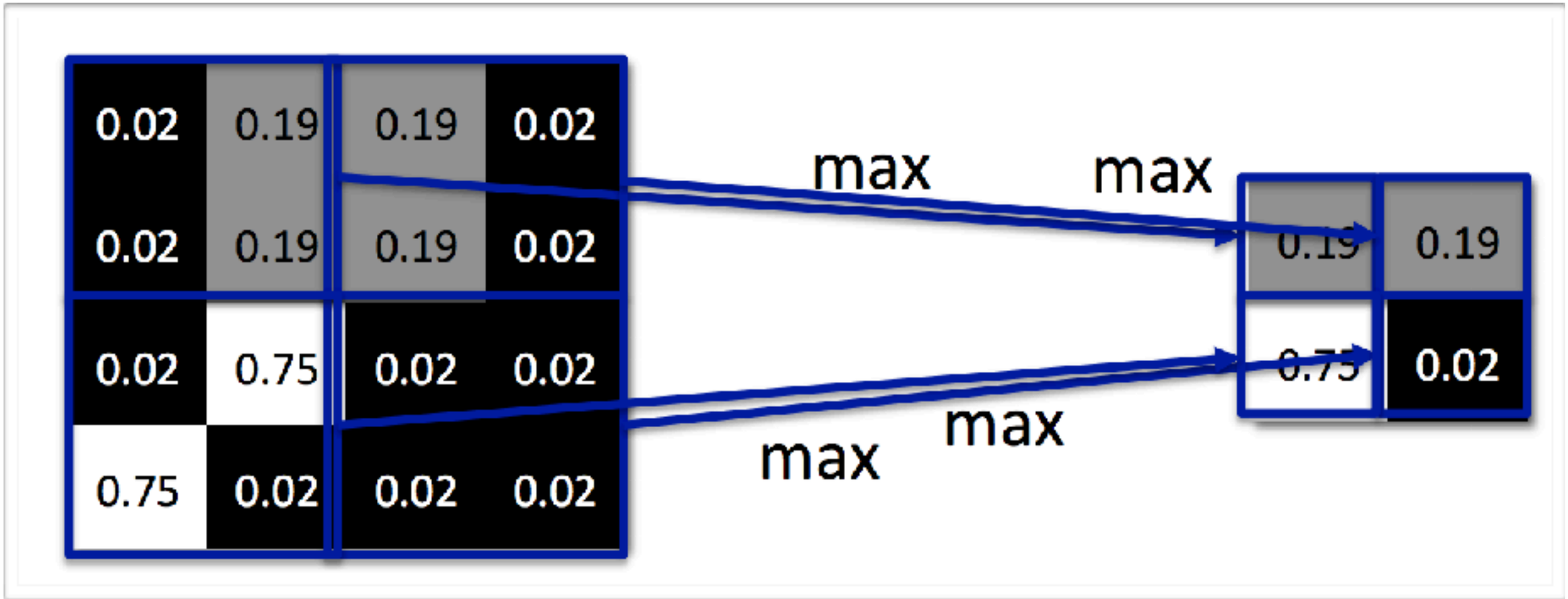
MaxPool

- Just take the max in a patch

AvgPool

- Just take the average in a patch

MaxPool


0.02
0.19
0.19
0.02
0.02
0.19
0.19
0.02
0.02
0.75
0.02
0.02
0.75
0.02
0.02
0.02
max
max
max
max
0.19
0.19
0.75
0.02

Code

- keras:

```
MaxPooling2D(
    pool_size=(2, 2),
    strides=None,
    padding='valid',
    data_format=None
)
```

What Do You Get

A New Shrunken(ish) image

Parameters

- None

Hyperparameters

- Stride

- Patch size

Now

- We discussed
- Image -> Image transformations
- Pooling, Conv2D

How does it fit in

- Cross entropy not defined for image
- Regression not defined for image
- Etc. Etc.

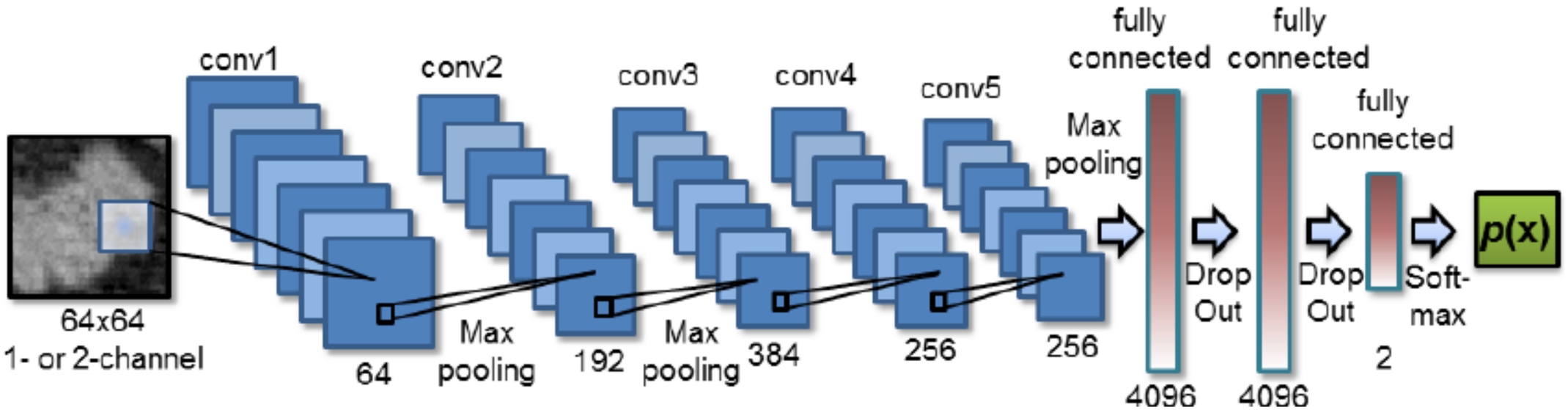
Deep Convnet

- Series of image transformations

- At the end you get an image

Just

- Flatten the resulting image
- Use normal fully connected layers from there on


conv1
conv2
conv3
conv4
conv5
fully connected
fully connected
fully connected
Max pooling
64x64
1- or 2-channel
64
Max pooling
192
Max pooling
384
256
256
4096
Drop Out
4096
Drop Out
2
Soft-max
p(x)

Code Example

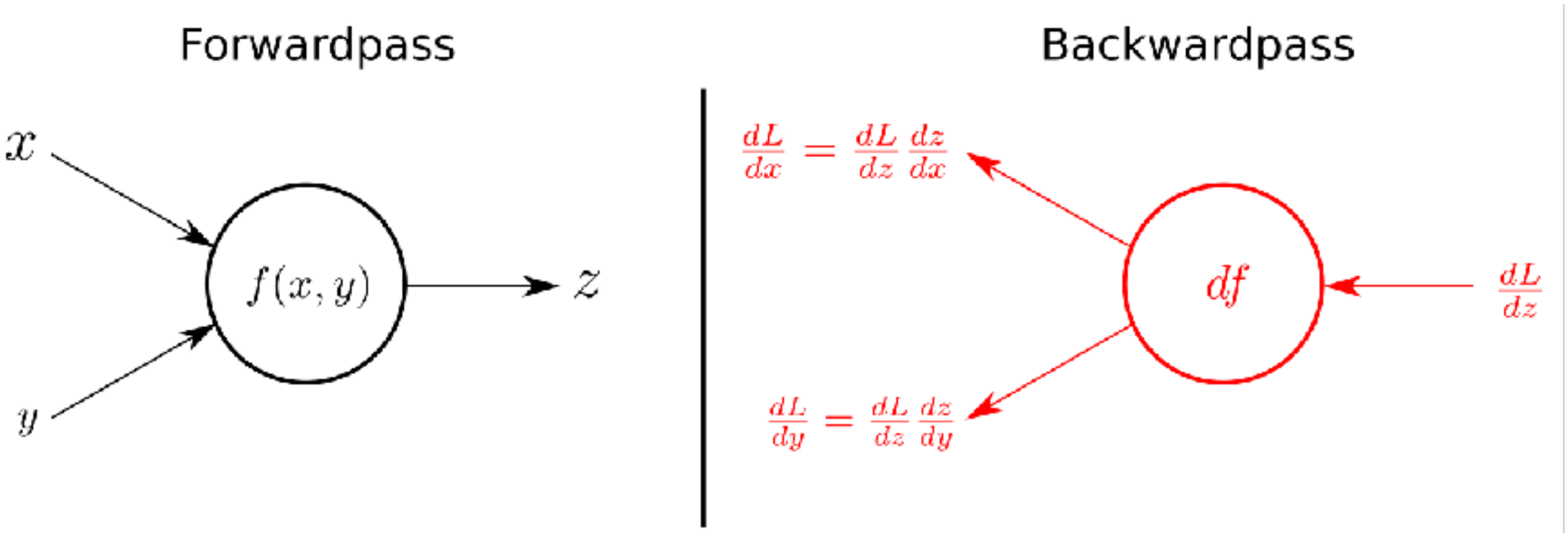
- MNIST

Code Example

- MNIST Tensorflow
- https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist_deep.py

Training Details

The Backprop Picture



Here

$$\begin{bmatrix} x_1^1 & x_1^2 & x_1^3 \\ x_2^1 & x_2^2 & x_2^3 \\ x_3^1 & x_3^2 & x_3^3 \end{bmatrix} \quad \begin{bmatrix} w_1^1 & w_1^2 \\ w_2^1 & w_2^2 \end{bmatrix}$$

$$\begin{bmatrix} h_1^1 & h_1^2 \\ h_2^1 & h_2^2 \end{bmatrix}$$

Backward Pass

$$\begin{bmatrix} \partial L/\partial h_1^1 & \partial L/\partial h_1^2 \\ \partial L/\partial h_2^1 & \partial L/\partial h_2^2 \end{bmatrix}$$

Observe

$$h_1^1 = w_1^1 x_1^1 + w_1^2 x_1^2 + w_2^1 x_2^1 + w_2^2 x_2^2$$

We Are Given

$$\partial L / \partial h_1^1$$

We Want

$$\partial L / \partial w_i^j$$

$$\partial L / \partial x_i^j$$

i.e. the partial derivative w.r.t the filter and the input

Consider

$$\partial L/\partial w_1^1 = \partial L/\partial h_1^1 \cdot X_1^1 + \partial L/\partial h_1^2 \cdot X_1^2 + \partial L/\partial h_2^1 \cdot X_2^1 + \partial L/\partial h_2^2 \cdot X_2^2$$

Gist

- Backward pass is also a convolution

Initialization

- Guess how?

Gist

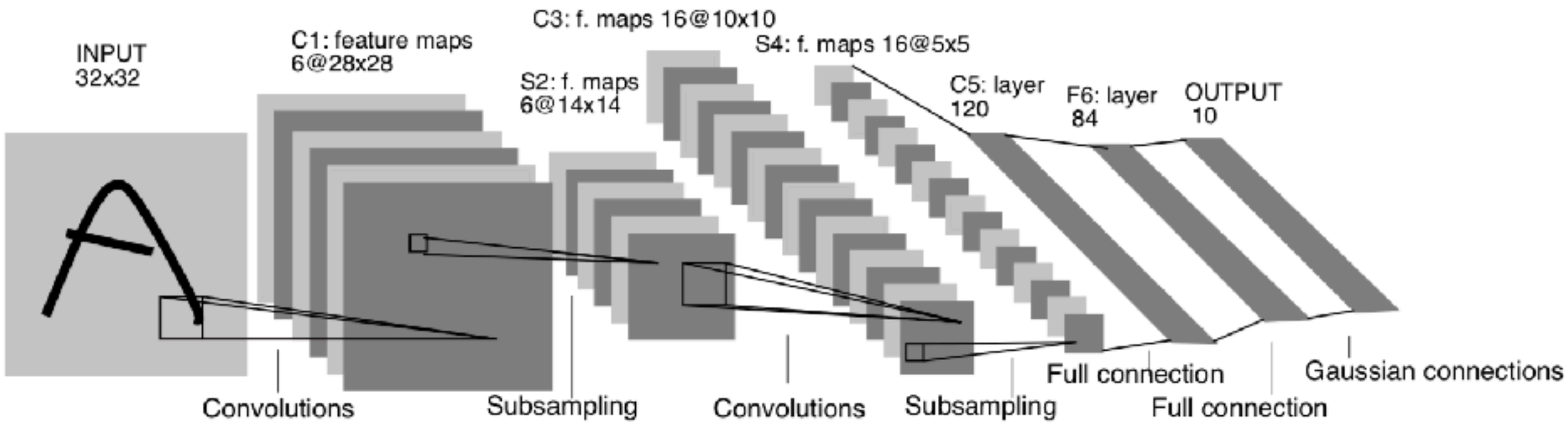
- Error gradient is also a convolution

Arsenal

- A few loss functions
- Fully connected layers
- Convolutional layers

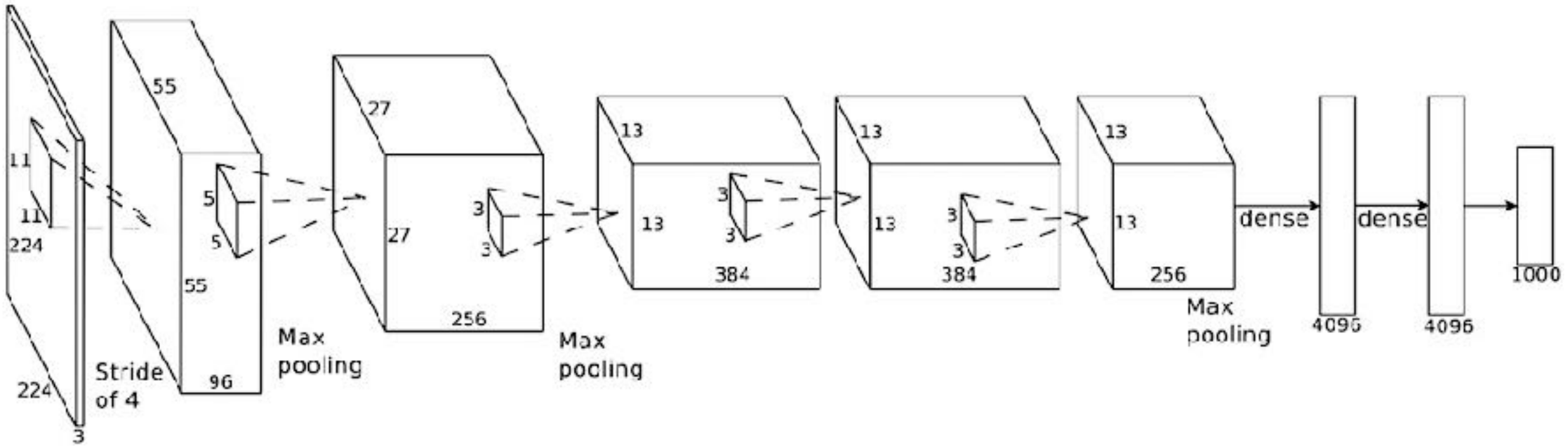
Examples

LeNet


INPUT
32x32
C1: feature maps
6@28x28
S2: f. maps
6@14x14
C3: f. maps 16@10x10
S4: f. maps 16@5x5
C5: layer
120
F6: layer
84
OUTPUT
10
Convolutions
Subsampling
Convolutions
Subsampling
Full connection
Full connection
Gaussian connections

AlexNet

- Imagenet LSVRC-2010 Challenge
- 1.2 Million images
- 1000 classes
- The first famous convnet


11
11
224
224
3
55
55
96
5
5
Stride
of 4
Max
pooling
27
27
256
3
3
Max
pocling
13
13
384
3
3
13
13
384
3
3
13
13
256
Max
pooling
dense
4096
dense
4096
1000

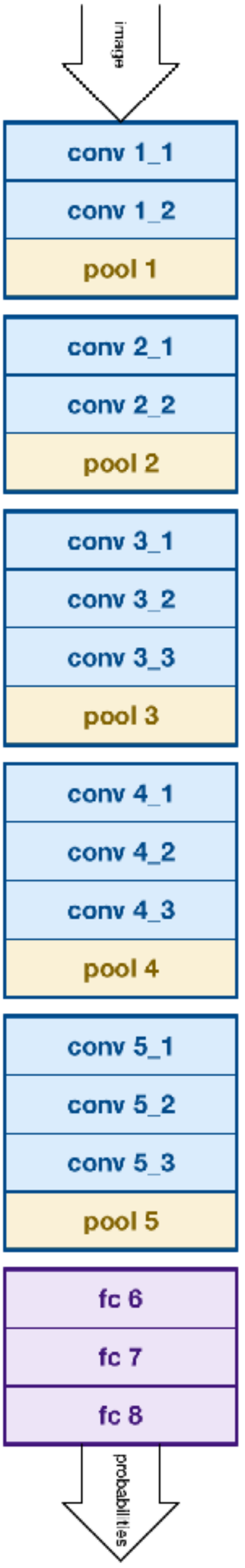


image
conv 1_1
conv 1_2
pool 1
conv 2_1
conv 2_2
pool 2
conv 3_1
conv 3_2
conv 3_3
pool 3
conv 4_1
conv 4_2
conv 4_3
pool 4
conv 5_1
conv 5_2
conv 5_3
pool 5
fc 6
fc 7
fc 8
probabilities

VGG

VGG

- All kernels are 3x3

- All pooling layers are 2x2

Thought Experiments

Flickr Example

- Images
- Attributes per image
- Figure out tags

Siamese Net Example

- Face pair

- Same person?

Object Detection Example

- Simple technique

Convolution Wins

- Big wins recently in
  - Machine translation
  - Audio synthesis
  - Video
  - etc.

Debugging

Recap

- Fully Connected Layer

- ReLUs fold the space

What Are The Convolutions Doing?

- Typical approaches are empirical
- Plot filter responses
- Visualize embeddings (representations)
- Visualize weights

Visualizing Responses

Deep Visualization Toolbox

yosinski.com/deepvis

#deepvis


Jason Yosinski


Jeff Clune


Anh Nguyen


Thomas Fuchs


Hod Lipson

Cornell University

UNIVERSITY OF WYOMING

NASA Jet Propulsion Laboratory
California Institute of Technology

https://www.youtube.com/watch?v=AgkfIQ4IGaM

Intuition

- Pick some layer you want to “enhance”
- Backprop all the way to the image
- Result = trippy images

MNIST TSNE

- Demo

Deepdream

- https://github.com/google/deepdream/blob/master/dream.ipynb

Deepdream

- Enhance responses

- how?

Acknowledgements

- Ruslan Salakhudtinov - http://www.cs.cmu.edu/~rsalakhu/10707/Lectures/Lecture_Conv1.pdf
- Some figures from CS231N @ Stanford
- Good vibes from the Keras and TF teams
- Vincent Dumoulin - https://github.com/vdumoulin/conv_arithmetic