CS 523: Multimedia Systems

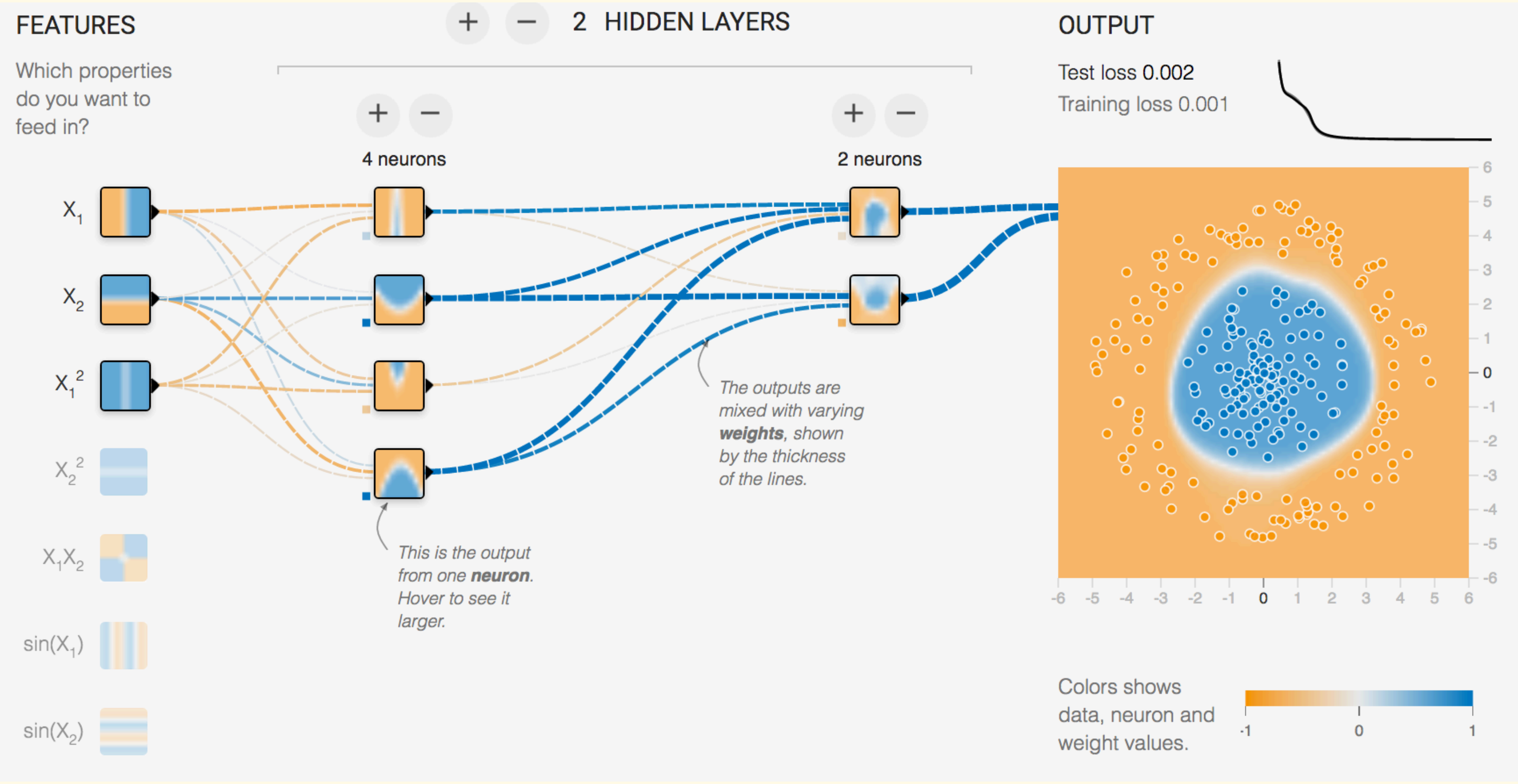
Angus Forbes

creativecoding.evl.uic.edu/courses/cs523

Today

- Convolutional Neural Networks
- Work on Project 1

http://playground.tensorflow.org/


FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
+ − 2 HIDDEN LAYERS
+ −
4 neurons
+ −
2 neurons
*This is the output from one **neuron**. Hover to see it larger.*
*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*
OUTPUT
Test loss 0.002
Training loss 0.001
Colors shows data, neuron and weight values.
-1
0
1

Convolutional Neural Networks

ConvNets, or CNNs:

- An approach to model visual processing, which has been suggested to be hierarchical or “pyramidal” in nature
- Detects high-level features first, then more fine grained features
- Uses convolution to identify rotation-invariant features
- Math is similar to commonly used mathematics in image processing kernels

Convolutional Neural Networks

- Have had enormous success at image classification tasks
- Widely used by Google, Facebook, Instagram for identifying both general categories and individual elements in photos and vidoes; core of self-driving car algorithms, etc.
- Have been adapted to wide range of other types of problems, even those that don't involve images, such as search tasks and recommendation systems

Vision neurons

- In 1962, Hubel & Wiesel inserted microscopic electrodes into the visual cortex of experimental animals to read the activity of single cells in the visual cortex while presenting various stimuli to the animal's eyes.

- Nearby cells in the cortex represented nearby regions in the visual field
- The visual cortex represents a spatial map of the visual field.

Vision neurons

- Individual cells in the cortex respond to the presence of edges in their region of the visual field.
- Some of these cells fire only in the presence of a vertical edge at a particular location in the visual field, while other nearby cells responded to edges of other orientations in that same region of the visual field.
- Orientation-sensitive cells, called "simple cells", are found all over the visual cortex.

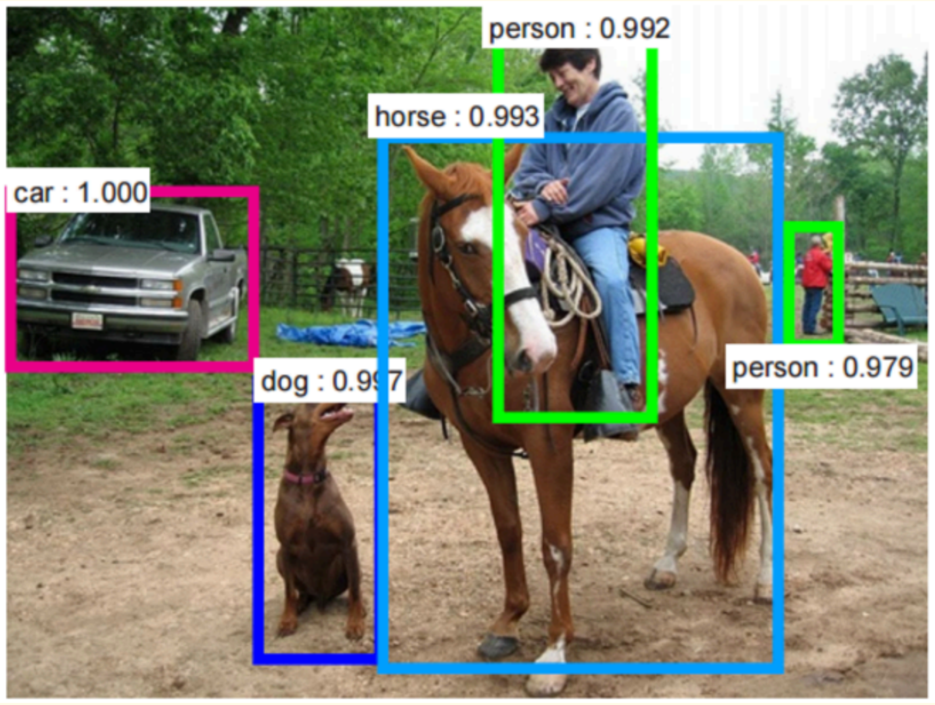
Vision neurons

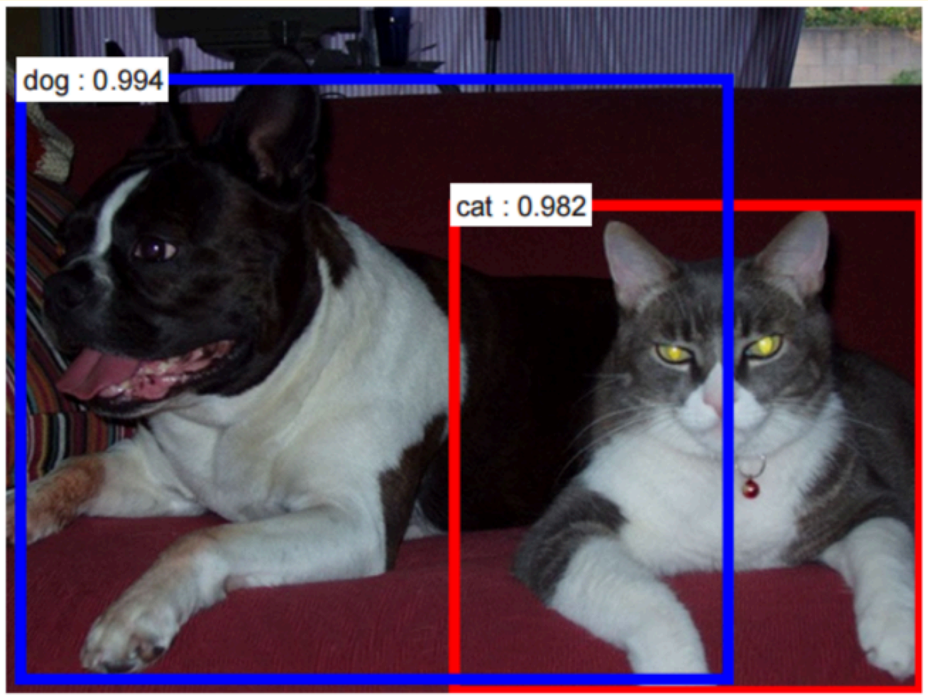
- “Complex cells” also responded to edges of a specific orientation at a specific location in the visual field, but also respond in a contrast-insensitive manner. - -

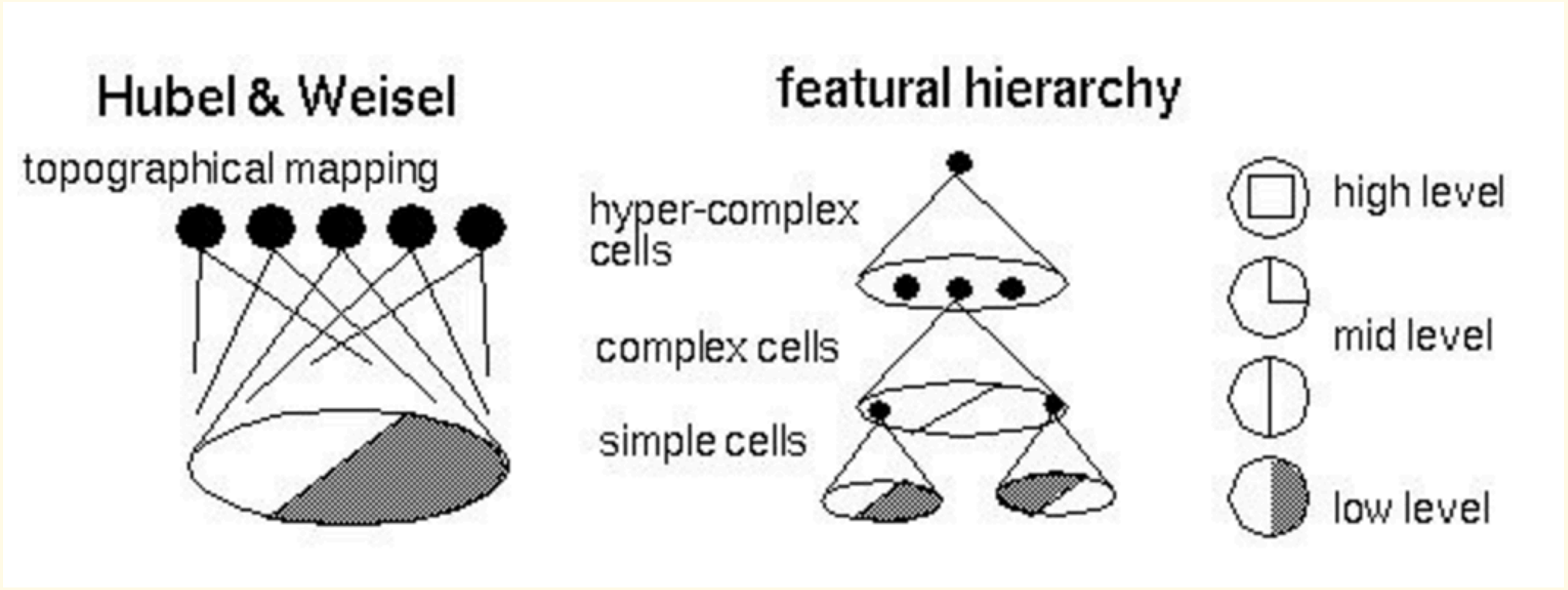
- Complex cells respond to their edges in a larger region of the visual field. This suggested that complex cells received input from lower level simple cells by way of a receptive field.

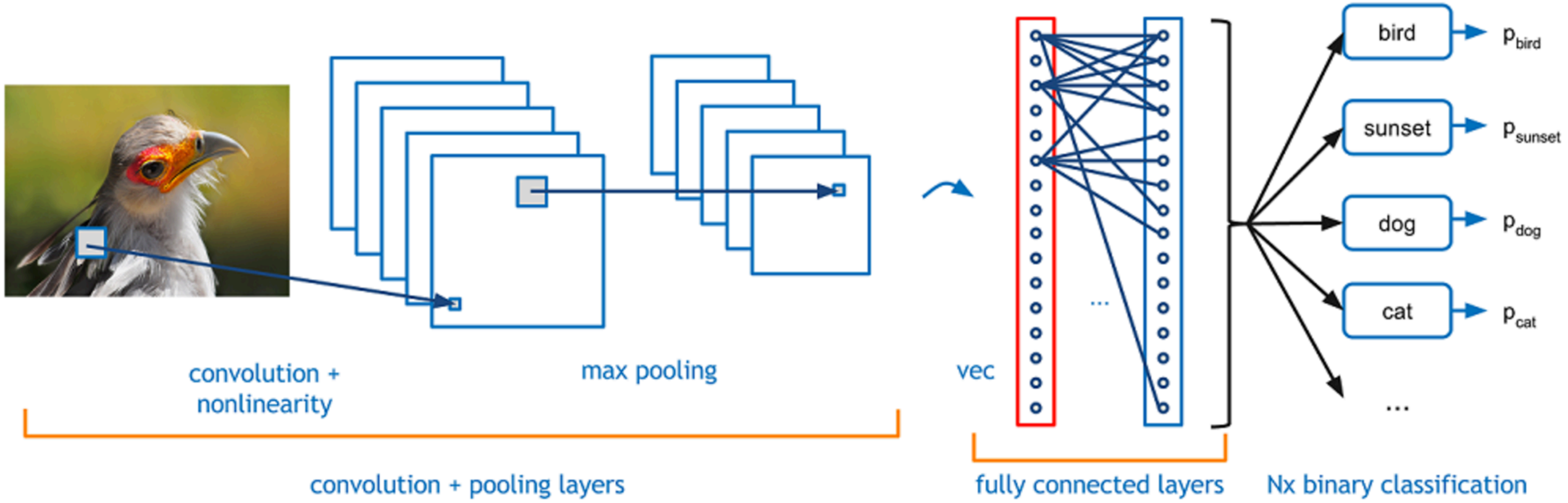
- Higher level "hypercomplex cells” respond to more complex combinations of the simple features, for example to two edges at right angles to each other in a yet larger region of the visual field.

- Steven Lehar http://cns-alumni.bu.edu/~slehar/webstuff/pcave/hubel.html


person : 0.992
horse : 0.993
car : 1.000
dog : 0.997
person : 0.979


dog : 0.994
cat : 0.982


Hubel & Weisel
topographical mapping
featural hierarchy
hyper-complex cells
complex cells
simple cells
high level
mid level
low level


convolution +
nonlinearity
max pooling
convolution + pooling layers
vec
...
fully connected layers
bird
p_{bird}
sunset
p_{sunset}
dog
p_{dog}
cat
p_{cat}
...
Nx binary classification

CNNs

Last week we talked about multilayer perceptrons, where each neuron “fired” (was included in the summation of the next layer) if its value was greater than a threshold value.

CNNs include an initial type of layer, called a “*convolution layer,*” in which a simple convolution operation is applied (perhaps successively) to the input data (eg, pixels).

CNNs

A convolution filter in image processing transforms each of the input pixels into a new “convolved” output pixel:

for each pixel:

- place the convolution filter so that it is centered on the pixel
- where the filter overlaps a pixel overlaps, multiply the the element in the filter by that pixel, and divide it by the number of overlaps
- add them all together
- that’s the convolved value of your output pixel

CNNs

These are called filters because the output will low or zero if the input doesn’t match the values in the filter.

For example, the part of the Sobel filter that detects vertical edges looks like this:

```
[
    -1  0  1
    -2  0  2
    -1  0  1
]
```

CNNs

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

- If we were applying this filter to a patch of pixels that were all black, it would return a 0, indicating that there were no vertical edges to the left or right of our input pixel

- but if, say all the pixels to the left were white, we'd get a value of ?

CNNs

$$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

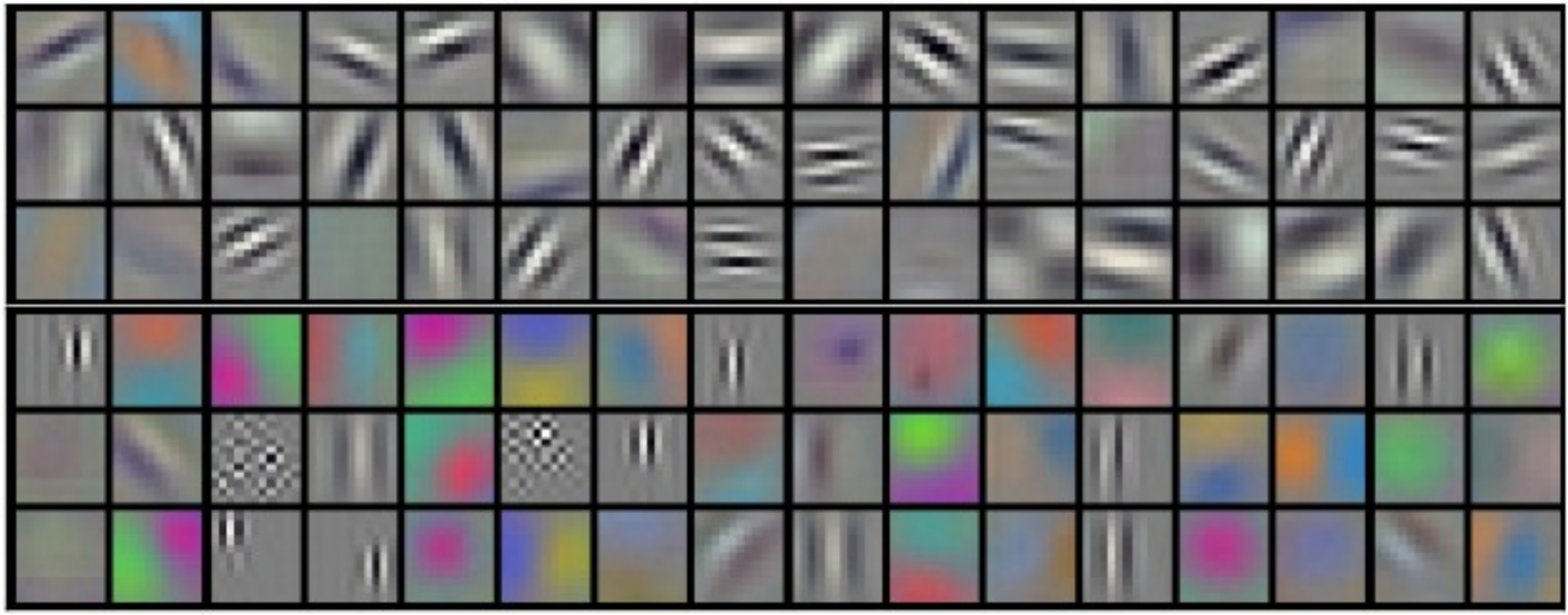
- If we were applying this filter to a patch of pixels that were all black, it would return a 0, indicating that there were no vertical edges to the left or right of our input pixel

- but if, say all the pixels to the left were white, we'd get a value of (-1*1.0 + -2*1.0 + -1*1.0) / 9 = -0.4444

CNNs

What the CNN does is that it *learns* these filters – that is, by setting the values each of the cells of the convolutional kernel.

The filters are generally larger than 3x3, and only ones that are successful in distinguishing features that help to classify the training data correctly will survive the training session.

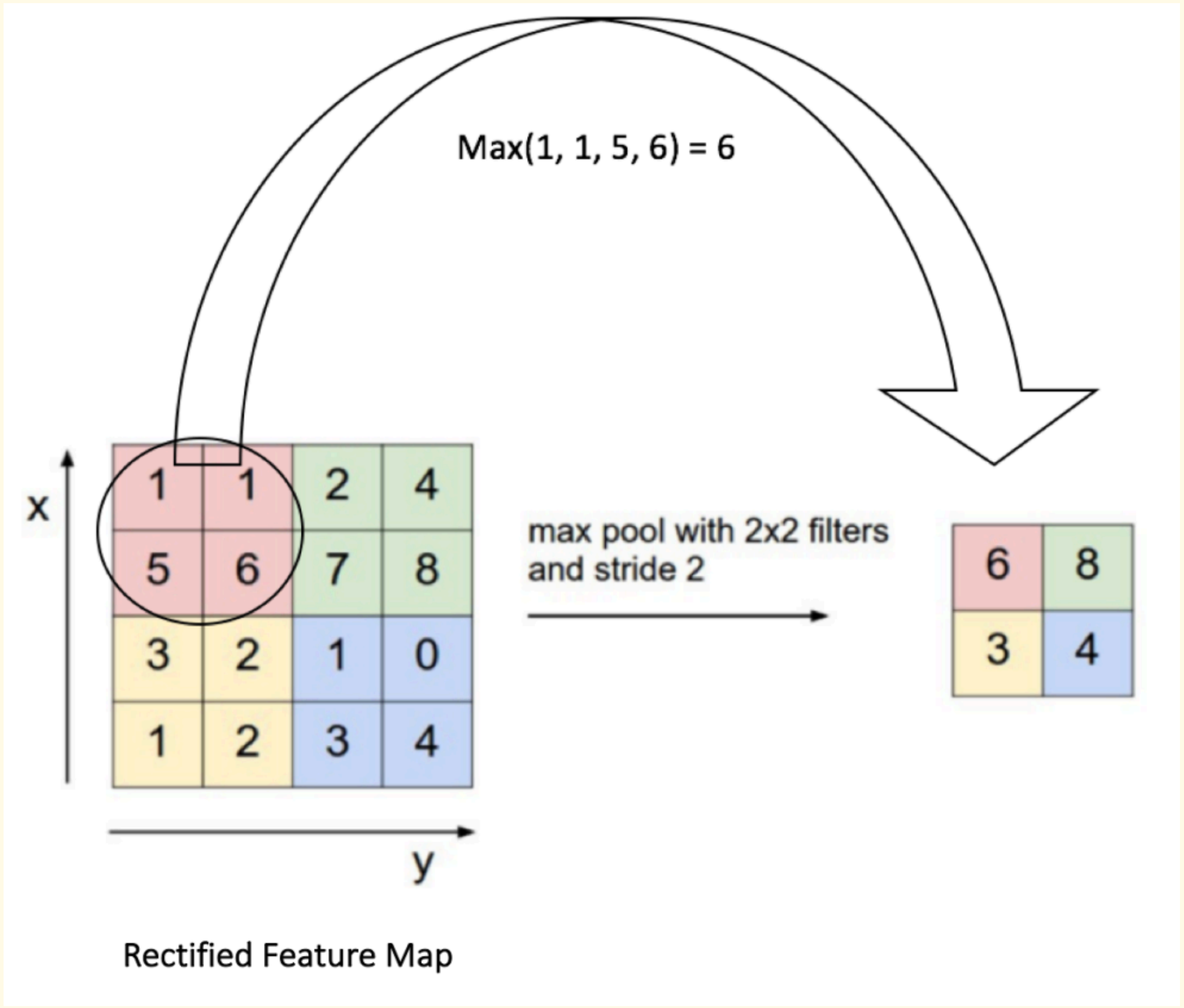


- Krizhevsky, et al., “ImageNet Classification with Deep Convolutional Neural Networks”

CNNs

The next layer is called the *pooling layer*, which basically performs a downsampling operation, usually by taking maximum within a particular range of elements, reducing the size of the *feature map*

- Helps to control overfitting
- Makes the network ignore small fluctuations or distortions
- Ideally, by performing multiple layers of convolution + pooling, our network becomes scale invariant, so that features of any size, anywhere in the image, can be detected

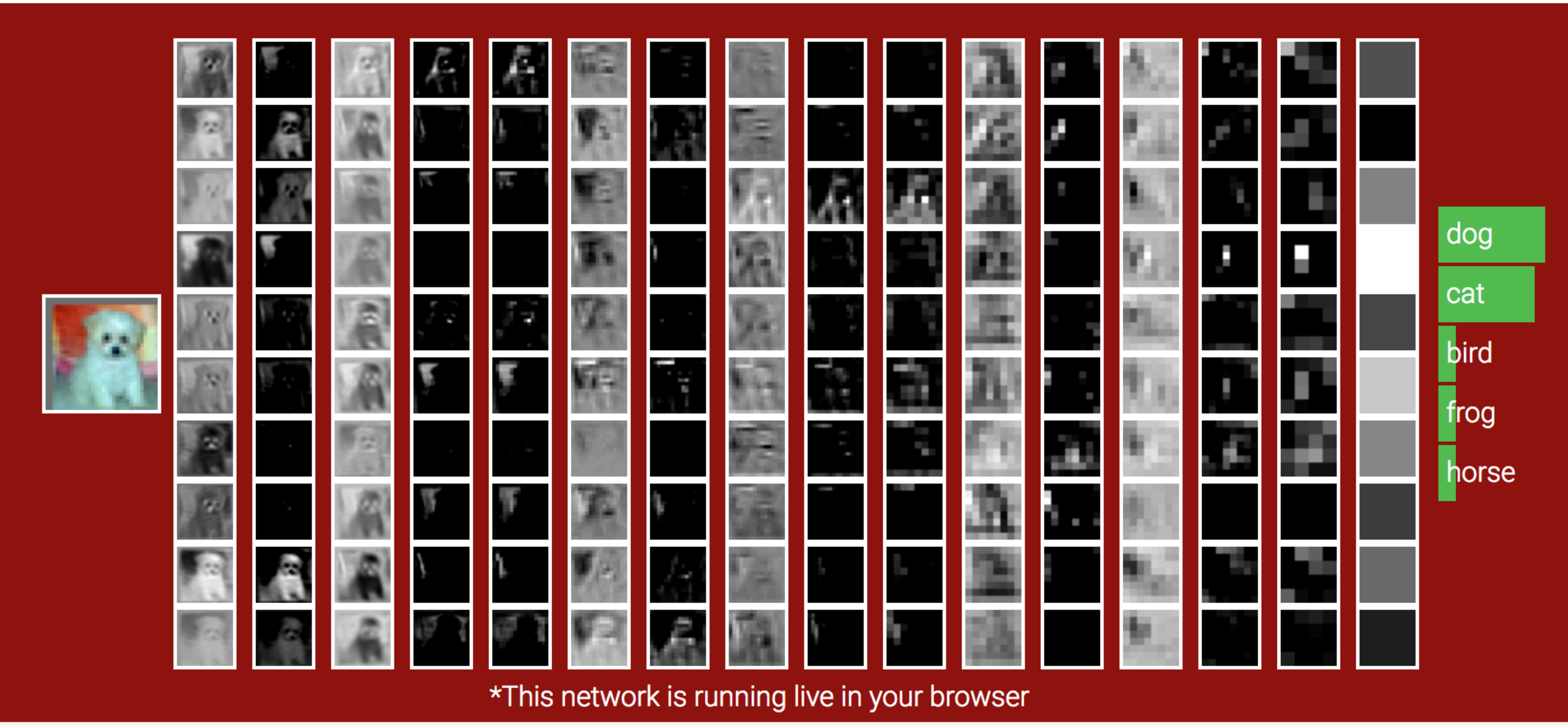

Max(1, 1, 5, 6) = 6
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters and stride 2
6 8
3 4
Rectified Feature Map

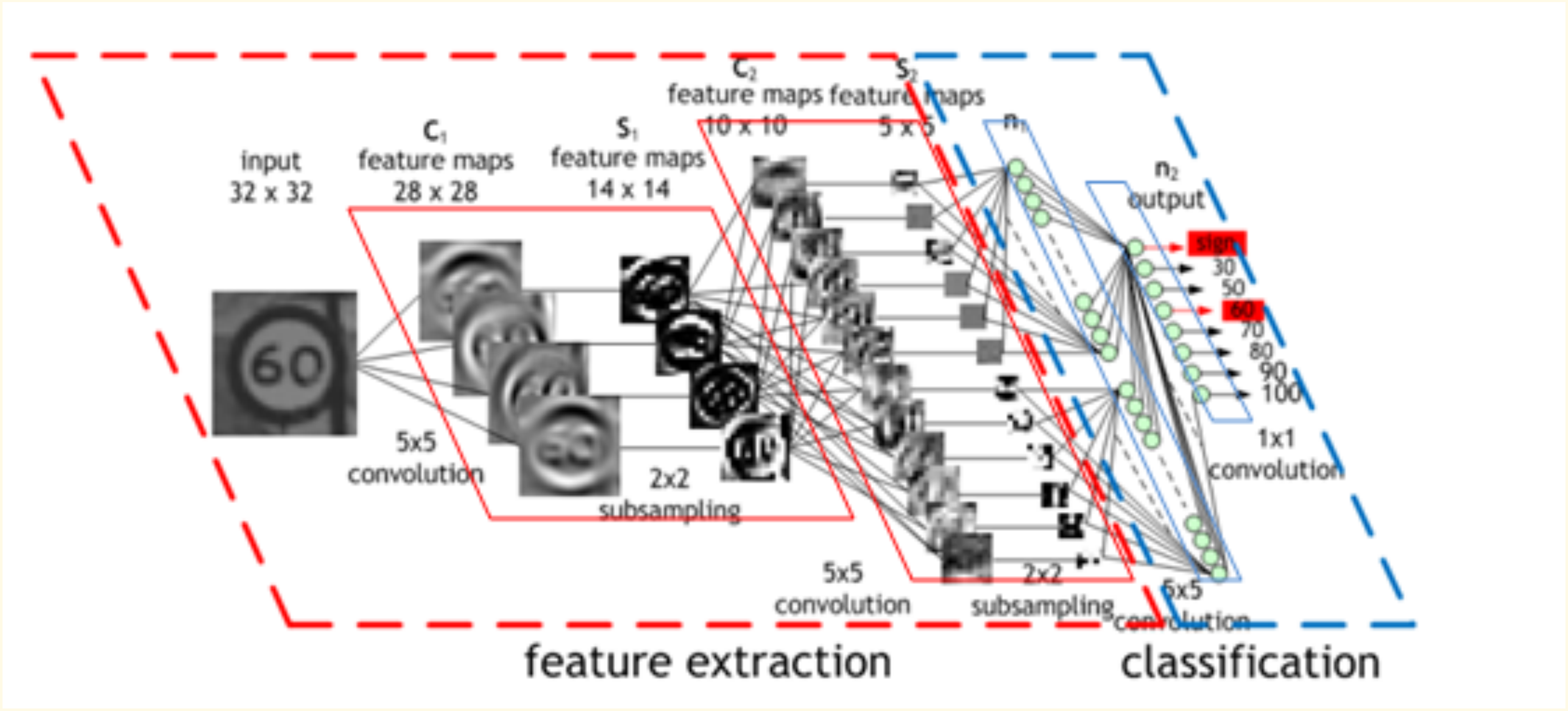
CNNs

By the end of the convolution layers and pooling layers, we should have discovered a set of high-level features that can be used to classify images. This is the *feature extraction* component of the network

The features are the inputs to a “normal” neural network, which can learn nonlinear combinations of these high-level features (i.e. similar to the simple MNIST example network from the TensorFlow demo). This is the *classification* component of the network.

http://cs231n.stanford.edu/




input
32 x 32
C_1
feature maps
28 x 28
S_1
feature maps
14 x 14
C_2
feature maps
10 x 10
S_2
feature maps
5 x 5
n_1
n_2
output
sign
30
50
60
70
80
90
100
5x5
convolution
2x2
subsampling
5x5
convolution
2x2
subsampling
5x5
convolution
1x1
convolution
feature extraction
classification

CNNs

Just like in a normal neural network, backpropogation is used to adjust the weights in the network, but in CNNs, the filter values are also updated.

You might also hear the term “dropout,” which simply means that during training certain neurons chosen at random are “turned off,” and don’t participate in firing (even if summation of weights*inputs is above the threshold that would make the activation function fire), also don’t participate in the backpropogation step.

Reduces “complex co-adaptations of neurons”, since a neuron cannot rely on the presence of particular other neurons, reducing overfitting, but requiring more training.

CNNs

Online Demo:

http://scs.ryerson.ca/~aharley/vis/conv/

http://scs.ryerson.ca/~aharley/vis/conv/flat.html

CNNs in the news

- “Deep learning algorithm does as well as dermatologists in identifying skin cancer”
  http://news.stanford.edu/2017/01/25/artificial-intelligence-used-identify-skin-cancer/

-“Traffic signs classification with a convolutional network”
  http://navoshta.com/traffic-signs-classification/

- “Indoor Space Recognition using Deep Convolutional Neural Network: A Case Study at MIT Campus”
  https://arxiv.org/abs/1610.02414

- Many CNN links on course website

Project 1

- Implement Deep Dream or Neural Style
- Learn by doing!
- Main goal is to understand:

    Tensorflow code (lots of examples, tutorials online to learn from)

    Neural Network architecture and algorithms
- Can work alone in groups of 2 or 3
- An interactive interface would be nice as well, plus adding your own twist to it… but main goal is get used to coding in python + TF

Lab

- Has everyone been able to run the MNIST example from the TF website?
- No?

    then work on it now!
- Yes?

    then:

    a) help others

    b) work through the other MNIST example

    c) figure out how to visualize neurons (the weights attached to the neurons) during (and at end of) training

    d) visualize the number that were misclassified – can you infer why the NN got confused?

Next Week

- Project 1
- Introduction to Recurrent Neural Networks (RNNs)
- See syllabus for reading assignment