Bayesian Network & ML Basics

Jinghao Zhao

11/19/2020

Chain Rule & Bayes Rule

- Chain rule:

$$
\begin{aligned}
P(A,B,C,D,E) &= P(A|B,C,D,E) \cdot P(B,C,D,E) = P(A|B,C,D,E)\ P(B|C,D,E)\ P(C,D,E) \\
&= P(A|B,C,D,E)\ P(B|C,D,E)\ P(C,D,E) \\
&= \ldots \\
&= P(A|B,C,D,E)\ P(B|C,D,E)\ P(C|D,E)\ P(D|E)\ P(E)
\end{aligned}
$$

- Bayes rule: important for reverse conditioning

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Bayesian Learning

- Use Bayes rule:

$$P(\theta \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \theta)P(\theta)}{P(\mathcal{D})}$$

- Or equivalently:

$$\underbrace{P(\theta \mid \mathcal{D})}_{\text{posterior}} \propto \underbrace{P(\mathcal{D} \mid \theta)}_{\text{likelihood}}\underbrace{P(\theta)}_{\text{prior}}$$

Task: Probability Query

Given a Bayesian Network, we know what's the <u>joint probability of all random variables</u>.
Now we want to compute some other probability!

1. **Conditional probability query**

Compute

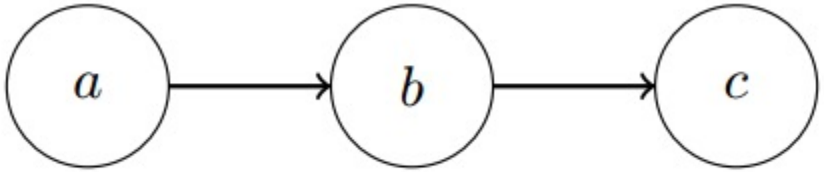
$$P(Y|E = e)$$

- E: *Evidence*
  - A subset of random variables with known (instantiated) values e
- Y: Query variables
  - A subset of random variables (values unknown)

2. Marginalize one or a set of variable

$$P(X_1, X_2, \dots)$$

Example – Inference



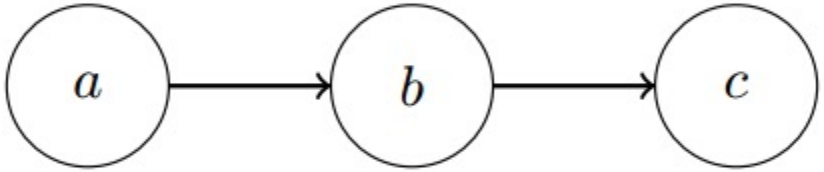
a	$\Pr(a)$
1	1/2
0	1/2

a	b	$\Pr(b \mid a)$
1	1	1/8
1	0	7/8
0	1	1/4
0	0	3/4

b	c	$\Pr(c \mid b)$
1	1	4/5
1	0	1/5
0	1	1/4
0	0	3/4

compute Pr(a=T|b=T)

Example – Inference



$$\Pr(a = \texttt{true} \mid b = \texttt{true}) = \frac{Pr(a = \texttt{true}, b = \texttt{true})}{Pr(b = \texttt{true})}$$
$$= \frac{\frac{1}{2} \cdot \frac{1}{8}}{\frac{1}{2} \cdot \frac{1}{8} + \frac{1}{2} \cdot \frac{1}{4}}$$
$$= \frac{\frac{1}{16}}{\frac{1}{16} + \frac{1}{8}}$$
$$= \frac{1}{3}$$

a	$\Pr(a)$
1	1/2
0	1/2

a	b	$\Pr(b \mid a)$
1	1	1/8
1	0	7/8
0	1	1/4
0	0	3/4

b	c	$\Pr(c \mid b)$
1	1	4/5
1	0	1/5
0	1	1/4
0	0	3/4

Variable Elimination

- Dynamic Programming
- **Sum out one variable at a time**
- Basic computation step: manipulation of factors
- Cache intermediate results to improve efficiency

Let's start from <u>a simple example</u> and move to <u>complex ones</u>.

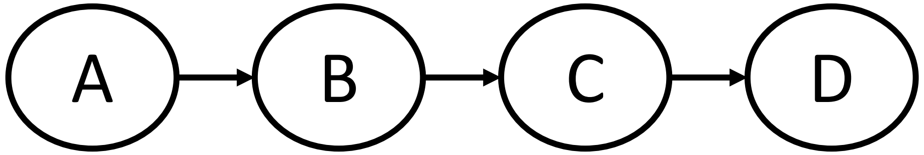
Example - Try to Compute Some Probability

$$A \rightarrow B \rightarrow C \rightarrow D$$

- **Goal: Compute $P(D)$**

Seems very easy!

Example - Try to Compute Some Probability



$$P(D) = \overset{\text{C=True}}{P(D|c)P(c)} + \overset{\text{C=False}}{P(D|\bar{c})P(\bar{c})}$$

written as

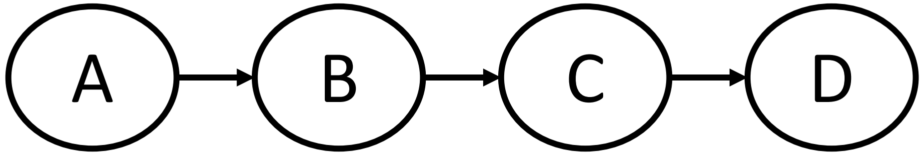
$$= \sum_C P(D|C)P(C)$$

$$= \sum_C P(D|C) \sum_B P(C|B)P(B)$$

$$= \sum_C P(D|C) \sum_B P(C|B) \sum_A P(B|A)P(A)$$

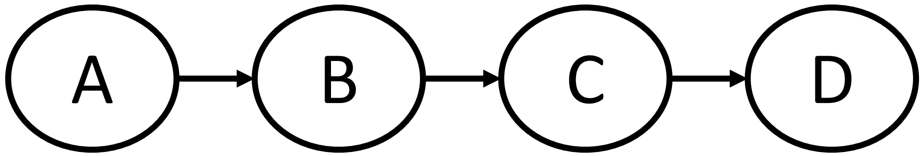
similarly

Example - Try to Compute Some Probability



$$
\begin{aligned}
P(D) &= P(D|c)P(c) + P(D|\bar{c})P(\bar{c}) \\
&= \sum_{C} P(D|C)P(C) \\
&= \sum_{C} P(D|C) \sum_{B} P(C|B)P(B) \\
&= \sum_{C} P(D|C) \sum_{B} P(C|B) \sum_{A} P(B|A)P(A) \\
&\boxed{= \sum_{C} \sum_{B} \sum_{A} P(A,B,C,D)}
\end{aligned}
$$

Example - Try to Compute Some Probability

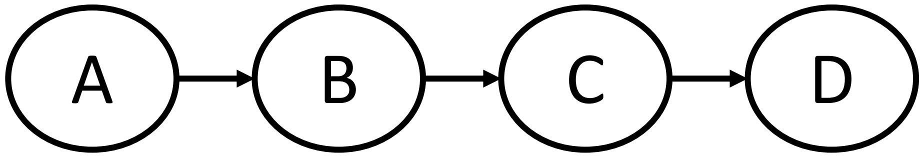


- Goal: Compute $P(D)$

$$P(D) = \sum_C \sum_B \sum_A P(A, B, C, D)$$

Sum out extra variables

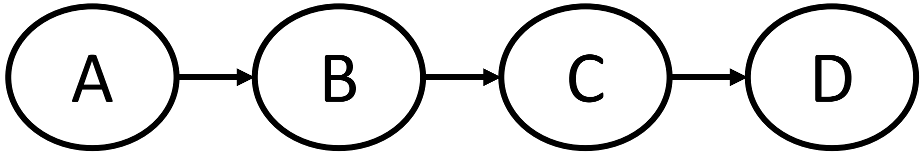
Example - Try to Compute Some Probability



- What if we want to compute $P(C)$? Does this equation hold?

$$P(C) = \sum_{D} \sum_{B} \sum_{A} P(A, B, C, D)$$

Variable Elimination



- It's not efficient to P(A,B,C,D) for all possibilities of (A,B,C,D) ! (Why?)
- In practice, we first write out $\sum_C \sum_B \sum_A P(A,B,C,D)$ and then **push in the summations** as follows

$$P(D) = \sum_C P(D|C) \sum_B P(C|B) \sum_A P(B|A)P(A)$$

How to efficiently compute it???

Summing Out a Variable (Factor Marginalization)

- $\boldsymbol{X}$: a set of variables
- Y: one variable. $Y \notin X$
- $\phi(\boldsymbol{X}, Y)$: a factor
  - $\phi: Val(\boldsymbol{X}) \longmapsto \mathbb{R}$
  - $\mathrm{Scope}(\phi) = \{\boldsymbol{X}, Y\}$

- **Sum out** of Y in ψ (marginalize Y in ϕ):

$$\psi(\boldsymbol{X}) = \sum_{Y} \phi(\boldsymbol{X}, Y)$$

The result is a new factor without Y.

Factors

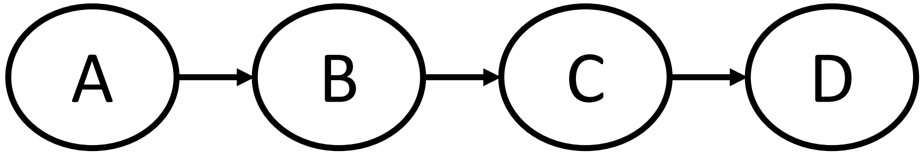
$$A \rightarrow B \rightarrow C \rightarrow D$$

$$P(D) = \sum_{C} P(D|C) \sum_{B} P(C|B) \sum_{A} \underbrace{P(B|A)}_{\phi_2(A,B)} \underbrace{P(A)}_{\phi_1(A)}$$

A	$\phi_1(A)$
True	0.4
False	0.6

A	B	$\phi_2(A,B)$
True	True	0.3
True	False	0.7
False	True	0.5
False	False	0.5

Factor Multiplication



A	$\phi_1(A)$
True	0.4
False	0.6

$$P(D) = \sum_C P(D|C) \sum_B P(C|B) \sum_A \boxed{\underbrace{P(B|A)}_{\phi_2(A,B)} \underbrace{P(A)}_{\phi_1(A)}}$$

A	B	$\phi_2(A,B)$
True	True	0.3
True	False	0.7
False	True	0.2
False	False	0.8

Factor Multiplication

⇒ intermediate result $\varphi_1(A,B)$

A	B	$\varphi_1(A,B)$
True	True	0.4*0.3=0.12
True	False	0.4*0.7=0.28
False	True	0.6*0.2=0.12
False	False	0.6*0.8=0.48

Summing Out a Variable

$$A \to B \to C \to D$$

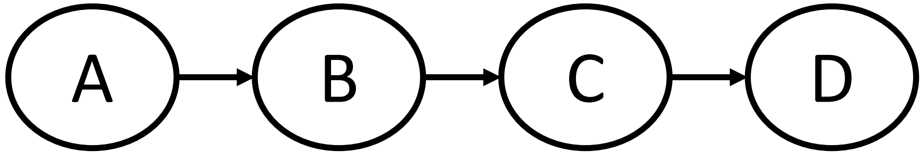
$$P(D) = \sum_C P(D|C) \sum_B P(C|B) \boxed{\sum_A \underbrace{P(B|A)P(A)}_{\varphi_1(A,B)}}$$

$$\Rightarrow \psi_1(B)$$

A	B	$\phi_2(A,B)$
True	True	0.3
True	False	0.7
False	True	0.2
False	False	0.8

A	$\phi_1(A)$
True	0.4
False	0.6

Summing Out a Variable



Summing out A

$$P(D) = \sum_C P(D|C) \sum_B P(C|B) \sum_A \underbrace{P(B|A)P(A)}_{\varphi_1(A,B)}$$

$\Rightarrow$ New factor $\psi_1(B)$

Intermediate Result $\varphi_1(A,B)$

A	**B**	$\varphi_1(A,B)$
True	True	0.4*0.3=0.12
True	False	0.4*0.7=0.28
False	True	0.6*0.2=0.12
False	False	0.6*0.8=0.48

new factor without *A*

B	$\psi_1(B)$
True	0.12+0.12=0.24
False	0.28+0.48=0.76

Variable Elimination

$$A \rightarrow B \rightarrow C \rightarrow D$$

$$P(D) = \sum_{C} \underbrace{P(D|C)}_{\phi_4(C,D)} \sum_{B} \underbrace{P(C|B)}_{\phi_3(B,C)} \sum_{A} \underbrace{P(B|A)}_{\phi_2(A,B)} \underbrace{P(A)}_{\phi_1(A)}$$

$$\Rightarrow \psi_1(B)$$

$$\Rightarrow \psi_2(C)$$

$$\Rightarrow \psi_3(D)$$

Result!

Entropy and Information

- Definition: Entropy If X is a discrete random variable and f (x) is the value of its probability distribution at x, then the entropy of X is:

$$H(X) = -\sum_{x \in X} f(x) \log_2 f(x)$$

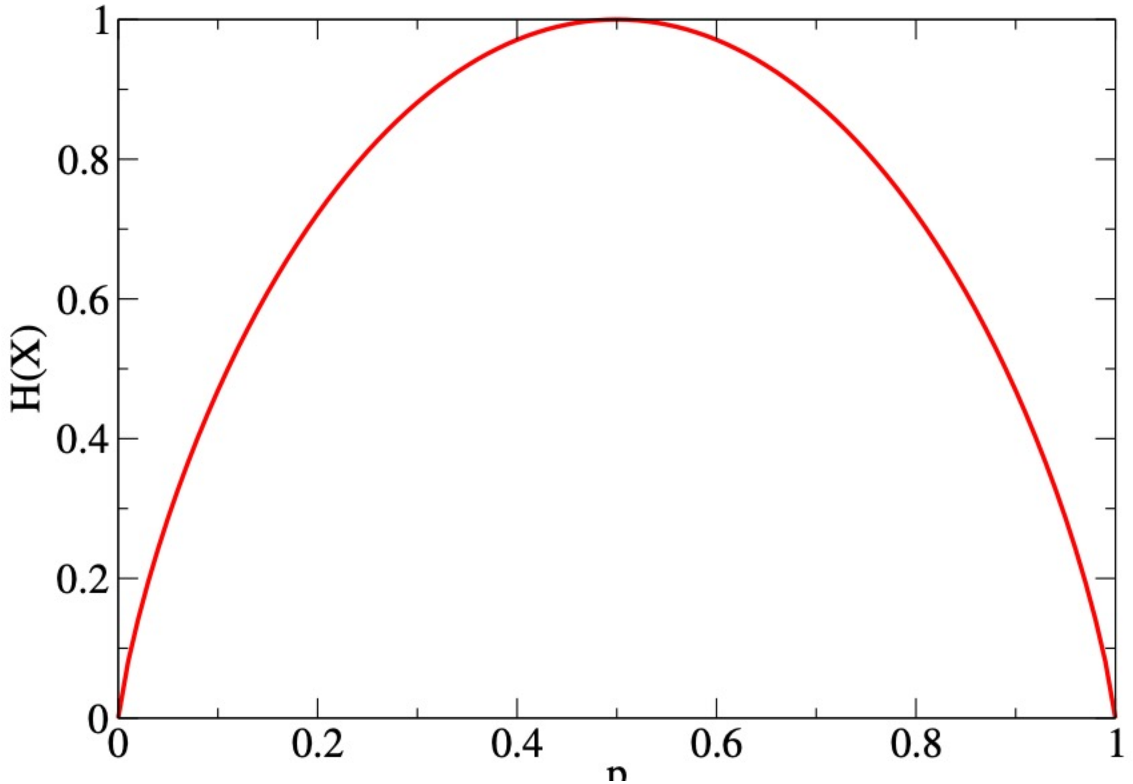
- Entropy measures amount of information (or uncertainty) in random variable;
- note that H(X) ≥ 0 by definition.

Properties of Entropy

- If X is a binary random variable with the distribution f(0)=p and f(1)=1−p, then:
  - H(X) = 0 if p = 0 or p = 1
  - max H(X) for p = ½
- Intuitively, an entropy of 0 means that the outcome of the random variable is determinate; it contains no information (or uncertainty).
- If both outcomes are equally likely (p = 1/2), then we have maximal uncertainty.

Properties of Entropy

- Visualize the content of the previous theorem:



Joint Entropy

- If X and Y are discrete random variables and f (x, y) is the value of their joint probability distribution at (x, y), then the joint entropy of X and Y is:

$$H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} f(x, y) \log f(x, y)$$

- The joint entropy represents the amount of information needed on average to specify the value of two discrete random variables.

Conditional Entropy

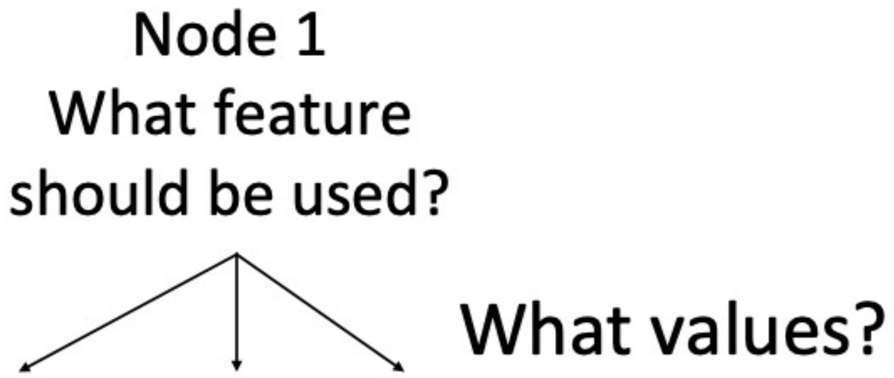
- If X and Y are discrete random variables and f (x, y) and f (y|x) are the values of their joint and conditional probability distributions, the conditional entropy of Y given X is:

$$H(Y|X) = -\sum_{x \in X} \sum_{y \in Y} f(x, y) \log f(y|x)$$

- The conditional entropy indicates how much extra information you still need to supply on average to communicate Y given that the other party knows X.

Entropy-Based Decision Tree Construction

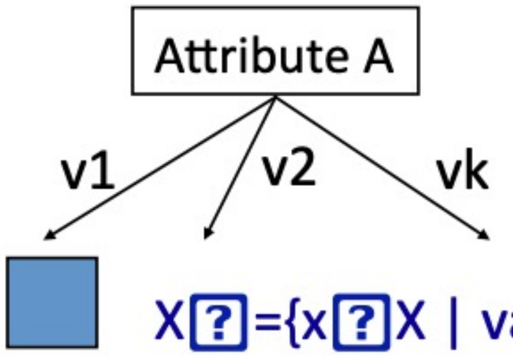
Training Set X
x1=(f11,f12,...f1m)
x2=(f21,f22, f2m)
.
.
xn=(fn1,f22, f2m)



Entropy-Based Decision Tree Construction

Choose the attribute A with the smallest expected entropy for the full training set at the root of the tree as the most discriminating attribute.

Full Training Set X



Construct child nodes for each value of A. Each has an associated subset of vectors in which A has a particular value.

Set X ?

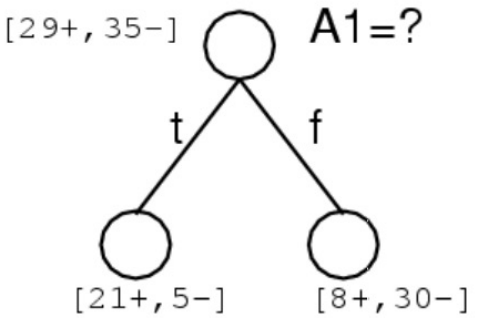
X?={x?X | value(A)=v1}

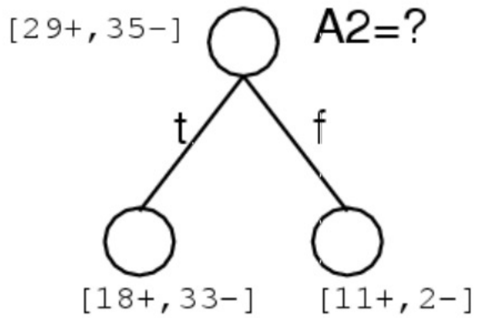
repeat recursively till when?

Information Gain

- Information Gain is the mutual information between input attribute A and target variable Y
- Information Gain is the expected reduction in entropy of target variable Y for data sample S, due to sorting on variable A

$$Gain(S, A) = I_S(A, Y) = H_S(Y) - H_S(Y|A)$$

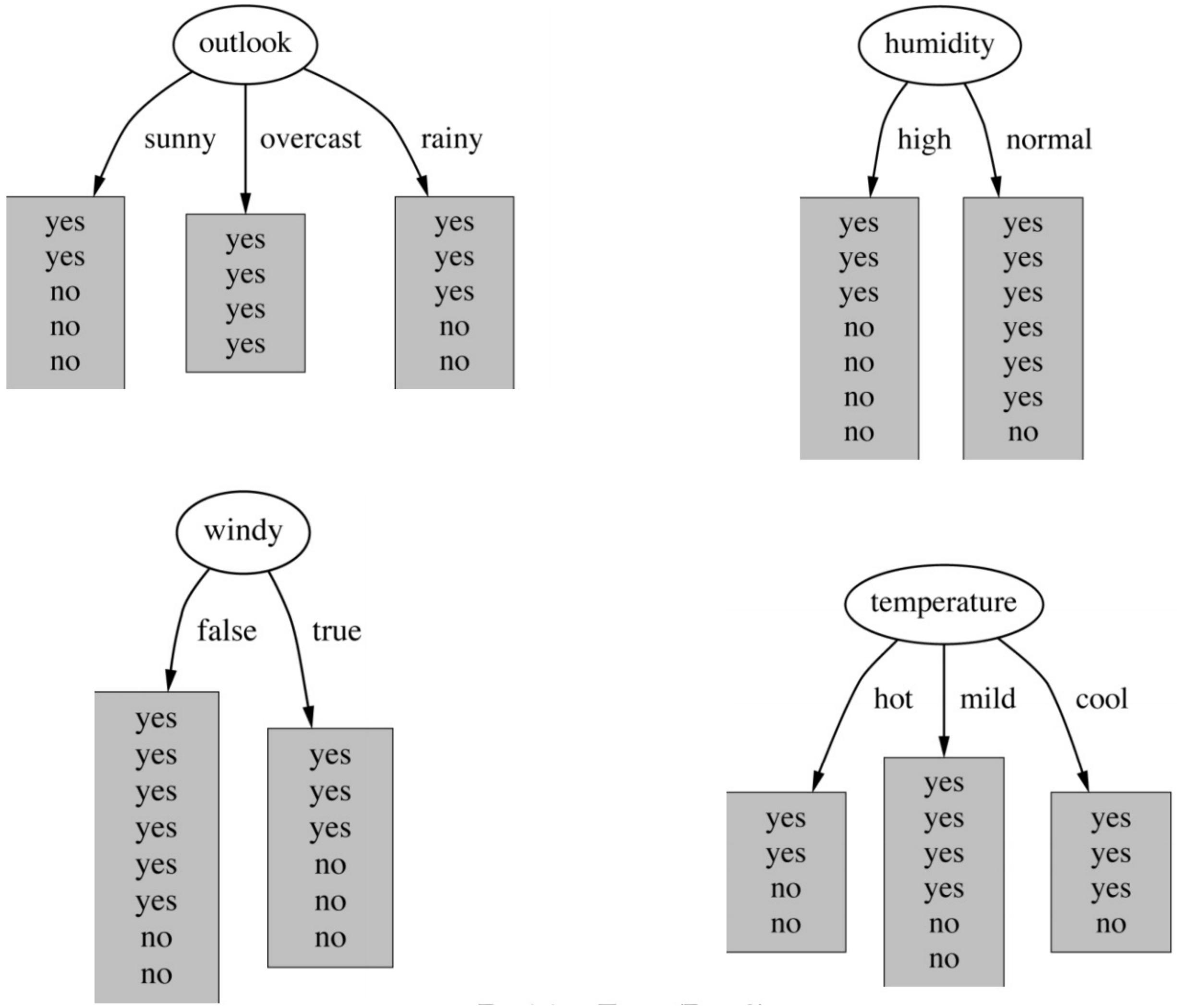




Example

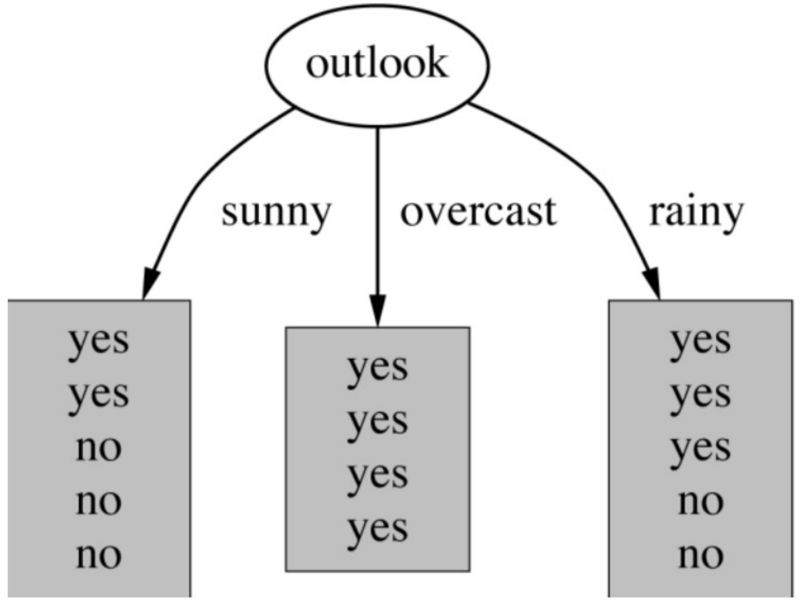
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Which attribute to select?

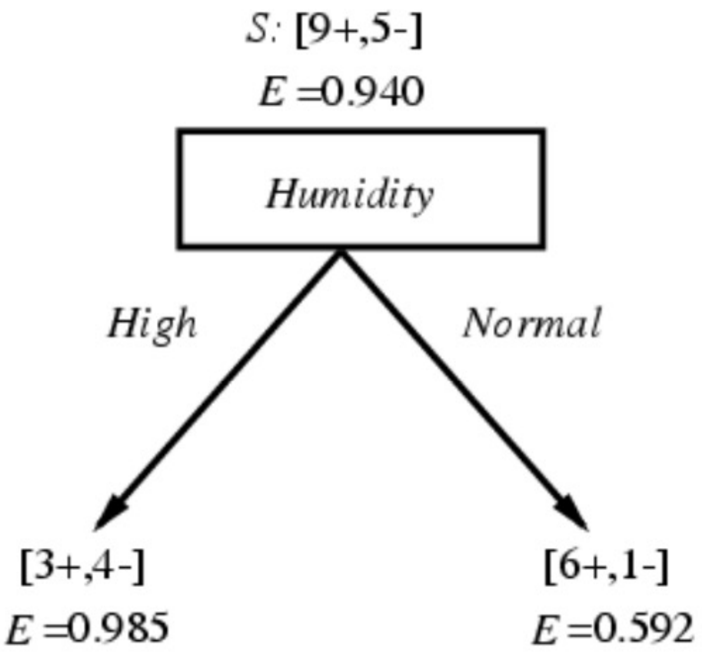


Find the smallest expected entropy

- EE(outlook) =
- $-((2/5 \cdot \log2(2/5) + 3/5 \cdot \log2(3/5)) \cdot 5/14$
  $-((4/4 \cdot \log2(4/4) + 0/4 \cdot \log2(0/4)) \cdot 4/14$
  $-((3/5 \cdot \log2(3/5) + 2/5 \cdot \log2(2/5)) \cdot 5/14$
  $=0.693$
- Information Gain = 0.940-0.693 = 0.247
- The smallest expected entropy => the highest information gain

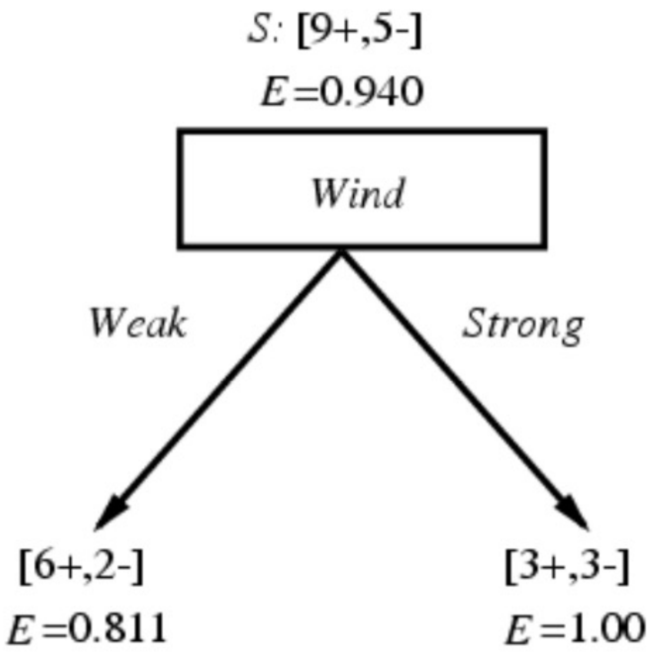


Find the smallest expected entropy



Gain (S, Humidity)

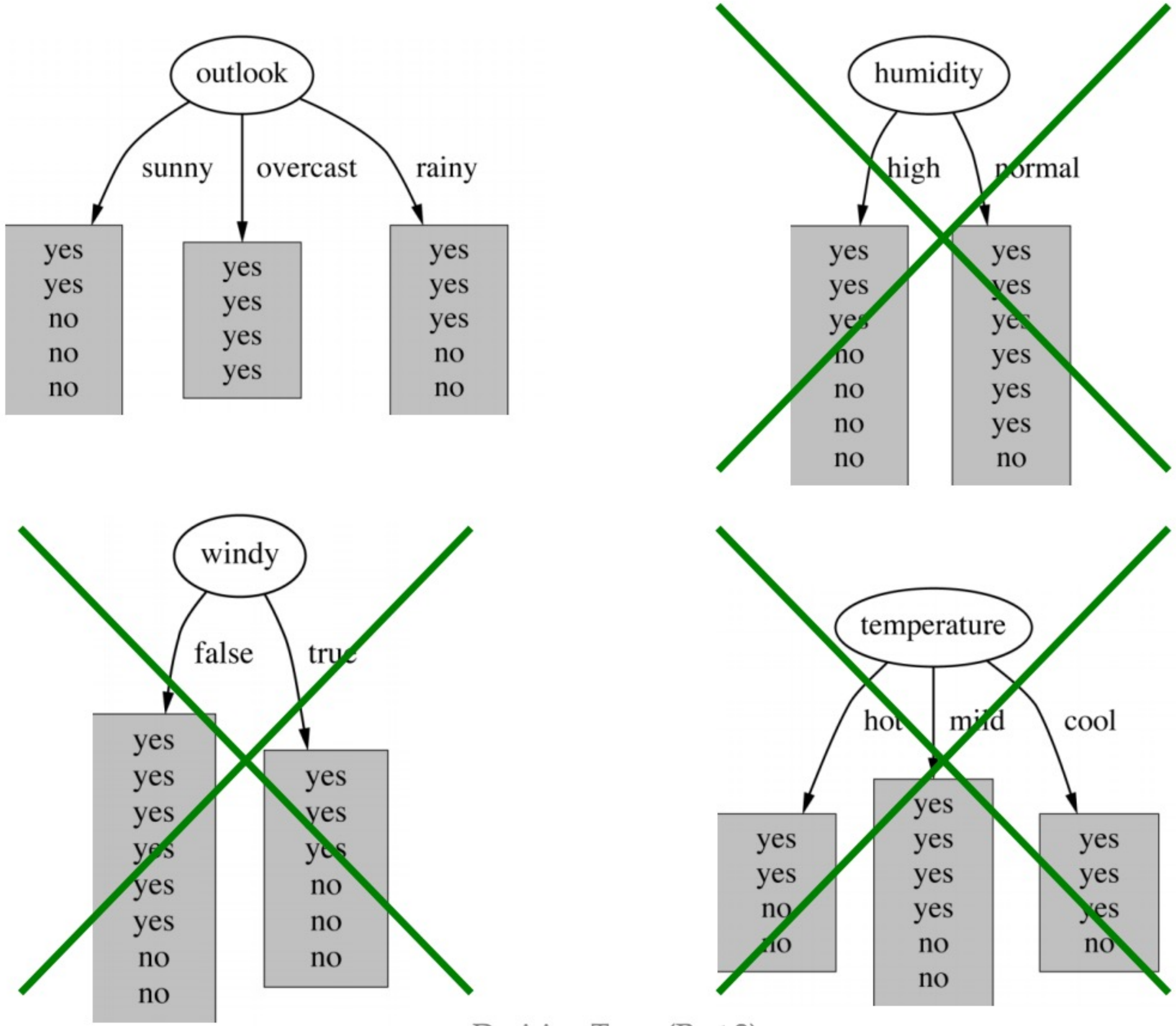
= .940 - (7/14).985 - (7/14).592
= .151



Gain (S, Wind)

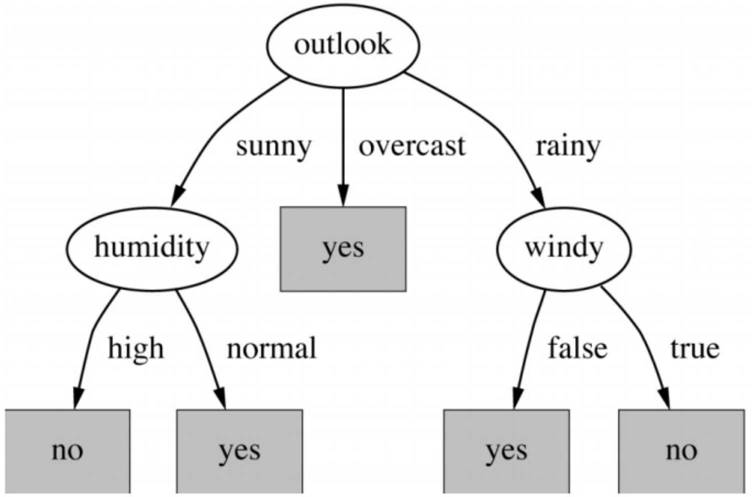
= .940 - (8/14).811 - (6/14)1.0
= .048

Which attribute to select?



Final decision tree

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



⇒ Splitting stops when data can’t be split any further

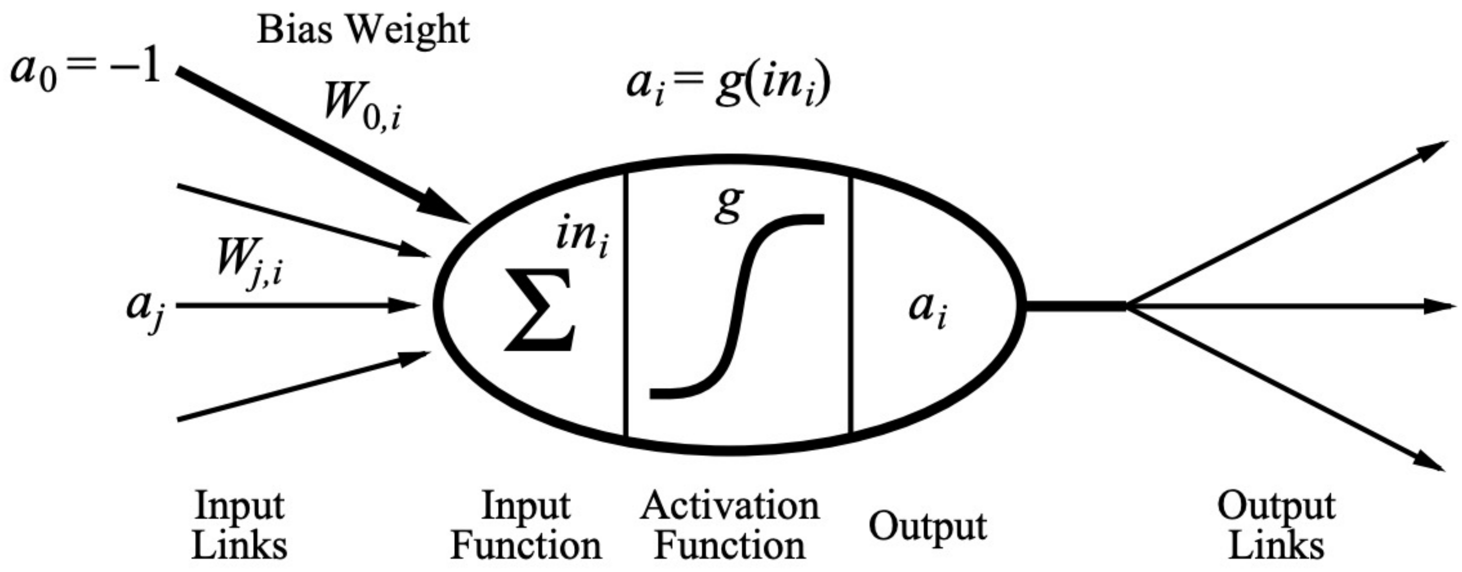
Neural Network

- Origins: Algorithms that try to mimic the brain.
- Very widely used in 80s and early 90s; popularity diminished in late 90s.
- Recent resurgence: State-of-the-art technique for many applications.
- Artificial neural networks are not nearly as complex or intricate as the actual brain structure.

Neurons

- Neuron is the basic part in the NN
- Output is a “squashed” linear function of the inputs:

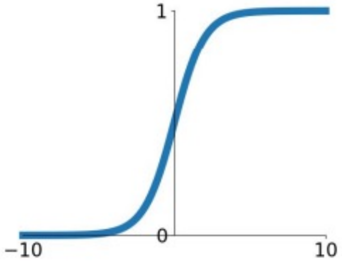
$$a_i \leftarrow g(in_i) = g\left(\Sigma_j W_{j,i} a_j\right)$$



Activation functions

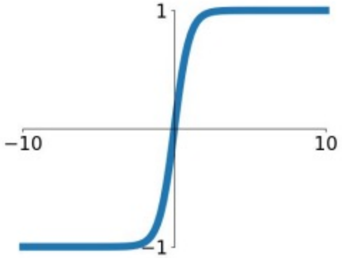
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



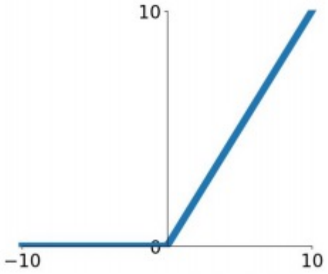
tanh

$\tanh(x)$



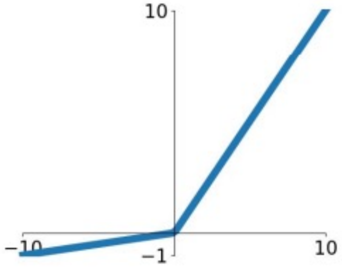
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

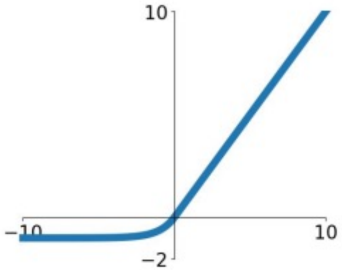


Maxout

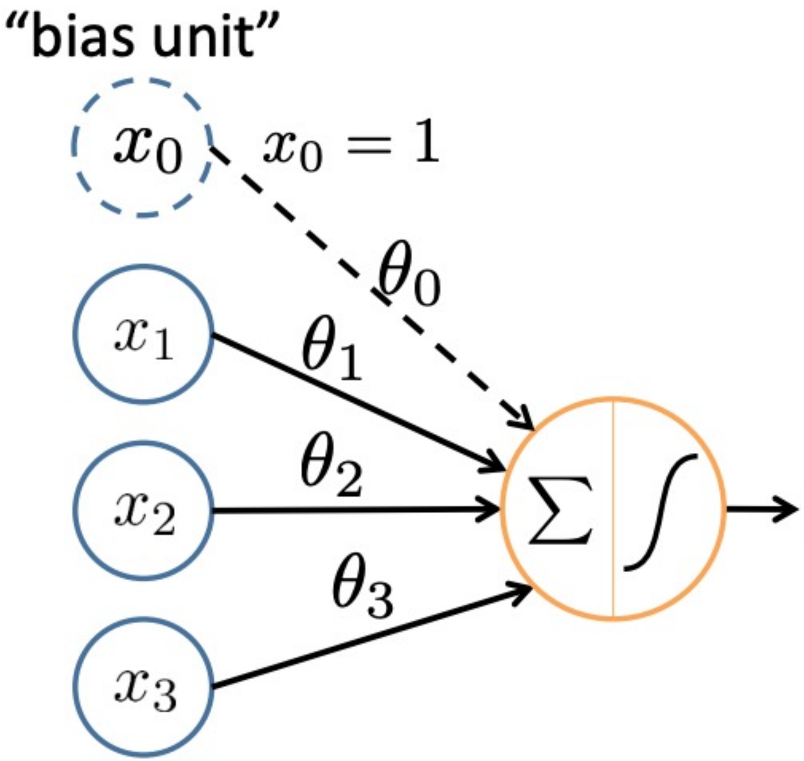
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Example

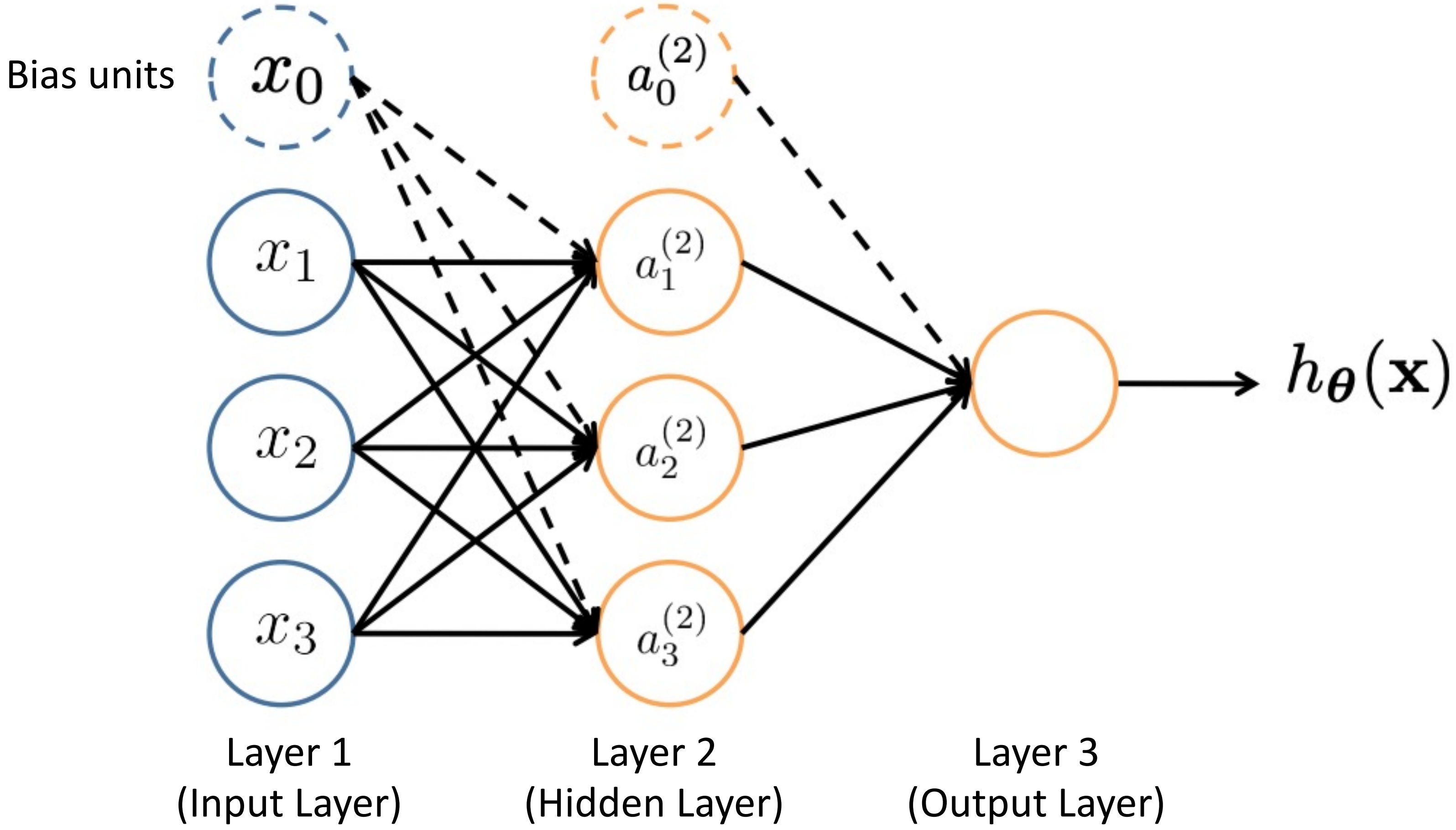


$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad \boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g\left(\boldsymbol{\theta}^\mathsf{T}\mathbf{x}\right) = \frac{1}{1 + e^{-\boldsymbol{\theta}^\mathsf{T}\mathbf{x}}}$$

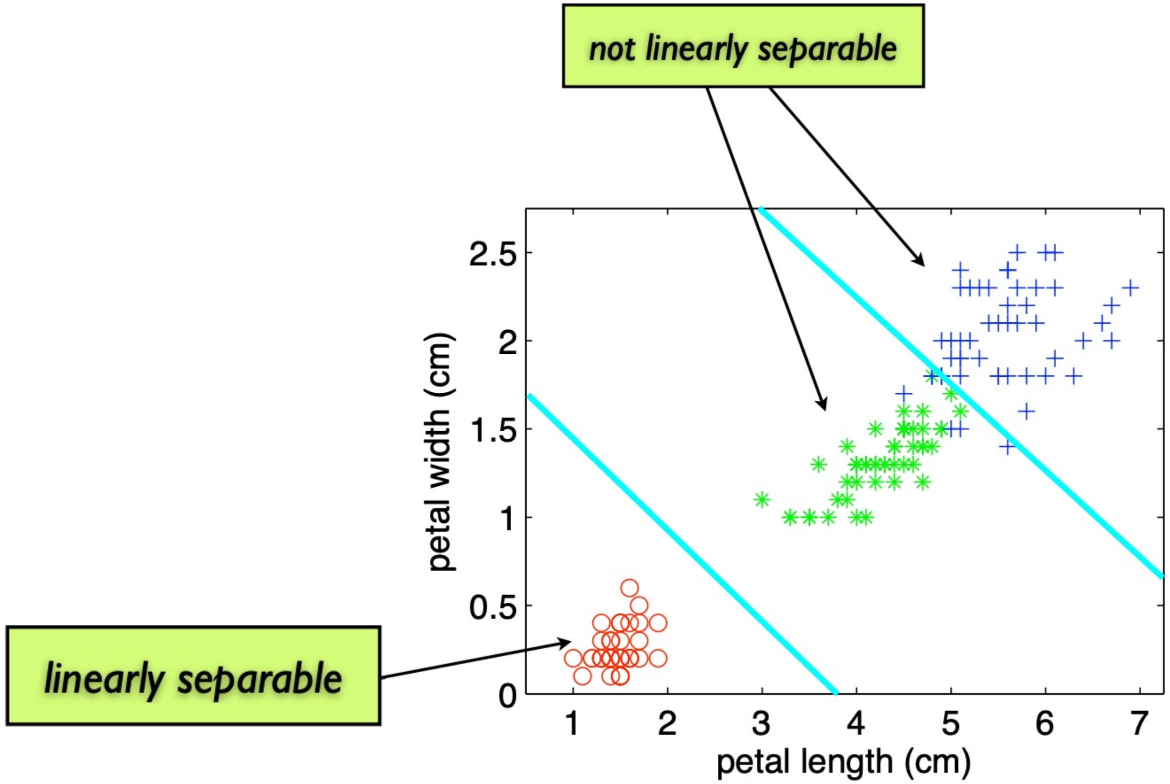
Sigmoid (logistic) activation function: $g(z) = \dfrac{1}{1 + e^{-z}}$

Neural Network



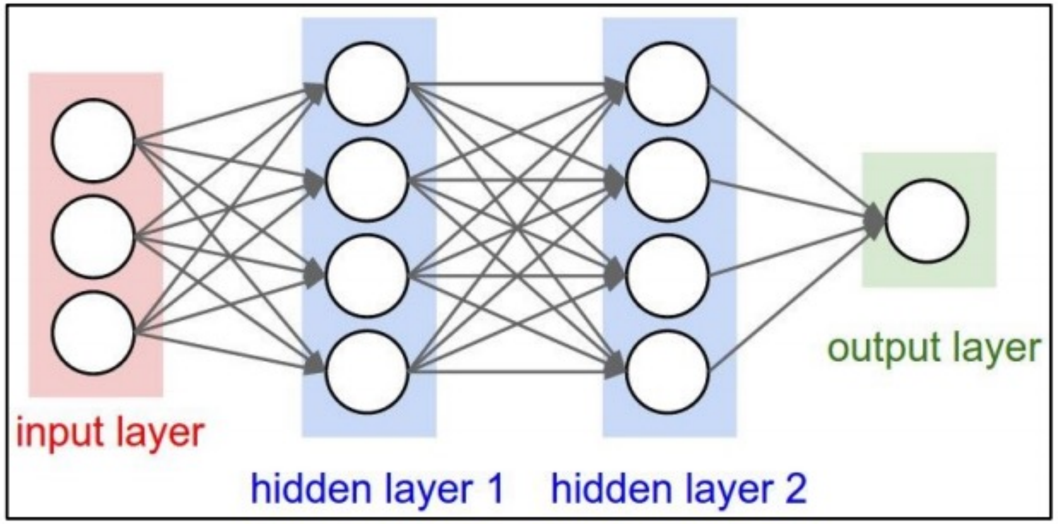
Linear separability

- Two classes are linearly separable if they can be separated by a linear combination of attributes
  - 1D: threshold
  - 2D: line
  - 3D: plane
  - M-D: hyperplane



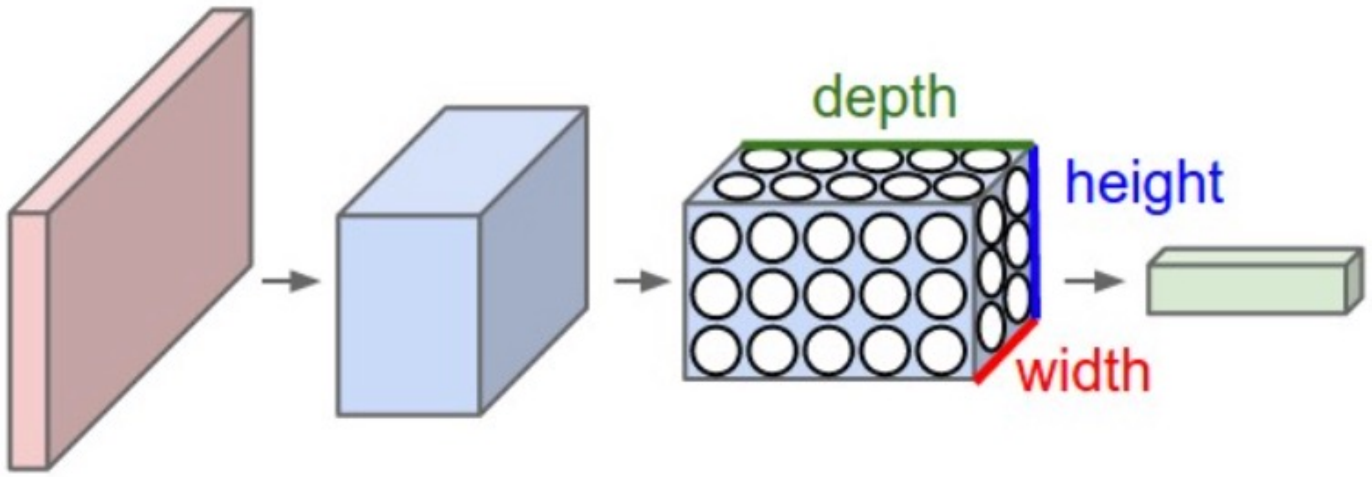
Mini-batch SGD

- Loop:
  1. Sample a batch of data
  2. Forward prop it through the graph, get loss
  3. Backprop to calculate the gradients
  4. Update the parameters using the gradient

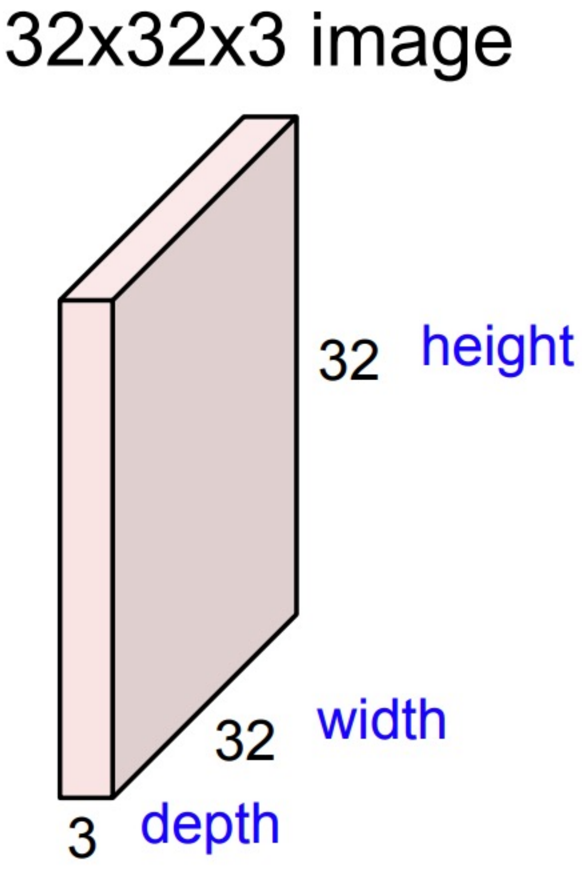


Convolutional Neural Networks

- *3D volumes of neurons*. ConvNet have neurons arranged in 3 dimensions: **width, height, depth**.

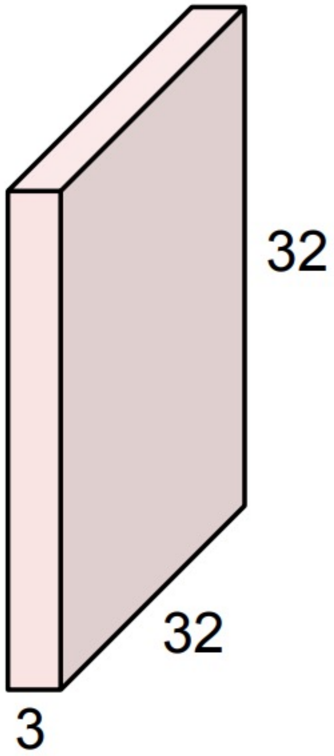


Convolution Layer

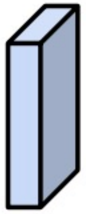


Convolution Layer

32x32x3 image



5x5x3 filter

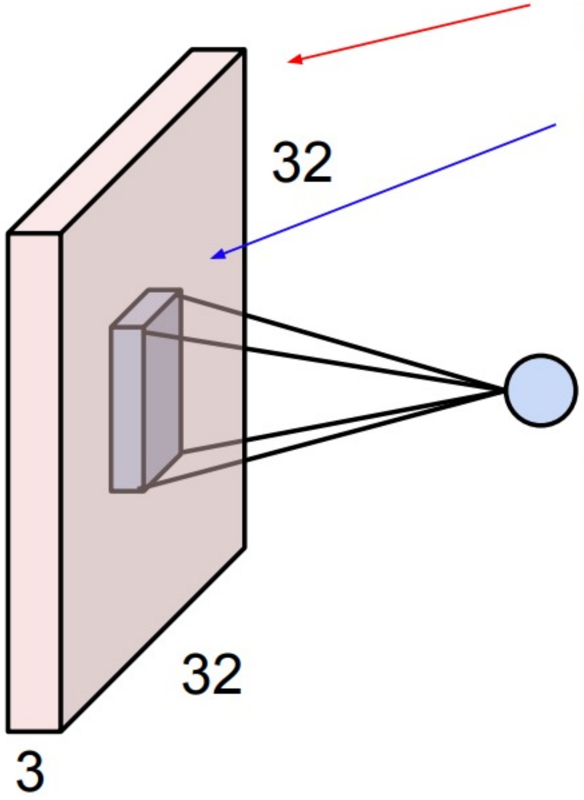


Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

Convolution Layer

32x32x3 image

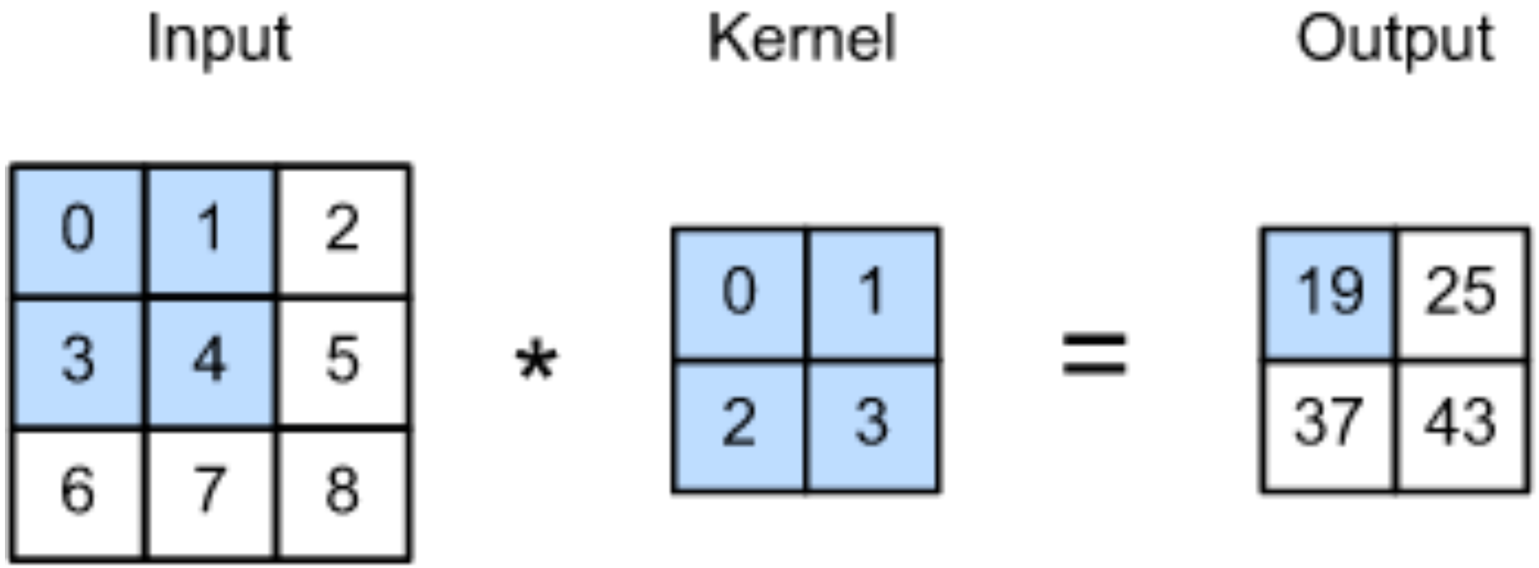
5x5x3 filter w



1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image (i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Example



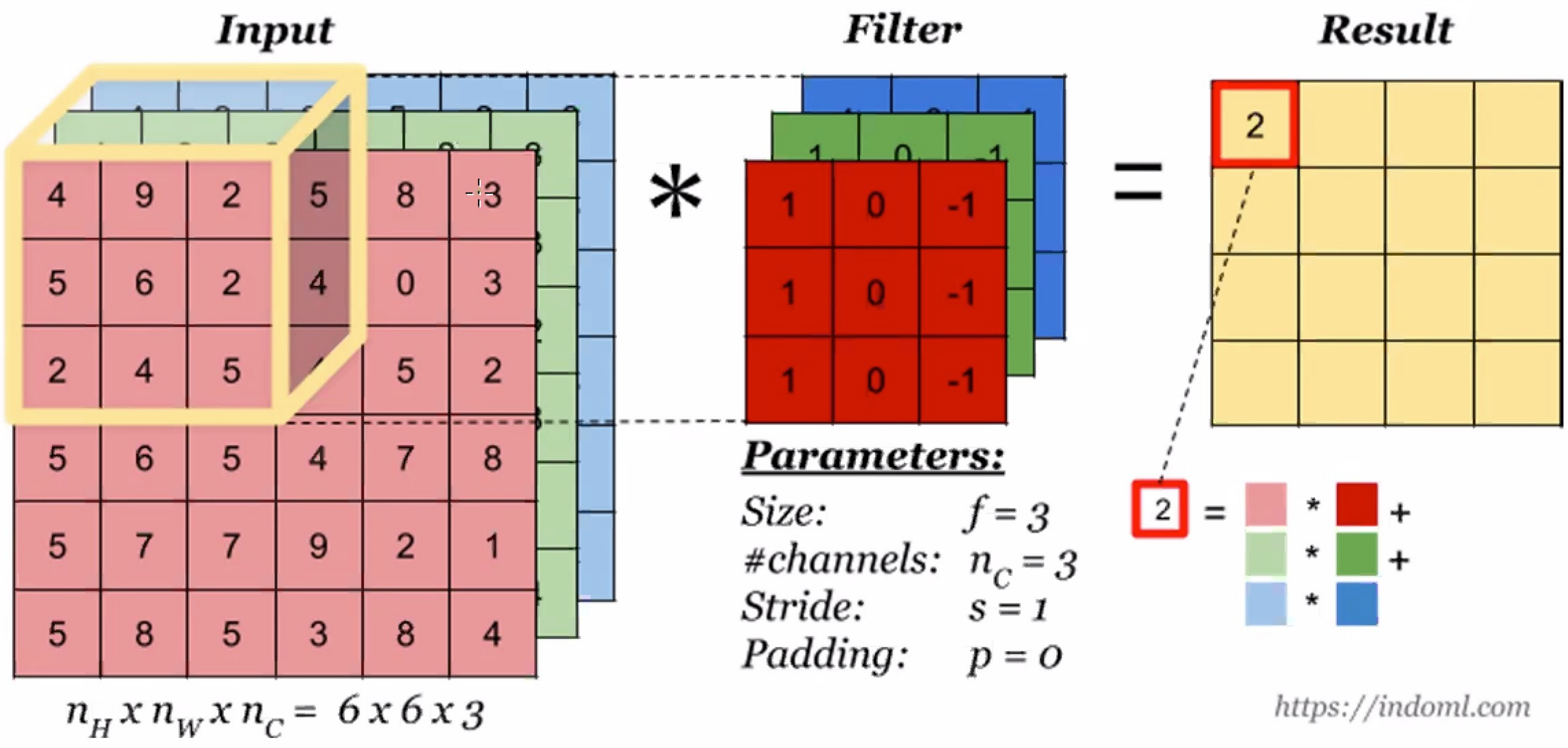
Example

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42
43	44	45	46	47	48	49

0.1	0.2	0.3
0.4	0.5	0.6
0.7	0.8	0.9

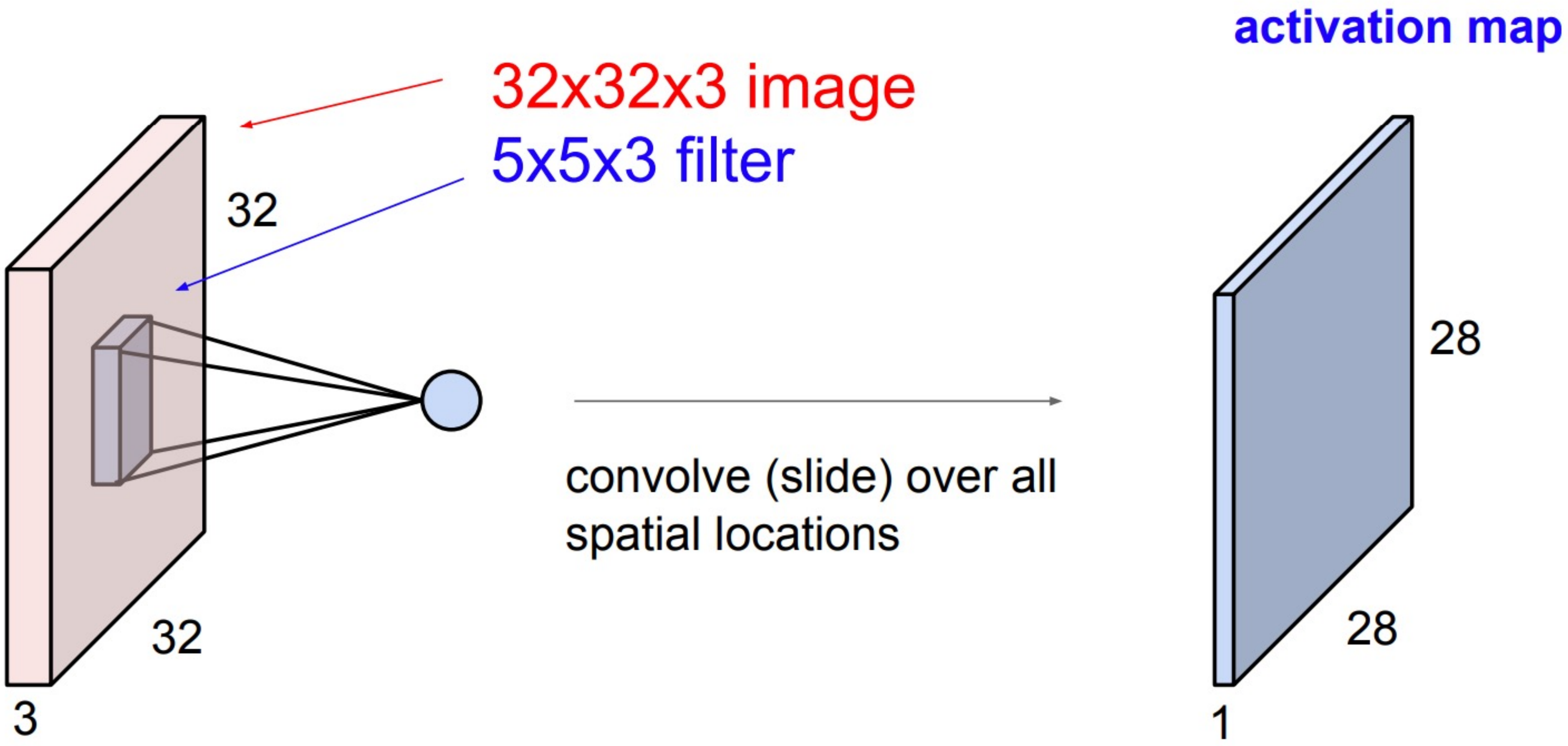
$$\begin{aligned} &= 0.1 \times 10 + 0.2 \times 11 + 0.3 \times 12 \\ &+ 0.4 \times 17 + 0.5 \times 18 + 0.6 \times 19 \\ &+ 0.7 \times 24 + 0.8 \times 25 + 0.9 \times 26 \\ &= 94.2 \end{aligned}$$

Example

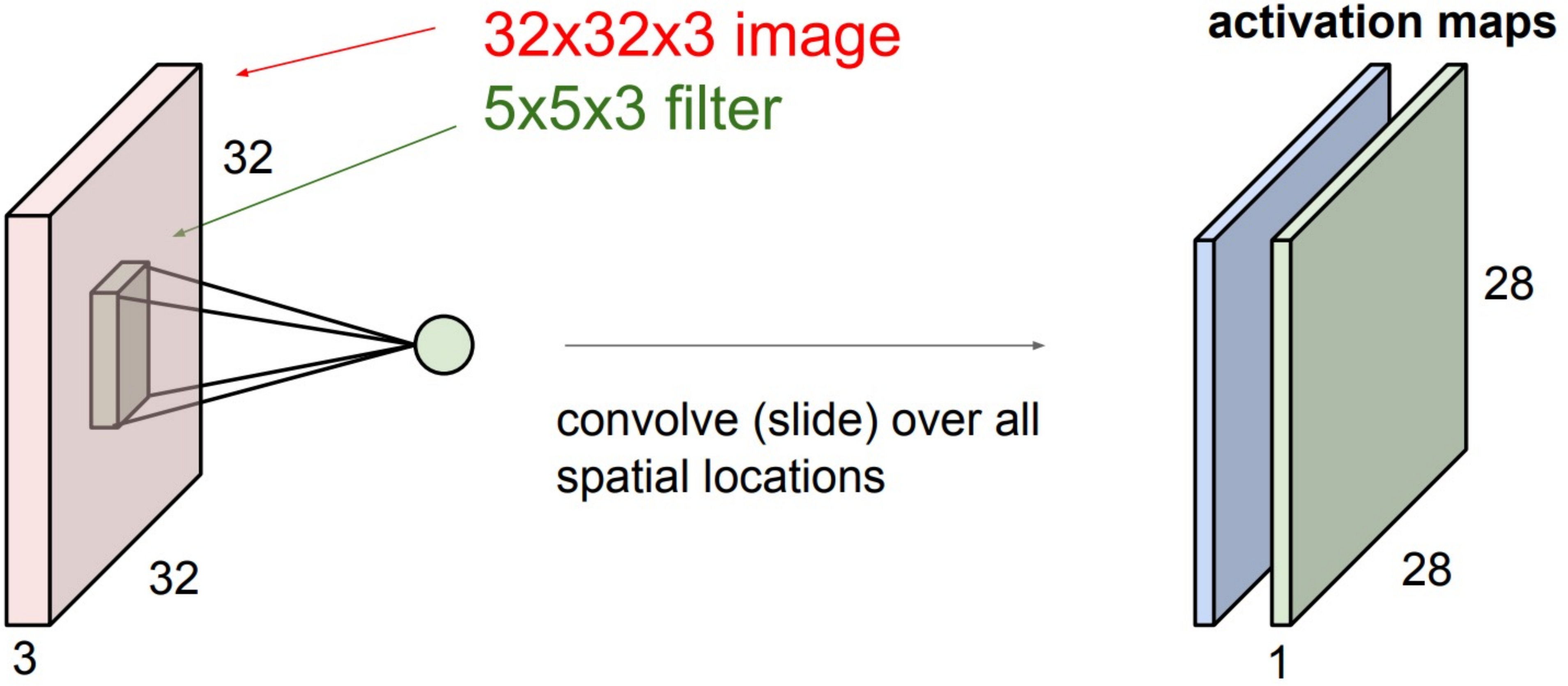


Input
Filter
Result
4 9 2 5 8 3
5 6 2 4 0 3
2 4 5 5 2
5 6 5 4 7 8
5 7 7 9 2 1
5 8 5 3 8 4
*
1 0 -1
1 0 -1
1 0 -1
=
2
$n_H \, x \, n_W \, x \, n_C = 6 \, x \, 6 \, x \, 3$
Parameters:
Size: $f = 3$
#channels: $n_C = 3$
Stride: $s = 1$
Padding: $p = 0$
2 = * + * + *
https://indoml.com

Convolution Layer



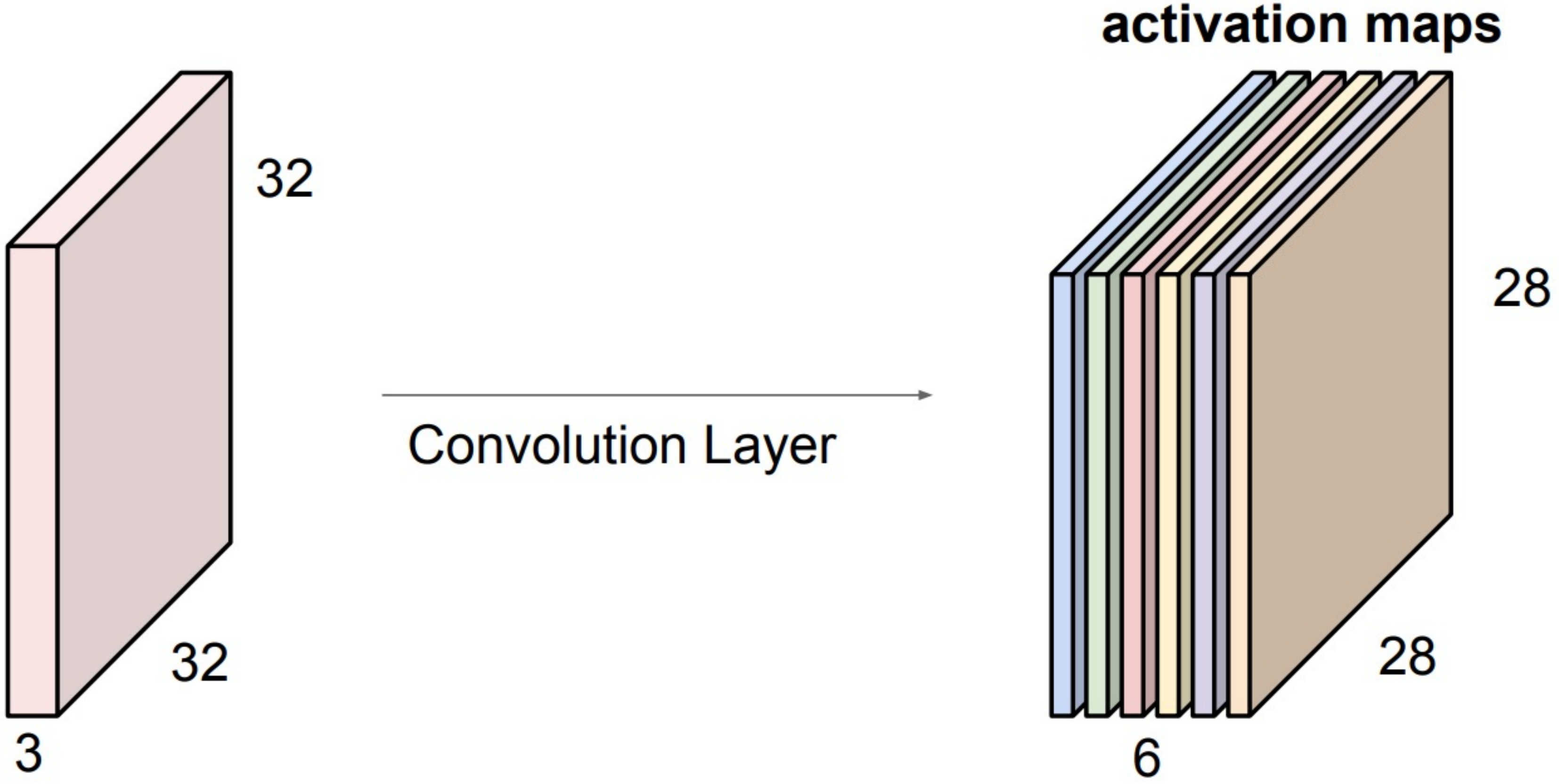
Convolution Layer



consider a second, green filter

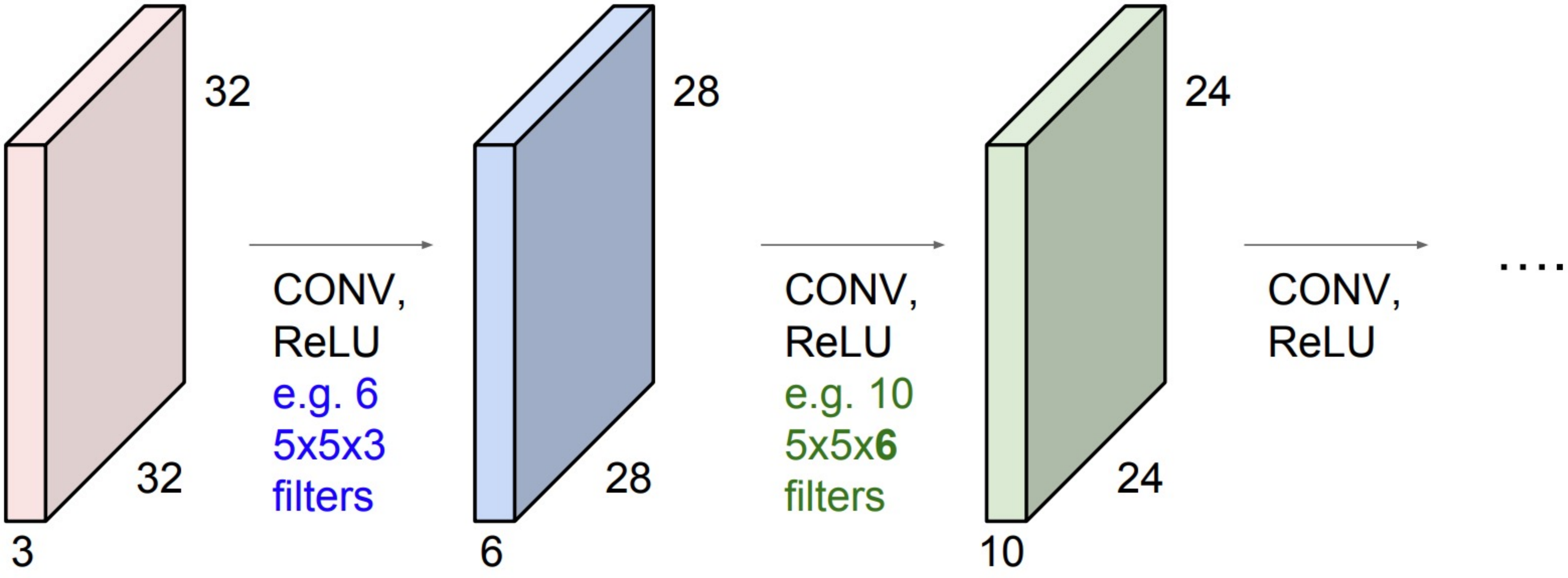
Convolution Layer

if we had 6 5x5 filters, we'll get 6 separate activation maps:



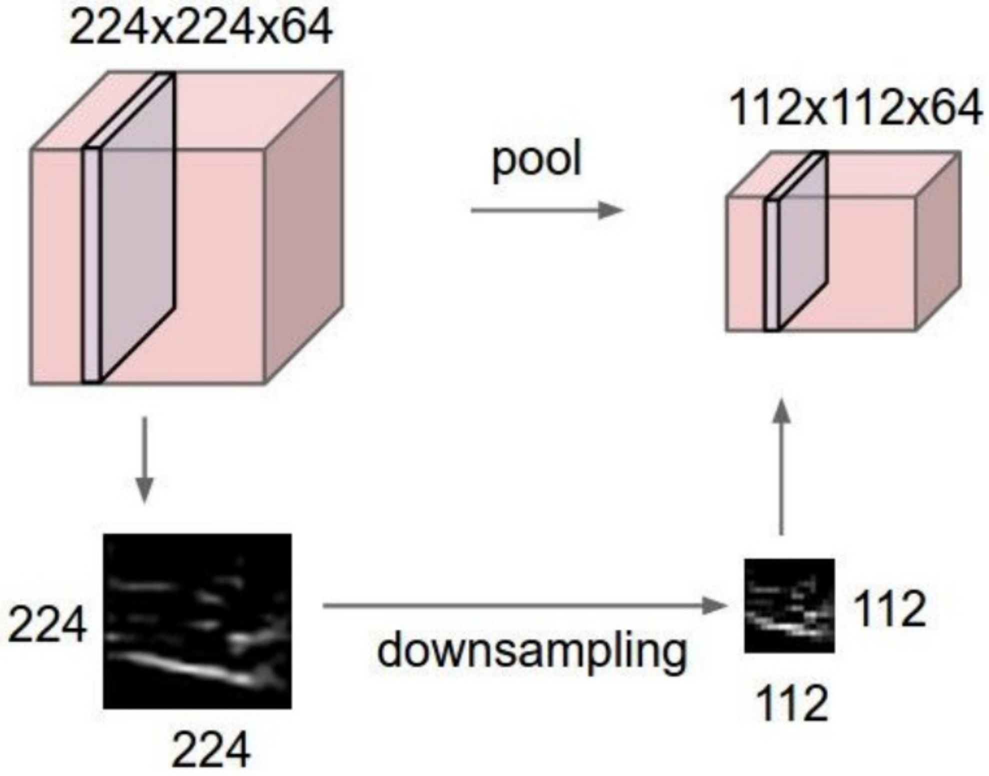
Convolution Layer

ConvNet is a sequence of Convolutional Layers, interspersed with activation functions

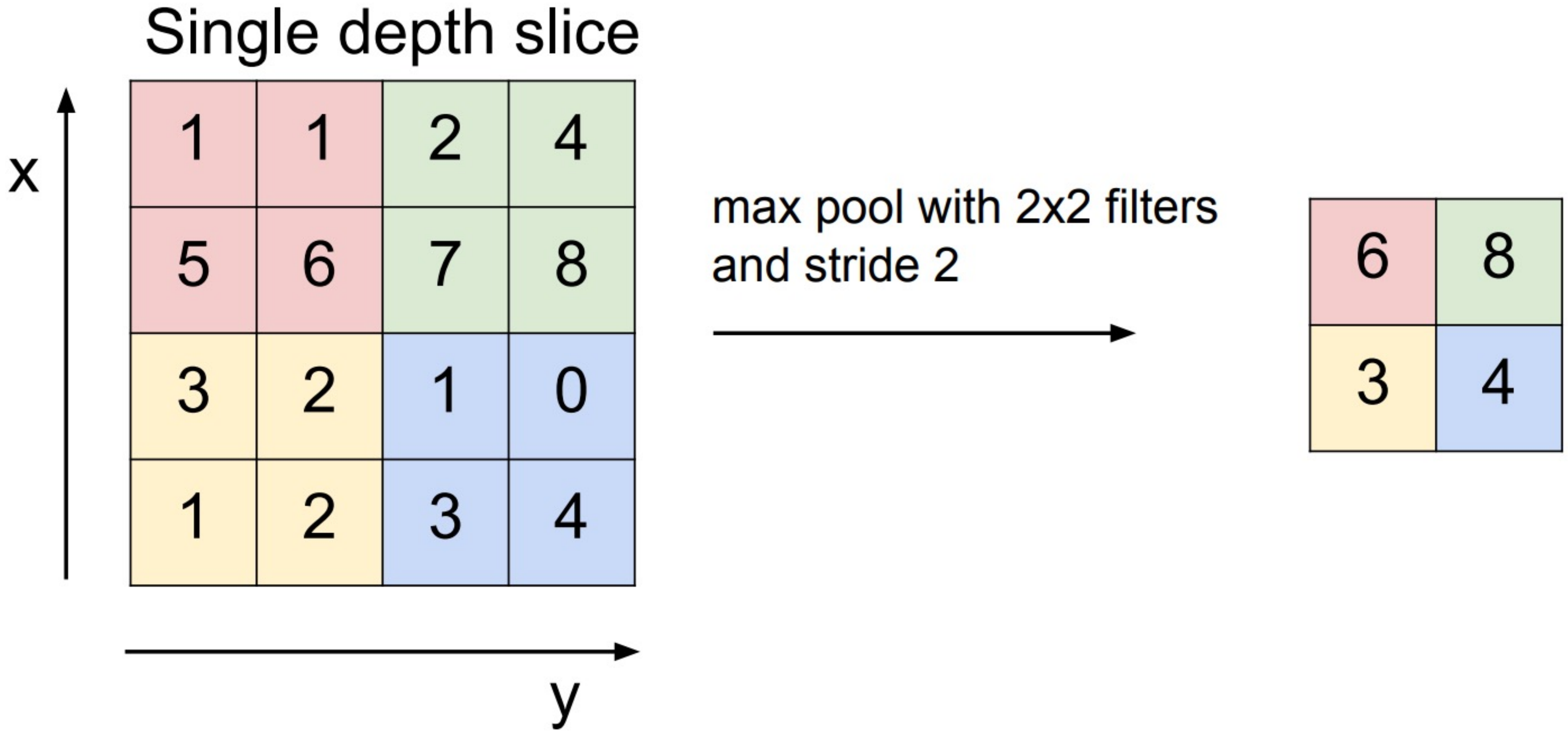


Pooling layer

- makes the representations smaller and more manageable
- operates over each activation map independently:



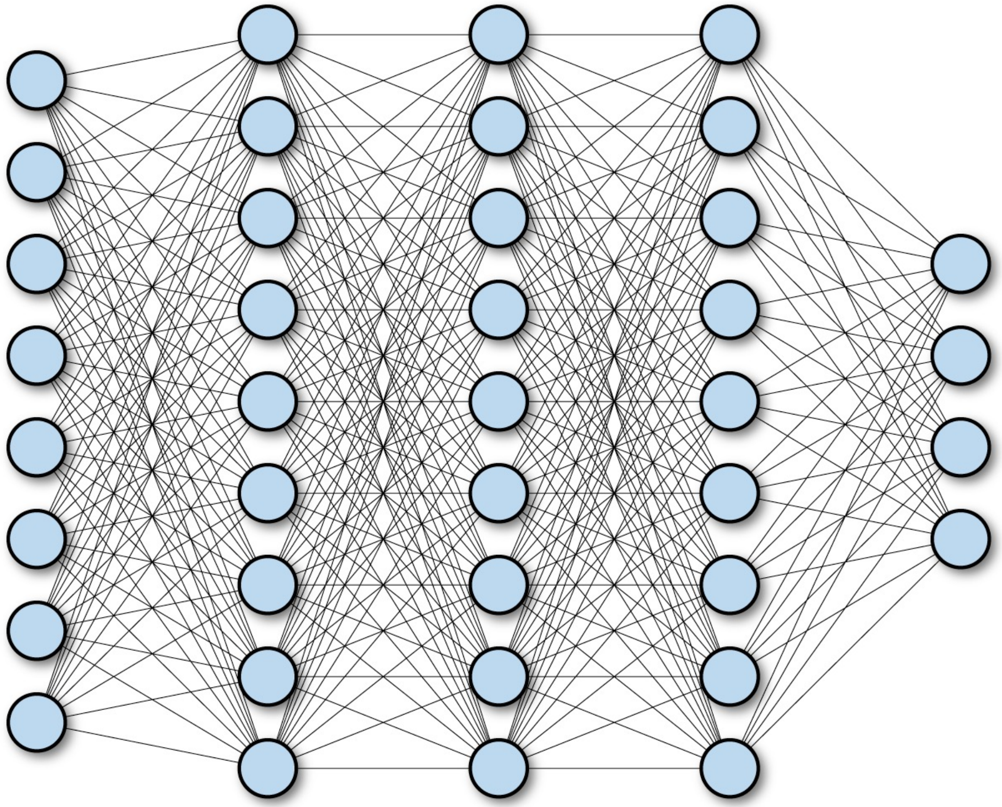
Max Pooling



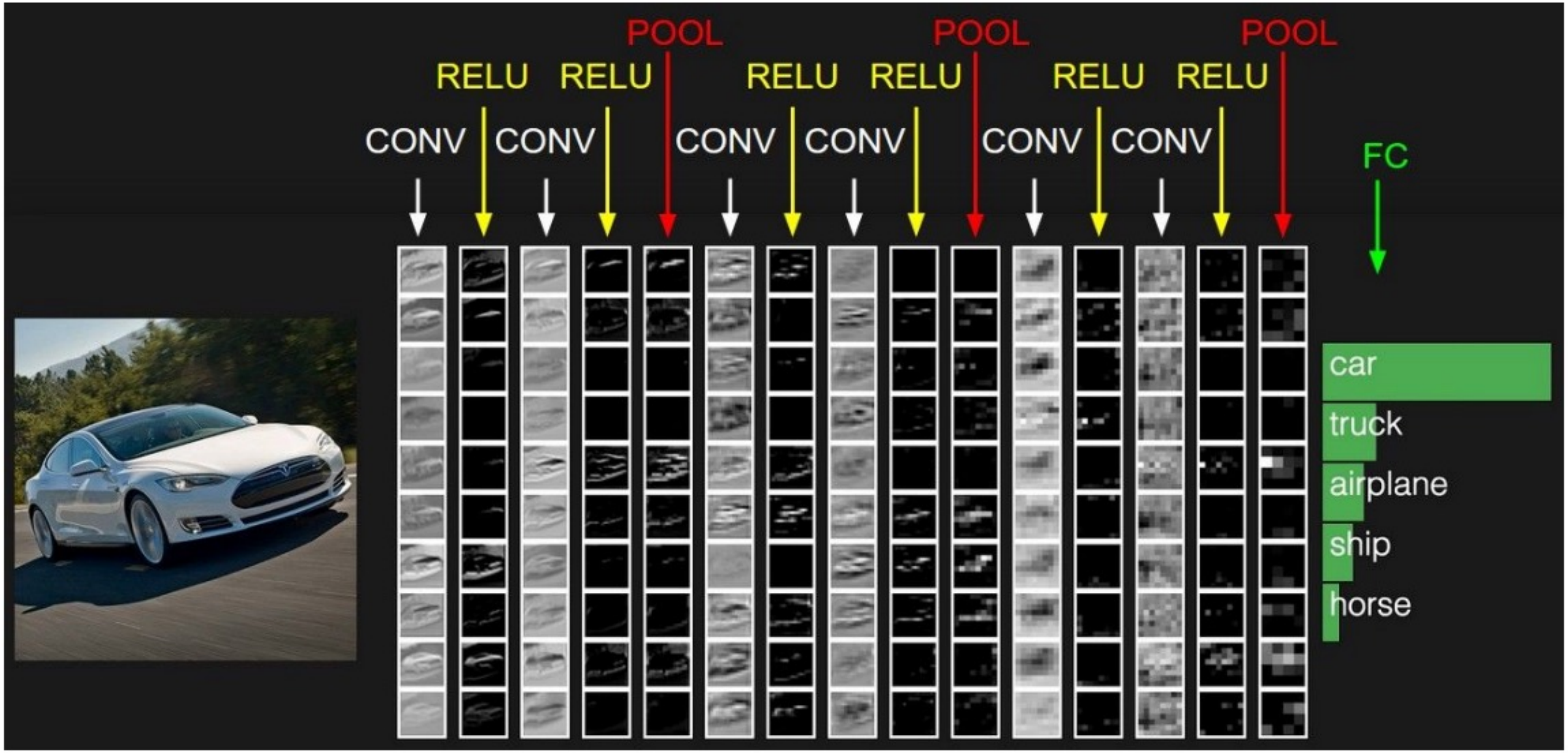
Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks



Example



POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

Summary

- ConvNets stack CONV,POOL,FC layers
- Trend towards smaller filters and deeper architectures
- Trend towards getting rid of POOL/FC layers (just CONV)
- Typical architectures look like

  [(CONV-RELU)*N-POOL?]*M-(FC-RELU)*K,SOFTMAX

  where N is usually up to ~5, M is large, 0 <= K <= 2.
  - but recent advances such as ResNet/GoogLeNet challenge this paradigm

Limitation of NN

- Hungry for data
- Brittle/lack of robustness
- Not easy to explain