Deep Learning for Natural Language Processing
Lecture 3: CNN and RNN for Sentiment Analysis

Quan Thanh Tho
Faculty of Computer Science and Engineering
Ho Chi Minh City University of Technology

Acknowledgement

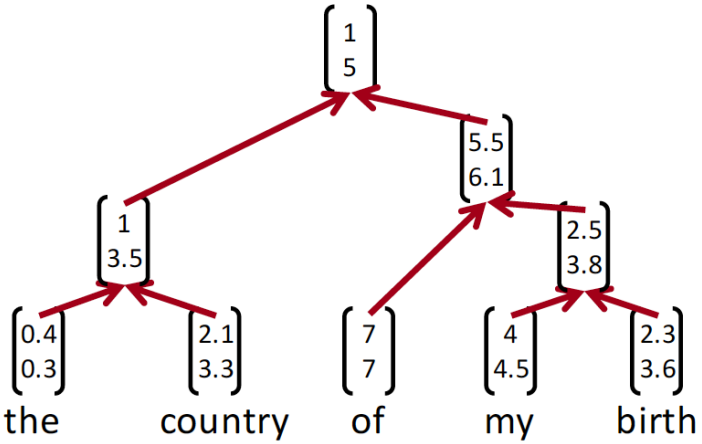
- Some slides are from Coursera course of Prof. Andrew Ng.

Agenda

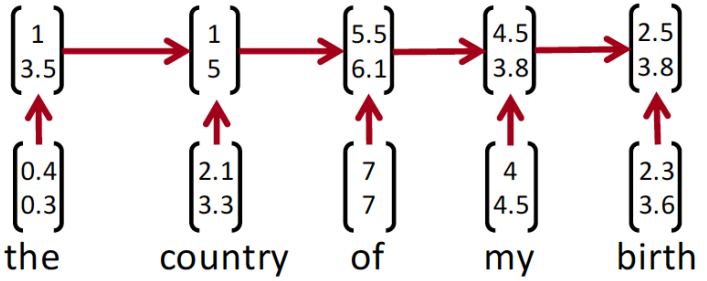
- CNN for feature (generation) and selection
- CNN for NLP
- Sentiment Analysis
- Deep Architecture: Combination of CNN and RNN for Sentiment Analysis

From RNN to CNN

- Recursive neural nets require a parser to get tree structure



- Recurrent neural nets cannot capture phrases without prefix context and often capture too much of last words in final vector

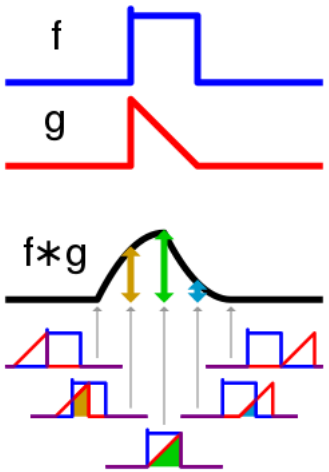


Convolution

In mathematics convolution is a mathematical operation on two functions to produce a third function that expresses how the shape of one is modified by the other. The term convolution refers to both the result function and to the process of computing it.

$$(f * g)(t) \triangleq \int_{-\infty}^{\infty} f(\tau) g(t - \tau)\, d\tau.$$

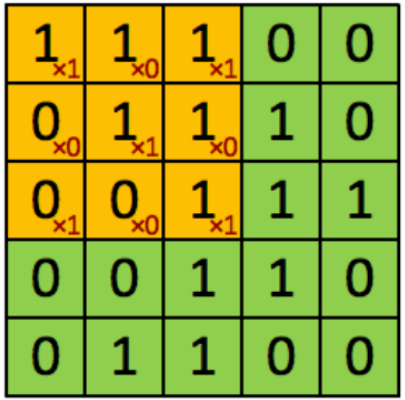
$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] g[n - m]$$



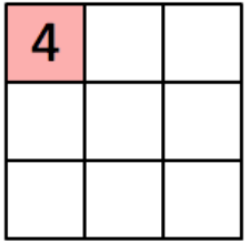
Convolution for feature extraction

- Convolution is great to extract features from images

- 2d example →
- Yellow shows filter weights
- Green shows input



Image



Convolved Feature

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

5	2	2	3	2	4	4
3	4	2	3	3	3	4
3	2	4	3	2	3	3
4	2	2	5	1	3	5
1	4	2	2	5	3	3
3	3	5	2	2	5	2
2	2	1	3	2	2	5

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

			1
		1	
	1		
1			1

1			1
1			
1			
1			

		1	
	1		1
		1	
		1	

5	2	2	3	2	4	4
3	4	2	3	3	3	4
3	2	4	3	2	3	3
4	2	2	5	1	3	5
1	4	2	2	5	3	3
3	3	5	2	2	5	2
2	2	1	3	2	2	5

3	3	2	3	2	4	5
3	3	2	2	4	5	2
2	3	2	2	4	4	3
2	2	1	5	3	3	4
1	2	3	3	3	3	2
0	3	3	3	3	3	4
2	3	2	3	2	3	4

5	2	2	3	2	4	4
3	4	2	3	3	3	4
3	2	4	3	2	3	3
4	2	2	5	1	3	5
1	4	2	2	5	3	3
3	3	5	2	2	5	2
2	2	1	3	2	2	5

3	3	2	3	2	4	5
3	3	2	2	4	5	2
2	3	2	2	4	4	3
2	2	1	5	3	3	4
1	2	3	3	3	3	2
0	3	3	3	3	3	4
2	3	2	3	2	3	4

5	3	3	5	1	1	5
4	3	0	4	2	2	4
3	2	3	3	3	2	3
2	2	2	3	3	3	4
0	1	1	2	5	3	3
0	2	2	1	4	3	3
1	2	1	3	4	2	3

5	3	1	1	4	4	3
2	3	1	2	4	3	4
4	2	3	3	2	4	3
2	3	3	3	4	3	4
1	3	4	2	3	3	4
2	1	4	4	4	4	4
2	3	3	1	3	2	3

5	3	4	4
4	5	3	5
4	5	5	3
2	3	2	5

3	3	4	5
3	5	4	4
3	3	3	4
3	3	3	4

5	5	2	5
3	3	3	4
2	2	5	3
2	3	4	3

5	2	4	4
4	3	4	4
3	4	4	4
3	3	3	3

x			
	x		x
	x	x	
			x

			x
	x		

x	x		x
		x	

x			

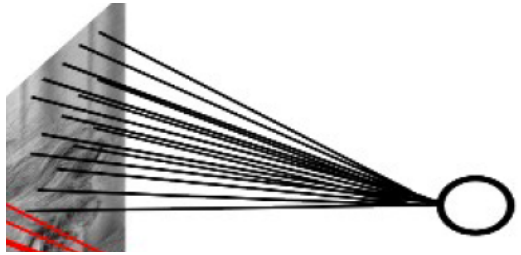
5	3	4	4
4	5	3	5
4	5	5	3
2	3	2	5
3	3	4	5
3	5	4	4
3	3	3	4
3	3	3	4
5	5	2	5
3	3	3	4
2	2	5	3
2	3	4	3
5	2	4	4
4	3	4	4
3	4	4	4
3	3	3	3

- How many convolutional windows (filters)
- What are they?

- → let the NN learn for us

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

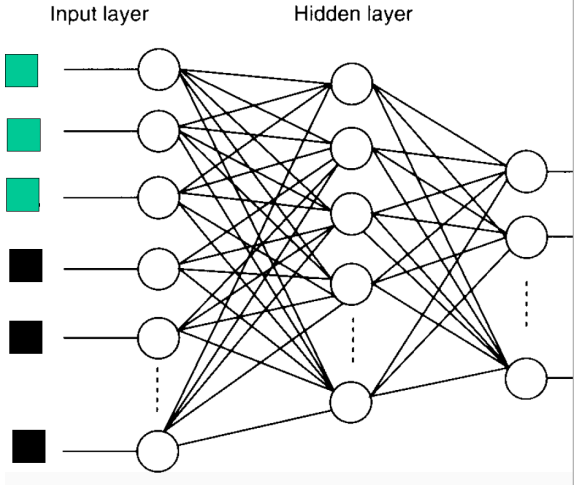


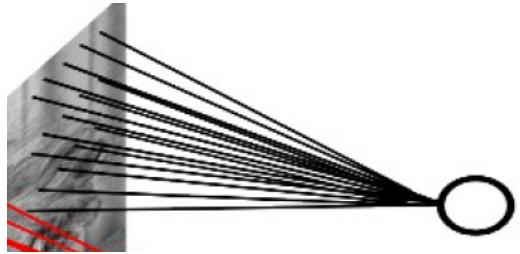
5	2	2	3	2	4	4
3	4	2	3	3	3	4
3	2	4	3	2	3	3
4	2	2	5	1	3	5
1	4	2	2	5	3	3
3	3	5	2	2	5	2
2	2	1	3	2	2	5

1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

5	2	2	3	2	4	4
3	4	2	3	3	3	4
3	2	4	3	2	3	3
4	2	2	5	1	3	5
1	4	2	2	5	3	3
3	3	5	2	2	5	2
2	2	1	3	2	2	5





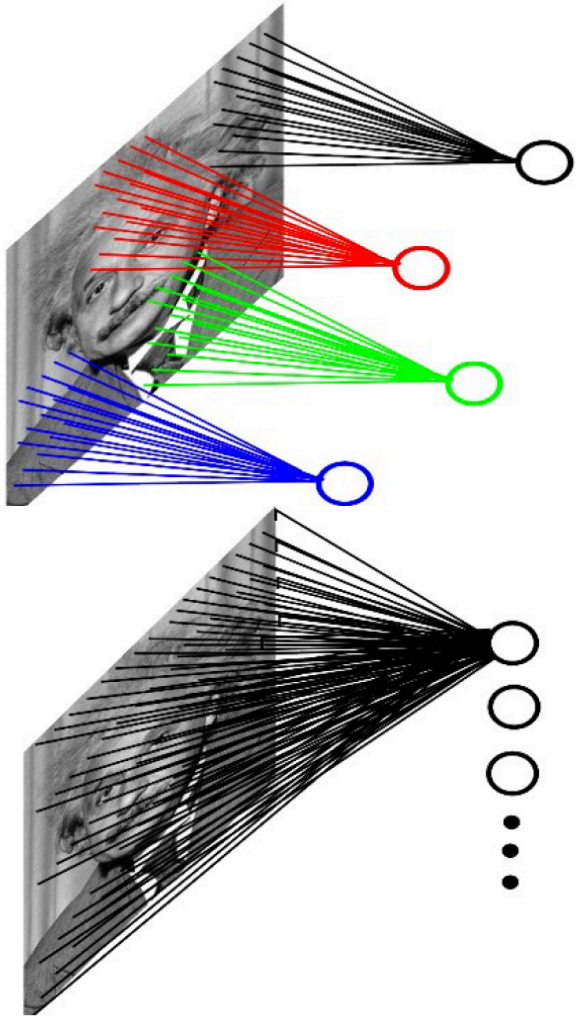
1	1	1	1			1	1	1	1
1	1		1			1	1	1	
1		1	1			1	1		
1	1	1	1			1		1	1
				1	1	1	1	1	1
	1	1		1	1			1	1
			1	1		1		1	1
				1			1	1	
				1	1	1	1	1	
	1	1	1	1					1

1			1
	1		
		1	
			1

			1
		1	
	1		
1			1

1			1
1			
1			
1			

		1	
	1		1
		1	
		1	



Shared weights

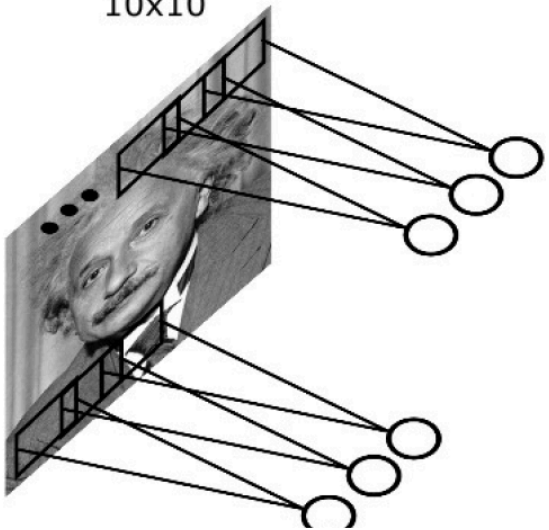
- Features that are useful on one part of the image and probably useful elsewhere.
- All units share the same set of weights
- Shift equivariant processing:
  - When the input shifts, the output also shifts but stays otherwise unchanged.
- Convolution
  - with a learned kernel (or filter)
  - Non-linearity: ReLU (rectified linear)

$$A_{ij} = \sum_{kl} W_{kl} X_{i+j.k+l}$$

- The filtered "image" Z is called a feature map

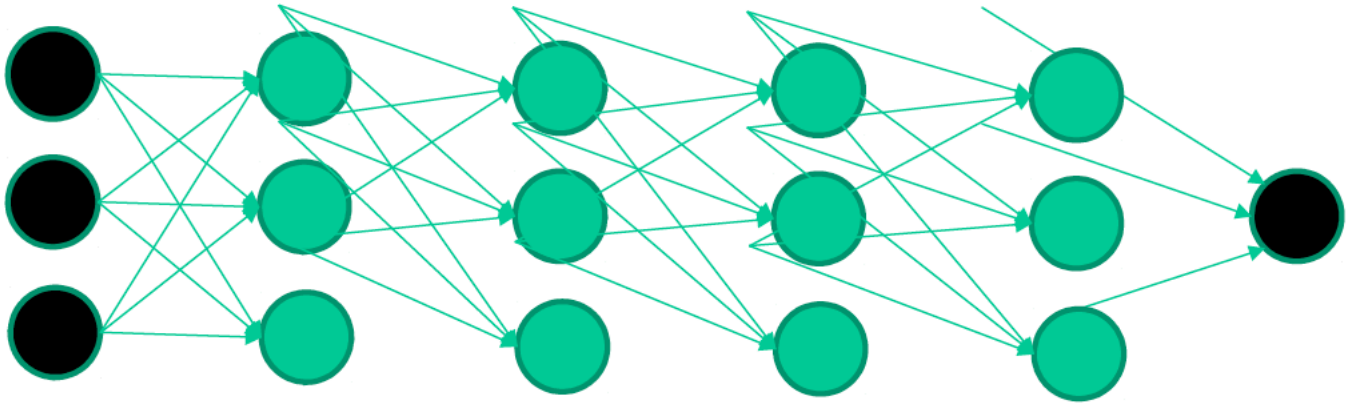
$$Z_{ij} = max(0, A_{ij})$$

- Example: 200x200 image
  - 400,000 hidden units with 10x10 fields = 1000 params
  - 10 feature maps of size 200x200, 10 filters of size 10x10

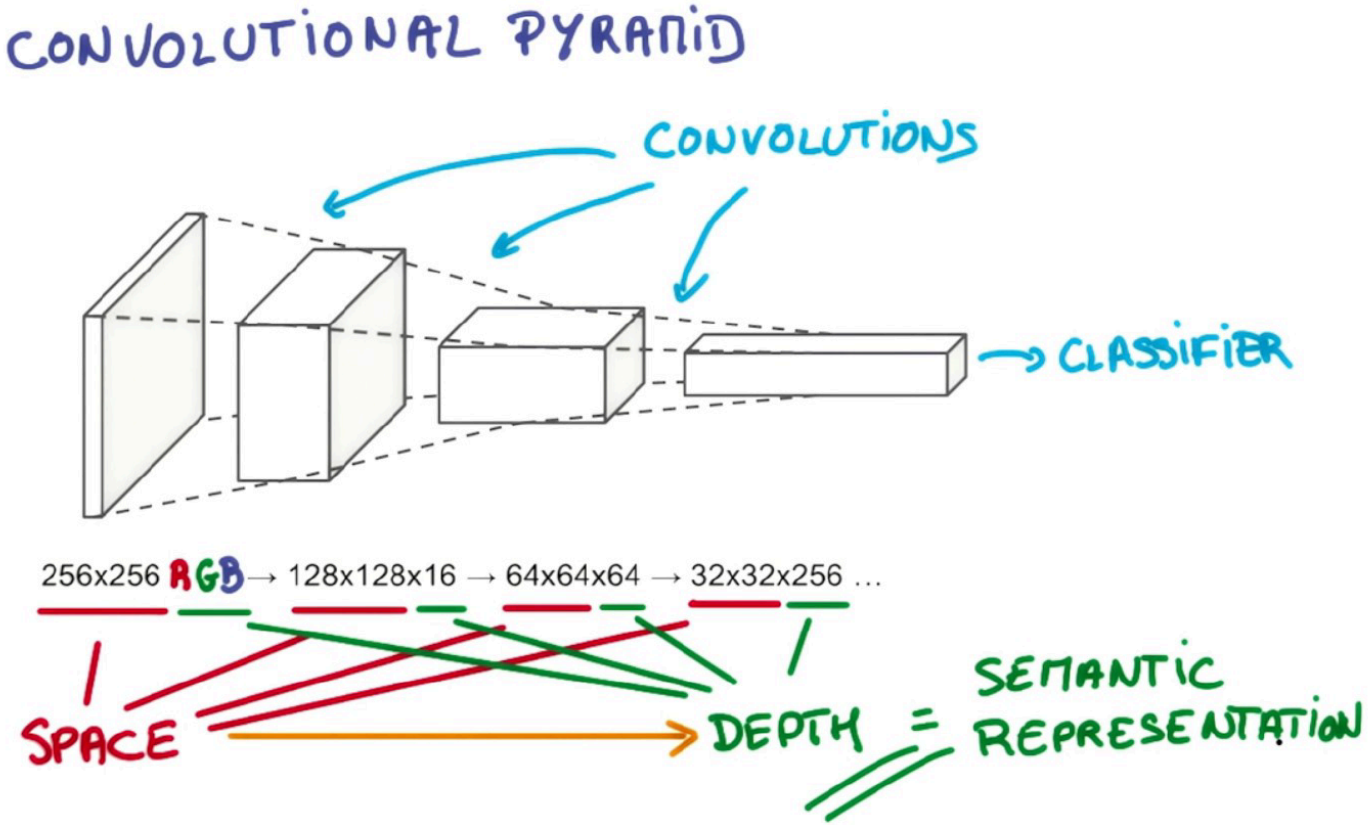


Multi-layer Neural Networks

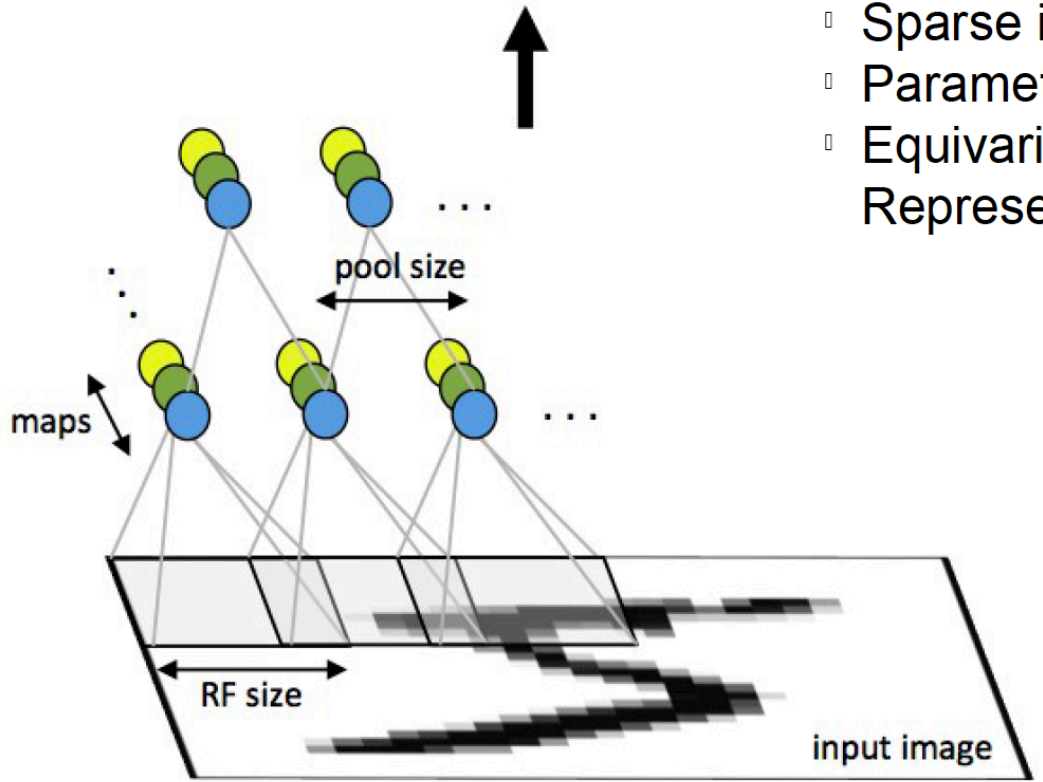
- Full connected
- Tied weight
- Hard to efficiently train a very deep networks



Convolutional Neural Networks



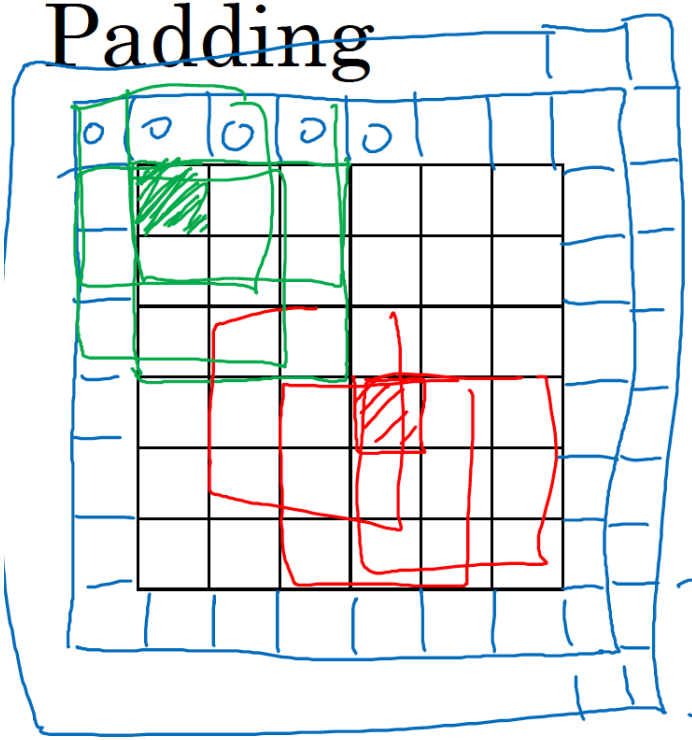
Convolutional Neural Networks



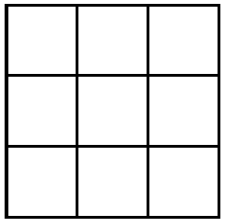
- Sparse interactions
- Parameter sharing
- Equivariant Representations

Padding

- Shrinky output
- throw away info from edge

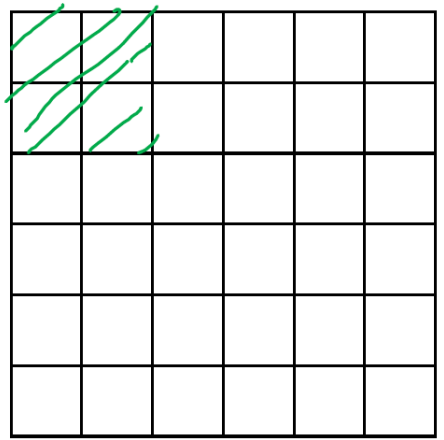


*



3×3
f×f

=



6×6

p=2

6×6 → 8×8
n×n

n-f+1 × n-f+1

6-3+1=4

→ ~~4×4~~

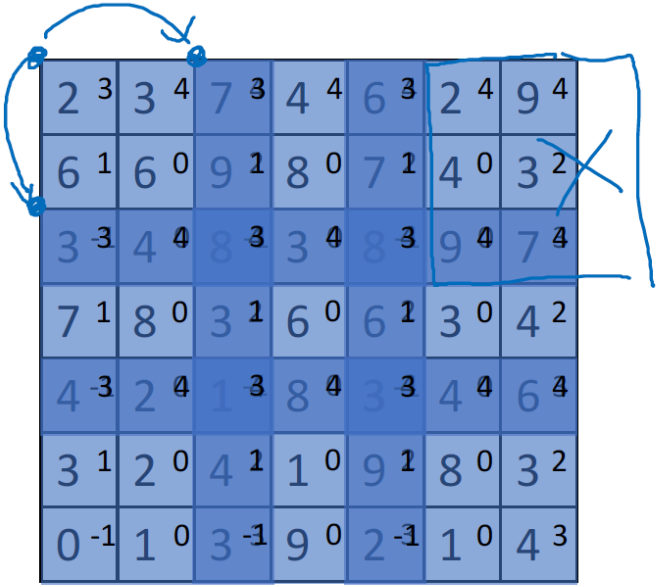
p = padding = 1

n+2p-f+1 × n+2p-f+1

6+2-3+1 × ___ = 6×6

Andrew Ng

Strided convolution



7×7

*

3	4	4
1	0	2
-1	0	3

3×3

stride = 2

=

91	100	83
69	91	127
44	72	74

3×3

$\lfloor z \rfloor = \text{floor}(z)$

$n \times n \quad * \quad f \times f$

padding p stride s

$s = 2$

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

$$\frac{7+0-3}{2} + 1 = \frac{4}{2} + 1 = 3$$

Summary of convolutions

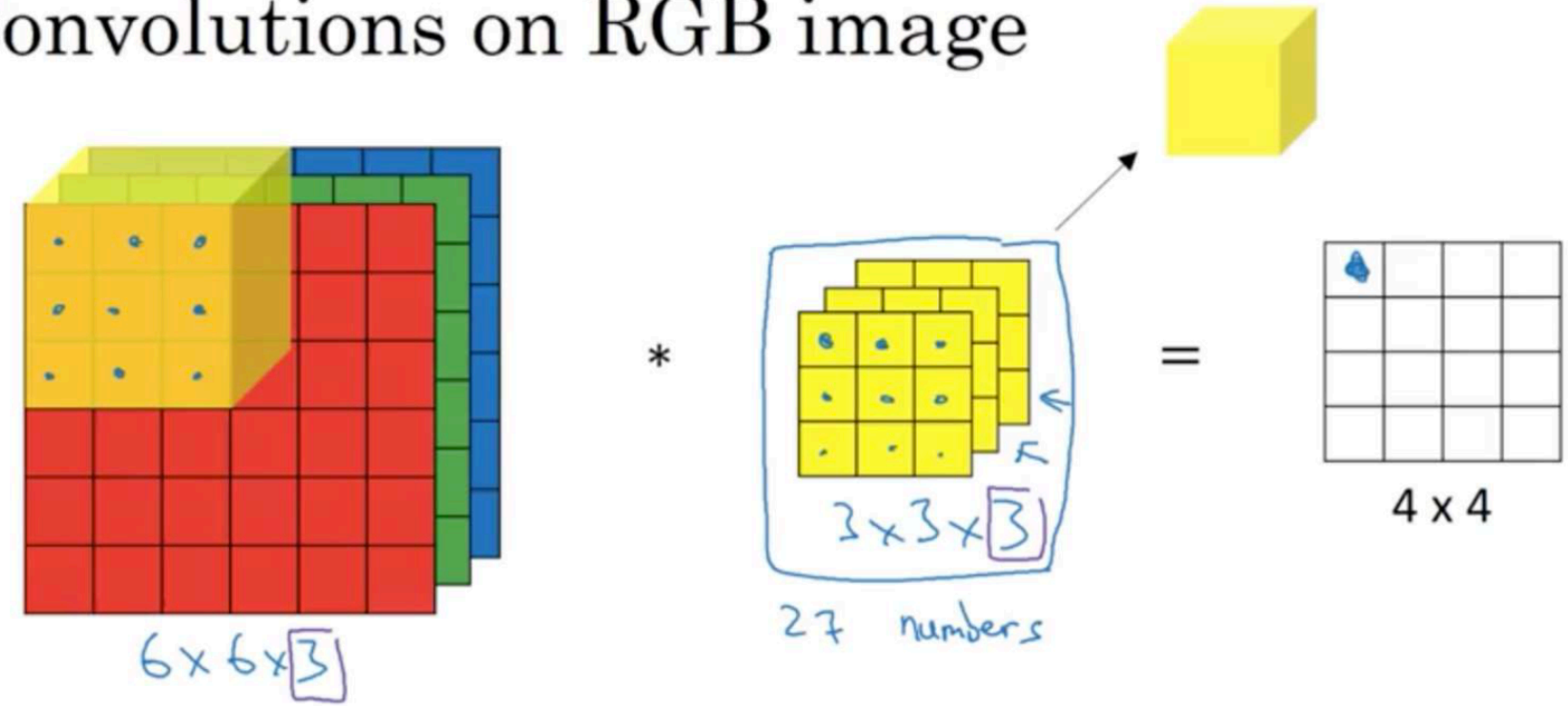
$n \times n$ image $\quad$ $f \times f$ filter

padding p $\quad$ stride s

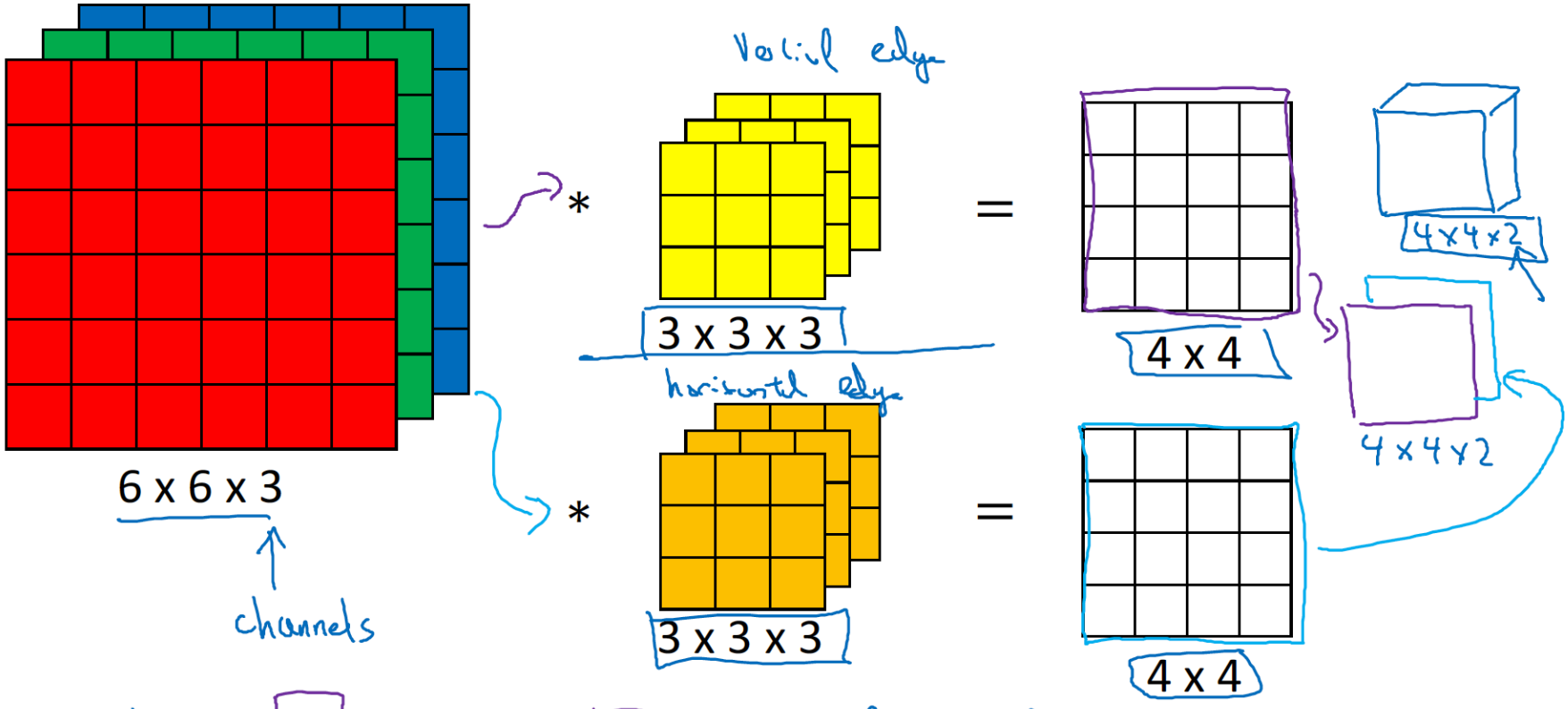
Output size:

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

Convolutions on RGB image



Multiple filters

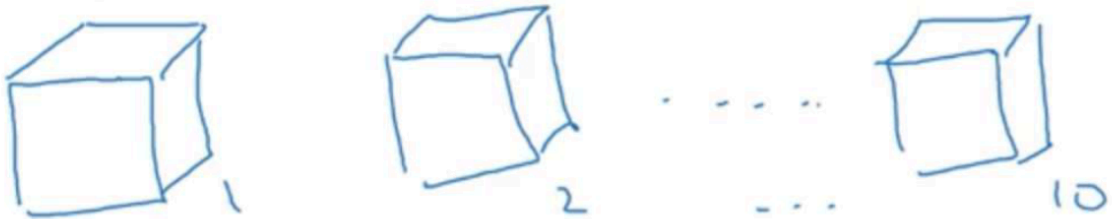


Summary: $n \times n \times n_c$ $*$ $f \times f \times n_c$ $\rightarrow$ $n-f+1 \times n-f+1 \times n_c'$

$6 \times 6 \times 3$ $*$ $3 \times 3 \times 3$ $\rightarrow$ $4 \times 4 \times 2$ (# filters)

Number of parameters in one layer

If you have 10 filters that are 3 x 3 x 3 in one layer of a neural network, how many parameters does that layer have?



Summary of notation

If layer l is a convolution layer:

$f^{[l]}$ = filter size

$p^{[l]}$ = padding

$s^{[l]}$ = stride

$n_c^{[l]}$ = number of filters

→ Each filter is: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]}$

Activations: $a^{[l]} \rightarrow n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

$A^{[l]} \rightarrow m \times n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$

bias: $n_c^{[l]} - (1,1,1,n_c^{[l]})$ ← #filters in layer l.

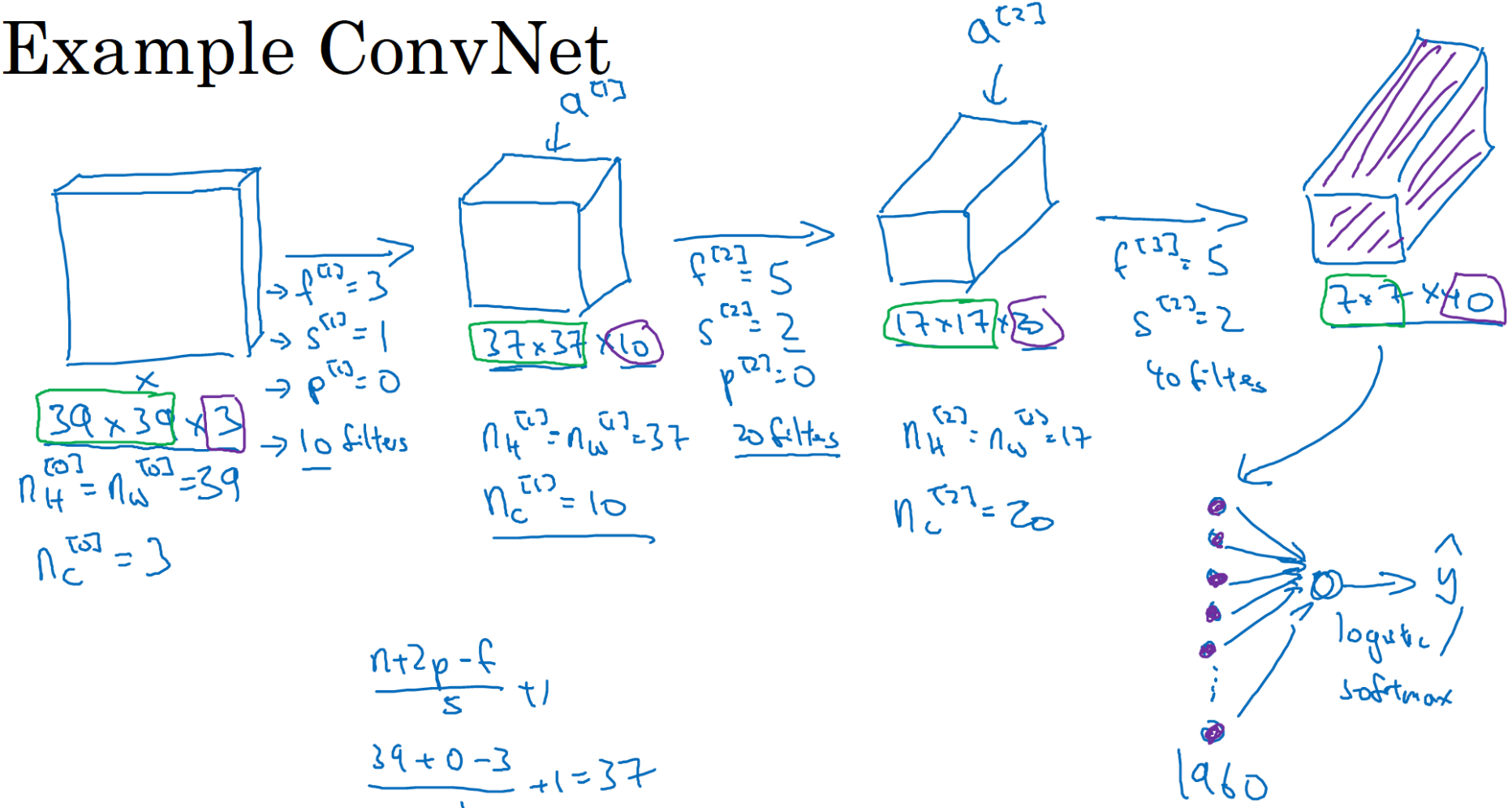
Input: $n_H^{[l-1]} \times n_W^{[l-1]} \times n_c^{[l-1]}$ ←

Output: $n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$ ←

$$n_{H,W}^{[l]} = \left\lfloor \frac{n_{H,W}^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$$

$n_c \times n_H \times n_W$

Example ConvNet



$$\frac{n+2p-f}{s}+1$$

$$\frac{39+0-3}{1}+1=37$$

Types of layer in a convolutional network:

- Convolution (CONV) ←
- Pooling (POOL) ←
- Fully connected (FC) ←

}

Summary of pooling

Hyperparameters:

- f : filter size
- s : stride
- Max or average pooling

f=2, s=2

f=3, s=2

~~p: padding~~

No parameters to learn!

$n_H \times n_W \times n_C$

$\downarrow$

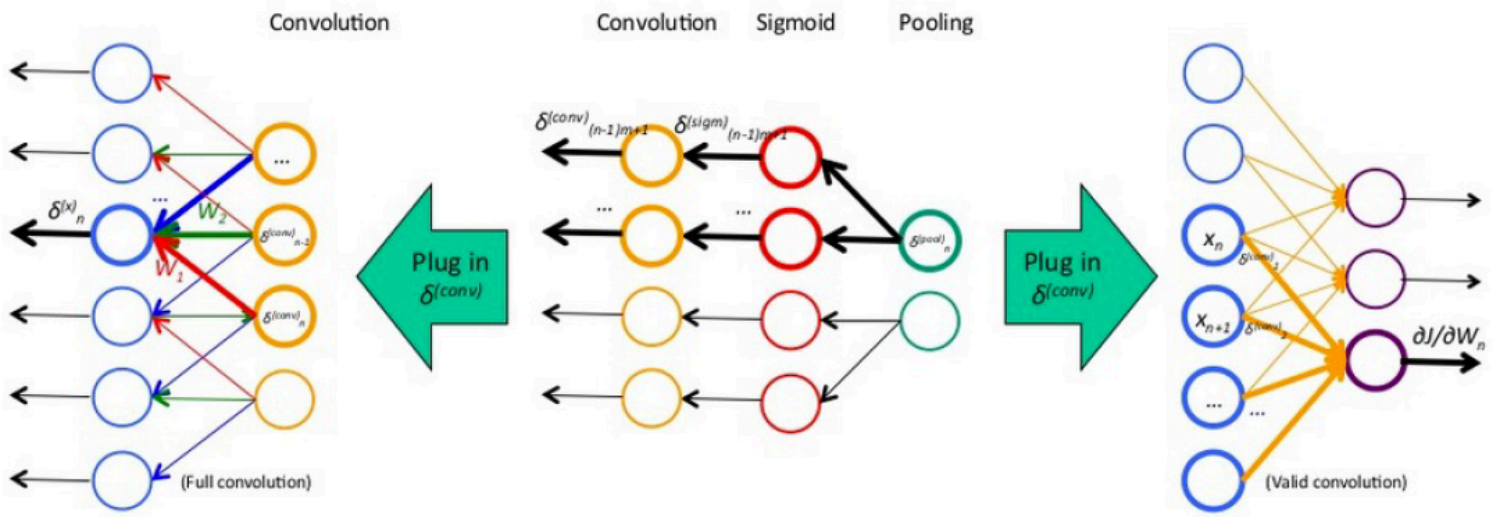
$\left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W - f}{s} + 1 \right\rfloor$

$\times n_C$

2. Propagate error signals $\delta^{(conv)}$

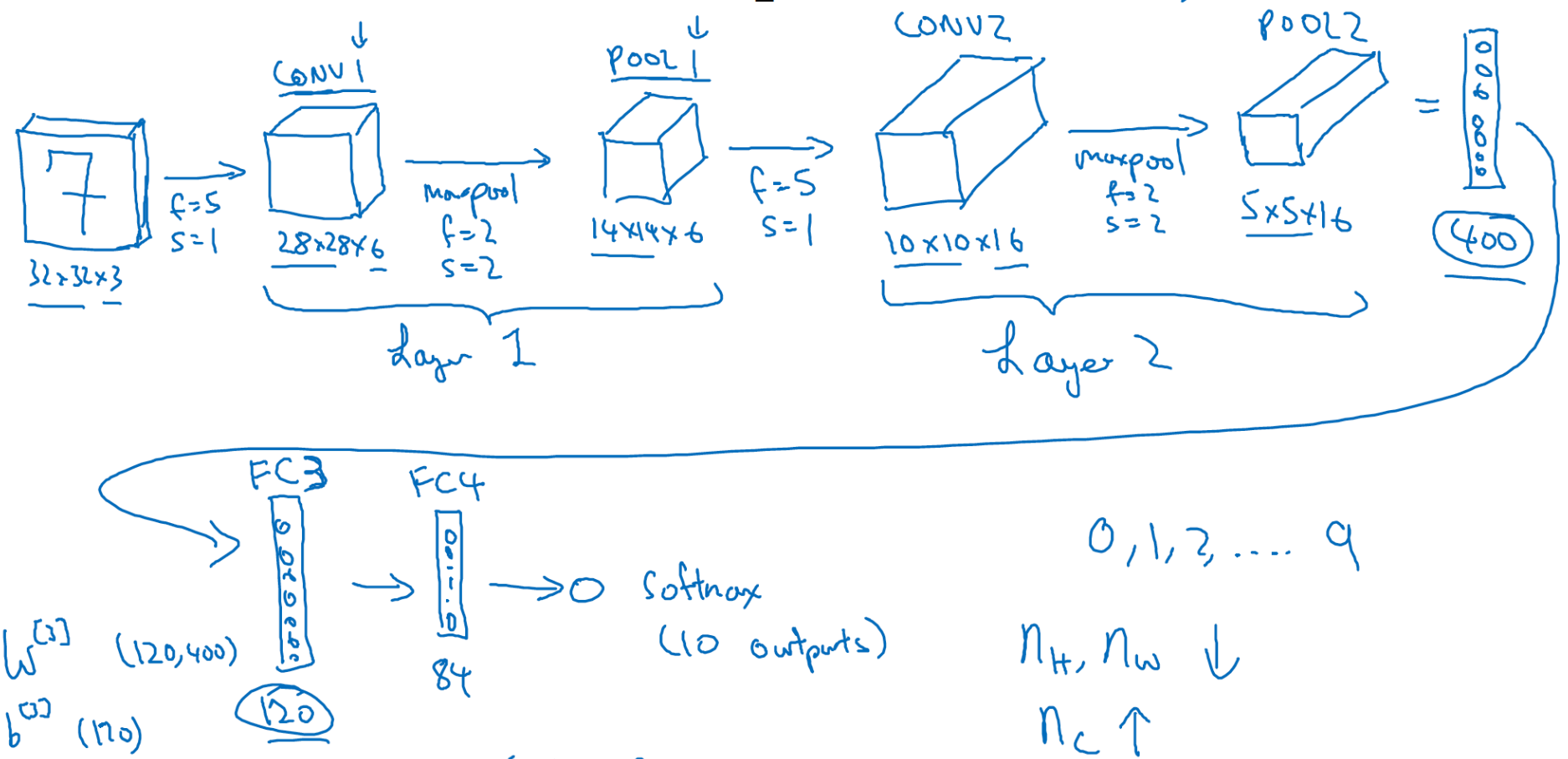
1. Propagate error signals $\delta^{(pool)}$

3. Compute gradient $\nabla_w J$



$$\delta^{(x)} = \delta^{(conv)} * \text{flip}(w) \qquad \delta^{(conv)} = \text{upsample}\left(\delta^{(pool)}, g'\right) \bullet \underbrace{f^{(sigm)} \bullet \left(1 - f^{(sigm)}\right)}_{\text{Derivative of sigmoid}} \qquad x * \delta^{(conv)} = \nabla_w J$$

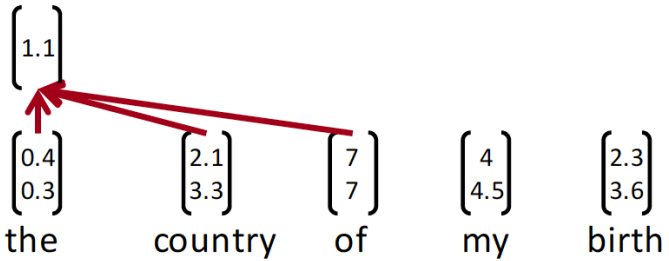
Neural network example (LeNet-5)



Andrew Ng

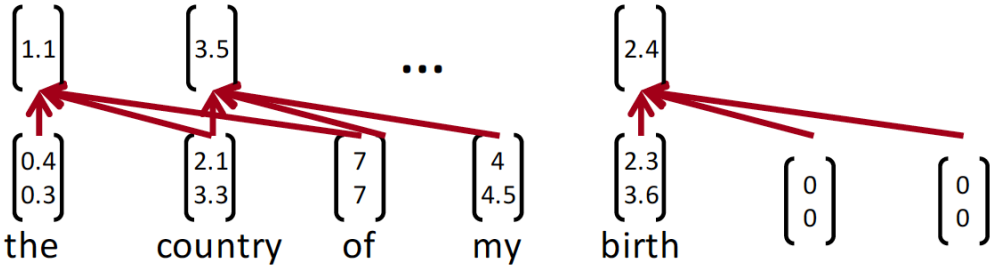
Single layer CNN for NLP

- A simple variant using one convolutional layer and **pooling**
- Based on Collobert and Weston (2011) and Kim (2014) "Convolutional Neural Networks for Sentence Classification"
- Word vectors: $\mathbf{x}_i \in \mathbb{R}^k$
- Sentence: $\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n$ (vectors concatenated)
- Concatenation of words in range: $\mathbf{x}_{i:i+j}$
- Convolutional filter: $\mathbf{w} \in \mathbb{R}^{hk}$ (goes over window of h words)
- Could be 2 (as before) higher, e.g. 3:



Representation of n-gram relations

- Filter w is applied to all possible windows (concatenated vectors)
- Sentence: $\mathbf{x}_{1:n} = \mathbf{x}_1 \oplus \mathbf{x}_2 \oplus \ldots \oplus \mathbf{x}_n$
- All possible windows of length h: $\{\mathbf{x}_{1:h}, \mathbf{x}_{2:h+1}, \ldots, \mathbf{x}_{n-h+1:n}\}$
- Result is a feature map: $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$



Pooling layers

- New building block: Pooling
- In particular: max-over-time pooling layer
- Idea: capture most important activation (maximum over time)

- From feature map $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$

- Pooled single number: $\hat{c} = \max\{\mathbf{c}\}$

- But we want more features!

Multiple filters

- Use multiple filter weights w
- Useful to have different window sizes h
- Because of max pooling $\hat{c} = \max\{\mathbf{c}\}$, length of **c** irrelevant

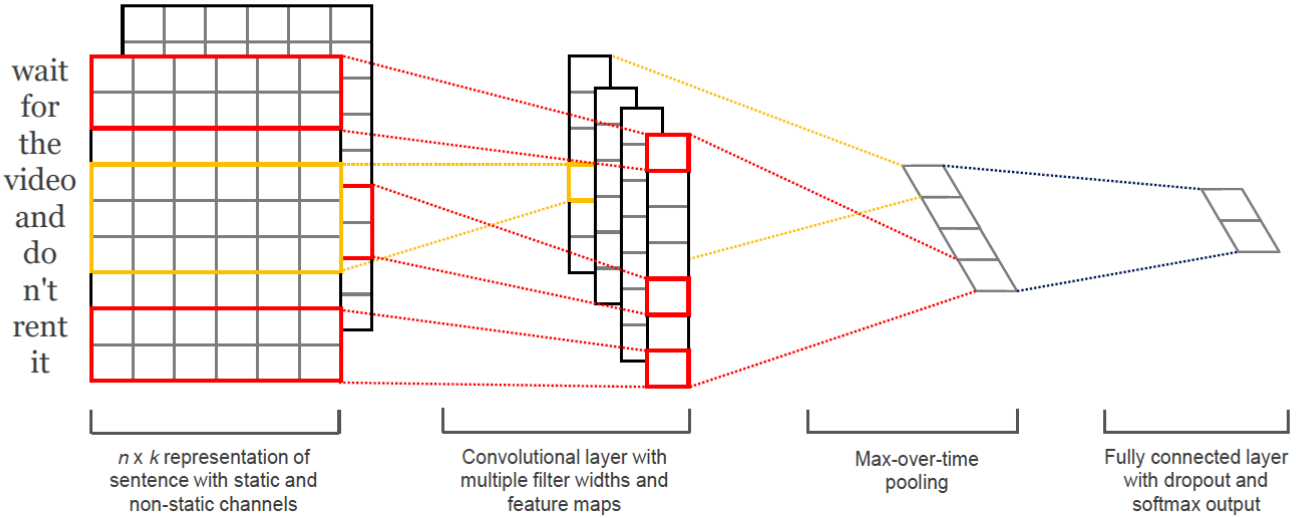
$$\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$$

- So we can have some filters that look at unigrams, bigrams, tri-grams, 4-grams, etc.

Classification of CNN

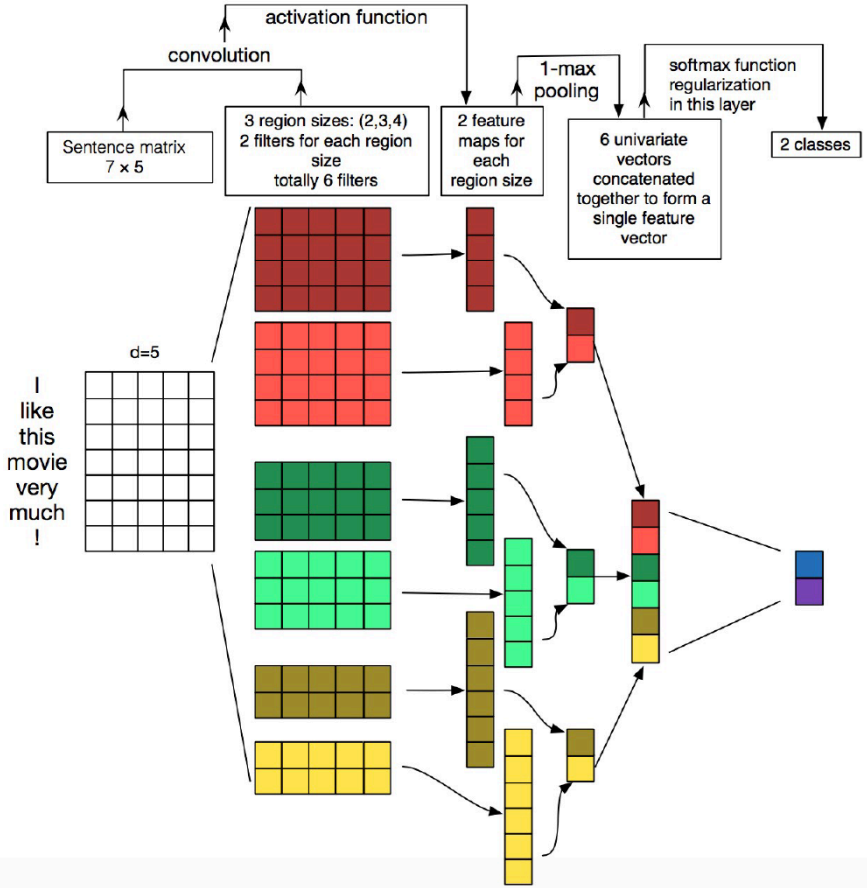
- First one convolution, followed by one max-pooling
- To obtain final feature vector: $\mathbf{z} = [\hat{c}_1, \ldots, \hat{c}_m]$ (assuming m filters w)
- Simple final softmax layer $y = softmax\left(W^{(S)} z + b\right)$

General architecture



n words (possibly zero padded) and each word vector has k dimensions

Multiple filters



Dropout

- Idea: randomly mask/dropout/set to 0 some of the feature weights z
- Create masking vector r of Bernoulli random variables with probability p (a hyperparameter) of being 1

- Delete features during training:

$$y = softmax\left(W^{(S)}(r \circ z) + b\right)$$

- Reasoning: Prevents co-adaptation (overfitting to seeing specific feature constellations)

Dropout

$$y = softmax\left(W^{(S)}(r \circ z) + b\right)$$

- At training time, gradients are backpropagated only through those elements of z vector for which $r_i = 1$

- At test time, there is no dropout, so feature vectors z are larger.
- Hence, we scale final vector by Bernoulli probability p

$$\hat{W}^{(S)} = pW^{(S)}$$

- Kim (2014) reports **2 – 4% improved accuracy** and ability to use very large networks without overfitting

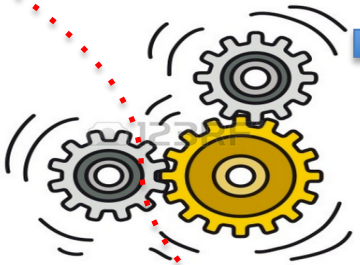
All hyperparameters in Kim (2014)

- Find hyperparameters based on dev set
- Nonlinearity: reLu
- Window filter sizes h = 3,4,5
- Each filter size has 100 feature maps
- Dropout p = 0.5
- L2 constraint s for rows of softmax s = 3
- Mini batch size for SGD training: 50
- Word vectors: pre-trained with word2vec, k = 300

- During training, keep checking performance on dev set and pick highest accuracy weights for final evaluation

S5 có thiết kết bắt mắt nhưng camera không được nét lắm

Subject: S5 (neutral)
--- Design (+)
--- Camera: (-)



SENTIMENT ENGINE

KNOWLEDGE BASE

Term	Score
bắt mắt	0.75
nét	0.75
xấu	- 0.75
tệ hại	- 0.75

Sentiment Dictionary

Pattern	Action
"không được" A	Lấy phản nghĩa của A
A "không bằng" B	A: positive B: negative

Pattern Rules

<Domain> Smartphone </>
<Brand> Samsung</>
<Product> S5, Note4 </>
<Attribute>
"thiết kế",
"camera"</>

Domain Ontology

Sentiment Analysis

- Infer *sentiment orientation/opinion* in a document (Liu, 2012)
- Three levels:
  - Document level
  - Sentence level
  - Aspect level

Choosing parameters

- https://arxiv.org/pdf/1803.09820.pdf

Aspect-based Sentiment Analysis

S5 has a nice design but its camera does not offer very high resolution

SENTIMENT ENGINE

Subject: S5 (neutral)
--- Design (+)
--- Camera: (-)

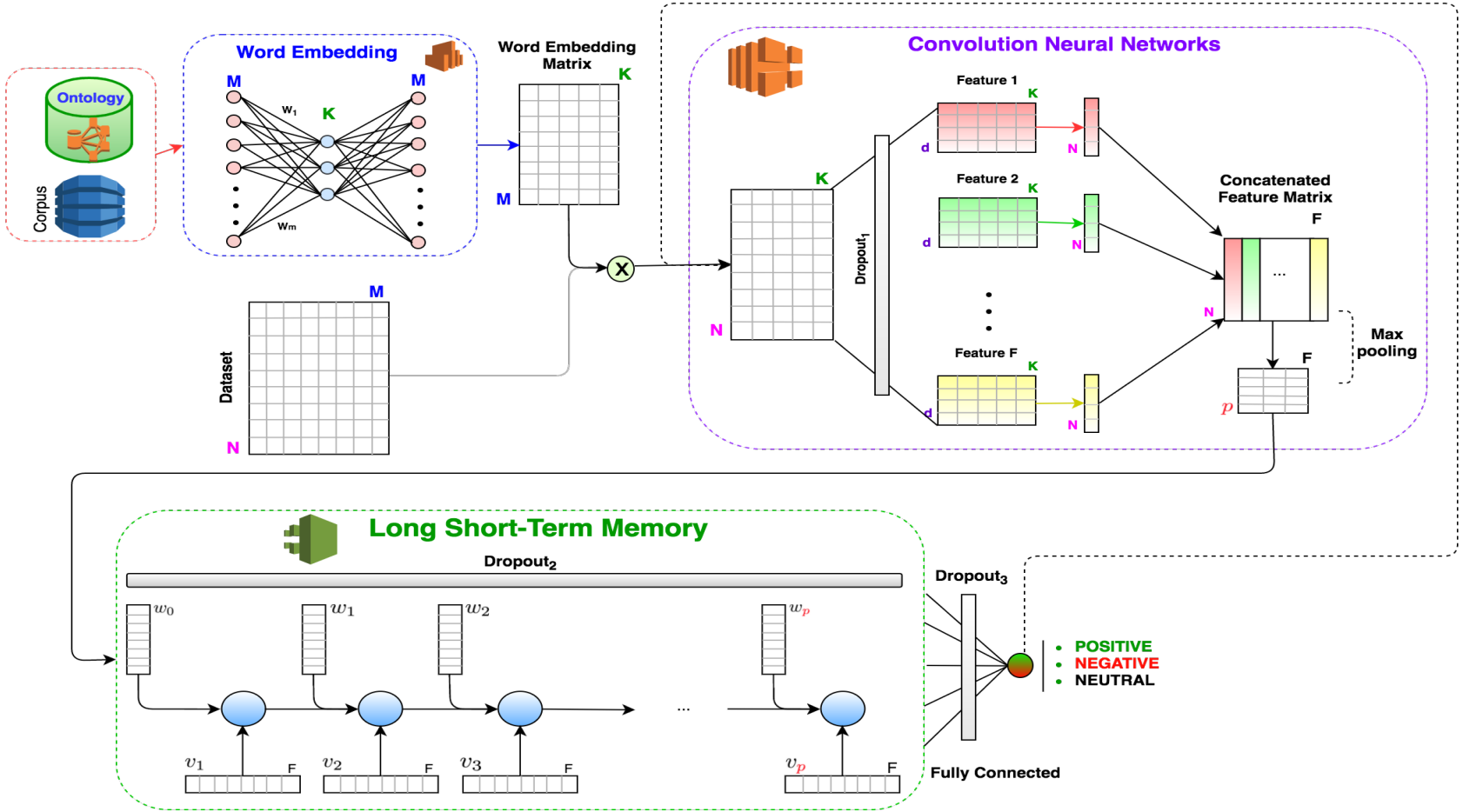
Machine Learning Approaches

- Topic Modeling
  - Extraction of aspect-related opinion works (Qiu et.al, 2013)
  - Probabilistic Latent Semantic Analysis (PLSA) for aspect and opinion modeling (Mei et. al., 2007)
  - Latent Dirichlet Allocation (LDA) (Li, 2010; Zhao,2010; Sauper,2011; Mukherjee, 2012)

Deep Learning Approaches

- Introduced by LeCun (1989), improved by Hilton (1990)
  - Standford Treebank and Stanford NLP using RNN (Pang,2005; Manning,2014; Socher, 2013)
  - CNN approach (Kim,2014)
  - LSTM approach (Wang,2016)

- Automatic recognition of “hidden features”
- Performing very well with document and sentence levels


Ontology
Corpus
Word Embedding
M
K
Word Embedding Matrix
Dataset
N
Convolution Neural Networks
Feature 1
Feature 2
Feature F
Dropout1
Concatenated Feature Matrix
Max pooling
Long Short-Term Memory
Dropout2
Dropout3
POSITIVE
NEGATIVE
NEUTRAL
Fully Connected

More sophisticated architecture