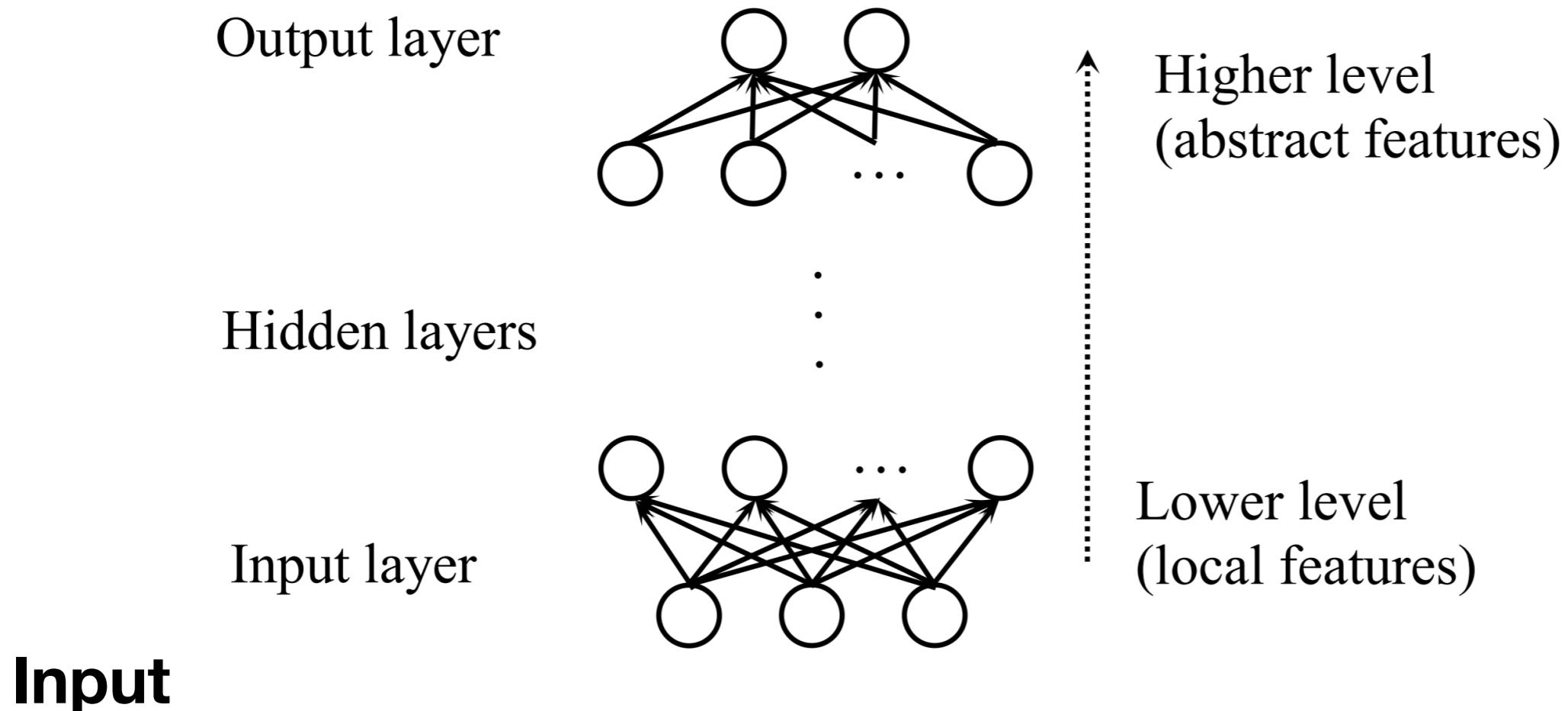


Convolutional Neural Networks & Recurrent Neural Networks

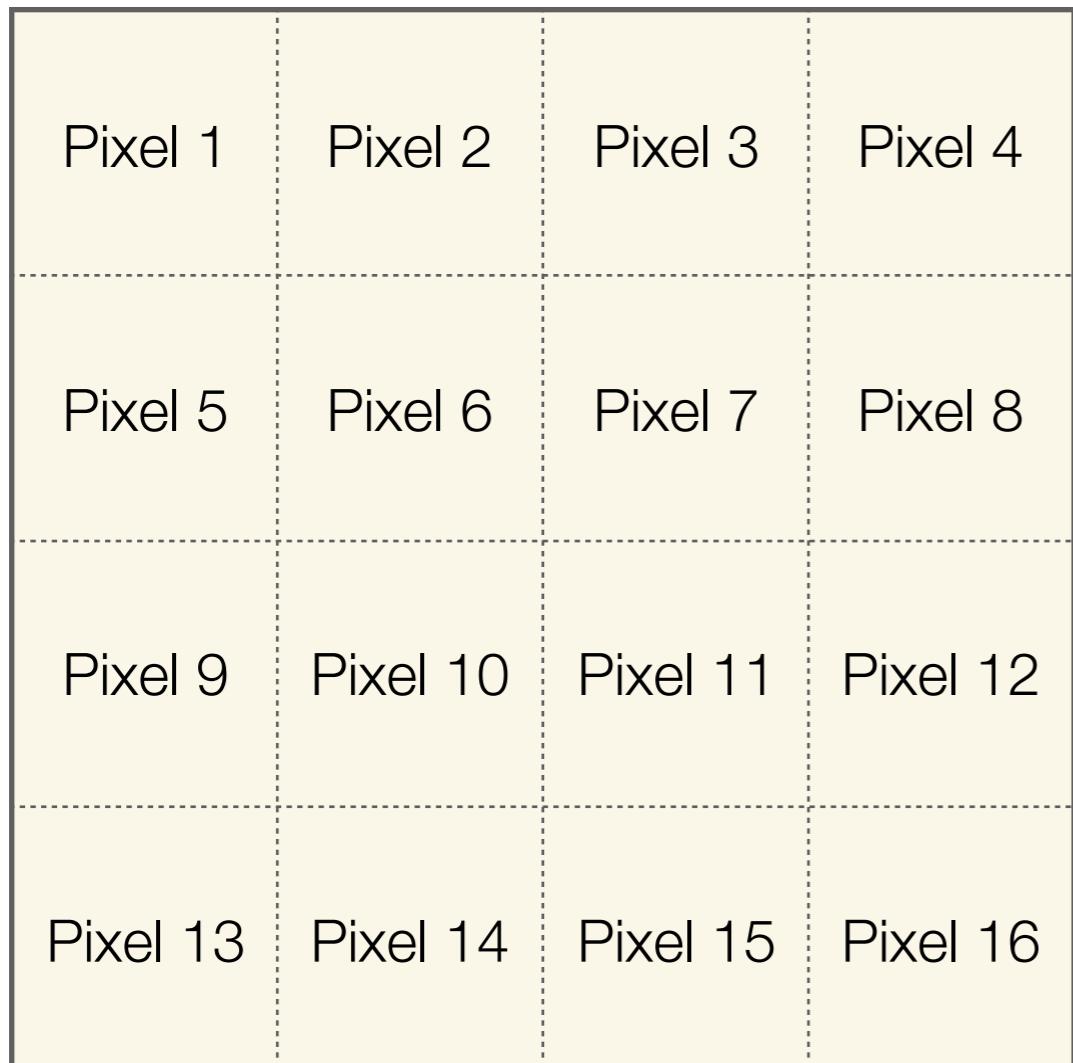
Lili Mou
l mou@ualberta.ca
lili-mou.github.io

A Generic NN

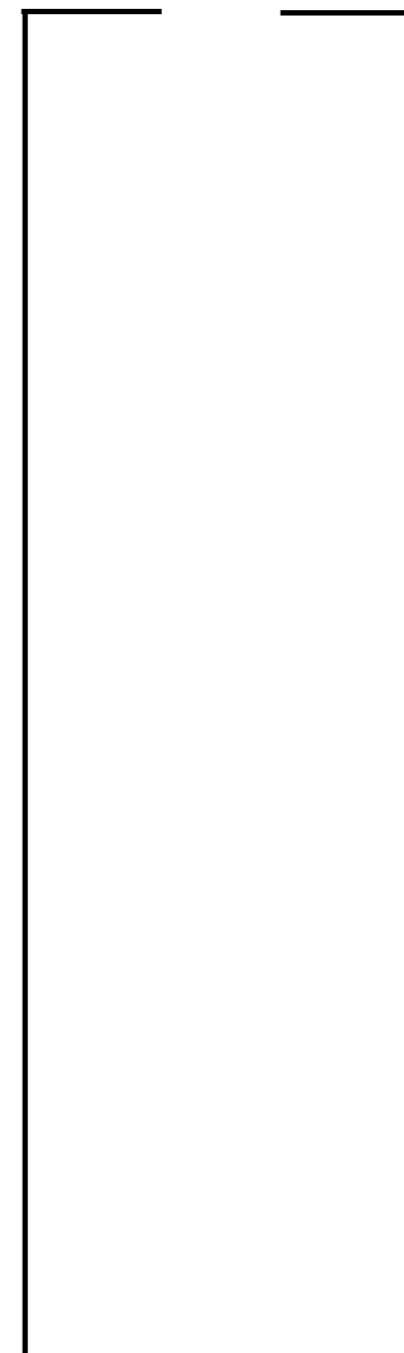


- Image of $n \times n$ pixels: n^2 -dimensional features
- Text: embeddings (how about varying-length sentences?)

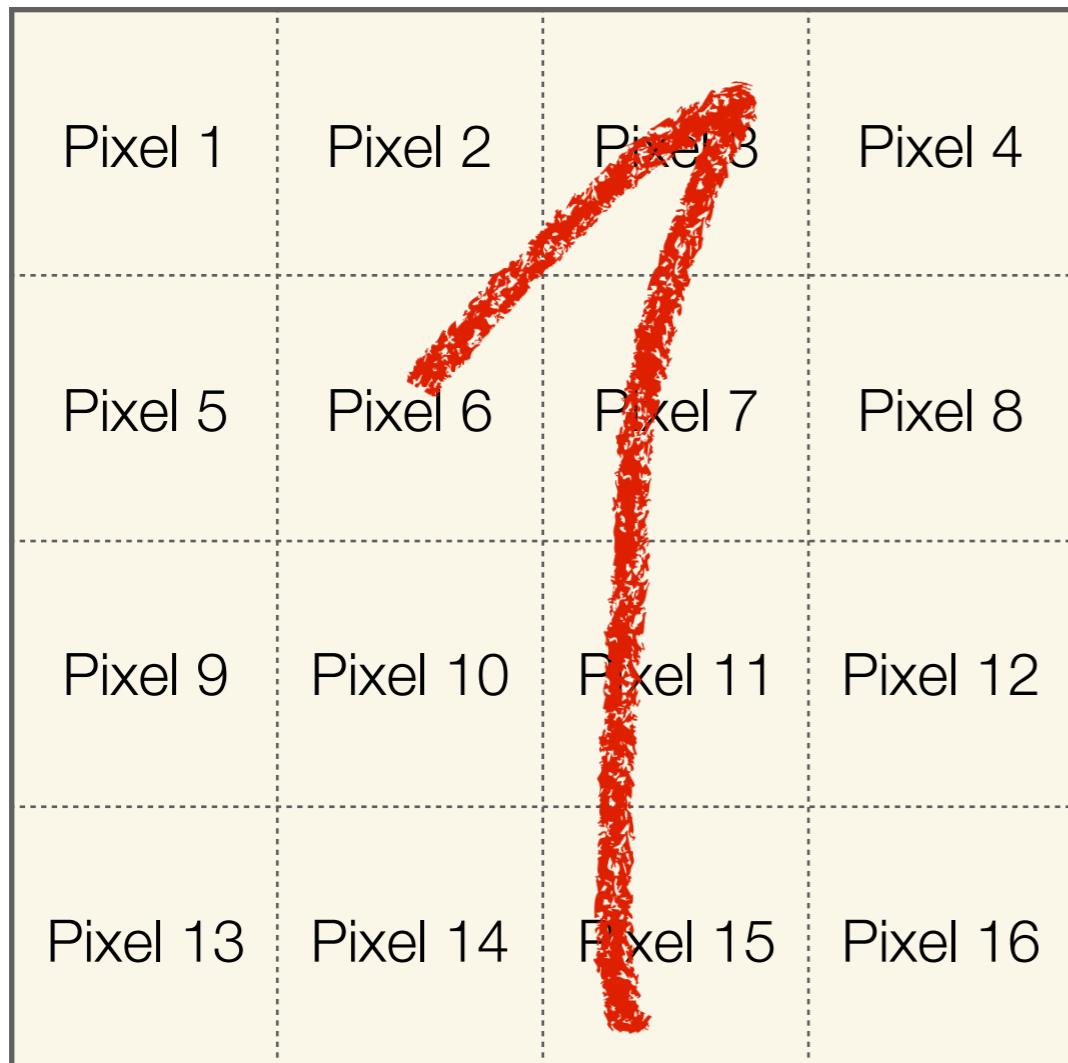
A Generic NN



Pixel 1
Pixel 2
Pixel 3
Pixel 4
Pixel 5
Pixel 6
Pixel 7
Pixel 8
Pixel 9
Pixel 10
Pixel 11
Pixel 12
Pixel 13
Pixel 14
Pixel 15
Pixel 16



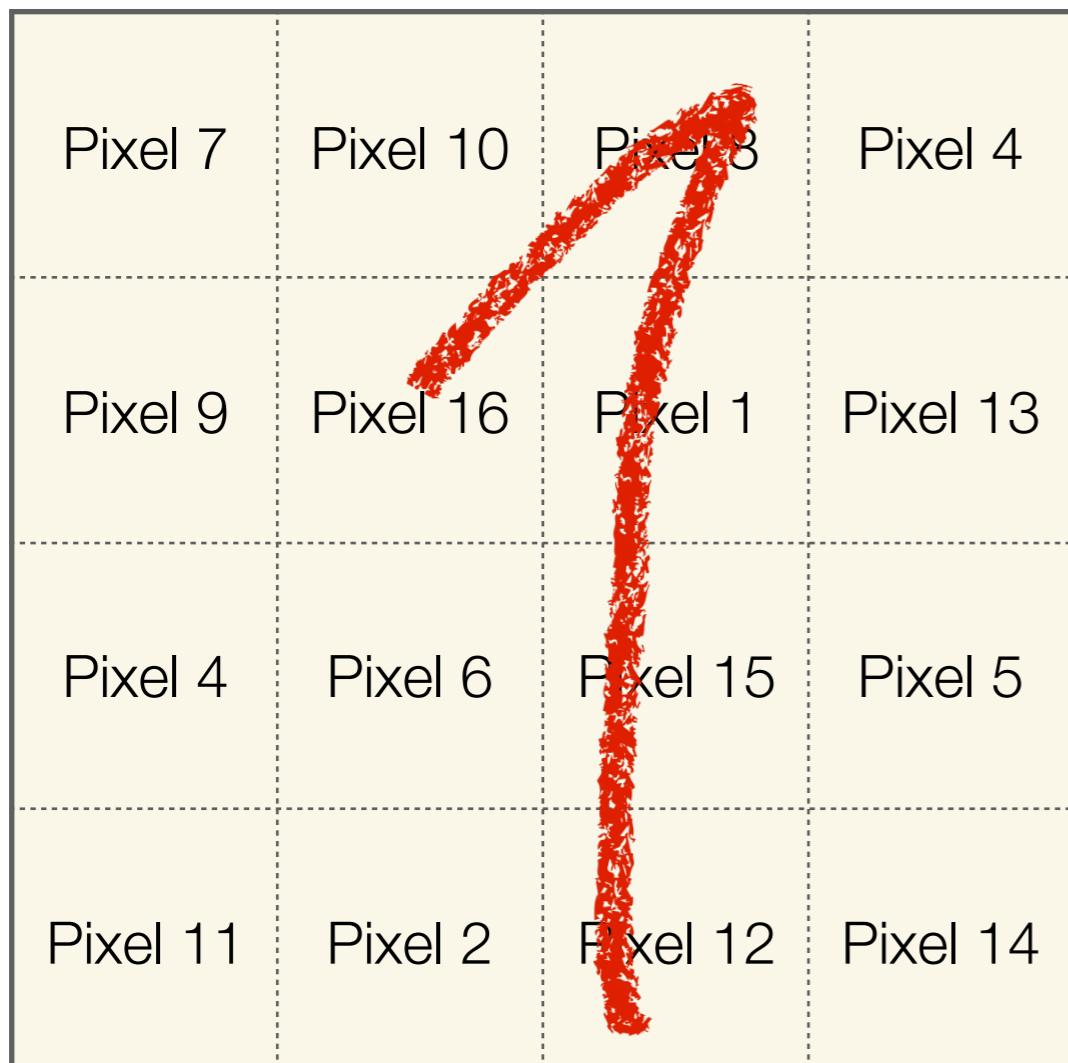
A Generic NN



Pixel 1	
Pixel 2	
Pixel 3	
Pixel 4	
Pixel 5	1
Pixel 6	1
Pixel 7	1
Pixel 8	
Pixel 9	
Pixel 10	
Pixel 11	1
Pixel 12	
Pixel 13	
Pixel 14	
Pixel 15	1
Pixel 16	

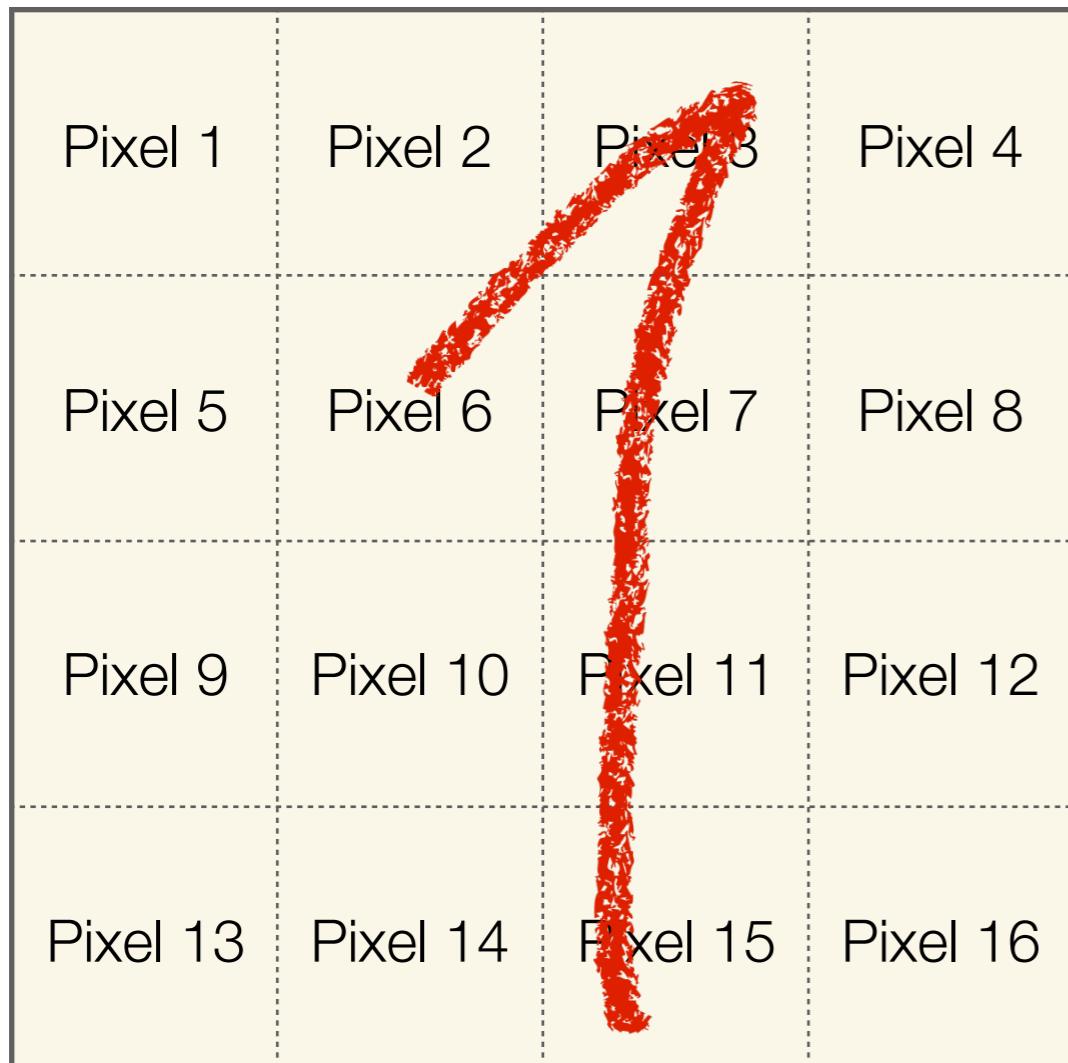


A Generic NN



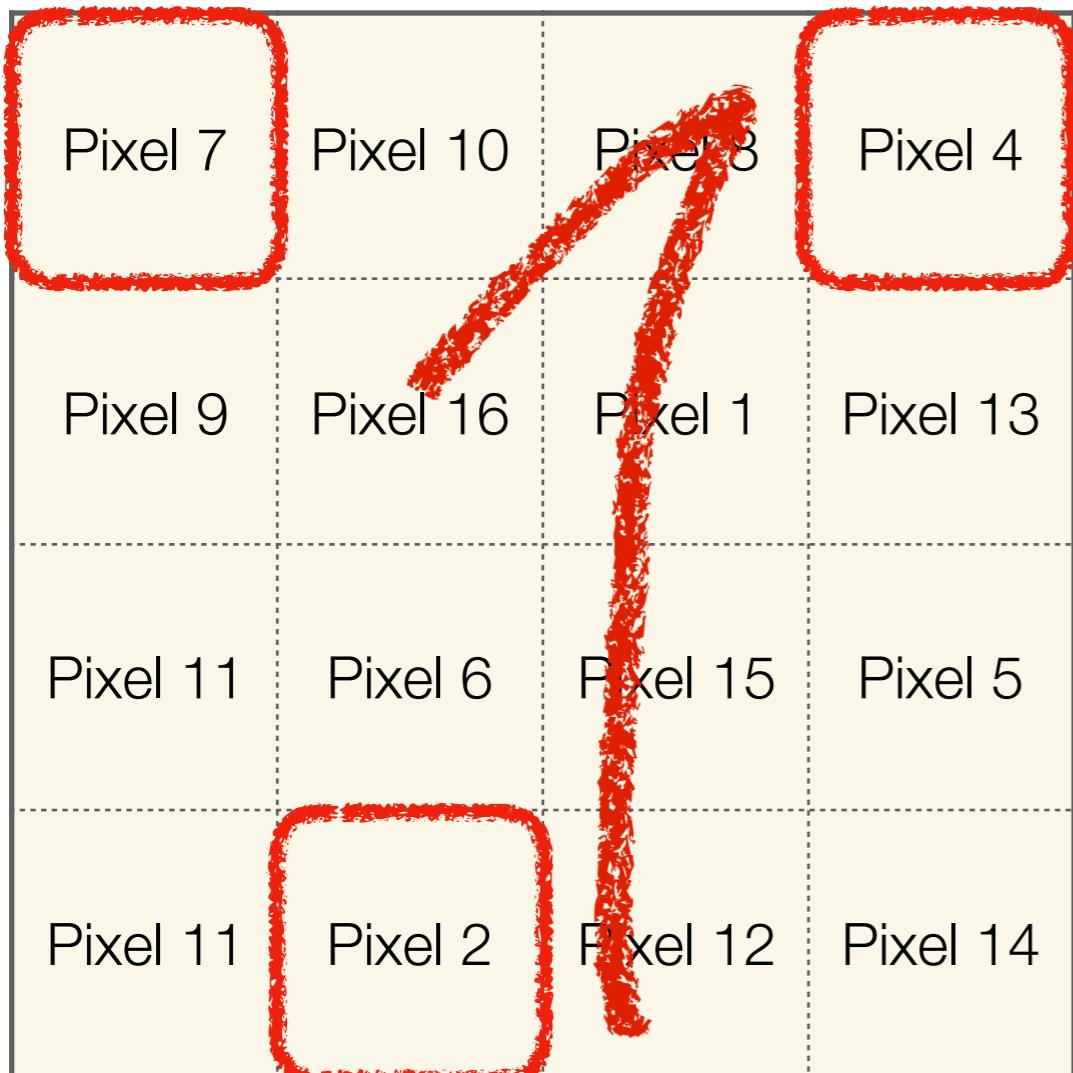
Pixel 1	1
Pixel 2	1
Pixel 3	1
Pixel 4	1
Pixel 5	1
Pixel 6	1
Pixel 7	1
Pixel 8	1
Pixel 9	1
Pixel 10	1
Pixel 11	1
Pixel 12	1
Pixel 13	1
Pixel 14	1
Pixel 15	1
Pixel 16	1

Prior Knowledge



Pixel 3 is most similar to
Pixels 2, 4, and 7.

Intuition

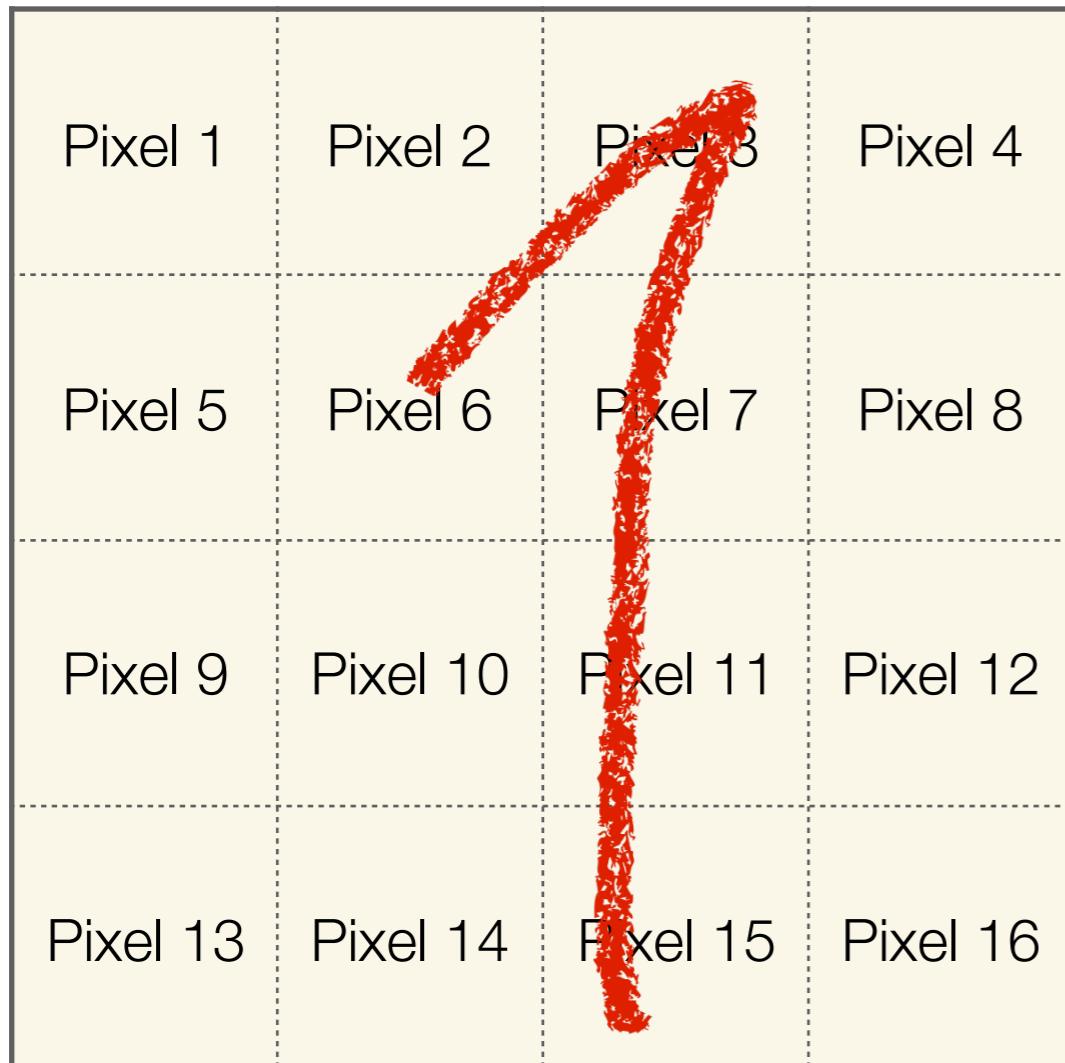


Pixel 3 is most similar to
Pixels 2, 4, and 7.

A layer-wise fully connected
NN cannot capture such
prior knowledge



Intuition

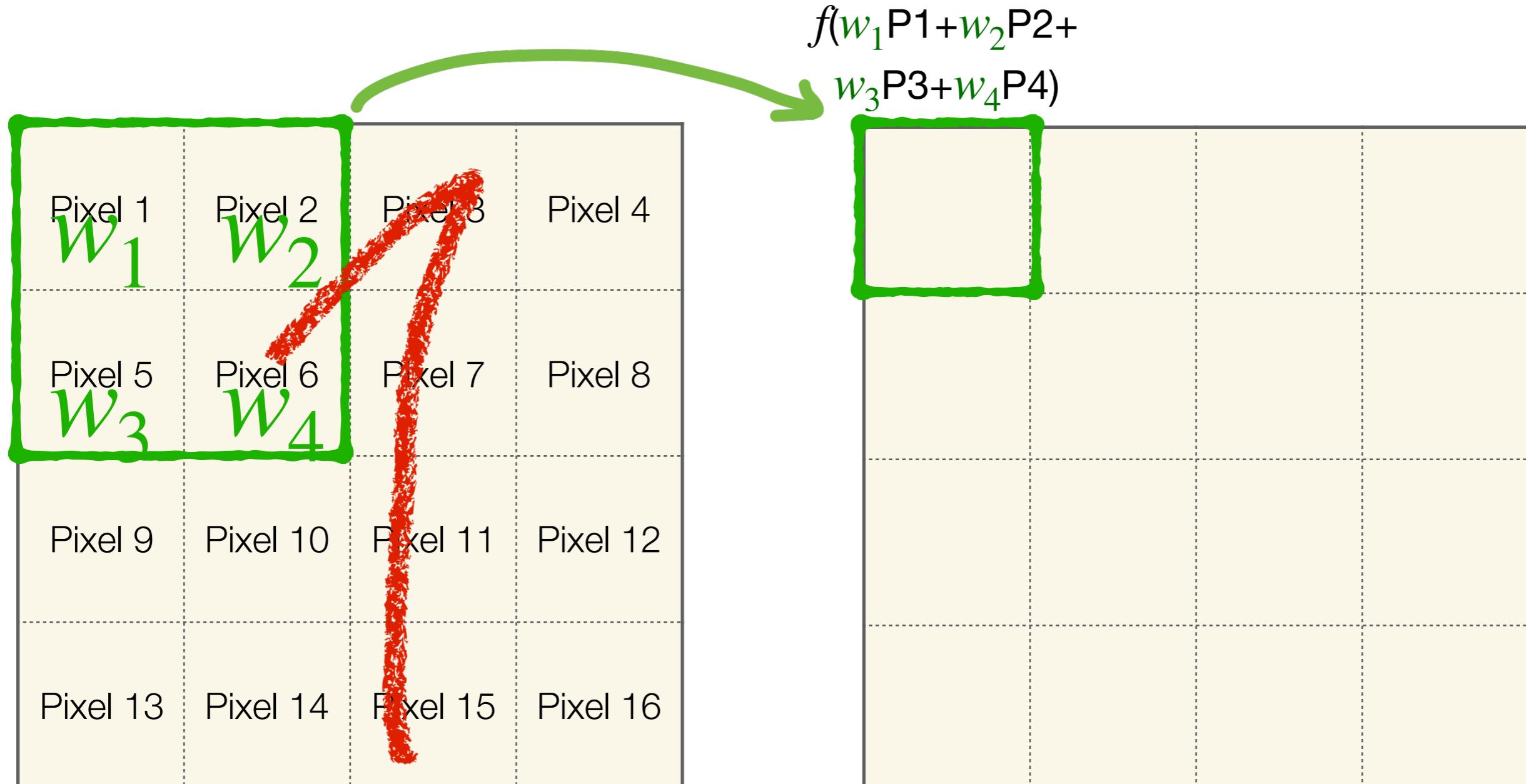


Pixel 3 is most similar to
Pixels 2, 4, and 7.

A layer-wise fully connected
NN cannot capture such
prior knowledge



Convolutional Neural Networks

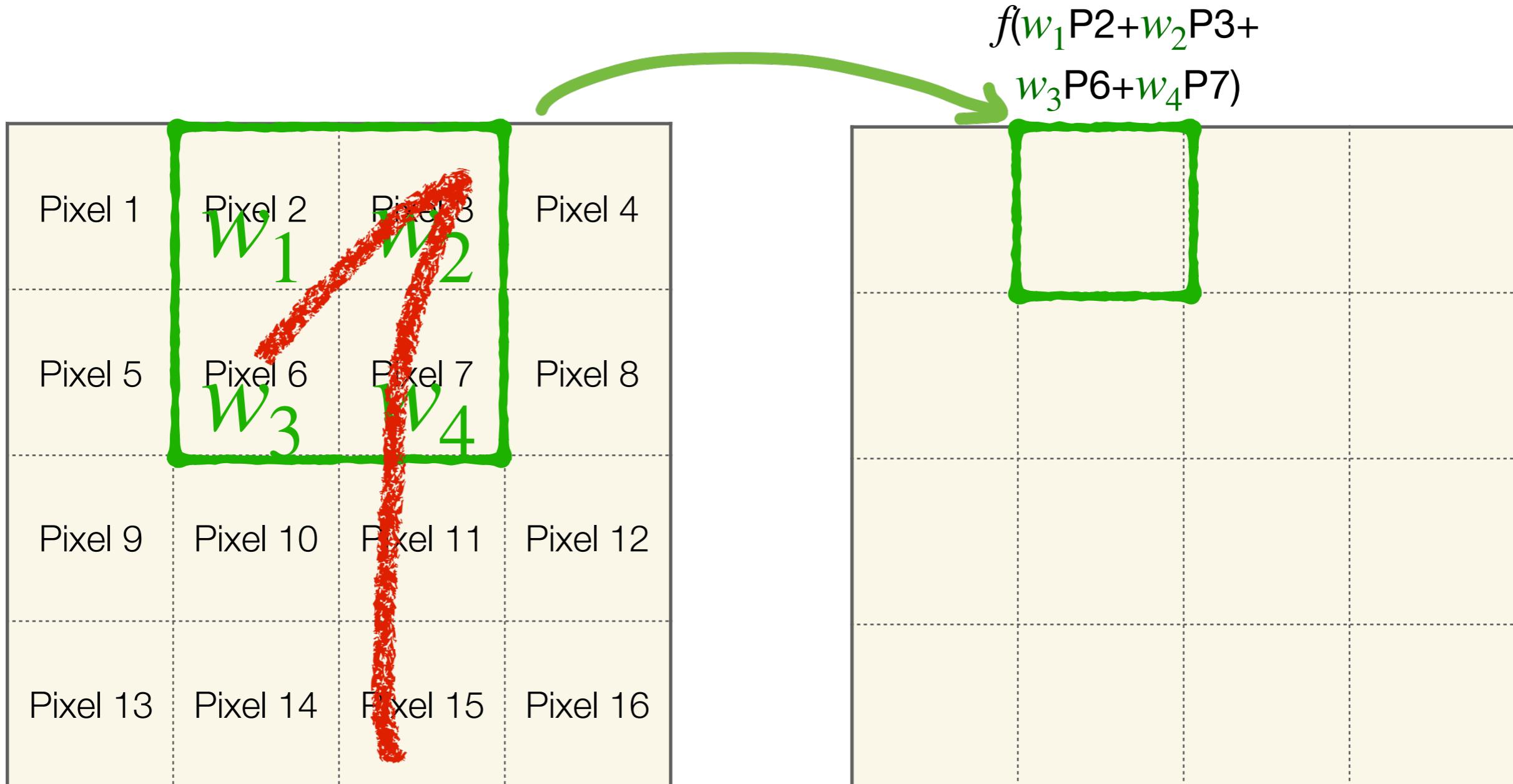


Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”



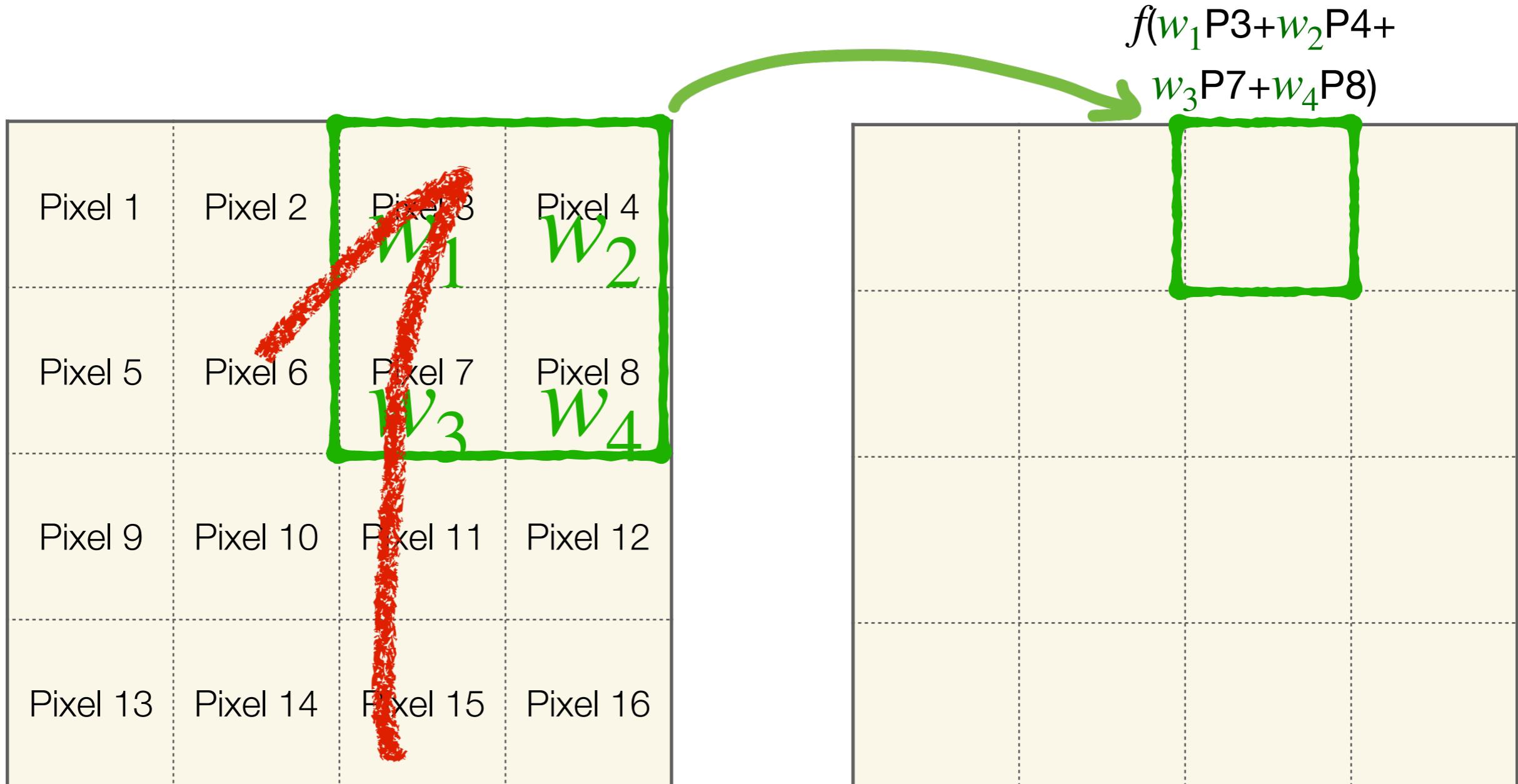
Convolutional Neural Networks



Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”
- Moving over the entire image

Convolutional Neural Networks



Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”
- Moving over the entire image

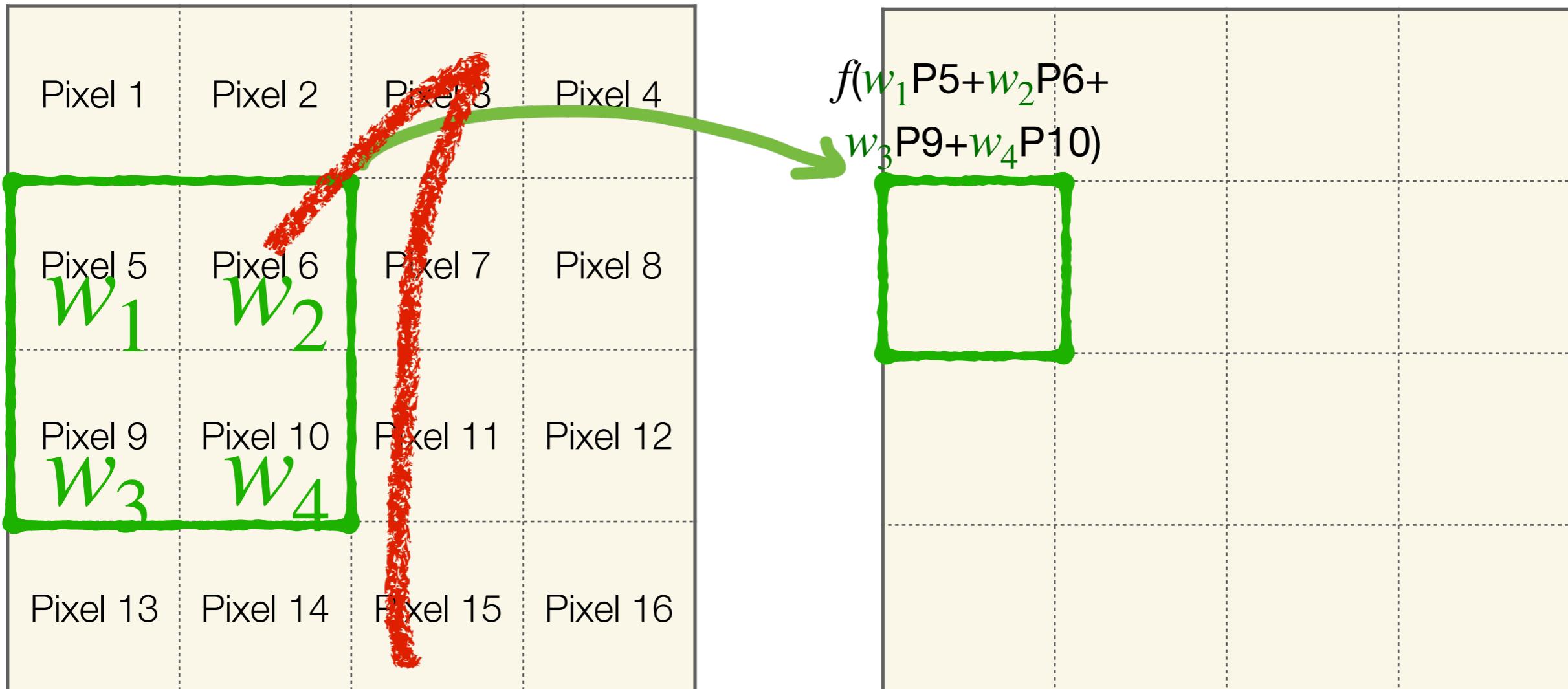
Convolutional Neural Networks

• • •



UNIVERSITY OF
ALBERTA

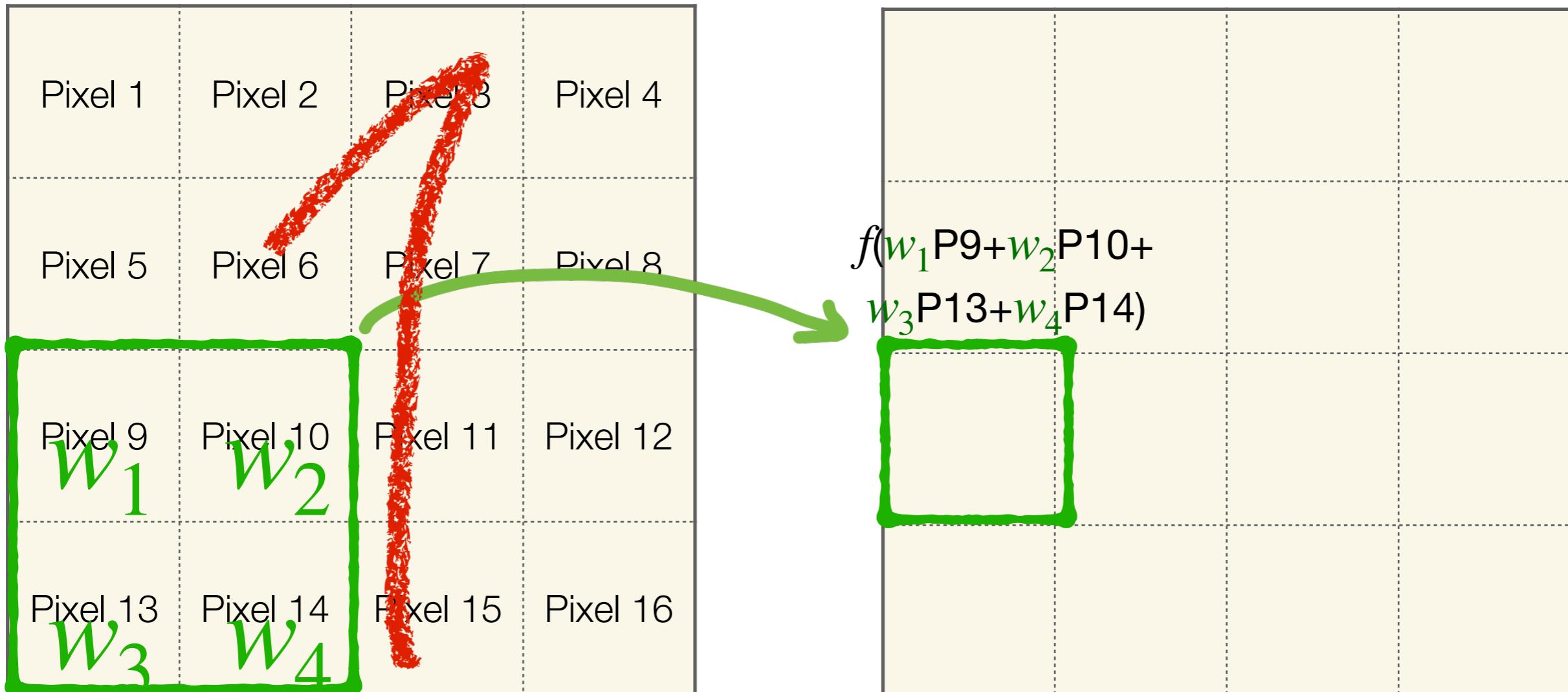
Convolutional Neural Networks



Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”
- Moving over the entire image

Convolutional Neural Networks



Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”
- Moving over the entire image



Convolutional Neural Networks

■

■

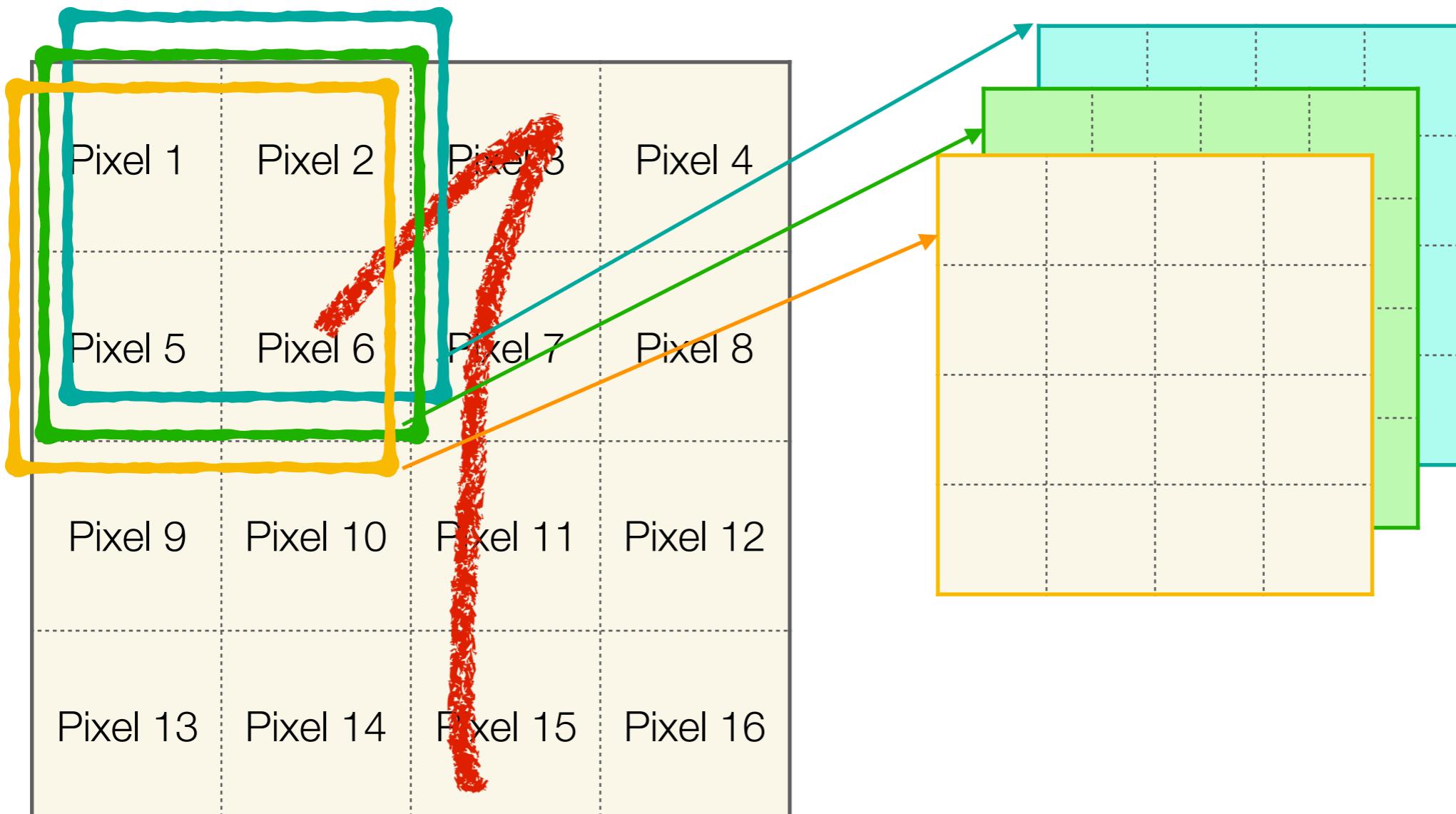
■



Convolutional Neural Networks

-
-
-

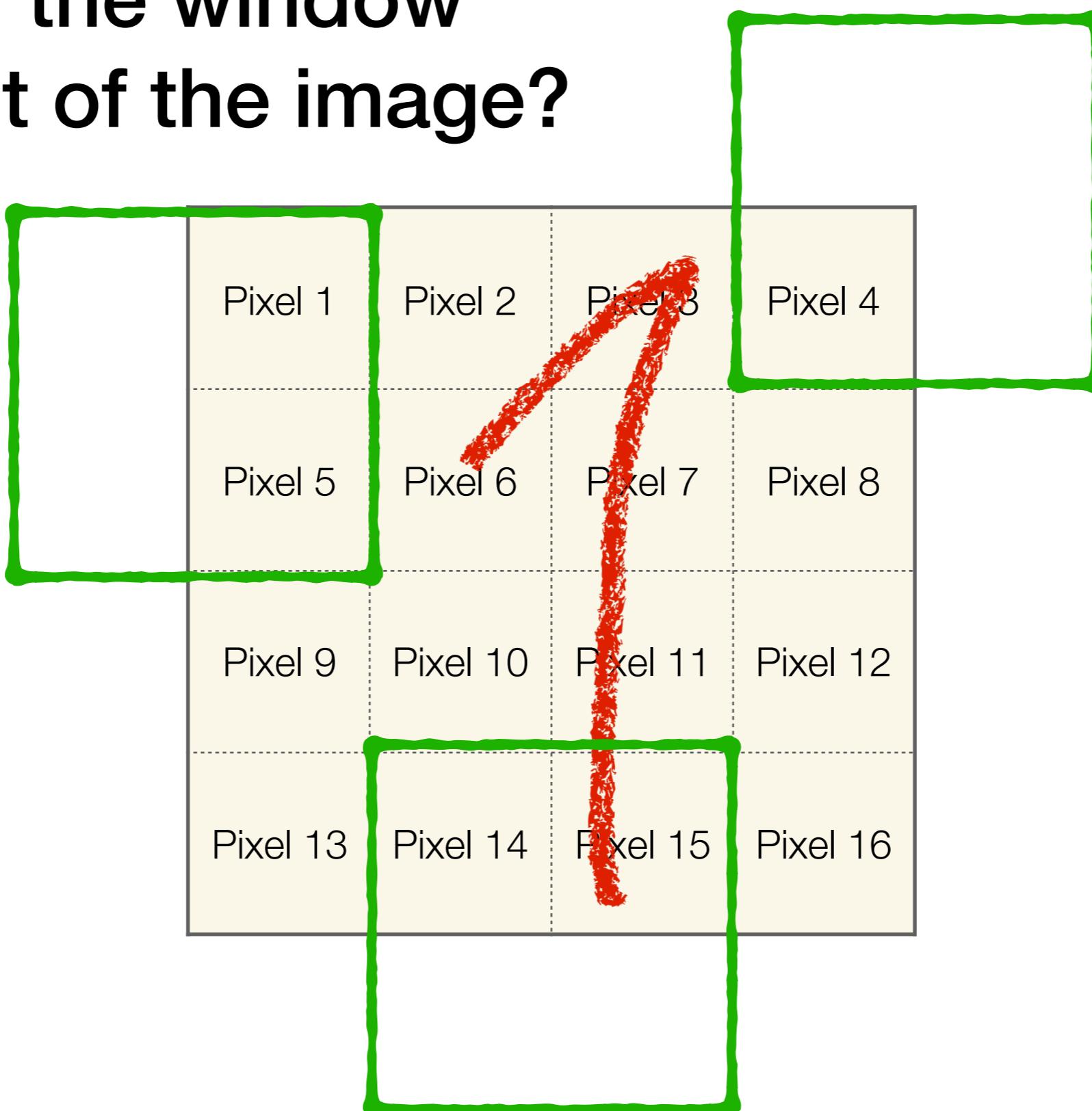
Multiple Filters



Local feature detector (kernel, filter, window)

- Working with a local “neighborhood”
- Moving over the entire image

What if the window moves out of the image?



Padding

- No padding (shrinking)
- Zero-padding
- “Same” padding
- etc.

Other Hyperparameters

- Window size
- Stride

Intuition of CNN

- CNN defines spatial neighbourhoods
 - Our prior knowledge
- The feature of a window is spatial invariant
 - The features of a convolutional layer is spatial equivariant

Pixel 1	Pixel 2	Pixel 3	Pixel 4
Pixel 5	Pixel 6	Pixel 7	Pixel 8
Pixel 9	Pixel 10	Pixel 11	Pixel 12
Pixel 13	Pixel 14	Pixel 15	Pixel 16

Pixel 1	Pixel 2	Pixel 3	Pixel 4
Pixel 5		Pixel 7	Pixel 8
Pixel 9	Pixel 10	Pixel 11	Pixel 12
Pixel 13	Pixel 14	Pixel 15	Pixel 16



Convolution in Signal Processing

- 1-D convolution [continuous]

If $f, g : \mathbb{R} \rightarrow \mathbb{R}$, then $f * g : \mathbb{R} \rightarrow \mathbb{R}$

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(x)g(t - x)dx$$

- 2-D convolution [continuous]

If $f, g : \mathbb{R}^2 \rightarrow \mathbb{R}$, then $f * * g : \mathbb{R}^2 \rightarrow \mathbb{R}$

$$(f * g)(t_1, t_2) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, x_2)g(t_1 - x_1, t_2 - x_2)dx_1dx_2$$



Convolution in Signal Processing

- Continuous [2D]

If $f, g : \mathbb{R}^2 \rightarrow \mathbb{R}$, then $f * * g : \mathbb{R}^2 \rightarrow \mathbb{R}$

$$(f * g)(t_1, t_2) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, x_2)g(t_1 - x_1, t_2 - x_2)dx_1dx_2$$

- Discrete [2D]

If $f, g : \mathbb{Z}^2 \rightarrow \mathbb{R}$, then $f * * g : \mathbb{Z}^2 \rightarrow \mathbb{R}$

$$(f * g)(n_1, n_2) \stackrel{\text{def}}{=} \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} f(k_1, k_2)g(n_1 - k_1, n_2 - k_2)$$



Convolution (2D discrete)

$$f, g : \mathbb{Z}^2 \rightarrow \mathbb{R}$$

$$f * g : \mathbb{Z}^2 \rightarrow \mathbb{R}$$

$$(f * g)(n_1, n_2) \stackrel{\text{def}}{=} \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} f(k_1, k_2) g(n_1 - k_1, n_2 - k_2)$$

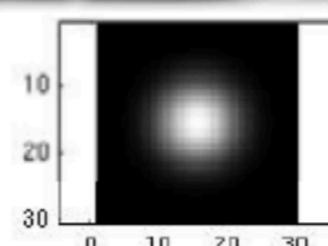
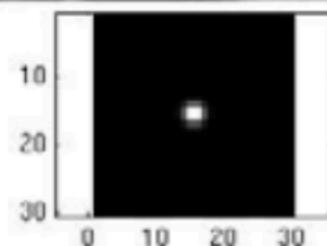
$f(k_1, k_2)$

$f * g(0,0)$

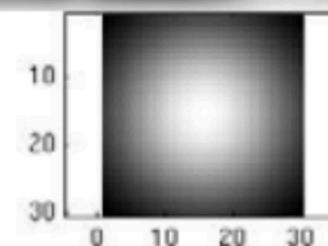
$f(k_1, k_2)$

$f * g(1,1)$

Low-Pass Filter



...



```
for sigma=1:3:10
    h = fspecial('gaussian', fsize, sigma);
    out = imfilter(im, h);
    imshow(out);
    pause;
end
```

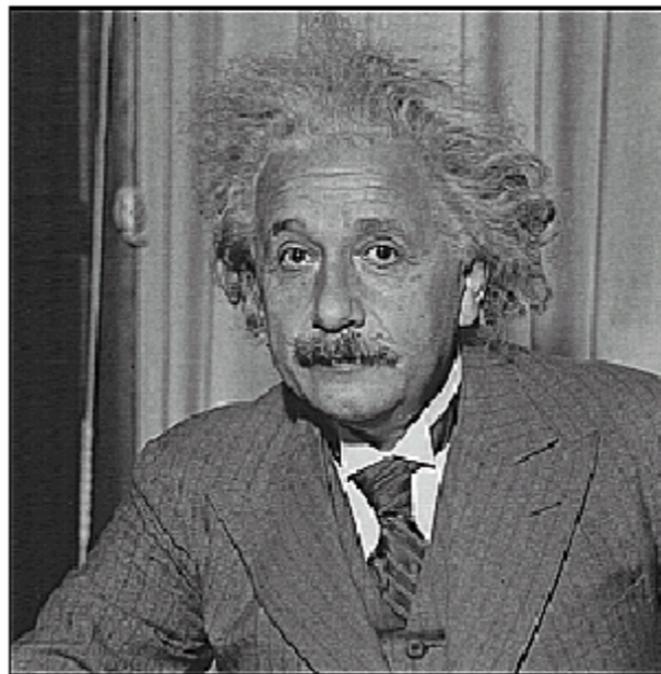
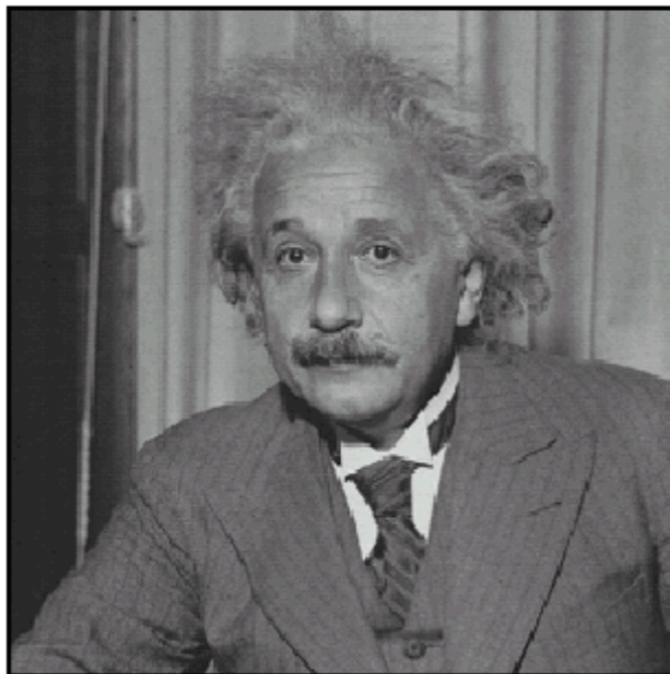
[Slides source: <https://www.cs.toronto.edu/~urtasun/courses/CV/lecture02.pdf>]

[Source: K. Grauman]



UNIVERSITY OF
ALBERTA

High-Pass Filter



before

after

[Source: D. Lowe]

$$\ast \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{array} \right) - \frac{1}{9} \left(\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right)$$

[Slides source: <https://www.cs.toronto.edu/~urtasun/courses/CV/lecture02.pdf>]



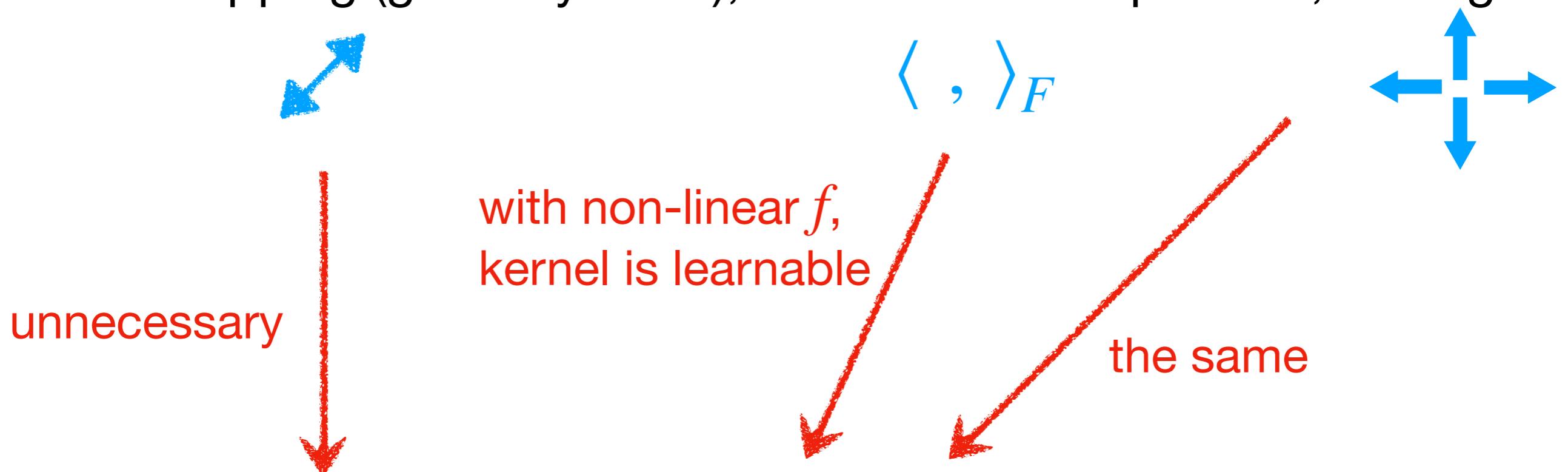
UNIVERSITY OF
ALBERTA

Signal Processing vs. NN

- Convolution in signal processing
 - Flipping (given by def'n), Frobenius inner-product, Sliding

Signal Processing vs. NN

- Convolution in signal processing
 - Flipping (given by def'n), Frobenius inner-product, Sliding



- Convolution in neural networks

Digression: A Few Tips of Learning Math

- Math is important
- No mathematician knows every theorem
 - Lifetime of a mathematician is finite
 - The number of theorems is obviously infinite
- A way to better learn math
 - Get high-level pictures
 - Derive every equation
 - Remember results (and derivations)

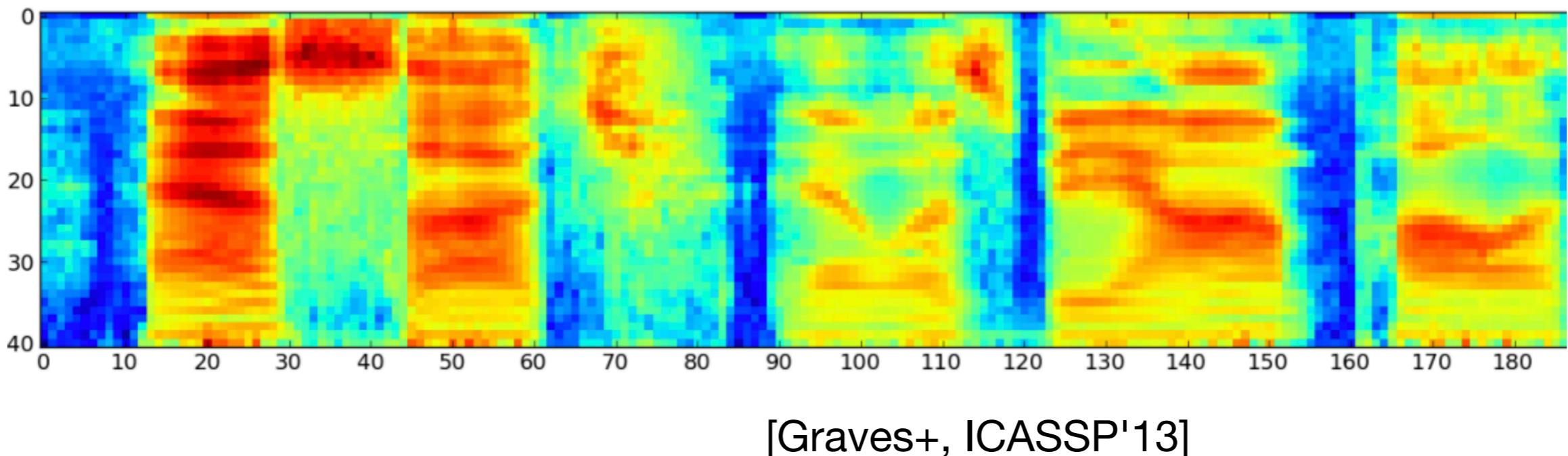
Learned CNN Features



Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

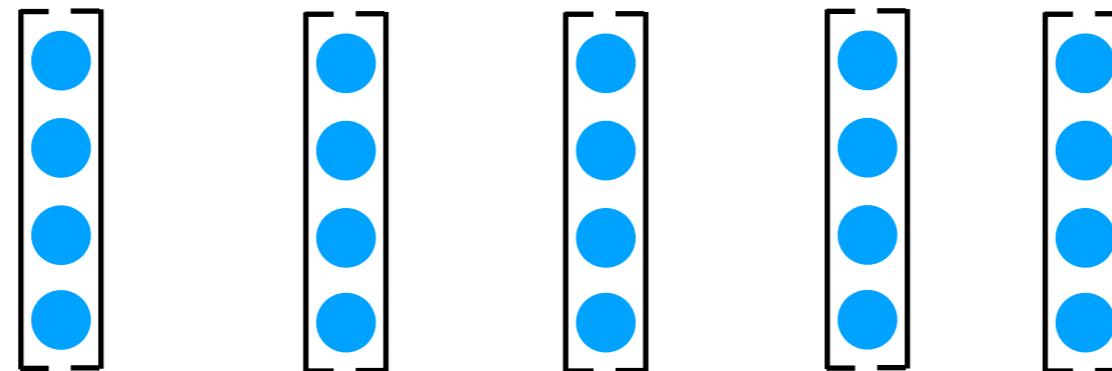
CNN for Speech Processing

- Convolutional window in the frequency domain



CNN for Text Processing

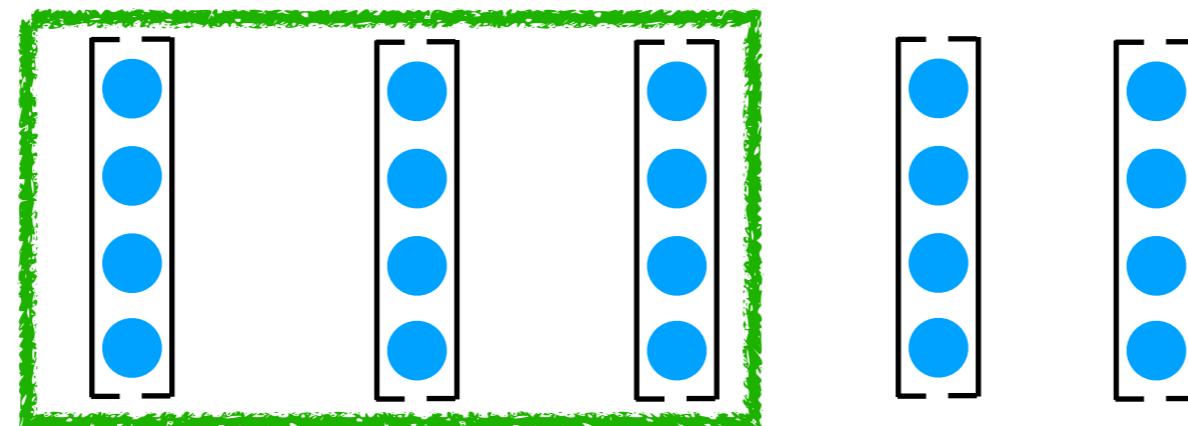
The course is boring .



- Question: Is text (with embeddings) 1D or 2D signal?

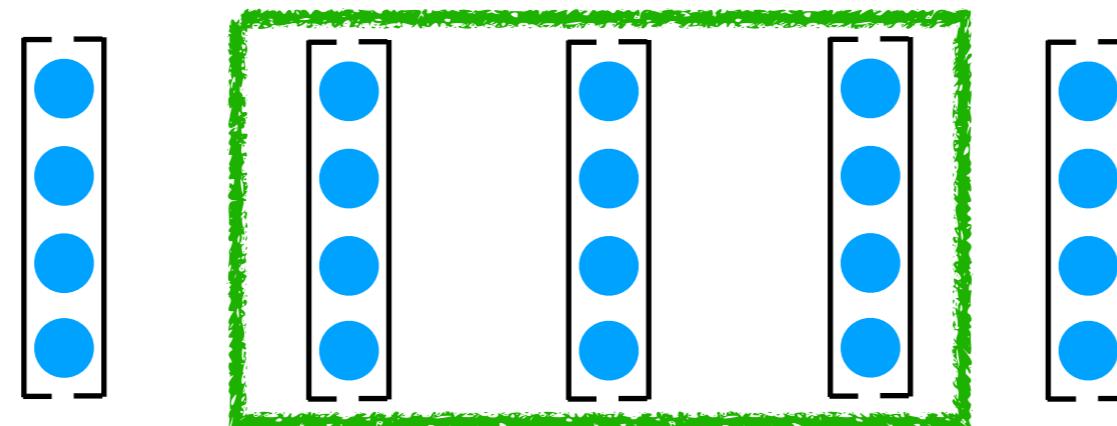
CNN for Text Processing

The course is boring .



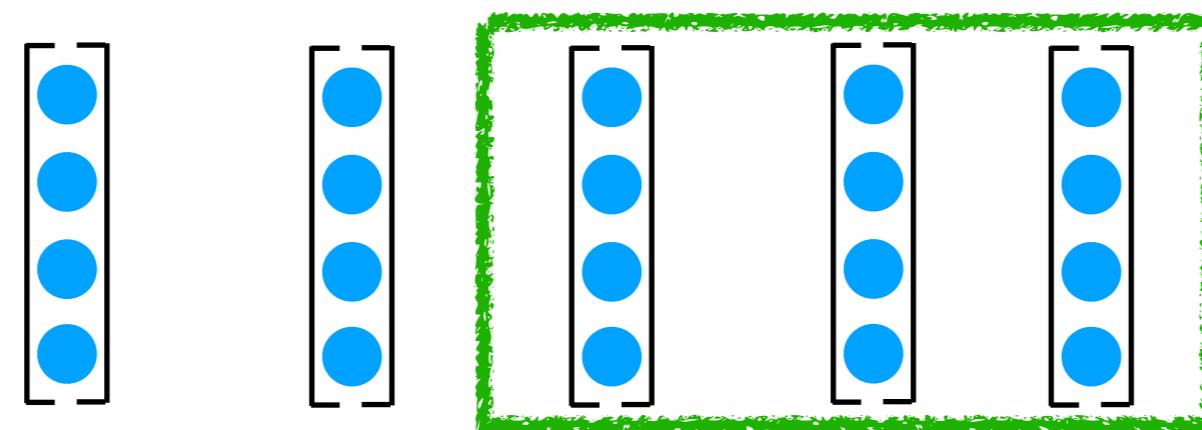
CNN for Text Processing

The course is boring .



CNN for Text Processing

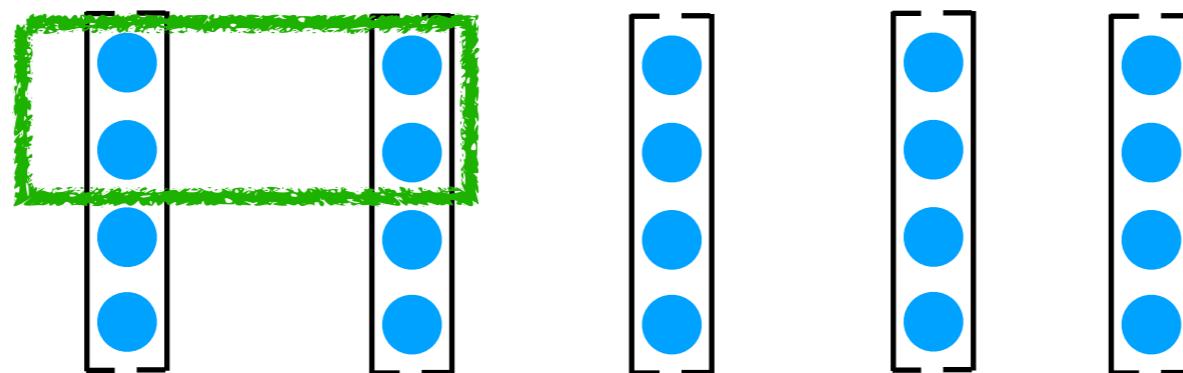
The course is boring .



- Invariance: the features of several consecutive words are the same regardless of the position

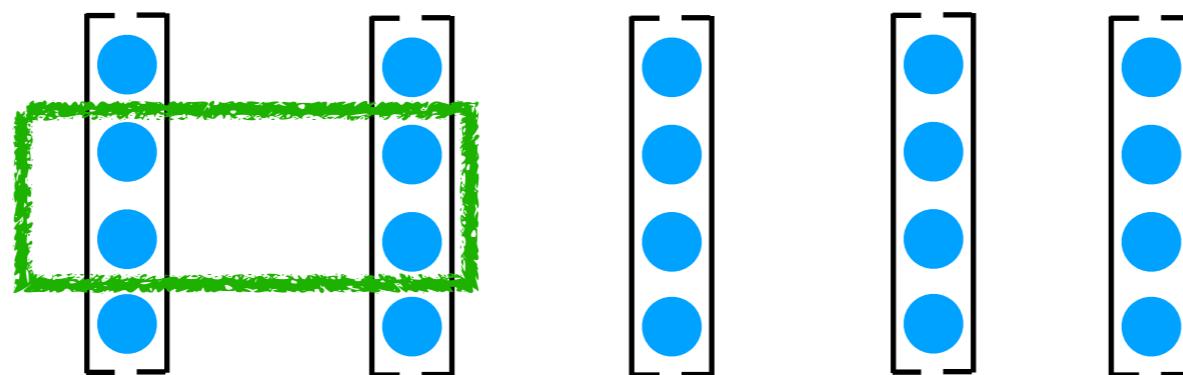
CNN for Text Processing

The course is boring .



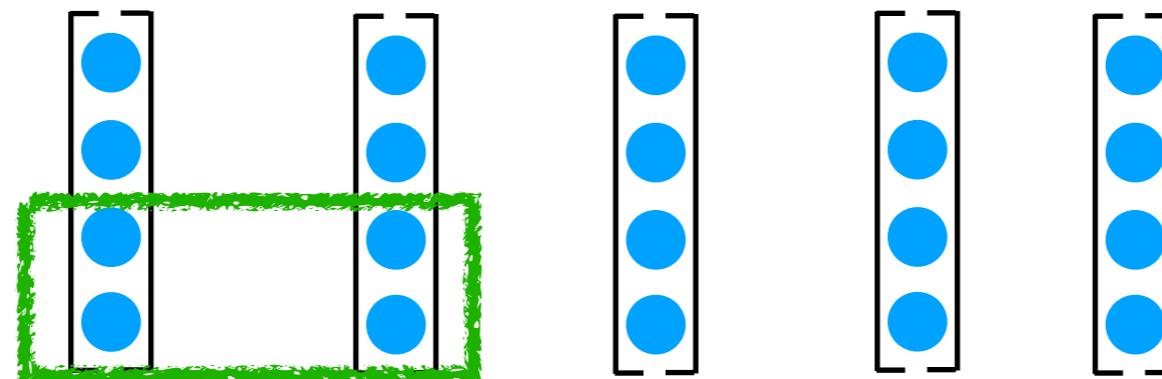
CNN for Text Processing

The course is boring .



CNN for Text Processing

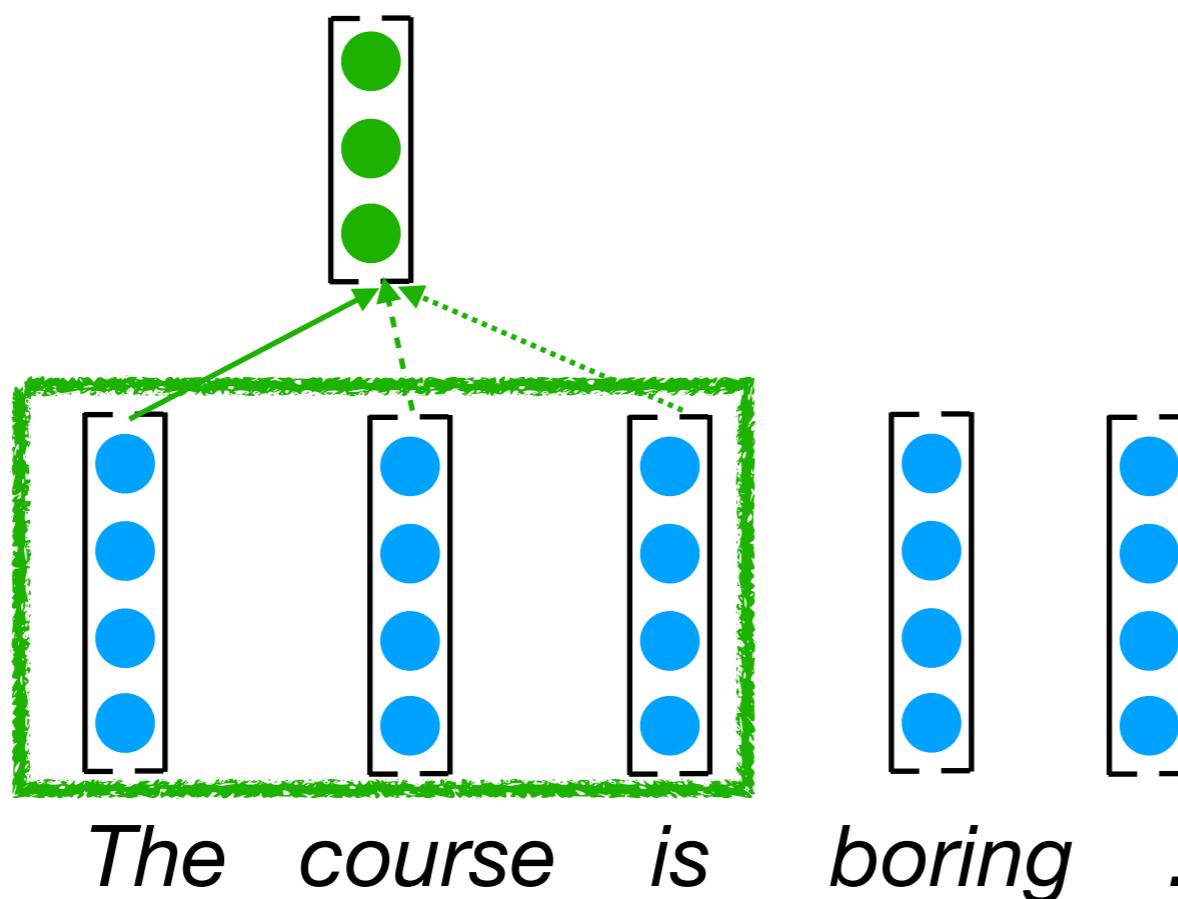
The course is boring .



- Invariance: the features of several consecutive words are the same regardless of the position

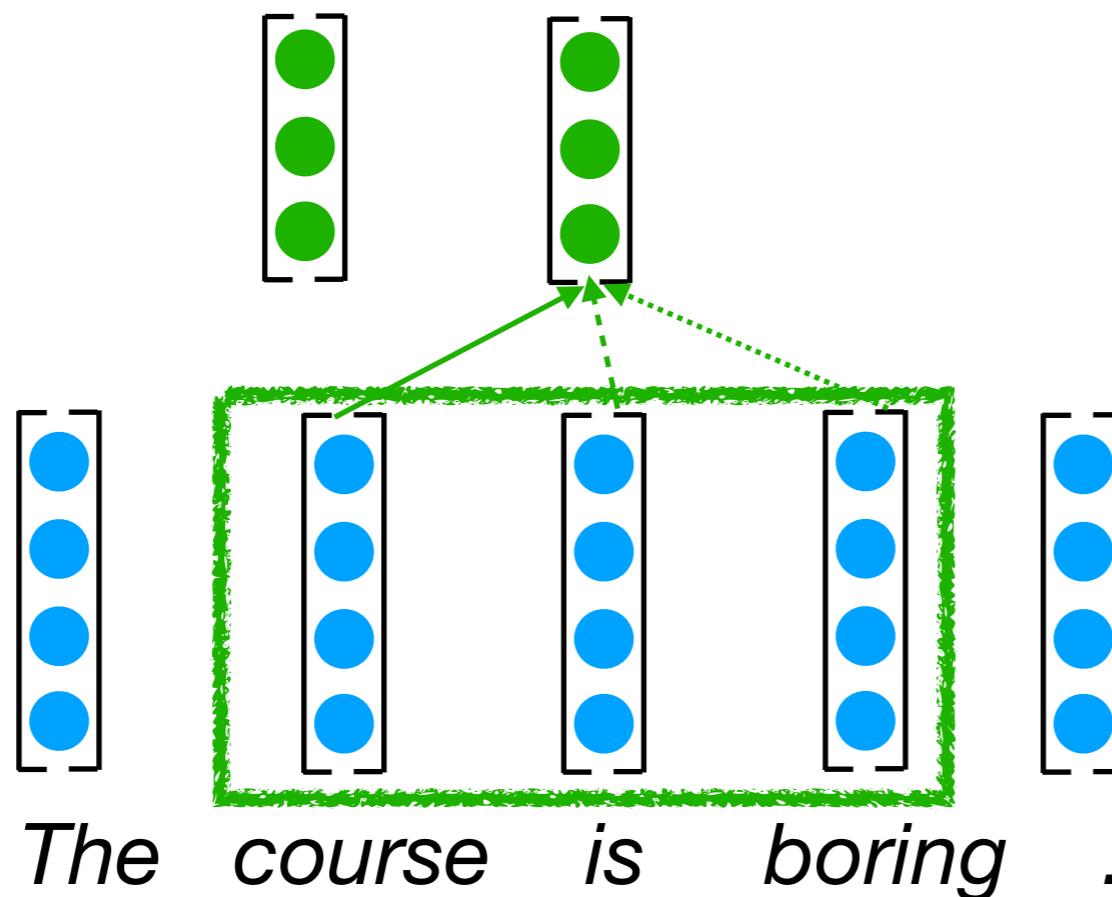
CNN for Text Processing

- Let w_1, \dots, w_n be the words in the window at some position
- Output at the current position: $y = f(W[w_1; \dots; w_n] + b)$



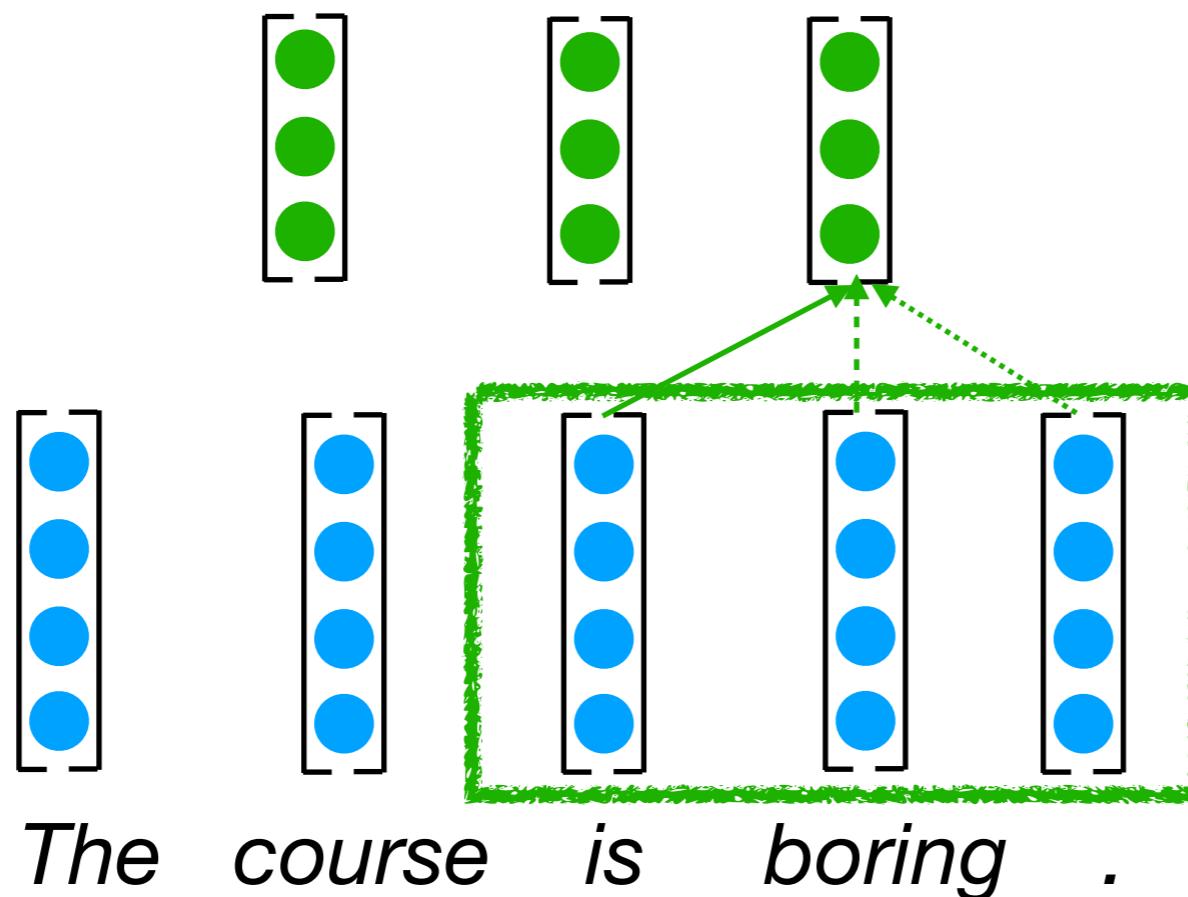
CNN for Text Processing

- Let w_1, \dots, w_n be the words in the window at some position
- Output at the current position: $y = f(W[w_1; \dots; w_n] + b)$



CNN for Text Processing

- Let w_1, \dots, w_n be the words in the window at some position
- Output at the current position: $y = f(W[w_1; \dots; w_n] + b)$

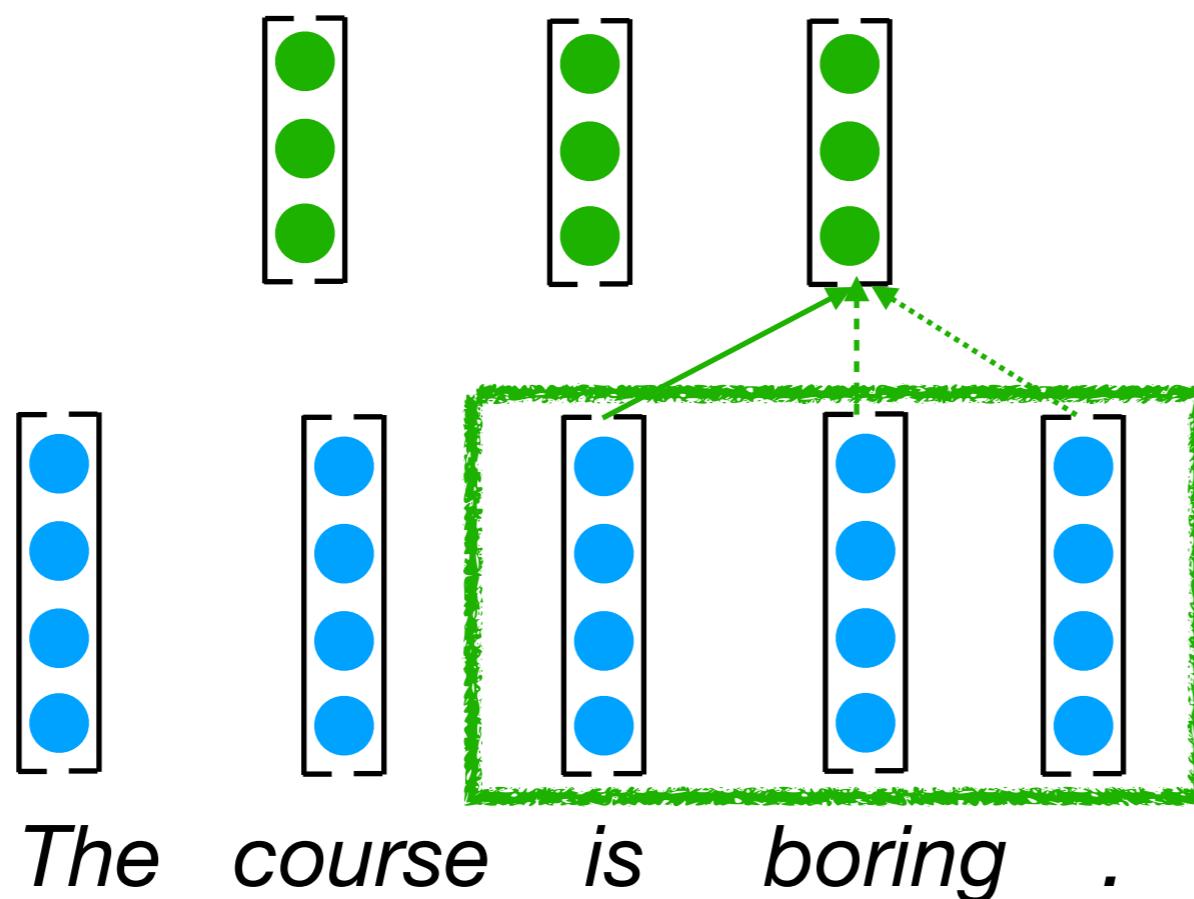


Suppose we have no padding
(which is a bad practice)



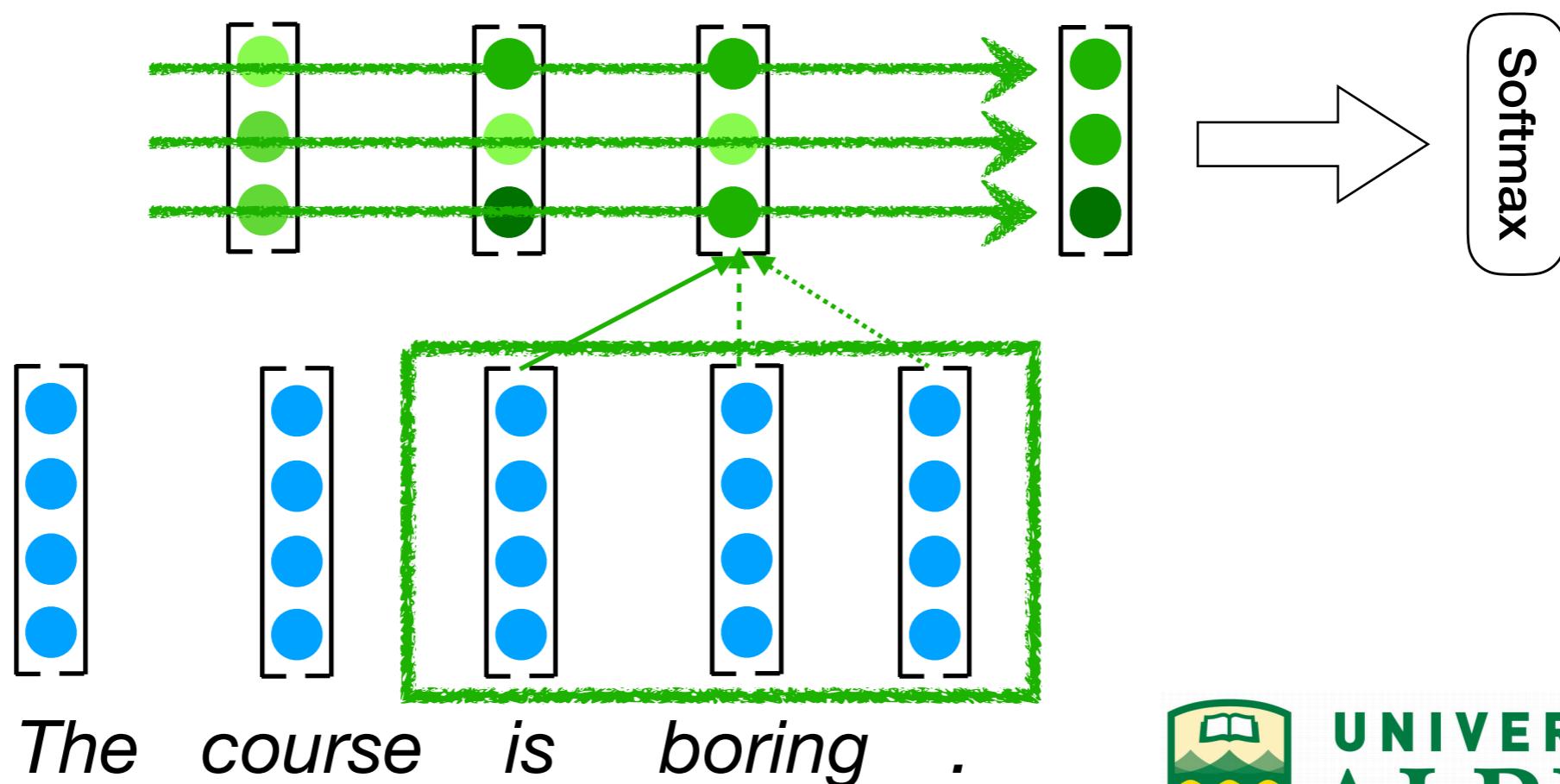
CNN for Text Processing

- Next problem:
 - Length varies among sentences

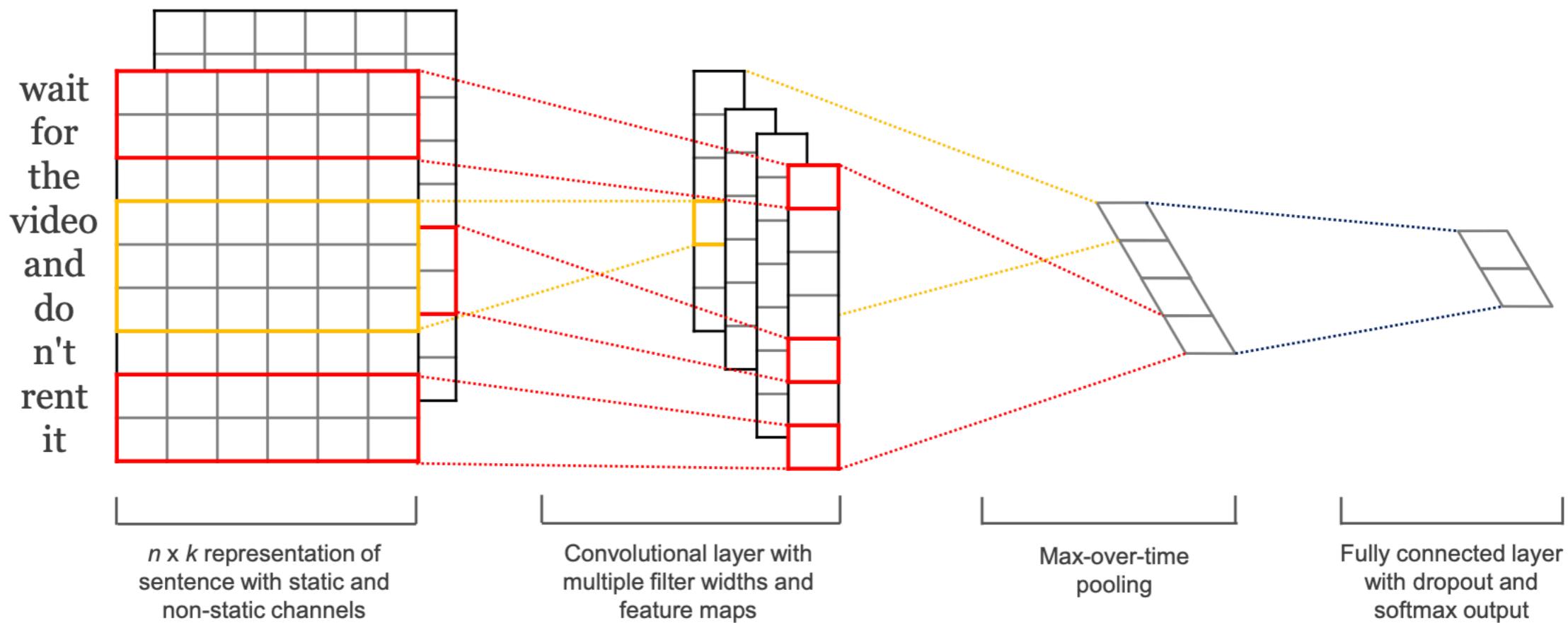


Pooling

- Max pooling: Takes the maximum value in each dimension
 - Intuition: To what extent is a feature satisfied most?
- Sum pooling, average pooling, etc.

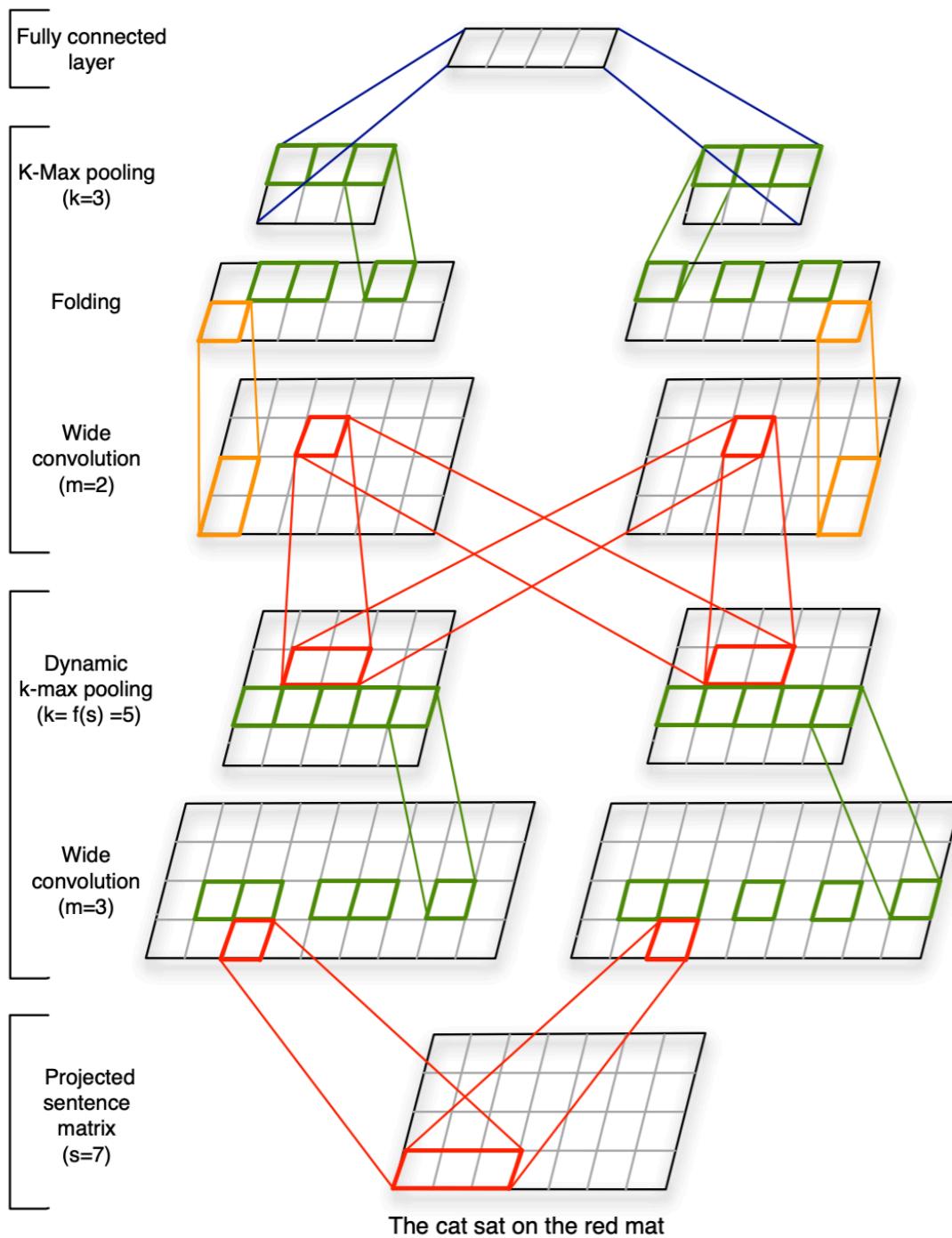


Example



Yoon Kim, Convolutional Neural Networks for Sentence Classification. In *EMNLP 2014*.

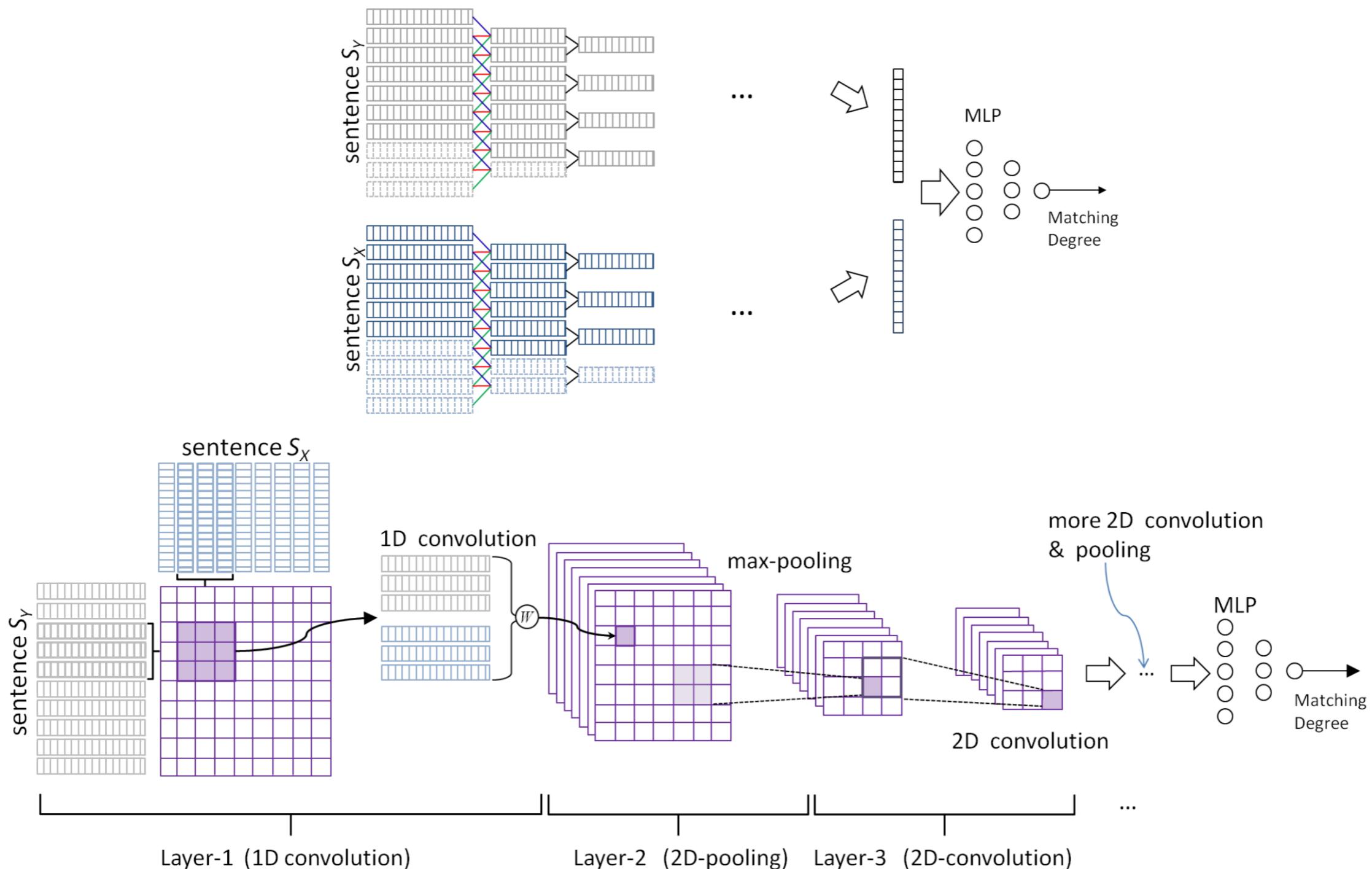
Example



Kalchbrenner et al, A Convolutional Neural Network for Modelling Sentences. In *ACL* 2014.



Example: Sentence Matching



Hu et al., Convolutional neural network architectures for matching natural language sentences. In *NIPS* 2014.



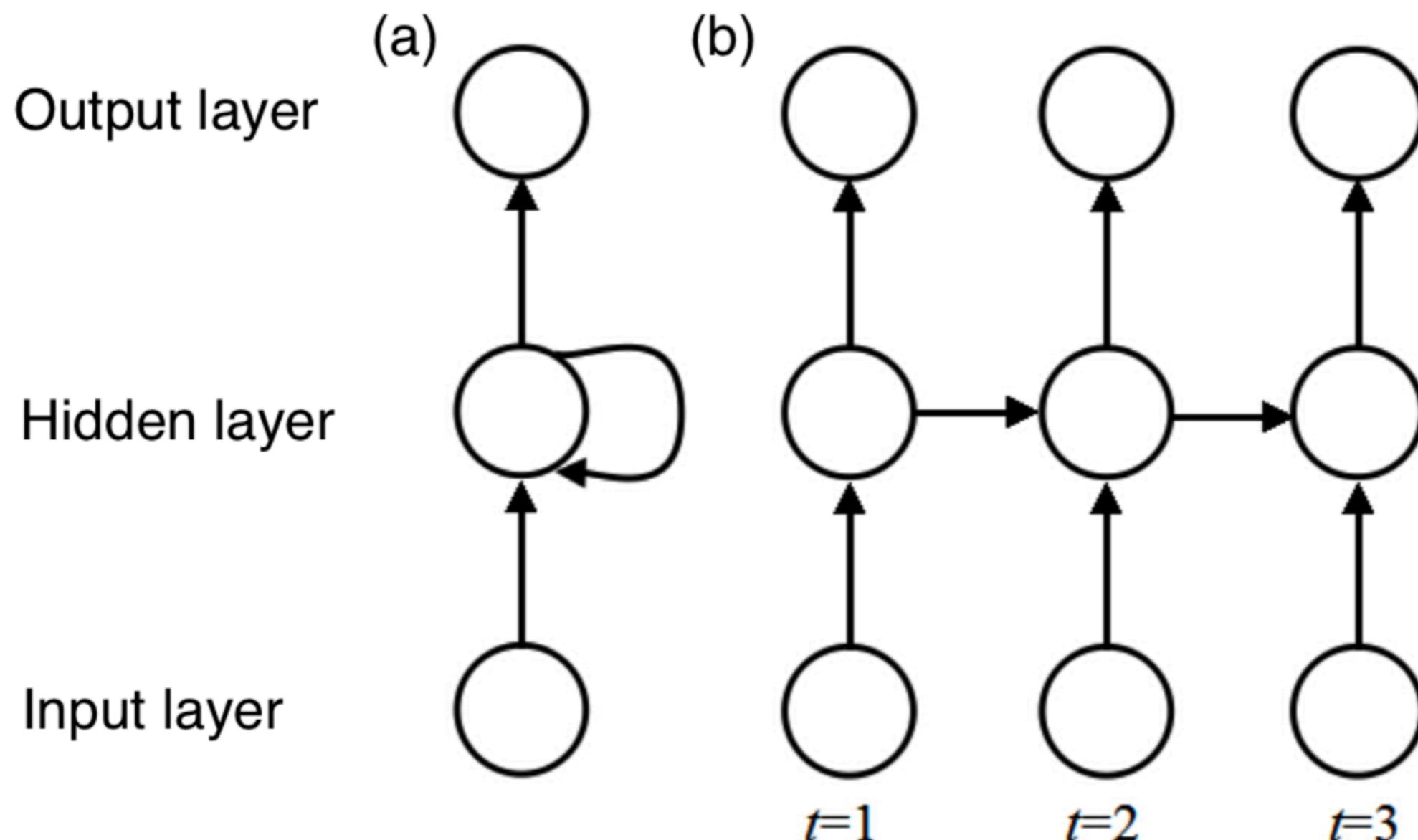
UNIVERSITY OF
ALBERTA

Problems with CNNs

- Direct interaction only within a window
- Too much information loss with max pooling

Want: Global dependency modelling for potentially infinitely long sequences

Recurrent Neural Networks



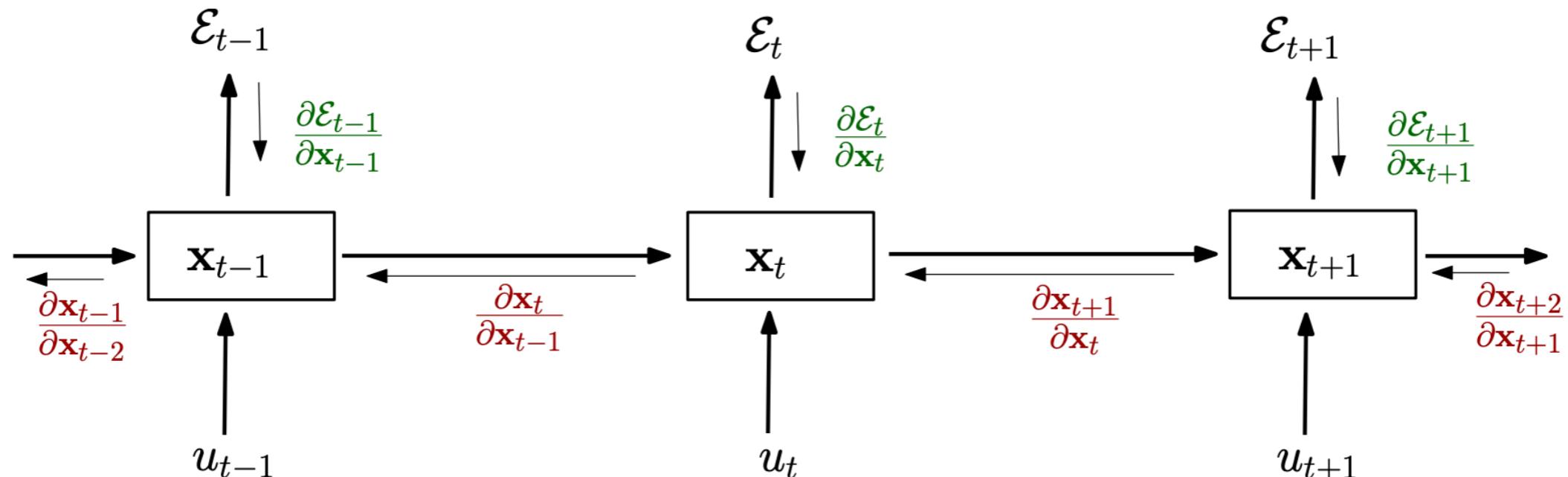
Recurrent Neural Networks

- Vanilla RNN

$$\mathbf{h}^{(t)} = f(W_h \mathbf{h}^{(t-1)} + W_x \mathbf{x}^{(t)} + \mathbf{b})$$

- Learning: Backpropagation through time (BPTT)
 - Just a fancy terminology
- Gradient vanishing or explosion
 - BP is a linear system
 - FP is a non-linear system (potentially chaotic)

Gradient Vanishing or Explosion

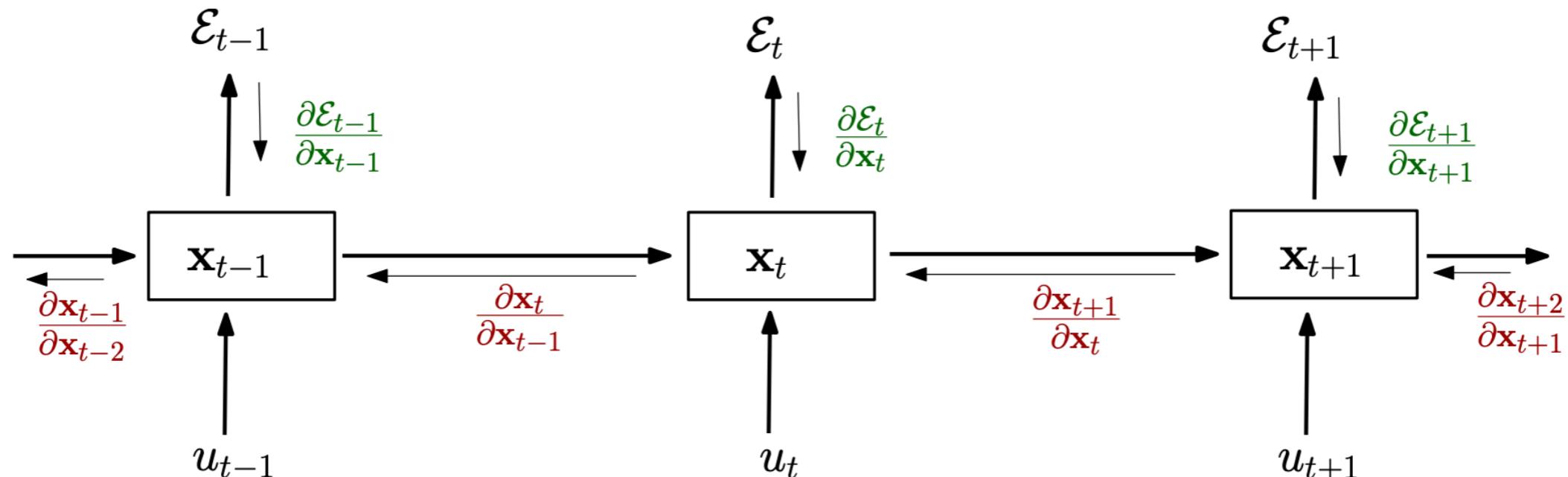


$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \leq t \leq T} \frac{\partial \mathcal{E}_t}{\partial \theta}$$

$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right)$$

$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1}))$$

Gradient Vanishing or Explosion



$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \leq t \leq T} \frac{\partial \mathcal{E}_t}{\partial \theta}$$

$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right)$$

$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \boxed{\prod_{t \geq i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1}))}$$

$$\| \quad \| \leq \prod \| \quad \| \quad \| \quad \| \quad \| \quad \|$$

L_2 -norm of a matrix is the max eigenvalue

What's wrong?

- BP doesn't give exact gradient for RNN
- Numerical underflow/overflow causes gradient vanishing and explosion
- RNNs do not have enough model capacity
- Exact gradient isn't what we want

Long Short Term Memory

- Keeps a cell c_t and hidden state h_t

Input gate

$$i_t = \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i)$$

Forget gate

$$f_t = \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f)$$

Output gate

$$o_t = \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o)$$

Cell content

$$g_t = \tanh(W_g \mathbf{x}_t + U_g \mathbf{h}_{t-1} + \mathbf{b}_g)$$

Cell

$$c_t = i_t \circ g_t + f_t \circ c_{t-1}$$

Hidden state

$$h_t = o_t \circ \tanh(c_t)$$

Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural Computation*, 9(8), pp.1735-1780.



UNIVERSITY OF
ALBERTA

Gated Recurrent Unit

- Keeps a cell c_t and hidden state \mathbf{h}_t

$$\mathbf{r} = \sigma(W_r \mathbf{h}_{t-1})$$

$$\mathbf{z} = \sigma(W_z \mathbf{h}_{t-1})$$

$$\tilde{\mathbf{h}} = \tanh(W_x \mathbf{h}_{t-1} + W_g(\mathbf{r} \odot \mathbf{h}_{t-1}))$$

$$\mathbf{h}_t = (1 - \mathbf{z}) \odot \mathbf{h}_{t-1} + \mathbf{z} \odot \tilde{\mathbf{h}}_t$$

Intuition of LSTM/GRU

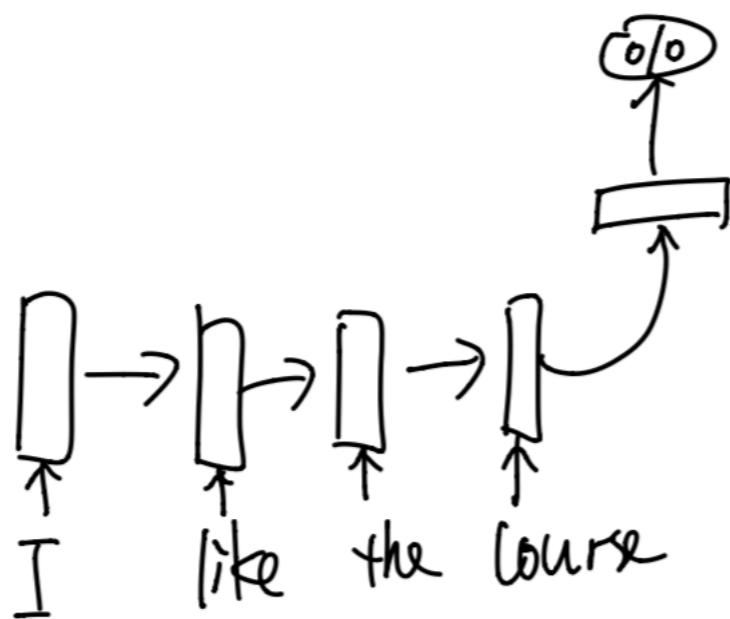
- **No Free-Lunch Theorem:** If someone's intuition can explain everything, then such intuition is not useful.
- If we have too much belated intuition, our intuition is overfitting to experiments.

RNN Usages

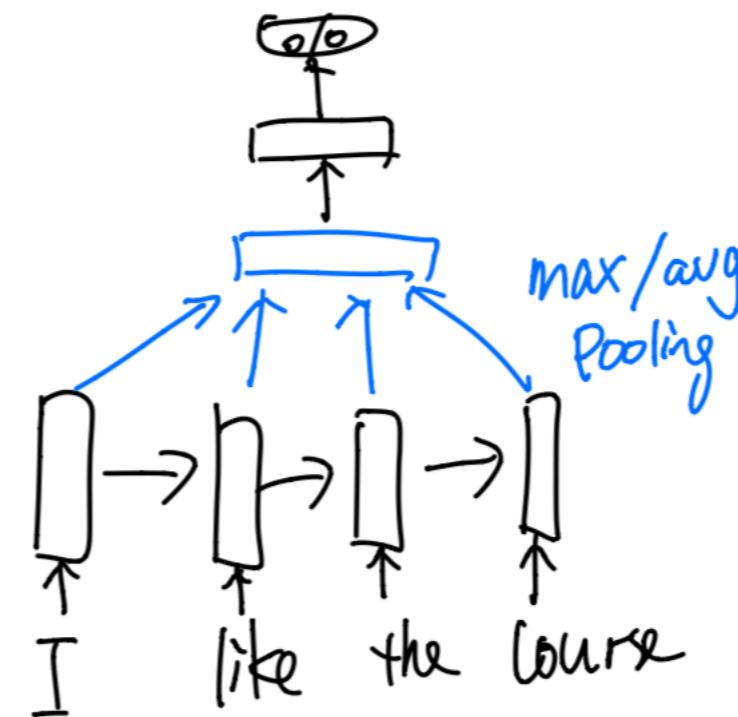
- Single output (e.g., classification)
- Sequential output (one-one corresponding to input tokens)
- Sequential output (no correspondence)

Single Output

Using the last hidden state



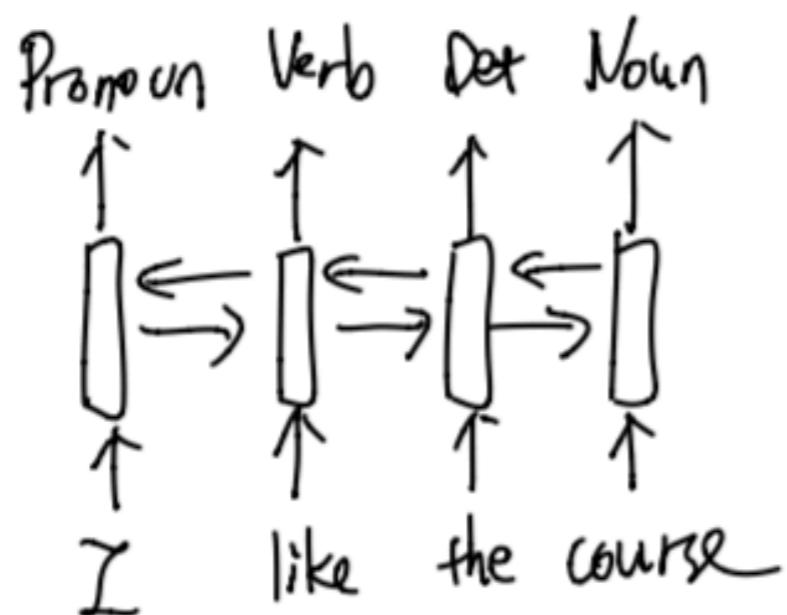
Pooling of all hidden states



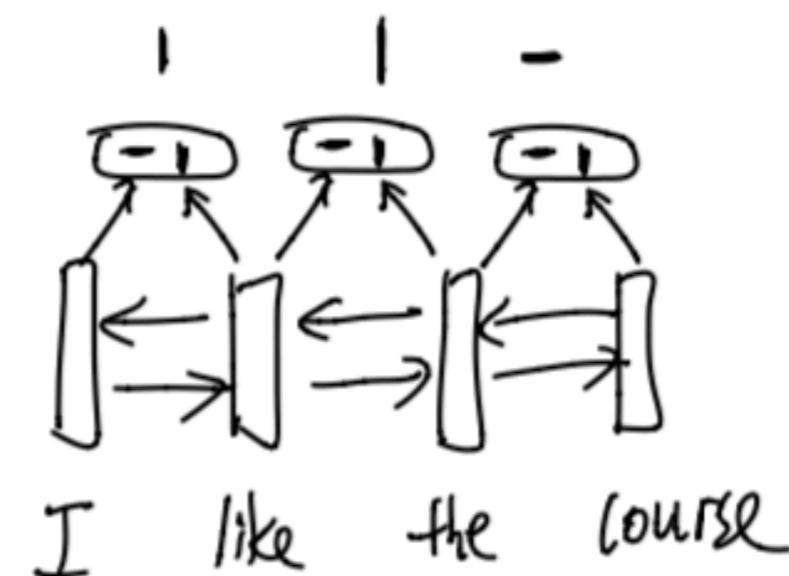
Sequential Output

(one-one corresponding to input tokens)

POS tagging

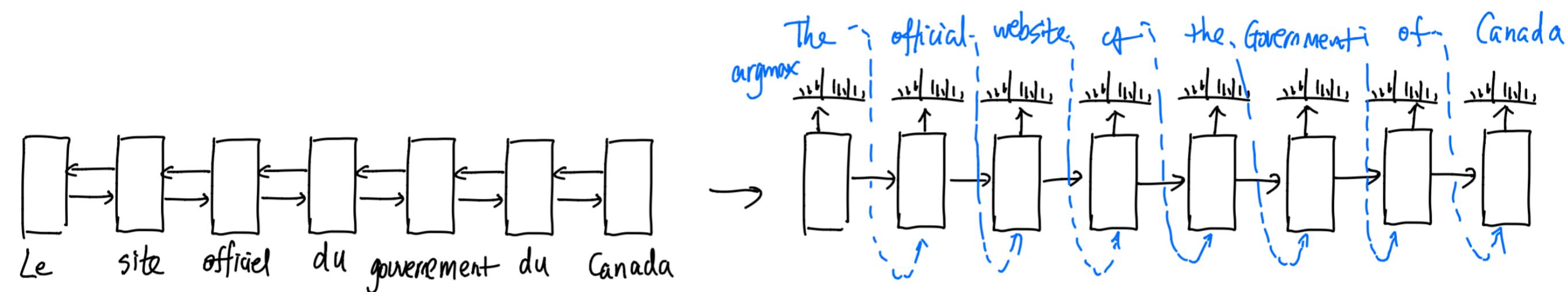


Chunking



Sequential Output

(No correspondence to input tokens)



Canada.ca

Le site officiel du gouvernement du Canada

Canada.ca

The official website of the Government of Canada

[Source: canda.ca]



UNIVERSITY OF
ALBERTA

Seq2Seq

- **Question:**

Why do we feed back the generated words?

How can we train Seq2Seq models?

How do we do inference?

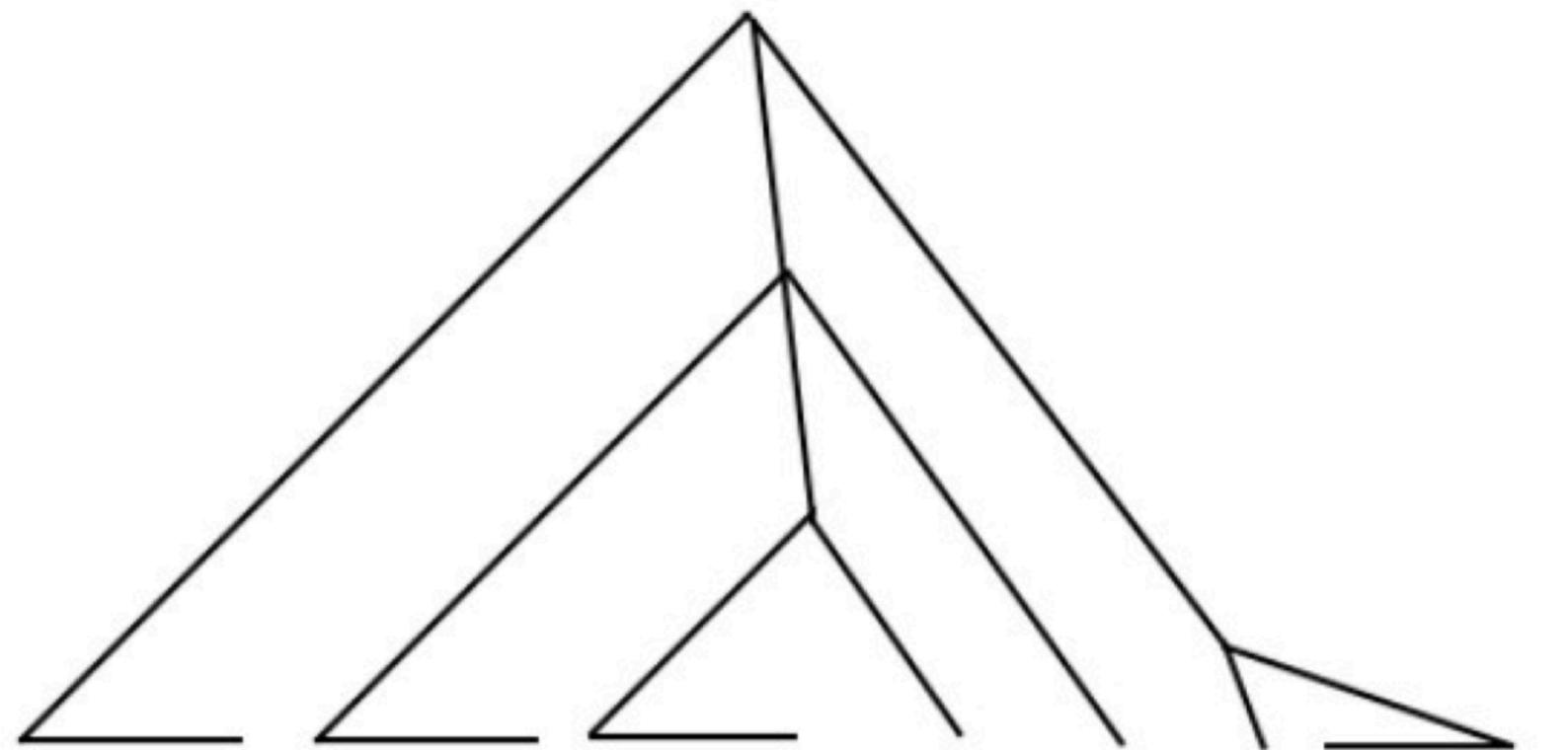


Prior Knowledge

- CNN
 - Spatial neighborhood
 - Sliding window
- RNN
 - Ordered information
 - Sequential processing

Why parse trees may be important?

Tree structure



Convolution



Pinker, Steven. *The Language Instinct: The New Science of Language and Mind*. Penguin UK, 1995.



UNIVERSITY OF
ALBERTA

Recursive Propagation

- Perception-like interaction

$$p = f(W[c_1 \ c_2]^T)$$

- Matrix-vector interaction

$$p_1 = f\left(W \begin{bmatrix} Cb \\ Bc \end{bmatrix}\right), P_1 = f\left(W_M \begin{bmatrix} B \\ C \end{bmatrix}\right)$$

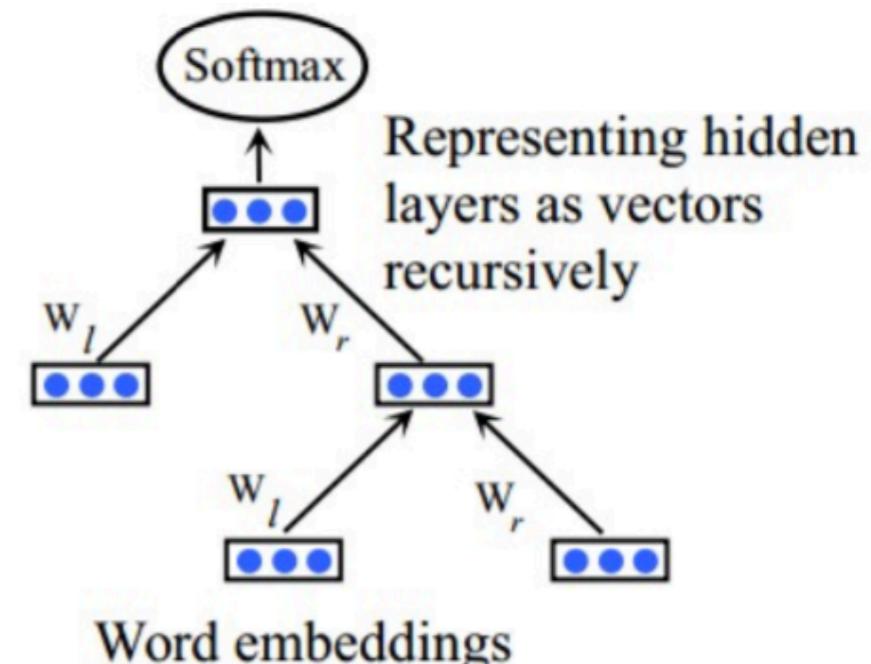
- Tensor interaction

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right)$$

Socher R, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions. EMNLP, 2011

Socher, R, et al. "Semantic compositionality through recursive matrix-vector spaces." EMNLP-CoNLL, 2012.

Socher, R, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." EMNLP, 2013.



Even More Interaction

- LSTM interaction

Tai, Kai Sheng, Richard Socher, and Christopher D. Manning. "Improved semantic representations from tree-structured long short-term memory networks." ACL, 2015

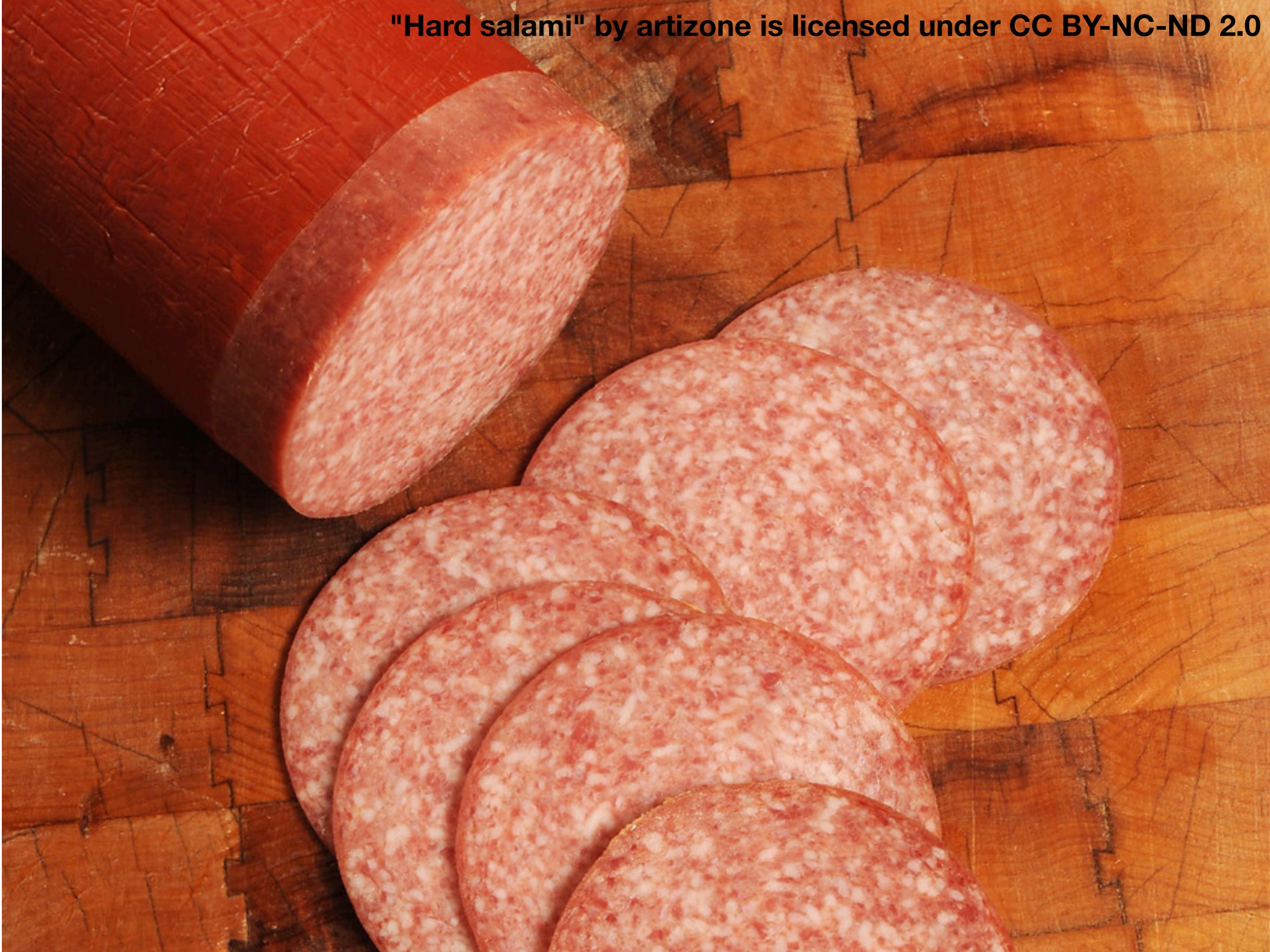
Zhu, Xiaodan, Parinaz Sobhani, and Hongyu Guo. "Long short-term memory over recursive structures." ICML, 2015.

Le, Phong, and Willem Zuidema. "Compositional distributional semantics with long short term memory." arXiv:1503.02510 (2015).

$$\begin{aligned}\tilde{h}_j &= \sum_{k \in C(j)} h_k, \\ i_j &= \sigma \left(W^{(i)} x_j + U^{(i)} \tilde{h}_j + b^{(i)} \right), \\ f_{jk} &= \sigma \left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)} \right), \\ o_j &= \sigma \left(W^{(o)} x_j + U^{(o)} \tilde{h}_j + b^{(o)} \right), \\ u_j &= \tanh \left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)} \right), \\ c_j &= i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k, \\ h_j &= o_j \odot \tanh(c_j),\end{aligned}$$



"Hard salami" by artizone is licensed under CC BY-NC-ND 2.0



Structure

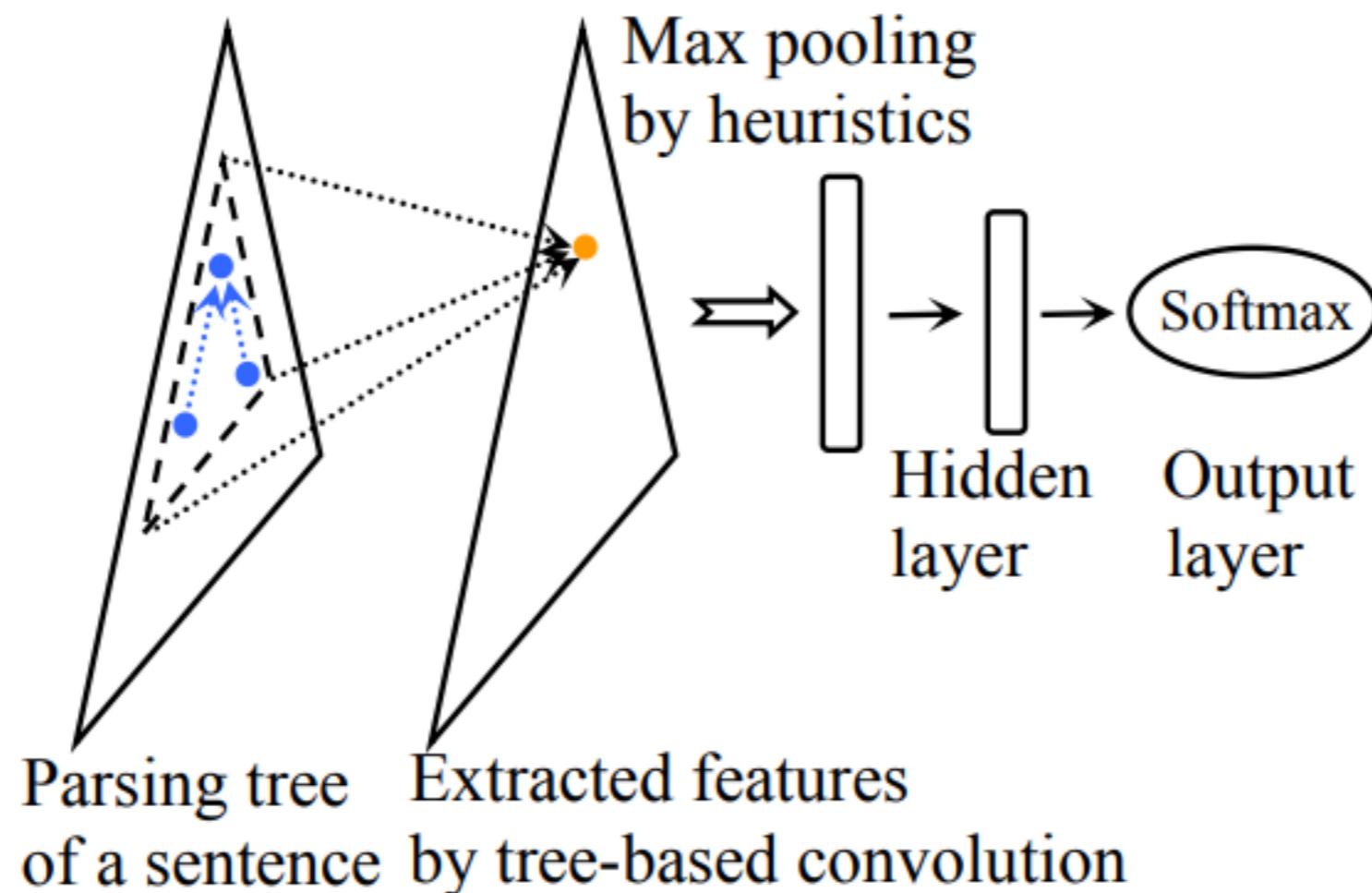
Information
aggregation

	Sequence	Tree
Sliding window	CNN	RNN
Iterative	?	TreeRNN

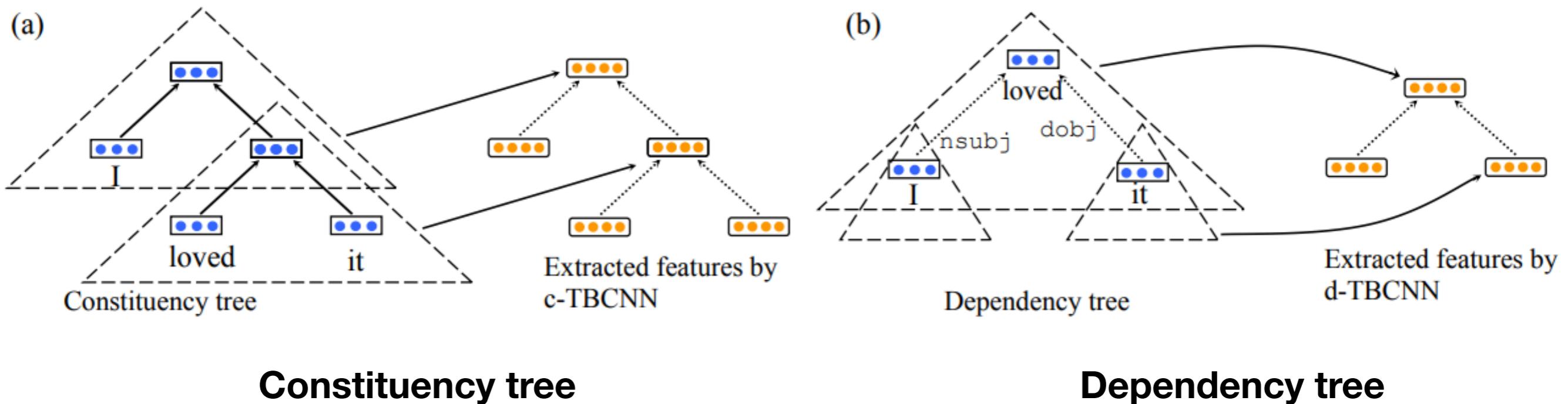


UNIVERSITY OF
ALBERTA

Tree-Based Convolution



Tree-Based Convolution



Constituency tree

Dependency tree

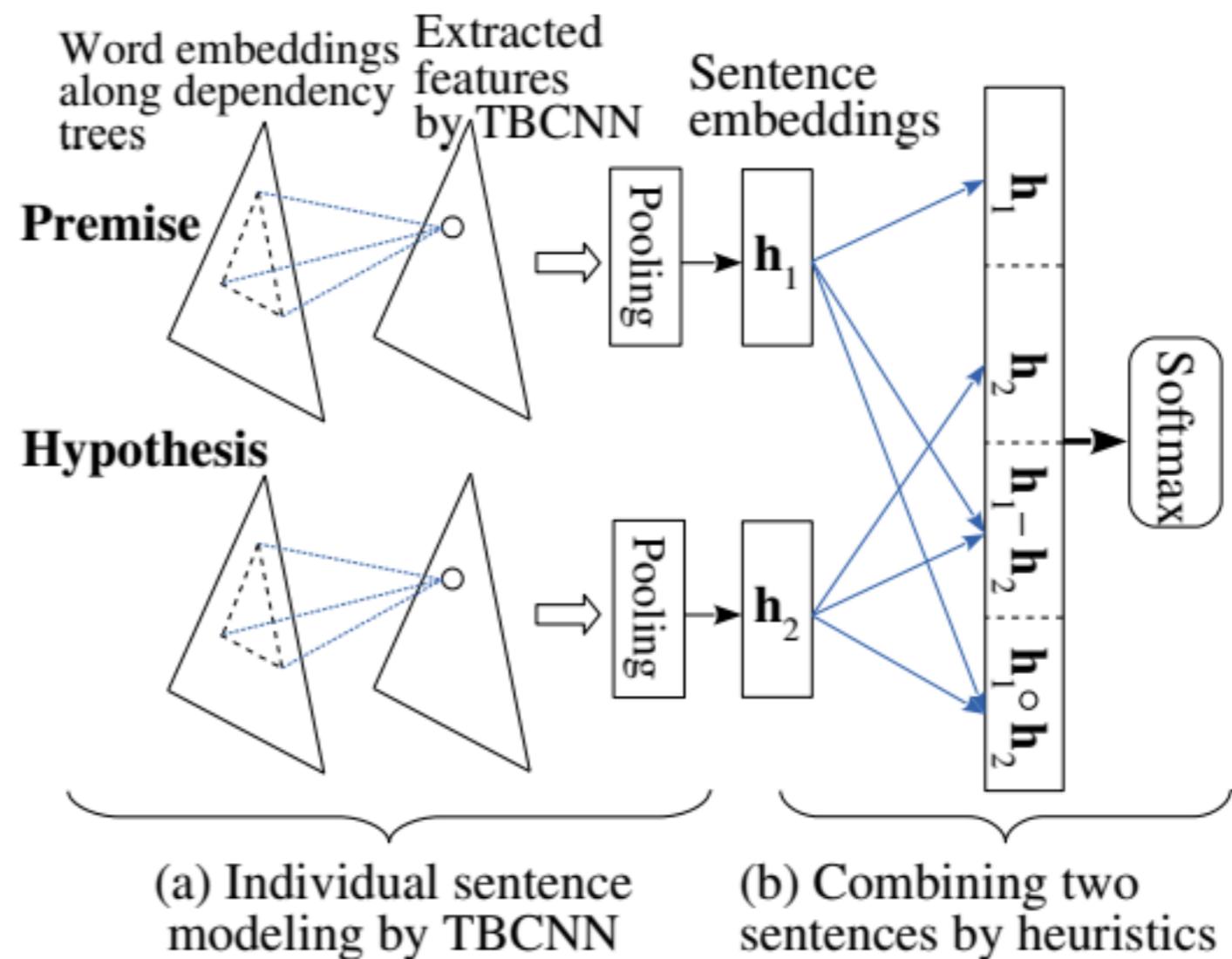
Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, Zhi Jin. Discriminative neural sentence modeling by tree-based convolution. In *EMNLP*, 2015.



UNIVERSITY OF
ALBERTA

Tree-Based Convolution

Sentence Matching



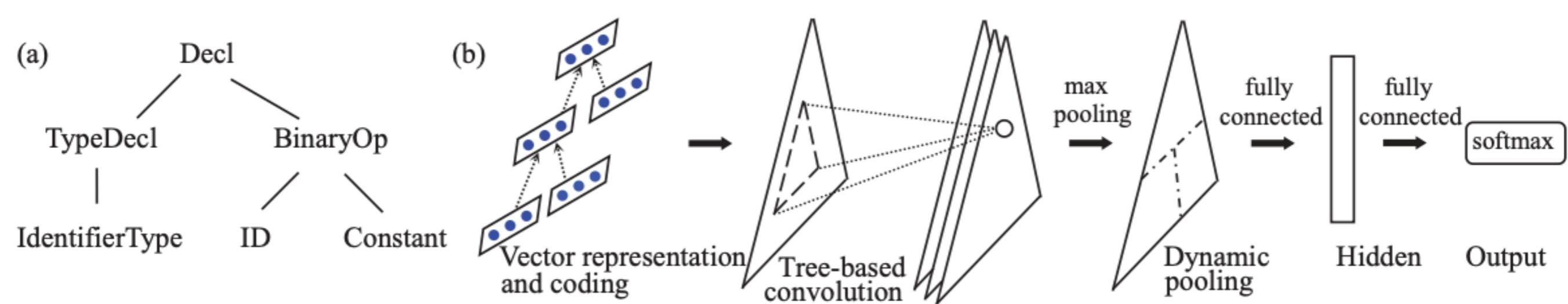
Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, Zhi Jin. Natural language inference by tree-based convolution and heuristic matching. In ACL, 2016.



UNIVERSITY OF
ALBERTA

Tree-Based Convolution

Programming language: Abstract Syntax Tree



Lili Mou, Ge Li, Lu Zhang, Tao Wang, Zhi Jin. Convolutional neural networks over tree structures for programming language processing. In AAAI, 2016.



Task-Specific Prior

Relation Classification

Input:

A trillion gallons of **water** have been poured into an empty region of outer **space**

Target: Entity_Destination



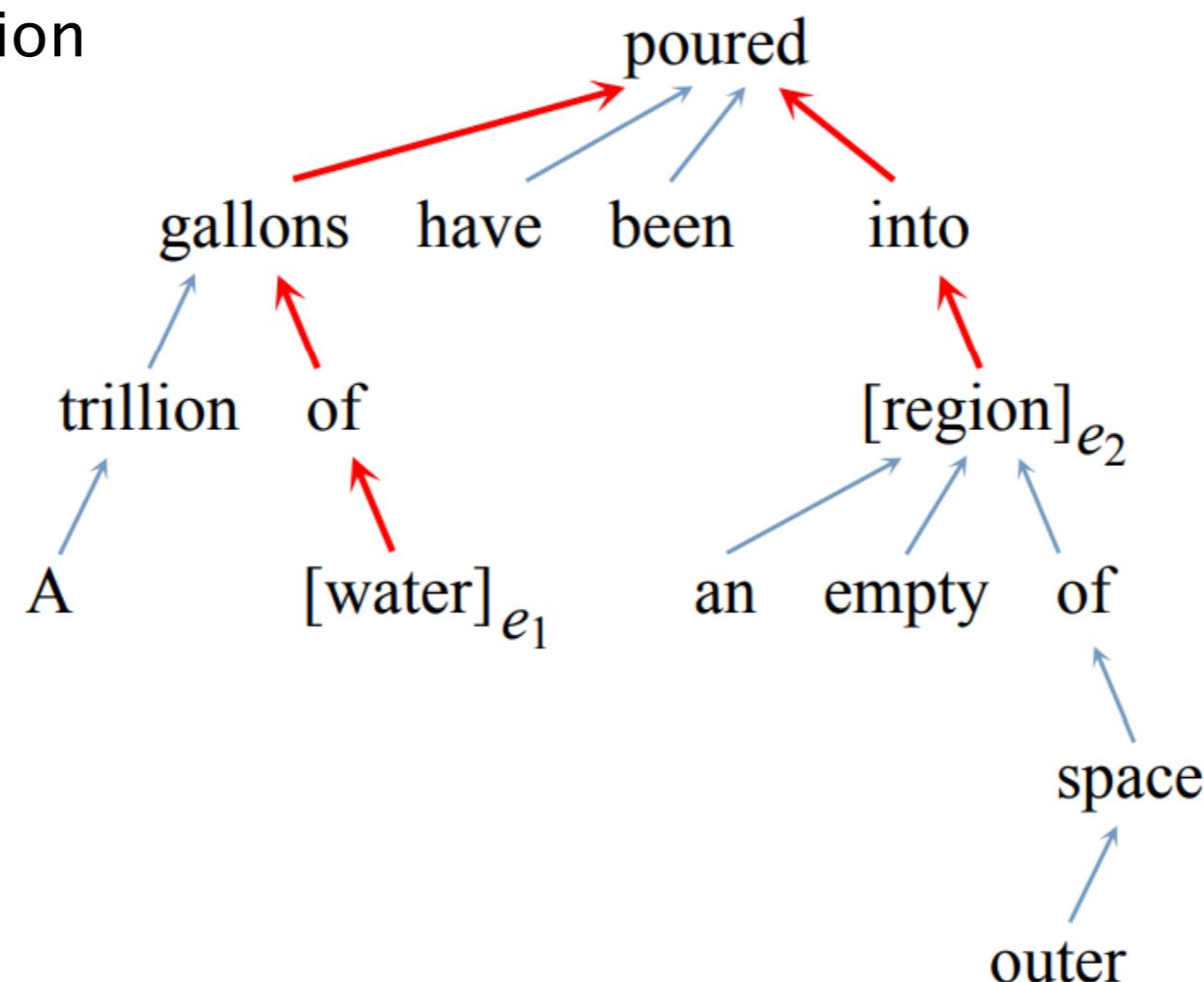
Task-Specific Prior

Relation Classification

Input:

A trillion gallons of **water** have been poured into an empty region of outer **space**

Target: Entity_Destination



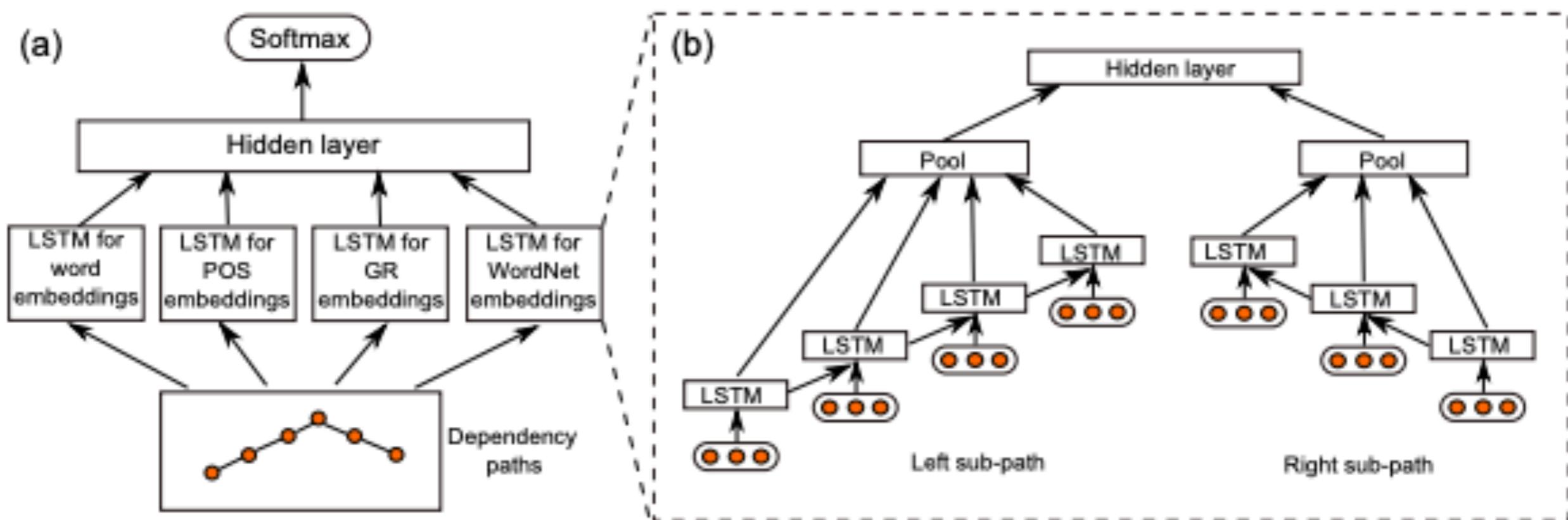
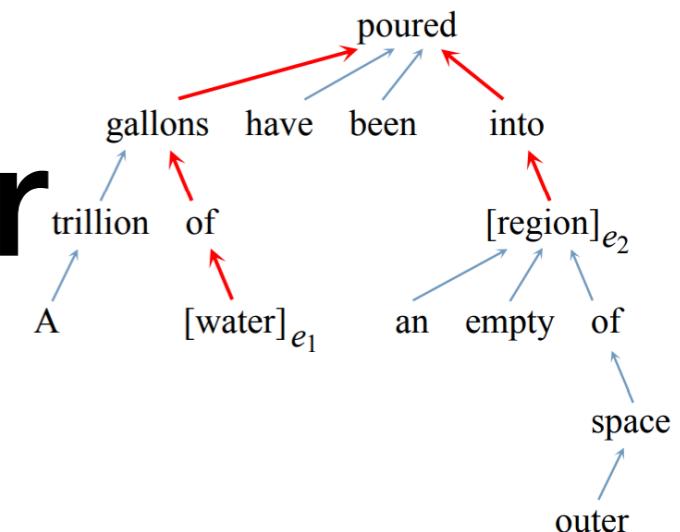
Task-Specific Prior

Relation Classification

Input:

A trillion gallons of **water** have been poured into an empty region of outer **space**

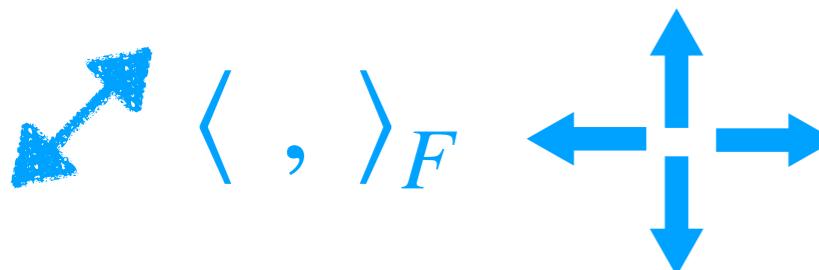
Target: Entity_Destination



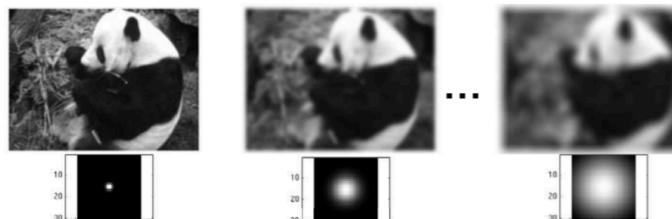


Mindmap

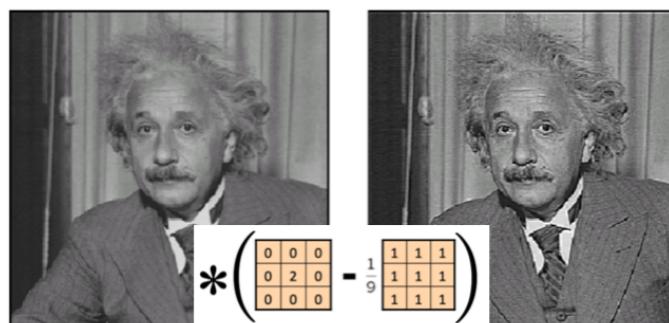
Convolution in signal processing



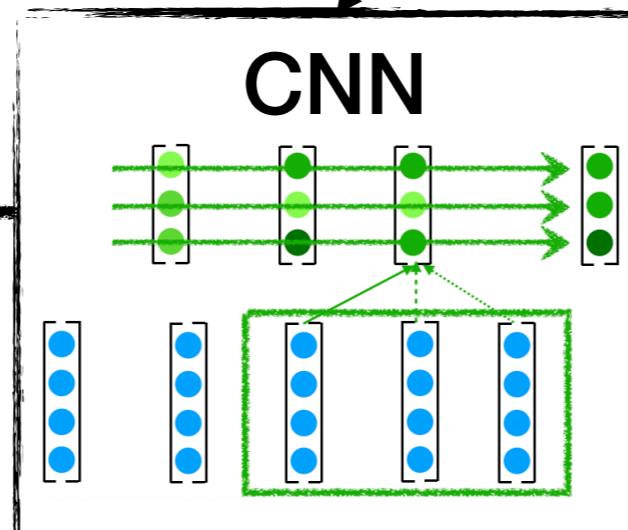
Low-pass filter



High-pass filter

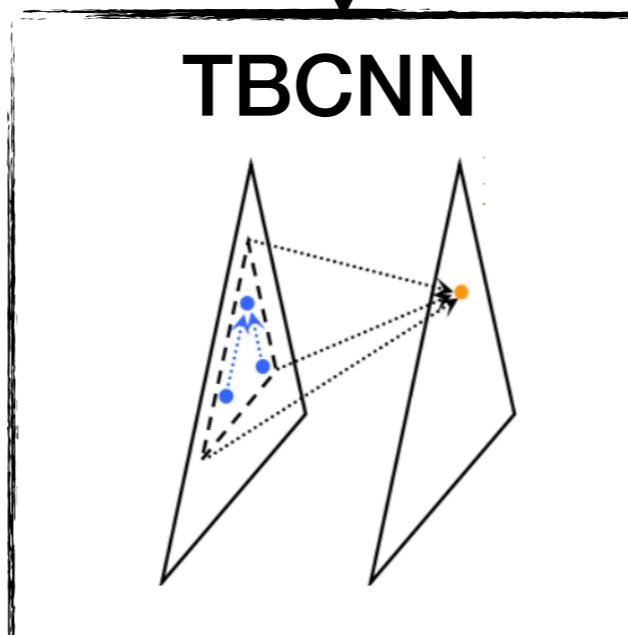
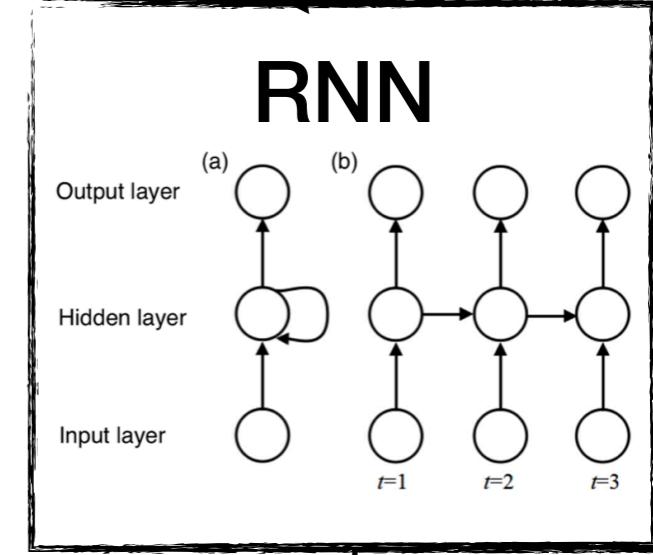


Generic DNN

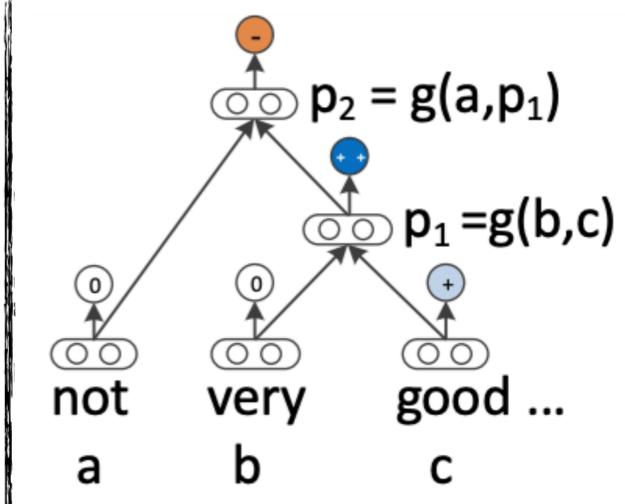


The dog the stick the fire burned beat bit the cat.

RNN



TreeRNN



**Task-specific
architectures**

Suggested Reading

- A textbook (or course) on Digital Signal Processing
- Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural Computation*, 9(8), pp.1735-1780.
- Pascanu et al., 2013. On the difficulty of training recurrent neural networks. In *ICML*.



More References

- Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.
- Yoon Kim, Convolutional Neural Networks for Sentence Classification. In *EMNLP* 2014.
- Kalchbrenner et al, A Convolutional Neural Network for Modelling Sentences. In *ACL* 2014.
- Hu et al., Convolutional neural network architectures for matching natural language sentences. In *NIPS* 2014.
- Sutskever, I., Vinyals, O. and Le, Q.V. Sequence to sequence learning with neural networks. In *NIPS* 2014.
- Socher, R., Pennington, J., Huang, E.H., Ng, A.Y. and Manning, C.D., July. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP* 2011.
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A. and Potts, C., October. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.
- Zhu, X., Sobhani, P. and Guo, H, June. Long short-term memory over recursive structures. In *ICML*, 2015.
- Le, P. and Zuidema, W., 2015. Compositional distributional semantics with long short term memory. arXiv preprint arXiv:1503.02510.
- Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, Zhi Jin. Discriminative neural sentence modeling by tree-based convolution. In *EMNLP*, 2015.
- Lili Mou, Ge Li, Lu Zhang, Tao Wang, Zhi Jin. Convolutional neural networks over tree structures for programming language processing. In *AAAI*, 2016.
- Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, Zhi Jin. Natural language inference by tree-based convolution and heuristic matching. In *ACL*, 2016.
- Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, 2015.

Thank you!

Q&A



UNIVERSITY OF
ALBERTA