ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Αλληλεπίδραση Ανθρώπου–Υπολογιστή

Β8. Επεξεργασία φυσικής γλώσσας με συνελικτικά νευρωνικά δίκτυα (CNNs)

(2023-24)

Ίων Ανδρουτσόπουλος

http://www.aueb.gr/users/ion/

Contents

- Text processing with CNNs.

Convolutions on text

Let's **pretend** that we know what the **dimensions** of the word **embeddings represent**, and that the dimensions are **binary**.

Words	Embeddings			
	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

2

Filter for "I like", "we admire"…			
1	0	0	0
0	**1**	0	0

Filter for "very much", "so much"…			
0	0	**1**	0
0	0	0	**1**

Convolutions on text

Let's **pretend** that we know what the **dimensions** of the word **embeddings represent**, and that the dimensions are **binary**.

	Embeddings			
Words	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

0

Filter for "I like", "we admire"…			
1	0	0	0
0	**1**	0	0

Filter for "very much", "so much"…			
0	0	**1**	0
0	0	0	**1**

Convolutions on text

Words	Embeddings			
	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

2
0
0
0
0
0

Filter for “I like”, “we admire”…			
1	0	0	0
0	**1**	0	0

Filter for “very much”, “so much”…			
0	0	**1**	0
0	0	0	**1**

Convolutions on text

Words	Embeddings			
	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

2 0
0 0
0 0
0 0
0
0

Filter for “I like”, “we admire”…			
1	0	0	0
0	**1**	0	0

Filter for “very much”, “so much”…			
0	0	**1**	0
0	0	0	**1**

Convolutions on text

Words	Embeddings			
	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

2 0
0 0
0 0
0 0
0 2
0

Filter for “I like”, “we admire”…			
1	0	0	0
0	**1**	0	0

Filter for “very much”, “so much”…			
0	0	**1**	0
0	0	0	**1**

Convolutions on text

Words	Embeddings			
	Subject	Positive	Stress	Quantity
I	**1**	0	0	0
like	0	**1**	0	0
this	0	0	0	0
movie	0	0	0	0
very	0	0	**1**	0
much	0	0	0	**1**
!	0	0	**1**	0

2	0
0	0
0	0
0	0
0	2
0	0

Best scores of the two filters: to what extent they **match anywhere** in the sentence.

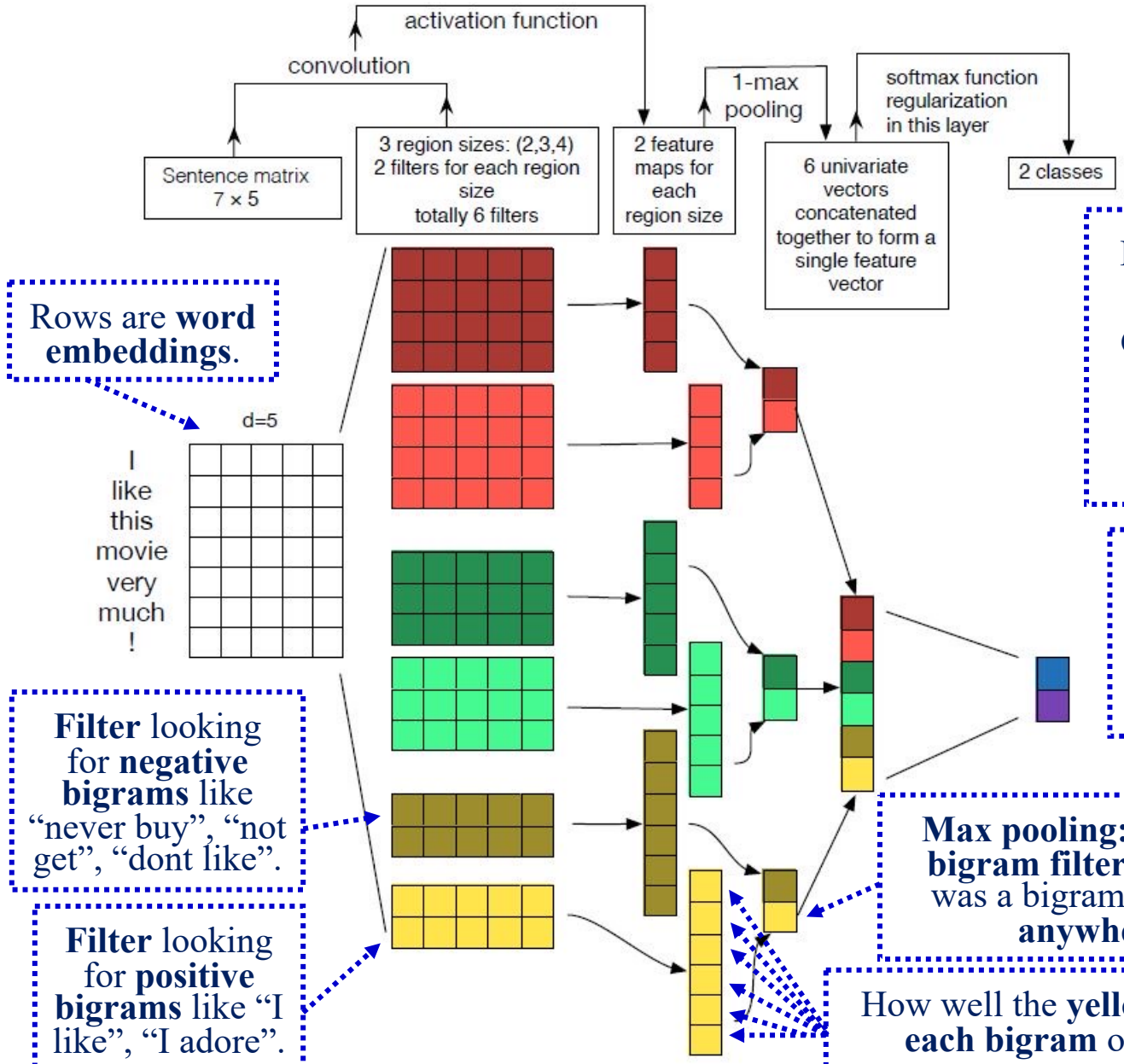
global max pooling

2 2

Filter for “I like”, “we admire”…			
1	0	0	0
0	**1**	0	0

Filter for “very much”, “so much”…			
0	0	**1**	0
0	0	0	**1**

Convolutional Neural Networks



Rows are **word embeddings**.

From "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification", Zhang et al., 2015. http://arxiv.org/abs/1510.03820

The **numbers in each filter** are learned by **backpropagation**. The **embeddings** can also be learned during backpropagation.

Filter looking for **negative bigrams** like "never buy", "not get", "dont like".

Max pooling: Max score of the **yellow bigram filter**. Best evidence that there was a bigram like "I like" or "I adore" **anywhere** in the sentence.

Filter looking for **positive bigrams** like "I like", "I adore".

How well the **yellow bigram filter** matched **each bigram** of the sentence (**scores**).

Convolutions on text – closer to reality

Words	Embeddings			
	d_1	d_2	d_3	d_4
I	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
like	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
this	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
movie	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$
very	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$
much	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$
!	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$

$\mathrm{ReLU}(wx + b)$

$x^T = \langle x_{1,1}, x_{1,2}, x_{1,3}, \ldots, x_{2,3}, x_{2,4} \rangle$

A bigram filter			
$w_{1,1}$	$w_{1,2}$	$w_{1,3}$	$w_{1,4}$
$w_{2,1}$	$w_{2,2}$	$w_{2,3}$	$w_{2,4}$

$w = \langle w_{1,1}, w_{1,2}, w_{1,3}, \ldots, w_{2,3}, w_{2,4} \rangle$

b

Convolutions on text – closer to reality

Words	Embeddings			
	d_1	d_2	d_3	d_4
I	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
like	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
this	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
movie	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$
very	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$
much	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$
!	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$

$\text{ReLU}(wx + b)$

$x^T = \langle x_{2,1}, x_{2,2}, x_{2,3}, \dots, x_{3,3}, x_{3,4} \rangle$

A bigram filter			
$w_{1,1}$	$w_{1,2}$	$w_{1,3}$	$w_{1,4}$
$w_{2,1}$	$w_{2,2}$	$w_{2,3}$	$w_{2,4}$

$w = \langle w_{1,1}, w_{1,2}, w_{1,3}, \dots, w_{2,3}, w_{2,4} \rangle$

b

Now applying **three** bigram filters

Words	Embeddings			
	d_1	d_2	d_3	d_4
I	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
like	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
this	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
movie	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$
very	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$
much	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$
!	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$

$$h_2 = \mathrm{ReLU}(Wx + b) \in \mathbb{R}^{3\times 1}$$

$$x^T = \langle x_{2,1}, x_{2,2}, \ldots, x_{3,3}, x_{3,4} \rangle \in \mathbb{R}^{1\times 8}$$

$$W = \begin{bmatrix} w_{1,1,1} & w_{1,1,2} & w_{1,1,3} & \cdots & w_{1,2,3} & w_{1,2,4} \\ w_{2,1,1} & w_{2,1,2} & w_{2,1,3} & \cdots & w_{2,2,3} & w_{2,2,4} \\ w_{3,1,1} & w_{3,1,2} & w_{3,1,3} & \cdots & w_{3,2,3} & w_{3,2,4} \end{bmatrix} \in \mathbb{R}^{3\times 8} \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \in \mathbb{R}^{3\times 1}$$

Applying 3 bigram filters

Words	Embeddings			
	d_1	d_2	d_3	d_4
I	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
like	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
this	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
movie	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$
very	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$
much	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$
!	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$

$$h_2 = \langle h_{2,1}, h_{2,2}, h_{2,3} \rangle^T \in \mathbb{R}^{3\times 1}$$

$$W = \begin{bmatrix} w_{1,1,1} & w_{1,1,2} & w_{1,1,3} & \cdots & w_{1,2,3} & w_{1,2,4} \\ w_{2,1,1} & w_{2,1,2} & w_{2,1,3} & \cdots & w_{2,2,3} & w_{2,2,4} \\ w_{3,1,1} & w_{3,1,2} & w_{3,1,3} & \cdots & w_{3,2,3} & w_{3,2,4} \end{bmatrix} \in \mathbb{R}^{3\times 8} \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \in \mathbb{R}^{3\times 1}$$

Applying 3 bigram filters

$$h^{max} = \langle \max(h_{*,1}), \max(h_{*,2}), \max(h_{*,3}) \rangle^T$$

global max pooling

Feature vector sent to a classifier, regressor, etc.

Words	Embeddings			
	d_1	d_2	d_3	d_4
I	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$
like	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$
this	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$
movie	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$
very	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$
much	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$
!	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$

$$h_1 = \langle h_{1,1}, h_{1,2}, h_{1,3} \rangle^T$$
$$h_2 = \langle h_{2,1}, h_{2,2}, h_{2,3} \rangle^T$$
$$h_3 = \langle h_{3,1}, h_{3,2}, h_{3,3} \rangle^T$$
$$h_4 = \langle h_{4,1}, h_{4,2}, h_{4,3} \rangle^T$$
$$\dots$$
$$h_7 = \langle h_{7,1}, h_{7,2}, h_{7,3} \rangle^T$$

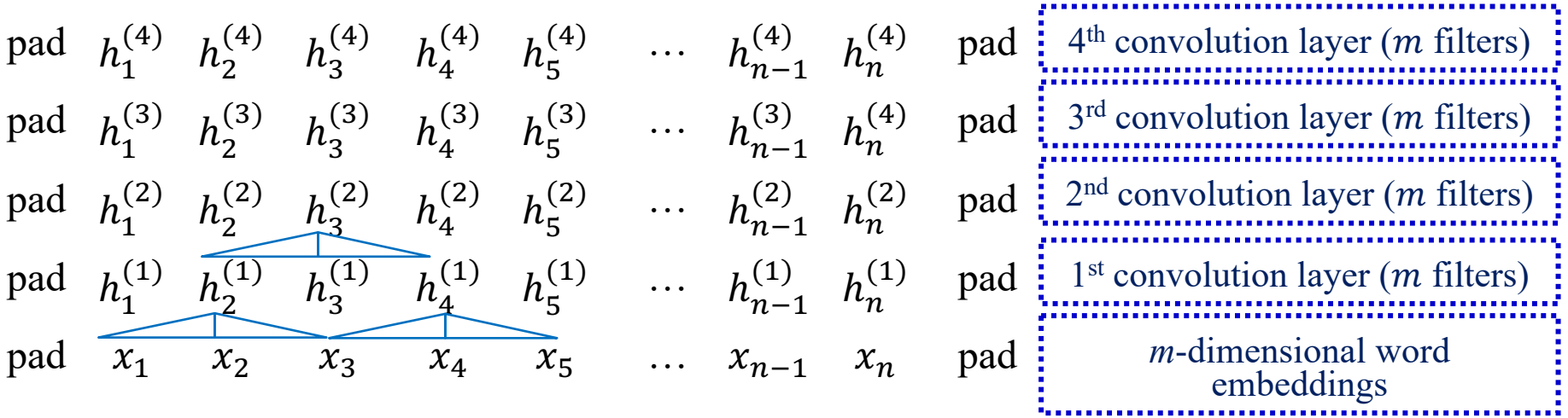
$$W = \begin{bmatrix} w_{1,1,1} & w_{1,1,2} & w_{1,1,3} & \dots & w_{1,2,3} & w_{1,2,4} \\ w_{2,1,1} & w_{2,1,2} & w_{2,1,3} & \dots & w_{2,2,3} & w_{2,2,4} \\ w_{3,1,1} & w_{3,1,2} & w_{3,1,3} & \dots & w_{3,2,3} & w_{3,2,4} \end{bmatrix} \in \mathbb{R}^{3\times 8} \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \in \mathbb{R}^{3\times 1}$$

Stacked CNNs for classification/regression

$$h^{max} = \left\langle \max\left(h_{*,1}^{(4)}\right), \max\left(h_{*,2}^{(4)}\right), \ldots, \max\left(h_{*,m}^{(4)}\right) \right\rangle^T \in \mathbb{R}^{1\times m}$$

global max pooling

Feature vector sent to a document classifier or regressor (e.g., MLP).

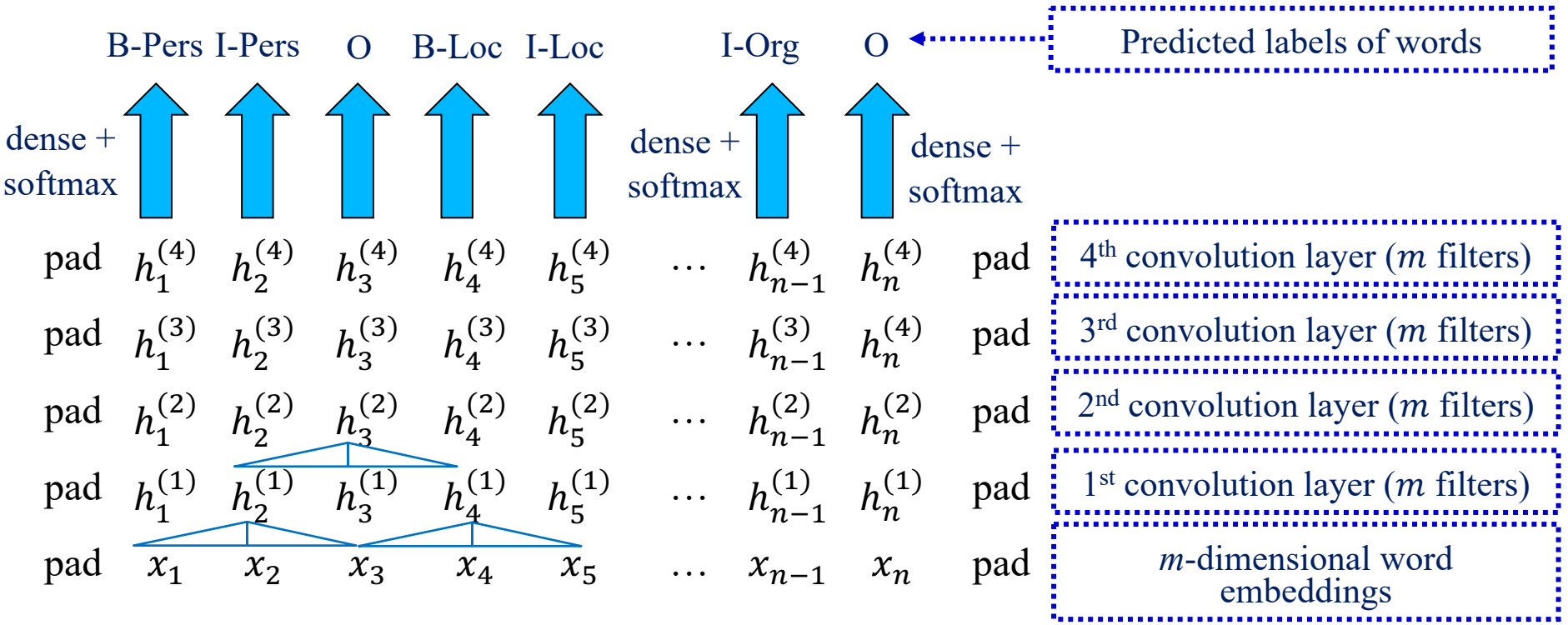


$$h_i^{(1)} = \text{ReLU}\left(W^{(1)}[x_{i-1}; x_i; x_{i+1}] + b^{(1)}\right) + x_i \in \mathbb{R}^{m\times 1}$$

$$h_i^{(j)} = \text{ReLU}\left(W^{(j)}\left[h_{i-1}^{(j-1)}; h_i^{(j-1)}; h_{i+1}^{(j-1)}\right] + b^{(j)}\right) + h_i^{(j-1)} \in \mathbb{R}^{m\times 1}$$

Residual (shortcut) connection, needed when stacking many CNNs (or RNNs).

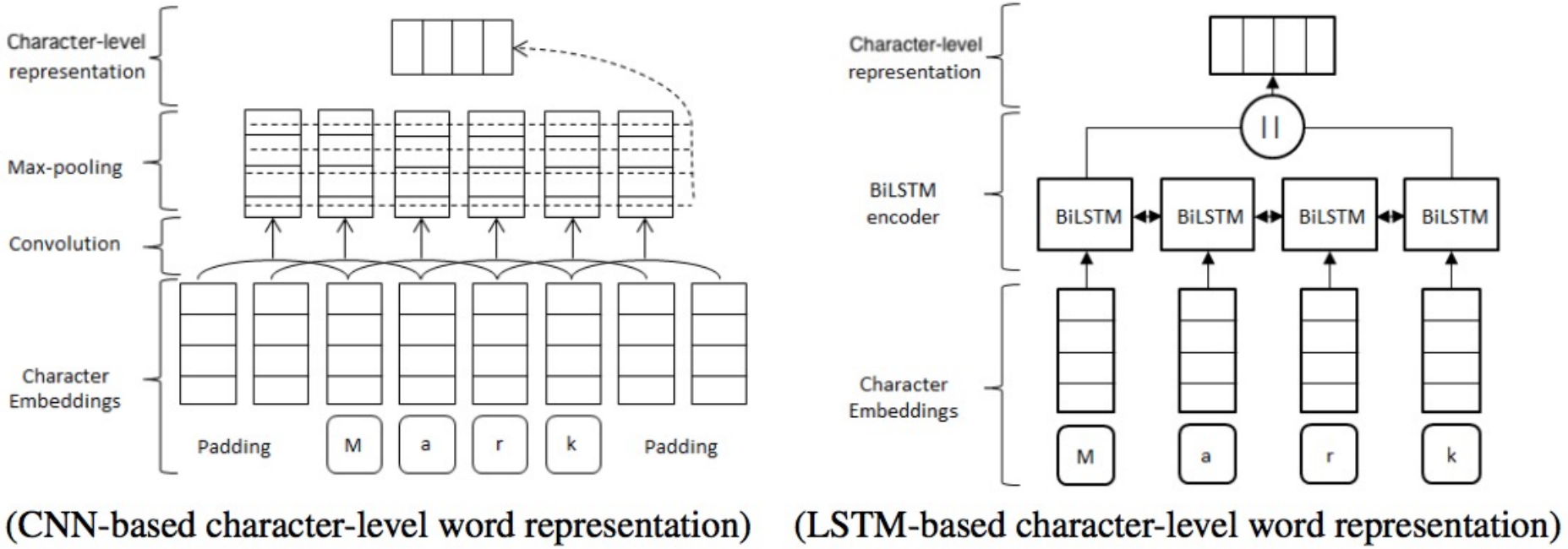
Stacked CNNs for token classification



$$h_i^{(1)} = \mathrm{ReLU}\left(W^{(1)}[x_{i-1}; x_i; x_{i+1}] + b^{(1)}\right) + x_i \in \mathbb{R}^{m\times 1}$$

$$h_i^{(j)} = \mathrm{ReLU}\left(W^{(j)}\left[h_{i-1}^{(j-1)}; h_i^{(j-1)}; h_{i+1}^{(j-1)}\right] + b^{(j)}\right) + h_i^{(j-1)} \in \mathbb{R}^{m\times 1}$$

CNNs/RNNs that produce word embeddings from character embeddings

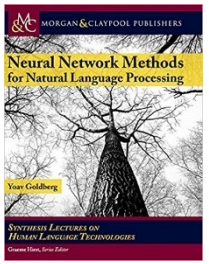


(CNN-based character-level word representation) (LSTM-based character-level word representation)

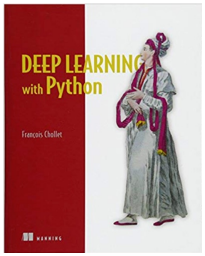
Figure 2: Character-level word representations. This figure is also adapted from Reimers and Gurevych (2017a).

Z. Zhai, D.Q. Nguyen and K. Verspoor, "Comparing CNN and LSTM Character-Level Embeddings in BiLSTM-CRF Models for Chemical and Disease Named Entity Eecognition". 9th Int. Workshop on Health Text Mining and Information Analysis, Brussels, Belgium, 2018. http://aclweb.org/anthology/W18-5605

Recommended reading



- Y. Goldberg, *Neural Network Models for Natural Language Processing*, Morgan & Claypool Publishers, 2017.
  - Mostly Chapter 13.
- Jurafsky and Martin's, *Speech and Language Processing* is being revised (3rd edition) to include DL methods.
  - http://web.stanford.edu/~jurafsky/slp3/ (free draft)
- F. Chollet, *Deep Learning in Python*, 1st edition, Manning Publications, 2017.
  - 1st edition freely available (and sufficient for this course): https://www.manning.com/books/deep-learning-with-python
  - See Chapter 6 for CNNs in Computer Vision.
  - 2nd edition (2022) now available, requires payment. Highly recommended.



Βιβλιογραφία – συνέχεια

- Αν έχετε από το μάθημα της ΤΝ το βιβλίο των Russel & Norvig «Τεχνητή Νοημοσύνη – Μια σύγχρονη προσέγγιση», 4η έκδοση, Κλειδάριθμος, 2021, μπορείτε να συμβουλευτείτε το κεφάλαιο 21.
  - Κυρίως την ενότητα 21.3.