CS 5740: Natural Language Processing

CNNs for Text Processing

Instructor: Yoav Artzi

Slides adapted from Yoav Goldberg's book

Overview

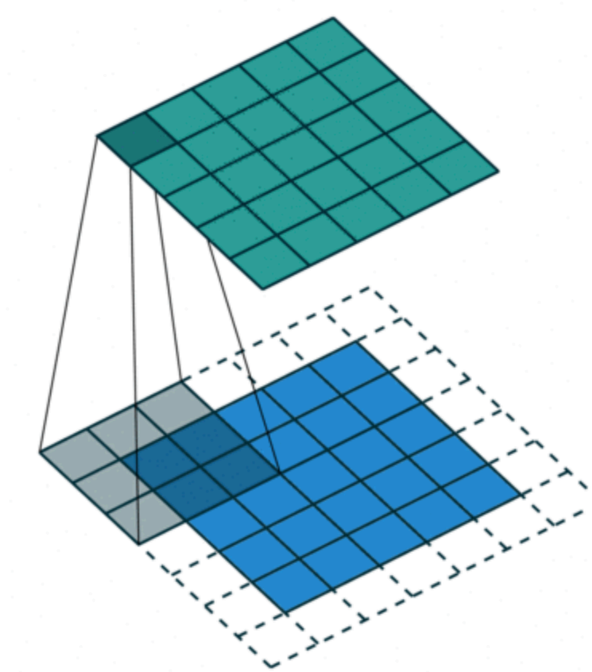
- Convolutional Neural Networks (CNNs) in a nutshell
- Convolution and pooling over text
- Hierarchical convolution

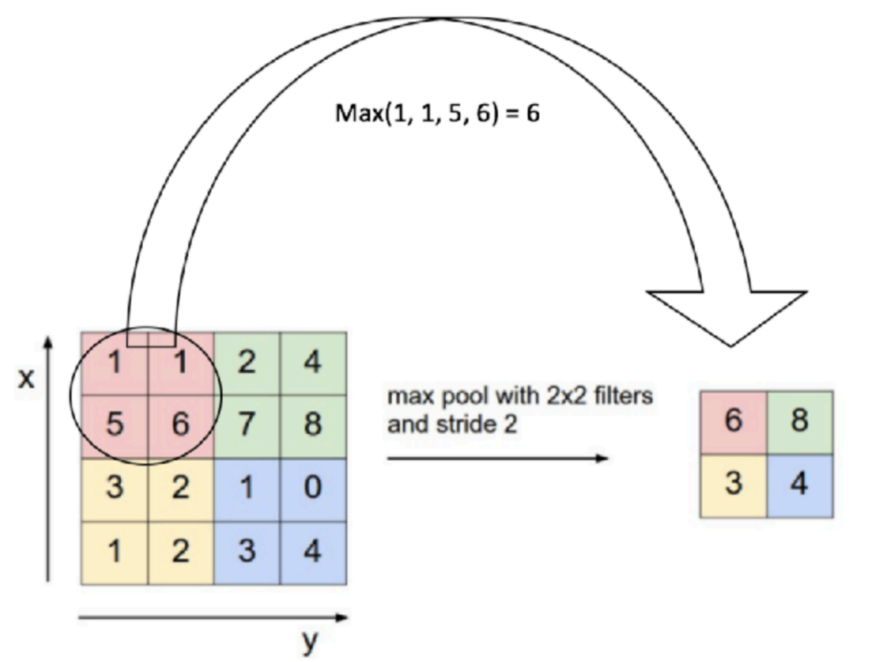
CNNs in a Nutshell

- Computer vision neural network architecture
- Method to process an input of <u>different sizes</u> with <u>few parameters</u>
- Basically: scan the input piece-by-piece with a parameterized or non-parameterized operation

CNNs in a Nutshell

- Two main types of operations:
  - Convolution (parameterized)
  - Pooling (not parameterized)





Convolution Over Text

ϕ – embedding function

$\bar{x}$ – sentence

$\mathbf{u}$ – filter, a weight vector

$\bar{x} = \langle x_1, \ldots, x_n \rangle$

$\mathbf{x}_i = \phi(x_i)$

$p_i = g([\mathbf{x}_i; \ldots; \mathbf{x}_{i+k-1}] \cdot \mathbf{u})$

- Map sequence to (shorter) sequence
- Map (filter) each k-gram to a single number
- Narrow: no padding, so output is $n - k + 1$
- Wide: add $k - 1$ padding on each side so output is $n + k + 1$

Multiple Filters

$$\mathbf{U} = \begin{bmatrix} | & | & & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_l \\ | & | & & | \end{bmatrix} \text{ – matrix of } l \text{ filters, each is a column}$$

ϕ – embedding function

$\bar{x}$ – sentence

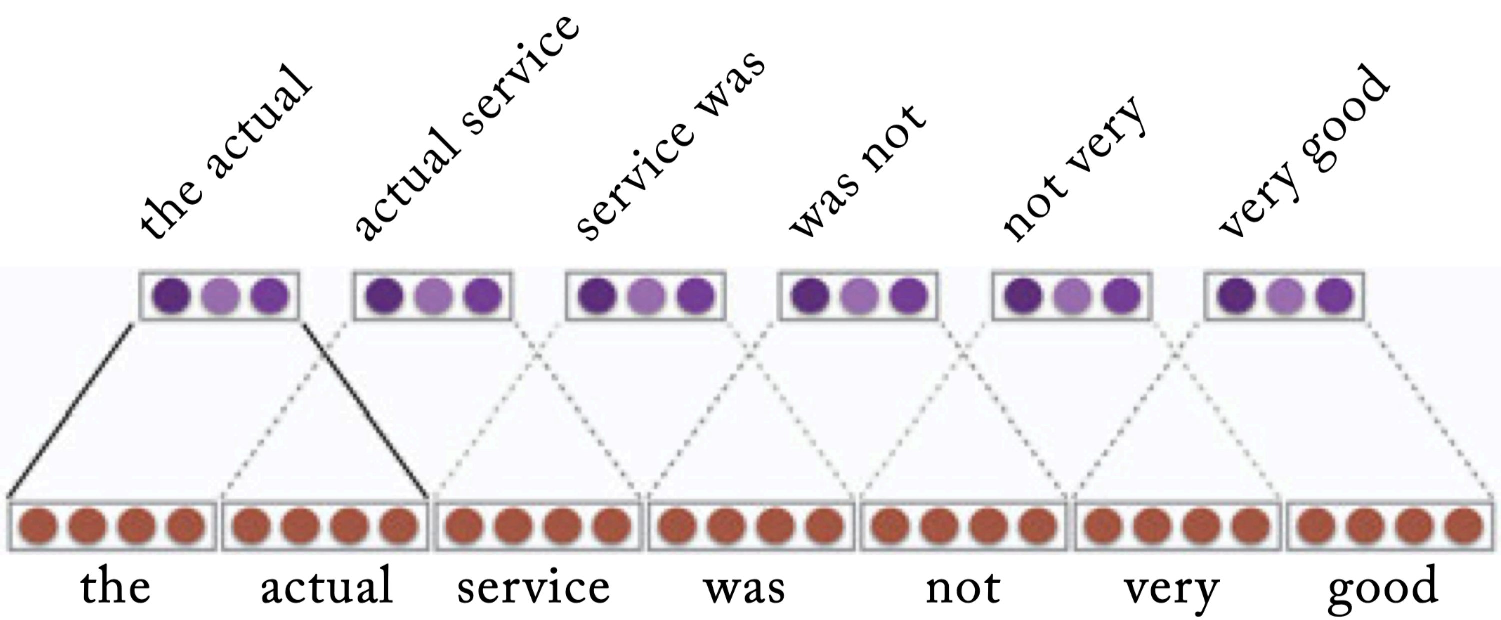
$\bar{x} = \langle x_1, \dots, x_n \rangle$

$\mathbf{x}_i = \phi(x_i)$

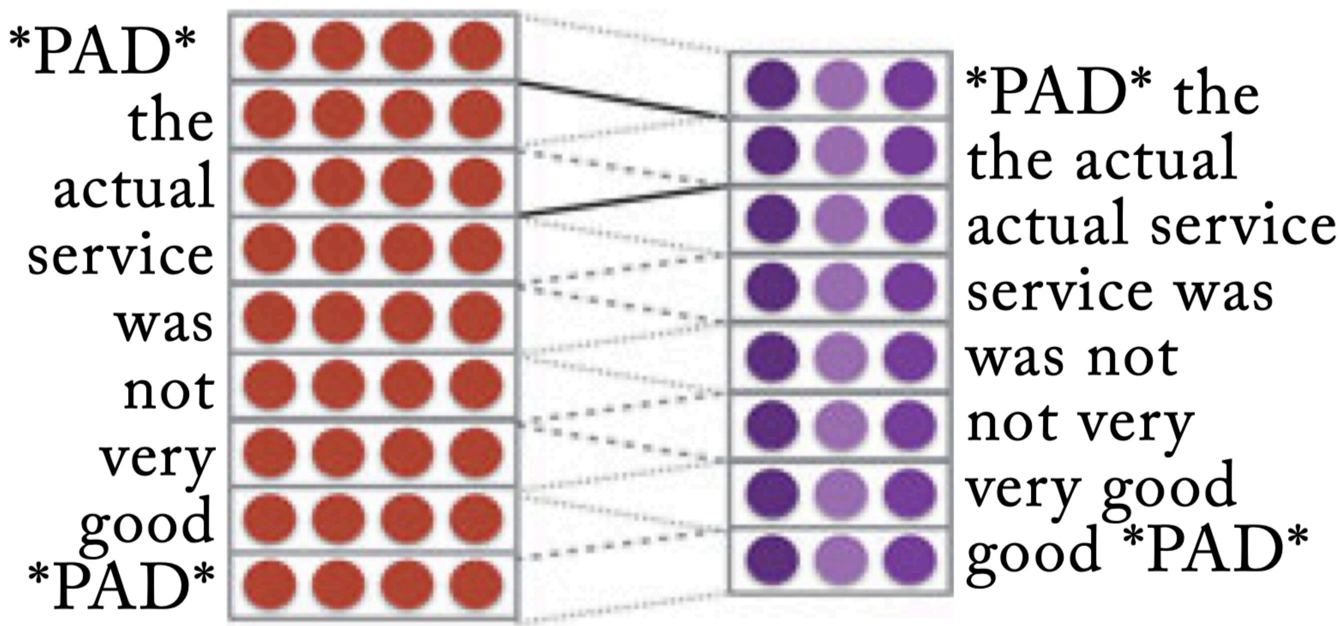
$\mathbf{p}_i = g([\mathbf{x}_i; \dots; \mathbf{x}_{i+k-1}] \cdot \mathbf{U})$

- Usually we use l different filters $\mathbf{u}_1, \dots, \mathbf{u}_l$
- Map each k-gram to a vector $\mathbf{p}_i$ with l values that represent the i-th window (i.e., k-gram)

Narrow Convolution

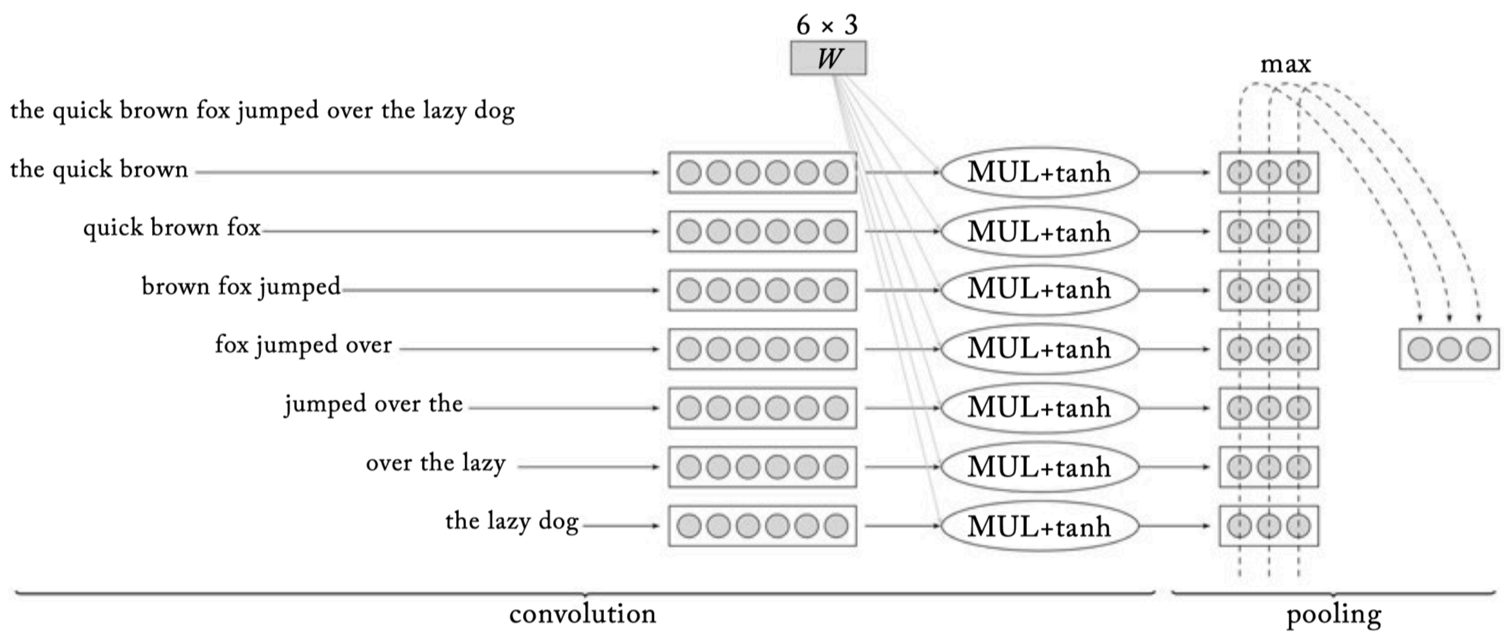


Wide Convolution



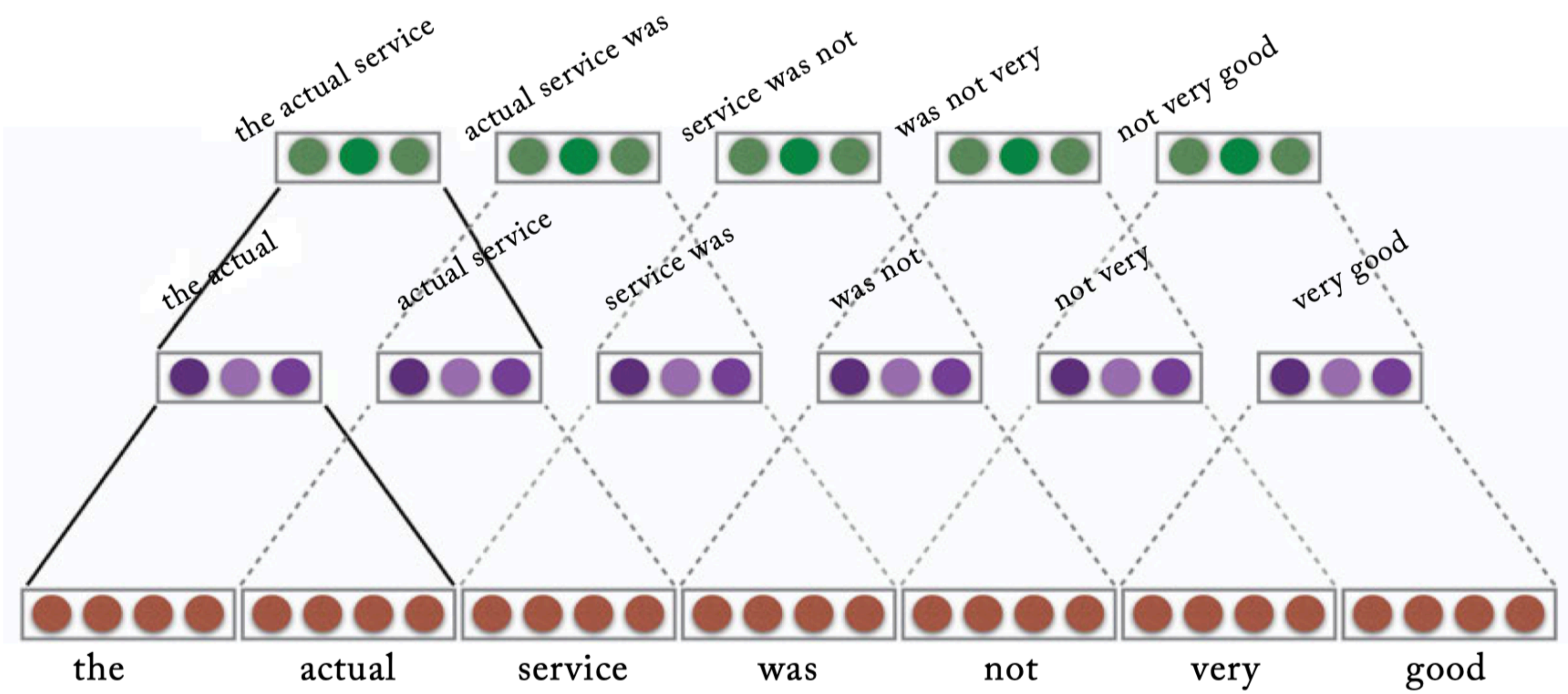
Pooling

- Can pool the values to get a fixed-length output
- Different pooling functions: max, average, k-max



Hierarchical Convolutions

- Stack convolutional layers
- Capture increasingly larger receptive fields (effective windows)



Hierarchical Convolutions

r – number of convolutional layers

CONV_θ^k – convolution with window k and parameters θ

$$m = \begin{cases} n-k+1 \text{ narrow} \\ n+k+1 \text{ wide} \end{cases} \quad \text{– number of convolution output elements}$$

ϕ – embedding function

$\bar{x}$ – sentence

$\bar{x} = \langle x_1, \ldots, x_n \rangle$

$\mathbf{x}_i = \phi(x_i)$

$\mathbf{p}_1^1, \ldots, \mathbf{p}_{m_1}^1 = \text{CONV}_{\mathbf{U}^1,\mathbf{b}^1}^{k_1}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$

$\mathbf{p}_1^2, \ldots, \mathbf{p}_{m_2}^2 = \text{CONV}_{\mathbf{U}^2,\mathbf{b}^2}^{k_2}(\mathbf{p}_1^1, \ldots, \mathbf{p}_{m_1}^1)$

$\ldots$

$\mathbf{p}_1^r, \ldots, \mathbf{p}_{m_2}^r = \text{CONV}_{\mathbf{U}^r,\mathbf{b}^r}^{k_r}(\mathbf{p}_1^{r-1}, \ldots, \mathbf{p}_{m_{r-1}}^{r-1})$

- Stack convolutional layers
- $\mathbf{p}_1^r, \ldots, \mathbf{p}_{m_r}^r$ capture increasingly larger receptive fields (effective windows)

Strides

- So far: convolution is applied to each k-word window → this is called a stride of 1

- Larger strides also possible