

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro
Convolut tion

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Convolutional Networks

<https://eduassistpro.github.io/>

- ▶ A convolutional network is designed to identify indicative local indicators in a large structure, and combine them into a **fixed size** vector representation of the structure. It uses a pooling function, capturing the local aspect of the structure, to be informative.
- ▶ A convolutional network is a type of feedforward network.
- ▶ It has been tremendously successful in image computer vision), where the input is the raw pixels of an image.
- ▶ In NLP, it has been shown to be effective in sentence classification, etc.

Why it has been so effective in image processing

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Image pixels

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Four operations in a convolutional network

<https://eduassistpro.github.io/>

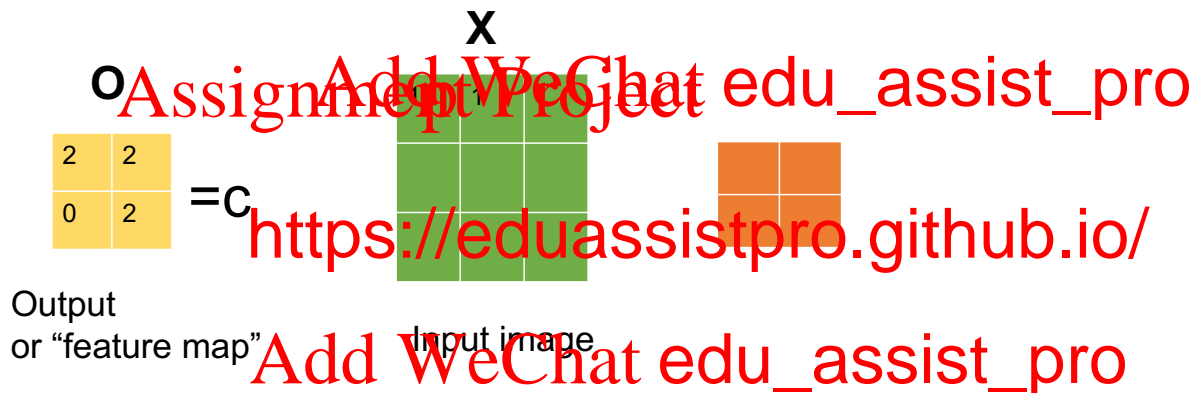
Assignment Project Exam Help

- ▶ Convolution
- ▶ Non-linear a
- ▶ Pooling or sub
- ▶ Classification with a fully-connected lay

Image convolution

<https://eduassistpro.github.io/>

Assignment Project Exam Help



Kernel size = (2,2), strides = 1


Image convolution

<https://eduassistpro.github.io/>

Assignment Project Exam Help

`conv(X, U)`

Add WeChat edu_assist_pro



<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Kernel size = (2,2), strides = 1

Image convolution

<https://eduassistpro.github.io/>

Assignment Project Exam Help

conv(X, U)

1	1x1	1x0
0	1x0	1x1
0	0	

Assignment Project

0	2	2

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Kernel size = (2,2), strides = 1

Image convolution

<https://eduassistpro.github.io/>

conv(X, U)

Assignment Project Exam Help



Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Kernel size = (2,2), strides = 1

Image convolution

<https://eduassistpro.github.io/>

conv(X, U)

Assignment Project Exam Help

1	1	1	0
0	1x1	1x0	2
0	0x0		2

Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Kernel size = (2,2), strides = 1

Forward computation

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Exam Help
Add WeChat edu_assist_pro

$$o_{11} = x_{11}u_{11} + x_{12}u_{12} + x_{13}u_{13}$$

$$o_1 = x_{11}u_{11} + x_{12}u_{12} + x_{13}u_{13}$$

$$o_2 = x_{21}u_{11} + x_{22}u_{12} + x_{23}u_{13}$$

$$o_{22} = x_{22}u_{11} + x_{23}u_{12} + x_{24}u_{13}$$

Add WeChat edu_assist_pro

ReLU

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- ▶ Nonlinear transformation with ReLU

[Add WeChat edu_assist_pro](https://eduassistpro.github.io/)
(0, $input$)

- ▶ As we know, ReLU does not change the dimension of the feature

- ▶ ReLU replaces all negative pixel values in t with 0

Image ReLU

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

ReLU activation and Max pooling

<https://eduassistpro.github.io/>

- ▶ ReLU activation is a component-wise function and does not change the dimension of the input

[Add WeChat edu_assist_pro](https://eduassistpro.github.io/)

<https://eduassistpro.github.io/>

- ▶ Max pooling does change the dimension
specify the pool size and strides

[Add WeChat edu_assist_pro](https://eduassistpro.github.io/)

$$[2] = \text{Max} \left(\begin{bmatrix} 2 & 2 \\ 0 & 2 \end{bmatrix} \right)$$

pool size = (2, 2), strides = 2

Training a CNN

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- ▶ Loss functions: Cross entropy loss, square
- ▶ What are the parameters of a CNN?
 - ▶ The filter and sizes
- ▶ Computing the gradient for the convolutional layer from a feedforward neural network...

Computing the gradient on U

$$\begin{aligned}
 \frac{\partial E}{\partial u_{11}} &= \frac{\partial o_{11}}{\partial u_{11}} x_{11} + \frac{\partial o_{12}}{\partial u_{11}} x_{12} + \frac{\partial o_{21}}{\partial u_{11}} x_{21} + \frac{\partial o_{22}}{\partial u_{11}} x_{22} \\
 &= \frac{\partial E}{\partial o_{11}} \frac{\partial o_{11}}{\partial u_{11}} + \frac{\partial E}{\partial o_{12}} \frac{\partial o_{12}}{\partial u_{11}} + \frac{\partial E}{\partial o_{21}} \frac{\partial o_{21}}{\partial u_{11}} + \frac{\partial E}{\partial o_{22}} \frac{\partial o_{22}}{\partial u_{11}} \\
 &= \frac{\partial E}{\partial o_{11}} x_{11} + \frac{\partial E}{\partial o_{12}} x_{12} + \frac{\partial E}{\partial o_{21}} x_{21} + \frac{\partial E}{\partial o_{22}} x_{22}
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial E}{\partial u_{12}} &= \frac{\partial o_{11}}{\partial u_{12}} x_{11} + \frac{\partial o_{12}}{\partial u_{12}} x_{12} + \frac{\partial o_{21}}{\partial u_{12}} x_{21} + \frac{\partial o_{22}}{\partial u_{12}} x_{22} \\
 &= \frac{\partial E}{\partial o_{11}} \frac{\partial o_{11}}{\partial u_{12}} + \frac{\partial E}{\partial o_{12}} \frac{\partial o_{12}}{\partial u_{12}} + \frac{\partial E}{\partial o_{21}} \frac{\partial o_{21}}{\partial u_{12}} + \frac{\partial E}{\partial o_{22}} \frac{\partial o_{22}}{\partial u_{12}} \\
 &= \frac{\partial E}{\partial o_{11}} x_{11} + \frac{\partial E}{\partial o_{12}} x_{12} + \frac{\partial E}{\partial o_{21}} x_{21} + \frac{\partial E}{\partial o_{22}} x_{22}
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial E}{\partial u_{21}} &= \frac{\partial o_{11}}{\partial u_{21}} x_{11} + \frac{\partial o_{12}}{\partial u_{21}} x_{12} + \frac{\partial o_{21}}{\partial u_{21}} x_{21} + \frac{\partial o_{22}}{\partial u_{21}} x_{22} \\
 &= \frac{\partial E}{\partial o_{11}} \frac{\partial o_{11}}{\partial u_{21}} + \frac{\partial E}{\partial o_{12}} \frac{\partial o_{12}}{\partial u_{21}} + \frac{\partial E}{\partial o_{21}} \frac{\partial o_{21}}{\partial u_{21}} + \frac{\partial E}{\partial o_{22}} \frac{\partial o_{22}}{\partial u_{21}} \\
 &= \frac{\partial E}{\partial o_{11}} x_{11} + \frac{\partial E}{\partial o_{12}} x_{12} + \frac{\partial E}{\partial o_{21}} x_{21} + \frac{\partial E}{\partial o_{22}} x_{22}
 \end{aligned}$$

$$\begin{aligned}
 \frac{\partial E}{\partial u_{22}} &= \frac{\partial o_{11}}{\partial u_{22}} x_{11} + \frac{\partial o_{12}}{\partial u_{22}} x_{12} + \frac{\partial o_{21}}{\partial u_{22}} x_{21} + \frac{\partial o_{22}}{\partial u_{22}} x_{22} \\
 &= \frac{\partial E}{\partial o_{11}} \frac{\partial o_{11}}{\partial u_{22}} + \frac{\partial E}{\partial o_{12}} \frac{\partial o_{12}}{\partial u_{22}} + \frac{\partial E}{\partial o_{21}} \frac{\partial o_{21}}{\partial u_{22}} + \frac{\partial E}{\partial o_{22}} \frac{\partial o_{22}}{\partial u_{22}} \\
 &= \frac{\partial E}{\partial o_{11}} x_{11} + \frac{\partial E}{\partial o_{12}} x_{12} + \frac{\partial E}{\partial o_{21}} x_{21} + \frac{\partial E}{\partial o_{22}} x_{22}
 \end{aligned}$$

Summing up errors from all outputs that the filter component has contributed to.

Reverse Convolution

<https://eduassistpro.github.io/>

Assignment Project Exam Help

The computation of the gradient on the filter can be viewed as a reverse convolution:

$$\begin{bmatrix} \frac{\partial E}{\partial u_{11}} & \frac{\partial E}{\partial u_{12}} & \frac{\partial E}{\partial u_{21}} & \frac{\partial E}{\partial u_{22}} \\ \frac{\partial E}{\partial u_{11}} & \frac{\partial E}{\partial u_{12}} & \frac{\partial E}{\partial u_{21}} & \frac{\partial E}{\partial u_{22}} \end{bmatrix} \times \begin{bmatrix} x_{11} & x_{12} & x_{21} & x_{22} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial o_{11}} & \frac{\partial E}{\partial o_{12}} & \frac{\partial E}{\partial o_{21}} & \frac{\partial E}{\partial o_{22}} \end{bmatrix}$$

Computing the gradient on **X** (if this is not the input layer)

<https://eduassistpro.github.io/>
 Assignment Project Exam Help
 Add WeChat edu_assist_pro

$$\begin{aligned}\frac{\partial E}{\partial x_{11}} &= \frac{\partial E}{\partial o_{11}} u_{11} + \frac{\partial E}{\partial o_{12}} 0 + \frac{\partial E}{\partial o_{21}} 0 + \frac{\partial E}{\partial o_{22}} 0 \\ \frac{\partial E}{\partial x_{12}} &= \frac{\partial E}{\partial o_{11}} u_{12} + \frac{\partial E}{\partial o_{12}} u_{11} + \frac{\partial E}{\partial o_{21}} 0 + \frac{\partial E}{\partial o_{22}} 0 \\ \frac{\partial E}{\partial x_{13}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} u_{12} + \frac{\partial E}{\partial o_{21}} 0 + \frac{\partial E}{\partial o_{22}} 0 \\ \frac{\partial E}{\partial x_2} &= \frac{\partial E}{\partial o_{11}} u_{22} + \frac{\partial E}{\partial o_{12}} u_{21} + \frac{\partial E}{\partial o_{21}} u_{12} + \frac{\partial E}{\partial o_{22}} u_{11} \\ \frac{\partial E}{\partial x_{22}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} u_{22} + \frac{\partial E}{\partial o_{21}} u_{21} + \frac{\partial E}{\partial o_{22}} u_{11} \\ \frac{\partial E}{\partial x_{23}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} u_{22} + \frac{\partial E}{\partial o_{21}} u_{21} + \frac{\partial E}{\partial o_{22}} u_{11} \\ \frac{\partial E}{\partial x_{31}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} 0 + \frac{\partial E}{\partial o_{21}} u_{21} + \frac{\partial E}{\partial o_{22}} 0 \\ \frac{\partial E}{\partial x_{32}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} 0 + \frac{\partial E}{\partial o_{21}} u_{22} + \frac{\partial E}{\partial o_{22}} u_{21} \\ \frac{\partial E}{\partial x_{12}} &= \frac{\partial E}{\partial o_{11}} 0 + \frac{\partial E}{\partial o_{12}} 0 + \frac{\partial E}{\partial o_{21}} 0 + \frac{\partial E}{\partial o_{22}} u_{22}\end{aligned}$$

Full convolution

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

$$\left[\begin{array}{ccc} \frac{\partial E}{\partial x_{11}} & \frac{\partial E}{\partial x_{12}} & \frac{\partial E}{\partial x_{13}} \\ \frac{\partial E}{\partial x_{21}} & \frac{\partial E}{\partial x_{22}} & \frac{\partial E}{\partial x_{23}} \\ \frac{\partial E}{\partial x_{31}} & \frac{\partial E}{\partial x_{32}} & \frac{\partial E}{\partial x_{33}} \end{array}, \begin{array}{cc} u_{11} & u_{12} \\ u_{21} & u_{22} \end{array} \right]$$

<https://eduassistpro.github.io/>
Add WeChat edu_assist_pro

Gradient on \mathbf{X} if it is not the inputs

<https://eduassistpro.github.io/AssignmentProjectExamHelp>
Add WeChat edu_assist_pro

u_{22}	u_{21}		22	21		22	u_{21}
u_{12}	$\delta o_{11} u_{11}$	δo_{12}	$\delta o_{11} u_{12}$	$\delta o_{12} u_{11}$	δo_{11}	$\delta o_{12} u_{12}$	u_{11}
	δo_{21}	δo_{22}	δo_{21}	δo_{22}			

<https://eduassistpro.github.io/AssignmentProjectExamHelp>
Add WeChat edu_assist_pro

u_{22}	$\delta o_{11} u_{21}$					x_{12}	u_{21}
u_{12}	$\delta o_{21} u_{11}$	δo_{22}	$\delta o_{21} u_{12}$	δo_{22}		x_{22}	u_{11}

	δo_{11}	δo_{12}	δo_{11}	δo_{12}	δo_{11}	δo_{12}
u_{22}	$\delta o_{21} u_{21}$	δo_{22}	$\delta o_{21} u_{22}$	$\delta o_{22} u_{21}$	δo_{21}	$\delta o_{22} u_{22}$
u_{12}	u_{11}		u_{12}	u_{11}		u_{12}

Why convolutional networks for NLP?

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- ▶ Even though bag-of-words models are simple, for some text classification tasks, they don't account for word order where multiple words may be important. <https://eduassistpro.github.io/>^s
- ▶ The analogy with image processing is if words are treated as separate features. (The analogy might be more accurate if words are treated as separate features.) [Add WeChat edu_assist_pro](https://eduassistpro.github.io/)

Alternative text representations

- <https://eduassistpro.github.io/>
- Assignment Project Exam Help
- <https://eduassistpro.github.io/>
- Add WeChat edu_assist_pro
- ▶ When using feed-forward networks for text classification, the input text is represented as a sparse vector that encodes bag-of-words features.
 - ▶ Alternatively, a text can be represented as a sequence of tokens $w_1, w_2, w_3, \dots, w_M$. This is useful for models such as **ConvNets**, which process
 - ▶ Each word token w_m is represented by a one-hot vector \mathbf{e}_{w_m} with dimension V . The complete document is represented by the horizontal concatenation of these one-hot vectors: $\mathbf{W} = [\mathbf{e}_{w_1}, \mathbf{e}_{w_2}, \dots, \mathbf{e}_{w_M}] \in R^{V \times M}$.
 - ▶ To show that this is equivalent to the bag-of-words model, we can recover the word count from the matrix-vector product $\mathbf{W}[1, 1, \dots, 1]^T \in R^V$.

Input to a convolutional network in a text classification task

<https://eduassistpro.github.io/>

Assignment Project Exam Help

When using conv nets for text classification, it is com

“look up” the pretrained word embedding (e.g., the word in the sequence: produced by Word2Vec or GLOVE¹)

<https://eduassistpro.github.io/>

<https://eduassistpro.github.io/>

$$= \Theta \begin{bmatrix} w_1, \end{bmatrix}$$

<https://eduassistpro.github.io/>

where \mathbf{e}_{w_m} is a column vector of zeros, with a 1 at position w_m , K_e is the size of embeddings

¹<https://nlp.stanford.edu/projects/glove>

“Convolve” the input with a set of filters (kernels)

<https://eduassistpro.github.io/>

- ▶ A *filter* is a weight matrix of dimension $\mathbf{C}^{(k)} \in \mathbb{R}^{l_e \times h}$ where $\mathbf{C}^{(k)}$ is the k th filter. Note the first dimension of the filter is the same as the size of the embedding.
 - ▶ Unlike image processing, the filter doesn't have a full width
- ▶ To merge all filters by sliding a set of filters across it:

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

$$\mathbf{x}^{(1)} = f(\mathbf{b})$$

where f is an activation function (e.g., tanh, ReLU), \mathbf{b} is a vector of bias terms, and $*$ is the convolution operator.

Computing the convolution

- ▶ At each position m , compute the element-wise product of the k th filter and the sequence of words of window size h (think of it as an n-gram of length h) starting at m and take its dot product with the vector $\mathbf{x}_{m:m+h-1}^{(0)}$.
- ▶ The value of the filter at position m is computed as

$$x_m^{(1)} = \sum_{k=1}^{K_e} b_k \left(\mathbf{x}_{m:m+h-1}^{(0)} \right)$$

- ▶ When we finish the convolution step, if we have K_f filters of dimension $\mathbb{R}^{K_e \times h}$, then $\mathbf{X}^{(1)} \in \mathbb{R}^{K_f \times M-h+1}$.
- ▶ In practice, filters of different sizes are often used to capture ngrams of different lengths, so $\mathbf{X}^{(1)}$ will be K_f vectors of variable lengths, and we can write the size of each vector of h_k .

Convolution step when processing text

<https://eduassistpro.github.io/>

Assignment Project Exam Help

$X^{(0)}$

U

Conv

0.4	0.7	0.3	0.3	2.3	3.2		
0.5	1.3	0.2	1.5	0.8	0.6	1.8	0
							1
							0

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Input text sequence

=

2.9	3.1	3.4	?	?	?
-----	-----	-----	---	---	---

$X^{(1)}$

Padding

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- ▶ To deal with the beginning and end of the input, the matrix is often padded with $h - 1$ columns at the beginning **convolution**
- ▶ If no padding is a <https://eduassistpro.github.io/>
layer will be $h - 1$ units smaller than the input. This is known as **narrow convolution**.
Add WeChat edu_assist_pro

Pooling

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- ▶ After D convolutional layers, assuming filters have identical lengths, we have a representation of the document $\mathbf{x}^{(D)} \in \mathbb{R}^{n \times 1}$.
- ▶ It is very likely that the filters have different lengths, so we need to turn them into a single vector before feeding them to a feedforward neural network for classification.
- ▶ This can be done by **pooling** across time (over the sequence of words)

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Prediction and training with CNN

<https://eduassistpro.github.io/>

- ▶ The CNN needs to be fed into a feedforward network to make a prediction \hat{y} and compute the loss $\ell^{(i)}$ in training.
- ▶ Parameters of a CNN includes the weight matrix, bias vector, and the filters as well as the biases.
- ▶ The parameter θ , which may involve computing the gradient for the loss function.

$$\frac{\partial z_k}{\partial x_{k,m}^{(D)}} = \begin{cases} 1, & x_{k,m}^{(D)} = \max \left(x_{k,1}^{(D)}, x_{k,2}^{(D)}, \dots, x_{k,M}^{(D)} \right) \\ 0, & \text{Otherwise} \end{cases}$$

Different pooling methods

<https://eduassistpro.github.io/>

Assignment Project Exam Help

- Max pooling

[Add WeChat edu_assist_pro](https://eduassistpro.github.io/)

$$z_k = \max_{k,M} x_{k,1}^{(b)}, x_{k,2}^{(b)}$$

- Average pool

<https://eduassistpro.github.io/>

[Add WeChat edu_assist_pro](https://eduassistpro.github.io/)

$$z_k = \frac{1}{M} \sum_{m=1}^M x_{k,m}$$

A graphic representation of a CNN

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro

Figure 1: Caption

Using Conv Nets in PyTorch

<https://eduassistpro.github.io/>

Assignment Project Exam Help

Assignment Project Add WeChat edu_assist_pro

<https://eduassistpro.github.io/>

Add WeChat edu_assist_pro