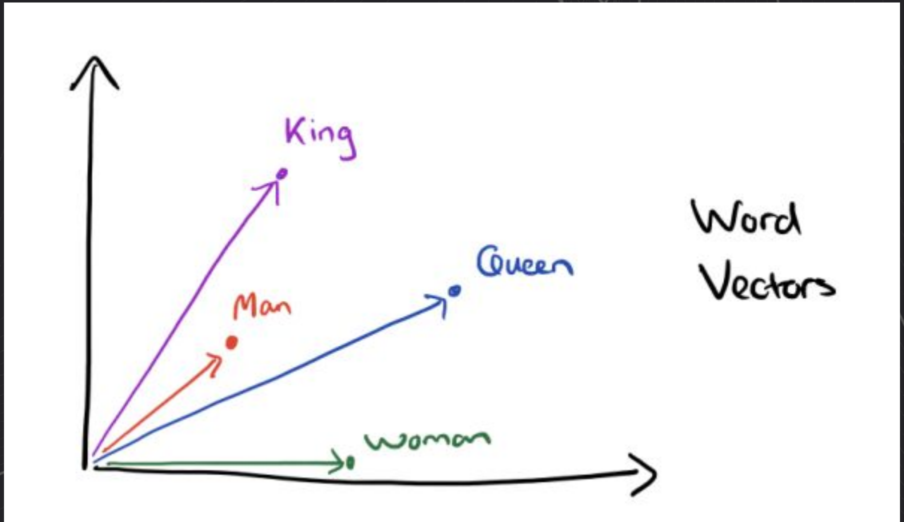
SkillFactory

# Word embeddings
# CNN for texts

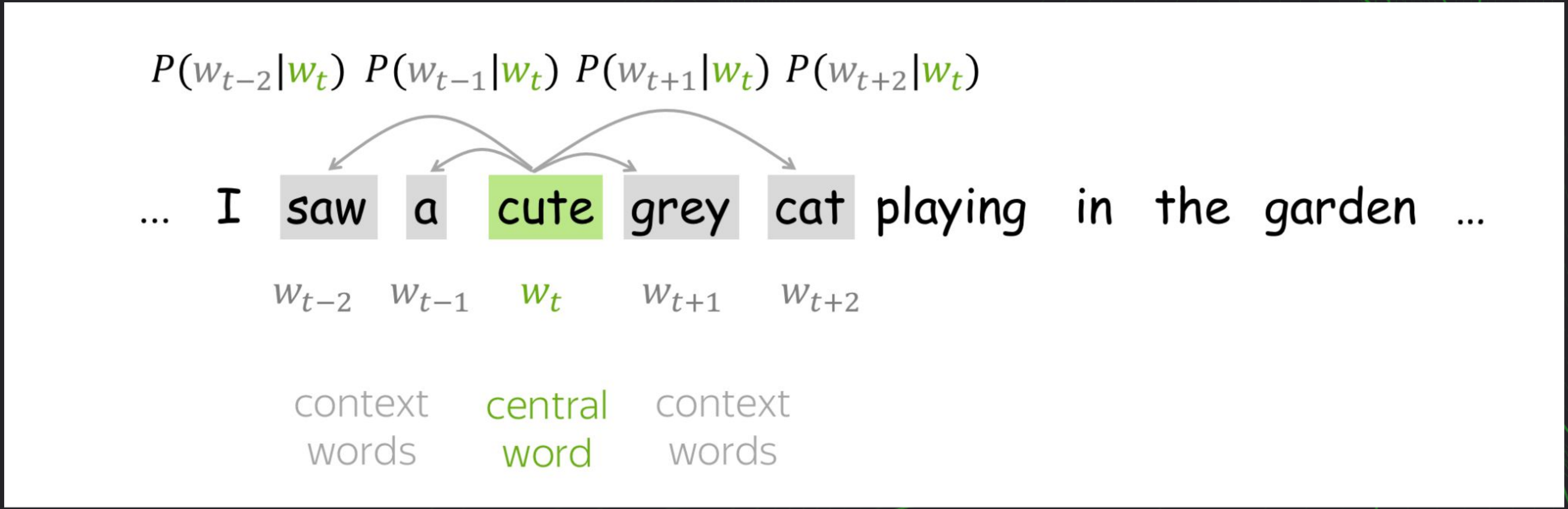

**Sidorov Nikita**

MLE (NLP) in Sber

# Word2vec

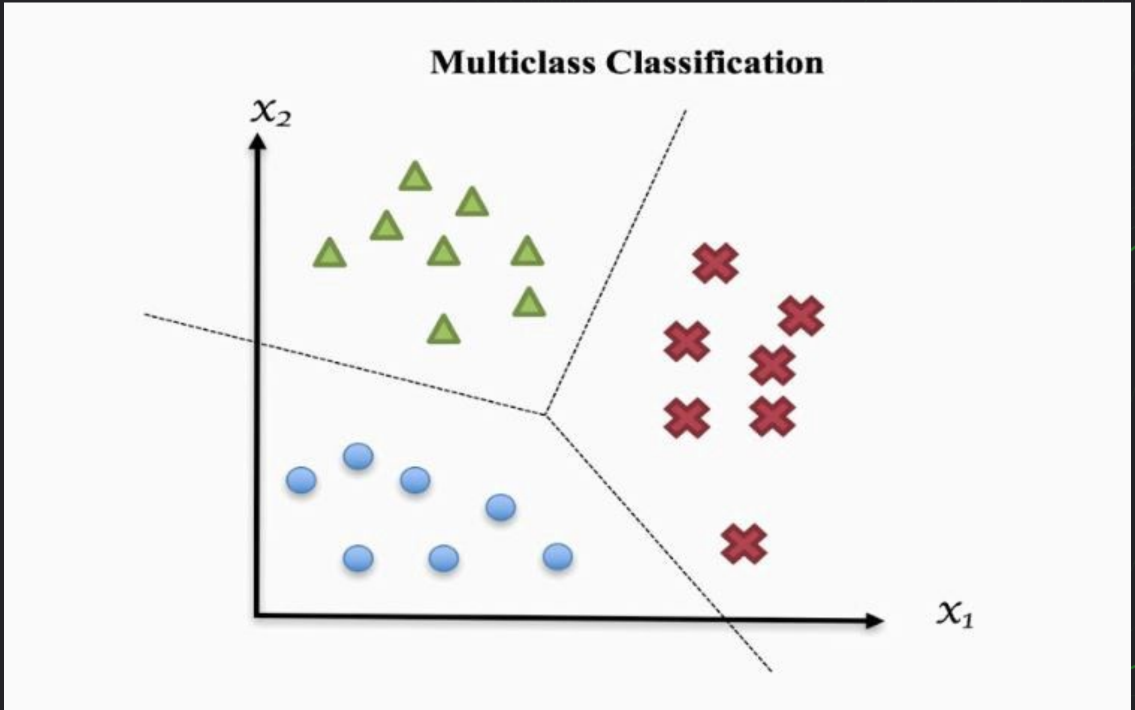


# What task is solving Word2vec

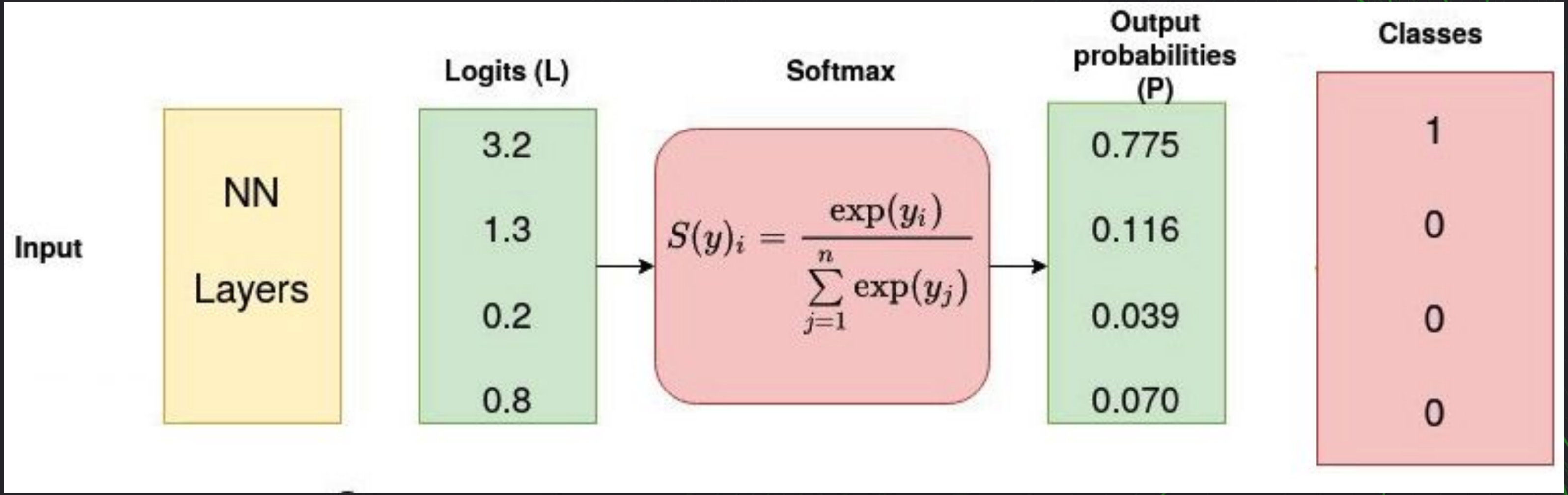



example from Lena Voita NLP course

# What task is solving Word2vec



# Cross-entropy



# Cross-entropy

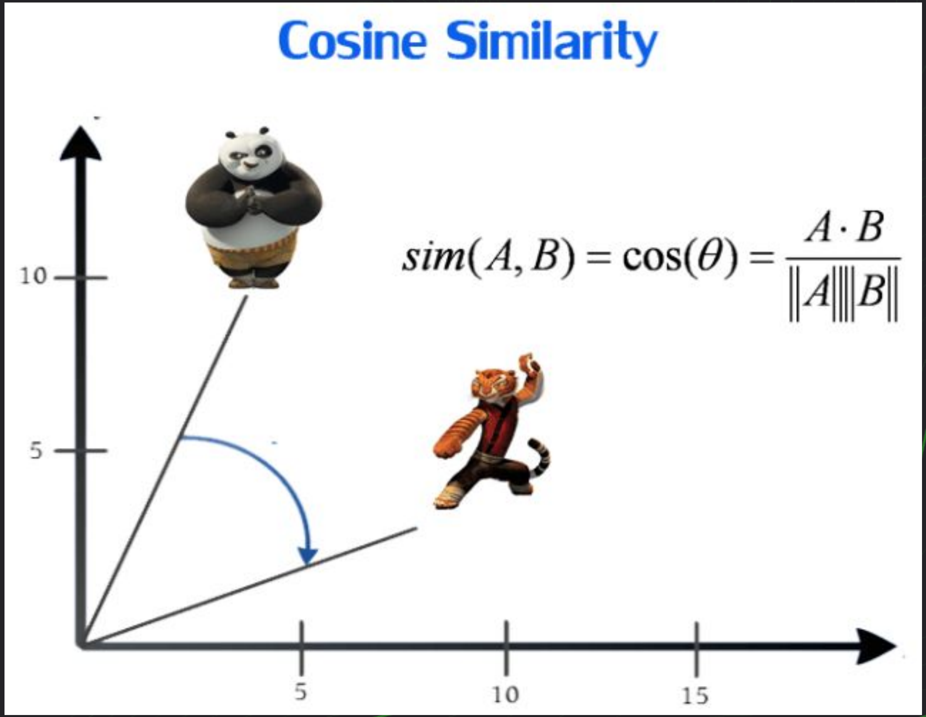
$$CCE(p,t) = -\sum_{c=1}^{C} t_{o,c} \log\left(p_{o,c}\right)$$

# Distributional semantics

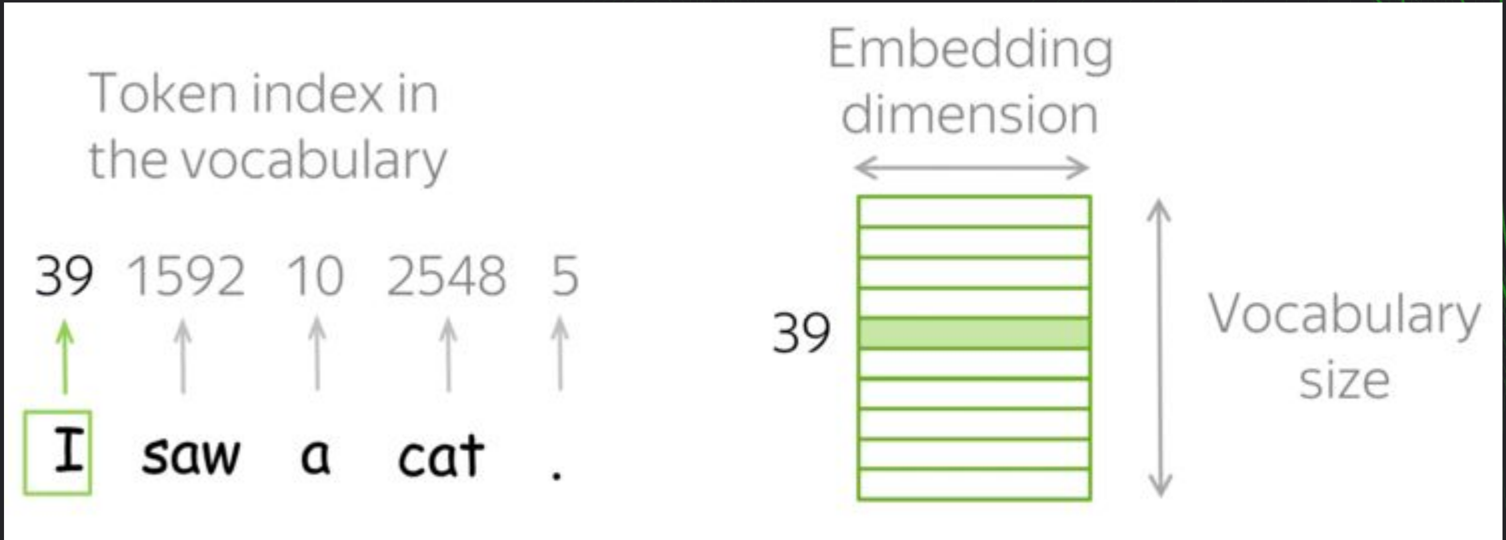
1. A bottle of _____ is on the table.
2. Everybody likes _____.
3. Don`t have _____ before you drive.
4. We make _____ out of corn.

example from Jacob Eisenstein's NLP notes

# How to measure distances between vectors?



## What we want from word embeddings?



# What we want from word embeddings?

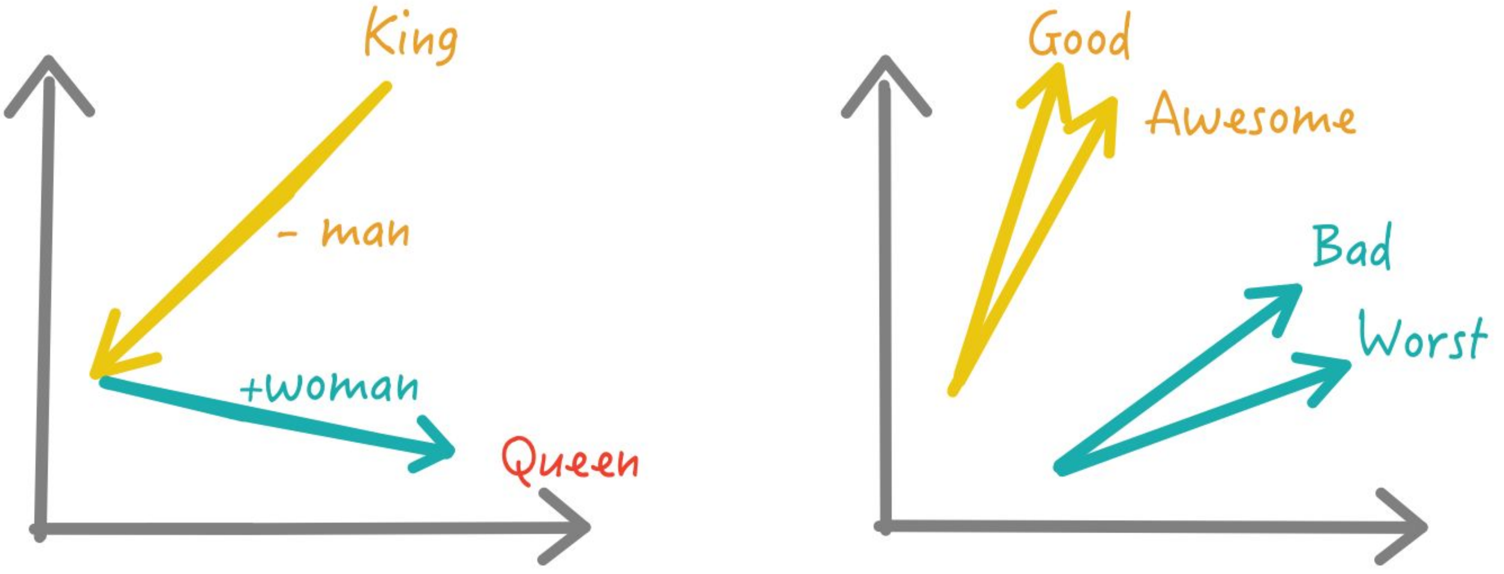
$$\begin{bmatrix} 0 & 0 & 0 & \boxed{1} & 0 \end{bmatrix} \times \begin{bmatrix} 8 & 2 & 1 & 9 \\ 6 & 5 & 4 & 0 \\ 7 & 1 & 6 & 2 \\ \boxed{1 \quad 3 \quad 5 \quad 8} \\ 0 & 4 & 9 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 & 8 \end{bmatrix}$$

One-hot vector

Embedding Weight Matrix

Hidden layer output

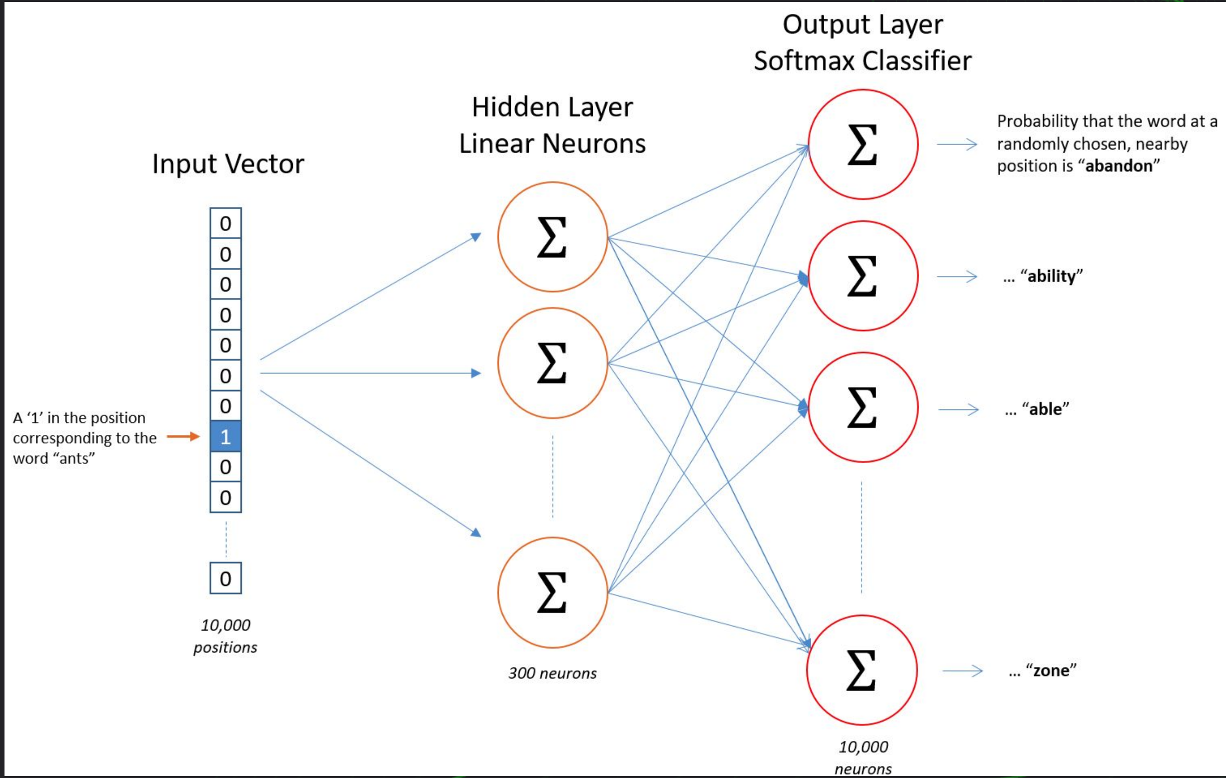
# Word2vec




a) Learns Analogy

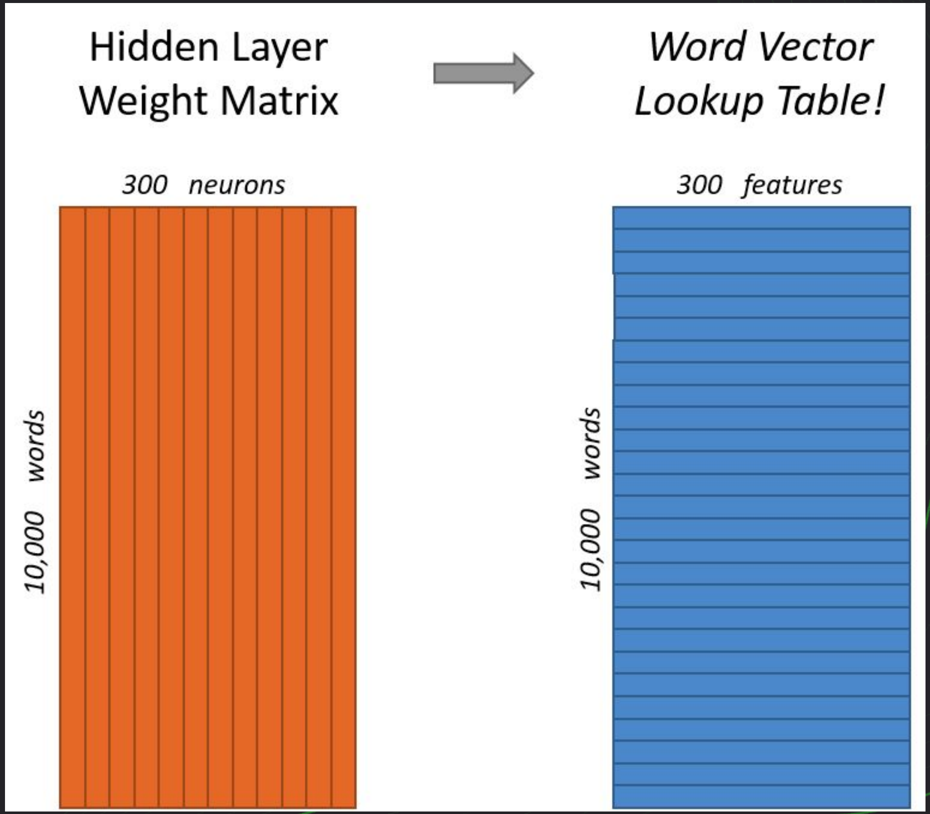
b) Similar Words have same angles

# Word2vec



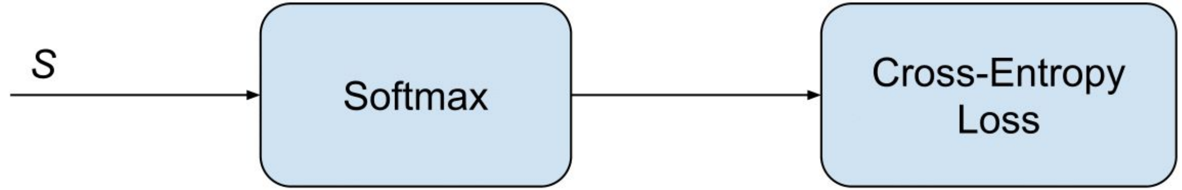

example from [Chris McCormick blog]()

# Word2vec


Hidden Layer
Weight Matrix
Word Vector
Lookup Table!
300 neurons
10,000 words
300 features
10,000 words

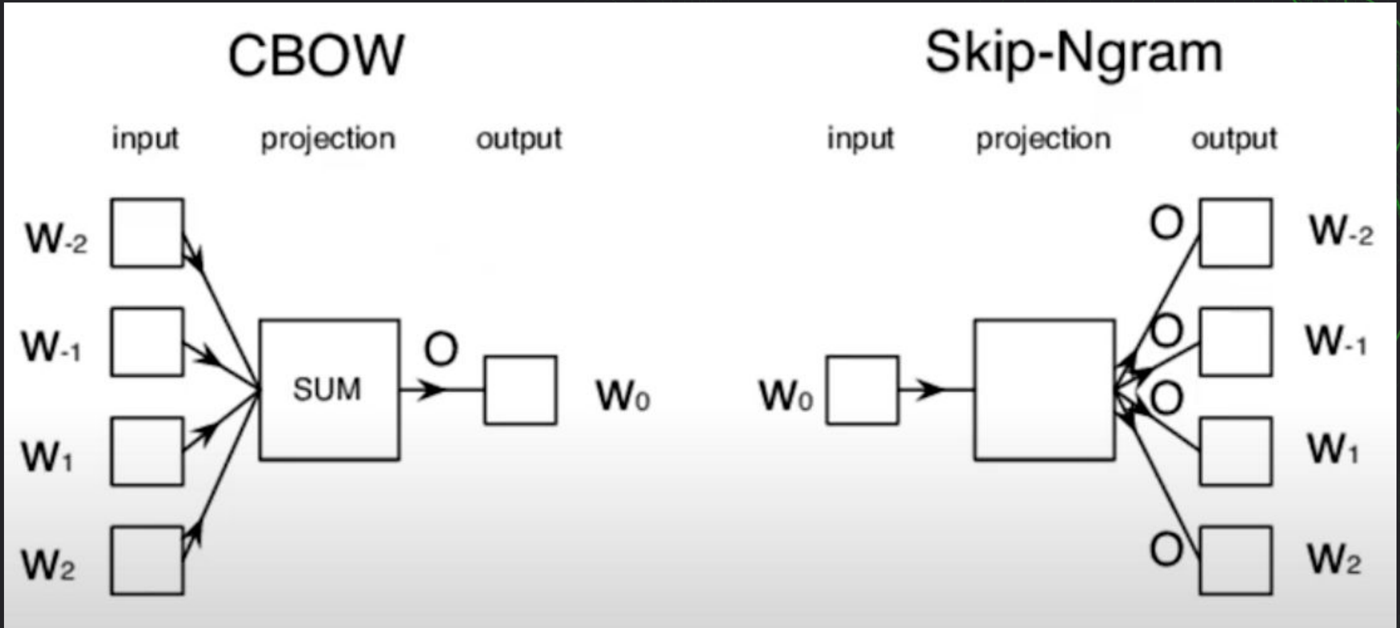
# Word2vec

How to get probabilities for occurrence of words?



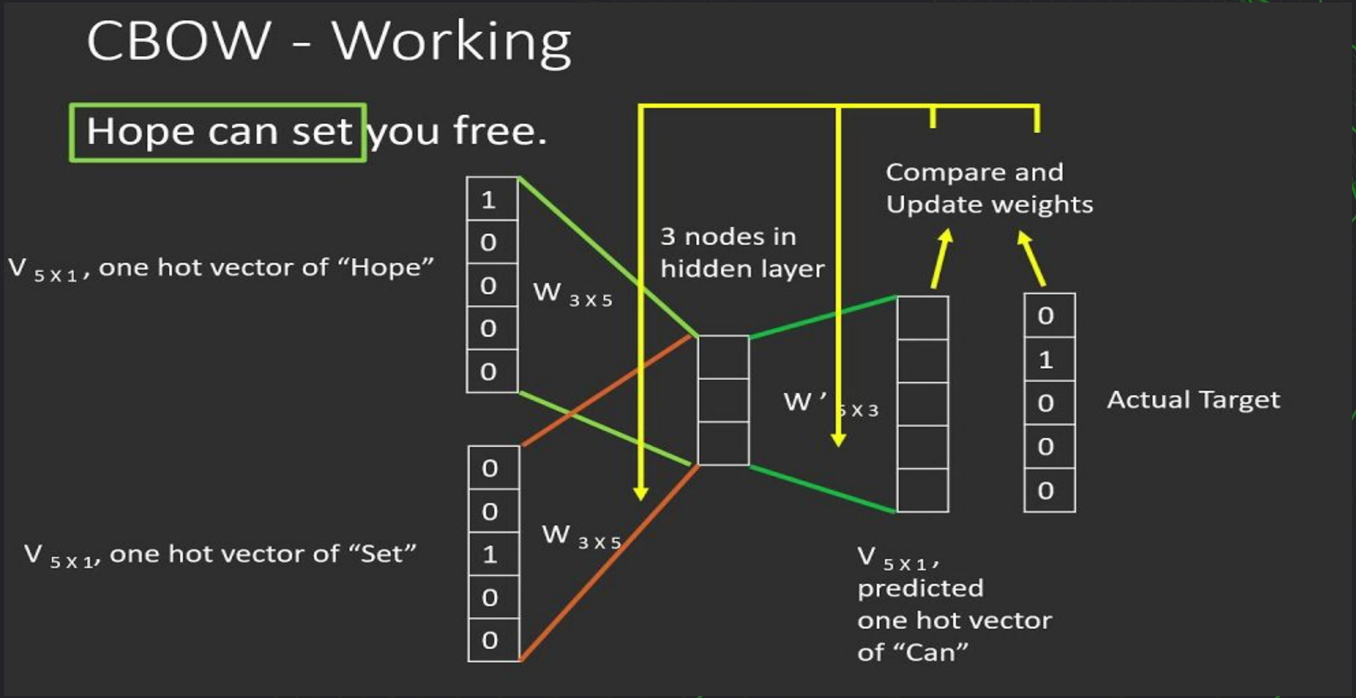

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_i^C t_i log(f(s)_i)$$

# Word2vec approaches for training

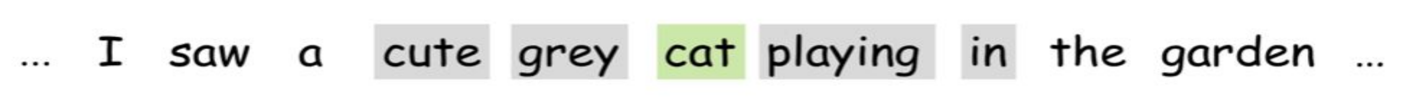


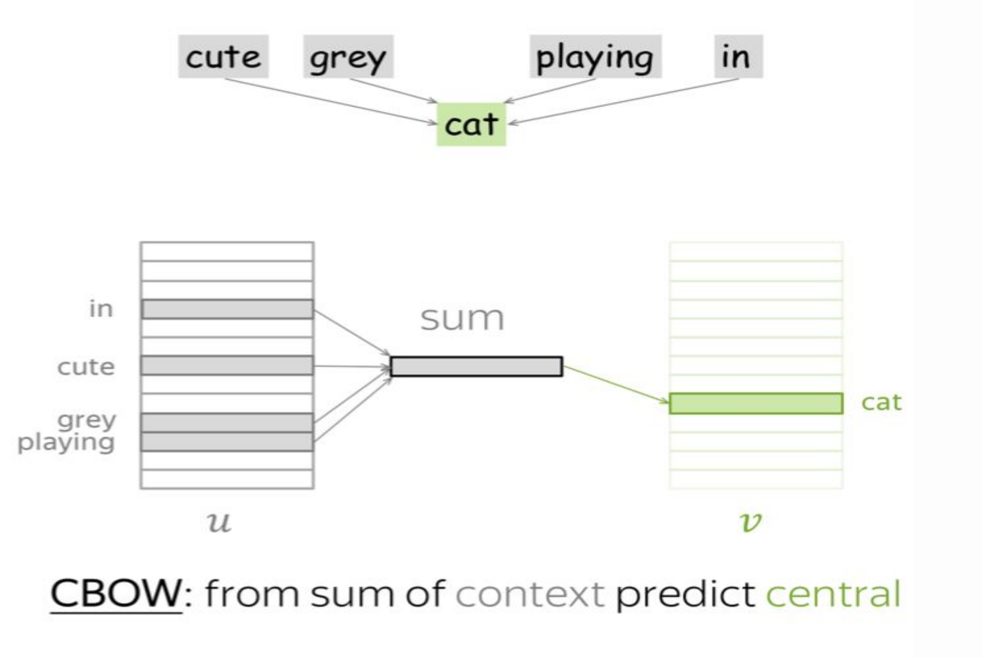
Word2vec
CNN for texts

# CBOW



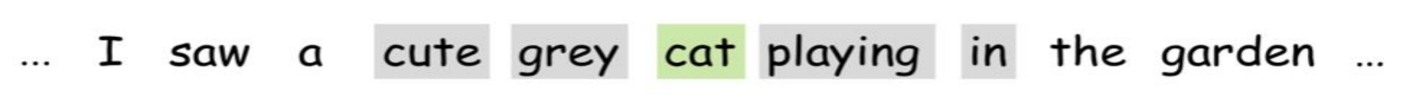
# CBOW

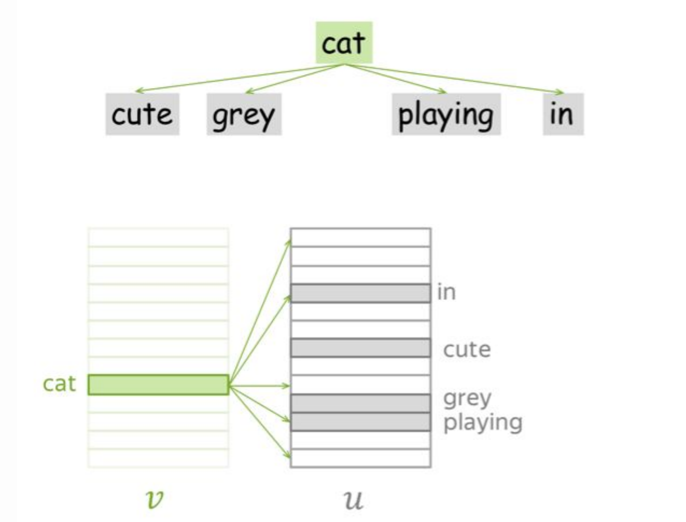



example from [Lena Voita NLP course]

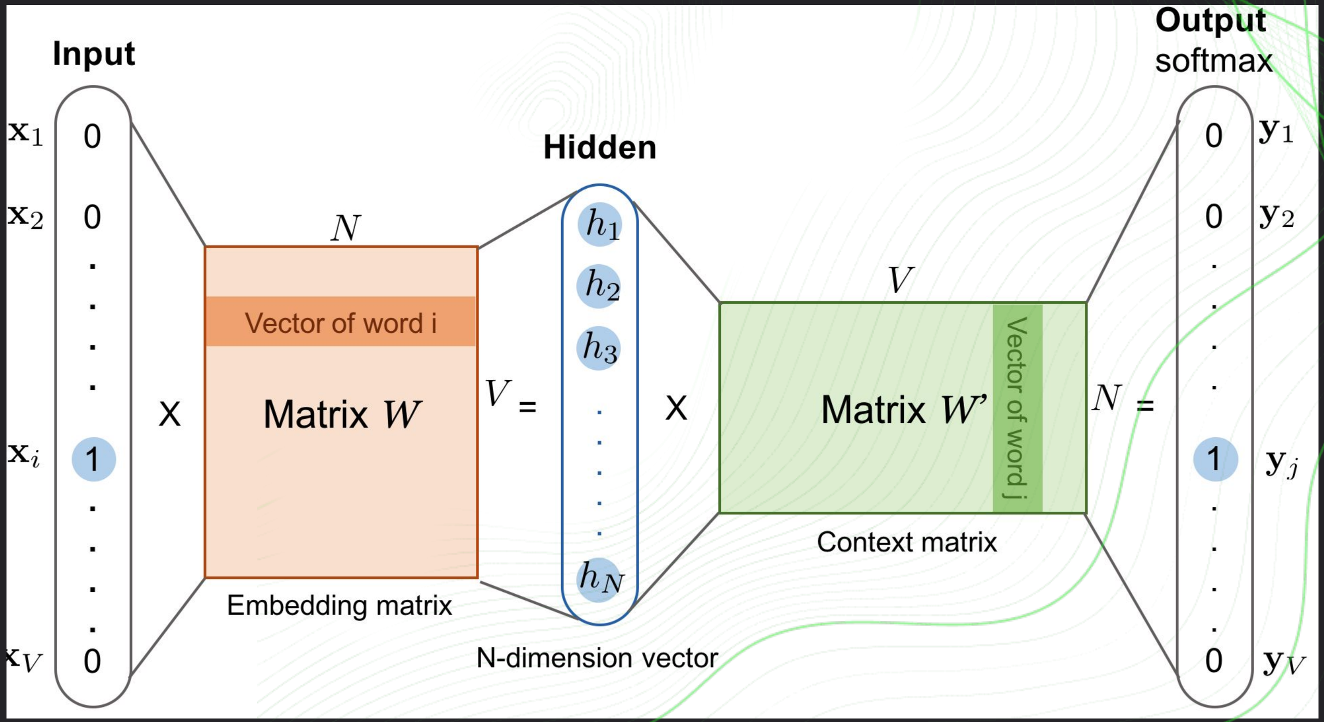
# Skip-gram



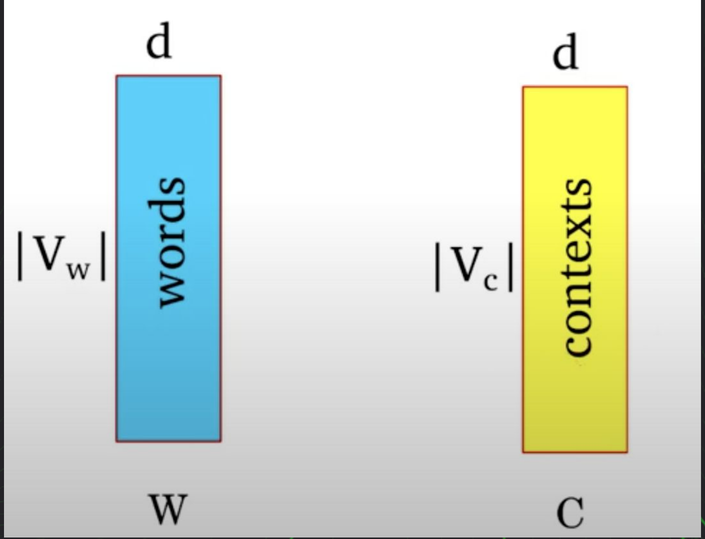

Skip-Gram: from central predict context (one at a time)

# How to learn


Input
Output
softmax
$x_1$
$x_2$
$x_i$
$x_V$
0
0
1
0
X
N
Vector of word i
Matrix $W$
Embedding matrix
Hidden
$h_1$
$h_2$
$h_3$
$h_N$
$V$ =
N-dimension vector
X
V
Matrix $W'$
Vector of word j
Context matrix
$N$ =
0
0
1
0
$y_1$
$y_2$
$y_j$
$y_V$

# How to learn

1) For each word we would have vector of context
2) Represent each context as a *d* dimensional vector
3) Initialize all vectors to random weights
4) Arrange vectors in two matrices W and C

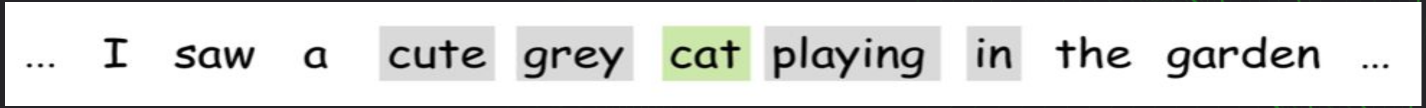


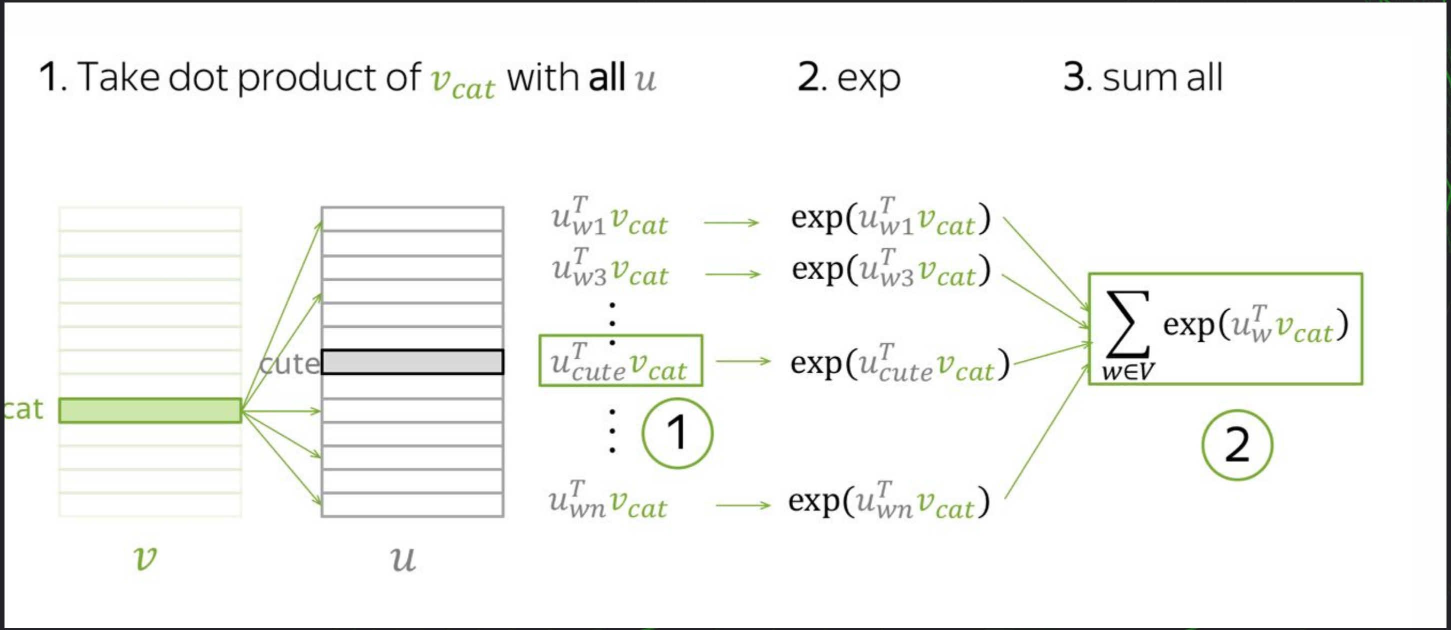
# How to learn

$$\log p(c \mid w; \theta) = \frac{\exp v_c \cdot v_w}{\sum_{c' \in C} \exp v_{c'} \cdot v_w}$$

- predict context word(s)
- from word w

## How to learn




example from Lena Voita NLP course

# How to learn

... I saw a cute grey cat playing in the garden ...

4. get loss (for this one step)

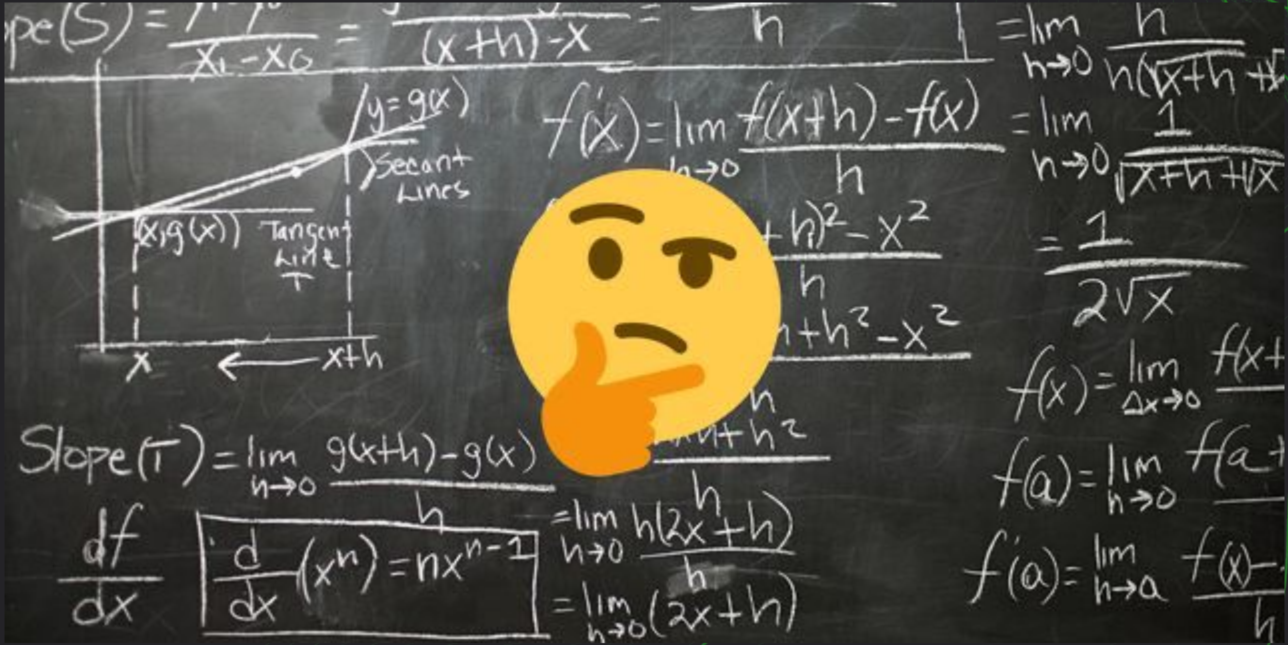
$$J_{t,j}(\theta) = \underbrace{-u_{cute}^{T} v_{cat}}_{1} + \underbrace{\log \sum_{w \in V} \exp(u_w^{T} v_{cat})}_{2}$$

5. evaluate the gradient, make an update

$$v_{cat} := v_{cat} - \alpha \frac{\partial J_{t,j}(\theta)}{\partial v_{cat}}$$

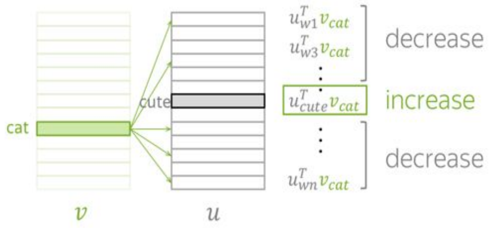
$$u_w := u_w - \alpha \frac{\partial J_{t,j}(\theta)}{\partial u_w} \ \forall\, w \in V$$

## What`s the problem?



# Negative sampling

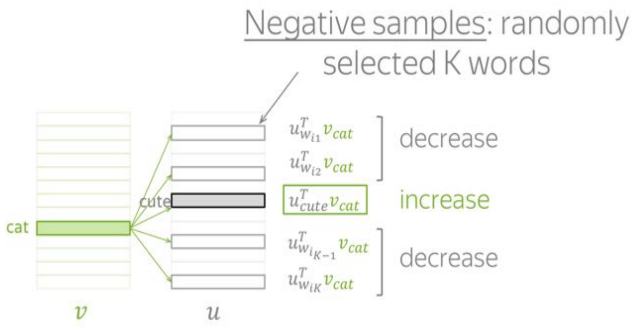
Dot product of $v_{cat}$:
- with $u_{cute}$ - increase,
- with **all other** $u$ - decrease




Parameters to be updated:
- $v_{cat}$
- $u_w$ for all $w$ in the vocabulary

$|V| + 1$ vectors

Dot product of $v_{cat}$:
- with $u_{cute}$ - increase,
- with **a subset of other** $u$ - decrease

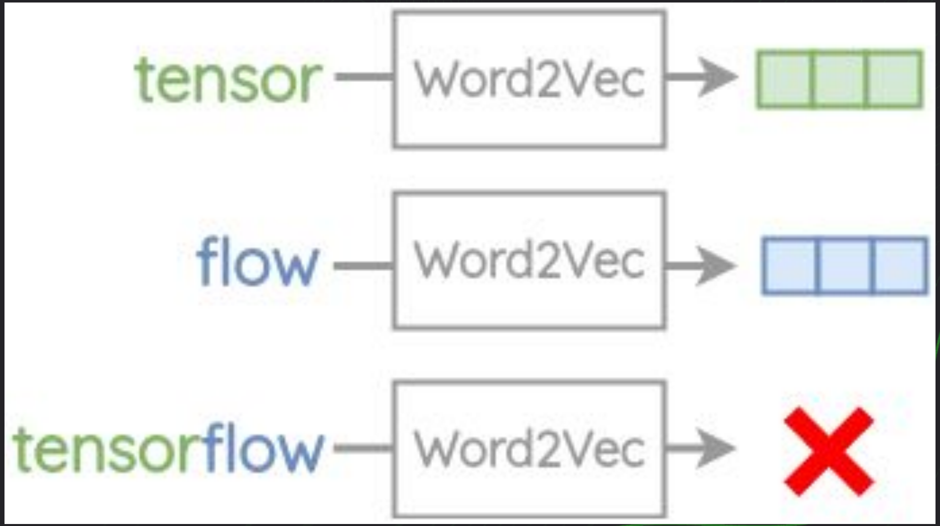



Parameters to be updated:
- $v_{cat}$
- $u_{cute}$ and $u_w$ for $w$ in K negative examples

$K + 2$ vectors

example from Lena Voita NLP course

# OOV words



# Fasttext

- Take not only words, but n-grams in this words
- harder to compute
- longer to train
- bigger models
- well works for morphologically rich languages

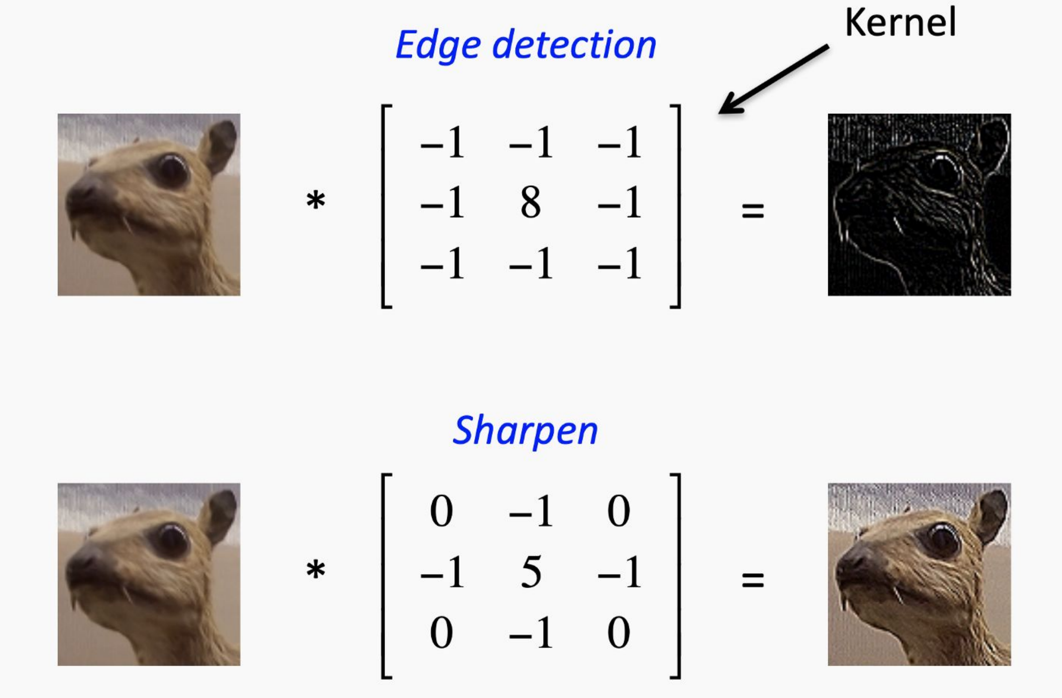
# Fasttext

- Skip-gram model as base model
- take each word and n-grams for it (from 3 to 6)
- for reducing space we use hashing trick
- negative sampling is our everything

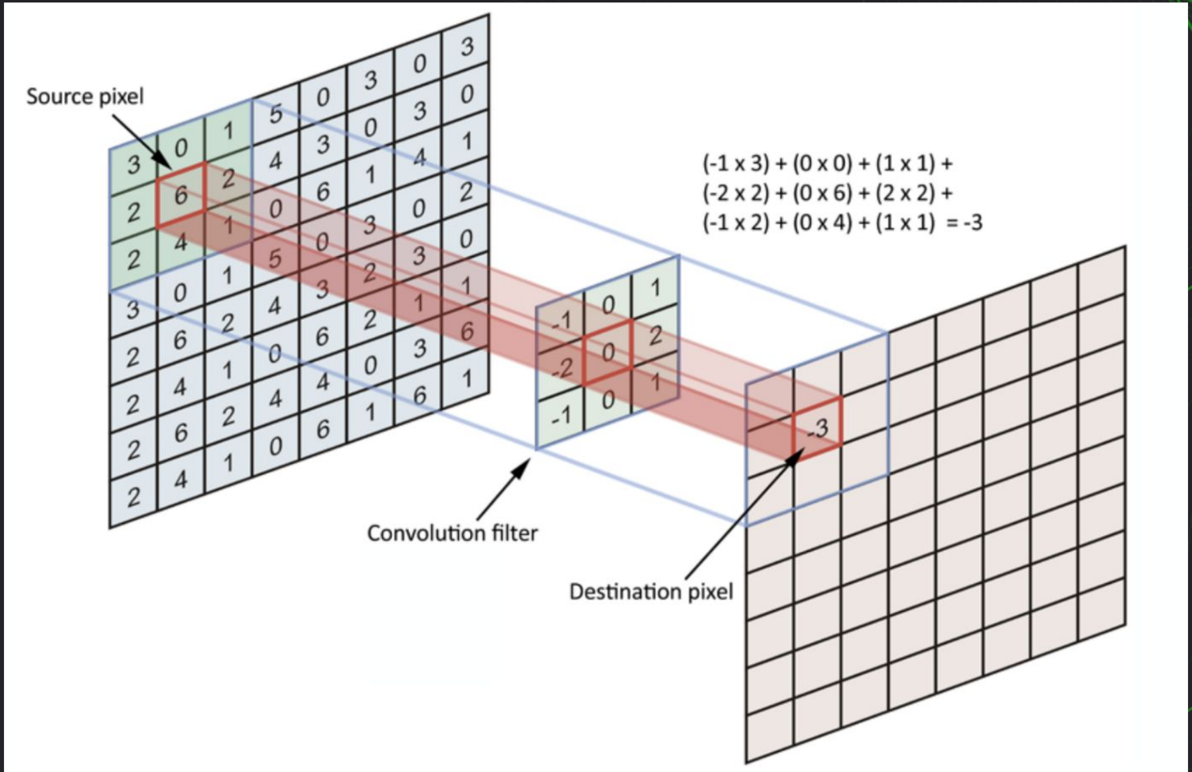
# Convolutions

In deep learning, a convolutional neural network (CNN) is a class of artificial neural network, most commonly applied to analyze visual imagery. They are also known as artificial neural networks that slide along input features and provide translation equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are only equivariant, as opposed to invariant, to translation.

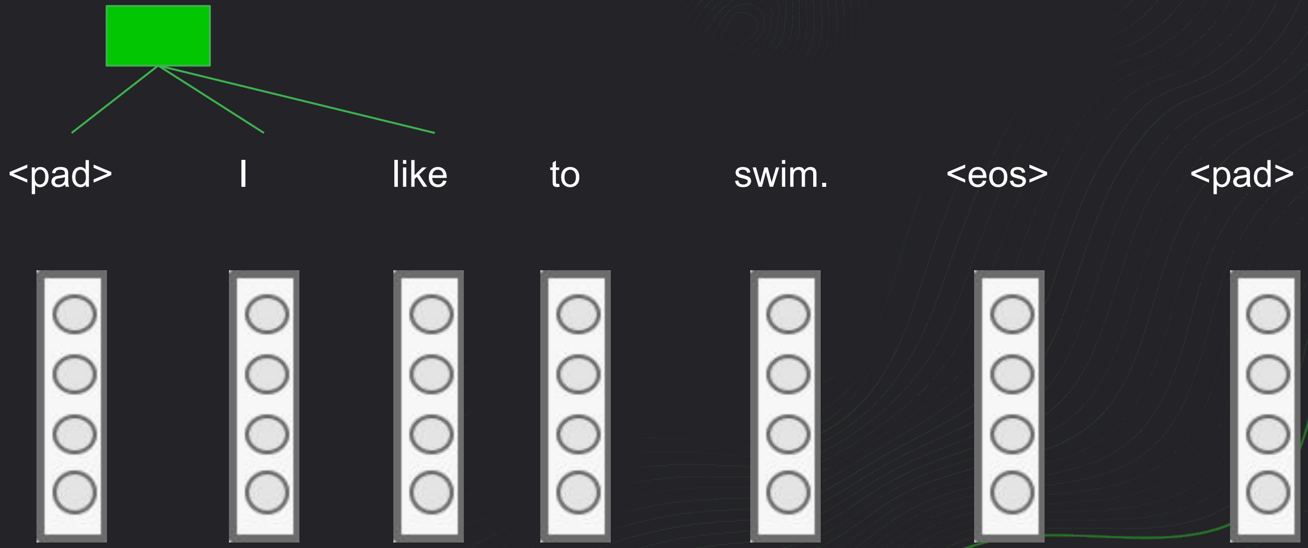
# Convolutions



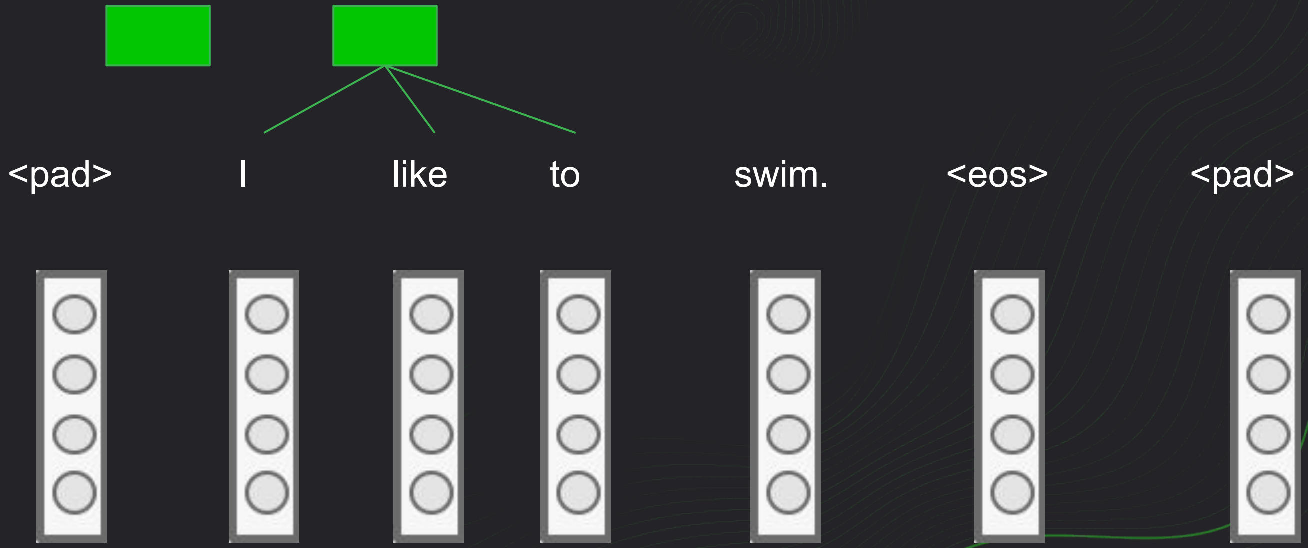
# Convolutions



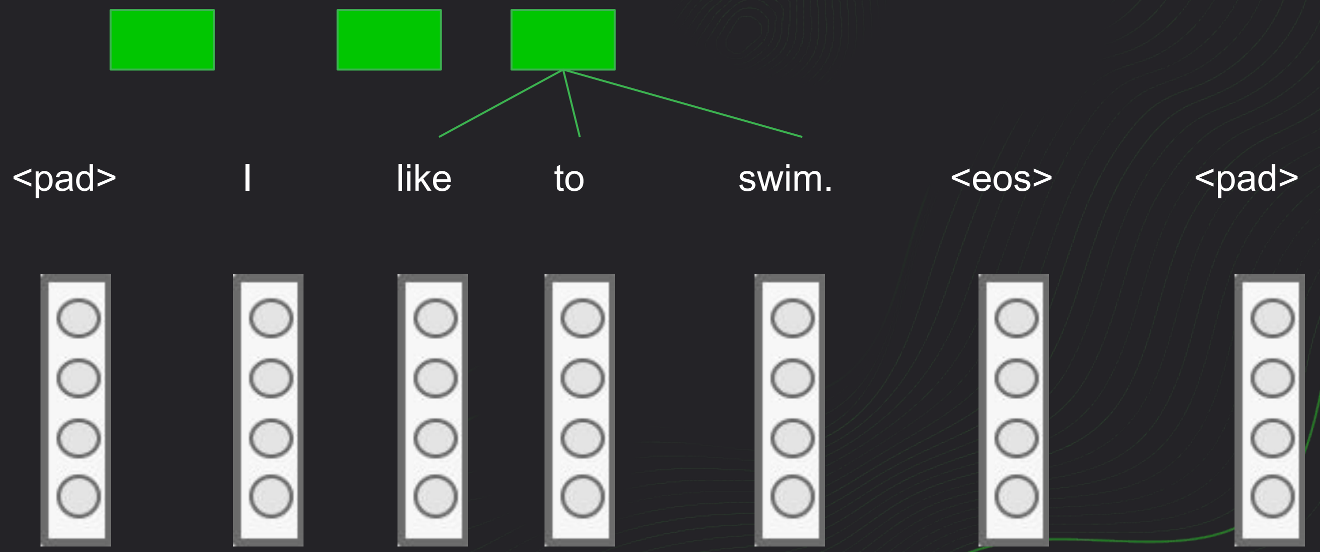
# Convolution for texts



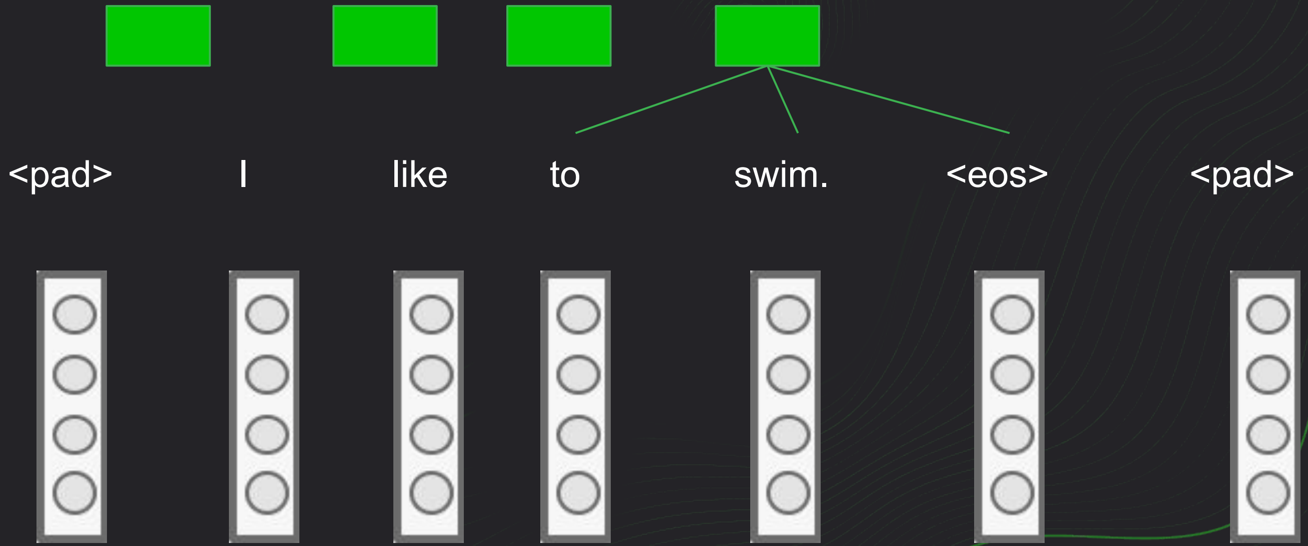
# Convolution for texts



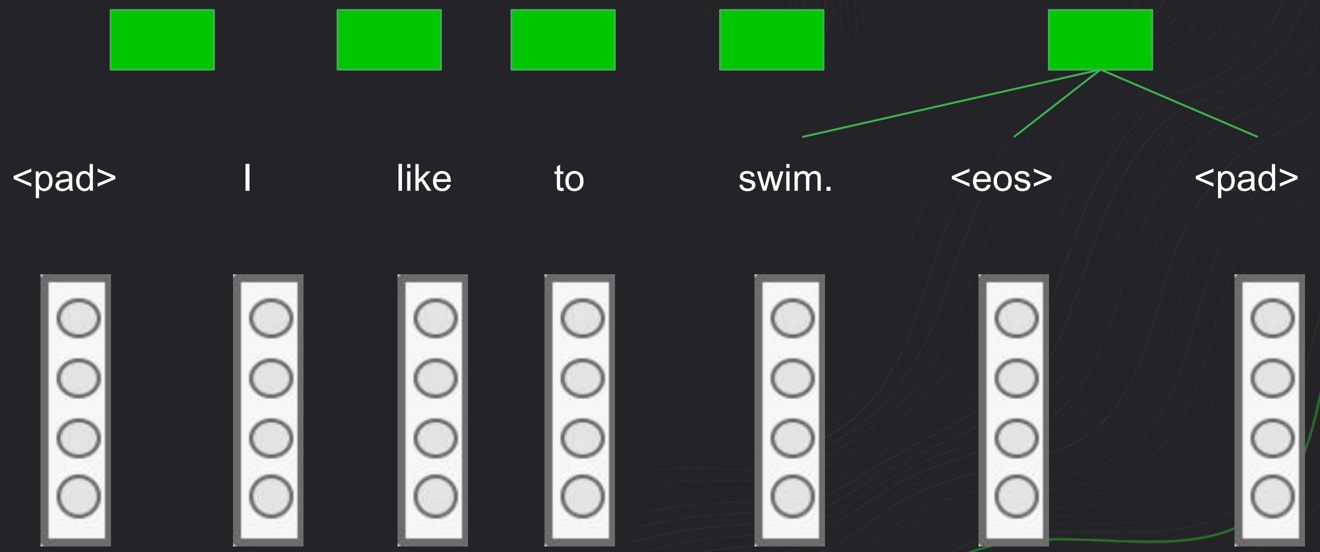
# Convolution for texts



# Convolution for texts



# Convolution for texts



# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | |
| ILT | |
| LTS | |
| TS< | |
| S<< | |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | 4.2 |
| ILT | |
| LTS | |
| TS< | |
| S<< | |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | 4.2 |
| ILT | 2.4 |
| LTS | |
| TS< | |
| S<< | |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | 4.2 |
| ILT | 2.4 |
| LTS | 2.5 |
| TS< | |
| S<< | |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

Apply filters of size 3 and that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | 4.2 |
| ILT | 2.4 |
| LTS | 2.5 |
| TS< | 3.2 |
| S<< | |

# Convolution for texts

Text

| | | | | |
|---|---|---|---|---|
| <pad> | 0.3 | 0.4 | -0.2 | -0.6 |
| I | 0.5 | 0.1 | -0.3 | 0.4 |
| like | -0.1 | 0.5 | 0.8 | -0.2 |
| to | 0.2 | -0.3 | -0.4 | -0.5 |
| swim. | -0.7 | 0.5 | 0.9 | 0.1 |
| <eos> | -0.4 | 0.4 | 0.1 | -0.5 |
| <pad> | 0.1 | -0.6 | -0.3 | 0.2 |

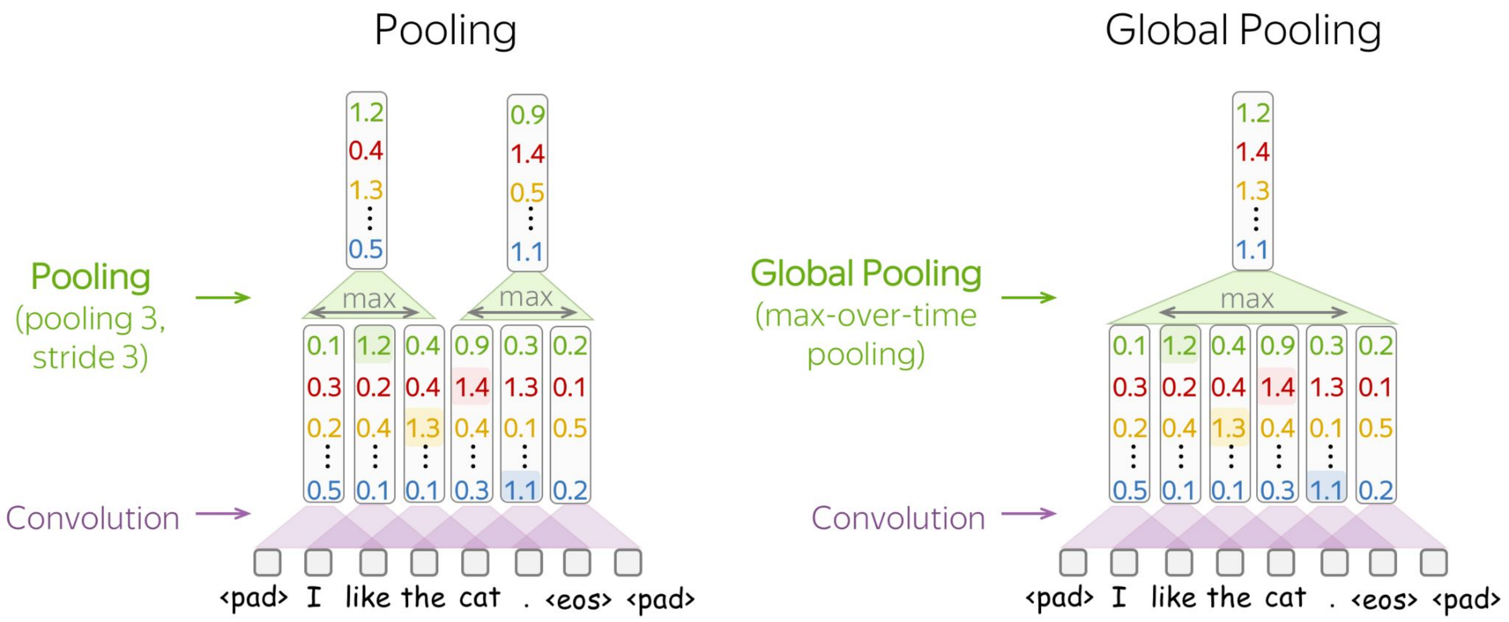
Apply filter of size 3, that have 4 channels

| | | | |
|---|---|---|---|
| 3 | 2 | 1 | -1 |
| 1 | 0 | 2 | 1 |
| -1 | 1 | 1 | -2 |

Result

| | |
|---|---|
| <IL | 4.2 |
| ILT | 2.4 |
| LTS | 2.5 |
| TS< | 3.2 |
| S<< | -2.4 |

# Pooling for convolutions




example from [Lena Voita NLP course]