Introduction to NLP

Oxana Vitman

Words representations

One-hot encoding

Bag of words

Term-document matrix

Word2Vec

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

Vocabulary
I
like
my
cat
the
is
black
funny

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

Index	*Vocabulary*
7	I
1	like
3	my
2	cat
5	the
4	is
0	black
6	funny

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

Index	*Vocabulary*	0 1 2 3 4 5 6 7
7	I	[0 0 0 0 0 0 0 1]
1	like	[0 1 0 0 0 0 0 0]
3	my	[0 0 0 1 0 0 0 0]
2	cat	[0 0 1 0 0 0 0 0]
5	the	[0 0 0 0 0 1 0 0]
4	is	[0 0 0 0 1 0 0 0]
0	black	[1 0 0 0 0 0 0 0]
6	funny	[0 0 0 0 0 0 1 0]

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

Index	*Vocabulary*	0 1 2 3 4 5 6 7
7	I	[0 0 0 0 0 0 0 1]
1	like	[0 1 0 0 0 0 0 0]
3	my	[0 0 0 1 0 0 0 0]
2	cat	[0 0 1 0 0 0 0 0]
5	the	[0 0 0 0 0 1 0 0]
4	is	[0 0 0 0 1 0 0 0]
0	black	[1 0 0 0 0 0 0 0]
6	funny	[0 0 0 0 0 0 1 0]

One-Hot Encoding

I like my cat

The cat is black

My cat is funny

Index	*Vocabulary*	0 1 2 3 4 5 6 7
7	I	[0 0 0 0 0 0 0 1]
1	like	[0 1 0 0 0 0 0 0]
3	my	[0 0 0 1 0 0 0 0]
2	cat	[0 0 1 0 0 0 0 0]
5	the	[0 0 0 0 0 1 0 0]
4	is	[0 0 0 0 1 0 0 0]
0	black	[1 0 0 0 0 0 0 0]
6	funny	[0 0 0 0 0 0 1 0]

One-Hot Encoding

Disadvantage?

Index	*Vocabulary*	0 1 2 3 4 5 6 7
7	I	[0 0 0 0 0 0 0 1]
1	like	[0 1 0 0 0 0 0 0]
3	my	[0 0 0 1 0 0 0 0]
2	cat	[0 0 1 0 0 0 0 0]
5	the	[0 0 0 0 0 1 0 0]
4	is	[0 0 0 0 1 0 0 0]
0	black	[1 0 0 0 0 0 0 0]
6	funny	[0 0 0 0 0 0 1 0]

One-Hot Encoding

Disadvantage?

- Size
- Sparsity
- Semantics (words meaning and their relations)

Index	*Vocabulary*	0 1 2 3 4 5 6 7
7	I	[0 0 0 0 0 0 0 1]
1	like	[0 1 0 0 0 0 0 0]
3	my	[0 0 0 1 0 0 0 0]
2	cat	[0 0 1 0 0 0 0 0]
5	the	[0 0 0 0 0 1 0 0]
4	is	[0 0 0 0 1 0 0 0]
0	black	[1 0 0 0 0 0 0 0]
6	funny	[0 0 0 0 0 0 1 0]

And what is words meaning?

tezgüino

And what is words meaning?

A bottle of tezgüino is on the table.

Everyone likes tezgüino.

Tezgüino makes you drunk.

We make tezgüino out of corn.

And what is words meaning?

A bottle of tezgüino is on the table.

Everyone likes tezgüino.

Tezgüino makes you drunk.

We make tezgüino out of corn.

→ Tezgüino is an alcoholic beverage made of corn

Thank you, context

Context

1. A bottle of _______ is on the table.
2. Everyone likes _______.
3. _______ makes you drunk.
4. We make _______ out of corn.

What other words could fit into these context?

Context

1. A bottle of _______ is on the table.
2. Everyone likes _______.
3. _______ makes you drunk.
4. We make _______ out of corn.

What other words could fit into these context?

	1	2	3	4
tezgüino	1	1	1	1
loud	0	0	0	0
motor oil	1	0	0	1
tortillas	0	1	0	1
wine	1	1	1	0

⟸ context

⟸ *1*: if word can appear in the context
0: it can not

Context

1. A bottle of ______ is on the table.
2. Everyone likes ______.
3. ______ makes you drunk.
4. We make ______ out of corn.

	1	2	3	4
tezgüino	1	1	1	1
loud	0	0	0	0
motor oil	1	0	0	1
tortillas	0	1	0	1
wine	1	1	1	0

rows are similar

Context

1. A bottle of _______ is on the table.
2. Everyone likes _______.
3. _______ makes you drunk.
4. We make _______ out of corn.

	1	2	3	4
tezgüino	1	1	1	1
loud	0	0	0	0
motor oil	1	0	0	1
tortillas	0	1	0	1
wine	1	1	1	0

Distributional hypothesis

rows are similar → word meanings are similar

Word2vec: Idea

Transform information about the **context** into **word vectors**

Word2vec: Idea

Transform information about the **context** into **word vectors**

How? **Learn** word vectors by teaching them to **predict context**

Word2vec: Pipeline

- take a huge text corpus

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

Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time

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

… I	saw	a	cute	grey	cat	playing	in	the	garden …
w_{t-2}	w_{t-1}	w_t	w_{t+1}	w_{t+2}					

Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time

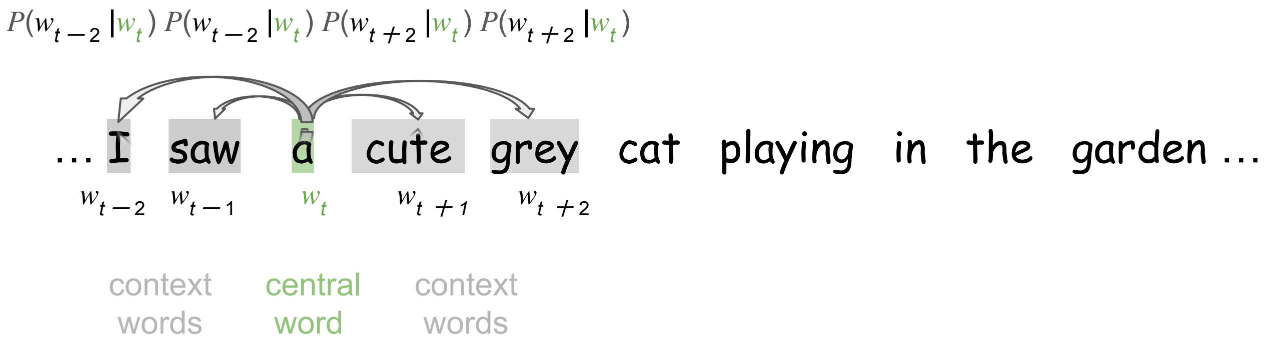
… I saw a cute grey cat playing in the garden …

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

context words | central word | context words

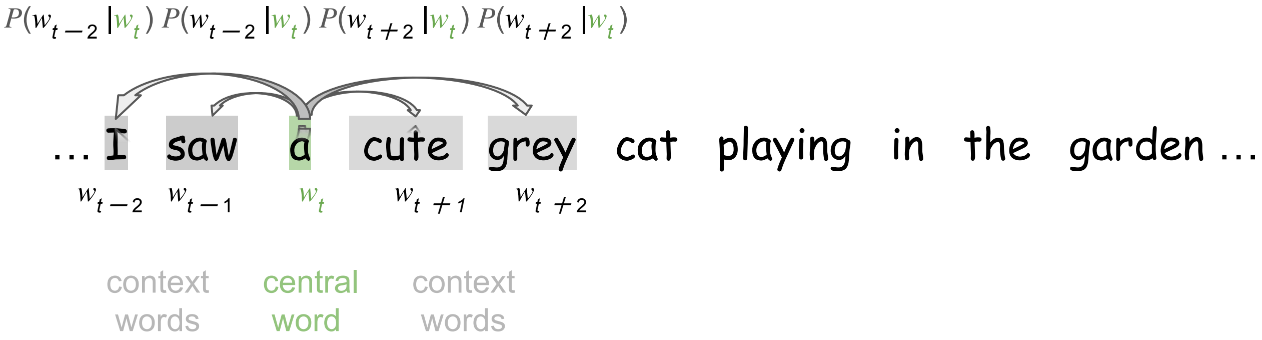
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word



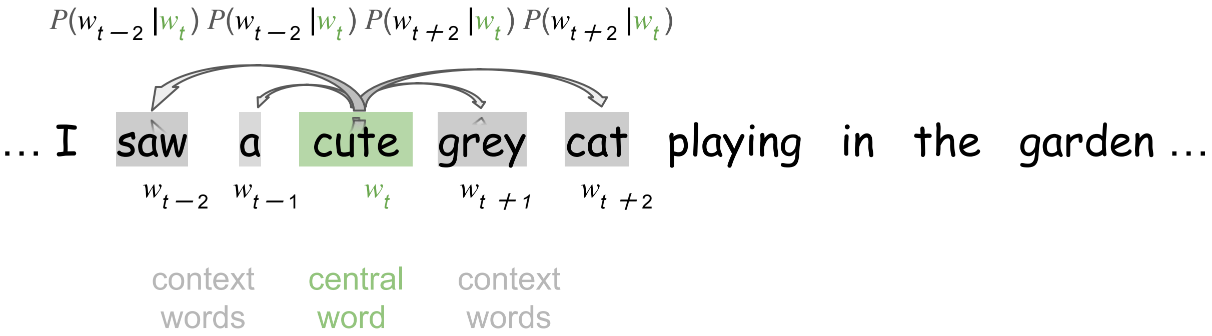
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



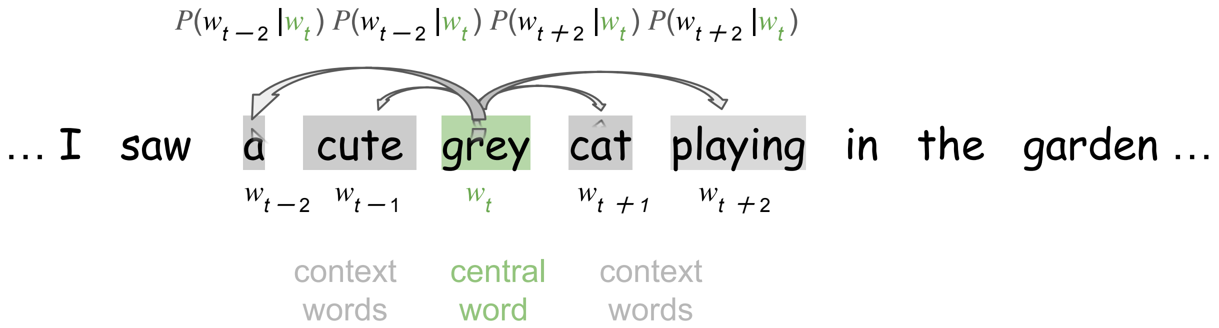
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



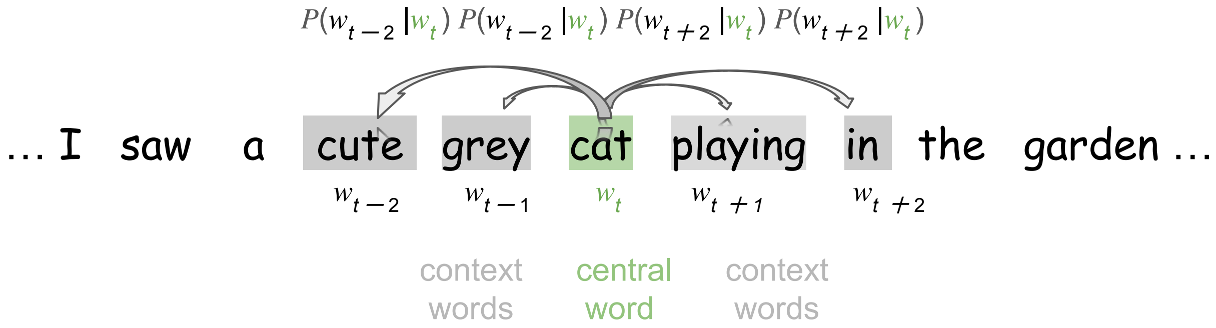
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



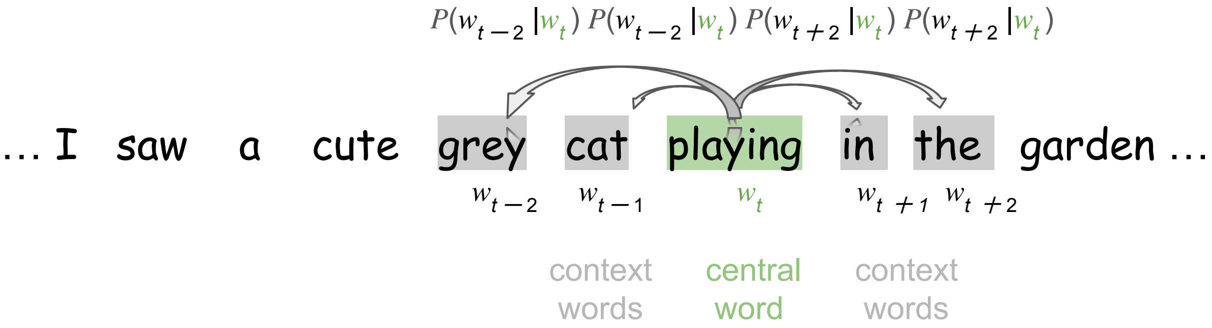
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



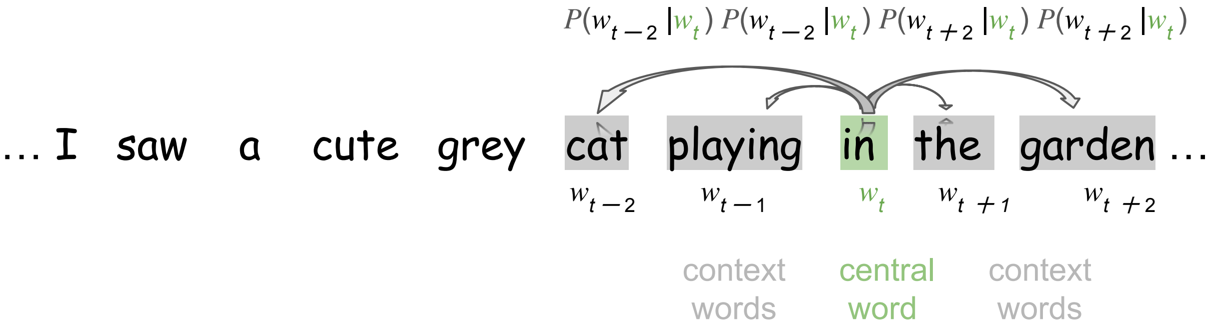
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



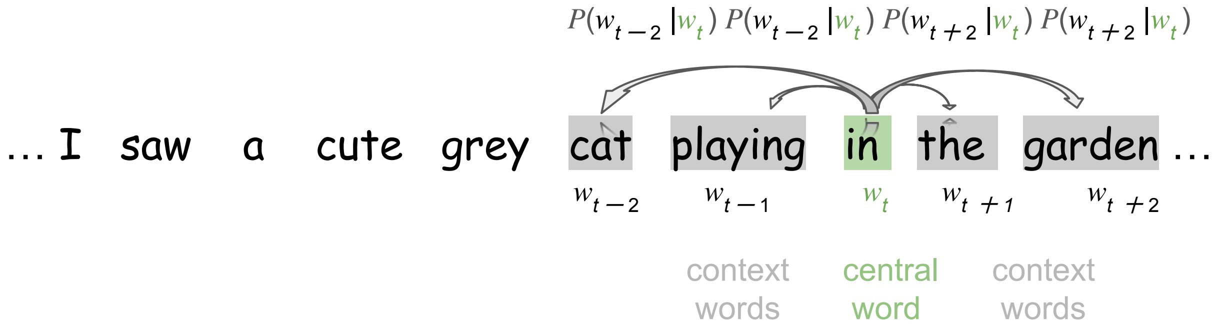
Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- calculate the loss and adjust the vectors



Word2vec: Pipeline

- take a huge text corpus
- go over the text with a sliding window, moving one word at a time
- for the central word, compute the probabilities of context word
- **calculate the loss** and adjust the vectors



Word2vec: Pipeline

- **calculate the loss** and adjust the vectors HOW?

What is the **input** and what are the **target values**?

Word2vec: Pipeline

I saw a cute grey cat playing in the garden

Main	Context	Label
a	I	1
a	saw	1
a	cute	1
a	grey	1

Word2vec: Pipeline

I saw a cute grey cat playing in the garden

I saw a cute grey cat playing in the garden

Main	Context	Label
a	I	1
a	saw	1
a	cute	1
a	grey	1
cute	saw	1
cute	a	1
cute	grey	1
cute	cat	1

Word2vec: Pipeline

I saw a cute grey cat playing in the
garden

value		target	
a	… 0 0 0 1 0 0 …	I	… 1 0 0 0 0 0 …
a	… 0 0 0 1 0 0 …	saw	… 0 1 0 0 0 0 …
a	… 0 0 0 1 0 0 …	cute	… 0 0 0 0 0 1 …
a	… 0 0 0 1 0 0 …	grey	… 0 0 1 0 0 0 …

Word2vec: Pipeline

$$[0\ \ 0\ \ 0\ \ 1\ \ 0] \times \begin{pmatrix} 1 & 0.5 & 1.7 & -1.3 \\ 0.7 & 2\ 3 & 1.2 & 5 \\ 2.1 & 1.3 & -0.5 & 0.2 \\ 3.2 & 1.3 & 0.2 & 0.8 \\ 0.2 & 3 & 5.7 & 0.5 \end{pmatrix} = [3.2\ \ 1.3\ \ 0.2\ \ 0.8]$$

Word2vec: Pipeline

$$[0 \;\; 0 \;\; 0 \;\; 1 \;\; 0] \times \begin{pmatrix} 1 & 0.5 & 1.7 & -1.3 \\ 0.7 & 2\ 3 & 1.2 & 5 \\ 2.1 & 1.3 & -0.5 & 0.2 \\ 3.2 & 1.3 & 0.2 & 0.8 \\ 0.2 & 3 & 5.7 & 0.5 \end{pmatrix} = [3.2 \;\; 1.3 \;\; 0.2 \;\; 0.8]$$

One-hot encoded word vector

Randomly initialized weights matrix

Word embedding

Word2vec: Pipeline

$$[0\ \ 0\ \ 0\ \ 1\ \ 0] \times \begin{pmatrix} 1 & 0.5 & 1.7 & -1.3 \\ 0.7 & 2\ 3 & 1.2 & 5 \\ 2.1 & 1.3 & -0.5 & 0.2 \\ 3.2 & 1.3 & 0.2 & 0.8 \\ 0.2 & 3 & 5.7 & 0.5 \end{pmatrix} = \mathit{Softmax}\left([3.2\ \ 1.3\ \ 0.2\ \ 0.8]\right) = [\mathit{0.775}\ \ \mathit{0.116}\ \ \mathit{0.039}\ \ \mathit{0.07}]$$

One-hot encoded word vector

Randomly initialized weights matrix

Word embedding

Probabilities

Cross-Entropy Loss Function

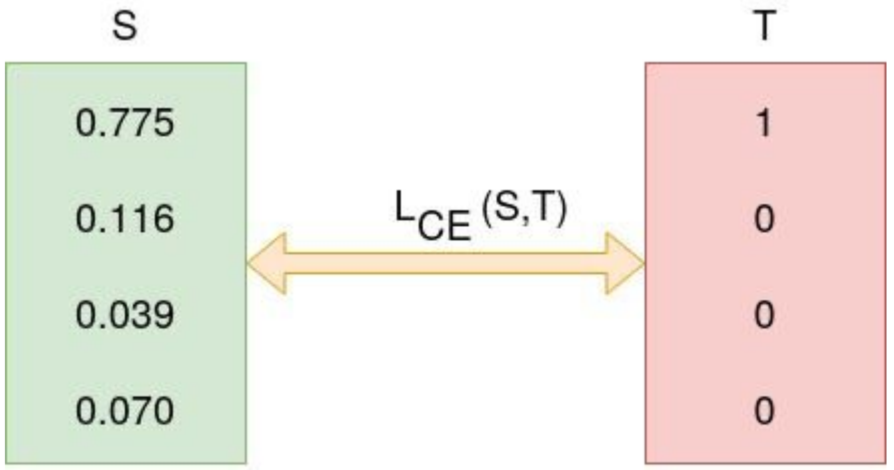


Illustration: Kiprono Elijan Koech

Cross-Entropy Loss Function

$$L_{\text{CE}} = -\sum_{i=1}^{n} t_i \log(p_i), \quad \text{for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

Illustration: Kiprono Elijan Koech

Cross-Entropy Loss Function

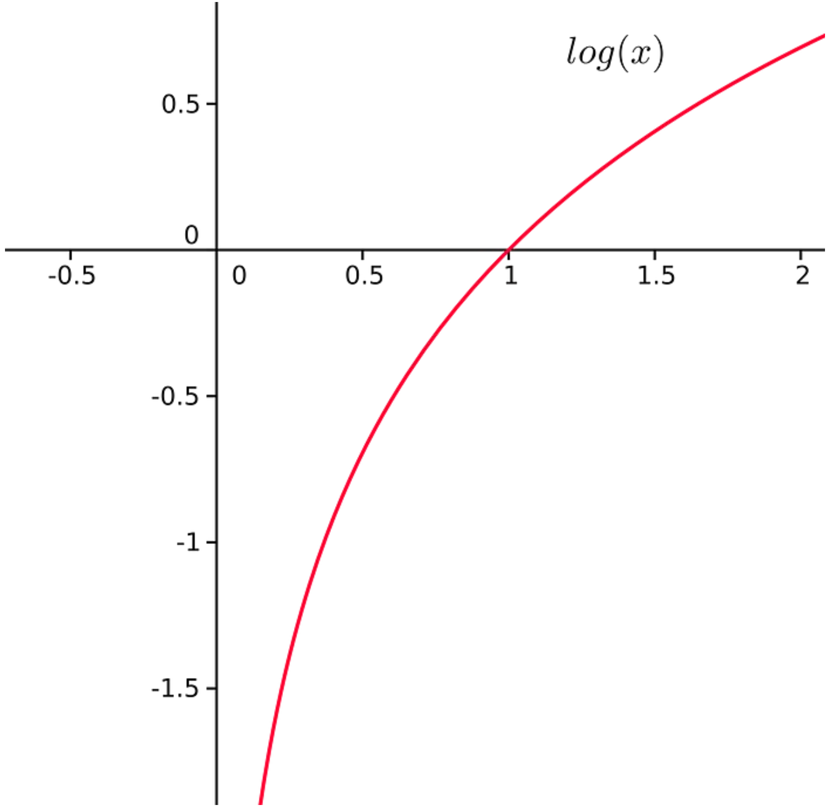
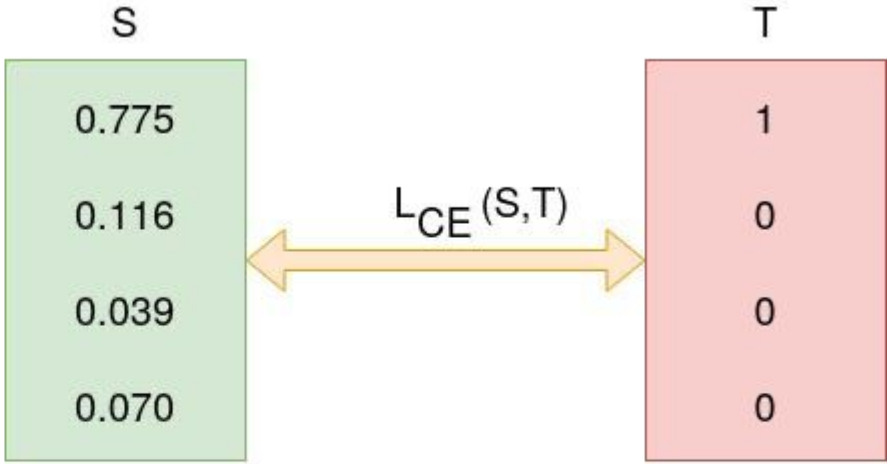


Illustration: Kiprono Elijan Koech

Cross-Entropy Loss Function



$$
\begin{aligned}
L_{CE} &= -\sum_{i=1} T_i \log(S_i) \\
&= -\left[1\log_2(0.775) + 0\log_2(0.126) + 0\log_2(0.039) + 0\log_2(0.070)\right] \\
&= -\log_2(0.775) \\
&= 0.3677
\end{aligned}
$$

Kiprono Elijan Koech

Cross-Entropy Loss Function

Updating weights matrix

$$\begin{pmatrix} 1 & 0.5 & 1.7 & -1.3 \\ 0.7 & 2\ 3 & 1.2 & 5 \\ 2.1 & 1.3 & -0.5 & 0.2 \\ 3.2 - \mathit{0.8} & 1.3 - \mathit{4.8} & 0.2 - \mathit{4.5} & 0.8 - \mathit{4.5} \\ 0.2 & 3 & 5.7 & 0.5 \end{pmatrix}$$

Word2vec: Pipeline

$$[0\ \ 0\ \ 0\ \ 1\ \ 0] \times \begin{pmatrix} 1 & 0.5 & 1.7 & -1.3 \\ 0.7 & 2\ 3 & 1.2 & 5 \\ 2.1 & 1.3 & -0.5 & 0.2 \\ 2.6, & -3.5\ , & -4.3, & -3.7 \\ 0.2 & 3 & 5.7 & 0.5 \end{pmatrix} = \mathit{Softmax} \left([2.6, -3.5, -4.3, -3.7] \right) = [\mathit{0.938}\quad \mathit{0.028}\quad \mathit{0.013}\quad \mathit{0.023}]$$

One-hot encoded word vector

Updated matrix

Word embedding

Probabilities

Cross-Entropy Loss Function

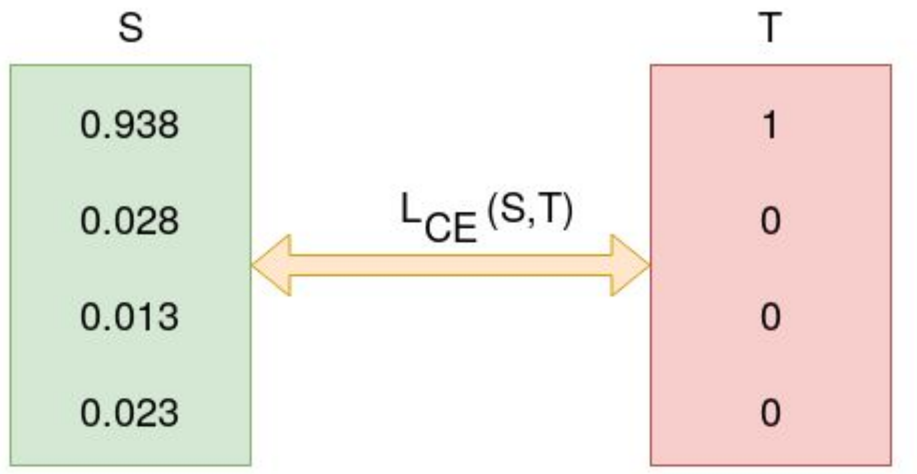
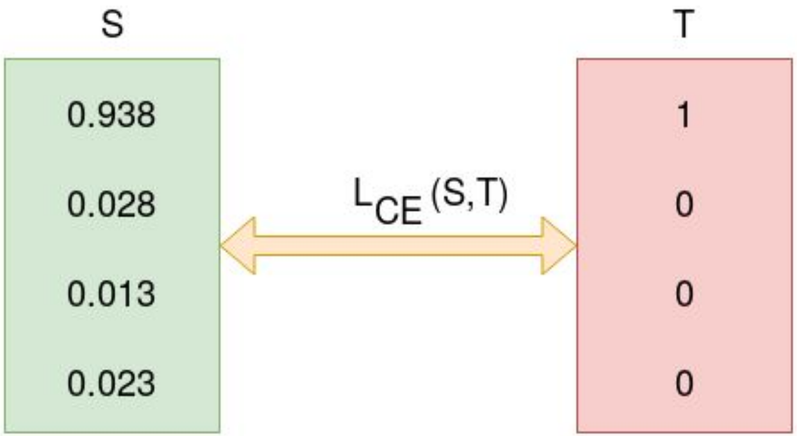


Illustration: Kiprono Elijan Koech

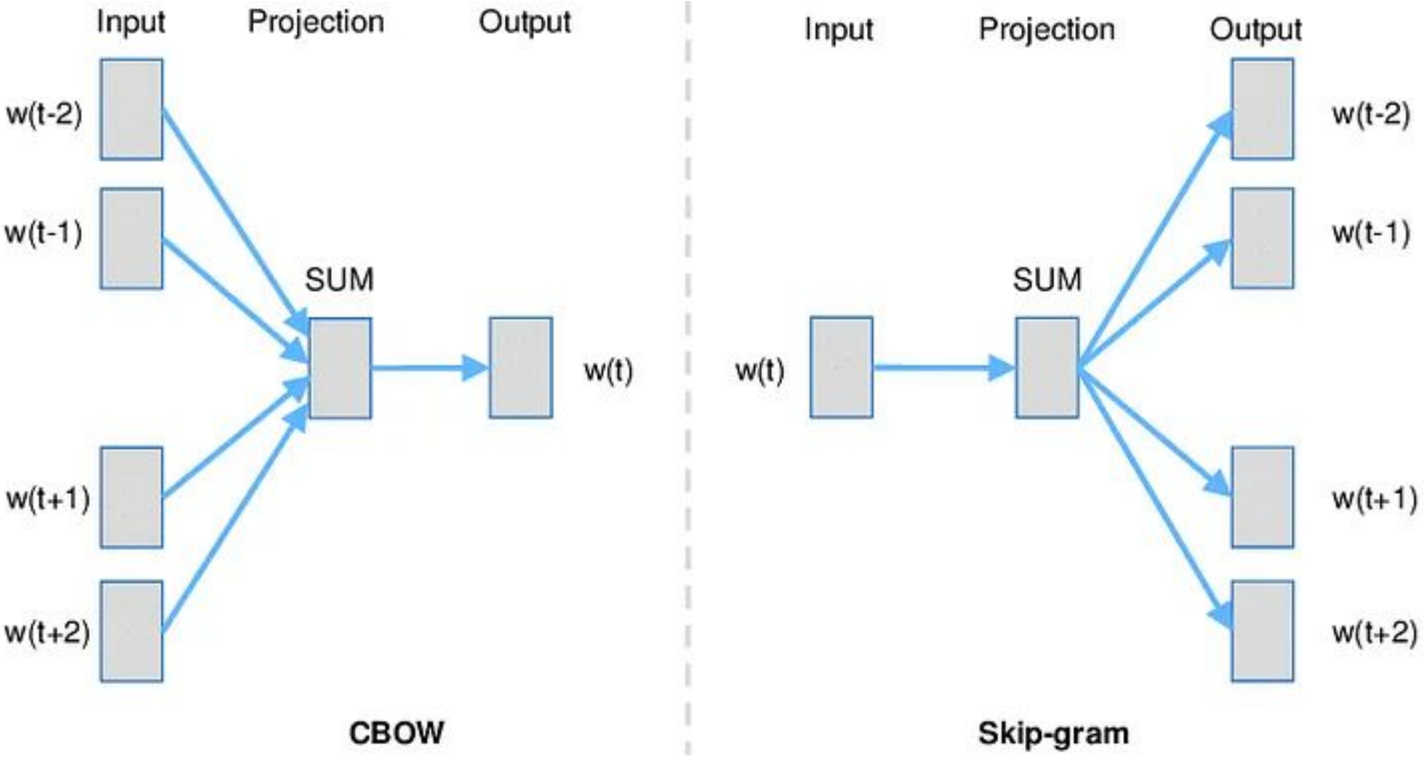
Cross-Entropy Loss Function



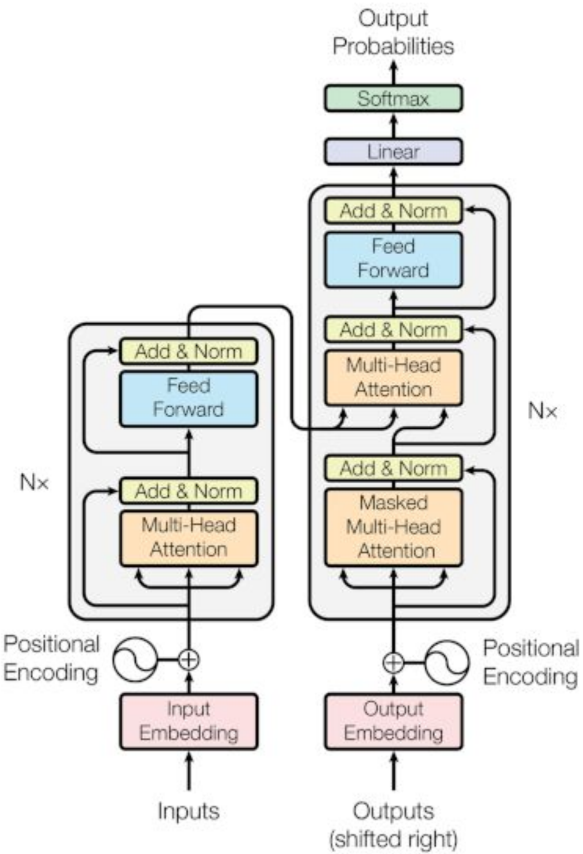
$$\begin{aligned} L_{CE} &= -1\log_2(0.936) + 0 + 0 + 0 \\ &= 0.095 \end{aligned}$$

Illustration: Kiprono Elijan Koech

CBOW, Skip-gram



BERT architecture



BERT pretrain objectives

1. Masked language modeling
2. Next sentence prediction

Masked language modeling

I saw a cute grey [MASK] playing in the garden.

Masked language modeling

I saw a cute grey [MASK] playing in the garden.

[CLS] ,“I”, “saw”, “a”, “cute”, “grey”, [MASK], “playing”, “in”, “the”, “garden”, “.”, [SEP]

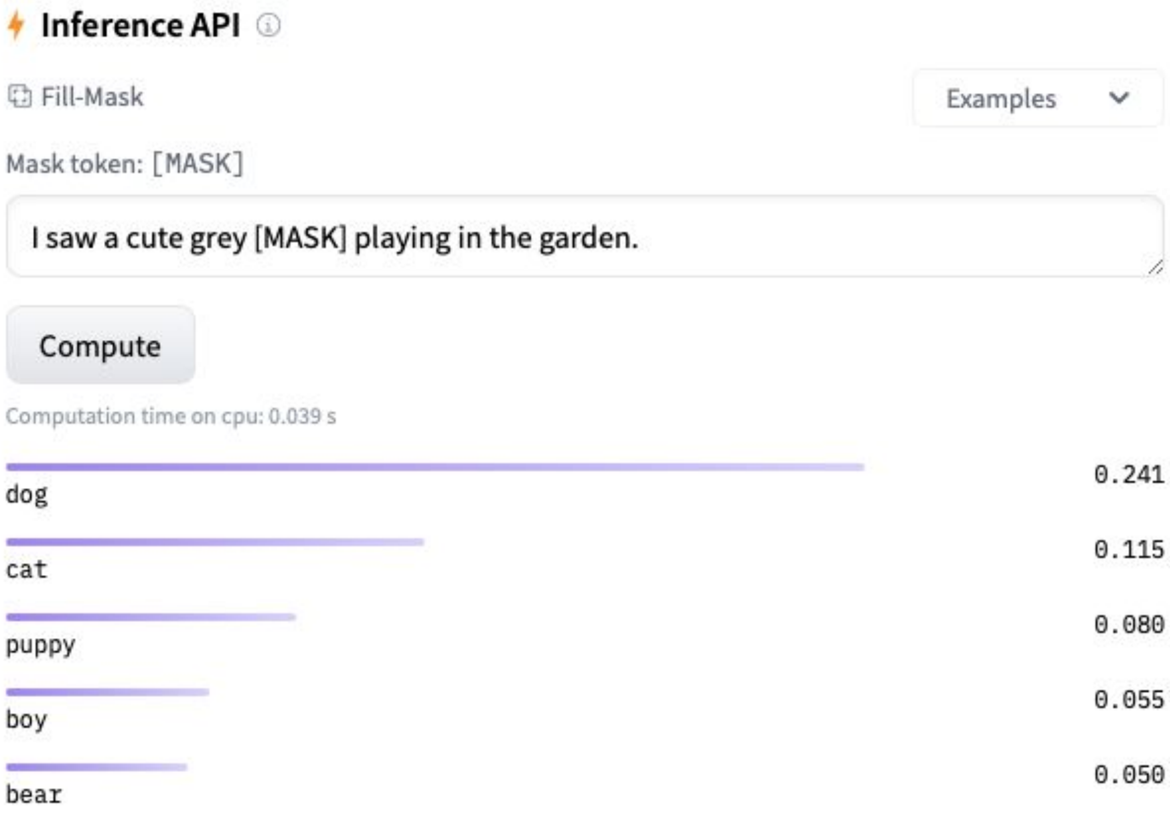
Masked language modeling

I saw a cute grey [MASK] playing in the garden.

[CLS] ,"I", "saw", "a", "cute", "grey", [MASK], "playing", "in", "the", "garden", ".", [SEP]

$$P(w_t \mid w_{1,} w_{2,} w_{3, \ldots.} w_{t\ /\ 1,} w_{t+1, \ldots} w_N)$$

Masked language modeling



Masked language modeling

```
>>> from transformers import pipeline
>>> unmasker = pipeline('fill-mask', model='bert-base-uncased')
>>> unmasker("Hello I'm a [MASK] model.")

[{'sequence': "[CLS] hello i'm a fashion model. [SEP]",
  'score': 0.1073106899857521,
  'token': 4827,
  'token_str': 'fashion'},
 {'sequence': "[CLS] hello i'm a role model. [SEP]",
  'score': 0.08774490654468536,
  'token': 2535,
  'token_str': 'role'},
 {'sequence': "[CLS] hello i'm a new model. [SEP]",
  'score': 0.05338378623127937,
  'token': 2047,
  'token_str': 'new'},
```

Next sentence prediction

Sentence_A

Sentence_B

$P(S_B | S_A)$

Next sentence prediction

Sentence_A: *"How old are you?"*

Sentence_B: *"I am 21 years old"*

$P(S_B | S_A) = 0.999$

Next sentence prediction

Sentence_A: "*How old are you?*"

Sentence_B: *"Queen's University is in Kingston Ontario Canada"*

$P(S_B | S_A) = 0.001$

Hugging Face

- Models
- Datasets
- Applications

PyTorch modules and classes

torch.nn

nn.Module

Dataset

DataLoader

Let’s practice!