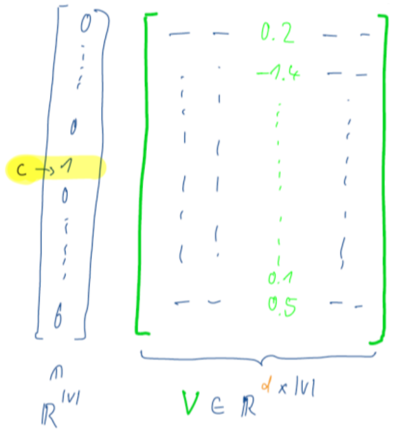
Word embedding

Skip-Gram Model

$$\begin{array}{cc} & \\ & \\ & \\ c \rightarrow & \\ & \\ & \\ & \end{array}\begin{bmatrix}0\\ \vdots\\ 0\\ 1\\ 0\\ \vdots\\ 0\end{bmatrix} \in \mathbb{R}^{|V|}$$

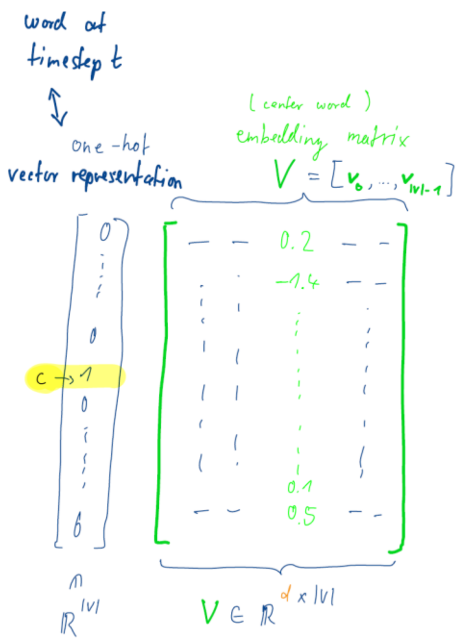
Word embedding

Skip-Gram Model



Word embedding

Skip-Gram Model



Word embedding

Skip-Gram Model

$$w_t \longmapsto V \cdot w_t =: \mathbf{v}_c$$

word at timestep t

one-hot vector representation

(center word) embedding matrix

$$V = [\mathbf{v}_0, \dots, \mathbf{v}_{|V|-1}]$$

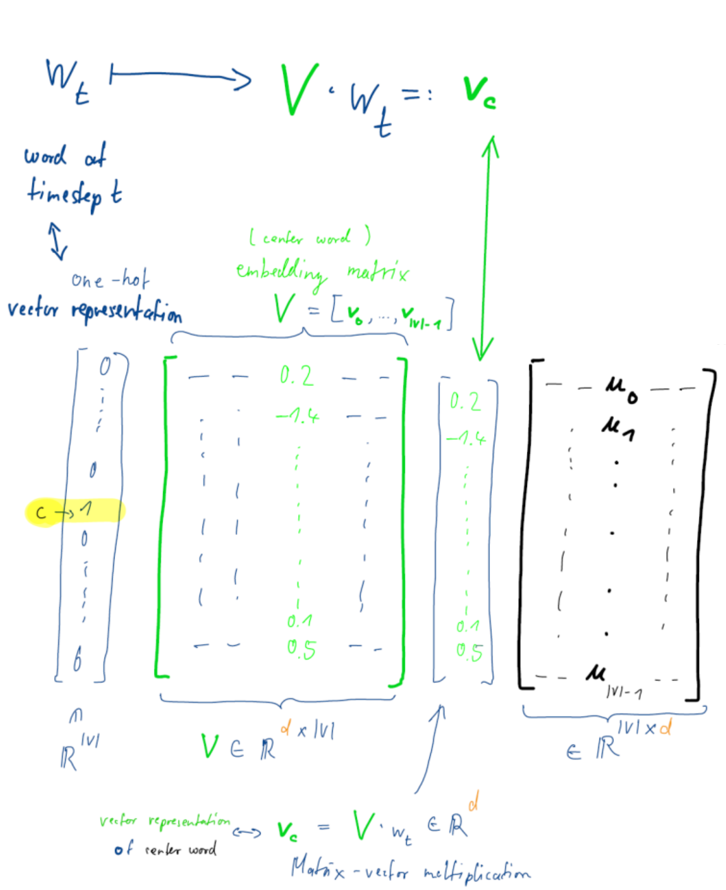
$$\begin{array}{c} \\ \\ \\ c \rightarrow \\ \\ \\ \end{array} \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\in \mathbb{R}^{|V|}} \qquad \underbrace{\begin{bmatrix} - & - & 0.2 & - & - \\ - & - & -1.4 & - & - \\ \vdots & \vdots & \vdots & & \vdots \\ & & 0.1 & & \\ - & - & 0.5 & - & - \end{bmatrix}}_{V \in \mathbb{R}^{d \times |V|}} \qquad \begin{bmatrix} 0.2 \\ -1.4 \\ \vdots \\ 0.1 \\ 0.5 \end{bmatrix}$$

vector representation of center word $\leftrightarrow \mathbf{v}_c = V \cdot w_t \in \mathbb{R}^d$

Matrix-vector multiplicati

Word embedding

Skip-Gram Model



Word embedding

Skip-Gram Model

$$w_t \longmapsto V \cdot w_t =: v_c$$

word at timestep t

$\updownarrow$

one-hot vector representation

(center word) embedding matrix

$$V = [v_0, \dots, v_{|V|-1}]$$

$$\begin{array}{r} \\ \\ \\ c \rightarrow \\ \\ \\ \\ \end{array}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} - & - & 0.2 & - & - \\ - & - & -1.4 & - & - \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ - & - & 0.1 & - & - \\ - & - & 0.5 & - & - \end{bmatrix} \quad \begin{bmatrix} 0.2 \\ -1.4 \\ \vdots \\ 0.1 \\ 0.5 \end{bmatrix} \quad \begin{bmatrix} - & - & u_0 & - & - \\ & & u_1 & & \\ & & \vdots & & \\ - & - & u_{|V|-1} & - & - \end{bmatrix}$$

$\in \mathbb{R}^{|V|}$ $\qquad V \in \mathbb{R}^{d \times |V|}$ $\qquad \in \mathbb{R}^{|V| \times d}$

vector representation of center word $\longleftrightarrow$ $v_c = V \cdot w_t \in \mathbb{R}^d$

Matrix-vector multiplication

$$u_o \in \underbrace{\mathbb{R}^d}_{= \mathbb{R}^{d \times 1}}$$

$$\Rightarrow u_o^T \in \mathbb{R}^{1 \times d}$$

$$\Rightarrow \underbrace{u_o^T \cdot v_c}_{\text{dot product}} \in \mathbb{R}$$

Word embedding

Skip-Gram Model

$$w_t \longmapsto V \cdot w_t =: v_c \longmapsto U v_c =: x$$

word at timestep t

$\updownarrow$

one-hot vector representation

(center word) embedding matrix

$V = [v_0, \dots, v_{|V|-1}]$

context word matrix

$U = [u_0, \dots, u_{|V|-1}]^T$

dot product scores

$$\underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ c \to 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\in \mathbb{R}^{|V|}} \quad \underbrace{\begin{bmatrix} - & - & 0.2 & - & - \\ & & -1.4 & & \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ & & 0.1 & & \\ - & - & 0.5 & - & - \end{bmatrix}}_{V \in \mathbb{R}^{d \times |V|}} \quad \begin{bmatrix} 0.2 \\ -1.4 \\ \vdots \\ 0.1 \\ 0.5 \end{bmatrix} \quad \underbrace{\begin{bmatrix} - & - & u_0 & - & - \\ & & u_1 & & \\ & & \vdots & & \\ - & - & u_{|V|-1} & - & - \end{bmatrix}}_{\in \mathbb{R}^{|V| \times d}}$$

vector representation of center word $\longleftrightarrow$ $v_c = V \cdot w_t \in \mathbb{R}^d$

Matrix-vector multiplication

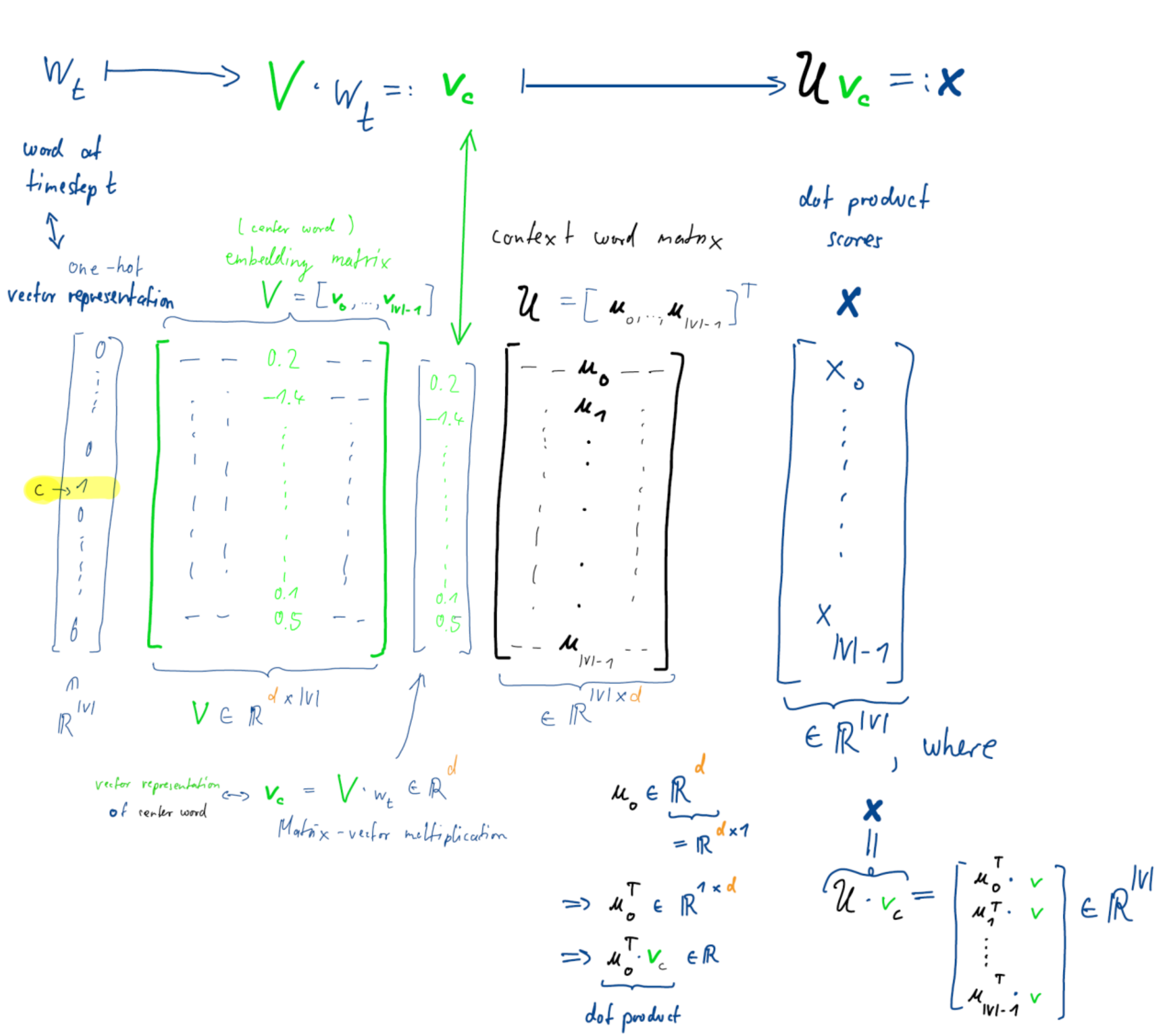
$$u_o \in \underbrace{\mathbb{R}^d}_{= \mathbb{R}^{d \times 1}}$$

$$\Rightarrow u_o^T \in \mathbb{R}^{1 \times d}$$

$$\Rightarrow \underbrace{u_o^T \cdot v_c}_{\text{dot product}} \in \mathbb{R}$$

Word embedding

Skip-Gram Model



Word embedding

Skip-Gram Model

$$w_t \longmapsto V \cdot w_t =: v_c \longmapsto \mathcal{U} v_c =: \mathbf{x} \longmapsto \mathrm{softmax}(\mathbf{x}) =: \hat{y}$$

word at timestep t $\updownarrow$ one-hot vector representation

(center word) embedding matrix $V = [v_0, \dots, v_{|V|-1}]$

context word matrix $\mathcal{U} = [u_0, \dots, u_{|V|-1}]^T$

dot product scores $\mathbf{x}$

probability distribution

$$\underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ c \to 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{\in \mathbb{R}^{|V|}} \quad \underbrace{\begin{bmatrix} - & - & 0.2 & - & - \\ & & -1.4 & & \\ \vdots & \vdots & \vdots & & \vdots \\ & & 0.1 & & \\ - & - & 0.5 & - & - \end{bmatrix}}_{V \in \mathbb{R}^{d \times |V|}} \quad \begin{bmatrix} 0.2 \\ -1.4 \\ \vdots \\ 0.1 \\ 0.5 \end{bmatrix} \quad \underbrace{\begin{bmatrix} - & - & u_0 & - & - \\ & & u_1 & & \\ & & \vdots & & \\ - & - & u_{|V|-1} & - & - \end{bmatrix}}_{\in \mathbb{R}^{|V| \times d}} \quad \underbrace{\begin{bmatrix} x_0 \\ \vdots \\ x_{|V|-1} \end{bmatrix}}_{\in \mathbb{R}^{|V|}} \text{, where} \quad \underbrace{\begin{bmatrix} \mathrm{softmax}(\mathbf{x})_0 \\ \mathrm{softmax}(\mathbf{x})_1 \\ \vdots \\ \boxed{\mathrm{softmax}(\mathbf{x})_k} \\ \vdots \\ \mathrm{softmax}(\mathbf{x})_{|V|-1} \end{bmatrix}}_{\in \mathbb{R}^{|V|}}$$

$\mathrm{softmax}(x)_k \longleftrightarrow$ probability of observing context word with vocabulary index k, given center word with index c

vector representation of center word $\longleftrightarrow$ $v_c = V \cdot w_t \in \mathbb{R}^d$

Matrix-vector multiplication

$$u_0 \in \underbrace{\mathbb{R}^d}_{= \mathbb{R}^{d \times 1}}$$

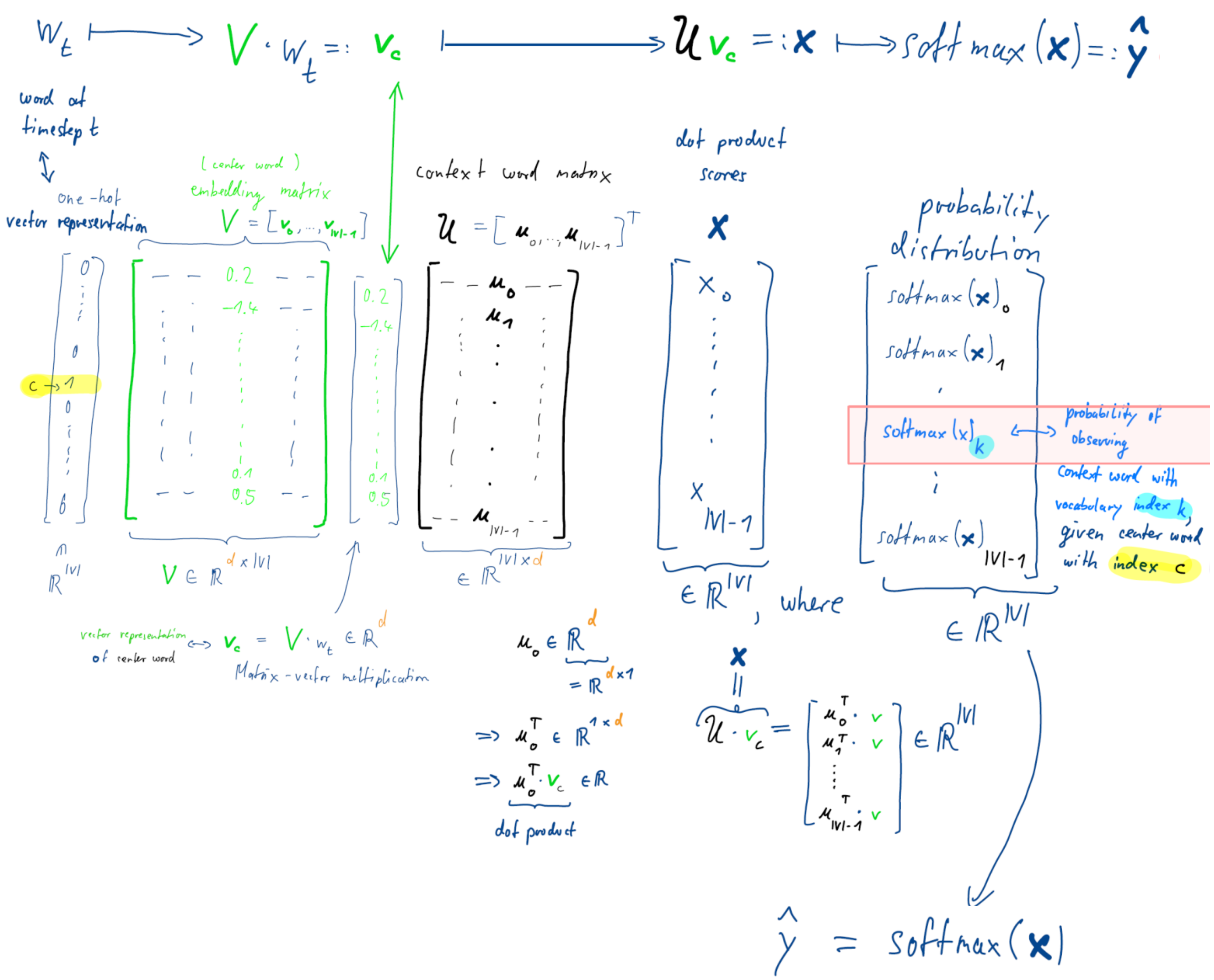
$$\Rightarrow u_0^T \in \mathbb{R}^{1 \times d}$$

$$\Rightarrow \underbrace{u_0^T \cdot v_c}_{\text{dot product}} \in \mathbb{R}$$

$$\mathbf{x} = \mathcal{U} \cdot v_c = \begin{bmatrix} u_0^T \cdot v \\ u_1^T \cdot v \\ \vdots \\ u_{|V|-1}^T \cdot v \end{bmatrix} \in \mathbb{R}^{|V|}$$

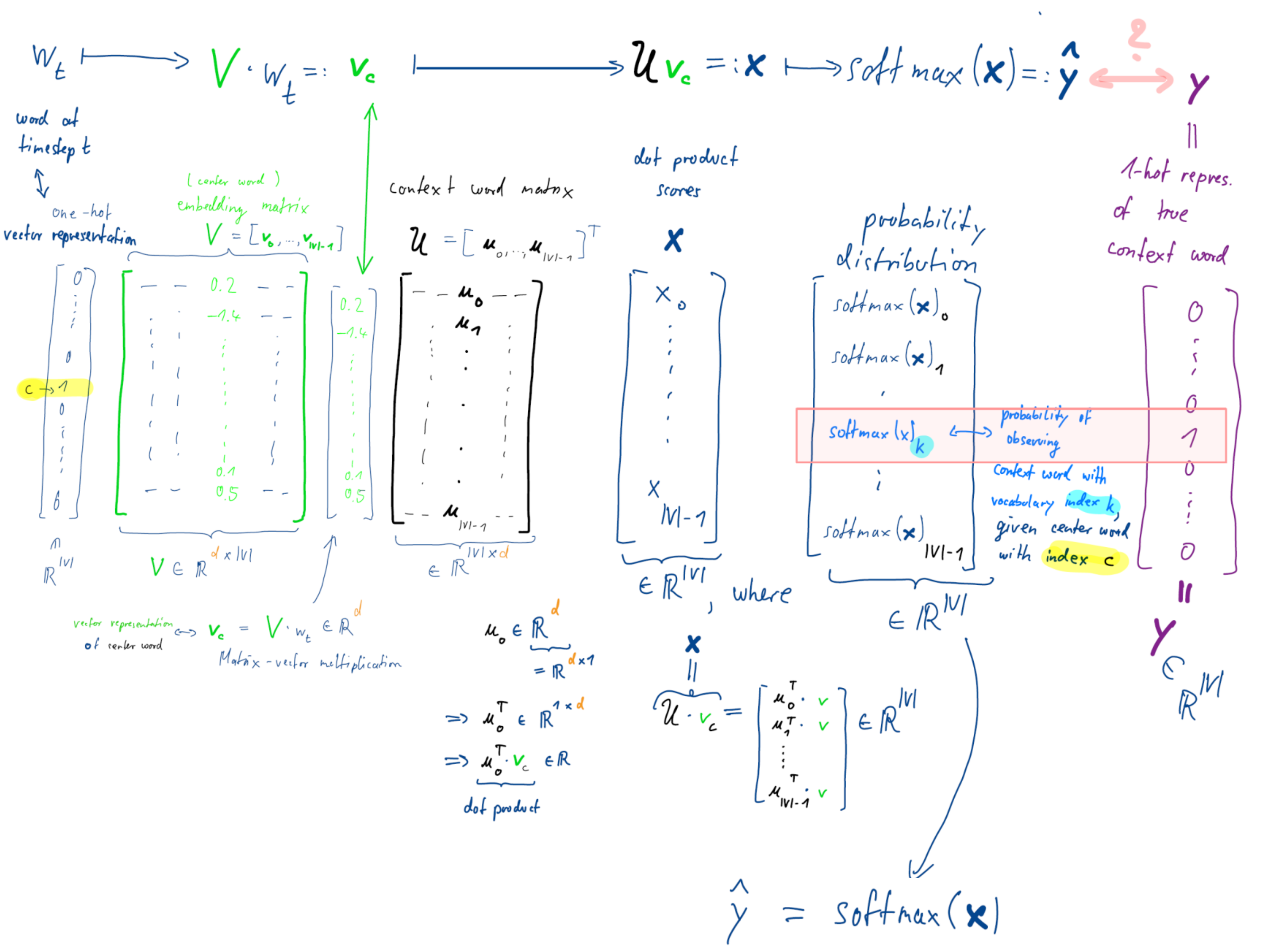
Word embedding

Skip-Gram Model



Word embedding

Skip-Gram Model



Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$w_i$$

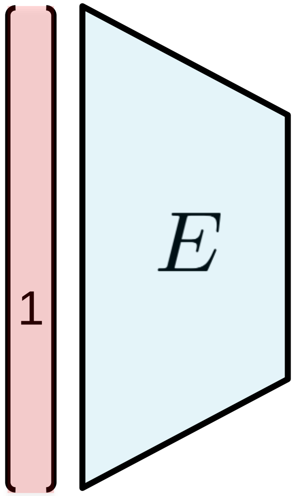
$$\begin{bmatrix} 1 \end{bmatrix}$$

Word Embedding

Transform a Word Vector

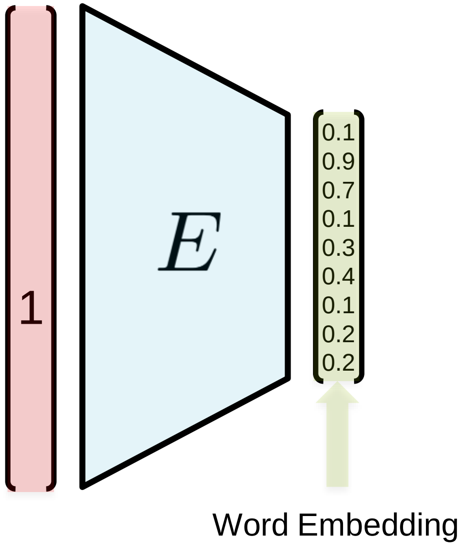
Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$Ew_i$$



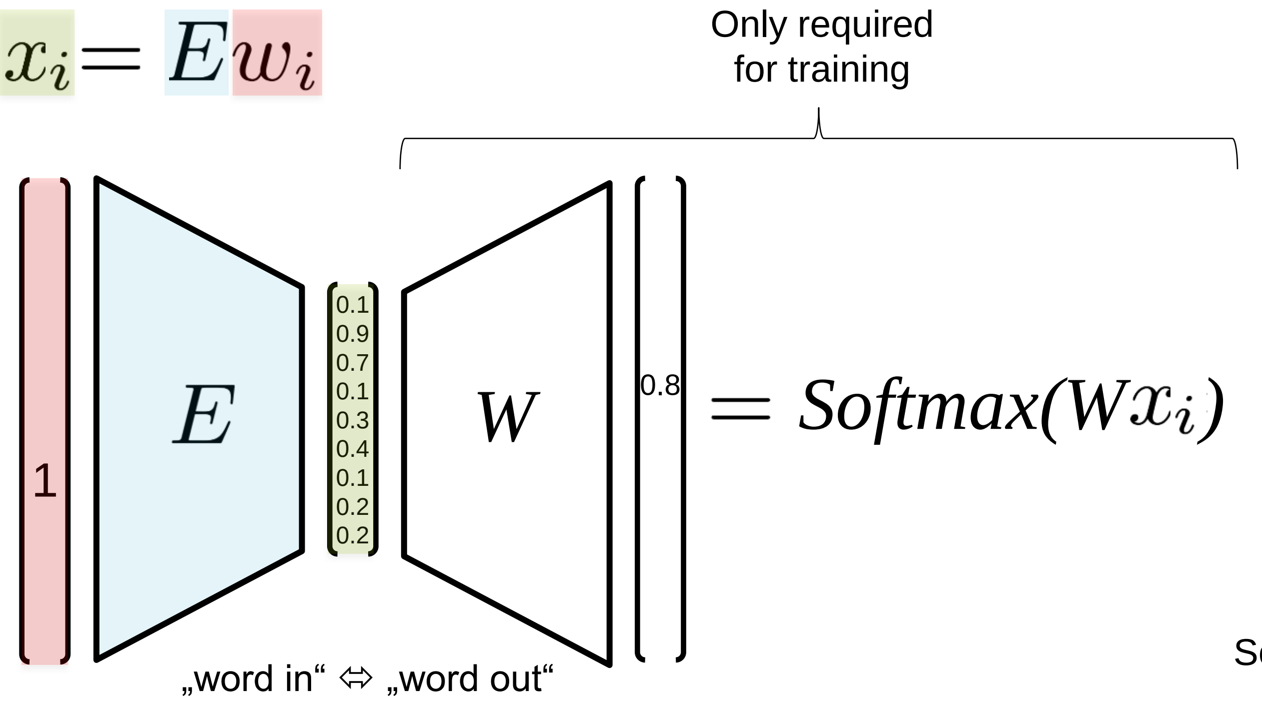
Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

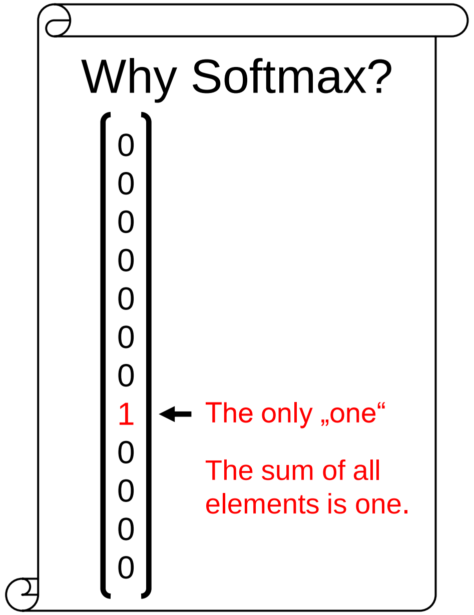
$$x_i = E w_i$$



Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$x_i = E w_i$$





Softmax:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

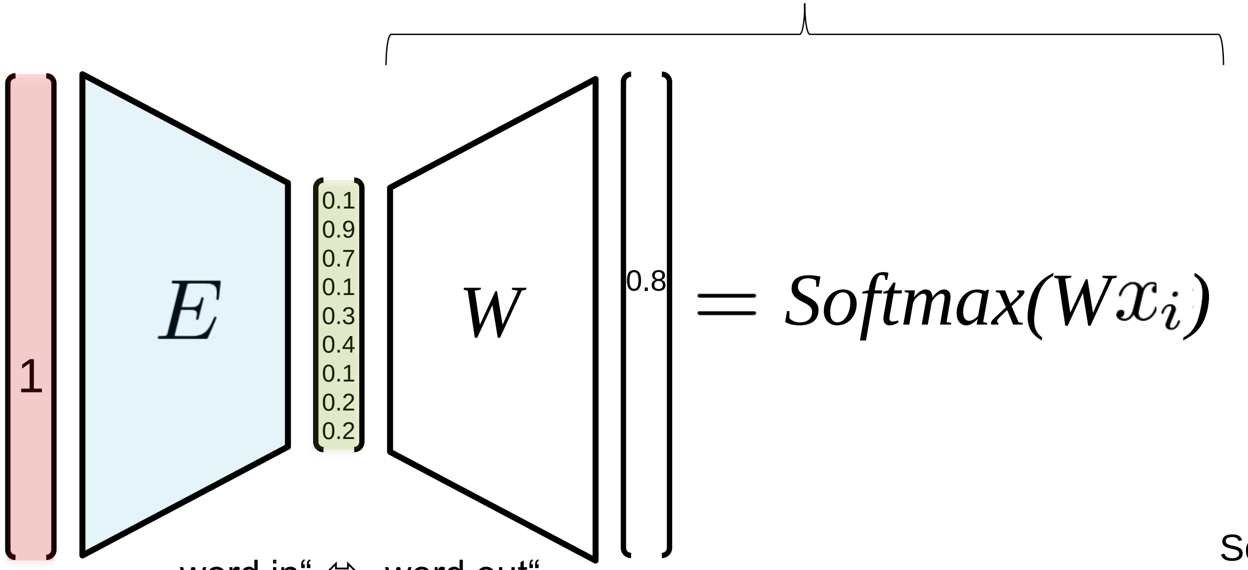
Word Embedding

Transform a Word Vector

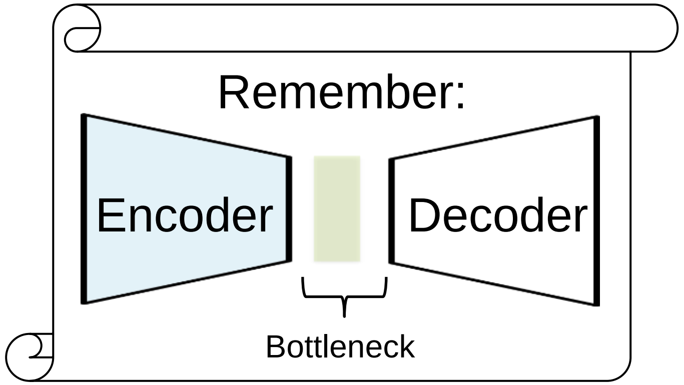
Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$x_i = E w_i$$

Only required for training



„word in“ ⇔ „word out“



Softmax:

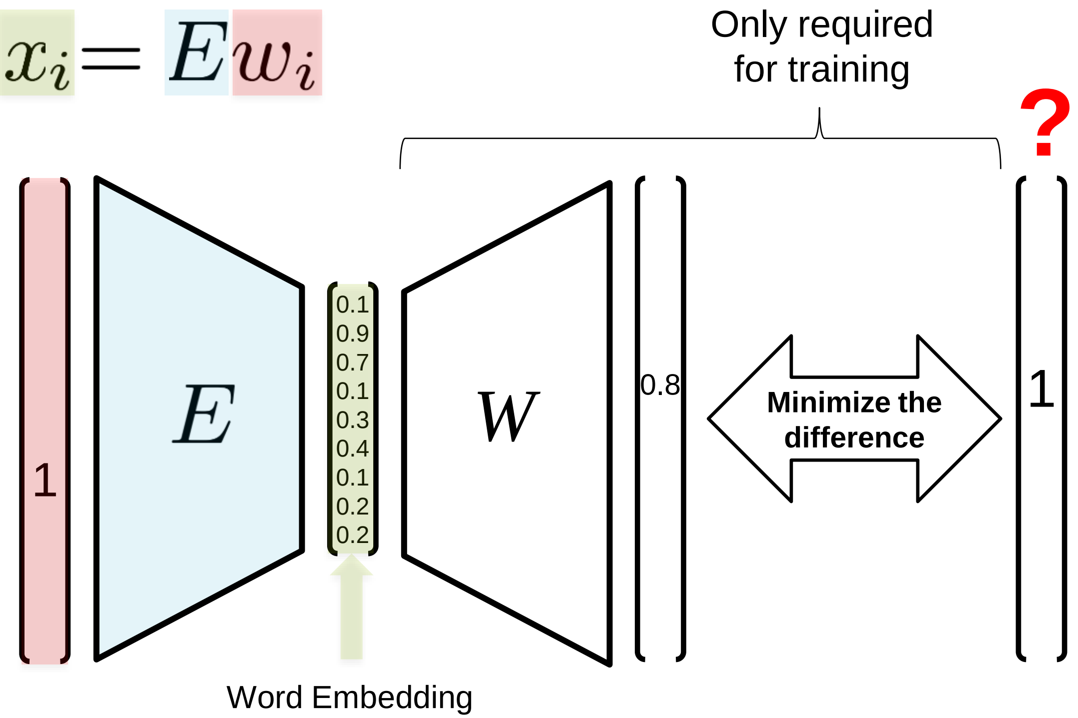
$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

Word Embedding

Transform a Word Vector

Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$x_i = E w_i$$

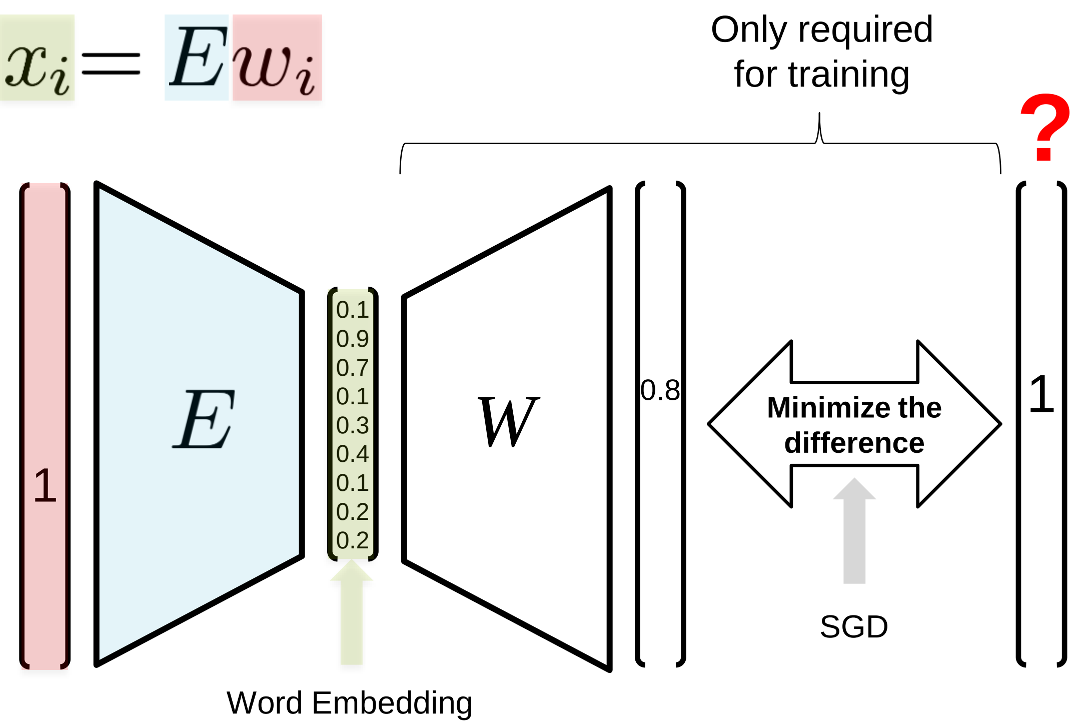


Word Embedding

Transform a Word Vector

Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$x_i = E w_i$$

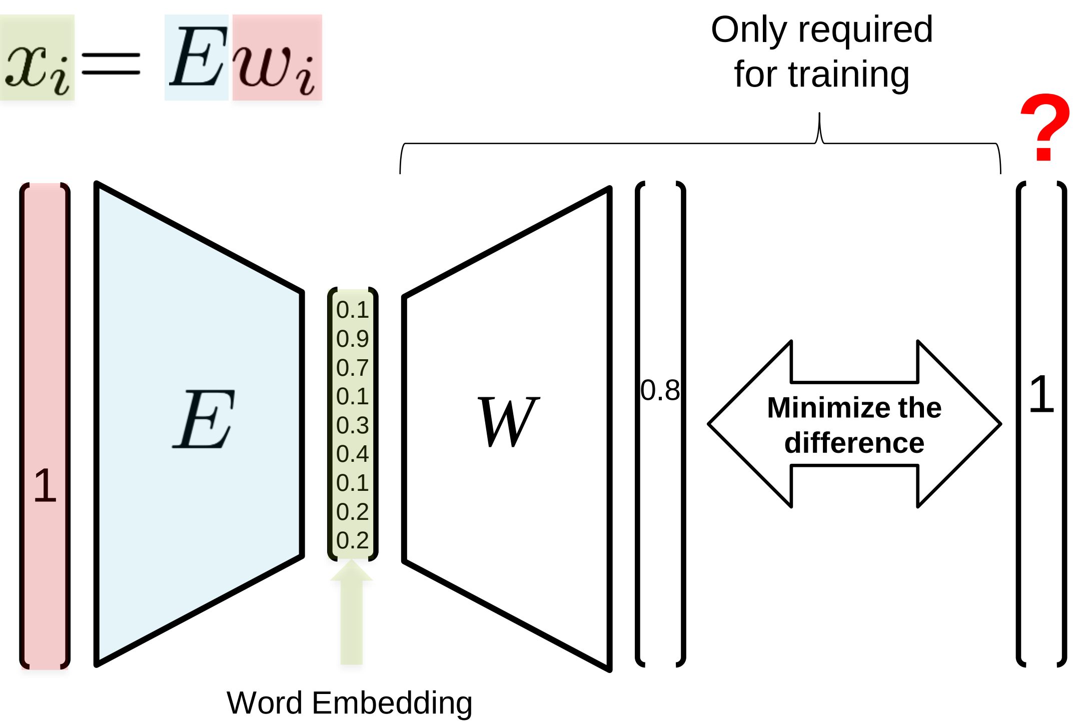


Word Embedding

Transform a Word Vector

Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

$$x_i = E w_i$$



continuous Bag-Of-Words (cBOW)

skip-gram word embedding (word2vec)

Note: in previous notation: E = V , W = U

Word Embedding

Transform a Word Vector

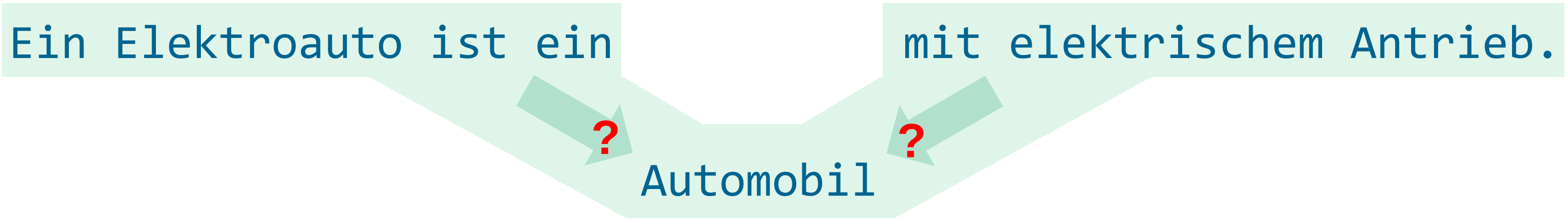
Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

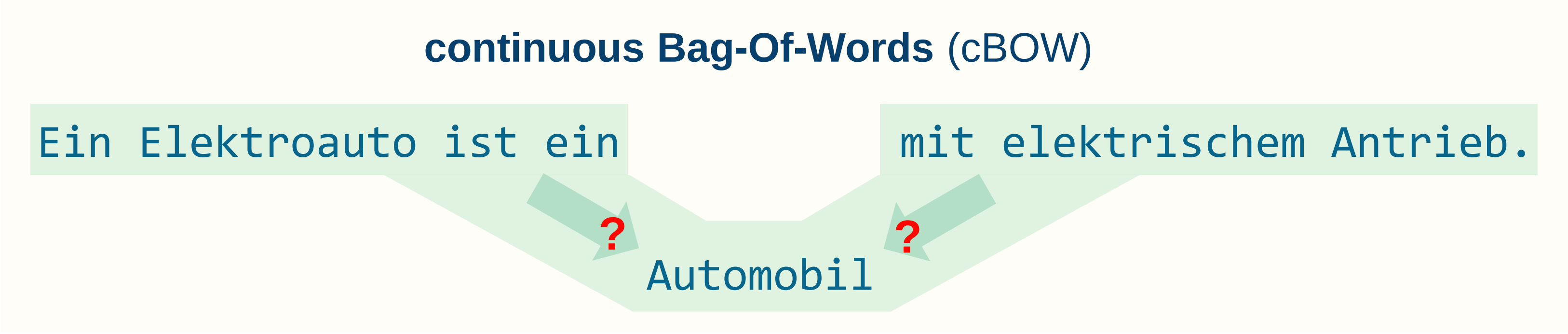
Word Embedding

Transform a Word Vector

Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

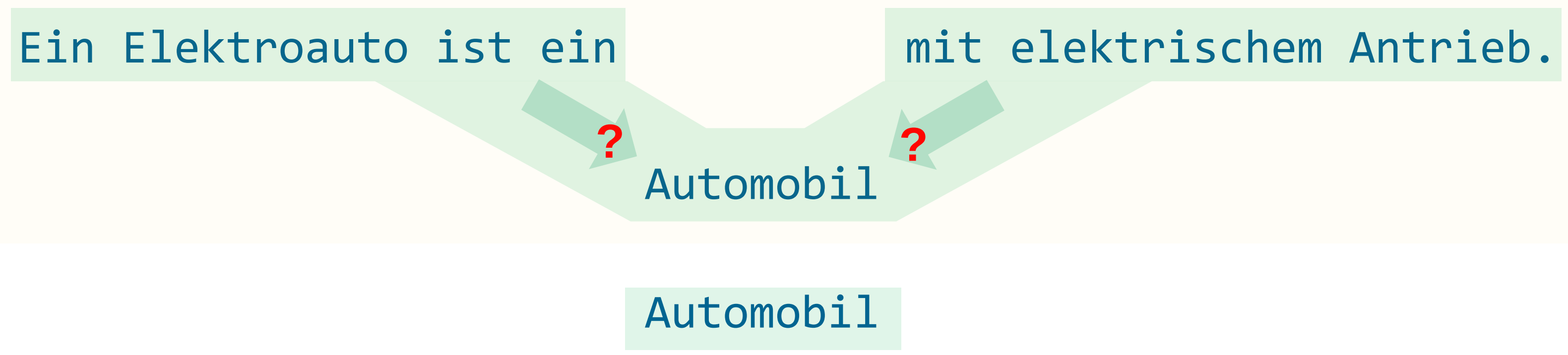


Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

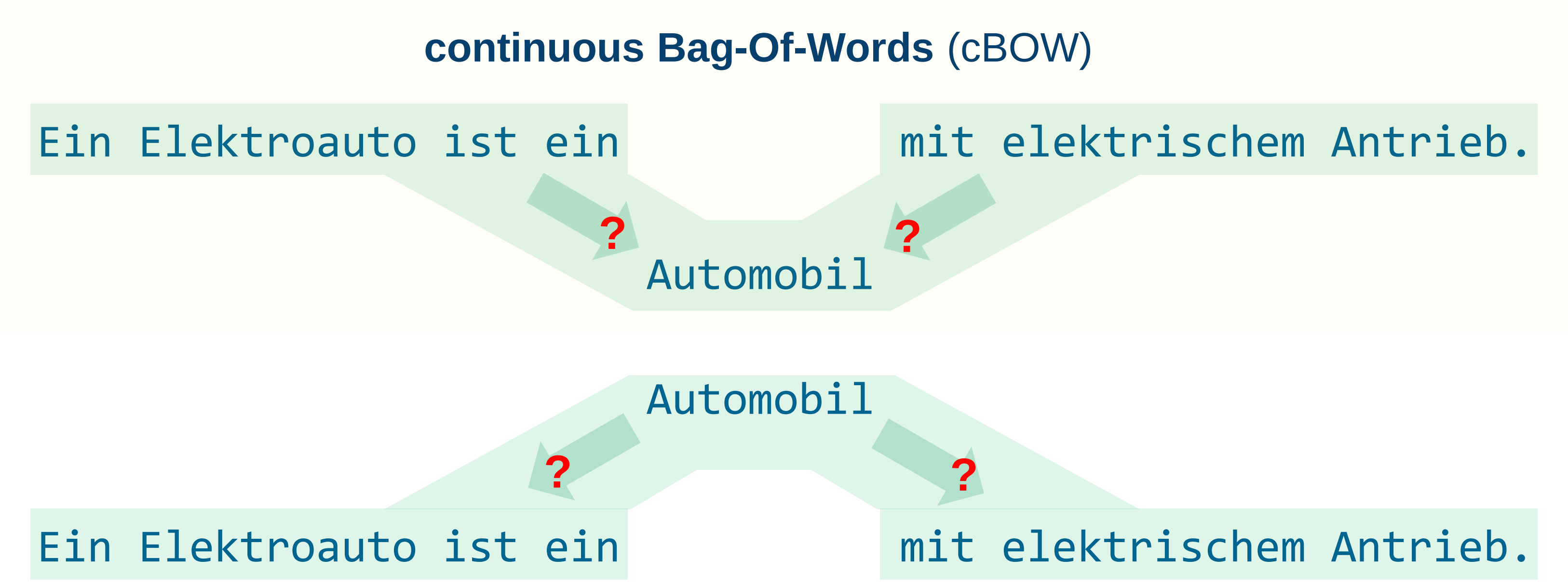


Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.

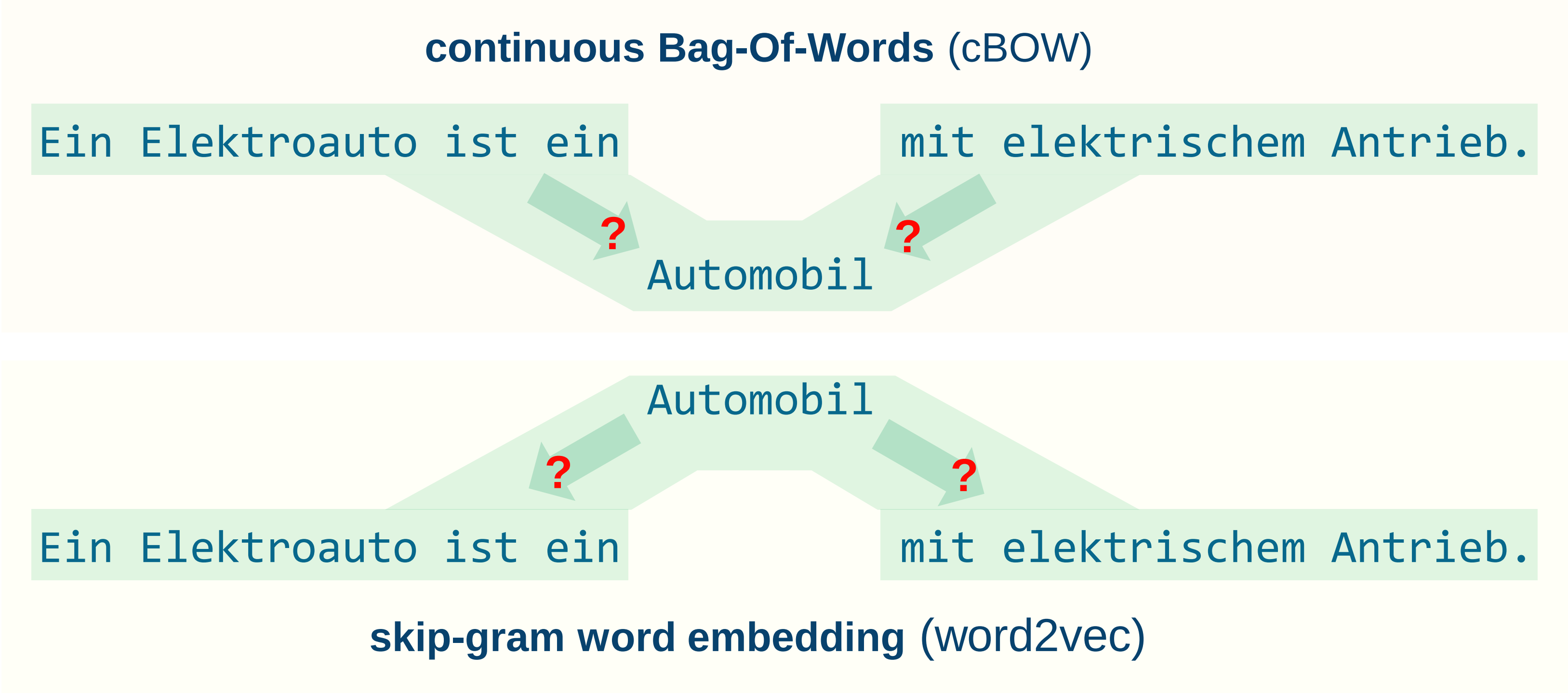
continuous Bag-Of-Words (cBOW)



Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.



Word embedding is any of a set of language modeling and feature learning techniques in natural language processing where words from the vocabulary are mapped to vectors of real numbers.



Pseudo code

Skip-Gram Word Embedding

The input is always one word of the sentence

```
Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.
```

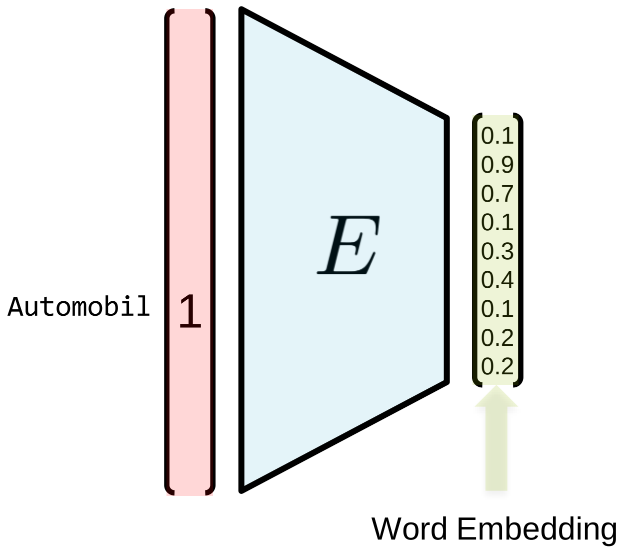
$$
\begin{array}{r}
\\
\texttt{Automobil} \\
\\
\end{array}
\begin{bmatrix} \\ 1 \\ \\ \end{bmatrix}
$$

```
for targetword in sentence:
```

$$x_i = E w_i$$

Compute the word embedding of given word

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`




```
for targetword in sentence:
    embedding = matmul(E,targetword)
```

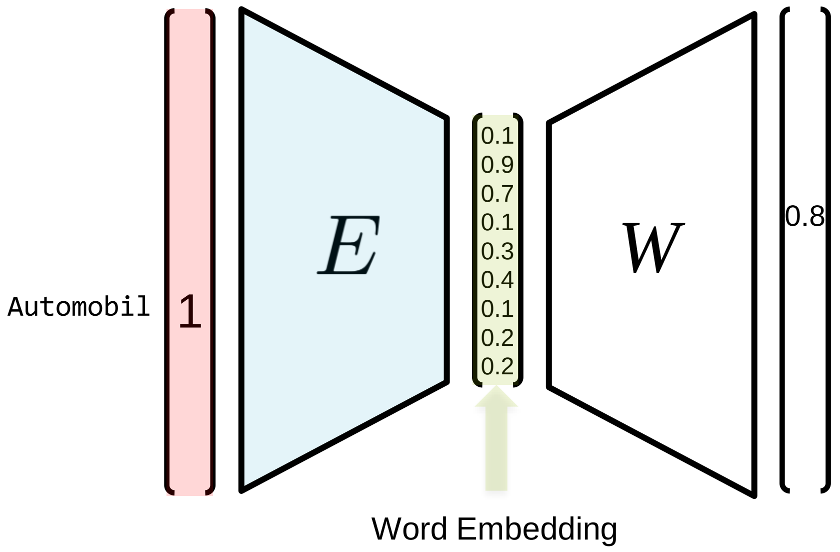
$$x_i = E w_i$$

Pseudo code

Skip-Gram Word Embedding

Make a prediction using the word embedding aka “use the word embedding”

```
Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.
```



$$= Softmax(Wx_i)$$

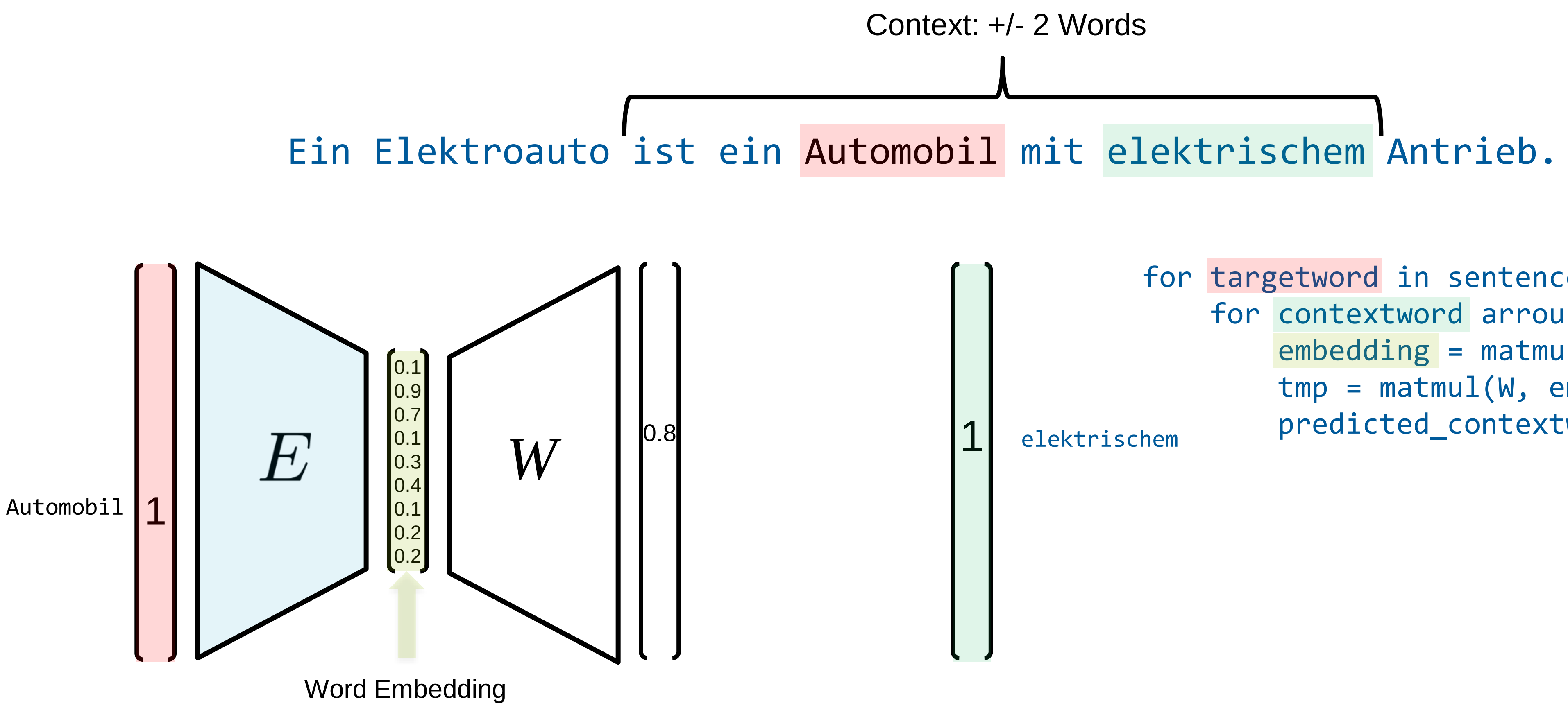
```
for targetword in sentence:

    embedding = matmul(E,targetword)
    tmp = matmul(W, embedding)
    predicted_contextword = softmax(tmp)
```

$$x_i = E w_i$$

Pseudo code

Skip-Gram Word Embedding

One possible prediction: Predict the context of the word




```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
```

$$x_i = E w_i$$

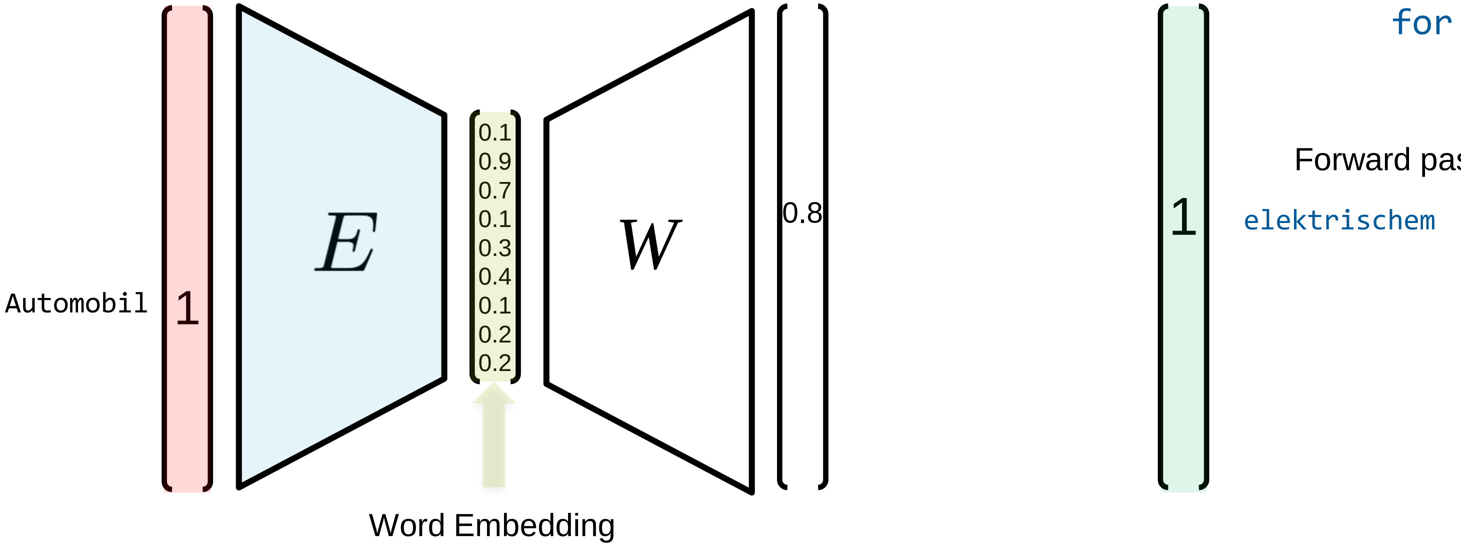
Pseudo code

Skip-Gram Word Embedding

One possible prediction: Predict the context of the word

Context: +/- 2 Words

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



Forward pass

```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
```

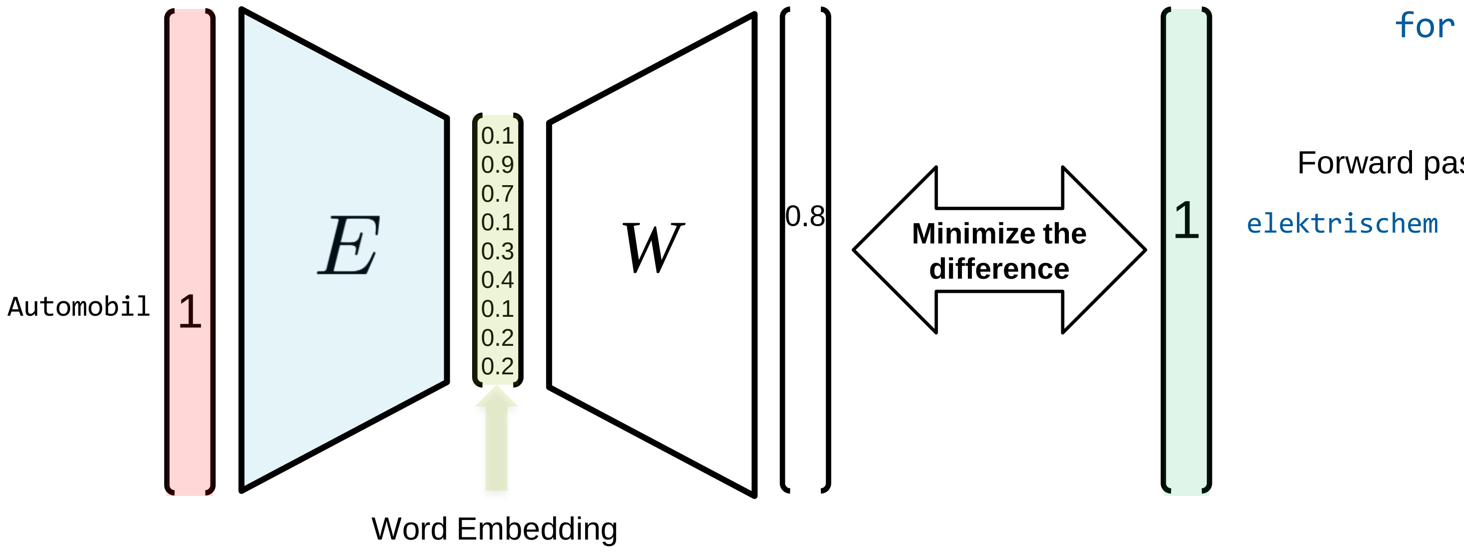
$$x_i = E w_i$$

Pseudo code

Skip-Gram Word Embedding

Minimize the error between prediction of the context and the real context by updating the network

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



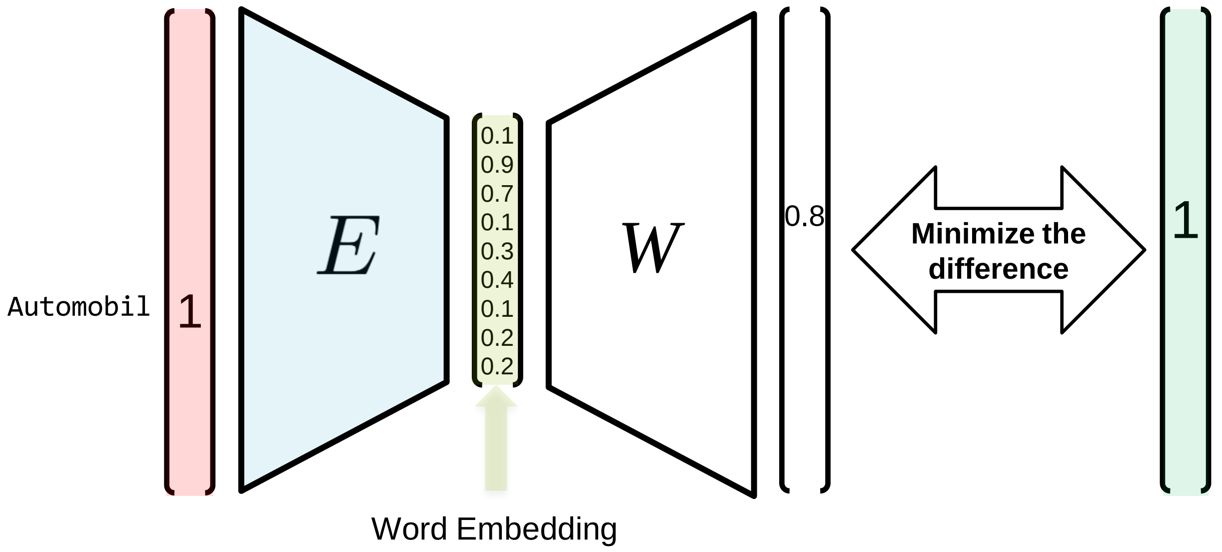
Forward pass

```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
        minimize(predicted_contextword, contextword)
```

$$x_i = E w_i$$

Minimize the error between prediction of the context and the real context by updating the network

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`




```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
        minimize(predicted_contextword, contextword)
```

Forward pass: `embedding = matmul(E,targetword)`, `tmp = matmul(W, embedding)`, `predicted_contextword = softmax(tmp)`

Backward pass Updates: `minimize(predicted_contextword, contextword)`

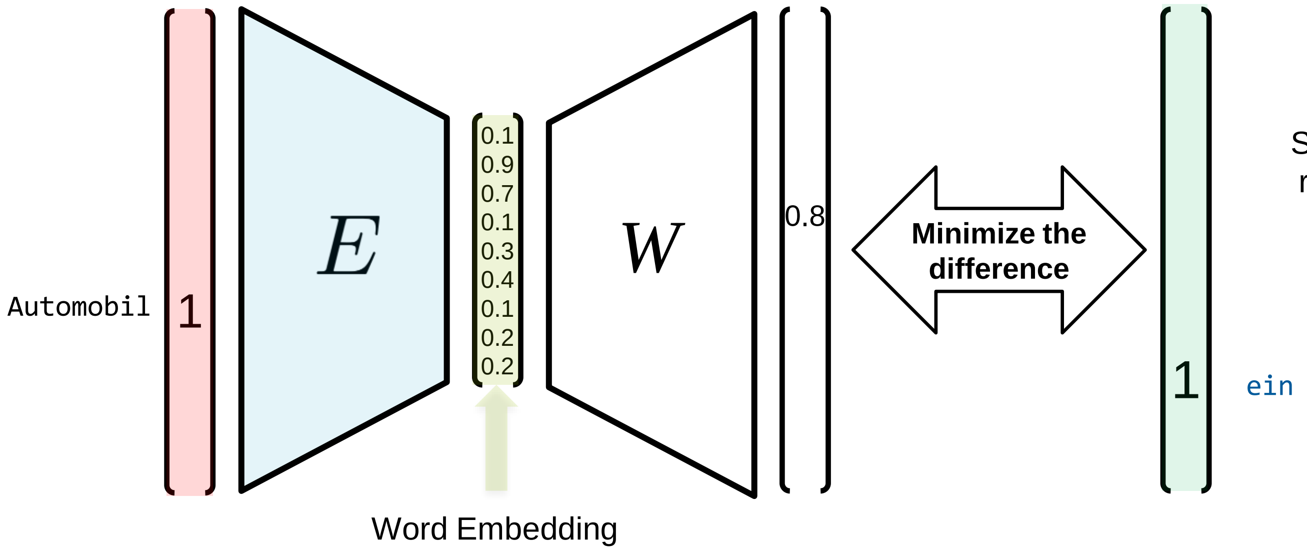
$$x_i = Ew_i$$

Pseudo code

Skip-Gram Word Embedding

Considering the complete context of a word is almost impossible: sample.

Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.



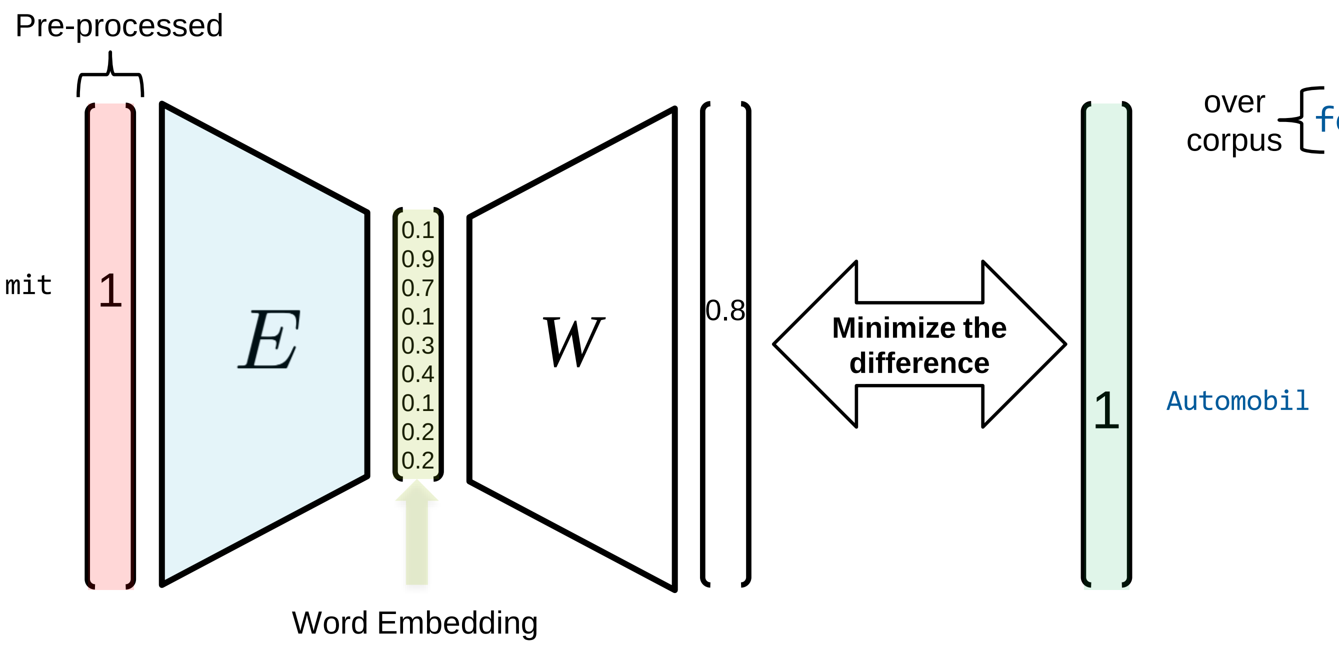
Sampling required

```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
        minimize(predicted_contextword, contextword)
```

$$x_i = E w_i$$

Repeat the process for any word in the corpus

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



over corpus

```
for targetword in sentence:
    for contextword arround targetword:
        embedding = matmul(E,targetword)
        tmp = matmul(W, embedding)
        predicted_contextword = softmax(tmp)
        minimize(predicted_contextword, contextword)
```

Note: in previous notation: E = V , W = U

$$x_i = E w_i$$

Skip-gram Loss

Recap

We want to minimize

$$-\sum_{t=1}^{T} \sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(\boldsymbol{t}+\boldsymbol{j})} \middle| \boldsymbol{w}^{(\boldsymbol{t})}\right)$$

Skip-gram Loss

Recap

Consider only inner sum

$$-\sum_{t=1}^{T} \sum_{j=-m, j \neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

Skip-gram Loss

Recap

But we keep the minus sign

$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

Skip-gram Loss

Recap

Question

How did we model the probability? $-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$

Skip-gram Loss

Recap

… Using softmax function

$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^{T} \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^{T}\, v_c)}$$

Skip-gram Loss

Recap

After applying logarithm rule

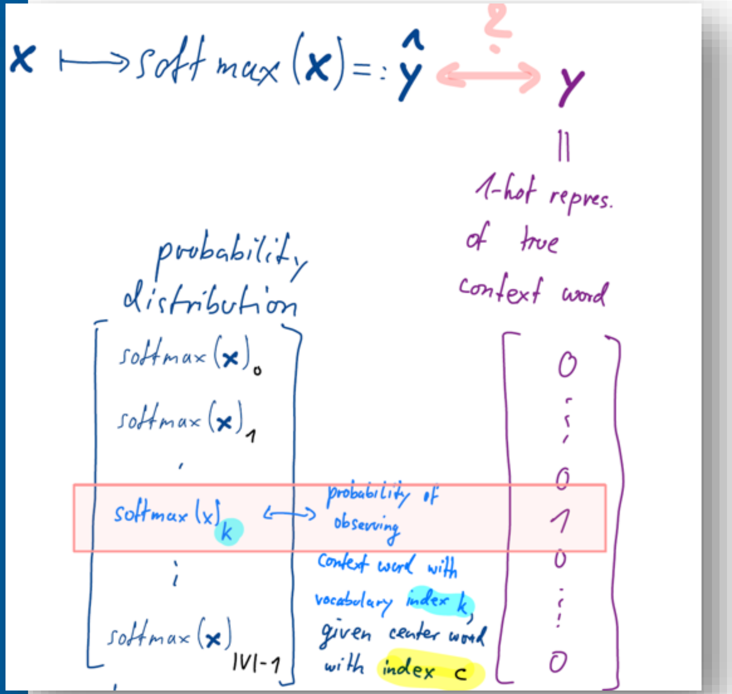
$$\log\left(\frac{a}{b}\right) = \log(a) - \log(b)$$

$$- \sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= - \sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^{T} \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= - \sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^{T} \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

Skip-gram Loss



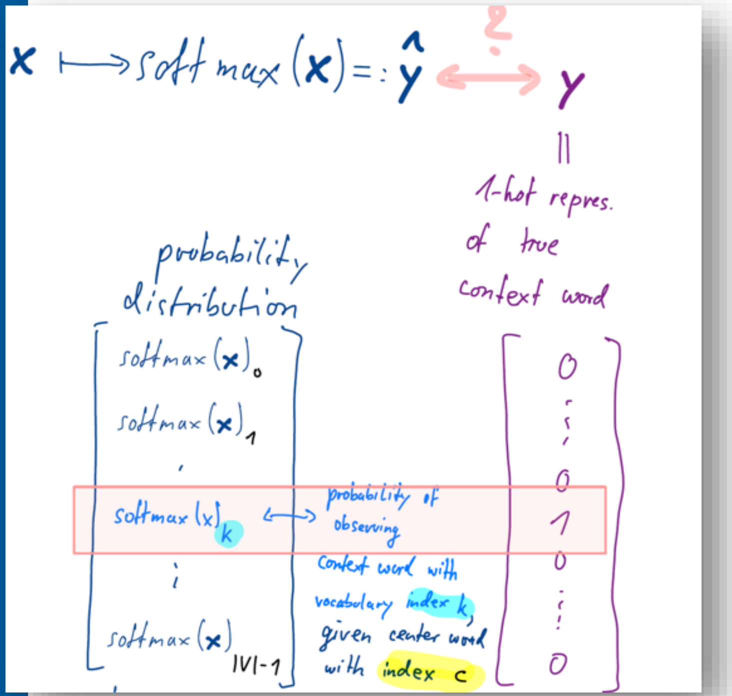
$$-\sum_{j=-m,j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0,j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0,j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

Skip-gram Loss

$$\hat{y}_k = softmax(u_k^T \cdot v_c)$$



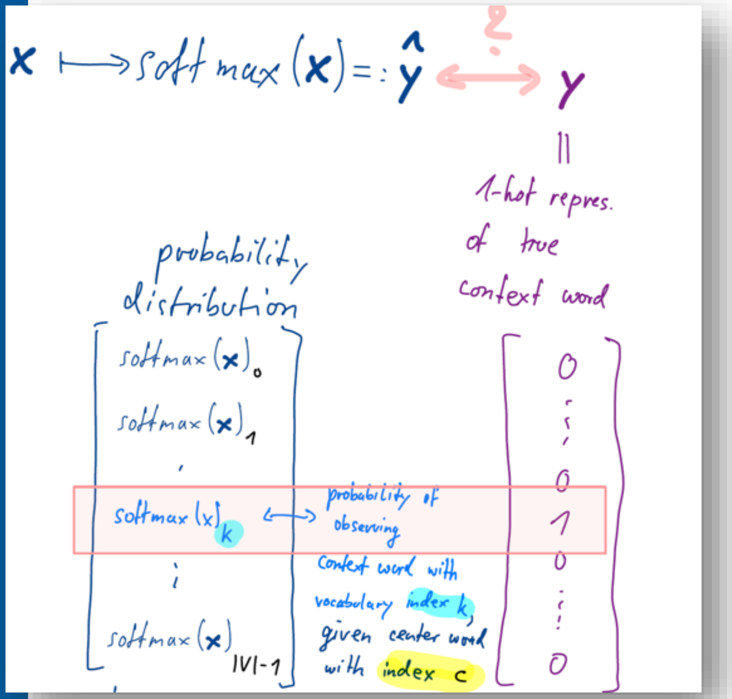
$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

Skip-gram Loss

$$\hat{\boldsymbol{y}}_k = softmax(u_k^T \cdot v_c)$$



$$-\sum_{j=-m,j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0,j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0,j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

Skip-gram Loss

And **Cross-entropy**

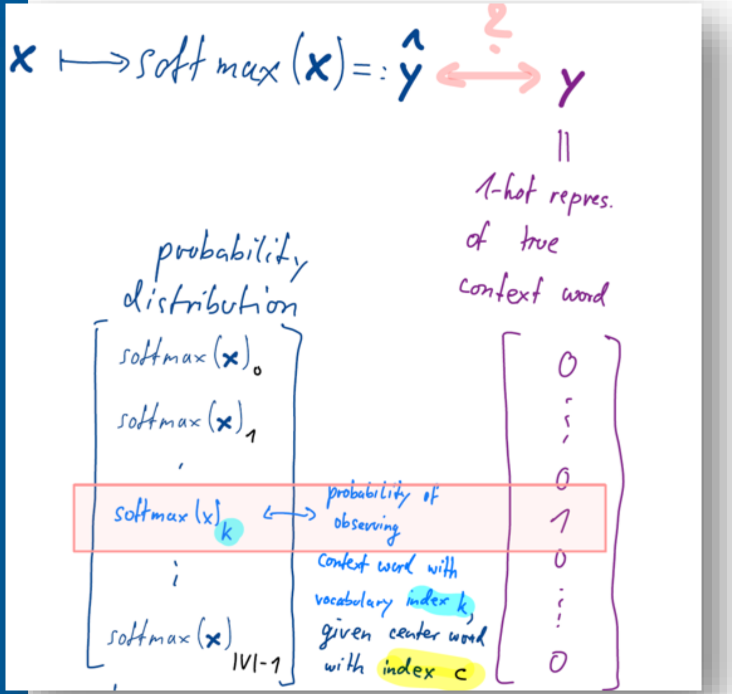
$$\hat{\boldsymbol{y}}_k = softmax(u_k^T \cdot v_c)$$

$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

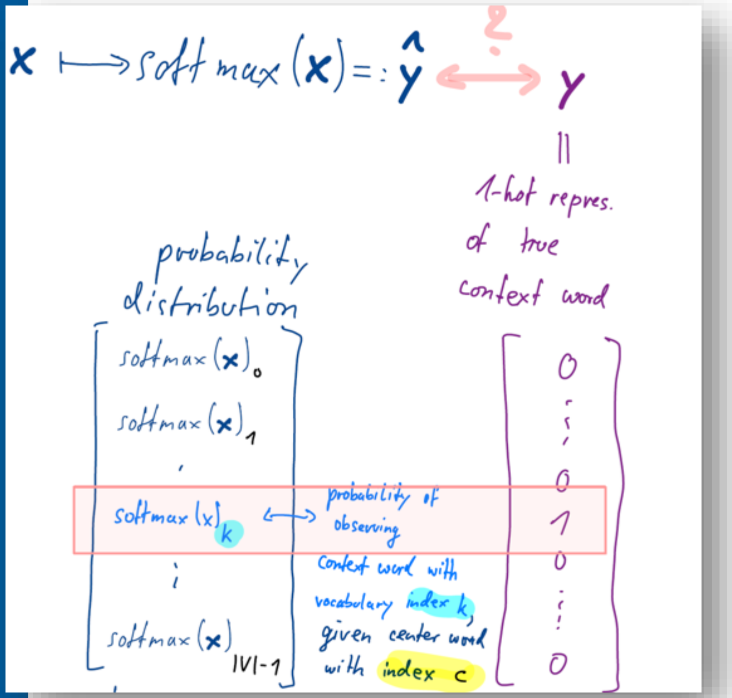
$$\mathrm{H}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{k=0}^{|V|-1} \boldsymbol{y}_k \log(\hat{\boldsymbol{y}}_k)$$



Skip-gram Loss

*Minimize **Cross-entropy** loss*

$$\hat{y}_k = softmax(u_k^T \cdot v_c)$$



$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

$$\mathrm{H}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum_{k=0}^{|V|-1} \boldsymbol{y}_k \log(\hat{\boldsymbol{y}}_k)$$

Goal

For each center word $\boldsymbol{w}^{(t)}$(with associated vector v_c)
we desire the estimated probability vector $\hat{\boldsymbol{y}}$
to be as close as possible to the *true* (1-hot encoded) context word vectors $\boldsymbol{y}^{(c-m)}, \ldots, \boldsymbol{y}^{(c+m)}$

Skip-gram Loss

Recap

Back to the loss…

How can we split the sums?

(Watch the variables in second sum)

$$-\sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^{T} \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^{T} \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

Skip-gram Loss

Recap

What do you notice?

$$-\sum_{j=-m,j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0,j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^{T} \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0,j\neq m}^{2m} \left(u_{c-m+j}^{T} \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

$$= -\sum_{j=0,j\neq m}^{2m} u_{c-m+j}^{T} \cdot v_c + 2m \log \sum_{k=1}^{|V|} \exp(u_k^T v_c)$$

Skip-gram Loss

Recap

$$- \sum_{j=-m, j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= - \sum_{j=0, j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^T \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= - \sum_{j=0, j\neq m}^{2m} \left(u_{c-m+j}^T \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

$$= - \sum_{j=0, j\neq m}^{2m} u_{c-m+j}^T \cdot v_c + 2m \log \boxed{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

Expensive Summation over entire vocabulary V

Skip-gram Loss

Recap

$$-\sum_{j=-m,j\neq 0}^{m} \log \mathrm{P}\left(\boldsymbol{w}^{(t+j)} \middle| \boldsymbol{w}^{(t)}\right)$$

$$= -\sum_{j=0,j\neq m}^{2m} \log \frac{\exp(u_{c-m+j}^{T} \cdot v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

$$= -\sum_{j=0,j\neq m}^{2m} \left(u_{c-m+j}^{T} \cdot v_c - \log \sum_{k=1}^{|V|} \exp(u_k^T v_c) \right)$$

$$= -\sum_{j=0,j\neq m}^{2m} u_{c-m+j}^{T} \cdot v_c + 2m \log \boxed{\sum_{k=1}^{|V|} \exp(u_k^T v_c)}$$

Expensive Summation over entire vocabulary V

Motivates „negative sampling“

word2vec

Negative Sampling Overview

1. Treat the target word t and a neighboring context word c as positive examples.
2. Randomly sample other words in the lexicon to get negative examples
3. Use logistic regression to train a classifier to distinguish those two cases
4. Use the learned weights as the embeddings

Negative Sampling

Word2vec, Skip-gram Classifier

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

Goal:

train a classifier that

- is given a candidate (**w**ord, **c**ontext) pair

  (apricot, jam), (apricot, aardvark), …

- And assigns each word a probability

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

Goal:

train a classifier that

- is given a candidate (**w**ord, **c**ontext) pair

(apricot, jam), (apricot, aardvark), …

- And assigns each word a probability

$$P(+|w,c)$$

$$P(-|w,c) = 1 - P(+|w,c)$$

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

$$P(-|w,c) = 1 - P(+|w,c) = \sigma(-c \cdot w)$$

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

target	context
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

For each positive example:

we'll grab k negative examples,

sampling by frequency

Negative Sampling

Word2vec, Skip-gram Classifier

…lemon, a [tablespoon of apricot jam, a] pinch…

c1 c2 w c3 c4

Positive examples (w, c_{pos})

target	context
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

Negative examples (w, c_{neg})

target	context
apricot	aardvark
apricot	my
apricot	where
apricot	coaxial
apricot	seven
apricot	forever
apricot	dear
apricot	if

Given the set of positive and negative training instances, and an initial set of embedding vectors,

Goal: adjust word vectors such that we:

Given the set of positive and negative training instances, and an initial set of embedding vectors,

Goal: adjust word vectors such that we:

Maximize the similarity of the word pairs (w, c_{pos}) drawn from the positive data

Given the set of positive and negative training instances, and an initial set of embedding vectors,

Goal: adjust word vectors such that we:

Maximize the similarity of the word pairs (w, c_{pos}) drawn from the positive data

Minimize the similarity of the (w, c_{neg}) pairs drawn from the negative data.

$$Sim(w, c) \approx w \cdot c$$

$$P(+|w, c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

$$P(-|w, c) = 1 - P(+|w, c) = \sigma(-c \cdot w)$$

How is it computed together?

Skip-gram classifier

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

For a window of size L:

How is it computed together?

Skip-gram classifier

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

For a window of size L:

$$P(+|w, c_{1:L}) = \prod_{i=1}^{L} \sigma(c_i \cdot w)$$

How is it computed together?

Skip-gram classifier

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1+\exp(-c \cdot w)}$$

For a window of size L:

$$\log P(+|w, c_{1:L}) = \log \prod_{i=1}^{L} \sigma(c_i \cdot w)$$

How is it computed together?

Skip-gram classifier

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

For a window of size L:

$$\log P(+|w, c_{1:L}) = \sum_{i=1}^{L} \log \sigma(c_i \cdot w)$$

Probability of sampling 1 positive sample and k negative samples:

How is it computed together?

Skip-gram classifier

$$P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1 + \exp(-c \cdot w)}$$

For a window of size L:

$$\log P(+|w, c_{1:L}) = \sum_{i=1}^{L} \log \sigma(c_i \cdot w)$$

Probability of sampling 1 positive sample and k negative samples:

$$P(+|w, c_{pos}) \cdot P(-|w, c_{1:k}^{neg})$$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

$$P(+|w, c_{pos}) \cdot P(-|w, c_{1:k}^{neg})$$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

$$P\left(+\middle|w, c_{pos}\right) \cdot P\left(-\middle|w, c_{1:k}^{neg}\right)$$

“maximize the probability of a word and context not being in the corpus data if it indeed is not”

$$P\left(-\middle|w, c_{1:k}^{neg}\right) = 1 - P\left(+\middle|w, c_{1:k}^{neg}\right) = \sigma(-c_{neg} \cdot w)$$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize $$-\log\left[P\left(+|w, c_{pos}\right) \cdot P\left(-|w, c_{1:k}^{neg}\right)\right]$$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize $$-\log\left[P(+|w, c_{pos})\prod_{i=1}^{k}P(-|w, c_{neg_i})\right]$$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize

$$-\log\left[P(+|w, c_{pos}) \prod_{i=1}^{k} P(-|w, c_{neg_i}) \right]$$

Remember: $P(+|w,c) = \sigma(c \cdot w) = \frac{1}{1+\exp(-c \cdot w)}$

How is it computed together?

Loss function

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize

$$-\log \sigma(c_{pos} \cdot w) - \sum_{i=1}^{k} \log \sigma(-c_{neg_i} \cdot w)$$

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize
$$-\log \sigma(c_{pos} \cdot w) - \sum_{i=1}^{k} \log \sigma(-c_{neg_i} \cdot w)$$

Or in previous notation:
$$-\log \sigma(u_{c-m+j}^T \cdot v_c) - \sum_{k=1}^{K} \log \sigma(-\tilde{u}_k^T \cdot v_c)$$

Maximize the similarity of the target with the actual context words, and minimize the similarity of the target with the k negative sampled non-neighbor words.

minimize

$$-\log \sigma(c_{pos} \cdot w) - \sum_{i=1}^{k} \log \sigma(-c_{neg_i} \cdot w)$$

Or in previous notation:

$$-\log \sigma(u_{c-m+j}^T \cdot v_c) - \sum_{k=1}^{K} \log \sigma(-\tilde{u}_k^T \cdot v_c)$$

$\{\tilde{u}_k | k = 1 \ldots K\}$ are sampled from weighted unigram distribution

Sampling from which distribution?

Motivation for weighted unigram distribution

$$\text{is: } 0.9^{3/4} = 0.92$$

$$\text{Constitution: } 0.09^{3/4} = 0.16$$

$$\text{bombastic: } 0.01^{3/4} = 0.032$$

Sampling from which distribution?

Weighted unigram distribution

Unigram probabilities

$$\text{is: } \boxed{0.9}^{3/4} = 0.92$$

$$\text{Constitution: } \boxed{0.09}^{3/4} = 0.16$$

$$\text{bombastic: } \boxed{0.01}^{3/4} = 0.032$$

Preparation for SGD

Derivatives of Loss function

$$\frac{\partial L_{CE}}{\partial c_{pos}} = [\sigma(c_{pos} \cdot w) - 1]w$$

$$\frac{\partial L_{CE}}{\partial c_{neg}} = [\sigma(c_{neg} \cdot w)]w$$

$$\frac{\partial L_{CE}}{\partial w} = [\sigma(c_{pos} \cdot w) - 1]c_{pos} + \sum_{i=1}^{k} [\sigma(c_{\mathrm{neg}_i} \cdot w)]c_{neg_i}$$

SGD

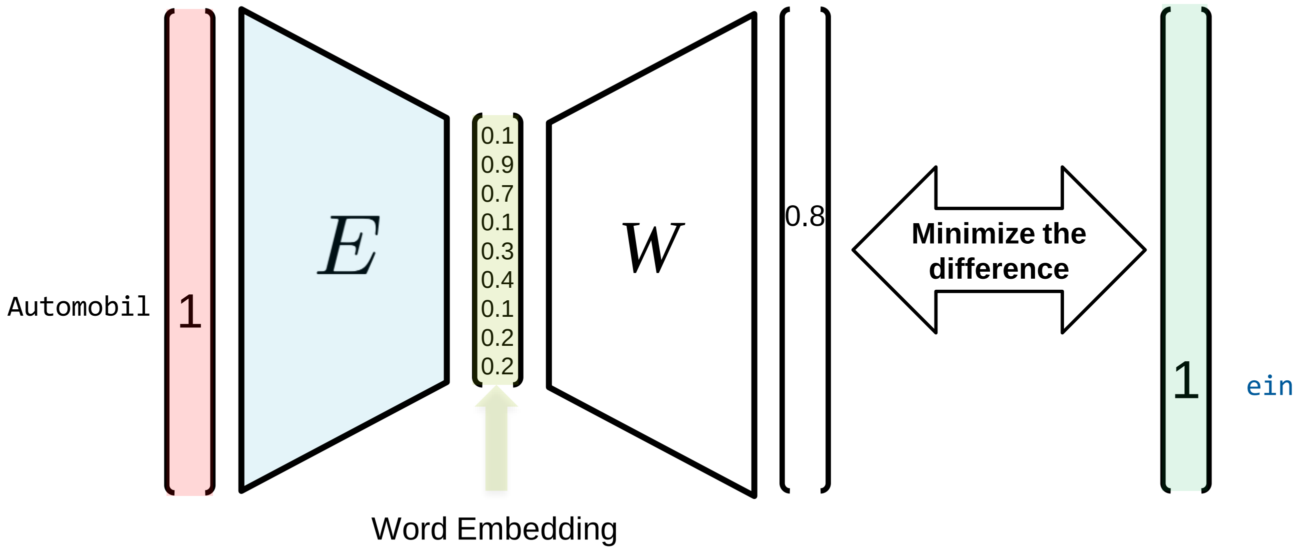
Update Equations

Start with randomly initialized C and W matrices, then incrementally do updates

$$
\begin{aligned}
c_{pos}^{t+1} &= c_{pos}^{t} - \eta[\sigma(c_{pos}^{t} \cdot w^{t}) - 1]w^{t} \\
c_{neg}^{t+1} &= c_{neg}^{t} - \eta[\sigma(c_{neg}^{t} \cdot w^{t})]w^{t} \\
w^{t+1} &= w^{t} - \eta\left[[\sigma(c_{pos} \cdot w^{t}) - 1]c_{pos} + \sum_{i=1}^{k}[\sigma(c_{\mathrm{neg}_i} \cdot w^{t})]c_{neg_i}\right]
\end{aligned}
$$

Training is computational expensive.

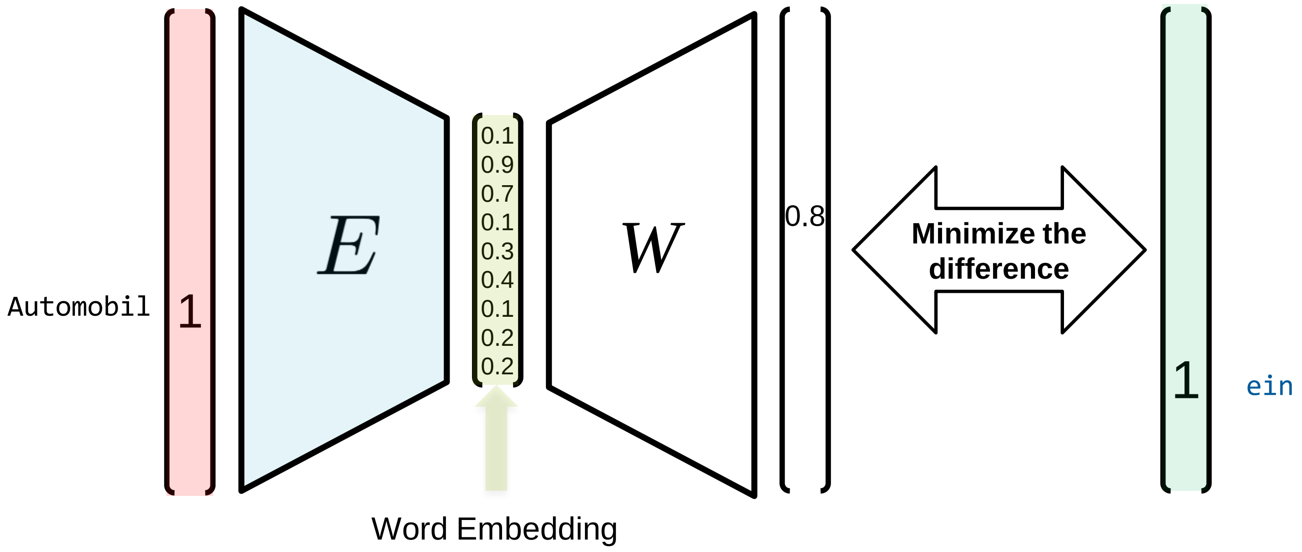
`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



- **Any pair of target/context word**

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



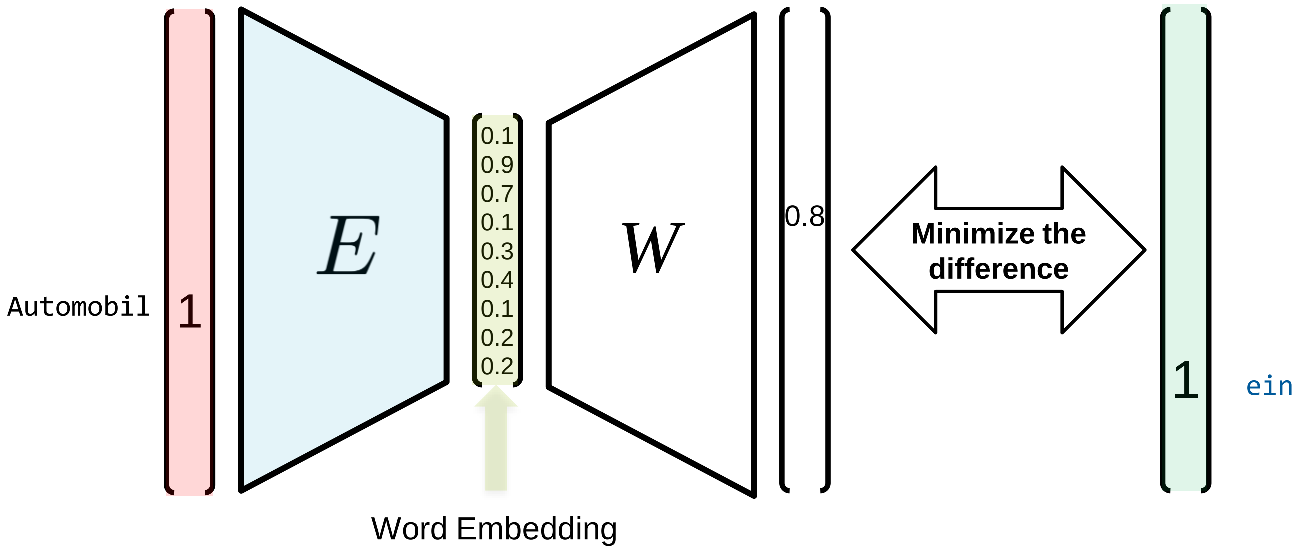
- **Any pair of target/context word**

  Solutions:

  - Sampling

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



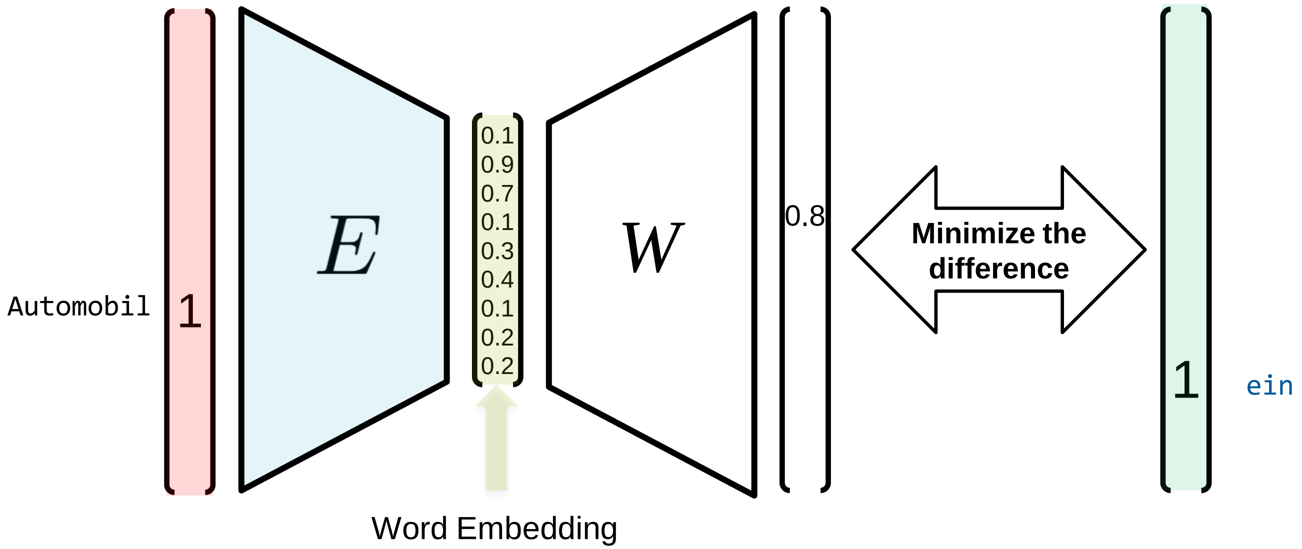
- **Any pair of target/context word**

  Solutions:

  - Sampling
- **Softmax over vocabulary**

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



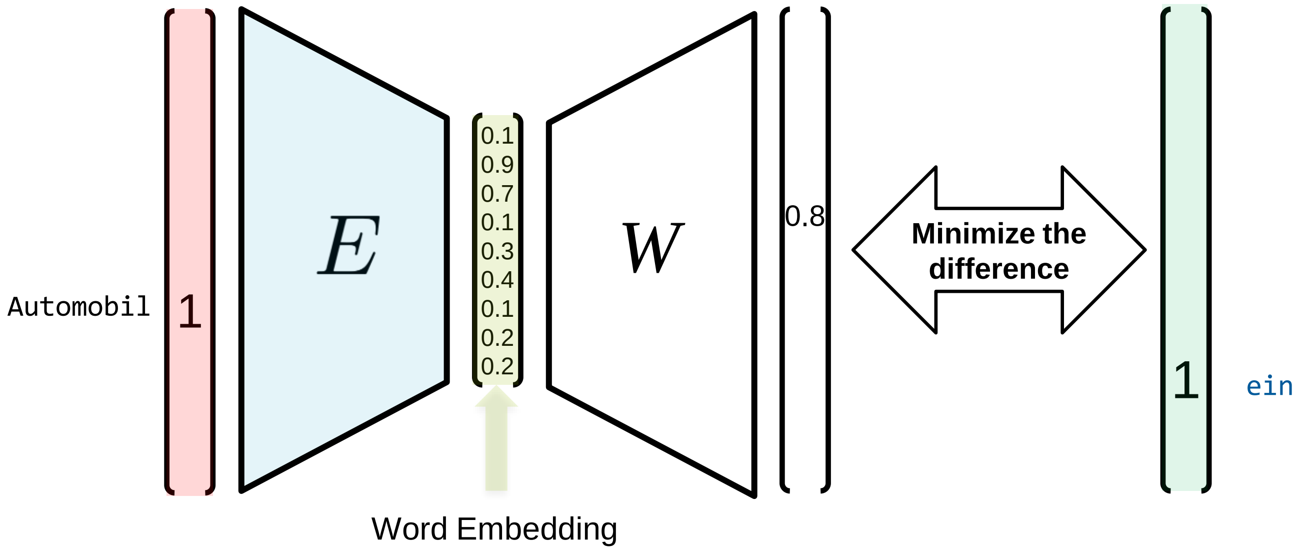
- **Any pair of target/context word**

  Solutions:
  - Sampling
- **Softmax over vocabulary**

  Solution:
  - Hierarchical Softmax

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



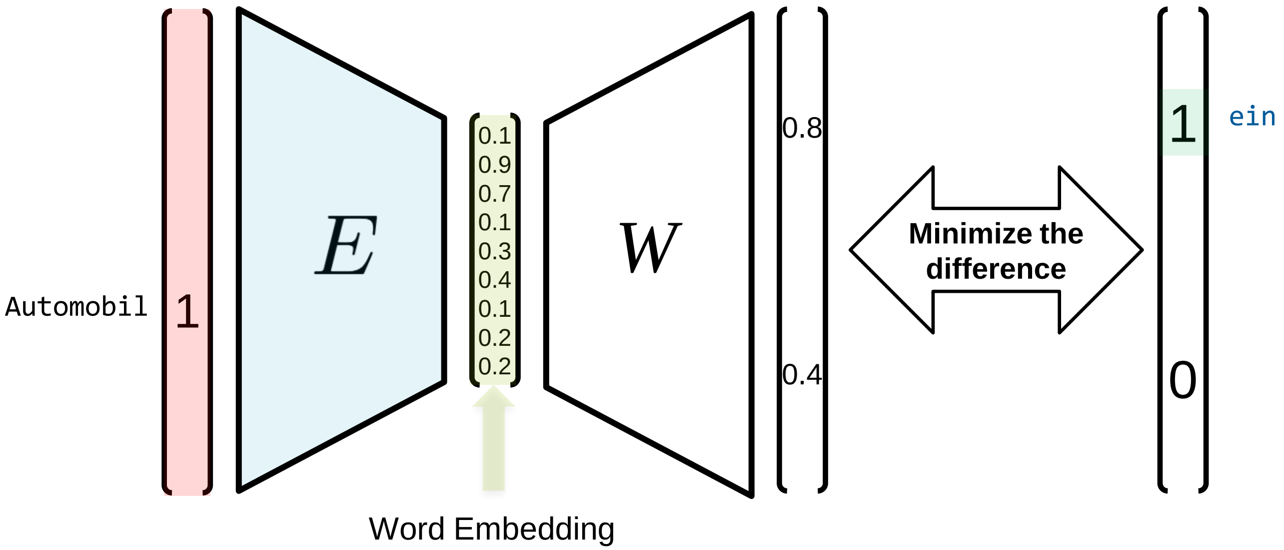
- **Any pair of target/context word**

  Solutions:
  - Sampling
- **Softmax over vocabulary**

  Solution:
  - Hierarchical Softmax
  - Noise Contrastive Estimation
    - Negative Sampling

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`



Move to binary classification:

- Replace Softmax by Sigmoid
- Train with positive and negative samples

Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	
Haste	Automobil	:= 0	Negative Sample
oben	Automobil	:= 0	
auf	Automobil	:= 0	

Example

Training Skip-Gram Word Embedding with Negative Sampling

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

$P(+|t, c)$

Positive Sample

Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

This is a binary classifier $\left\{ \begin{array}{l} P(+|t,c) \\ \text{Positive Sample} \end{array} \right.$

Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	

Training is computational expensive. Replace softmax with sigmoid.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

This is a binary classifier $\left\{ \begin{array}{l} P(+|t,c) \\ \text{Positive Sample} \end{array} \right.$

⇩

$$P(+|t,c) = \frac{1}{1+e^{-t \cdot c}}$$

Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

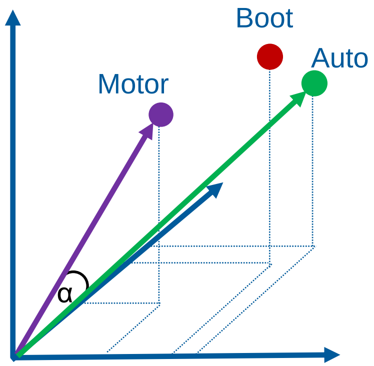
Training is computational expensive. Replace softmax with sigmoid.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

This is a binary classifier $\left\{ \begin{array}{l} P(+|t,c) \\ \text{Positive Sample} \end{array} \right.$

$$P(+|t,c) = \frac{1}{1+e^{-t \cdot c}}$$

$$Similarity(t,c) \approx t \cdot c$$



Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	

Cosine Similarity:

$$\cos \sphericalangle(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}|\,|\vec{b}|}$$

Training is computational expensive. A (binary) classifier needs “negative samples”.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

$$P(-|t,c) = 1 - P(+|t,c)$$

This is a binary classifier $\left\{ \begin{array}{l} P(+|t,c) \\ \text{Positive Sample} \end{array} \right.$

Negative Sample

`Schinken`

$$P(+|t,c) = \frac{1}{1+e^{-t \cdot c}}$$

Context	Target		
ein	Automobil	:= 1	Positive Sample
mit	Automobil	:= 1	
Haste	Automobil	:= 0	Negative Sample
oben	Automobil	:= 0	
auf	Automobil	:= 0	

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Example

Training Skip-Gram Word Embedding with Negative Sampling

Training is computational expensive. A (binary) classifier needs “negative samples”.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

$$P(-|t,c) = 1 - P(+|t,c)$$

This is a binary classifier $\left\{ P(+|t,c) \right.$

Positive Sample

Negative Sample

`Schinken`

$$P(+|t,c) = \frac{1}{1+e^{-t\cdot c}}$$

$$P(-|t,c) = 1 - P(+|t,c) = \frac{e^{-t\cdot c}}{1+e^{-t\cdot c}}$$

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Example

Training Skip-Gram Word Embedding with Negative Sampling

- Maximize the similarity of the target word, context word pairs (t,c) drawn from the positive examples
- Minimize the similarity of the (t,c) pairs drawn from the negative examples.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

$$P(-|t,c) = 1 - P(+|t,c)$$

This is a binary classifier $\left\{ P(+|t,c) \right.$

Positive Sample

Negative Sample

`Schinken`

$$\underbrace{P(+|t,c) = \frac{1}{1+e^{-t\cdot c}} \qquad P(-|t,c) = 1 - P(+|t,c) = \frac{e^{-t\cdot c}}{1+e^{-t\cdot c}}}$$

$$L(\theta) = \log \frac{1}{1+e^{-c\cdot t}} + \sum_{i=1}^{k} \log \frac{1}{1+e^{n_i\cdot t}}$$

For k noise samples

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Example

Training Skip-Gram Word Embedding with Negative Sampling

Training is computational expensive.

Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.

$P(-|t,c) = 1 - P(+|t,c)$

Remember: We have a context window

$$P(+|t,c_{1:k}) = \prod_{i=1}^{k} \frac{1}{1+e^{-t\cdot c_i}}$$

$$\log P(+|t,c_{1:k}) = \sum_{i=1}^{k} \log \frac{1}{1+e^{-t\cdot c_i}}$$

Independents of context words assumed

This is a binary classifier $\left\{ \begin{array}{l} P(+|t,c) \\ \text{Positive Sample} \end{array} \right.$

Negative Sample

Schinken

$$P(+|t,c) = \frac{1}{1+e^{-t\cdot c}}$$

$$P(-|t,c) = 1 - P(+|t,c) = \frac{e^{-t\cdot c}}{1+e^{-t\cdot c}}$$

$$L(\theta) = \log \frac{1}{1+e^{-c\cdot t}} + \sum_{i=1}^{k} \log \frac{1}{1+e^{n_i\cdot t}}$$

For k noise samples

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Example

Training Skip-Gram Word Embedding with Negative Sampling

Training is computational expensive.

Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.

$P(-|t,c) = 1 - P(+|t,c)$

Remember: We have a context window

$$P(+|t,c_{1:k}) = \prod_{i=1}^{k} \frac{1}{1+e^{-t\cdot c_i}}$$

$$\log P(+|t,c_{1:k}) = \sum_{i=1}^{k} \log \frac{1}{1+e^{-t\cdot c_i}}$$

Independents of context words assumed

This is a binary classifier $\left\{ P(+|t,c) \right.$ Positive Sample

Negative Sample

Schinken

How to select the negative samples from the vocabulary?

$$P(w) = \frac{count(w)}{\sum_{w'} count(w')}$$

Uni-Gram Probabilities

$$P(+|t,c) = \frac{1}{1+e^{-t\cdot c}} \qquad P(-|t,c) = 1 - P(+|t,c) = \frac{e^{-t\cdot c}}{1+e^{-t\cdot c}}$$

$$L(\theta) = \log \frac{1}{1+e^{-c\cdot t}} + \sum_{i=1}^{k} \log \frac{1}{1+e^{n_i\cdot t}}$$

For k noise samples

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Example

Training Skip-Gram Word Embedding with Negative Sampling

Training is computational expensive.

`Ein Elektroauto ist ein Automobil mit elektrischem Antrieb.`

$P(-|t,c) = 1 - P(+|t,c)$

Remember: We have a context window

$$P(+|t,c_{1:k}) = \prod_{i=1}^{k} \frac{1}{1+e^{-t \cdot c_i}}$$

$$\log P(+|t,c_{1:k}) = \sum_{i=1}^{k} \log \frac{1}{1+e^{-t \cdot c_i}}$$

Independents of context words assumed

This is a binary classifier $\left\{ P(+|t,c) \right.$ Positive Sample

Negative Sample

`Schinken`

How to select the negative samples from the vocabulary?

$$P_\alpha(w) = \frac{count(w)^\alpha}{\sum_{w'} count(w')^\alpha}$$

Weighted Uni-Gram Probabilities

Rare words: $P_\alpha(w) > P(w)$

$$P(+|t,c) = \frac{1}{1+e^{-t \cdot c}}$$

$$P(-|t,c) = 1 - P(+|t,c) = \frac{e^{-t \cdot c}}{1+e^{-t \cdot c}}$$

$$L(\theta) = \log \frac{1}{1+e^{-c \cdot t}} + \sum_{i=1}^{k} \log \frac{1}{1+e^{n_i \cdot t}}$$

For k noise samples

Sigmoid:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$