NLP Basic Architectures

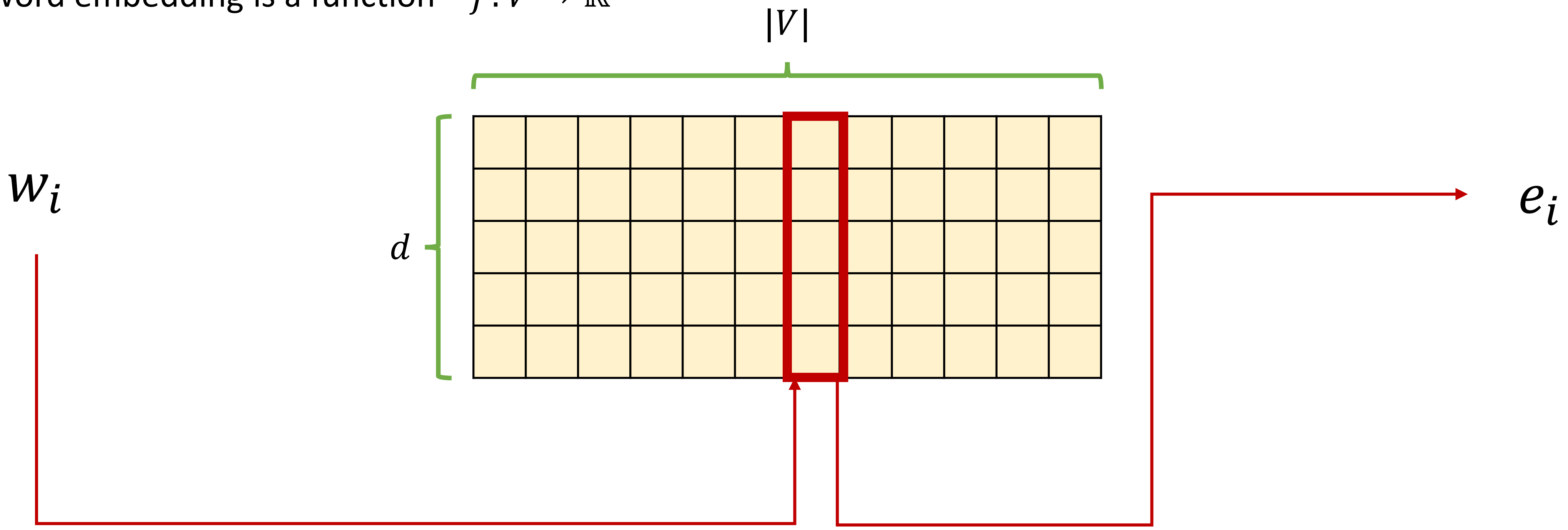
Seminar in Deep Neural Network

Presenter: Nathan Corecco
Supervisor: Ard Kastrati
11/04/2022

Word Embedding

$V = \{w_1, w_2, \dots, w_m\}$

A word embedding is a function $f: V \to \mathbb{R}^d$



Or equivalent to: $w_i \to o_i \to E \cdot o_i = e_i$

Example

Vocabulary:

$V = \{\text{ a, Alfred, an, apple, farmer, grower, is, John, Kiwano, Smith, Weber, } \ldots \}$

$|V| = 10'000$

Task:

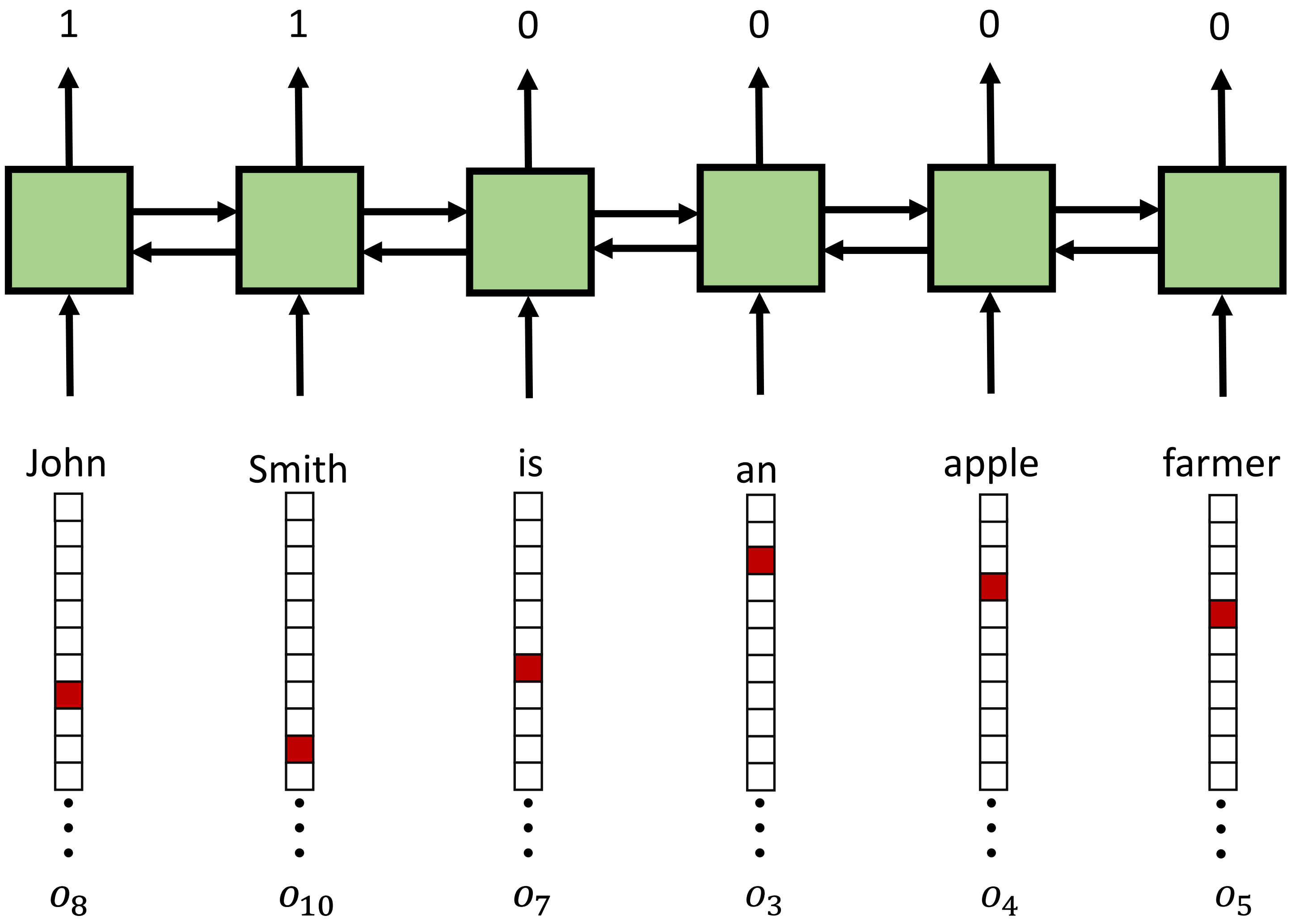
Name entity recognition

Training set: D_1 = {…, John Smith is an apple farmer,… }

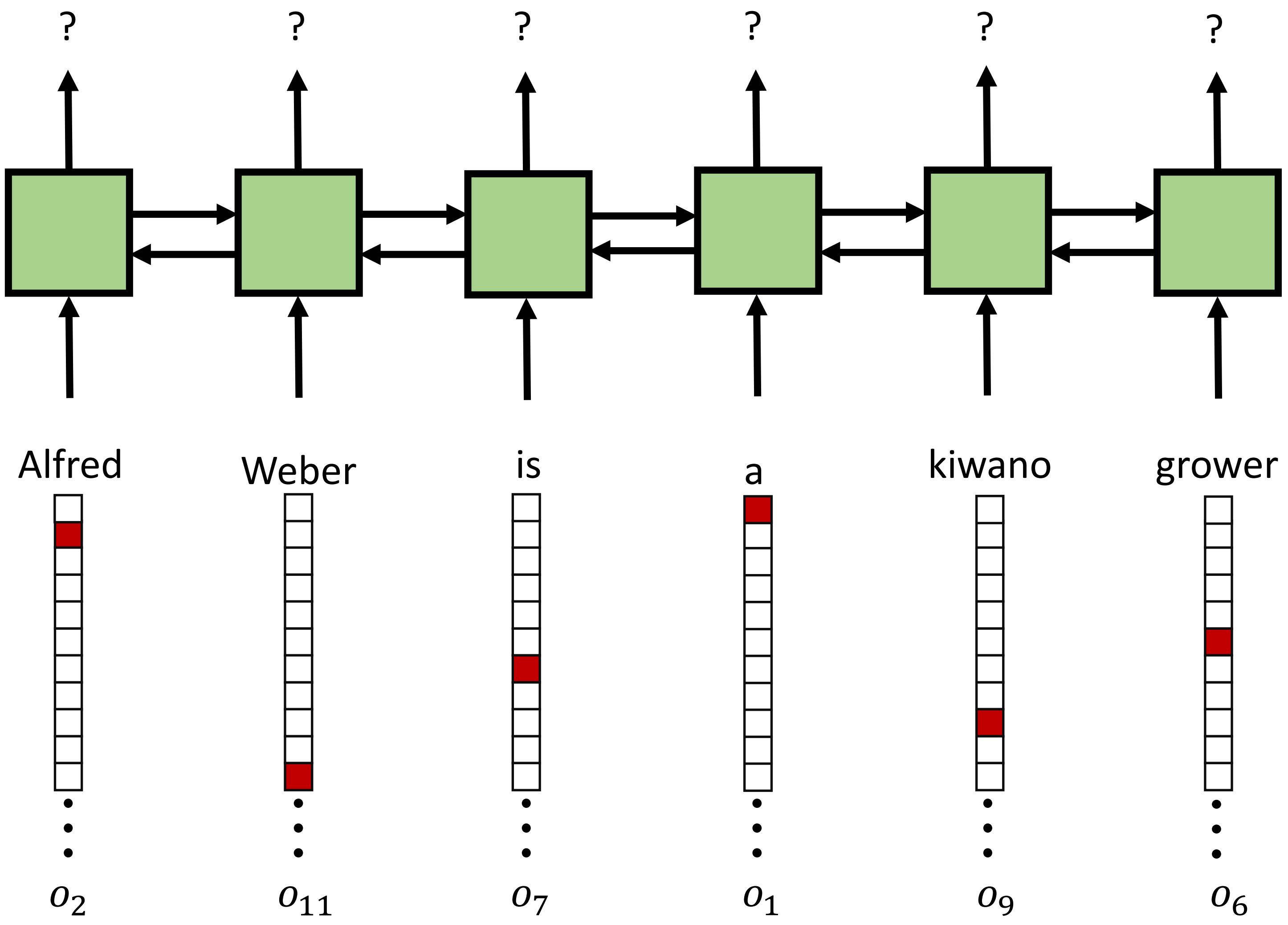
Test set: D_2 = {…, Alfred Weber is a Kiwano grower,… }

Word embedding: we use the one hot encoding, for word j in the vocabulary we map it to o_j

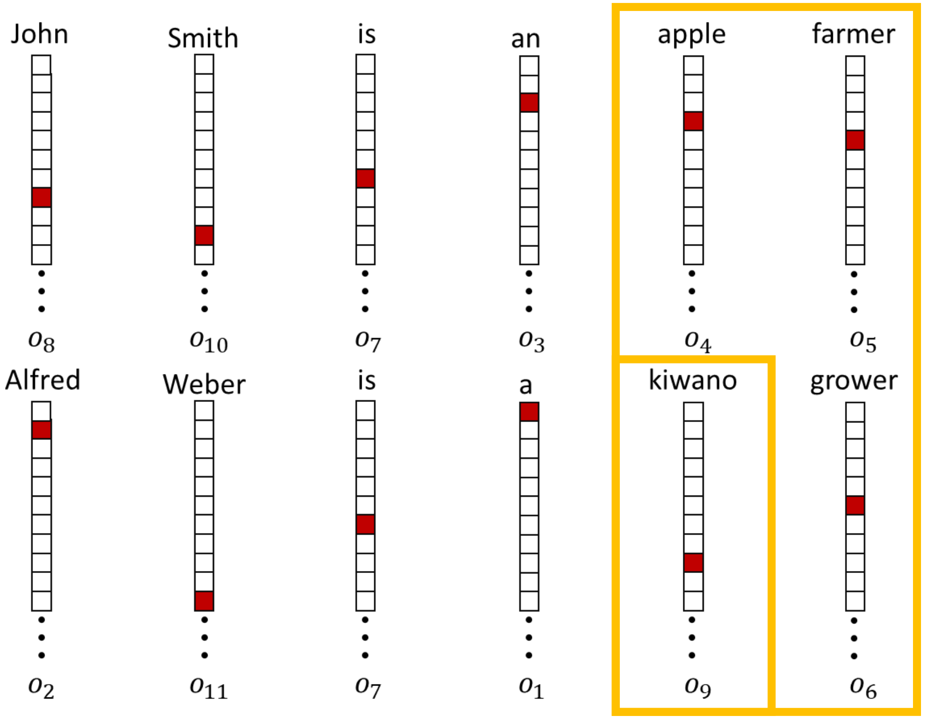
Training Phase



Testing Phase



Problems



- Kiwano is a rare word

- Word with similar meaning don't have a similar vector

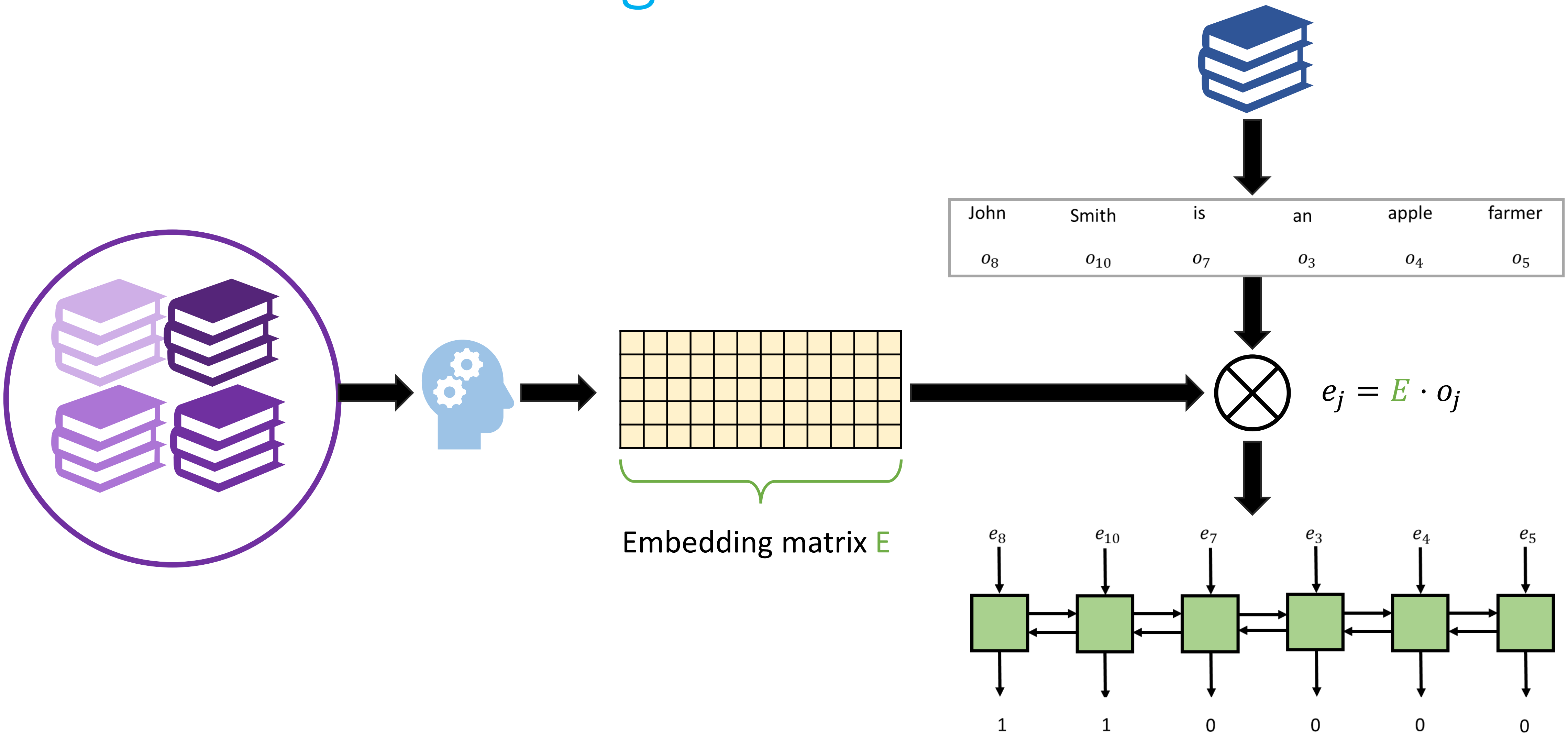
$$sim(u, v) = \frac{u^T v}{\|u\| \cdot \|v\|}$$

$$sim(\text{apple}, \text{kiwano}) = \frac{o_4^T o_9}{\|o_4\| \cdot \|o_9\|} = 0$$

$$sim(\text{farmer}, \text{grower}) = \frac{o_5^T o_6}{\|o_5\| \cdot \|o_6\|} = 0$$

Hard to learn !

Word Embedding



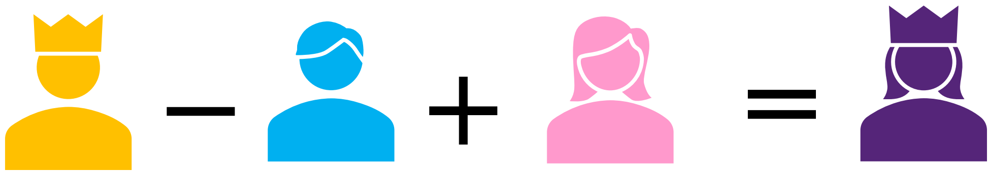
Properties of a good Word Embedding

	Man	Woman	King	Queen	Apple	Kiwano	Granpa
Gender	-1	1	-0.96	0.97	0.01	0	-0.97
Royal	0.02	0.03	0.94	0.93	0	0.02	0.1
Age	0.01	0.02	0.65	0.63	0.03	-0.04	0.072
Fruit	-0.07	0.06	0.02	0.03	0.98	0.99	0.01

- Word with similar meaning have a similar embedding:

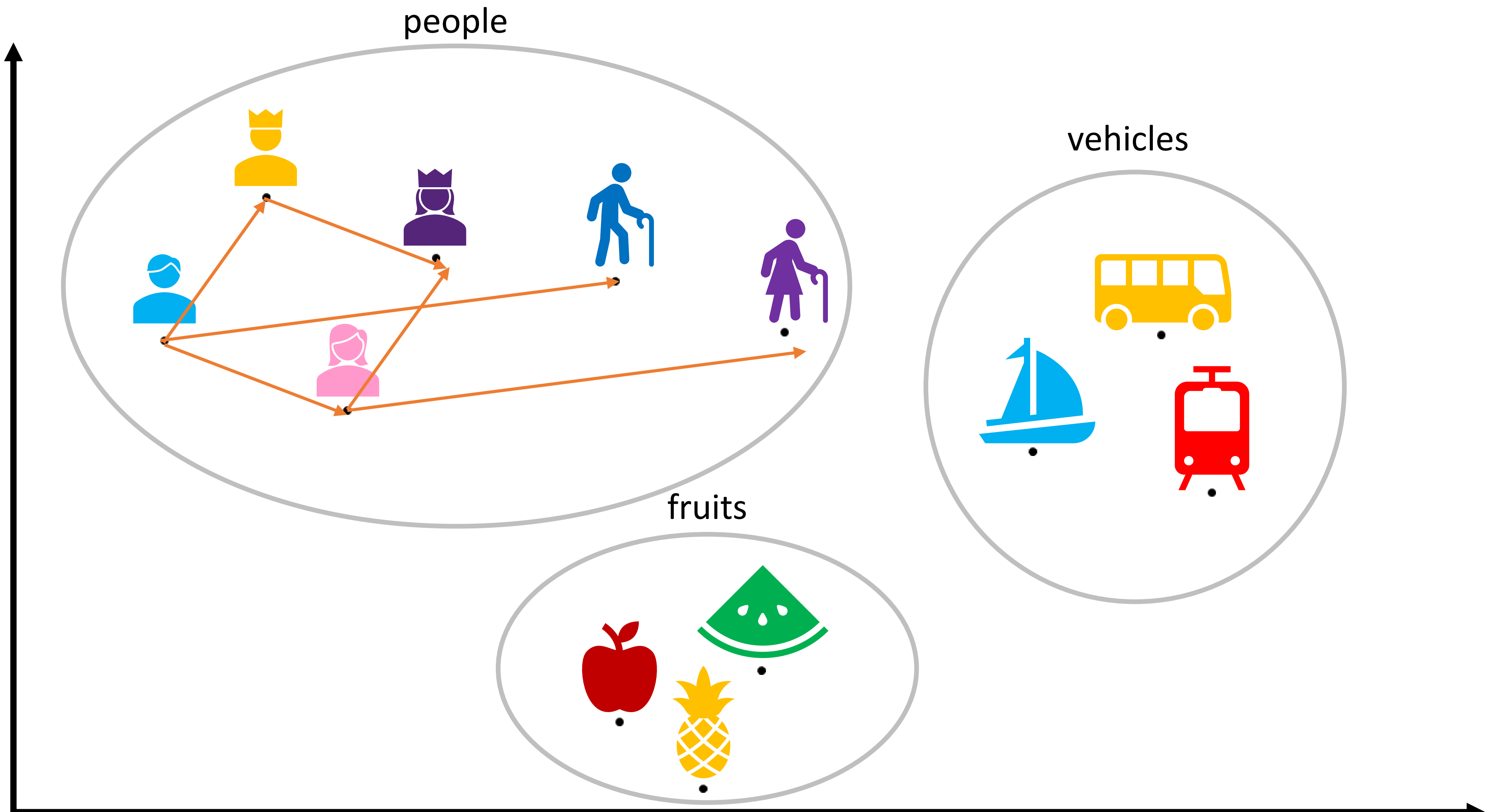
$$sim(\text{apple}, \text{kiwano}) = \frac{(\mathrm{e}_{\text{apple}}^{\mathrm{T}} \cdot e_{kiwano})}{\left|\left|e_{apple}\right|\right| \cdot \left|\left|e_{kiwano}\right|\right|} \approx 1$$

- Relation are preserved:



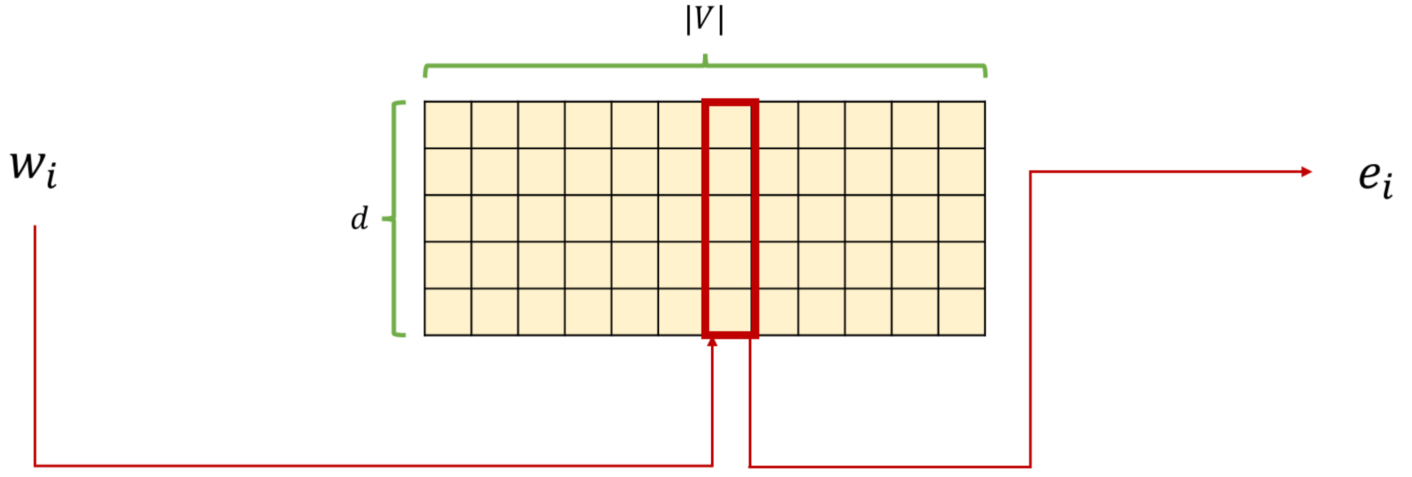
$$sim(\text{king} - \text{man} + \text{woman}, \text{queen}) \approx 1$$

What a good word embedding can learn



Skip Gram Model

- Easy model that allows to learn a good word embedding
- We learn E by predicting words
- Vocabulary: $V = \{w_1, w_2, \dots, w_{10000}\}$
- Embedding dimension: d = 300

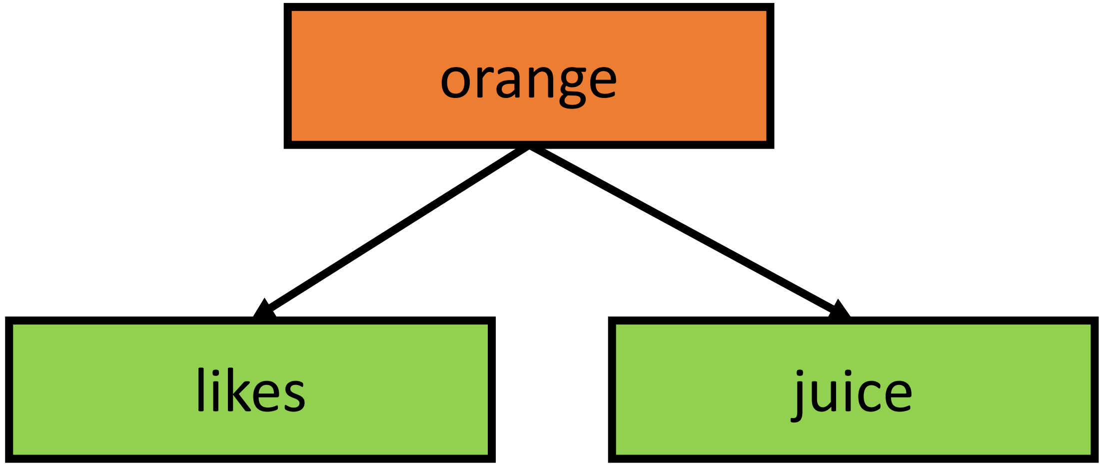


Or equivalent to: $w_i \rightarrow o_i \rightarrow E \cdot o_i = e_i$

Skip Gram Model

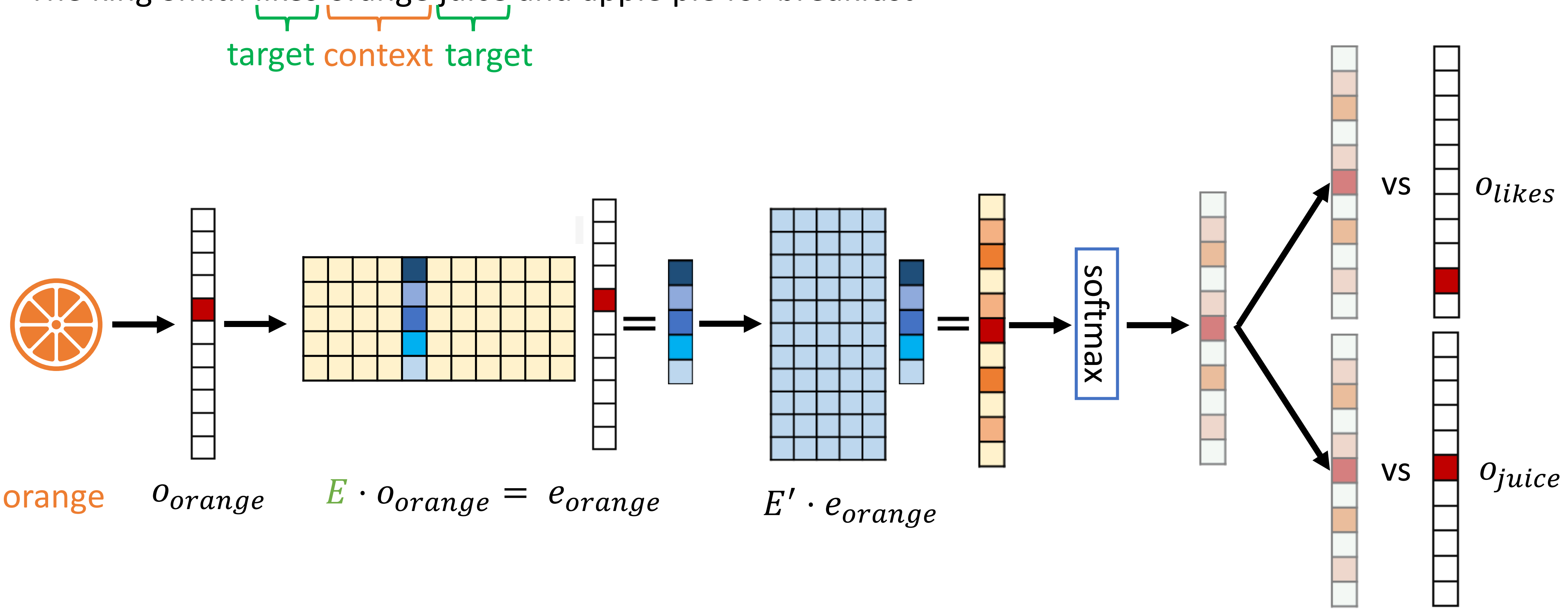
The king Smith likes orange juice and apple pie for breakfast

target context target



Skip Gram Model

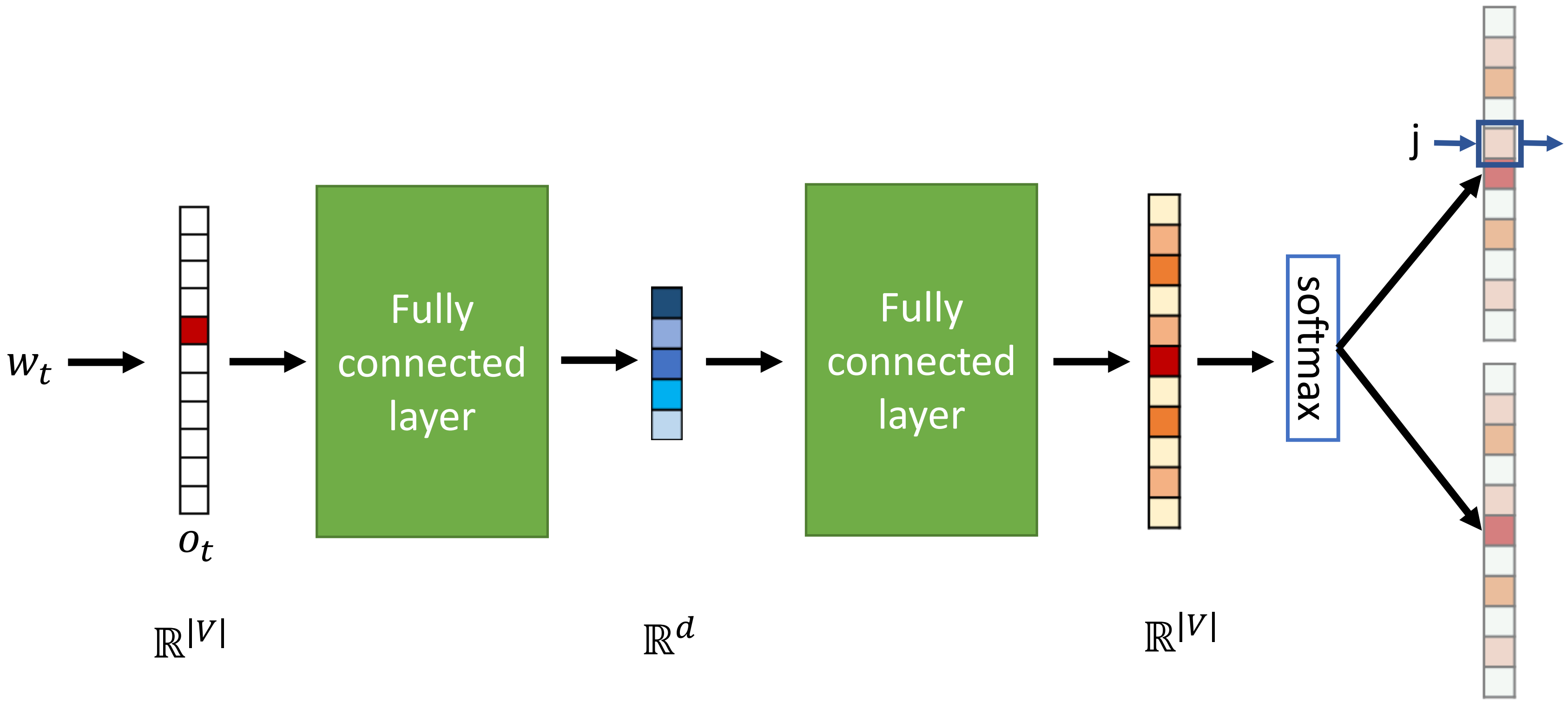
The king Smith likes orange juice and apple pie for breakfast



Notation: j-th column of E:= e_j, j–th row of E':= $e_j{}'$

Skip Gram Model as Neural Network

d: dimension of the embedding

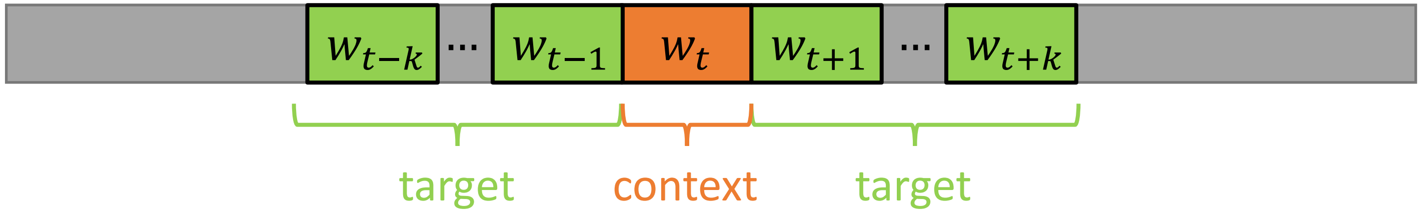


$$p(w_j|w_t) = \frac{\exp(e'^{T}_{w_j} e_{w_t})}{\sum_{w \in V} \exp({e'_w}^{T} e_{w_t})}$$

Loss: $L_\theta(w_t) = \mathrm{NLL} = -\log\big(p_\theta(w_{t-1}|w_t)\big) - \log\big(p_\theta(w_{t+1}|w_t)\big)$

Skip Gram Model

- We can generalize to a window size of k



- We can consider a sequence of words $w_1, w_2. \dots, w_T$ instead of a single word at the time

- General loss:

$$L_\theta(w_1, \dots, w_T) = -\frac{1}{T} \sum_{t \in [T]} \sum_{-k \leq j \leq k, j \neq 0} \log(p_\theta(w_{t+j} | w_t)) \text{ , where } p_\theta\left(w_j \middle| w_t\right) = \frac{\exp(e'^T_{w_j} e_{w_t})}{\sum_{w \in V} \exp(e'^T_w e_{w_t})}$$

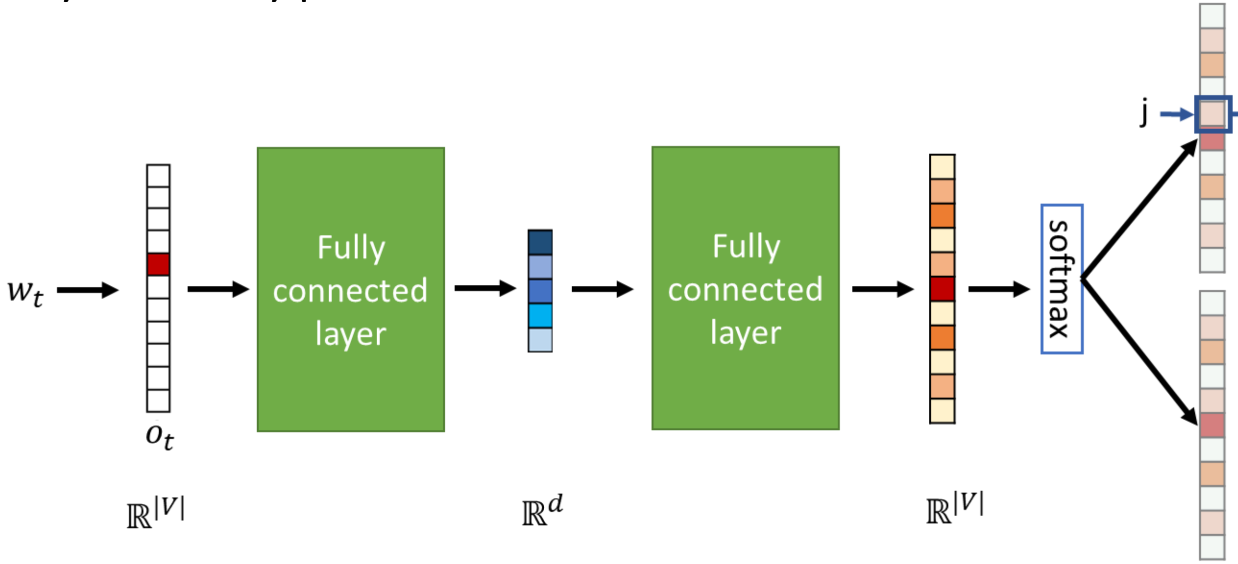
Why this model works?

John likes $\underbrace{\text{orange}}_{\text{context}}$ juice with cereals for breakfast

Alice likes to drink $\underbrace{\text{apple}}_{\text{context}}$ juice with cereals

Problems

Do you see any problem?

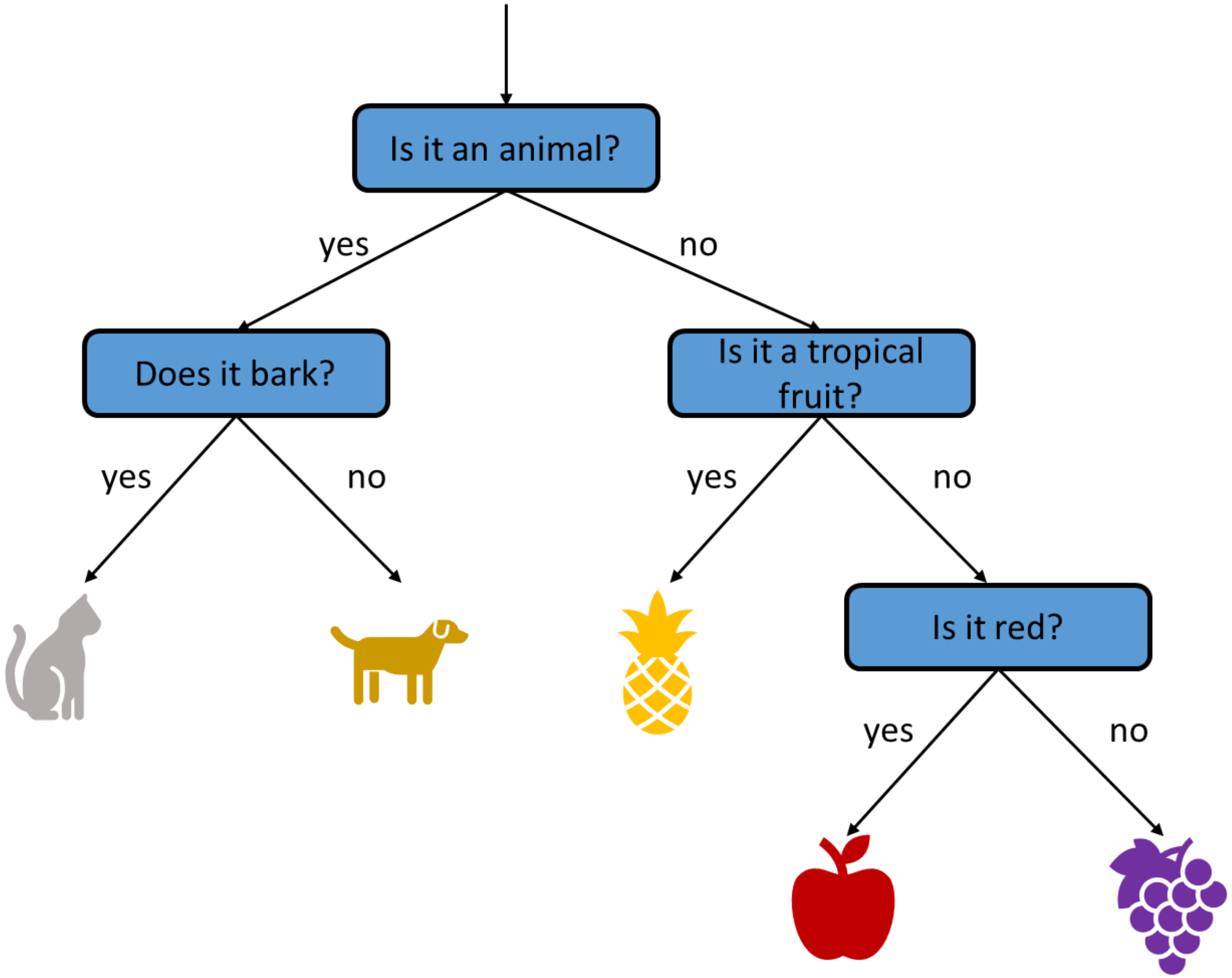


$p(w_j|w_t) = \dfrac{\exp(e'^{T}_{w_j} e_{w_t})}{\boxed{\sum_{w \in V} \exp(e'^{T}_{w} e_{w_t})}}$

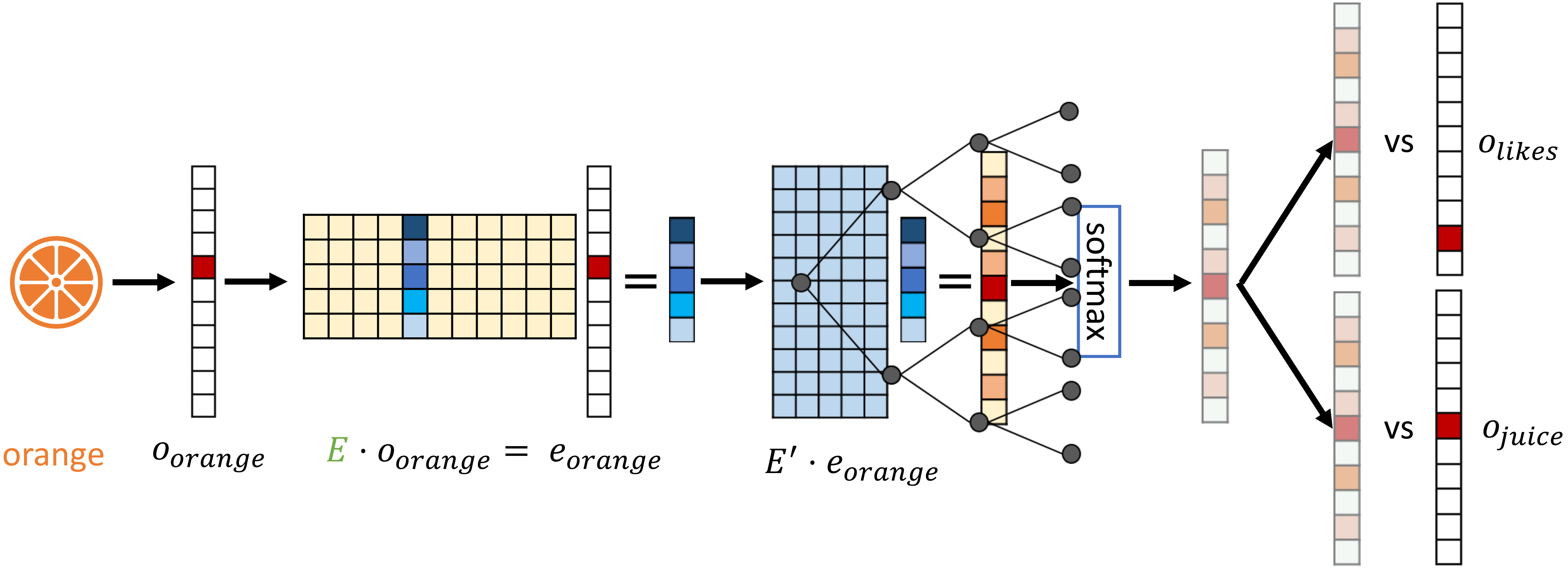
In the English vocabulary there are 171,146 words!

Loss: $L_\theta(w_t) = \text{NLL} = -\log\big(p_\theta(w_{t-1}|w_t)\big) - \log\big(p_\theta(w_{t+1}|w_t)\big)$

Hierarchical Softmax



Hierarchical Softmax



Loss: $L_\theta(w_t) = \mathrm{NLL} = -\log\big(p_\theta(w_{t-1}|w_t)\big) - \log\big(p_\theta(w_{t+1}|w_t)\big)$

Negative sampling

- This methods define a new learning problem: distinguish context target pair from k context random pair

The king Smith likes $\underbrace{\text{orange}}_{\text{context}}$ $\underbrace{\text{juice}}_{\text{target}}$ and apple pie for breakfast

$\underbrace{\text{queen, auto, mountain, the}}_{k\,=\,4\text{ random words}}$

Context	Word	Target?	
orange	juice	1	positive example
orange	queen	0	negative example
orange	auto	0	
orange	mountain	0	
orange	the	0	

Negative sampling

- The task is a supervised learning from x to y, more precisely we have k+1 binary classification

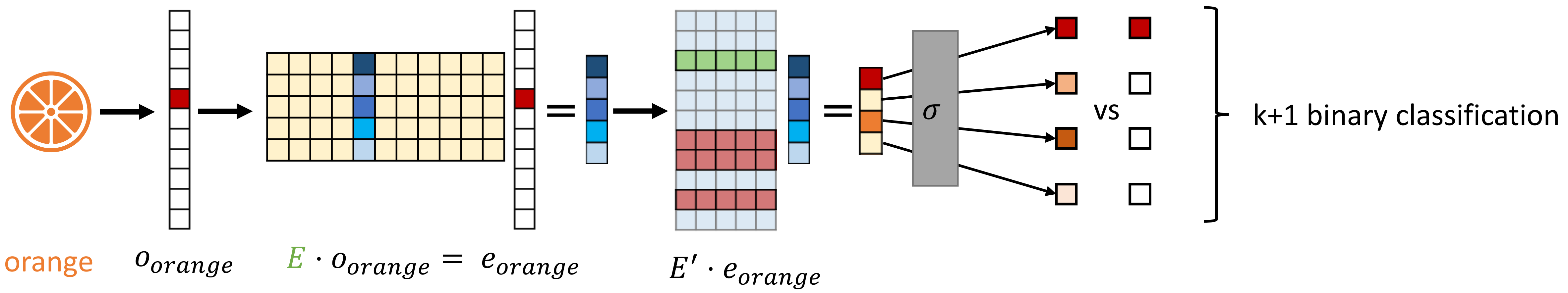
x y

c w

Context	Word	Target?
orange	juice	1
orange	queen	0
orange	auto	0
orange	mountain	0
orange	the	0

$$p_\theta(y = 1|c, w) = \sigma(e_w'^T e_c)$$

Negative sampling



- We use the sum of NLL for the loss

$$-\left(\log\left(\sigma\left(e_t'^{T} e_c\right)\right) + \sum_{w, w \neq t} \log\left(1 - \sigma\left(e_w'^{T} e_c\right)\right)\right)$$

Negative sampling

- For a context target pair (c,t) we define the negative sampling by the objective

Distribution used to draw negative samples

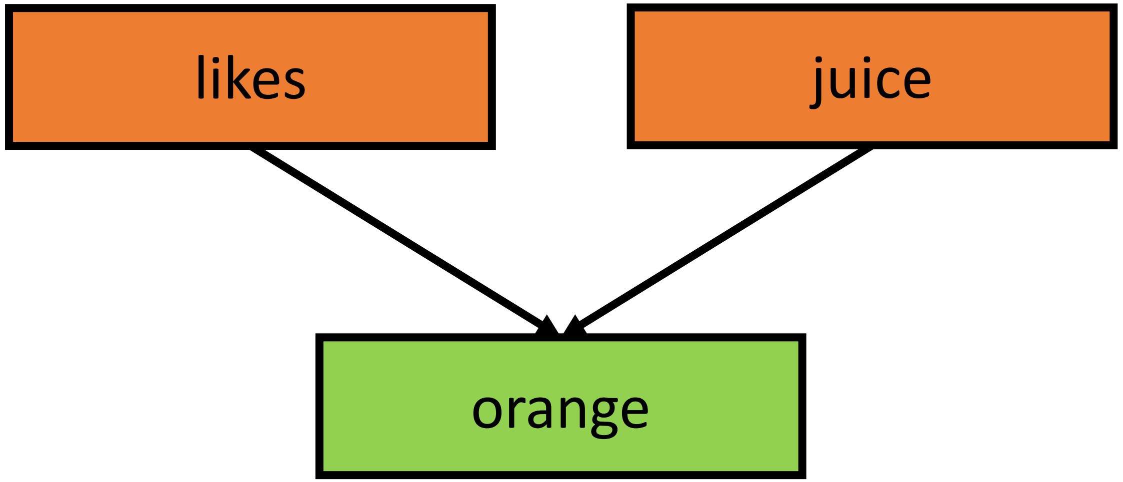
$$\underbrace{\log\left(\sigma\left(e_t'^T e_c\right)\right)}_{\text{LL of positive target}} + \sum_{i\in[\mathrm{k}]} \mathbb{E}_{w_i\sim P_n(w)} \underbrace{\left[\log\left(\sigma(-e_{w_i}'^T e_c)\right)\right]}_{\text{LL of negative sample } w_i}$$

- How to choose k? $k \in \{5, \dots, 20\}$ for small dataset, and $k \in \{2, \dots, 5\}$ for large dataset

- How to choose $P_n(w)$? $p(w_i) = \dfrac{f(w_i)^{3/4}}{\sum_w f(w)^{3/4}}$

Continuous Bag of Words (CBOW)

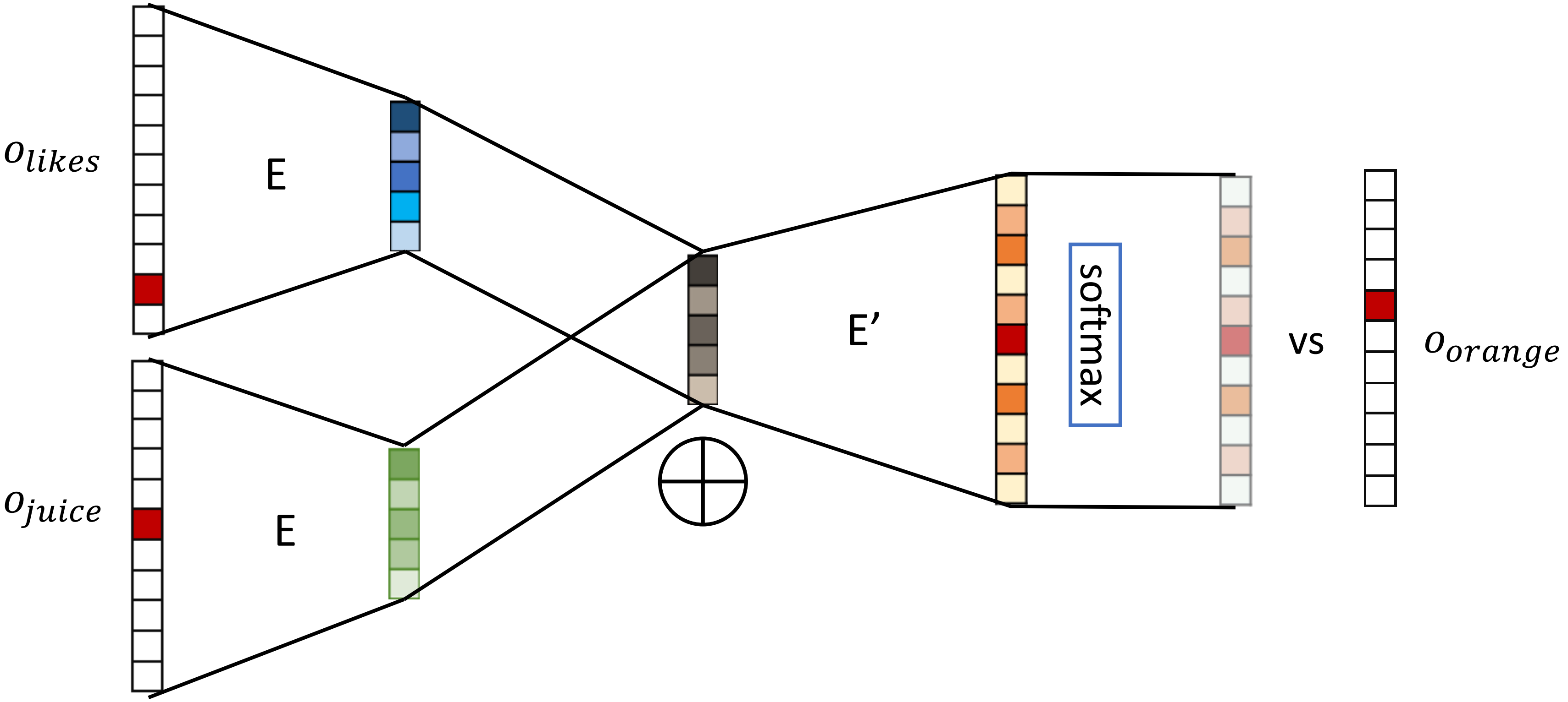
The king Smith $\underbrace{\text{likes}}_{\text{context}}$ $\underbrace{\text{orange}}_{\text{target}}$ $\underbrace{\text{juice}}_{\text{context}}$ and apple pie for breakfast



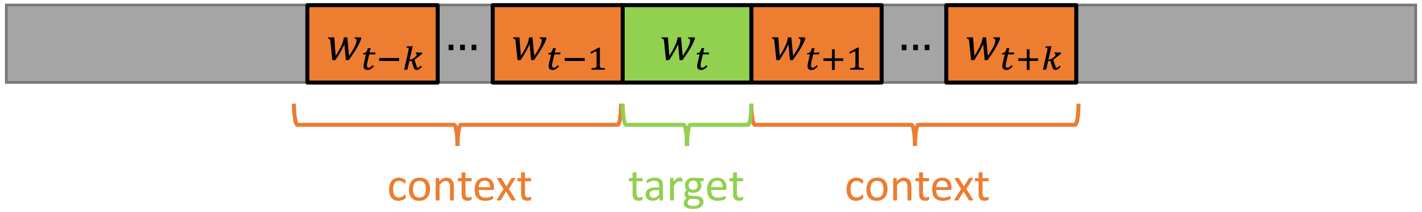
Continuous Bag of Words (CBOW)

The king Smith likes orange juice and apple pie for breakfast

$\underbrace{\text{likes}}_{\text{context}}\ \underbrace{\text{orange}}_{\text{target}}\ \underbrace{\text{juice}}_{\text{context}}$



Global Vectors for Word Representation (GloVe)



X_{ij} = #times i appear in context of the target j

Note: $X_{ij} = X_{ji}$

Global Vectors for Word Representation (GloVe)

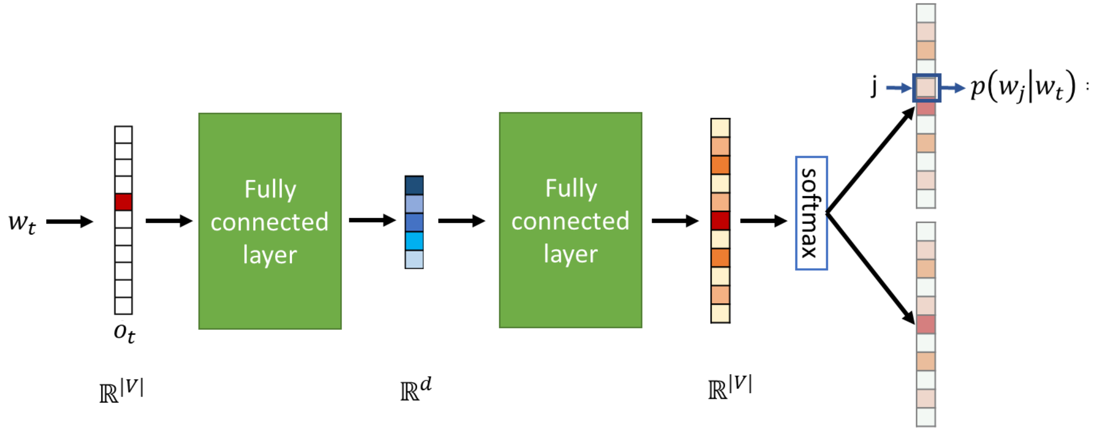
- The more i and j are related (X_{ij} large) the more the two vectors e_i', e_j should be correlated ($e_i'^T e_j$ large)
- If X_{ij} is zero we cannot take the logarithm
- We add bias

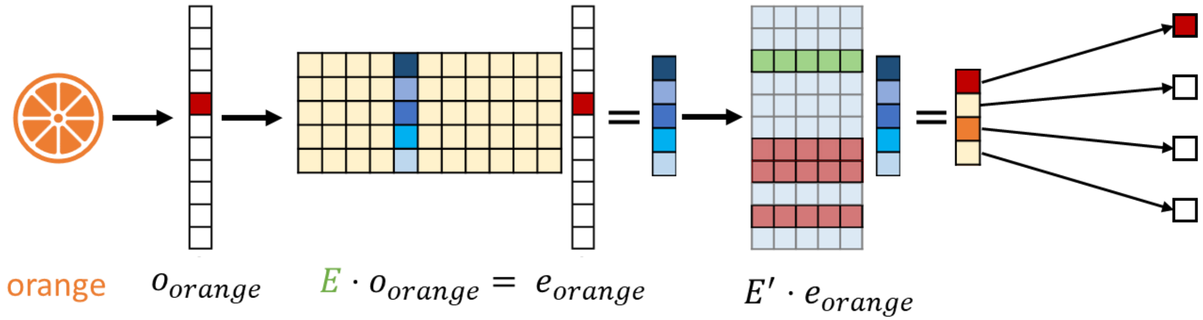
Objective:

$$\text{minimize} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \underbrace{f(X_{ij})}\left(e_i'^T e_j \underbrace{+ b_i + b_j}_{\text{bias}} - \log(X_{ij})\right)^2$$

$$f(X_{ij}) = \begin{cases} \left(\frac{X_{ij}}{x_{max}}\right)^\alpha & if\ x_{ij} < x_{max} \\ 1\ otherwise \end{cases}$$

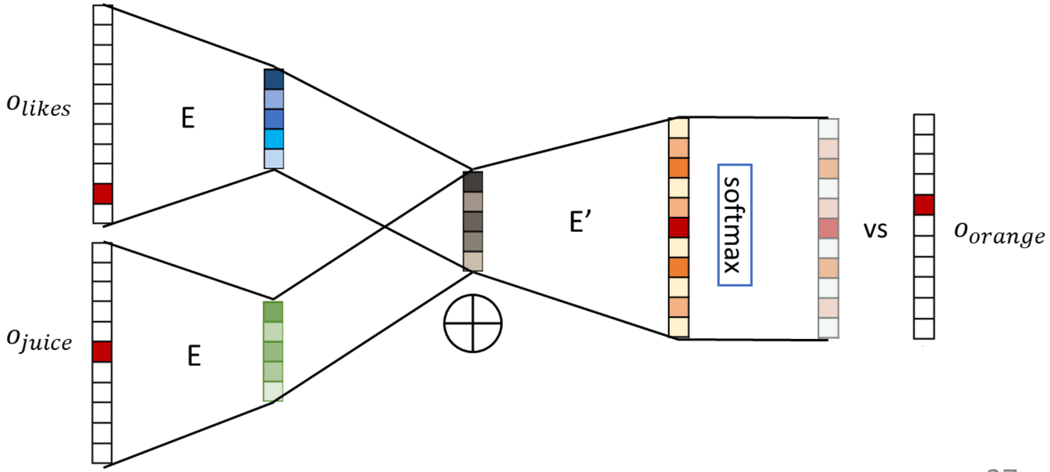
Summary





Loss: $L_\theta(w_t) = \text{NLL} = -\log\big(p_\theta(w_{t-1}|w_t)\big) - \log\big(p_\theta(w_{t+1}|w_t)\big)$

$$\text{minimize} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \underbrace{f(X_{ij})}_{f(X_{ij}) = \begin{cases} \left(\frac{X_{ij}}{x_{max}}\right)^{\alpha} & if\ x_{ij} < x_{max} \\ 1 & otherwise \end{cases}} \left(e_i'^T e_j \underbrace{+ b_i + b_j}_{\text{bias}} - \log(X_{ij})\right)^2$$



References

- Distributed Representations of Words and Phrases and their Composability: https://arxiv.org/pdf/1310.4546.pdf
- Global Vectors for Word Representation: https://nlp.stanford.edu/pubs/glove.pdf
- https://www.youtube.com/watch?v=36XuT5c9qvE
- https://www.youtube.com/watch?v=UqRCEmrv1gQ