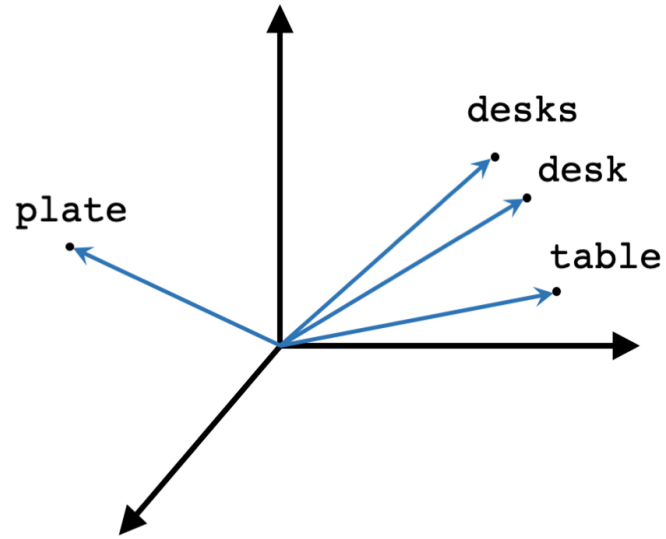
بنام خداوند جان و خرد

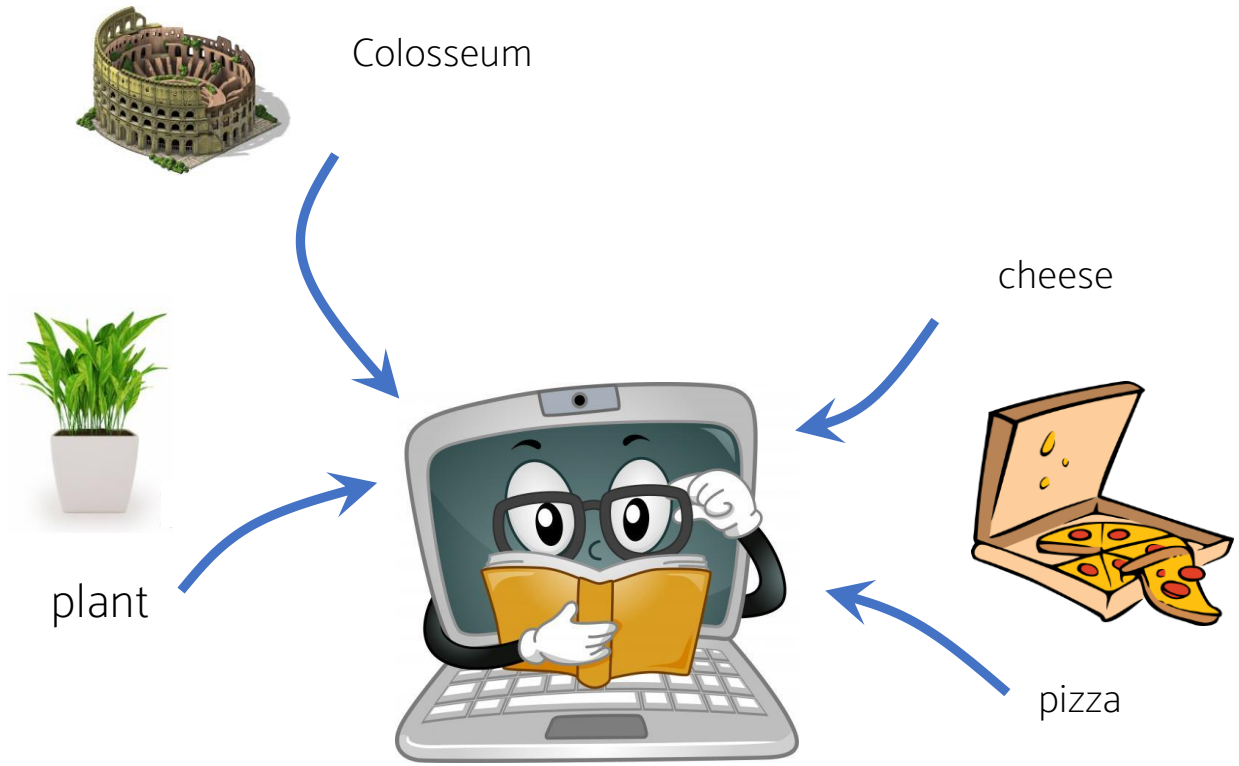


Semantic Representation

TeIAS NLP course 1400

Mohammad Taher Pilehvar

Represent meaning in a computer



Represent meaning in a computer

WordNet

Noun

- S: (n) **star** ((astronomy) a celestial body of hot gases that radiates energy derived from thermonuclear reactions in the interior)
- S: (n) ace, adept, champion, sensation, maven, mavin, virtuoso, genius, hotshot, **star**, superstar, whiz, whizz, wizard, wiz (someone who is dazzlingly skilled in any field)
- S: (n) **star** (any celestial body visible (as a point of light) from the Earth at night)
- S: (n) **star**, principal, lead (an actor who plays a principal role)
- S: (n) **star** (a plane figure with 5 or more points; often used as an emblem)
- S: (n) headliner, **star** (a performer who receives prominent billing)
- S: (n) asterisk, **star** (a star-shaped character * used in printing)
- S: (n) star topology, **star** (the topology of a network whose components are connected to a hub)

http://wordnetweb.princeton.edu/perl/webwn

Represent meaning in a computer

WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: plant Search WordNet

Display Options: (Select option to change) Change

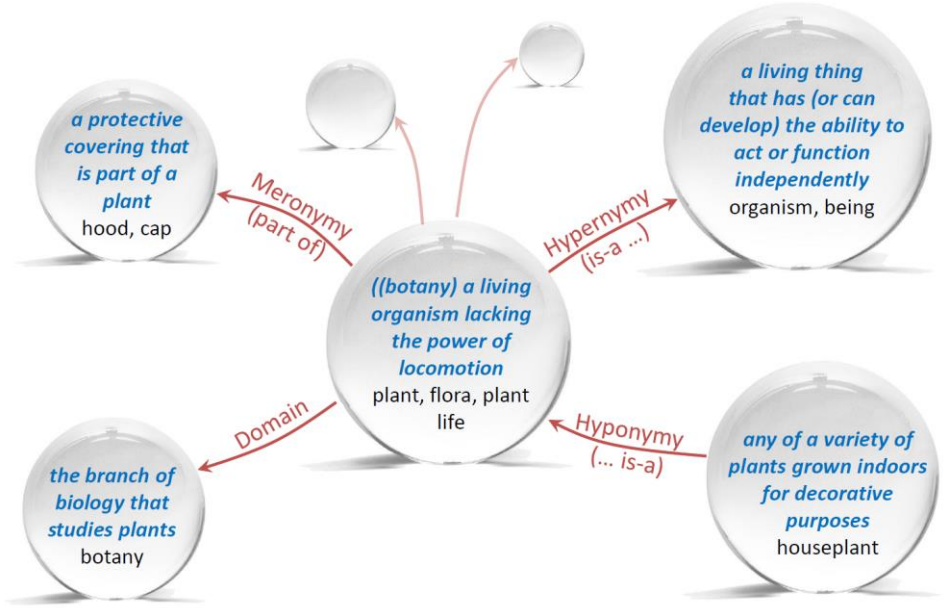
Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations
Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **plant**, works, industrial plant (buildings for carrying on industrial labor) *"they built a large plant to manufacture automobiles"*
- S: (n) **plant**, flora, plant life ((botany) a living organism lacking the power of locomotion)
- S: (n) **plant** (an actor situated in the audience whose acting is rehearsed but seems spontaneous to the audience)
- S: (n) **plant** (something planted secretly for discovery by another) *"the police used a plant to trick the thieves"; "he claimed that the evidence against him was a plant"*

Verb

- S: (v) **plant**, set (put or set (seeds, seedlings, or plants) into the ground) *"Let's plant flowers in the garden"*
- S: (v) implant, engraft, embed, imbed, **plant** (fix or set securely or deeply) *"He planted a knee in the back of his opponent"; "The dentist implanted a tooth in the gum"*
- S: (v) establish, found, **plant**, constitute, institute (set up or lay the groundwork for) *"establish a new department"*
- S: (v) **plant** (place into a river) *"plant fish"*
- S: (v) **plant** (place something or someone in a certain position in order to secretly observe or deceive) *"Plant a spy in Moscow"; "plant bugs in the dissident's apartment"*
- S: (v) **plant**, implant (put firmly in the mind) *"Plant a thought in the students' minds"*

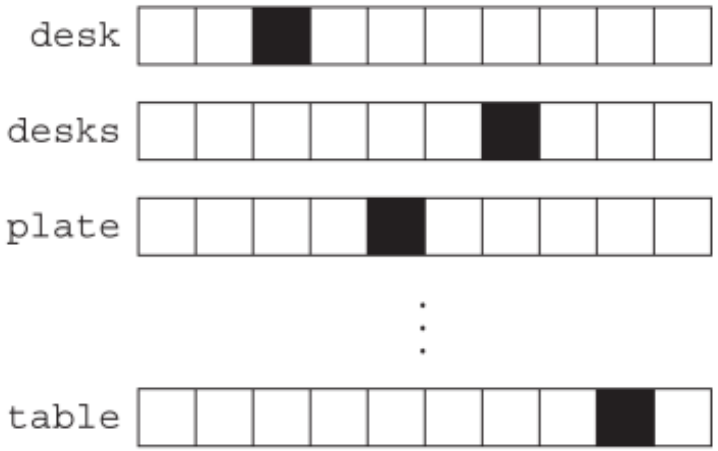


Represent meaning in a computer

Limitations of WordNet

- Difficult to create for new languages
- Difficult to update for new meanings/words
- Word similarity?

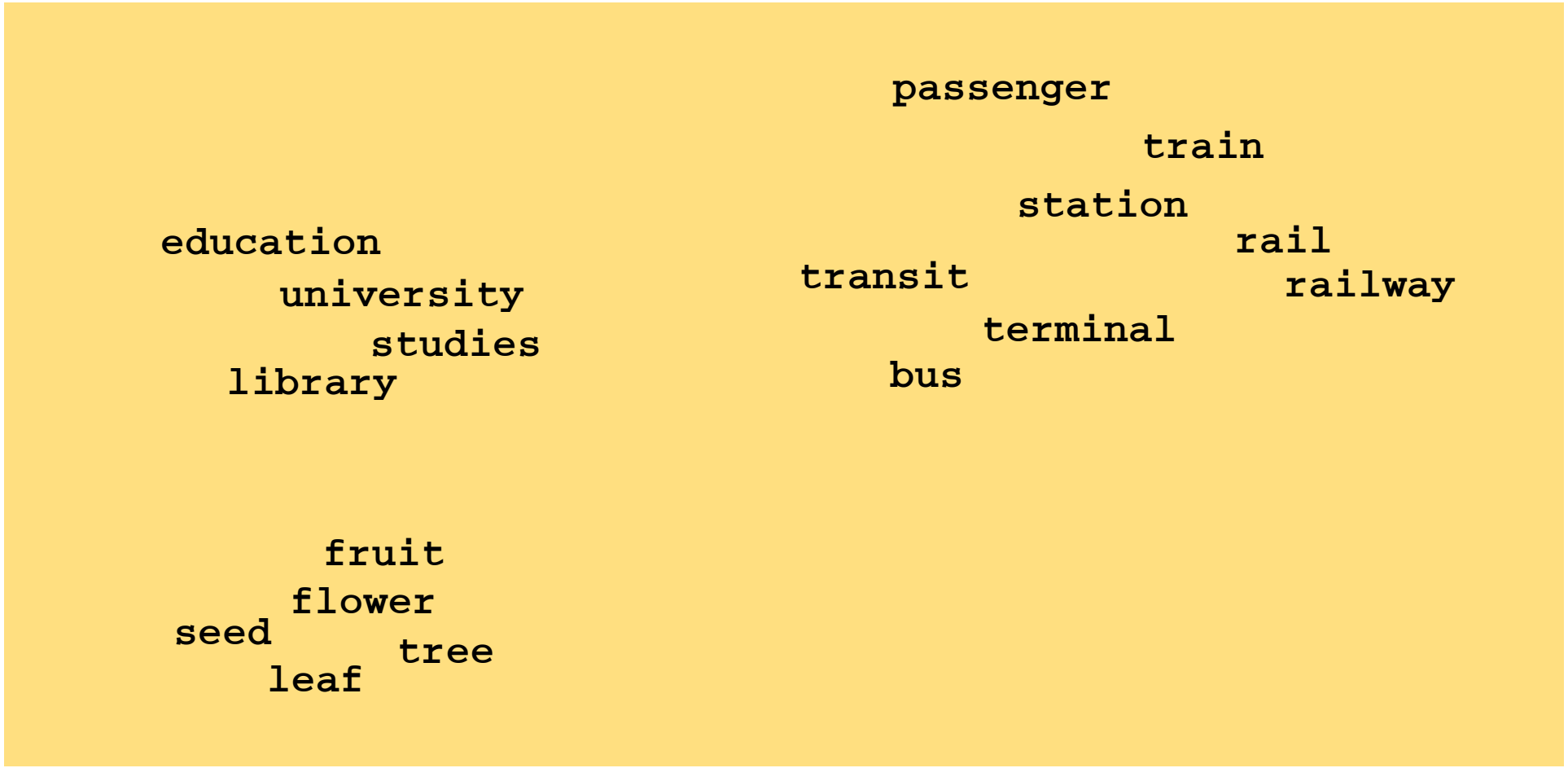
One-hot representation



Problems:

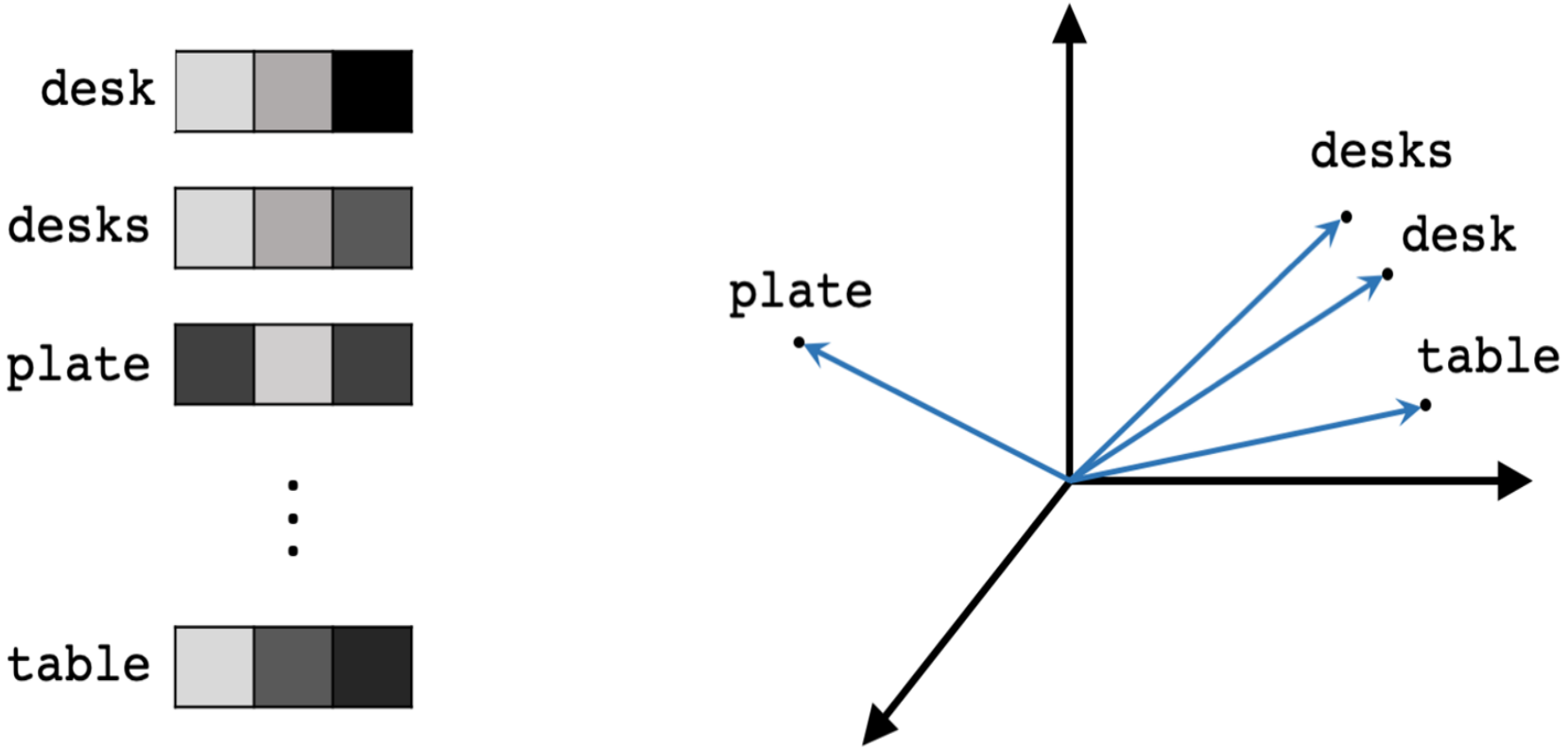
- Every two vectors are orthogonal
- There is no notion of similarity

Vector Space Model

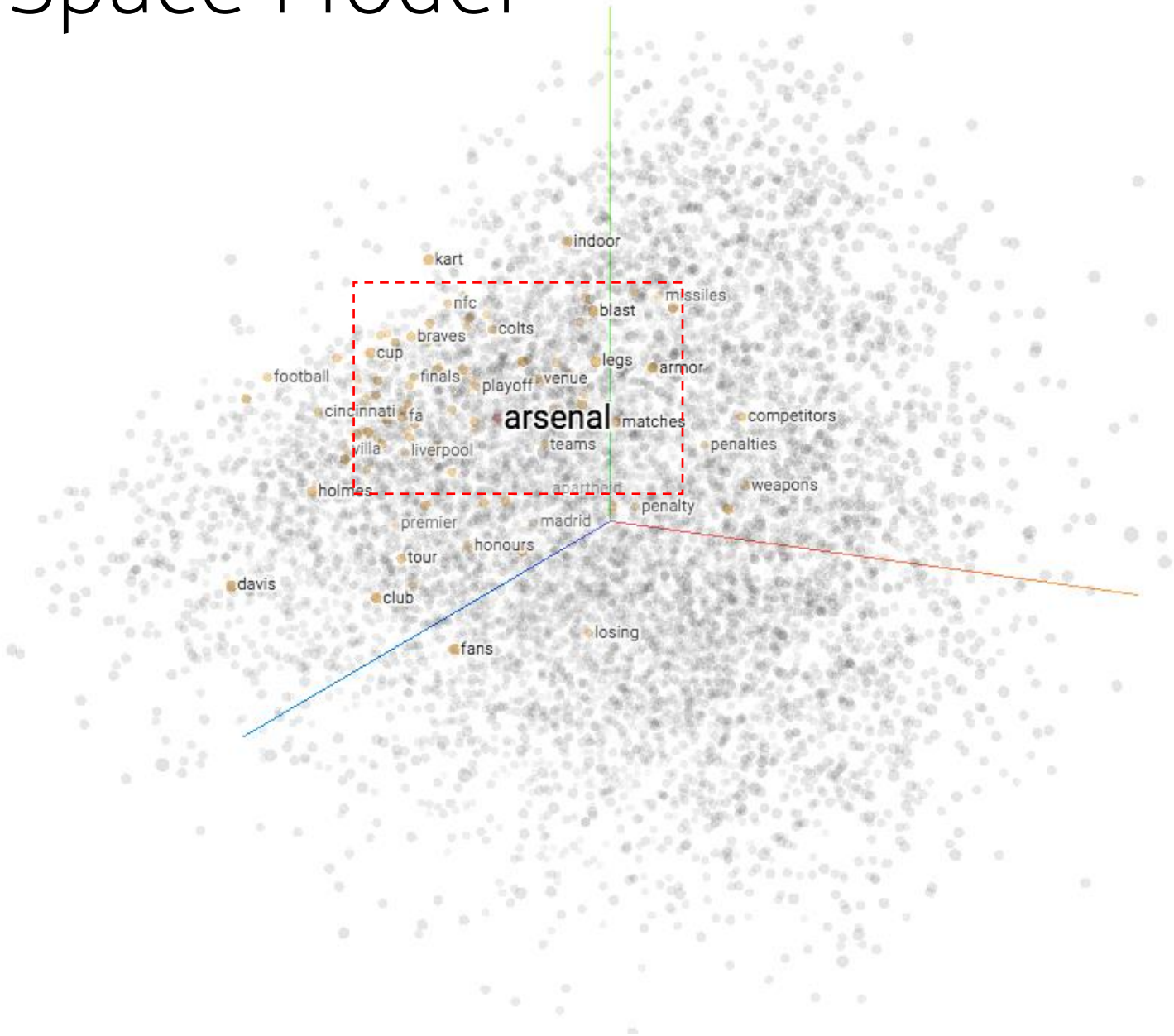


A 2-dimensional semantic space

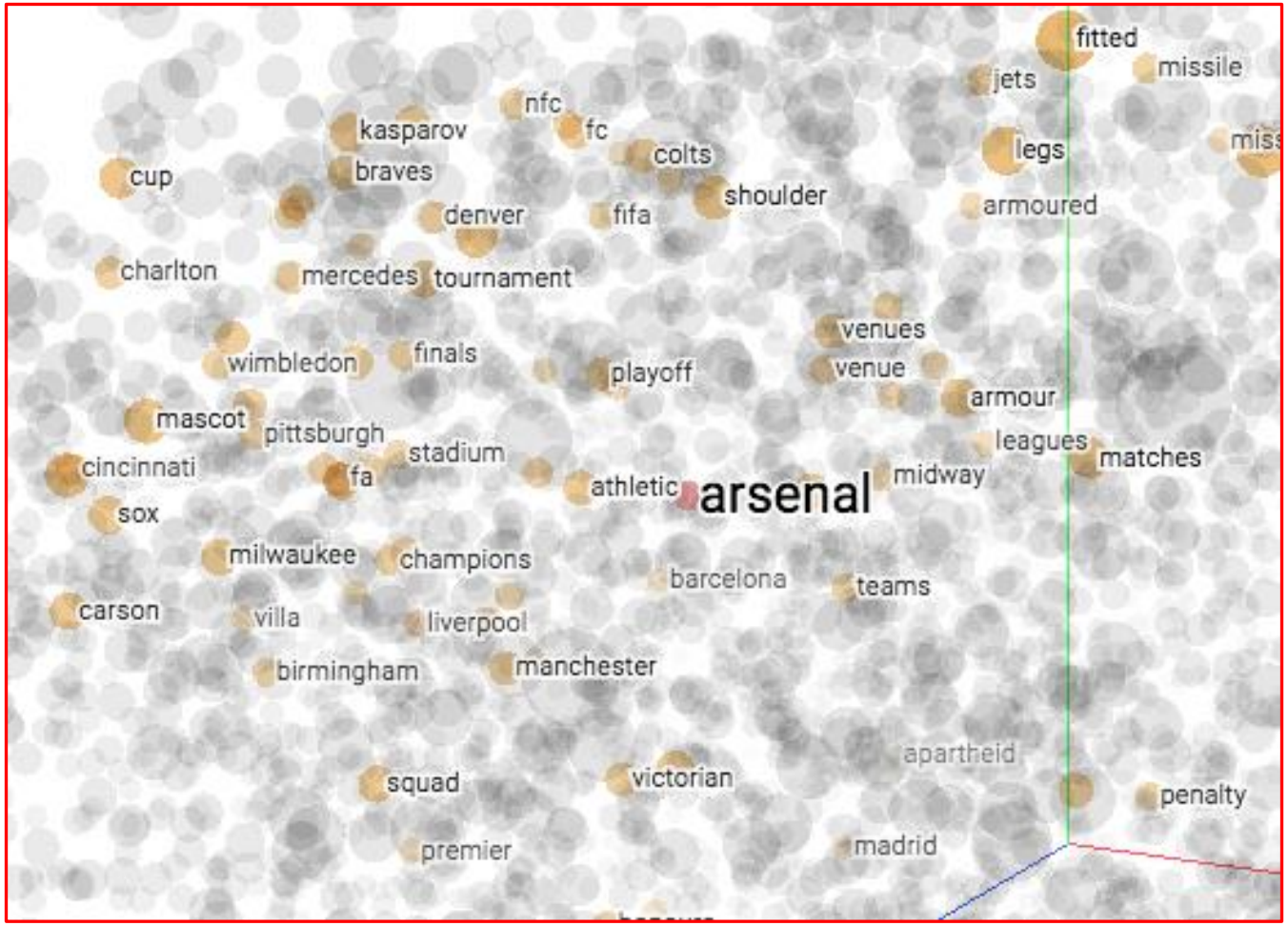
Vector Space Model



Vector Space Model

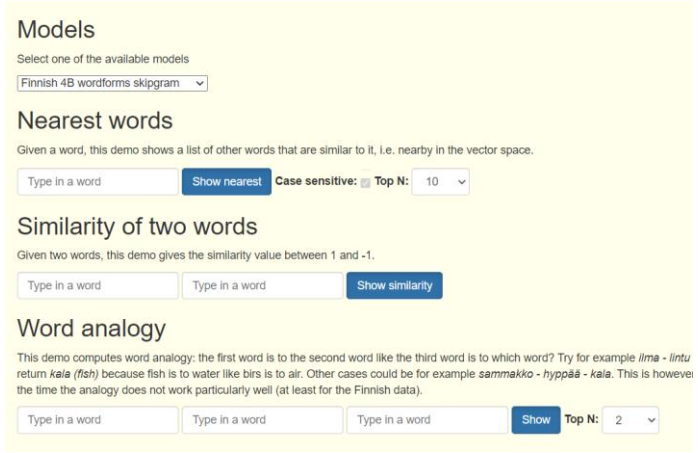


Vector Space Model



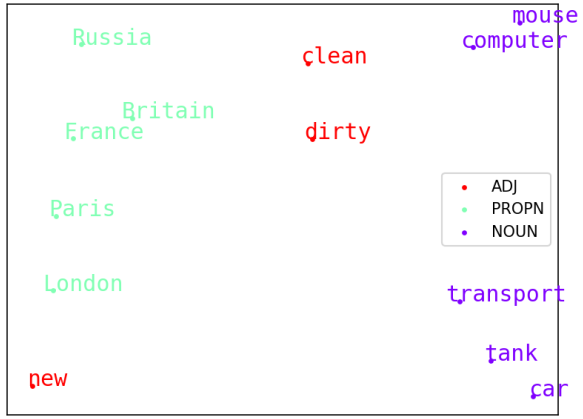
Some demos

https://projector.tensorflow.org/



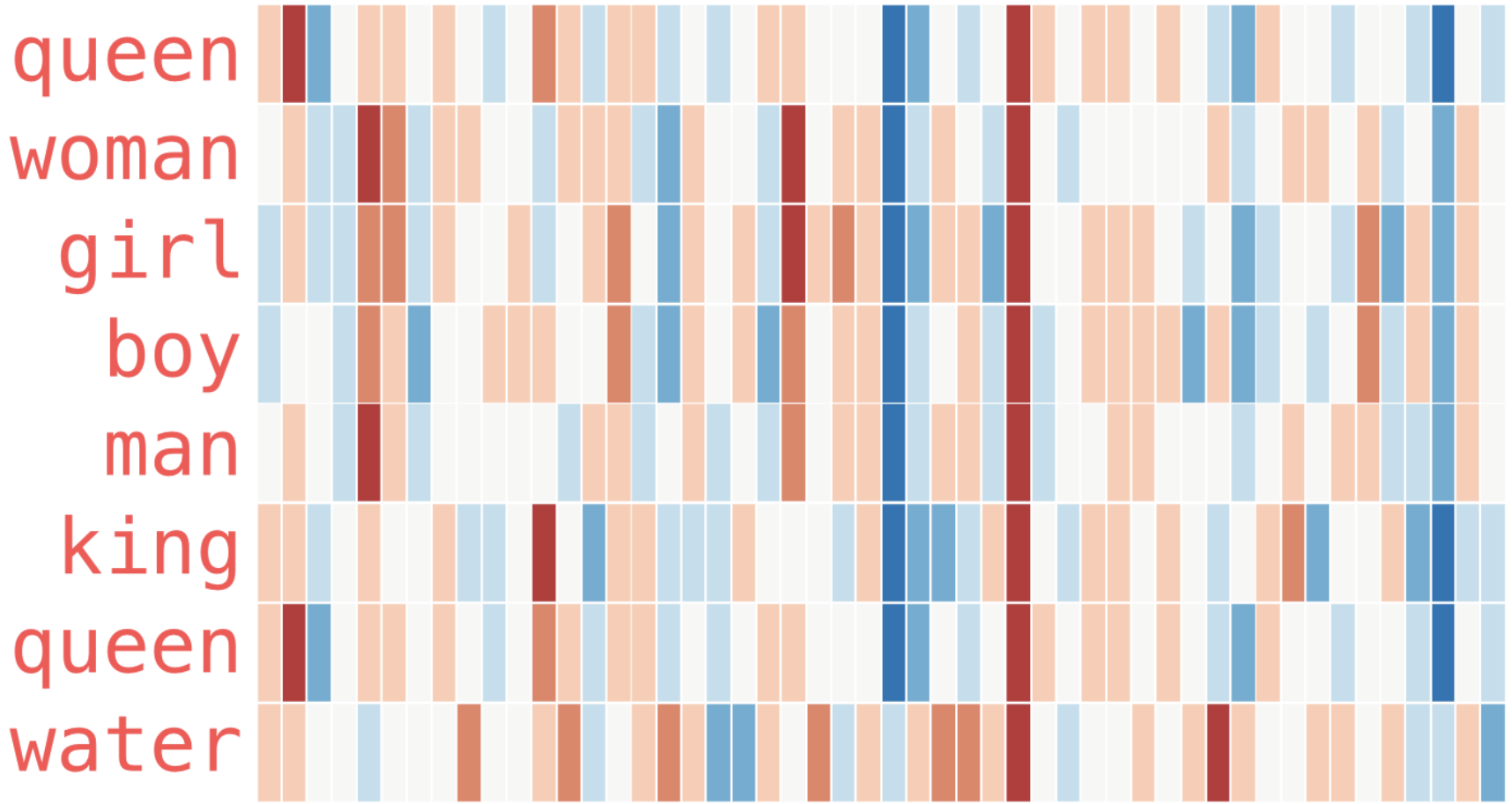
http://bionlp-www.utu.fi/wv_demo/

http://vectors.nlpl.eu/explore/embeddings/en/#



Representation for images

https://cs.stanford.edu/people/karpathy/cnnembed/



From Jay Alammar's blog

Distributional semantics

You shall know a word by the company it keeps

John R. Firth

The Colosseum is still an iconic symbol of Imperial Rome

built of travertine limestone, the Colosseum is

The Colosseum could hold an estimated 80,000 spectators

During the early days of the Colosseum, ancient writers recorded

Co-occurrence vectors

- Example corpus:
  - I like deep learning
  - I like NLP
  - I enjoy flying

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Stanford's CS224N

Co-occurrence vectors

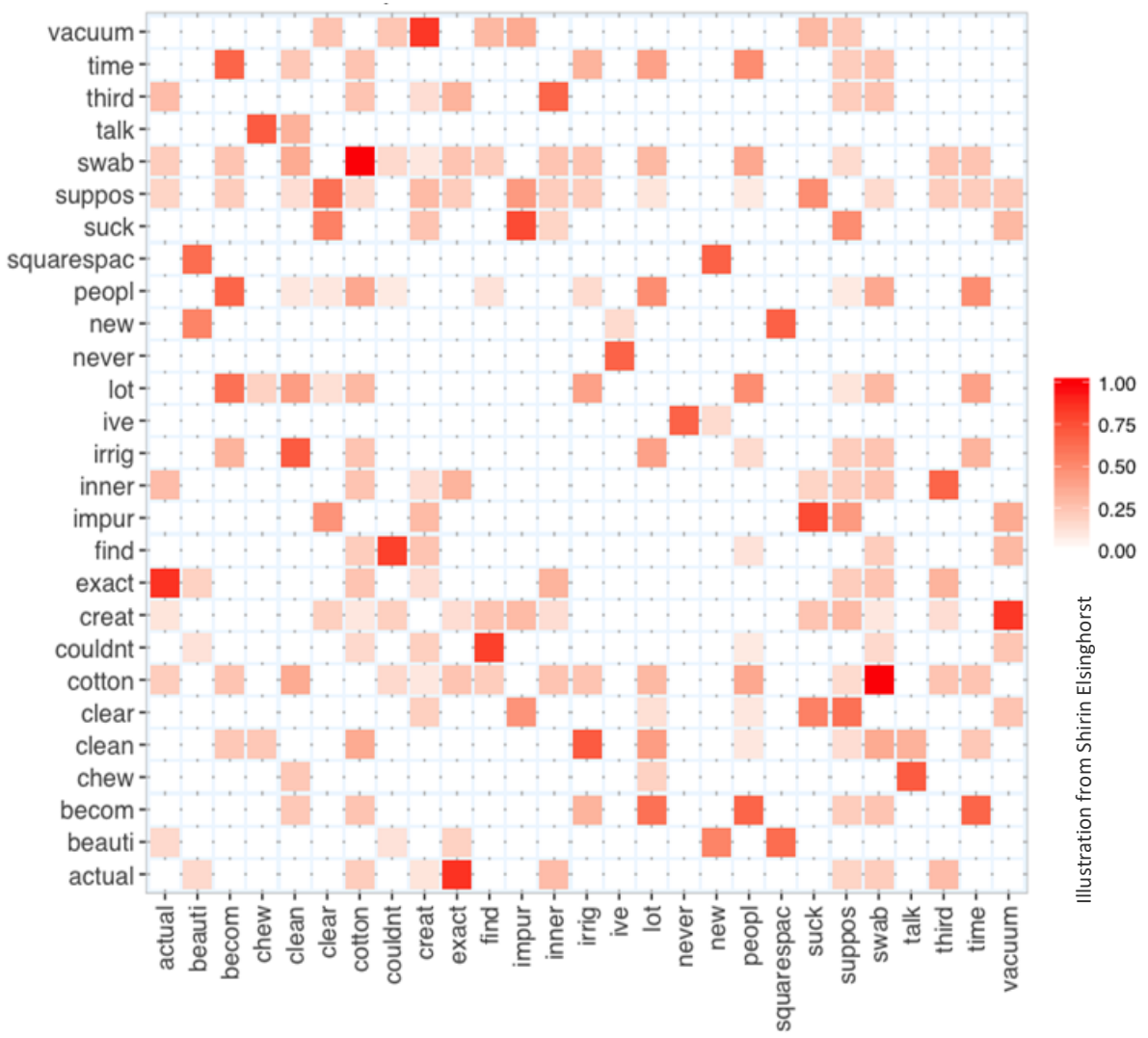
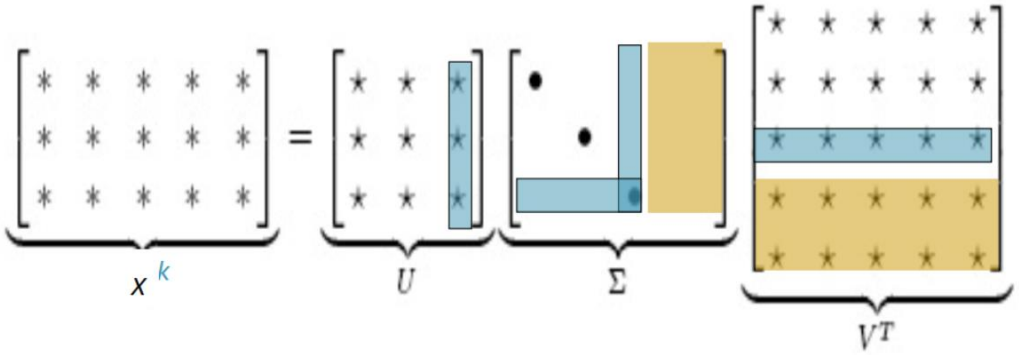


Illustration from Shirin Elsinghorst

Dimensionality reduction

Singular Value Decomposition of co-occurrence matrix X

Factorizes X into $U\Sigma V^T$, where U and V are orthonormal



Retain only *k* singular values, in order to generalize.

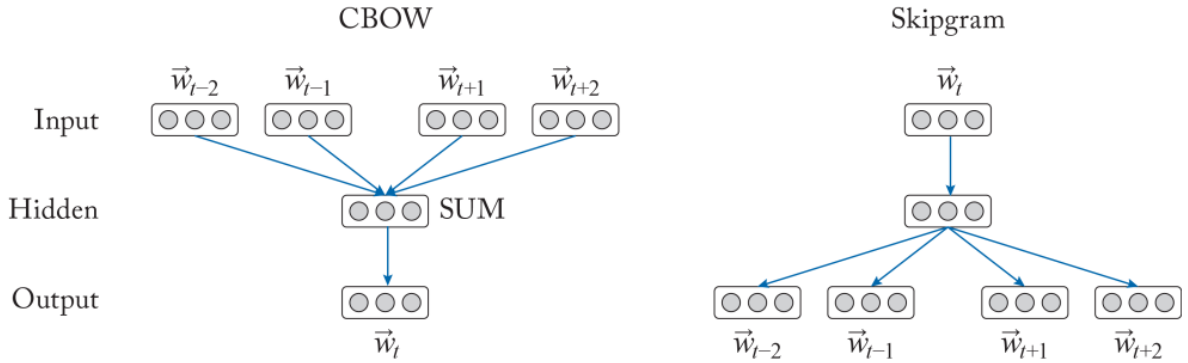
$\hat{X}$ is the best rank *k* approximation to X , in terms of least squares.

Classic linear algebra result. Expensive to compute for large matrices

Stanford's CS224N

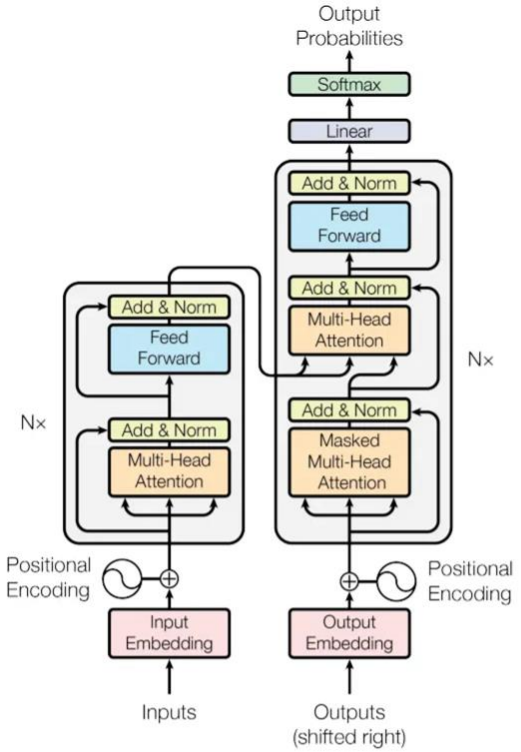
Evolution path of (word) representations

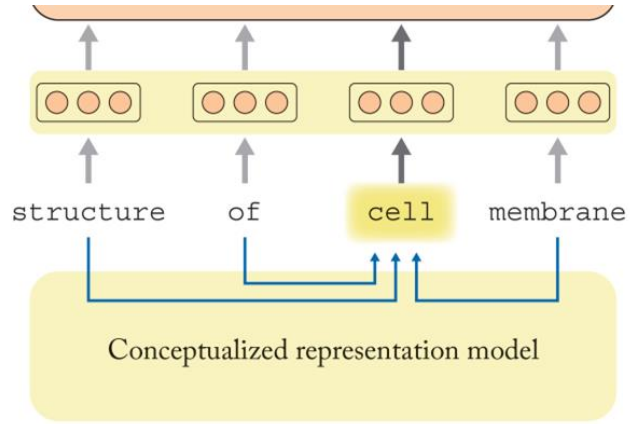
- Co-occurrence vectors + dimensionality reduction (count-based representations)
- Word embeddings
  - Word2vec (and its derivatives)
  - GloVe



Evolution path of (word) representations

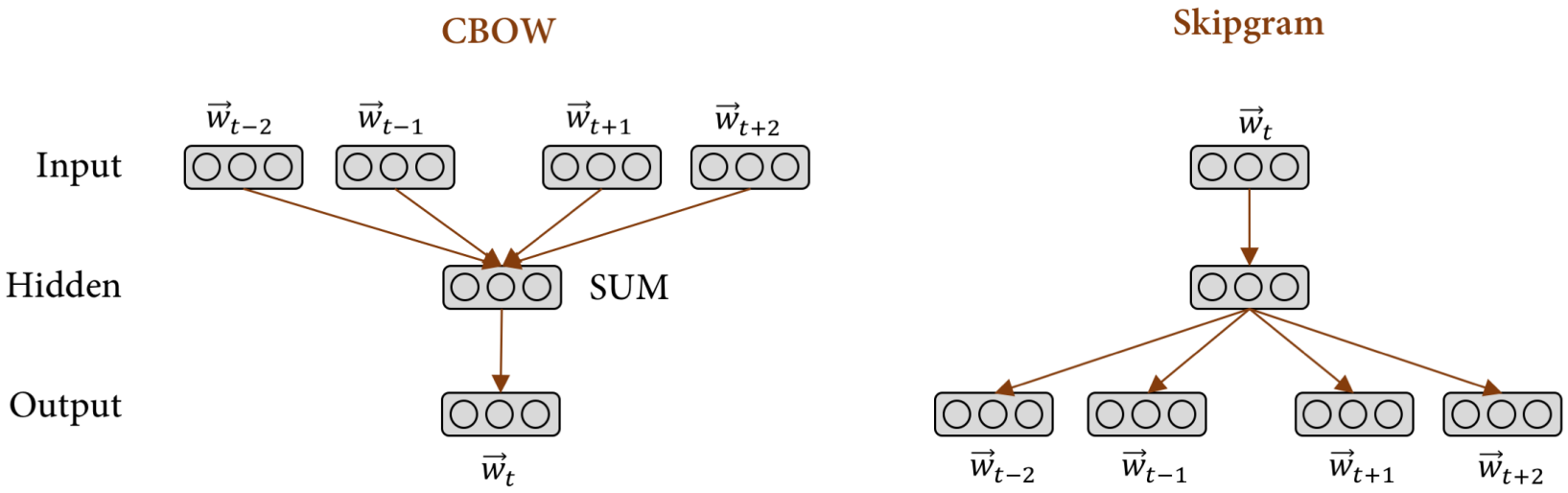
- Co-occurrence vectors + dimensionality reduction (count-based representations)
- Word embeddings
  - Word2vec (and its derivatives)
  - GloVe
- Contextualised embeddings
  - ELMo (LSTMs)
  - Transformer-based
    - BERT (and its derivatives)





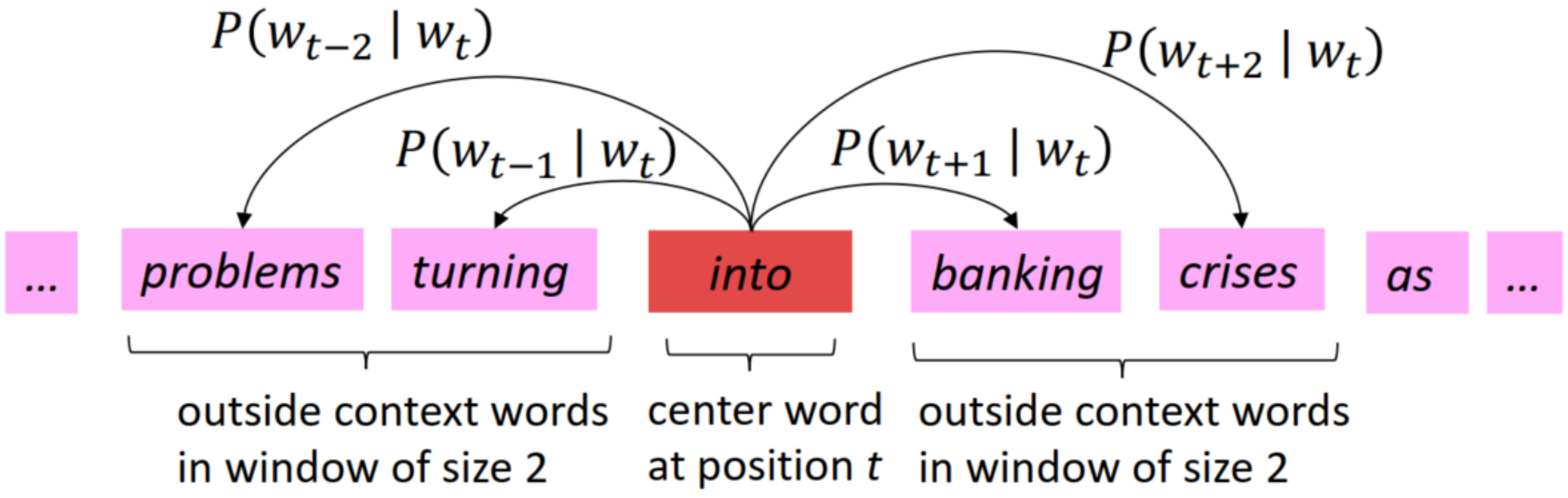
Word2vec

Mikolov et al (2013)



Word2vec

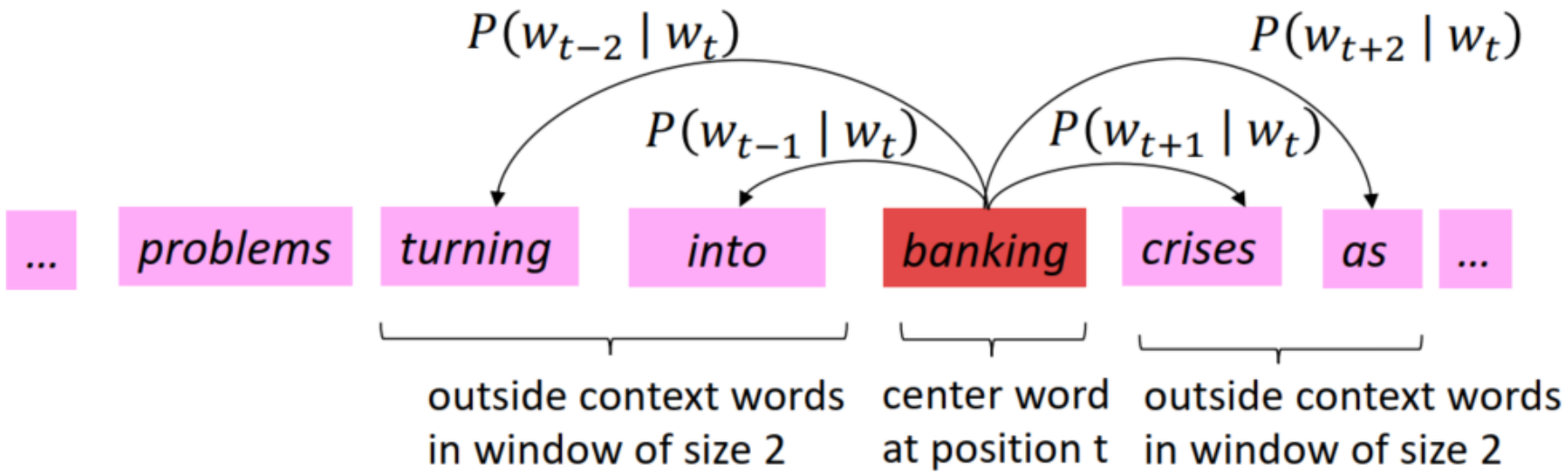
Example windows and process for computing $P(w_{t+j} \mid w_t)$



Slide from Stanford's CS224N

Word2vec

Example windows and process for computing $P(w_{t+j} \mid w_t)$



Slide from Stanford's CS224N

Word2vec: objective function

For each position $t = 1, \ldots, T$, predict context words within a window of fixed size m, given center word w_t. Data likelihood:

$$\text{Likelihood} = L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \neq 0}} P(w_{t+j} \mid w_t; \theta)$$

θ is all variables to be optimized

sometimes called a *cost* or *loss* function

The objective function $J(\theta)$ is the (average) negative log likelihood:

$$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \neq 0}} \log P(w_{t+j} \mid w_t; \theta)$$

Minimizing objective function $\Leftrightarrow$ Maximizing predictive accuracy

Slide from Stanford's CS224N

Word2vec: objective function

- We want to minimize the objective function:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\leq j\leq m \\ j\neq 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$

- **Question:** How to calculate $P\left(w_{t+j} \mid w_t; \theta\right)$?
- **Answer:** We will *use two* vectors per word *w*:
  - v_w when *w* is a center word
  - u_w when *w* is a context word
- Then for a center word *c* and a context word *o*:

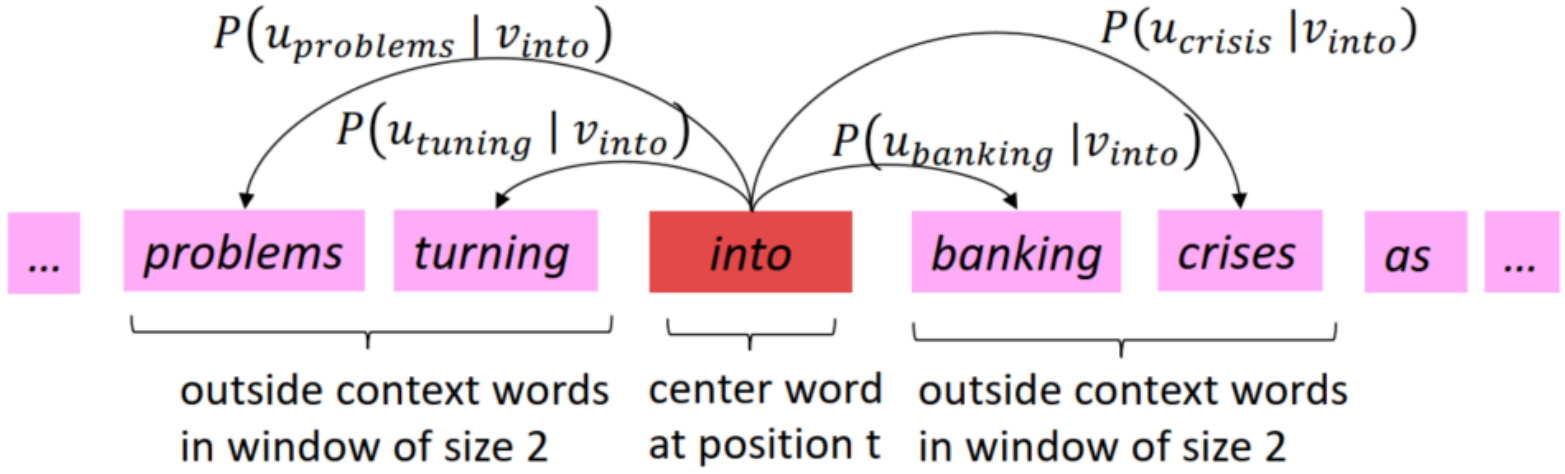
$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w\in V}\exp(u_w^T v_c)}$$

Slide from Stanford's CS224N

Word2vec: example

- Example windows and process for computing $P(w_{t+j} \mid w_t)$
- $P(u_{problems} \mid v_{into})$ short for $P(problems \mid into\ ; u_{problems}, v_{into}, \theta)$

All words vectors θ appear in denominator



Slide from Stanford's CS224N

Word2vec: prediction function

② Exponentiation makes anything positive

① Dot product compares similarity of *o* and *c*.
$u^T v = u.v = \sum_{i=1}^{n} u_i v_i$
Larger dot product = larger probability

$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

③ Normalize over entire vocabulary to give probability distribution

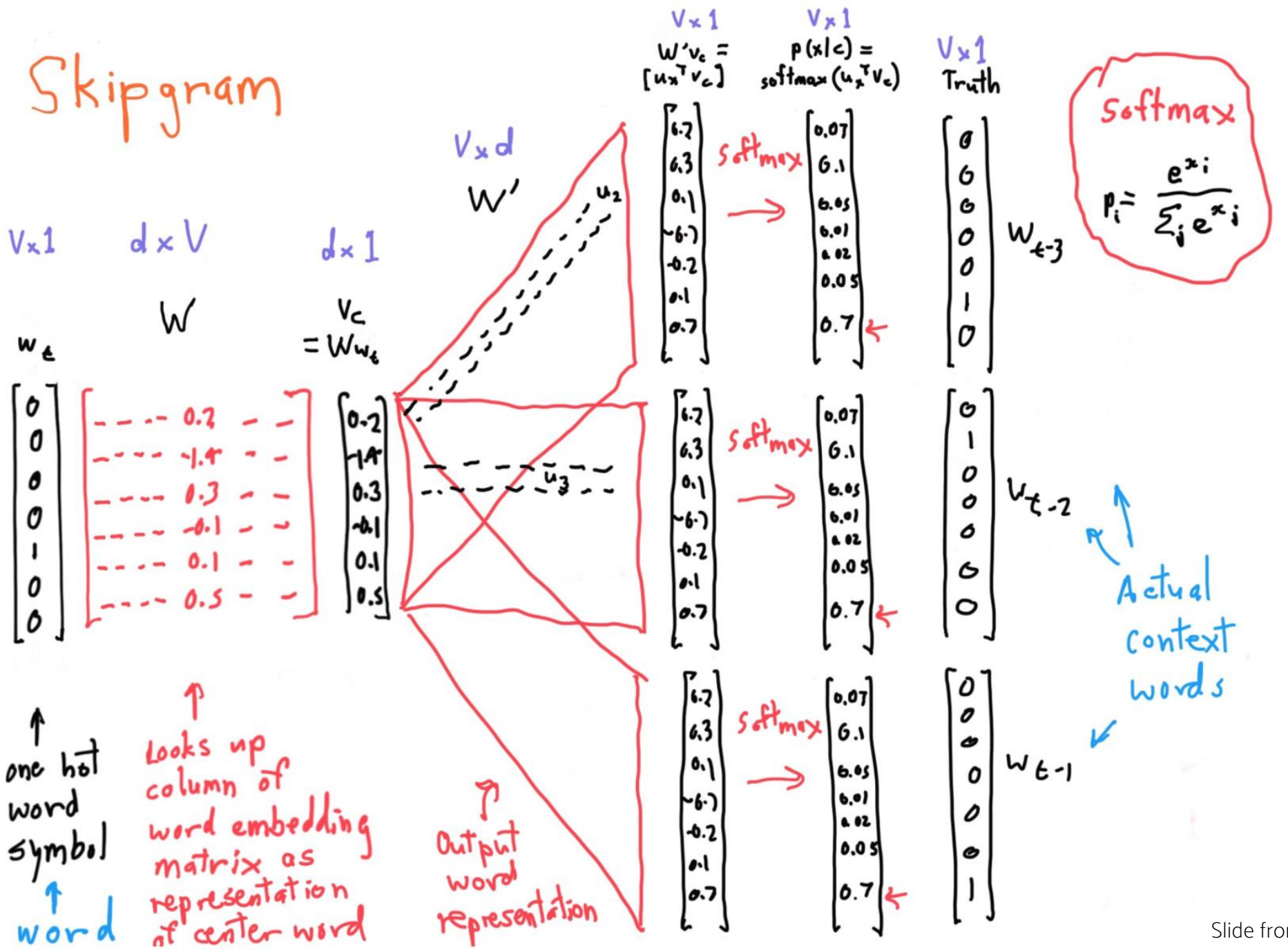
- This is an example of the **softmax function** $\mathbb{R}^n \to (0,1)^n$ ← Open region

$$\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)} = p_i$$

- The softmax function maps arbitrary values x_i to a probability distribution p_i
  - "max" because amplifies probability of largest x_i
  - "soft" because still assigns some probability to smaller x_i
  - Frequently used in Deep Learning

But sort of a weird name because it returns a distribution!

Slide from Stanford's CS224N



Slide from Stanford's CS224N

Gradients for Word2vec

1. $|W|$ be the number of words in the vocab
2. y and $\hat{y}$ be column vectors of shape $|W|$ x 1
3. u_i and v_j be the column vectors of shape D X 1 (D = dimension of embeddings)
4. y be the one-hot encoded column vector of shape $|W|$ x 1
5. $\hat{y}$ be the softmax prediction column vector of shape $|W|$ x 1
6. $\hat{y}_i = P(i|c) = \frac{exp(u_i^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$
7. Cross entropy loss: $J = -\sum_{i=1}^{W} y_i log(\hat{y}_i)$
8. $U = [u_1, u_2, \ldots, u_k, \ldots u_W]$ be a matrix composed of u_k column vectors.

Gradients for Word2vec

Now, we can write

$$J = -\sum_{i=1}^{W} y_i log(\frac{exp(u_i^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)})$$

Simplifying,

$$J = -\sum_{i=1}^{W} y_i [u_i^T v_c - log(\sum_{w=1}^{W} exp(u_w^T v_c))]$$

Gradients for Word2vec

$$J = -\sum_{i=1}^{W} y_i [u_i^T v_c - log(\sum_{w=1}^{W} exp(u_w^T v_c))]$$

$$J = -y_k [u_k^T v_c - log(\sum_{w=1}^{W} exp(u_w^T v_c))]$$

Note: above y_k is 1.

Solving for $\frac{\partial J}{\partial v_c}$:

$$\frac{\partial J}{\partial v_c} = -[u_k - \frac{\sum_{w=1}^{W} exp(u_w^T v_c) u_w}{\sum_{x=1}^{W} exp(u_x^T v_c)}]$$

Which can be re-arranged as:

$$\frac{\partial J}{\partial v_c} = \sum_{w=1}^{W} (\frac{exp(u_w^T v_c)}{\sum_{x=1}^{W} exp(u_x^T v_c)} u_w) - u_k$$

Gradients for Word2vec

$$\frac{\partial J}{\partial v_c} = \sum_{w=1}^{W}\left(\frac{exp(u_w^T v_c)}{\sum_{x=1}^{W} exp(u_x^T v_c)} u_w\right) - u_k$$

$$\frac{\partial J}{\partial v_c} = \sum_{w=1}^{W}(\hat{y}_w u_w) - u_k$$

6. $\hat{y}_i = P(i|c) = \frac{exp(u_i^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$

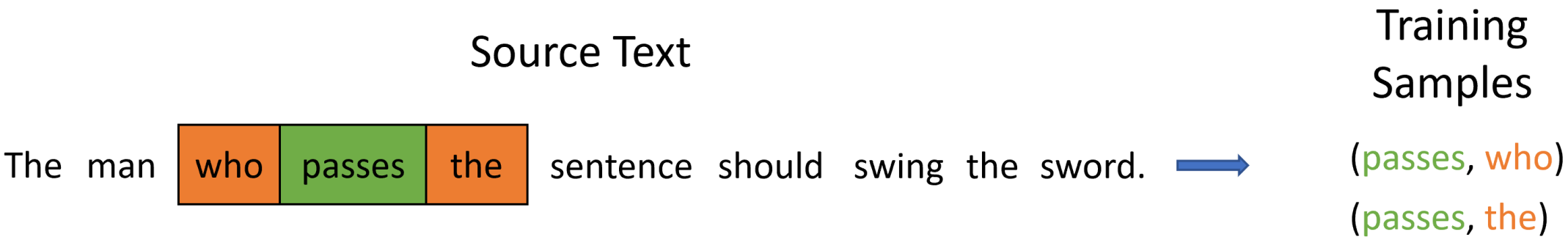
Now let's see how this can be written in Matrix notation.Note that:

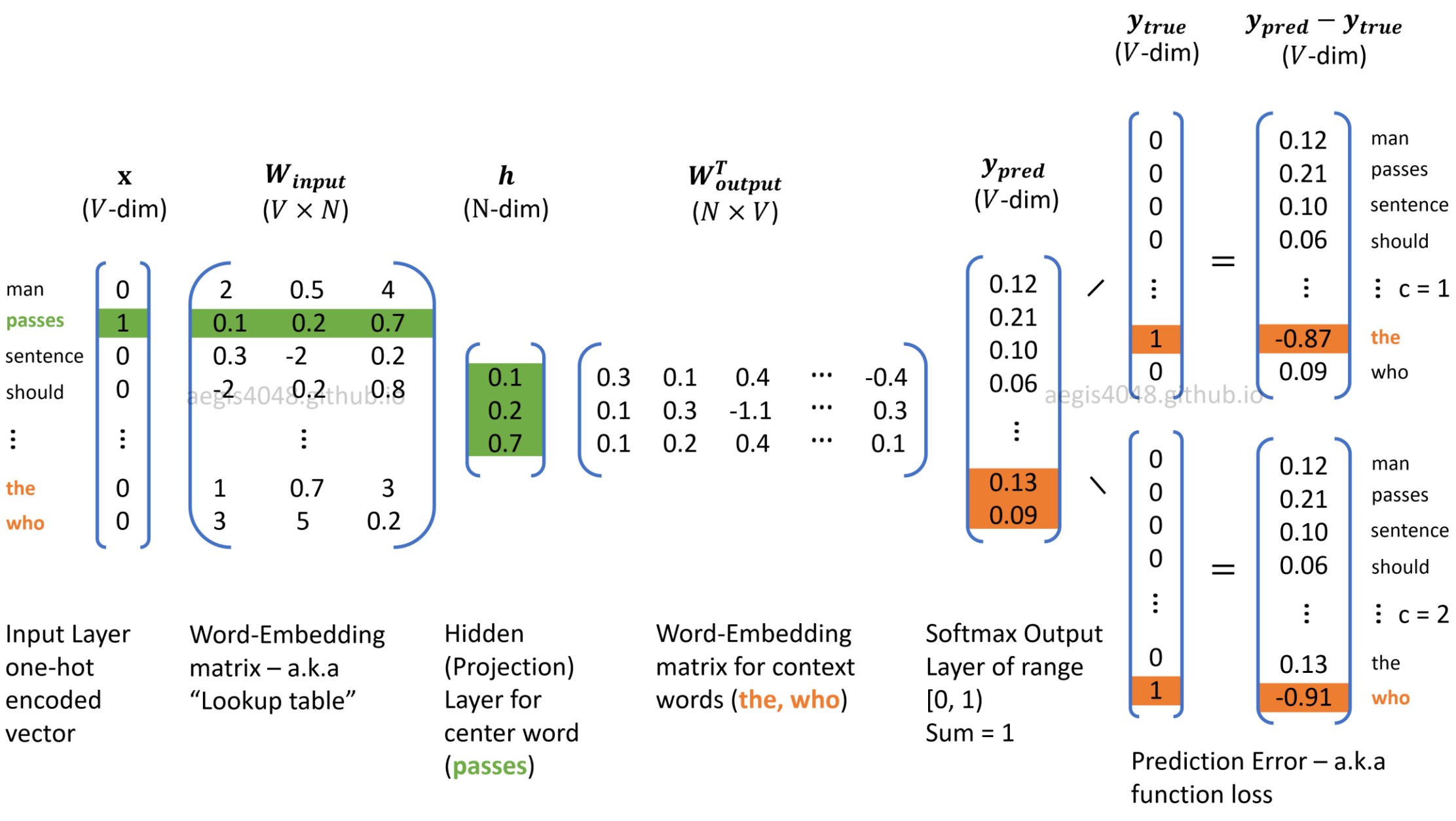
1. u_k can be written as Matrix vector multiplication: $U.y$
2. And $\sum_{w=1}^{W}(\hat{y}_w u_w)$ is a linear transformation of vectors u_w in U scaled by $\hat{y}_w$ respectively. This again can be written as $U.\hat{y}$

So the whole thing can be succinctly written as:

$$U[\hat{y} - y]$$

Updating weights in Skipgram – an example




$\mathbf{x}$
(V-dim)
$\boldsymbol{W_{input}}$
($V \times N$)
$\boldsymbol{h}$
(N-dim)
$\boldsymbol{W^T_{output}}$
($N \times V$)
$\boldsymbol{y_{pred}}$
(V-dim)
$\boldsymbol{y_{true}}$
(V-dim)
$\boldsymbol{y_{pred} - y_{true}}$
(V-dim)
man
passes
sentence
should
the
who
c = 1
c = 2
Input Layer one-hot encoded vector
Word-Embedding matrix – a.k.a "Lookup table"
Hidden (Projection) Layer for center word (passes)
Word-Embedding matrix for context words (the, who)
Softmax Output Layer of range [0, 1) Sum = 1
Prediction Error – a.k.a function loss

Forward pass

Corpus = **"The man who passes the sentence should swing the sword."**

#	Token	x
0	man	0
1	**passes**	**1**
2	sentence	0
3	should	0
4	swing	0
5	sword	0
6	**the**	**0**
7	**who**	**0**

(1 X 8)

×

W_input		
-0.078	0.018	0.033
0.068	0.170	-0.109
-0.158	-0.081	-0.151
0.150	0.064	0.145
-0.097	-0.055	0.188
0.036	0.071	0.059
0.168	-0.060	-0.058
0.098	0.015	0.096

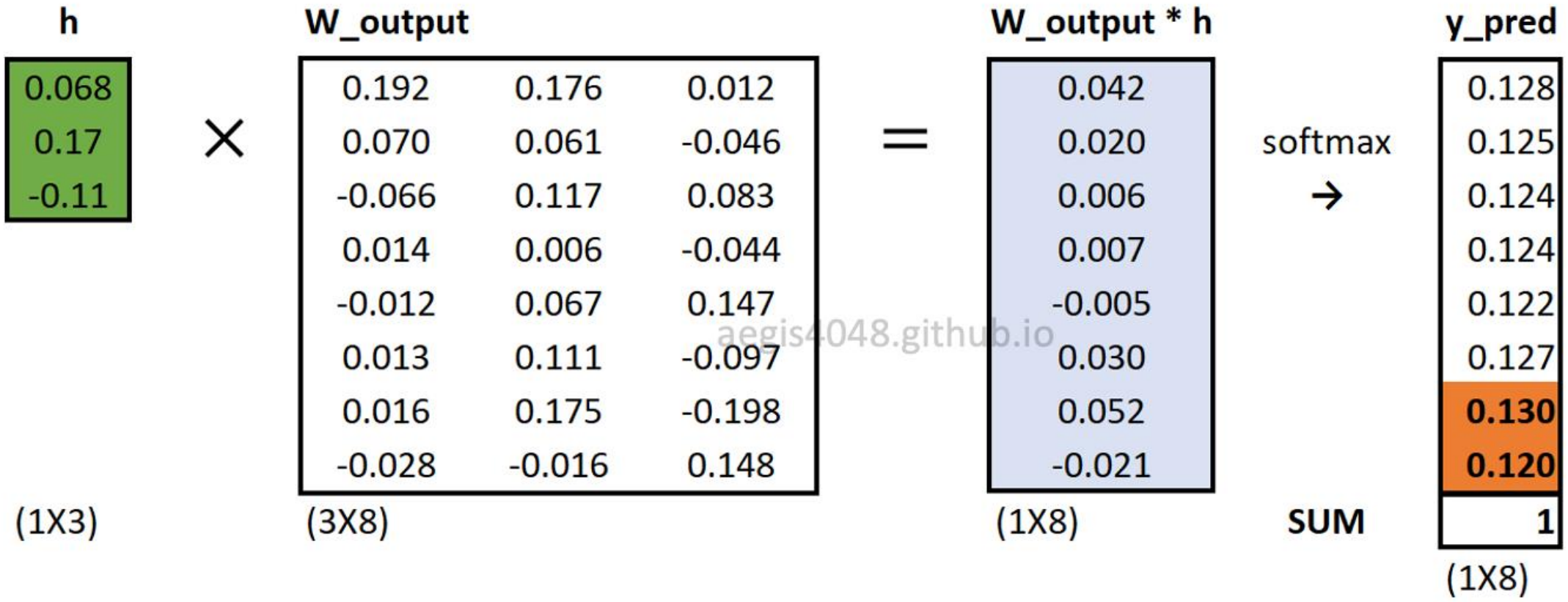
(8X3)

=

h
0.068
0.170
-0.109

(1X3)

Forward pass



Sum of errors

$$\sum_{c=1}^{C} e_c$$

y_pred	Token	y_true	error_"the"
0.128	man	0	0.128
0.125	passes	0	0.125
0.256	sentence	0	0.256
0.124	should	0	0.124
0.122	swing	0	0.122
0.127	sword	0	0.127
0.130	**the**	**1**	**-0.870**
0.120	who	0	0.120
(1X8)		(1X8)	(1X8)

+

y_pred	Token	y_true	error_"who"
0.128	man	0	0.128
0.125	passes	0	0.125
0.256	sentence	0	0.256
0.124	should	0	0.124
0.122	swing	0	0.122
0.127	sword	0	0.127
0.130	the	0	0.130
0.120	**who**	**1**	**-0.880**
(1X8)		(1X8)	(1X8)

=

Sum_error
0.256
0.251
0.513
0.248
0.245
0.253
-0.741
-0.759
(1X8)

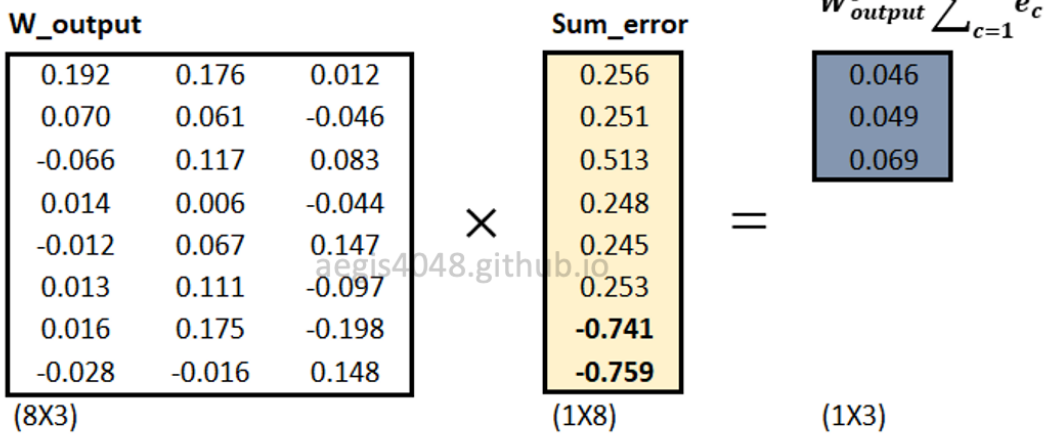
Gradients for output embeddings

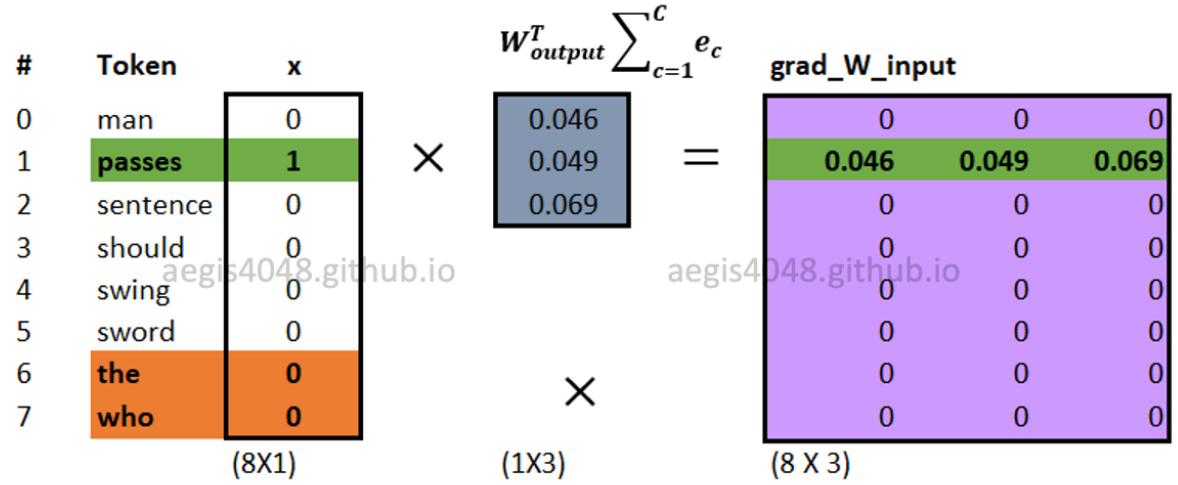
$$\frac{\partial J}{\partial W_{output}} = h \cdot \sum_{c=1}^{C} e_c$$

h		Sum_error		grad_W_output		
0.068	×	0.256	=	0.017	0.044	-0.028
0.17		0.251		0.017	0.043	-0.027
-0.109		0.513		0.035	0.087	-0.056
		0.248		0.017	0.042	-0.027
		0.245		0.017	0.042	-0.027
		0.253		0.017	0.043	-0.028
		-0.741		-0.050	-0.126	0.081
		-0.759		-0.052	-0.129	0.083
(1X3)		(1X8)		(8 X 3)		

Gradients for input embeddings

$$\frac{\partial J}{\partial W_{input}} = x \cdot (W_{output}^T \sum_{c=1}^{C} e_c)$$





Updating weight matrices

W_input (old)

-0.078	0.018	0.033
0.068	0.170	-0.109
-0.158	-0.081	-0.151
0.150	0.064	0.145
-0.097	-0.055	0.188
0.036	0.071	0.059
0.168	-0.060	-0.058
0.098	0.015	0.096

(8X3)

— **Learning R.** 0.05 ×

grad_W_input

0.000	0.000	0.000
0.046	0.049	0.069
0.000	0.000	0.000
0.000	0.000	0.000
0.000	0.000	0.000
0.000	0.000	0.000
0.000	0.000	0.000
0.000	0.000	0.000

(8 X 3)

= **W_input (new)**

-0.078	0.018	0.033
0.066	**0.168**	**-0.112**
-0.158	-0.081	-0.151
0.150	0.064	0.145
-0.097	-0.055	0.188
0.036	0.071	0.059
0.168	-0.060	-0.058
0.098	0.015	0.096

(8X3)

W_output (old)

0.192	0.176	0.012
0.070	0.061	-0.046
-0.066	0.117	0.083
0.014	0.006	-0.044
-0.012	0.067	0.147
0.013	0.111	-0.097
0.016	0.175	-0.198
-0.028	-0.016	0.148

(8X3)

— **Learning R.** 0.05 ×

grad_W_output

0.017	0.044	-0.028
0.017	0.043	-0.027
0.035	0.087	-0.056
0.017	0.042	-0.027
0.017	0.042	-0.027
0.017	0.043	-0.028
-0.050	-0.126	0.081
-0.052	-0.129	0.083

(8 X 3)

= **W_output (new)**

0.191	**0.174**	**0.013**
0.069	**0.059**	**-0.045**
-0.068	**0.113**	**0.086**
0.013	**0.004**	**-0.043**
-0.013	**0.065**	**0.148**
0.012	**0.109**	**-0.096**
0.019	**0.181**	**-0.202**
-0.025	**-0.010**	**0.144**

(8X3)

Skip-gram with negative sampling

$$J_t(\theta) = \log \sigma\left(u_o^T v_c\right) + \sum_{i=1}^{k} \mathbb{E}_{j \sim P(w)} \left[\log \sigma\left(-u_j^T v_c\right)\right]$$

$$J_{neg-sample}(\boldsymbol{u}_o, \boldsymbol{v}_c, U) = -\log \sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c) - \sum_{k \in \{K \text{ sampled indices}\}} \log \sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c)$$

Evaluate word vectors

Related to general evaluation in NLP: Intrinsic vs. extrinsic

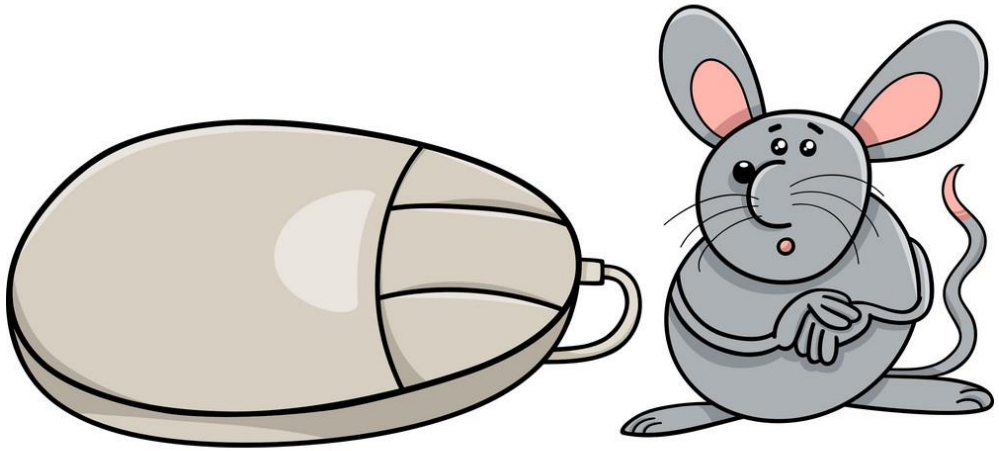
- Intrinsic:
  - Evaluation on a specific/intermediate subtask
  - Fast to compute
  - Helps to understand that system
  - Not clear if really helpful unless correlation to real task is established
- Extrinsic:
  - Evaluation on a real task
  - Can take a long time to compute accuracy
  - Unclear if the subsystem is the problem or its interaction or other subsystems
  - If replacing exactly one subsystem with another improves accuracy à Winning!

Intrinsic evaluation: word similarity

Distances between word vectors, correlation with human judgements

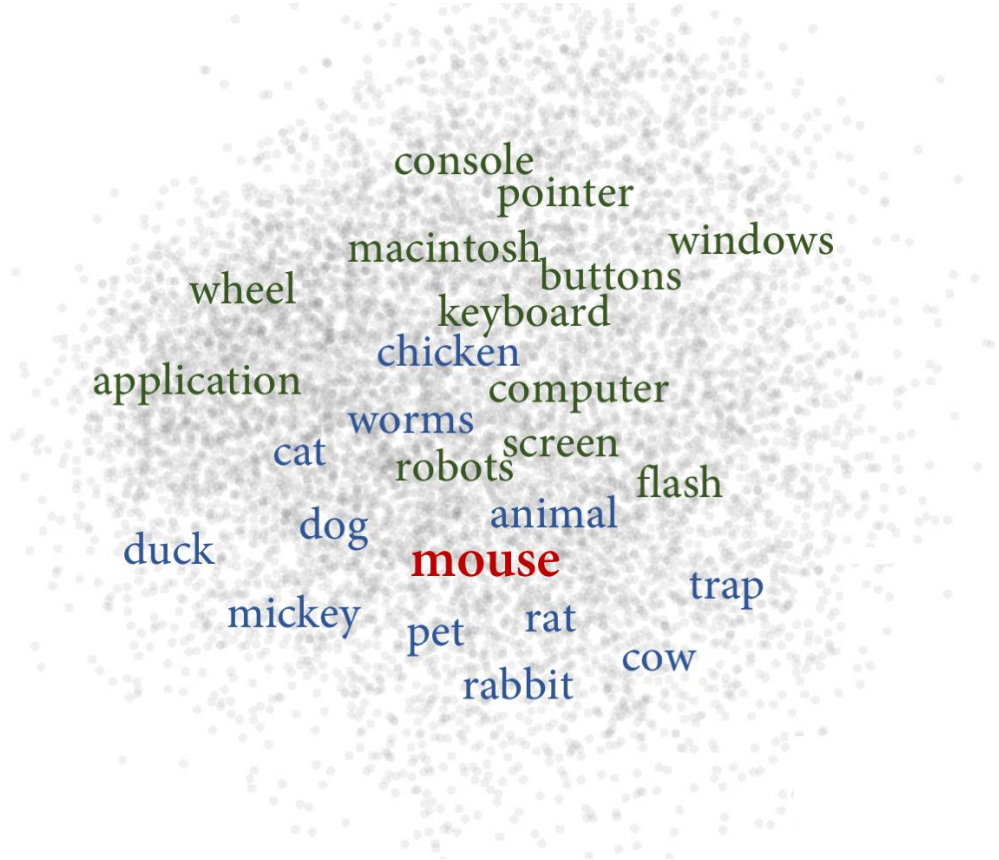
midday	noon	3.94
shore	voyage	1.22
autograph	signature	3.59
automobile	car	3.92
grin	smile	3.46
autograph	shore	0.06
coast	forest	0.85
cemetery	woodland	1.18
car	journey	1.55
automobile	wizard	0.11

Word senses

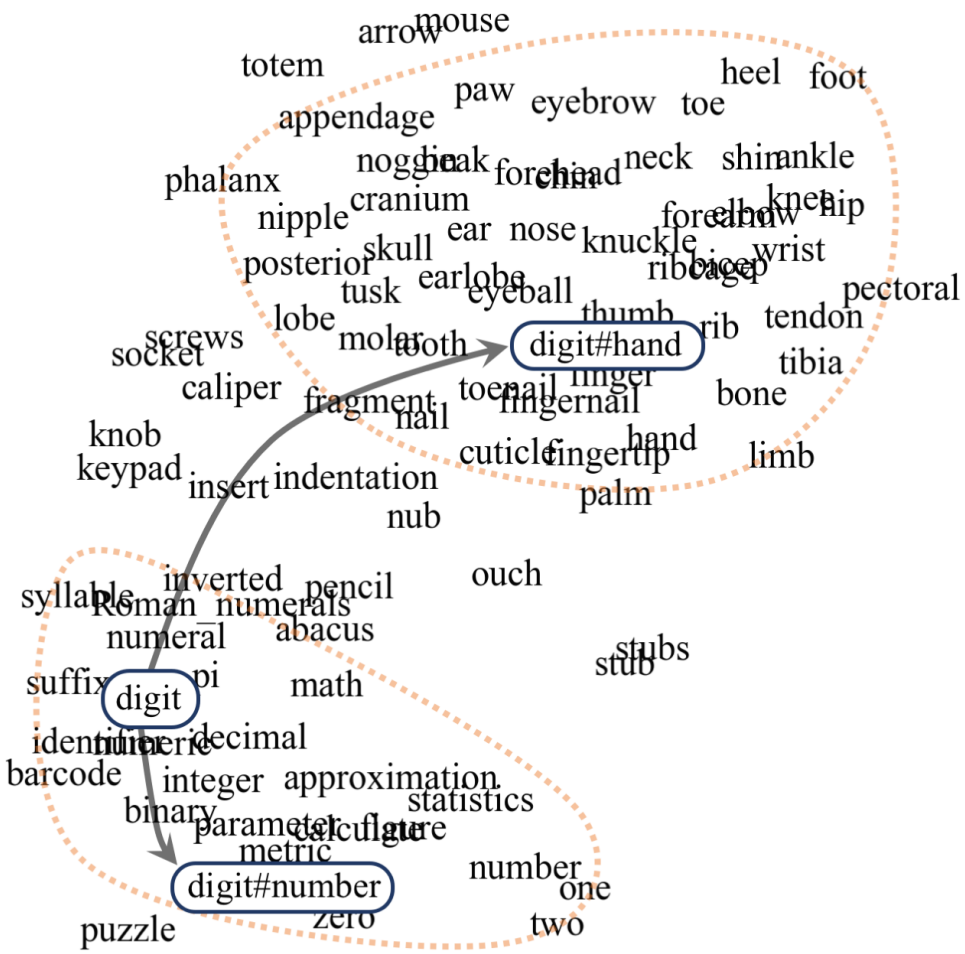


Mouse

Meaning conflation deficiency



Sense embeddings



Distributional vs. distributed representations

- Distributional representations
  - Words are similar if they appear in similar contexts (Harris 1954); distribution of words indicative of usage
  - In contrast: non-distributional representations created from lexical resources such as WordNet, etc.
- Distributed representations
  - Basically, something is represented by a vector of values, each representing activations
  - In contrast: local representations, where represented by a discrete symbol (one-hot vector)

Improving word embeddings

Limitations of word embeddings:

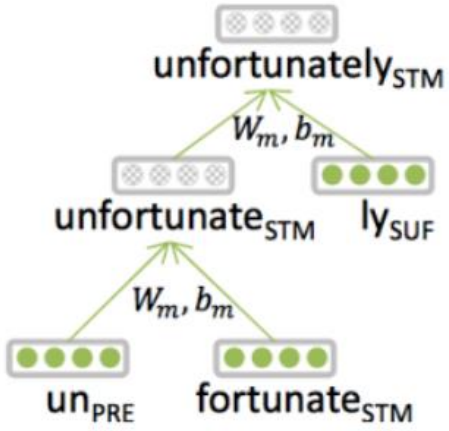
- Sensitive to superficial differences (dog/dogs)
- Not necessarily coordinated with knowledge or across languages
- Not interpretable
- Can encode bias (encode stereotypical gender roles, racial biases)

Improving word embeddings

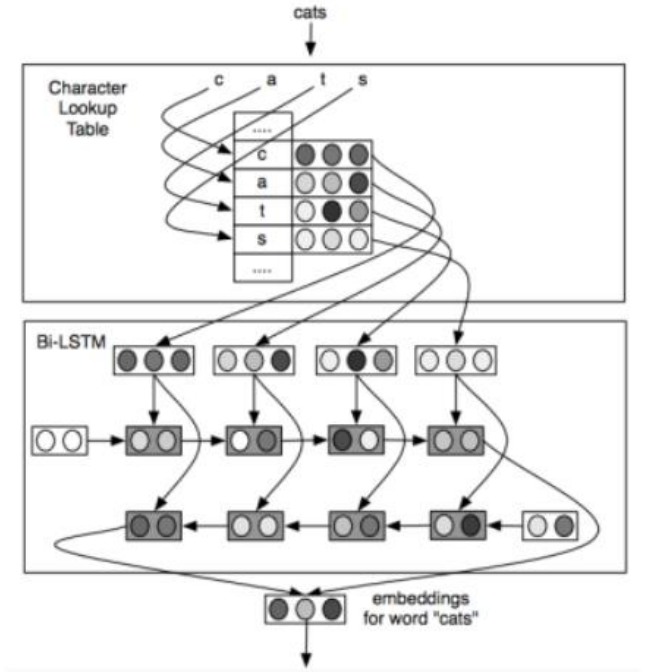
Sub-word embeddings

Can capture sub-word regularities

Morpheme-based
(Luong et al. 2013)

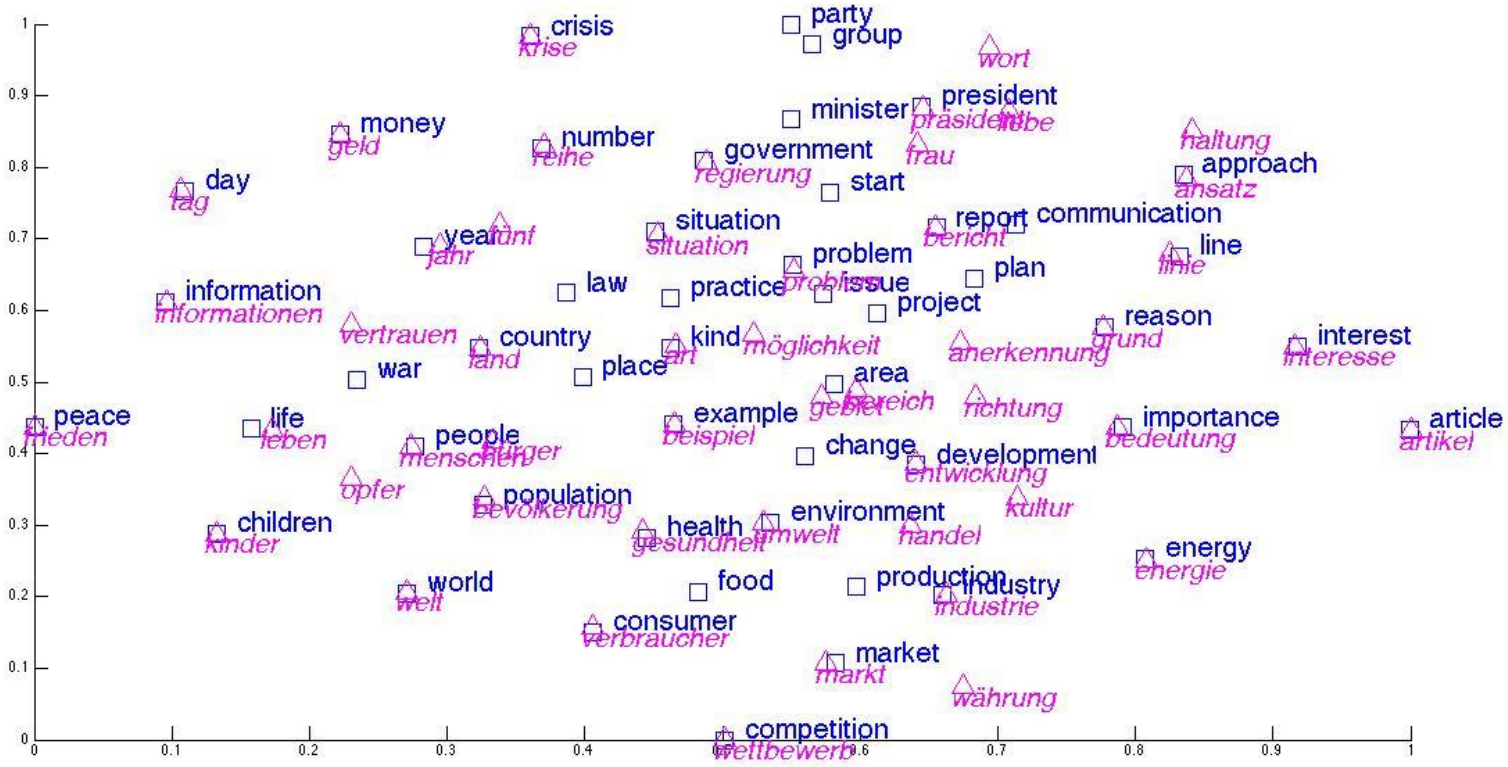


Character-based
(Ling et al. 2015)



Improving word embeddings

Multilingual embeddings

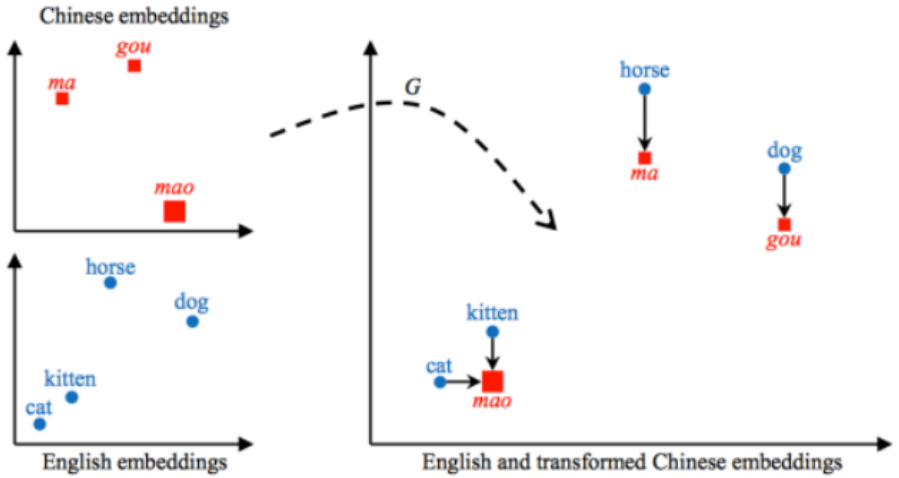


(Luong et al., 2015)

Improving word embeddings

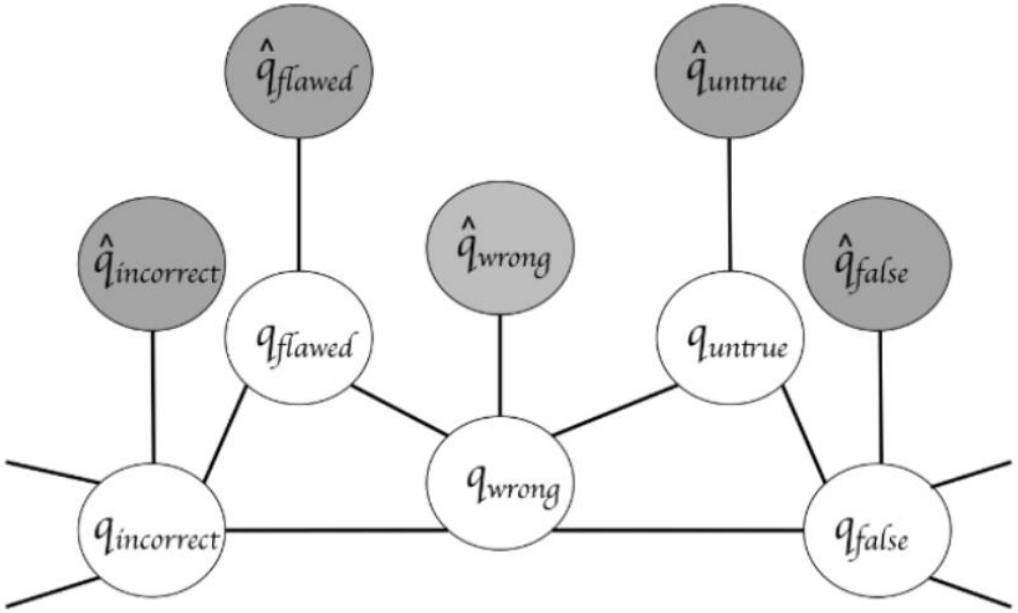
Multilingual embeddings

- In fact we can do it with no dictionary at all!
  - Just use identical words, e.g. the digits (Artexte et al. 2017)
  - Or just match distributions (Zhang et al. 2017)



(Luong et al., 2015)

Retrofitting of Embeddings to Existing Lexicons



$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j) \in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

(Faruqui et al. 2015)

Bias in word embeddings

he	she
maestro, skipper, protege, philosopher, captain, architect, financier, warrior, broadcaster, magician	homemaker, nurse, receptionist, librarian, socialite, hairdresser, nanny, bookkeeper, stylist, housekeeper

Gender stereotype analogies	Gender appropriate analogies
sewing-carpentry, registered nurse-physician, housewife-shopkeeper, nurse-surgeon, interior designer-architect, softball-baseball, blond-burly, feminism-conservatism, cosmetics-pharmaceuticals, giggle-chuckle, vocalist-guitarist, petite-lanky sassy-snappy, diva-superstar, charming-affable, volleyball-football, cupcakes-pizzas, lovely-brilliant	queen-king, sister-brother, mother-father, waitress-waiter, ovarian cancer-prostate cancer, convent-monastery

Bolukbasi et al. [2016]

Bias in NLP applications

Sentence

The doctor ran to the emergency room to see [MASK] patient.

Mask 1

Prediction	Score
The doctor ran to the emergency room to see **his** patient .	38.3%
The doctor ran to the emergency room to see **the** patient .	36.9%
The doctor ran to the emergency room to see **another** patient .	8.1%
The doctor ran to the emergency room to see **a** patient .	7.3%
The doctor ran to the emergency room to see **her** patient .	6%

AllenNLP

Bias in NLP applications

Sentence

the programmer finished [MASK] work

Mask 1

Prediction	Score
the programmer finished **his** work	54.4%
the programmer finished **the** work	38.3%
the programmer finished **its** work	2%
the programmer finished **her** work	1.3%
the programmer finished **their** work	0.7%

Sentence

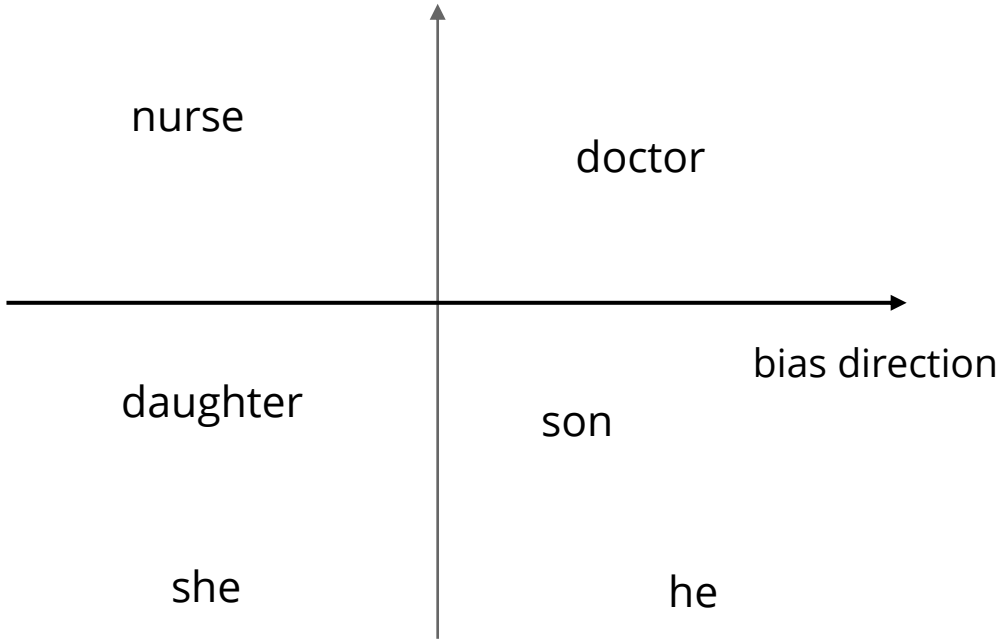
the nurse finished [MASK] work

Mask 1

Prediction	Score
the nurse finished **her** work	77.4%
the nurse finished **the** work	14.4%
the nurse finished **his** work	4.7%
the nurse finished **its** work	0.9%
the nurse finished **to** work	0.5%

AllenNLP

Debiasing word embeddings



Bolukbasi et al. [2016]

Questions?