DISTRIBUTIONAL SEMANTICS AND COMPOSITIONALITY

Corina Dima

25rd April 2019

REVIEW

- use the co-occurrence information in large corpora to create distributional representations for words

	iced	(to) drink	owner	in
cappuccino	6	2	0	3
espresso	1	1	0	4
cat	0	1	4	3
latte	6	5	0	4
leaf	0	0	0	5

COMPOSITIONALITY

$$f\left(\begin{bmatrix}0.3\\0.1\\0.7\end{bmatrix}u, \begin{bmatrix}0.5\\0.9\\0.1\end{bmatrix}v\right) = \begin{bmatrix}??\\??\\??\end{bmatrix}p$$

What f makes p most similar to w?

wmask

$$p = g(W[u \odot u'; v \odot v''] + b)$$

where $p, u, u', v, v'', b \in \mathbb{R}^n$; $W \in \mathbb{R}^{n \times 2n}$; $g = tanh$

multimatrix

$$p = Wg(W_1[u; v] + b_1; W_2[u; v] + b_2; ...; W_k[u; v] + b_k) + b$$

where $p, u, v, b, b_i \in \mathbb{R}^n$; $W_i \in \mathbb{R}^{n \times 2n}$; $W \in \mathbb{R}^{n \times kn}$; $g = relu$

- Composition models for distributional semantics can extend the vector spaces

- they learn how to build representations for complex words (e.g. ‘apple tree’) and phrases (e.g. ‘black car’) from the representations of individual words.

WHY MODEL COMPOSITION?

WHY MODEL COMPOSITION?

- when using a search engine to look for job ads
  - programmer
  - software developer

WHY MODEL COMPOSITION?

- when using a search engine to look for job ads
  - programmer
  - software developer

WHY MODEL COMPOSITION?

- when using a search engine to look for job ads
  - programmer
  - software developer

- when shopping online
  - blue jacket
  - dark-blue coat

WHY MODEL COMPOSITION? (2)

Handpuppentheateraufführung

WHY MODEL COMPOSITION? (2)

Handpuppentheateraufführung

Hand | Puppen | Theater | Aufführung

WHY MODEL COMPOSITION? (2)

Handpuppentheateraufführung

Hand|Puppen|Theater|Aufführung

hand|doll-s|theater|performance

WHY MODEL COMPOSITION? (2)

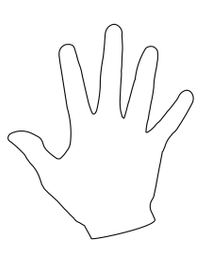
2,019,439 *35,921* *408,726* *91,268*

Hand | Puppe(n) | Theater | Aufführung

WHY MODEL COMPOSITION? (2)

2,019,439 *35,921* *408,726* *91,268*

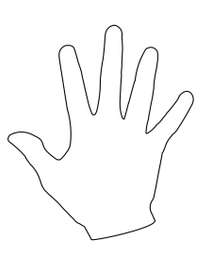
Hand | Puppe(n) | Theater | Aufführung



WHY MODEL COMPOSITION? (2)

2,019,439 35,921 408,726 91,268

Hand | Puppe(n) | Theater | Aufführung



Handpuppe

2,052

WHY MODEL COMPOSITION? (2)

2,019,439 35,921 408,726 91,268

Hand | Puppe(n) | Theater | Aufführung

Handpuppe

2,052

WHY MODEL COMPOSITION? (2)

2,019,439 35,921 408,726 91,268

Hand | Puppe(n) | Theater | Aufführung

Handpuppe

2,052

Handpuppentheater

90

WHY MODEL COMPOSITION? (2)

2,019,439 35,921 408,726 91,268

Hand | Puppe(n) | Theater | Aufführung

Handpuppe

2,052

Handpuppentheater

90

WHY MODEL COMPOSITION? (2)

2,019,439 35,921 408,726 91,268

Hand | Puppe(n) | Theater | Aufführung

Handpuppe

2,052

Handpuppentheater

90

Handpuppentheateraufführung

1

“

“The meaning of the whole is a function of the meaning of the parts and their mode of combination”

Frege’s Principle of Compositionality

COMPOSITION FUNCTIONS (1)

COMPOSITION FUNCTIONS (1)

- use distributional representations as a proxy for the meaning of individual words - “meaning for the parts”
  - any type of vector representations for words

COMPOSITION FUNCTIONS (1)

- use distributional representations as a proxy for the meaning of individual words - “meaning for the parts”
  - any type of vector representations for words
- look for a “mode of combination”

COMPOSITION FUNCTIONS (1)

- use distributional representations as a proxy for the meaning of individual words - “meaning for the parts”
  - any type of vector representations for words
- look for a “mode of combination”
- in the context of vector space models of language
  - find a composition function f that fulfils certain requirements

COMPOSITION FUNCTIONS (2)

COMPOSITION FUNCTIONS (2)

- *f* takes as input (two) *d*-dimensional distributional representations of the constituents (learned from a corpus)

COMPOSITION FUNCTIONS (2)

- f takes as input (two) d-dimensional distributional representations of the constituents (learned from a corpus)

$$f(u^{corpus}, v^{corpus})$$

COMPOSITION FUNCTIONS (2)

- f takes as input (two) d-dimensional distributional representations of the constituents (learned from a corpus)

$$f(u^{corpus}, v^{corpus})$$

- f produces as output another d-dimensional vector, which is a combination of the input vectors

COMPOSITION FUNCTIONS (2)

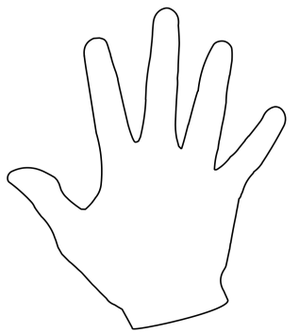
- f takes as input (two) d-dimensional distributional representations of the constituents (learned from a corpus)

$$f(u^{corpus}, v^{corpus})$$

- f produces as output another d-dimensional vector, which is a combination of the input vectors

$$p^{composed} = f(u^{corpus}, v^{corpus})$$

COMPOSITION AS A MATHEMATICAL TRANSFORMATION



Hand ‘hand’

Puppe ‘doll’

Handpuppe ‘hand puppet’

0.2
0.1
0.4

u

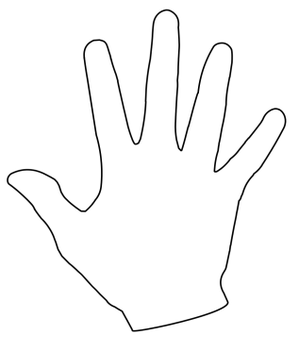
0.1
0.3
0.2

v

0.4
0.3
0.2

$\tilde{p}$

COMPOSITION AS A MATHEMATICAL TRANSFORMATION



Hand ‘hand’

Puppe ‘doll’

Handpuppe ‘hand puppet’

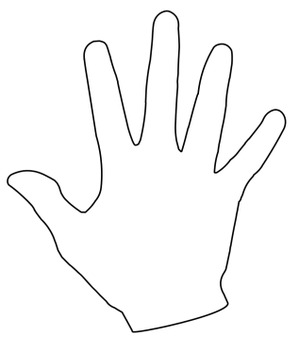
0.2
0.1
0.4
u

0.1
0.3
0.2
v

0.4
0.3
0.2
$\tilde{p}$

original (learned from the corpus)

COMPOSITION AS A MATHEMATICAL TRANSFORMATION



Hand ‘hand’

Puppe ‘doll’

Handpuppe ‘hand puppet’

u = (0.2, 0.1, 0.4)

v = (0.1, 0.3, 0.2)

$\tilde{p}$ = (0.4, 0.3, 0.2)

original (learned from the corpus)

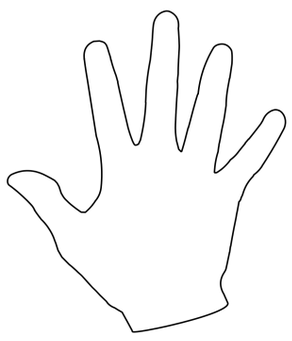
$$f(u, v) = p$$

u = (0.2, 0.1, 0.4)

v = (0.1, 0.3, 0.2)

p = (?, ?, ?)

COMPOSITION AS A MATHEMATICAL TRANSFORMATION



Hand ‘hand’

Puppe ‘doll’

Handpuppe ‘hand puppet’

u = (0.2, 0.1, 0.4)

v = (0.1, 0.3, 0.2)

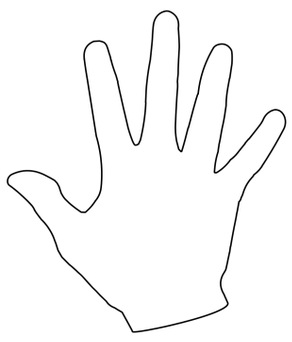
$\tilde{p}$ = (0.4, 0.3, 0.2)

original (learned from the corpus)

$f(u, v) = p$ = (?, ?, ?)

composed

COMPOSITION AS A MATHEMATICAL TRANSFORMATION



Hand ‘hand’

Puppe ‘doll’

Handpuppe ‘hand puppet’

$$\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u \qquad \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v \qquad \begin{bmatrix}0.4\\0.3\\0.2\end{bmatrix} \tilde{p}$$

original (learned from the corpus)

$$f\left(\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u\ ,\ \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v\right) = \begin{bmatrix}?\\?\\?\end{bmatrix} p$$

composed

$$\operatorname{argmax} \frac{p \cdot \tilde{p}}{\|p\|_2 \|\tilde{p}\|_2}$$

ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010

Hand ‘hand’ Puppe ‘doll’ Handpuppe ‘hand puppet’

$$\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u + \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v = \begin{bmatrix}0.3\\0.4\\0.6\end{bmatrix} p$$

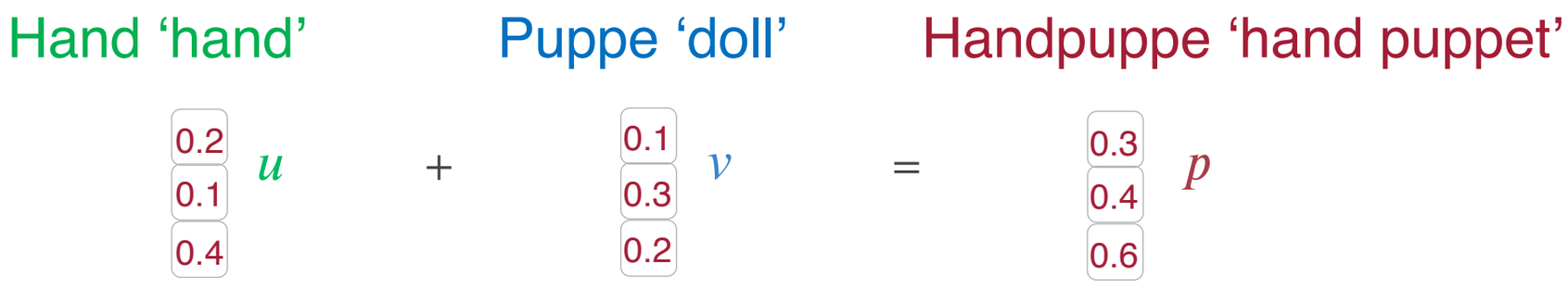
ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010

Hand ‘hand’ Puppe ‘doll’ Handpuppe ‘hand puppet’

$$\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u + \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v = \begin{bmatrix}0.3\\0.4\\0.6\end{bmatrix} p$$

Handpuppe ‘hand puppet’ = Puppenhand ‘doll hand’

ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010



Handpuppe
‘hand puppet’

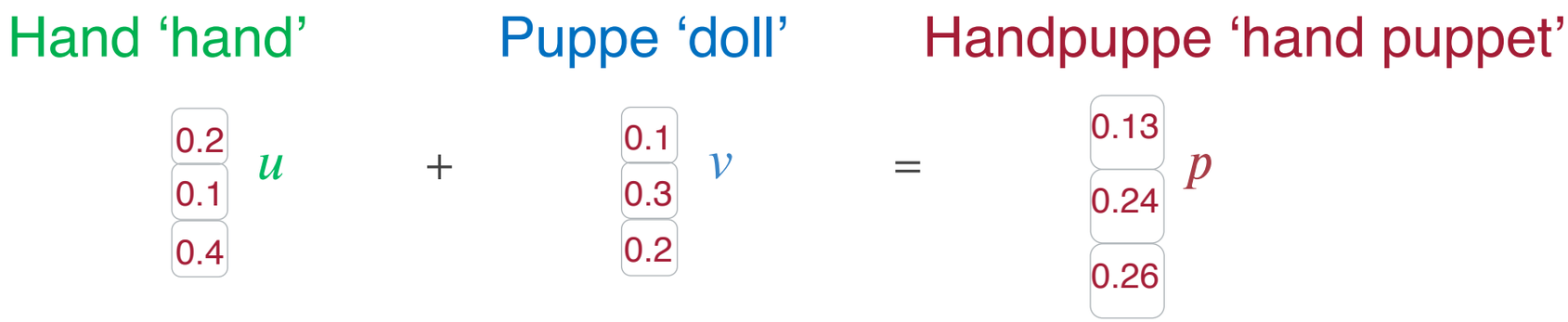
=

Puppenhand
‘doll hand’

“You might just as well say that ‘I see what I eat’ is the same thing as ‘I eat what I see’!”

(The Mad Hatter in Lewis Caroll, Alice’s Adventures in Wonderland)

WEIGHTED ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010



WEIGHTED ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010

Hand ‘hand’ Puppe ‘doll’ Handpuppe ‘hand puppet’

$$\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u + \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v = \begin{bmatrix}0.13\\0.24\\0.26\end{bmatrix} p \qquad \alpha = 0.3, \beta = 0.7$$

$$\alpha u + \beta v = p$$

$$\alpha, \beta \in \mathbb{R}$$
$$u, v, p \in \mathbb{R}^d$$

WEIGHTED ADDITIVE COMPOSITION – MITCHELL & LAPATA 2010

Hand ‘hand’ Puppe ‘doll’ Handpuppe ‘hand puppet’

$$\begin{bmatrix}0.2\\0.1\\0.4\end{bmatrix} u + \begin{bmatrix}0.1\\0.3\\0.2\end{bmatrix} v = \begin{bmatrix}0.13\\0.24\\0.26\end{bmatrix} p \qquad \alpha = 0.3, \beta = 0.7$$

$$\alpha u + \beta v = p$$

$$\alpha, \beta \in \mathbb{R}$$
$$u, v, p \in \mathbb{R}^d$$

Handpuppe
‘hand puppet’

$\neq$

Puppenhand
‘doll hand’

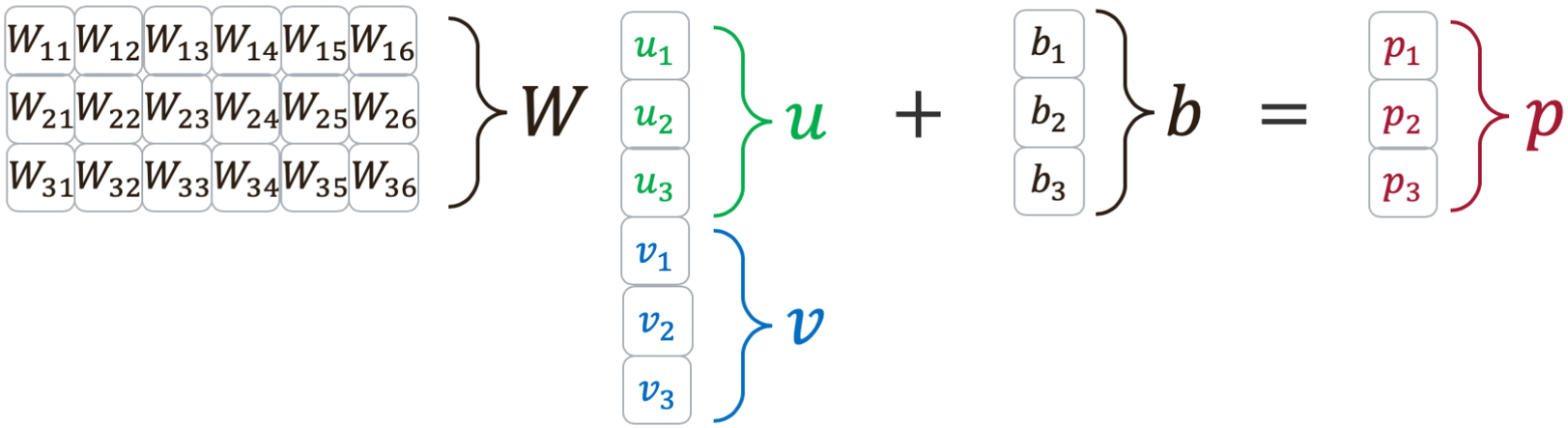
NEIGHBOURS OF ADDITIVE COMPOSITION

initiation_rite:1	**sewing_machine:1**	**poverty_trap:1**	**dipole_antenna:1**
initiation_rite 1.00000	sewing_machine 1.00000	poverty_trap 1.00000	dipole_antenna 1.00000
initiation_rite_c 0.48302	sewing_machine_c 0.59397	poverty_trap_c 0.48964	dipole_antenna_c 0.58292
initiation 0.41953	sewing 0.57117	benefit_system 0.46336	dipole 0.51744
kickback_scheme 0.39792	quilting 0.48184	welfare_system 0.41871	antenna 0.46497
prison_program 0.38725	machine_stitch 0.47350	marriage_penalty 0.40658	radiation_pattern 0.44861
rite 0.38340	machine_operator 0.46966	means_test 0.38529	dish_antenna 0.44799
ritual_killing 0.37734	embroidery 0.46061	rat_race 0.38416	throwing_knife 0.42321
rescue_vehicle:100	**reaction_time:100**	**advance_copy:100**	**phase_space:100**
rescue_vehicle 1.00000	reaction_time 1.00000	advance_copy 1.00000	phase_space 1.00000
rescue_equipment 0.49602	working_memory 0.51014	review_copy 0.59946	space_time 0.44398
rescue_squad 0.45921	response_time 0.47491	draft_copy 0.56645	phase_transition 0.42528
rescue_personnel 0.45826	learning_ability 0.46824	paperback_copy 0.51407	ground_state 0.41160
rescue_crew 0.45364	processing_time 0.44939	sneak_preview 0.47378	field_theory 0.41095
rescue_vessel 0.42381	cycle_time 0.44638	galley_proof 0.41045	quantum 0.40099
...	...	...	...
rescue_vehicle_c 0.30007	reaction_time_c 0.27750	advance_copy_c 0.27408	phase_space_c 0.26103
paper_market:1000	**vacuum_aspiration:1000**	**showpiece_event:1000**	**cart_track:1000**
paper_market 1.00000	vacuum_aspiration 1.00000	showpiece_event 1.00000	cart_track 1.00000
property_lending 0.63559	self_stimulation 0.46542	surprise_team 0.55356	dirt_path 0.58237
property_share 0.50744	crystal_gazing 0.45362	budget_session 0.54997	go_cart 0.56240
aircraft_market 0.50299	garden_cart 0.45026	weekend_election 0.53930	dirt_track 0.52775
property_sector 0.48177	exchange_transfusion 0.44189	rating_period 0.53279	logging_road 0.49539
air_lane 0.47996	motorized_wheelchair 0.43895	year_area 0.52381	gravel_road 0.47347
...	...	...	...
paper_market_c 0.16682	vacuum_aspiration_c 0.07520	showpiece_event_c 0.28813	cart_track_c 0.23656

NEIGHBOURS OF WEIGHTED ADDITIVE COMPOSITION

election_campaign:1	**decree_nisi:1**	**dung_beetle:1**	**drinking_alcohol:1**
election_campaign 1.00000	decree_nisi 1.00000	dung_beetle 1.00000	drinking_alcohol 1.00000
election_campaign_c 0.70908	decree_nisi_c 0.51042	dung_beetle_c 0.53302	drinking_alcohol_c 0.64101
election 0.65634	decree 0.45393	stag_beetle 0.45368	smoking 0.61267
fall_election 0.58337	annulment 0.45290	beetle 0.42449	drinking 0.60360
election_cycle 0.57361	summary_judgement 0.44723	pill_bug 0.41598	alcohol_consumption 0.57452
campaign_period 0.57320	committal_order 0.39749	dung 0.39514	binge_drinking 0.54170
campaign_team 0.55469	divorce 0.39232	water_strider 0.38161	alcohol_use 0.53805
rescue_vehicle:100	**application_procedure:100**	**music_market:100**	**phase_space:100**
rescue_vehicle 1.00000	application_procedure 1.00000	music_market 1.00000	phase_space 1.00000
rescue_equipment 0.49602	application_process 0.67551	music_world 0.58263	space_time 0.44398
rescue_squad 0.45921	application_form 0.59632	music_chart 0.54964	phase_transition 0.42528
rescue_personnel 0.45826	job_application 0.54862	pop_music 0.52977	ground_state 0.41160
rescue_crew 0.45364	grant_application 0.50493	music_industry 0.51918	field_theory 0.41095
rescue_vessel 0.42381	visa_application 0.46575	music_sale 0.51078	quantum 0.40099
...	...	...	...
rescue_vehicle_c 0.29996	application_procedure_c 0.30313	music_market_c 0.34352	phase_space_c 0.26121
shoo_fly:1000	**tobacco_market:1000**	**paper_wasp:1000**	**sports_vehicle:1000**
shoo_fly 1.00000	tobacco_market 1.00000	paper_wasp 1.00000	sports_vehicle 1.00000
pepper_steak 0.52205	wine_market 0.59988	garden_spider 0.58116	dangling_modifier 0.65794
potato_bug 0.50899	cigarette_market 0.59143	mud_dauber 0.56821	cutting_implement 0.63771
tub_thumper 0.49676	tea_market 0.58791	cicada_killer 0.56456	four-wheel-drive_vehicle 0.62591
rum_baba 0.49114	arms_market 0.57372	mining_bee 0.52083	toy_vehicle 0.62213
dice_box 0.48834	sugar_market 0.55928	house_centipede 0.49953	import_system 0.62018
...	...	...	...
shoo_fly_c -0.02019	tobacco_market_c 0.14430	paper_wasp_c 0.19814	sports_vehicle_c -0.13707

MATRIX COMPOSITION – SOCHER ET AL., 2010



$$p_i = W_{i1}u_1 + W_{i2}u_2 + W_{i3}u_3 + W_{i4}v_1 + W_{i5}v_2 + W_{i6}v_3 + b_i, \forall i \in \{1,2,3\}$$

$$p_i = W_{i*} \cdot [u; v] + b_i, \forall i \in \{1,2,3\}$$

MATRIX COMPOSITION – SOCHER ET AL., 2010

$$W \in \mathbb{R}^{d \times 2d}$$

$$u, v, b, p \in \mathbb{R}^{d}$$

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} u \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} v + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

$$p_i = W_{i1}u_1 + W_{i2}u_2 + W_{i3}u_3 + W_{i4}v_1 + W_{i5}v_2 + W_{i6}v_3 + b_i, \forall i \in \{1,2,3\}$$

$$p_i = W_{i*} \cdot [u; v] + b_i, \forall i \in \{1,2,3\}$$

MATRIX COMPOSITION – SOCHER ET AL., 2010

$$W \in \mathbb{R}^{d \times 2d}$$

$$u, v, b, p \in \mathbb{R}^{d}$$

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} u \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} v + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

$$p_i = W_{i1}u_1 + W_{i2}u_2 + W_{i3}u_3 + W_{i4}v_1 + W_{i5}v_2 + W_{i6}v_3 + b_i, \forall i \in \{1,2,3\}$$

$$p_i = W_{i*} \cdot [u; v] + b_i, \forall i \in \{1,2,3\}$$

- ***global composition*** *- the two words from any pair of words - e.g. (apple, tree), (car, factory), (hand, puppet) will be composed in the same way*

NEIGHBOURS OF MATRIX COMPOSITION

dog__collar:1	rap__artist:1	property__sale:1	hill__town:1
dog__collar 1.00000	rap__artist 1.00000	property__sale 1.00000	hill__town 1.00000
dog__collar__c 0.54199	rap__artist__c 0.66841	property__sale__c 0.58137	hill__town__c 0.49966
leash 0.48995	rap__star 0.64815	property__transaction 0.54844	mountain__village 0.44740
collar 0.44047	recording__artist 0.56339	property__management 0.52949	market__town 0.40680
dog__tag 0.41794	hip__hop 0.54667	home__sale 0.50604	beach__town 0.37561
shoulder__strap 0.38407	rap 0.54657	estate__agent 0.47905	university__town 0.37451
sports__jacket 0.38068	rap__music 0.52175	property__market 0.46321	market__square 0.37379

brand__manager:3	time__period:3	lake__poets:3	cheese__dip:3
brand__manager 1.00000	time__period 1.00000	lake__poets 1.00000	cheese__dip 1.00000
marketing__manager 0.70351	time__frame 0.75858	folk__poet 0.65036	cheese__sauce 0.55767
product__manager 0.65581	period 0.68544	sand__hopper 0.63909	taco__sauce 0.54454
brand__manager__c 0.64731	time__period__c 0.59635	lake__poets__c 0.59926	cheese__dip__c 0.51926
marketing__director 0.64690	time__interval 0.59194	forest__god 0.59435	bechamel__sauce 0.51546
product__marketing 0.62479	year__period 0.58240	sea__swallow 0.59369	bean__dip 0.50829
marketing__executive 0.59426	time__span 0.57384	horse__latitude 0.59048	wax__bean 0.50158

rose__hip:1000	developing__cost:1000	snow__job:1000	walkie__talkie:1000
rose__hip 1.00000	developing__cost 1.00000	snow__job 1.00000	walkie__talkie 1.00000
pomegranate 0.54706	morale__building 0.52340	squeeze__play 0.47097	police__radio 0.50134
dog__rose 0.51804	profit__projection 0.51079	snipe__hunt 0.46792	radio__set 0.46869
borage 0.50045	data__formatting 0.49961	tub__thumper 0.44253	cell__phone 0.46803
evening__primrose 0.48132	shareholder__litigation 0.49381	conversation__stopper 0.43753	crystal__set 0.45369
rose__oil 0.47833	inventory__adjustment 0.49292	ham__actor 0.42959	communication__device 0.45242
...	...	...	...
rose__hip__c 0.18460	developing__cost__c 0.31569	snow__job__c 0.27037	walkie__talkie__c 0.16783

TRAINING COMPOSITION FUNCTIONS

Apfel ‘apple’	Baum ‘tree’	Apfelbaum ‘apple tree’
Holz ‘wood’	Löffel ‘spoon’	Holzlöffel ‘wooden spoon’
Abend ‘evening’	Kleid ‘dress’	Abendkleid ‘evening dress’
Chef ‘chef’	Koch ‘cook’	Chefkoch ‘chef cook’
Mandel ‘almond’	Öl ‘oil’	Mandelöl ‘almond oil’
Orange ‘orange’	Saft ‘juice’	Orangensaft ‘orange juice’
Gast ‘guest’	Bad ‘bathroom’	Gästebad ‘guest bathroom’
Hochzeit ‘wedding’	Bild ‘photo’	Hochzeitsbild ‘wedding photo’
…	…	…

TRAINING COMPOSITION FUNCTIONS

Apfel ‘apple’	Baum ‘tree’	Apfelbaum ‘apple tree’
Holz ‘wood’	Löffel ‘spoon’	Holzlöffel ‘wooden spoon’
Abend ‘evening’	Kleid ‘dress’	Abendkleid ‘evening dress’
Chef ‘chef’	Koch ‘cook’	Chefkoch ‘chef cook’
Mandel ‘almond’	Öl ‘oil’	Mandelöl ‘almond oil’
Orange ‘orange’	Saft ‘juice’	Orangensaft ‘orange juice’
Gast ‘guest’	Bad ‘bathroom’	Gästebad ‘guest bathroom’
Hochzeit ‘wedding’	Bild ‘photo’	Hochzeitsbild ‘wedding photo’
…	…	…

u_1	v_1	$\tilde{p}_1$	$p_1 = f(u_1, v_1)$
u_2	v_2	$\tilde{p}_2$	$p_2 = f(u_2, v_2)$
u_3	v_3	$\tilde{p}_3$	$p_3 = f(u_3, v_3)$
u_4	v_4	$\tilde{p}_4$	$p_4 = f(u_4, v_4)$
u_5	v_5	$\tilde{p}_5$	$p_5 = f(u_5, v_5)$
u_6	v_6	$\tilde{p}_6$	$p_6 = f(u_6, v_6)$
u_7	v_7	$\tilde{p}_7$	$p_7 = f(u_7, v_7)$
u_8	v_8	$\tilde{p}_8$	$p_8 = f(u_8, v_8)$
…	…	…	…

TRAINING COMPOSITION FUNCTIONS

Apfel 'apple'	Baum 'tree'	Apfelbaum 'apple tree'
Holz 'wood'	Löffel 'spoon'	Holzlöffel 'wooden spoon'
Abend 'evening'	Kleid 'dress'	Abendkleid 'evening dress'
Chef 'chef'	Koch 'cook'	Chefkoch 'chef cook'
Mandel 'almond'	Öl 'oil'	Mandelöl 'almond oil'
Orange 'orange'	Saft 'juice'	Orangensaft 'orange juice'
Gast 'guest'	Bad 'bathroom'	Gästebad 'guest bathroom'
Hochzeit 'wedding'	Bild 'photo'	Hochzeitsbild 'wedding photo'
…	…	…

u_1	v_1	$\tilde{p}_1$	$p_1 = f(u_1, v_1)$
u_2	v_2	$\tilde{p}_2$	$p_2 = f(u_2, v_2)$
u_3	v_3	$\tilde{p}_3$	$p_3 = f(u_3, v_3)$
u_4	v_4	$\tilde{p}_4$	$p_4 = f(u_4, v_4)$
u_5	v_5	$\tilde{p}_5$	$p_5 = f(u_5, v_5)$
u_6	v_6	$\tilde{p}_6$	$p_6 = f(u_6, v_6)$
u_7	v_7	$\tilde{p}_7$	$p_7 = f(u_7, v_7)$
u_8	v_8	$\tilde{p}_8$	$p_8 = f(u_8, v_8)$
…	…	…	…

Training objective: $\operatorname{argmax} \dfrac{p \cdot \tilde{p}}{\|p\|_2 \|\tilde{p}\|_2}$

TRAINING COMPOSITION FUNCTIONS (2)

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
  - addition:

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition:

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition: *2 parameters*

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition: *2 parameters*
    - matrix:

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition: *2 parameters*
    - matrix: *d x 2d + d parameters (d=200: 200x400+200 = 80,200)*

TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition: *2 parameters*
    - matrix: *d x 2d + d parameters (d=200: 200x400+200 = 80,200)*
- choose hyperparameters using the *dev* data (10%) - learning rate, optimizer, embedding size, embedding type, etc.

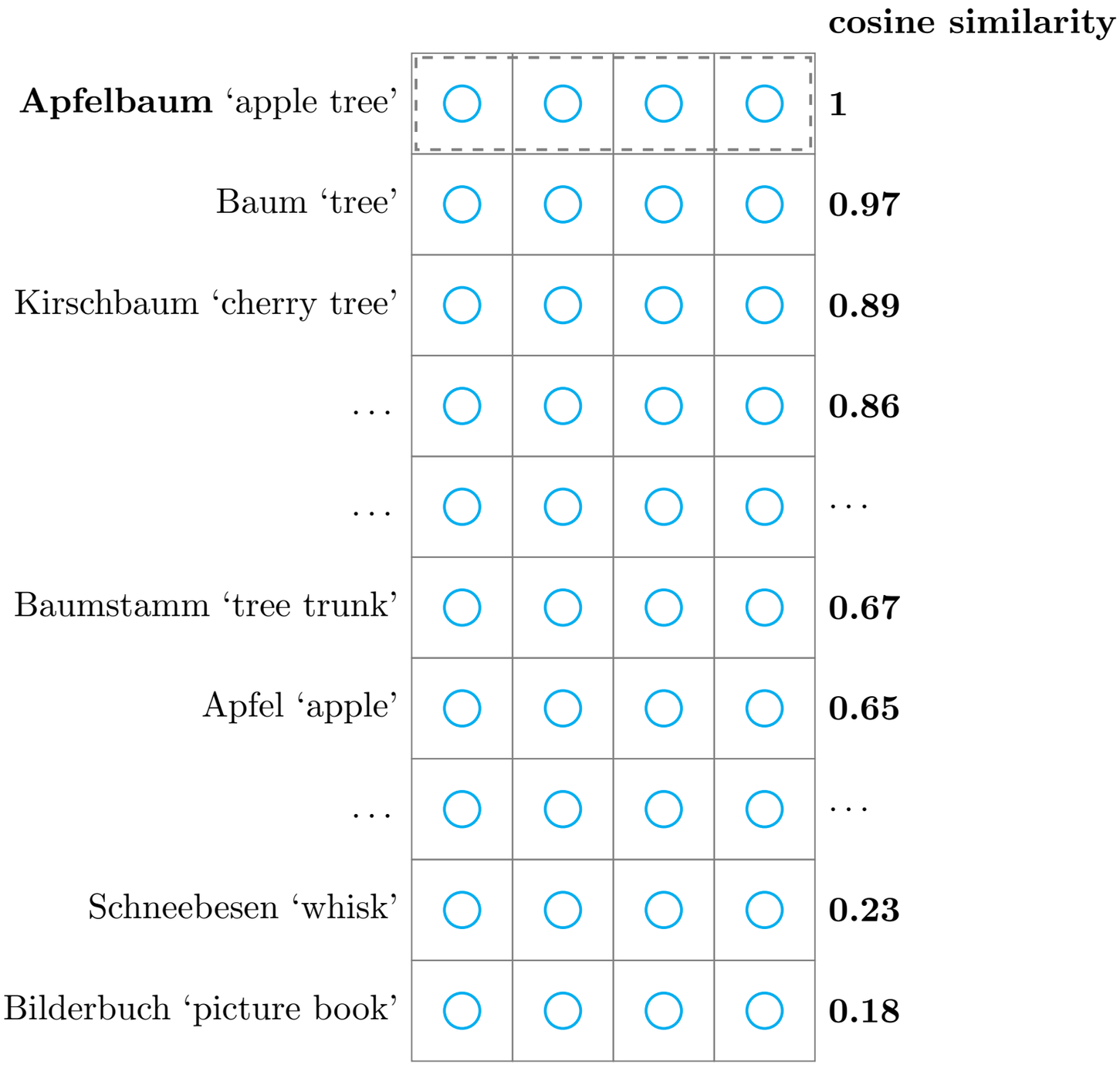
TRAINING COMPOSITION FUNCTIONS (2)

- use as many such triples as you can get (datasets ranging from 4,500 to 240,000 triples)
- split the data into *train-test-dev* portions (70-20-10%)
- train the composition model on the *train* data (70%)
  - estimate all the parameters of composition model
    - addition: *0 parameters (non-parametric)*
    - weighted addition: *2 parameters*
    - matrix: *d x 2d + d parameters (d=200: 200x400+200 = 80,200)*
- choose hyperparameters using the *dev* data (10%) - learning rate, optimizer, embedding size, embedding type, etc.
- test the composition model with the best hyperparameters **once** on the *test* data (20%)

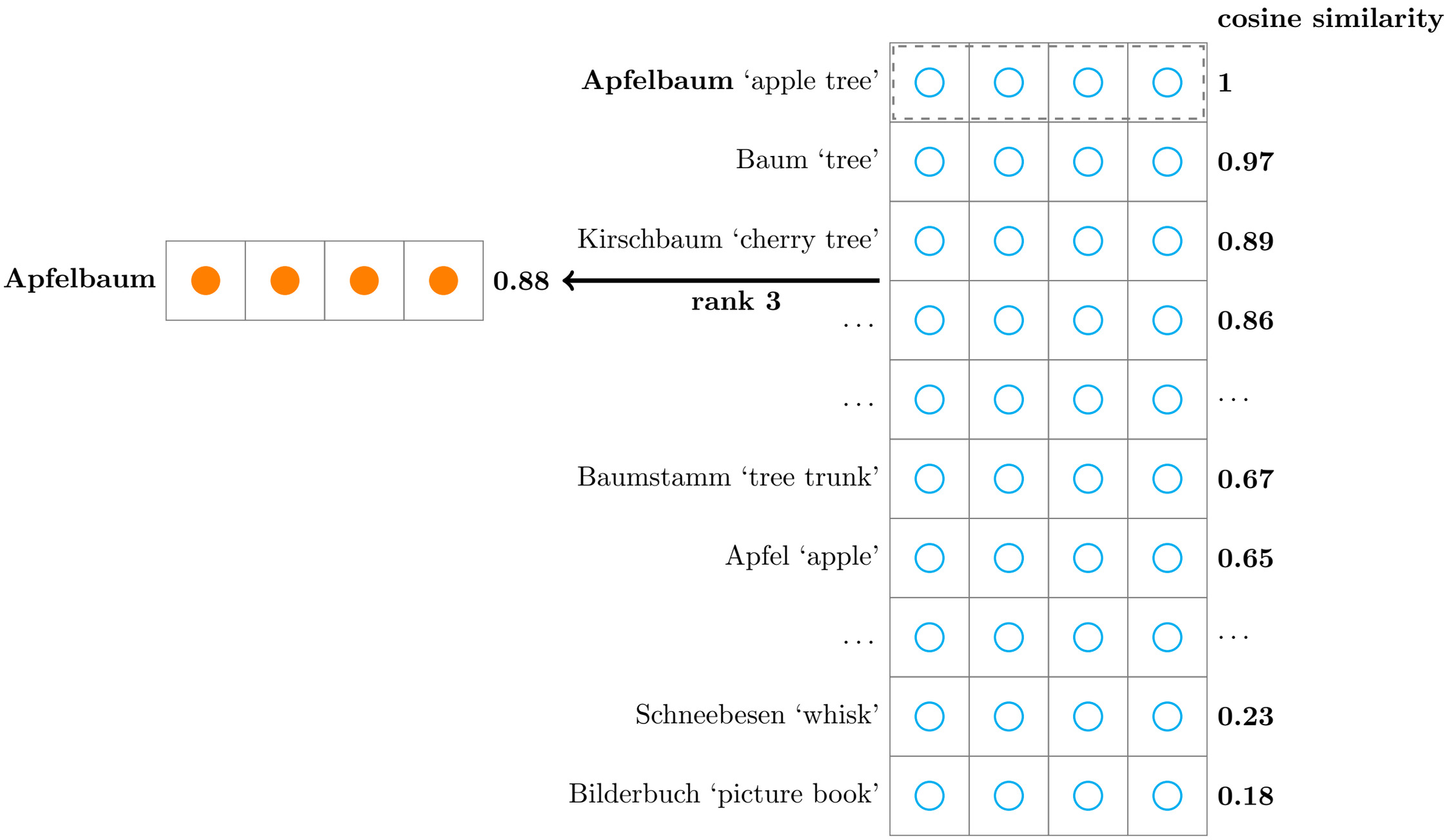
EVALUATION FOR COMPOSITION MODELS

- intrinsic evaluation - how good are the composed representations created by the model when compared to corpus-derived representations of the same words?
- extrinsic evaluation - how good are the composed representations for some external task (e.g. for identifying the semantic relations between words, for parsing, for sentiment analysis etc.)

INTRINSIC EVALUATION

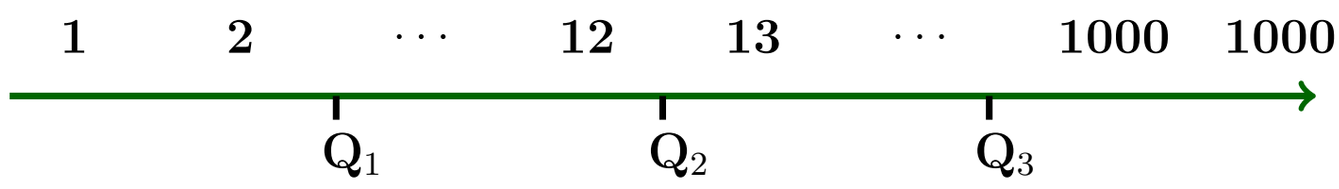


INTRINSIC EVALUATION (2)



INTRINSIC EVALUATION (3)

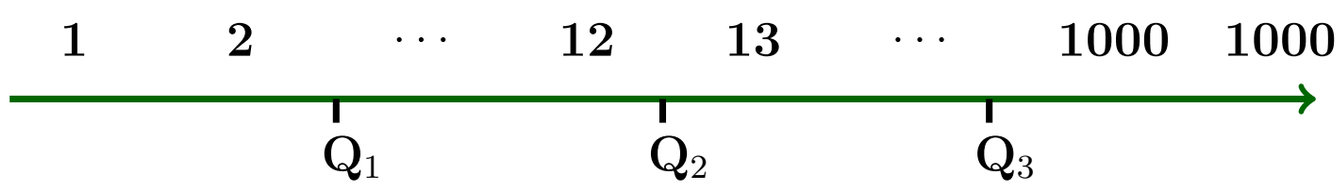
test compound	**rank**
Telefonkabel ‘telephone cable’	**1**
Schlossturm ‘castle tower’	**2**
...	...
Hundekuchen ‘dog biscuit’	**12**
Sitzheizung ‘seat heating’	**13**
...	...
Maulwurf ‘mole’	**1000**
Milchmädchen ‘milkmaid’	**1000**



INTRINSIC EVALUATION (3)

test compound	rank
Telefonkabel ‘telephone cable’	**1**
Schlossturm ‘castle tower’	**2**
...	...
Hundekuchen ‘dog biscuit’	**12**
Sitzheizung ‘seat heating’	**13**
...	...
Maulwurf ‘mole’	**1000**
Milchmädchen ‘milkmaid’	**1000**

Q_1, Q_2(median), Q_3 are the quartiles of the sorted list of ranks



INTRINSIC EVALUATION (4)

	English Nominal Compounds	
Composition Model	Q1, Q2, Q3	<=5
Addition	2,7,38	46,14 %
Weighted Addition	2,7,38	46,14 %
Matrix	1,2,9	67,37 %

MATRIX COMPOSITION – SOCHER ET AL., 2010

$$W \in \mathbb{R}^{d \times 2d}$$

$$u, v, b, p \in \mathbb{R}^{d}$$

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} u \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} v + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

$$p_i = W_{i1}u_1 + W_{i2}u_2 + W_{i3}u_3 + W_{i4}v_1 + W_{i5}v_2 + W_{i6}v_3 + b_i, \forall i \in \{1,2,3\}$$

$$p_i = W_{i*} \cdot [u; v] + b_i, \forall i \in \{1,2,3\}$$

- ***global composition*** *- the two words from any pair of words - e.g. (apple, tree), (car, factory), (hand, puppet) will be composed in the same way*

FULLLEX COMPOSITION – SOCHER ET AL., 2012 (1)

II.

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \begin{bmatrix} u_{V_1} \\ u_{V_2} \\ u_{V_3} \\ v_{U_1} \\ v_{U_2} \\ v_{U_3} \end{bmatrix} u_V \; v_U + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

FULLLEX COMPOSITION – SOCHER ET AL., 2012 (1)

$$u_V, v_U, b, p \in \mathbb{R}^d$$

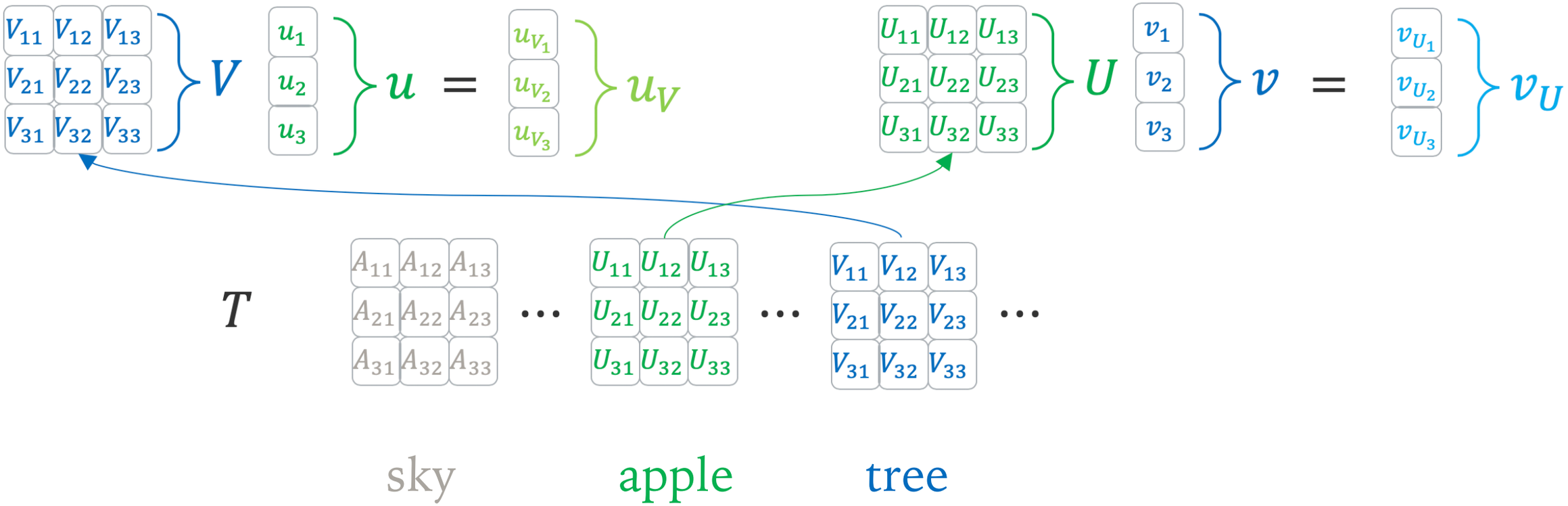
$$W \in \mathbb{R}^{d \times 2d}$$

II.

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \begin{bmatrix} u_{V_1} \\ u_{V_2} \\ u_{V_3} \\ v_{U_1} \\ v_{U_2} \\ v_{U_3} \end{bmatrix} u_V \; v_U + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

FULLLEX COMPOSITION – SOCHER ET AL., 2012 (2)

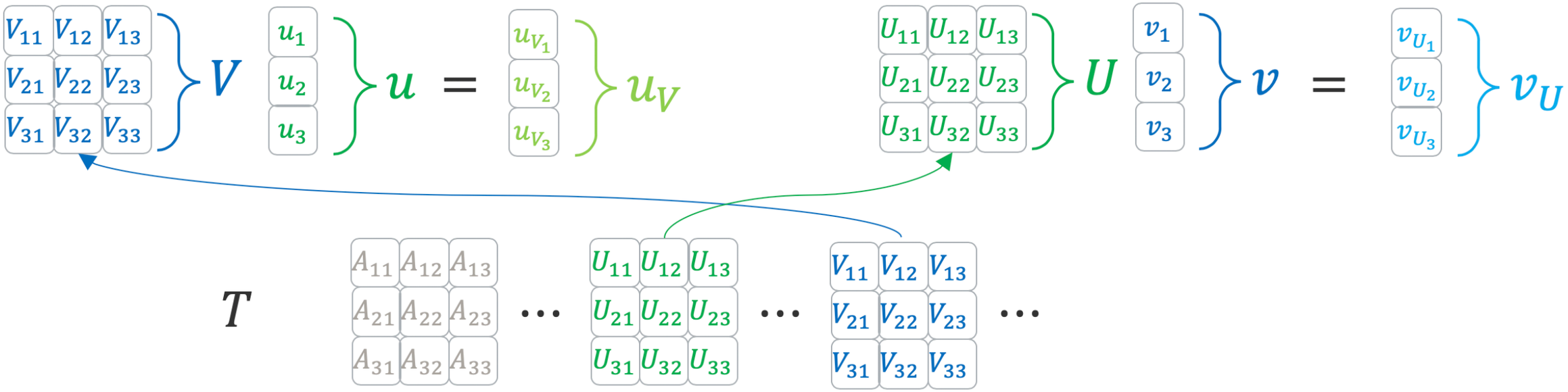
I.



FULLLEX COMPOSITION – SOCHER ET AL., 2012 (2)

$u = vector\ for$ apple

I.

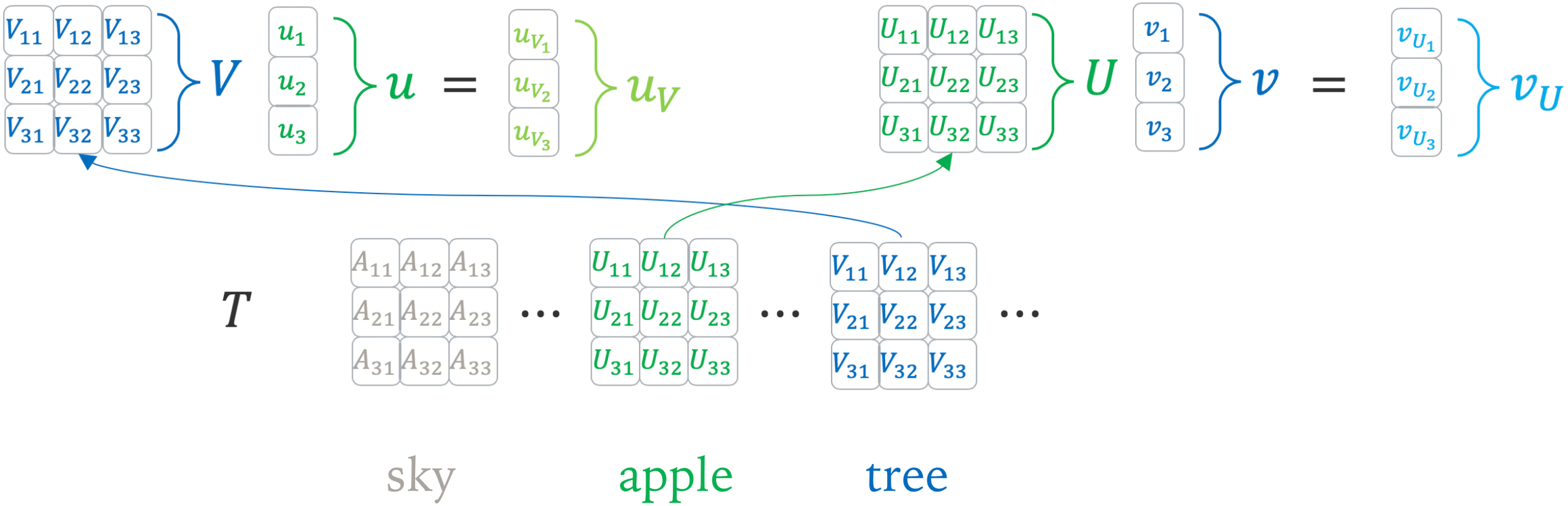


FULLLEX COMPOSITION – SOCHER ET AL., 2012 (2)

u = *vector for* apple

v = *vector for* tree

I.

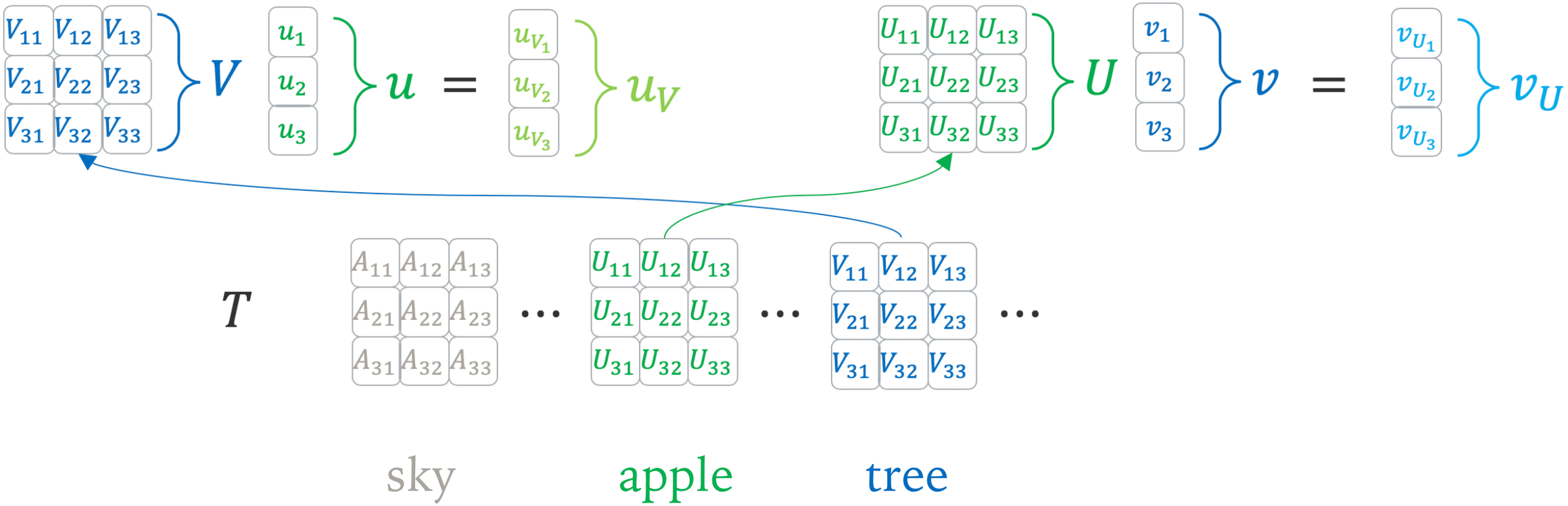


FULLLEX COMPOSITION – SOCHER ET AL., 2012 (2)

u = *vector for* apple

v = *vector for* tree

I.



$u, u_V, v, v_U \in \mathbb{R}^d$

$V, U \in \mathbb{R}^{d \times d}$

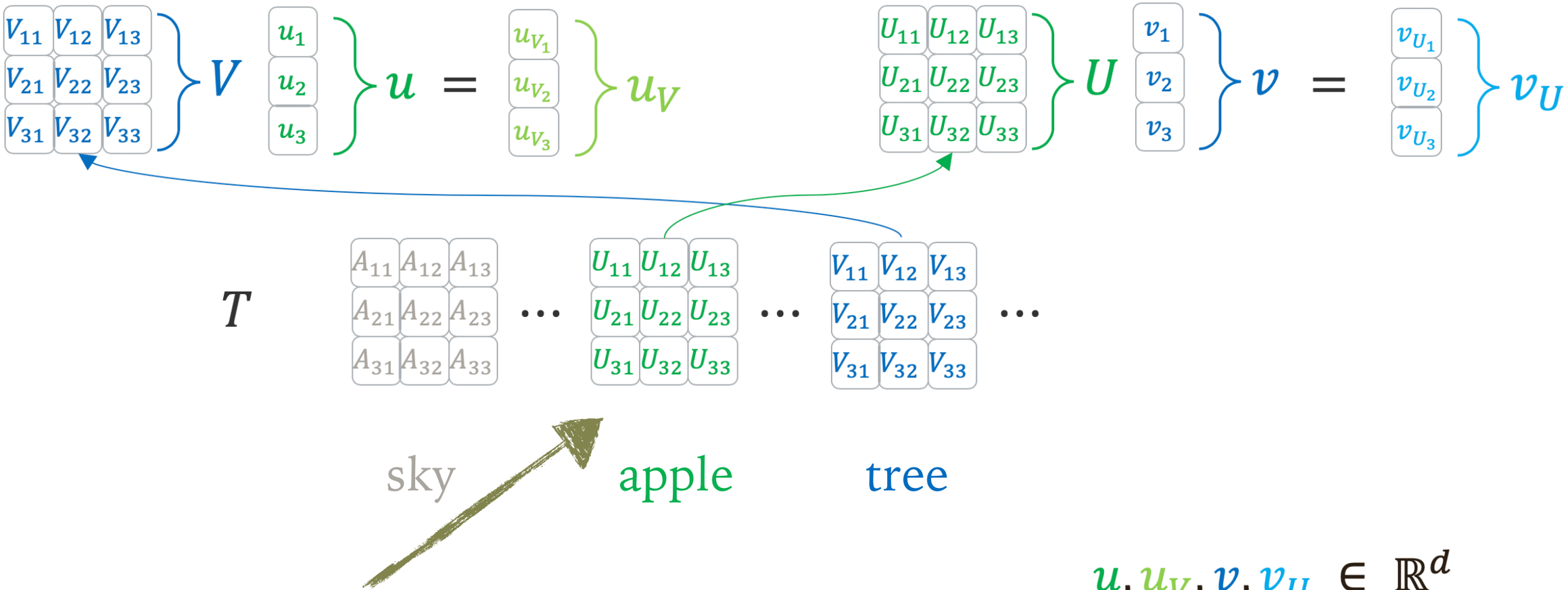
$T \in \mathbb{R}^{|V| \times d \times d}$

FULLLEX COMPOSITION – SOCHER ET AL., 2012 (2)

u = *vector for* apple

v = *vector for* tree

I.



initialized with the identity matrix

$u, u_V, v, v_U \in \mathbb{R}^d$

$V, U \in \mathbb{R}^{d \times d}$

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

FULLLEX COMPOSITION – SOCHER ET AL., 2012 (3)

I.

$$\begin{bmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \\ V_{31} & V_{32} & V_{33} \end{bmatrix} V \quad \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} u = \begin{bmatrix} u_{V_1} \\ u_{V_2} \\ u_{V_3} \end{bmatrix} u_V$$

$$\begin{bmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & U_{32} & U_{33} \end{bmatrix} U \quad \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} v = \begin{bmatrix} v_{U_1} \\ v_{U_2} \\ v_{U_3} \end{bmatrix} v_U$$

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

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \cdots \begin{bmatrix} U_{11} & U_{12} & U_{13} \\ U_{21} & U_{22} & U_{23} \\ U_{31} & U_{32} & U_{33} \end{bmatrix} \cdots \begin{bmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \\ V_{31} & V_{32} & V_{33} \end{bmatrix} \cdots$$

lexicalized

II.

$$\begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{14} & W_{15} & W_{16} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} & W_{26} \\ W_{31} & W_{32} & W_{33} & W_{34} & W_{35} & W_{36} \end{bmatrix} W \quad \begin{bmatrix} u_{V_1} \\ u_{V_2} \\ u_{V_3} \\ v_{U_1} \\ v_{U_2} \\ v_{U_3} \end{bmatrix} \begin{matrix} u_V \\ v_U \end{matrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} b = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} p$$

global

$$p_i = W_{i*} \cdot [u_V ; v_U] + b_i, \forall i \in \{1,2,3\}$$

NEIGHBOURS OF FULLLEX COMPOSITION

monitor_program:1	party_chief:1	rogue_elephant:1	service_division:1
monitor_program 1.00000	party_chief 1.00000	rogue_elephant 1.00000	service_division 1.00000
monitor_program_c 0.48685	party_chief_c 0.57671	rogue_elephant_c 0.66267	service_division_c 0.70384
import_system 0.44099	party_boss 0.45157	horse_doctor 0.66240	service_department 0.56786
program_quality 0.43015	party_official 0.43346	plow_horse 0.65587	service_facility 0.56749
traffic_mitigation 0.42661	party_leader 0.42197	chicken_yard 0.65222	service_operation 0.51915
personality_quiz 0.42372	politburo_member 0.41440	bank_guard 0.63444	service_program 0.51402
buffer_store 0.42320	party_branch 0.41280	government_man 0.62989	service_office 0.51309

mail_system:2	vegetable_garden:2	shooting_lodge:2	guide_rope:2
mail_system 1.00000	vegetable_garden 1.00000	shooting_lodge 1.00000	guide_rope 1.00000
email_system 0.64818	vegetable_patch 0.76619	hunting_lodge 0.61701	center_pole 0.68092
mail_system_c 0.60377	vegetable_garden_c 0.74822	shooting_lodge_c 0.51058	guide_rope_c 0.64363
voice_mail 0.55972	kitchen_garden 0.73704	shooting_box 0.50183	grappling_iron 0.64078
mail_service 0.54594	herb_garden 0.71074	cow_barn 0.47860	water_wagon 0.61621
mail_box 0.45023	flower_bed 0.67242	army_hut 0.46117	snatch_block 0.60451
snail_mail 0.42475	fruit_tree 0.64774	manor_house 0.45763	starting_stall 0.60083

custom_officer:1000	sounding_board:1000	service_break:1000	home_plate:1000
custom_officer 1.00000	sounding_board 1.00000	service_break 1.00000	home_plate 1.00000
custom_official 0.76468	rallying_point 0.42654	rating_period 0.60125	plate_umpire 0.68733
stretcher_party 0.71902	launching_pad 0.41923	chow_line 0.56250	center_field 0.49705
slip_coach 0.69349	testing_ground 0.41371	operation_officer 0.55321	umpire 0.49245
riot_officer 0.68981	reference_point 0.38087	pace_lap 0.55097	dugout 0.46930
poll_taker 0.68704	teaching_tool 0.37899	moving_staircase 0.54872	strike_zone 0.45223
…	…	…	…
custom_officer_c 0.43134	sounding_board_c 0.15510	service_break_c 0.35390	home_plate_c 0.09510

ISSUES WITH FULLLEX

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - $|V| = 100$ words; d (size of word representation) $= 200$

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - $|V| = 100$ words; d (size of word representation) $= 200$
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
  - total: 4,080,200 parameters

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 4,080,200 parameters
  - |V| = 10,000 words; d=200

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 4,080,200 parameters
  - |V|= 10,000 words; d=200
    - T size:

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 4,080,200 parameters
  - |V|= 10,000 words; d=200
    - T size:
      - 10,000 x 200 x 200 = 10,000 x 40,000 = 400,000,000

ISSUES WITH FULLLEX

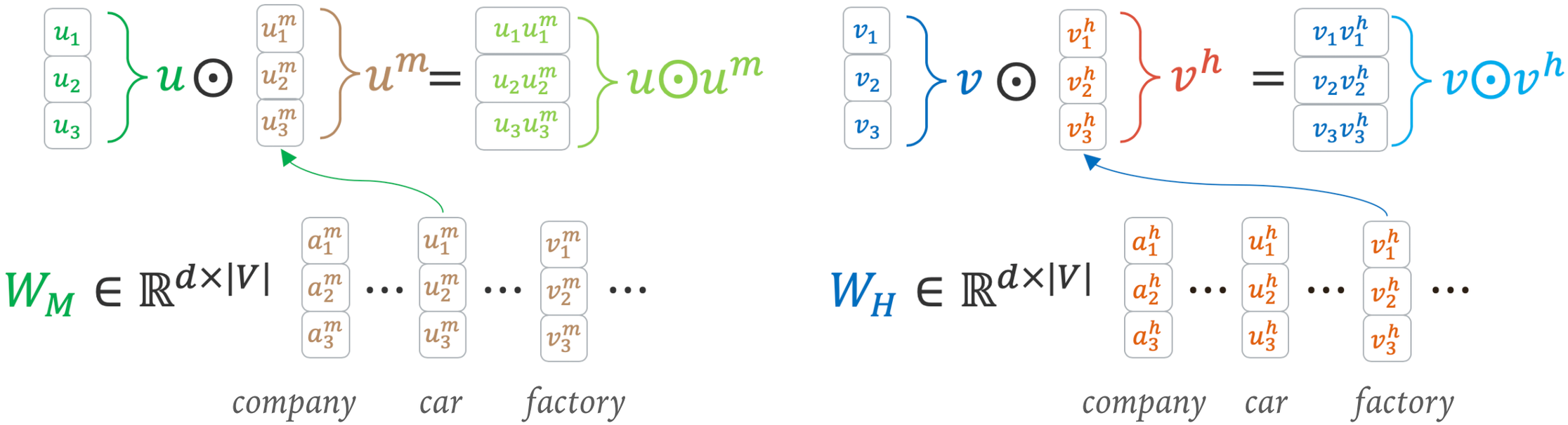
- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 4,080,200 parameters
  - |V|= 10,000 words; d=200
    - T size:
      - 10,000 x 200 x 200 = 10,000 x 40,000 = 400,000,000
    - W size: 200 x 400 + 200 = 80,200

ISSUES WITH FULLLEX

- the number of parameters increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - T size: 100 x 200 x 200 = 100 x 40,000 = 4,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 4,080,200 parameters
  - |V|= 10,000 words; d=200
    - T size:
      - 10,000 x 200 x 200 = 10,000 x 40,000 = 400,000,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 400,080,000 parameters

MASK COMPOSITION – DIMA, 2015 (1)

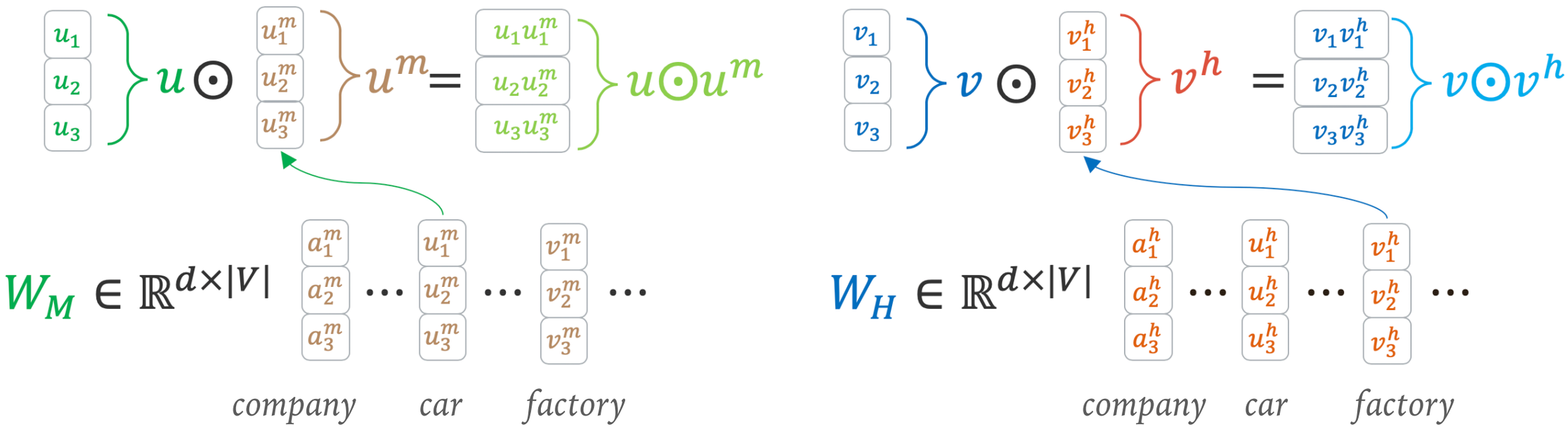
I.



MASK COMPOSITION – DIMA, 2015 (1)

$u, v, u^m, v^h, u \odot u^m, v \odot v^h \in \mathbb{R}^d$

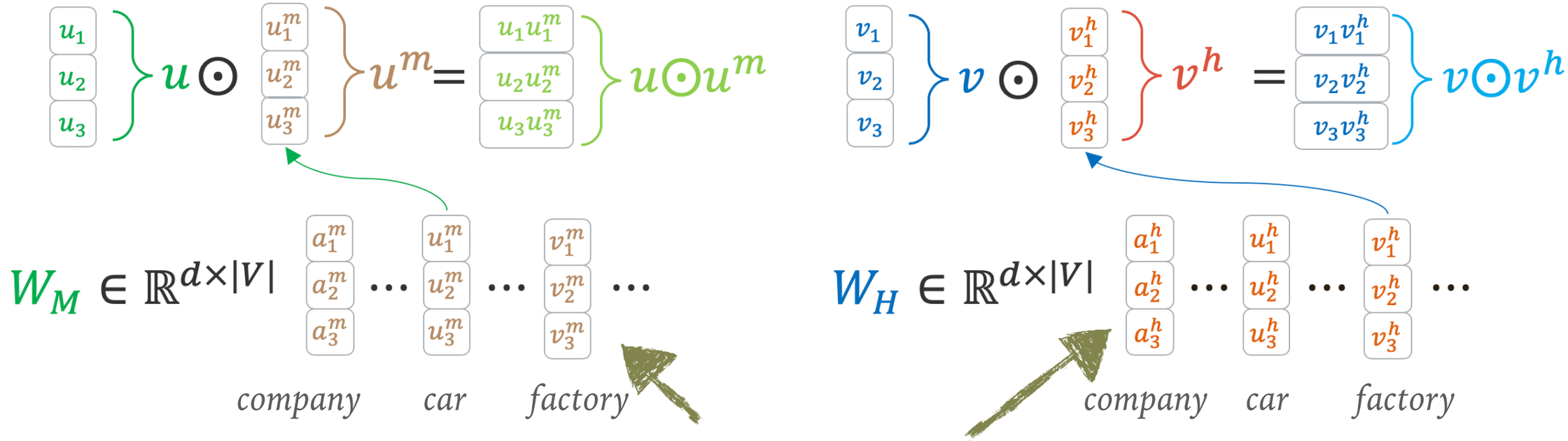
I.



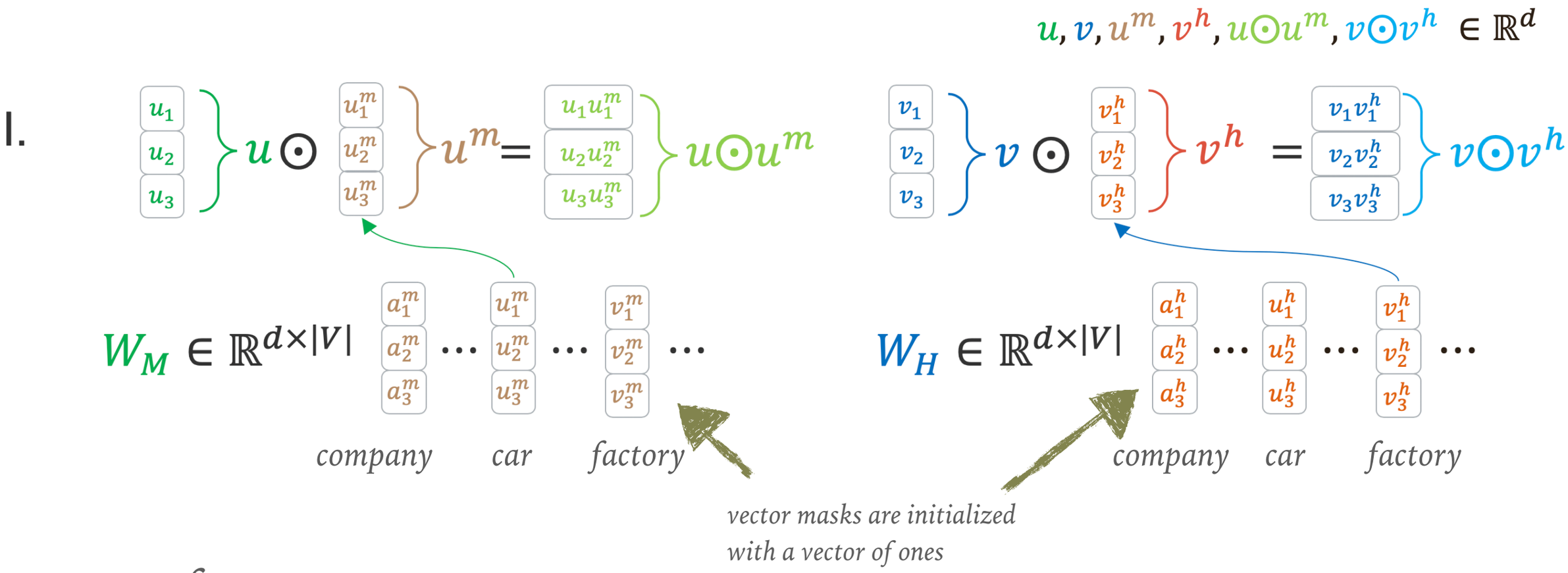
MASK COMPOSITION – DIMA, 2015 (1)

$u, v, u^m, v^h, u \odot u^m, v \odot v^h \in \mathbb{R}^d$

I.



MASK COMPOSITION – DIMA, 2015 (1)

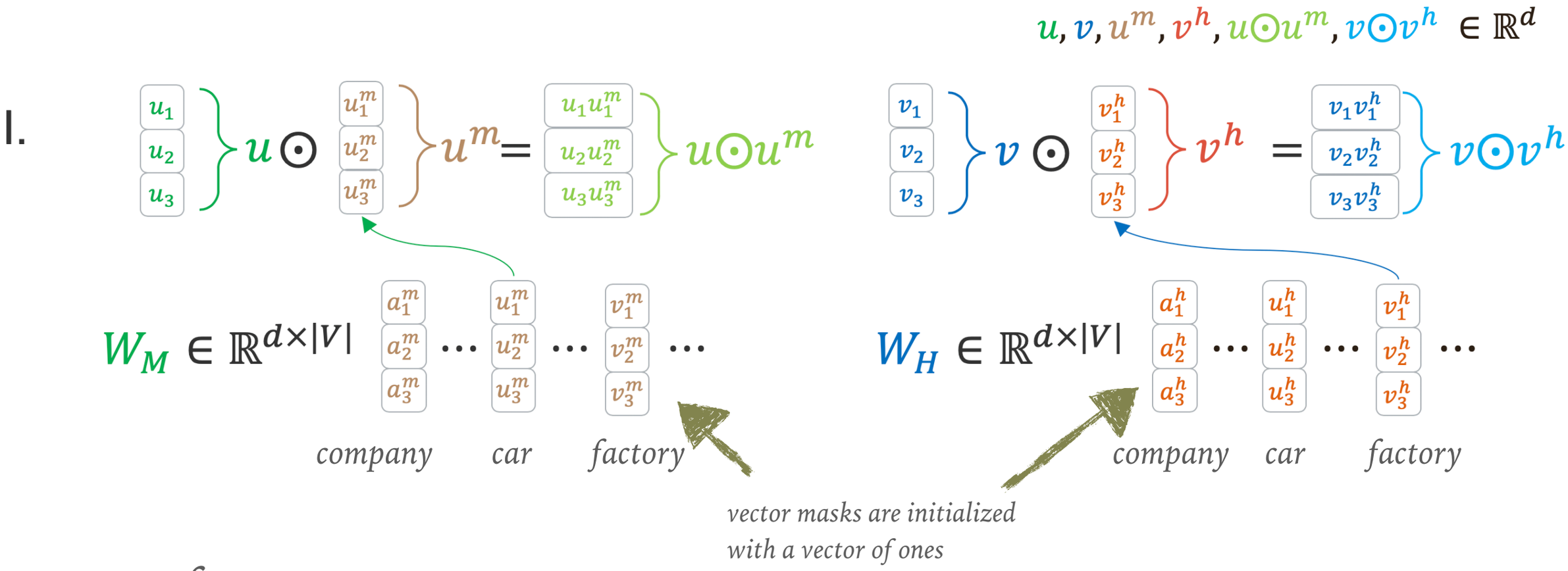


car factory:

u = vector for car; u^m = vector mask for modifier car

v = vector for factory; v^h = vector mask for head factory

MASK COMPOSITION – DIMA, 2015 (1)



car factory:

u = *vector for car;* u^m = *vector mask for modifier car*

v = *vector for factory;* v^h = *vector mask for head factory*

company car:

u = *vector for ?;* u^m = *vector mask for modifier ?*

v = *vector for ?;* v^h = *vector mask for head ?*

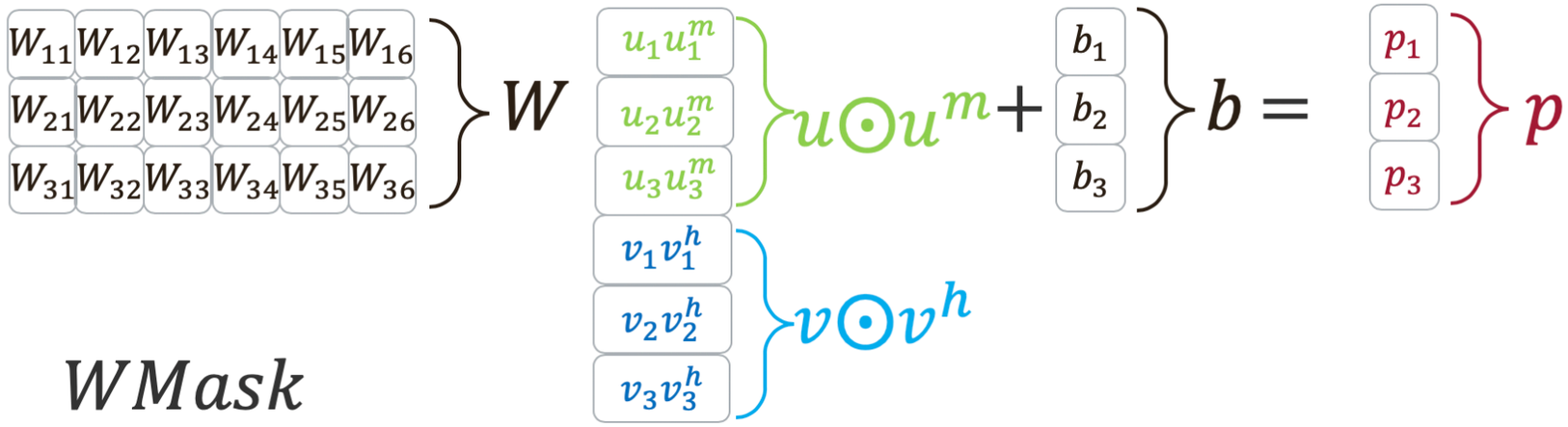
MASK COMPOSITION – DIMA, 2015 (2)

II.

MASK COMPOSITION – DIMA, 2015 (2)

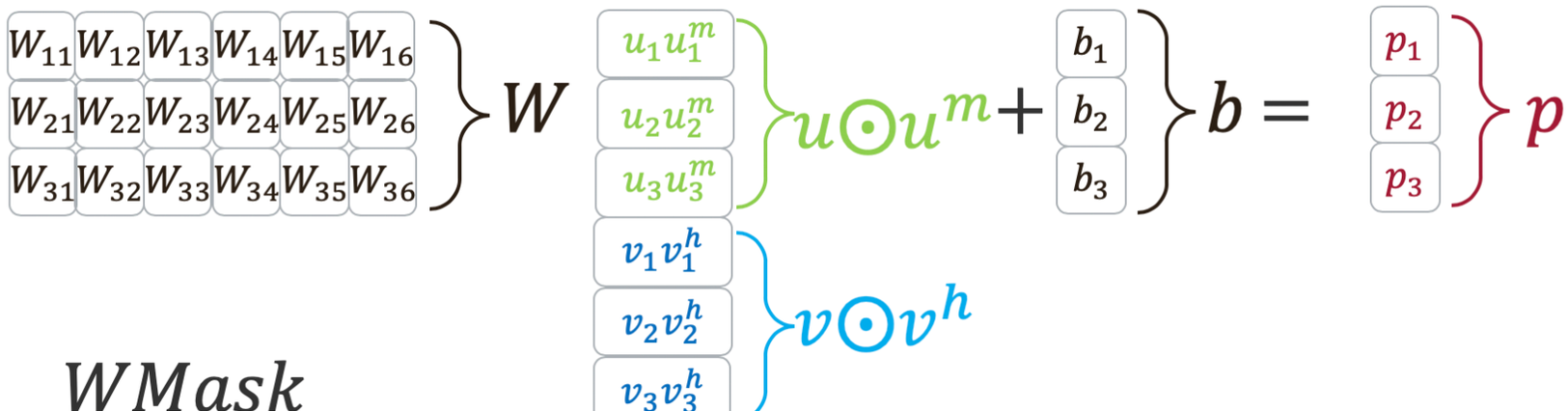
II.

MASK COMPOSITION – DIMA, 2015 (2)

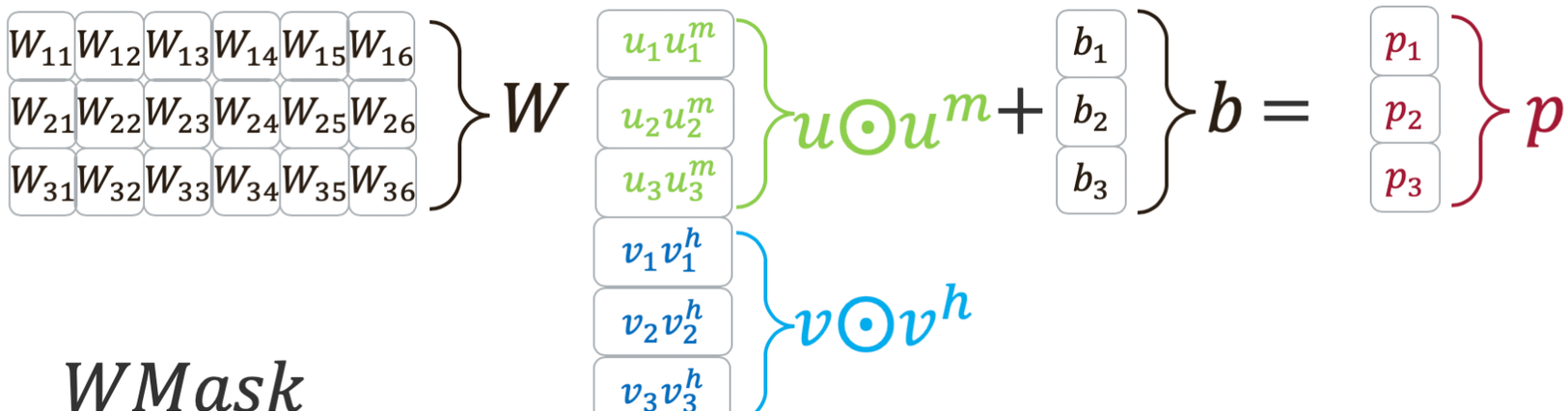


II.

MASK COMPOSITION – DIMA, 2015 (2)



MASK COMPOSITION – DIMA, 2015 (2)



$$WMask\text{: } p_i = W_{i*} \cdot [u \odot u^m \,; v \odot v^h\,] + b_i,\ \forall i \in \{1,2,3\}$$

$$AddMask\text{: } p_i = u \odot u^m + v \odot v^h$$

NEIGHBOURS OF WMASK COMPOSITION

disk__error:1	**currency__value:1**	**employee__manual:1**	**metal__armor:1**
disk__error 1.00000	currency__value 1.00000	employee__manual 1.00000	metal__armor 1.00000
disk__error__c 0.63742	currency__value__c 0.68259	employee__manual__c 0.69876	metal__armor__c 0.68537
hardware__error 0.62881	currency__rate 0.61195	company__attorney 0.59591	chain__armour 0.67611
program__error 0.56904	exchange__rate 0.54930	apple__fritter 0.58195	face__guard 0.65524
system__error 0.54074	currency__depreciation 0.53445	redundancy__policy 0.57816	bicycle__clip 0.64259
head__crash 0.52071	equity__price 0.49077	intelligence__cell 0.57682	bathing__trunks 0.63836
computer__error 0.51923	currency__policy 0.49042	x-ray__scan 0.57464	bathing__cap 0.61547
church__leader:3	**majority__party:3**	**picnic__table:3**	**sweetheart__deal:3**
church__leader 1.00000	majority__party 1.00000	picnic__table 1.00000	sweetheart__deal 1.00000
christian__leader 0.74565	minority__party 0.73798	fire__pit 0.72293	backroom__deal 0.65540
church__member 0.69506	party__caucus 0.60063	picnic__area 0.69909	barter__deal 0.54495
church__leader__c 0.64688	majority__party__c 0.49051	picnic__table__c 0.60321	sweetheart__deal__c 0.54421
church__official 0.61824	opposition__party 0.47730	picnic 0.58632	oil__deal 0.51831
clergy 0.61303	party__leader 0.46938	barbecue__pit 0.55666	business__deal 0.51084
community__leader 0.56897	majority__coalition 0.46083	basketball__court 0.54764	marketing__deal 0.48358
screen__price:1000	**police__lineup:1000**	**fossil__oil:1000**	**deed__poll:1000**
screen__price 1.00000	police__lineup 1.00000	fossil__oil 1.00000	deed__poll 1.00000
coach__fare 0.56077	bank__guard 0.57622	import__system 0.68093	marriage__certificate 0.53045
one-off__charge 0.53977	mopping-up__operation 0.57314	banana__export 0.65954	family__name 0.46612
manufacturing__level 0.51958	police__sweep 0.57112	price__distribution 0.64716	company__name 0.44564
seat__price 0.51609	surprise__inspection 0.56619	self__loader 0.64239	birth__certificate 0.40850
carriage__charge 0.51505	trash__barrel 0.56103	trading__transaction 0.64030	given__name 0.39348
…	…	…	…
screen__price__c 0.31463	police__lineup__c 0.32903	fossil__oil__c 0.37813	deed__poll__c 0.10163

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - $|V| = 100$ words; d (size of word representation) $= 200$

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - $|V|$ = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - $|V|$ = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - $|V|$ = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
  - total: 120,200 parameters

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
      - total: 120,200 parameters
  - |V|= 10,000 words; d=200

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
      - total: 120,200 parameters
  - |V|= 10,000 words; d=200
    - W_M, W_H size:

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - $|V|$ = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
      - total: 120,200 parameters
- $|V|$= 10,000 words; d=200
  - W_M, W_H size:
    - 10,000 x (200 + 200) = 10,000 x 400 = 4,000,000

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
      - total: 120,200 parameters
- |V|= 10,000 words; d=200
  - W_M, W_H size:
    - 10,000 x (200 + 200) = 10,000 x 400 = 4,000,000
  - W size: 200 x 400 + 200 = 80,200

MASK MODELS ALLEVIATE THE SIZE ISSUE OF FULLLEX

- the number of parameters still increases with the size of the vocabulary
  - |V| = 100 words; d (size of word representation) = 200
    - W_M, W_H size: 100 x (200 + 200) = 100 x 400 = 40,000
    - W size: 200 x 400 + 200 = 80,200
    - total: 120,200 parameters
- |V|= 10,000 words; d=200
  - W_M, W_H size:
    - 10,000 x (200 + 200) = 10,000 x 400 = 4,000,000
  - W size: 200 x 400 + 200 = 80,200
  - total: 4,080,000 parameters

INTRINSIC EVALUATION (5)

	English Nominal Compounds	
Composition Model	Q1, Q2, Q3	<=5
Addition	2,7,38	46,14 %
Weighted Addition	2,7,38	46,14 %
Matrix	1,2,9	67,37 %
WMask	1,2,7	71,53 %
FullLex	**1,2,7**	**72,82 %**

ISSUES WITH FULLLEX/MASK

ISSUES WITH FULLLEX/MASK

- the word matrices/masks are “trained” only for those words in the training data

ISSUES WITH FULLLEX/MASK

- the word matrices/masks are “trained” only for those words in the training data
- for words that are not seen during training, fulllex/mask models are equivalent to

ISSUES WITH FULLLEX/MASK

- the word matrices/masks are “trained” only for those words in the training data
- for words that are not seen during training, fulllex/mask models are equivalent to
  - the matrix model!

ISSUES WITH FULLLEX/MASK (3)

ISSUES WITH FULLLEX/MASK (3)

- the word matrices/masks are individual - no sharing of parameters between similar words: e.g. ‘blue dress’ ‘sky-blue dress’

ISSUES WITH FULLLEX/MASK (3)

- the word matrices/masks are individual - no sharing of parameters between similar words: e.g. ‘blue dress’ ‘sky-blue dress’
- if words are rare in training, their matrices/masks will get less updates

ISSUES WITH FULLLEX/MASK (3)

- the word matrices/masks are individual - no sharing of parameters between similar words: e.g. ‘blue dress’ ‘sky-blue dress’
- if words are rare in training, their matrices/masks will get less updates
- they will remain closer to the identity matrices/one vector, and therefore have less of an impact on the final representation

ISSUES WITH FULLLEX/MASK (3)

- the word matrices/masks are individual - no sharing of parameters between similar words: e.g. ‘blue dress’ ‘sky-blue dress’
- if words are rare in training, their matrices/masks will get less updates
- they will remain closer to the identity matrices/one vector, and therefore have less of an impact on the final representation
- the amount of updates that the matrix/mask of each word will get depends on its train frequency

ISSUES WITH FULLLEX/MASK (3)

- the word matrices/masks are individual - no sharing of parameters between similar words: e.g. ‘blue dress’ ‘sky-blue dress’
- if words are rare in training, their matrices/masks will get less updates
- they will remain closer to the identity matrices/one vector, and therefore have less of an impact on the final representation
- the amount of updates that the matrix/mask of each word will get depends on its train frequency
- cannot use the fact that ‘blue’ and ‘sky-blue’ are colors

REFERENCES

- Jeff Mitchell and Mirella Lapata. 2010. *Composition in Distributional Models of Semantics*. Cognitive Science 34 (2010) 1388–1429.
- Richard Socher, Christopher Manning and Andrew Ng. 2010. *Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks*. In Proceedings of NIPS-2010 Deep Learning Workshop.
- Richard Socher, Brody Huval, Christopher Manning and Andrew Ng. 2012. *Semantic Compositionality through Recursive Matrix-Vector Spaces*. In Proceedings of EMNLP-CoNLL 2012.
- Corina Dima. 2015. Reverse-engineering Language: *A Study on the Semantic Compositionality of German Compounds*. In Proceedings of EMNLP 2015.

IMAGE CREDITS

https://pixabay.com/photos/code-geek-talk-code-to-me-coffee-cup-2680204/

https://commons.wikimedia.org/wiki/File:Hand.svg

https://en.wikipedia.org/wiki/File:Kewpie_doll.jpg

https://bar.wikipedia.org/wiki/Datei:Hand_Puppet.jpg

https://de.wikipedia.org/wiki/Datei:Theater_Baden-Baden-22-gje.jpg

https://commons.wikimedia.org/wiki/File:Bamboccianti_c1700_Theaterauff%C3%BChrung_im_Freien.jpg

https://commons.wikimedia.org/wiki/File:Bundesarchiv_Bild_183-76973-0003,_Leipzig,_Opernhaus,_Auff%C3%BChrung_%22Dornr%C3%B6schen%22.jpg

https://pixabay.com/de/photos/kasperle-puppentheater-puppe-kasper-677526/