Word2Vec and all the things

Data Labs @
StitchFix

About

@chrisemoody

Former astrophysicist, supercomputing

Data Labs at Stitch Fix

Fun stuff with Gaussian Processes,
t-SNE & word2vec

Credit

Large swathes of this talk are from previous presentations by:

- Tomas Mikolov
- Christopher Olah
- Radim Rehurek
- Omer Levy & Yoav Goldberg
- Richard Socher
- Xin Rong

1 word2vec

2 Stitch Fix

3 Fun Stuff

word2vec

1. *king* - *man* + *woman* = *queen*
2. Learns from raw text
3. Huge splash in NLP world
4. Pretty simple algorithm
5. Comes pretrained

Predict current word given previous words

$$P(W) = \prod_i P(w_i | w_1 \dots w_{i-1})$$

What if we estimate $\boldsymbol{P}(w)$ empirically?

How far back should $w_1 \dots w_{i-1}$ go?

Q

$$\mathrm{argmax}[\mathrm{p}(\mathrm{w} \mid \mathrm{w}_0 \ldots \mathrm{w}_\mathrm{k})]$$

Q argmax[p(w | ‘now’, ‘with’,’a’,’new’,’model’)]

If context is too **long** you overfit....

Q argmax[p(w | ‘now’, ‘with’,’a’,’new’,’model’)]

A ‘GM’

now with a new model GM

If context is too **long** you overfit....

Q argmax[p(w | ‘with’,’a’,’new’,’model’,’GM’)]

A ‘is’

now with a new model GM is

If context is too **long** you overfit....

Q	argmax[p(w | 'a','new','model','GM','is')]
A	‘trying’

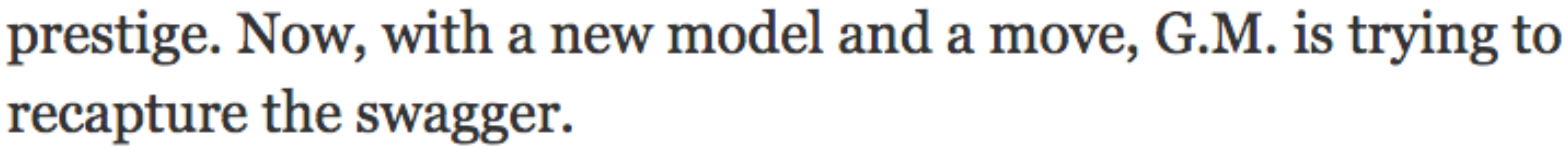
now with a new model GM is trying

If context is too **long** you overfit....

Q	argmax[p(w |'new','model','GM','is','trying')]
A	'to'

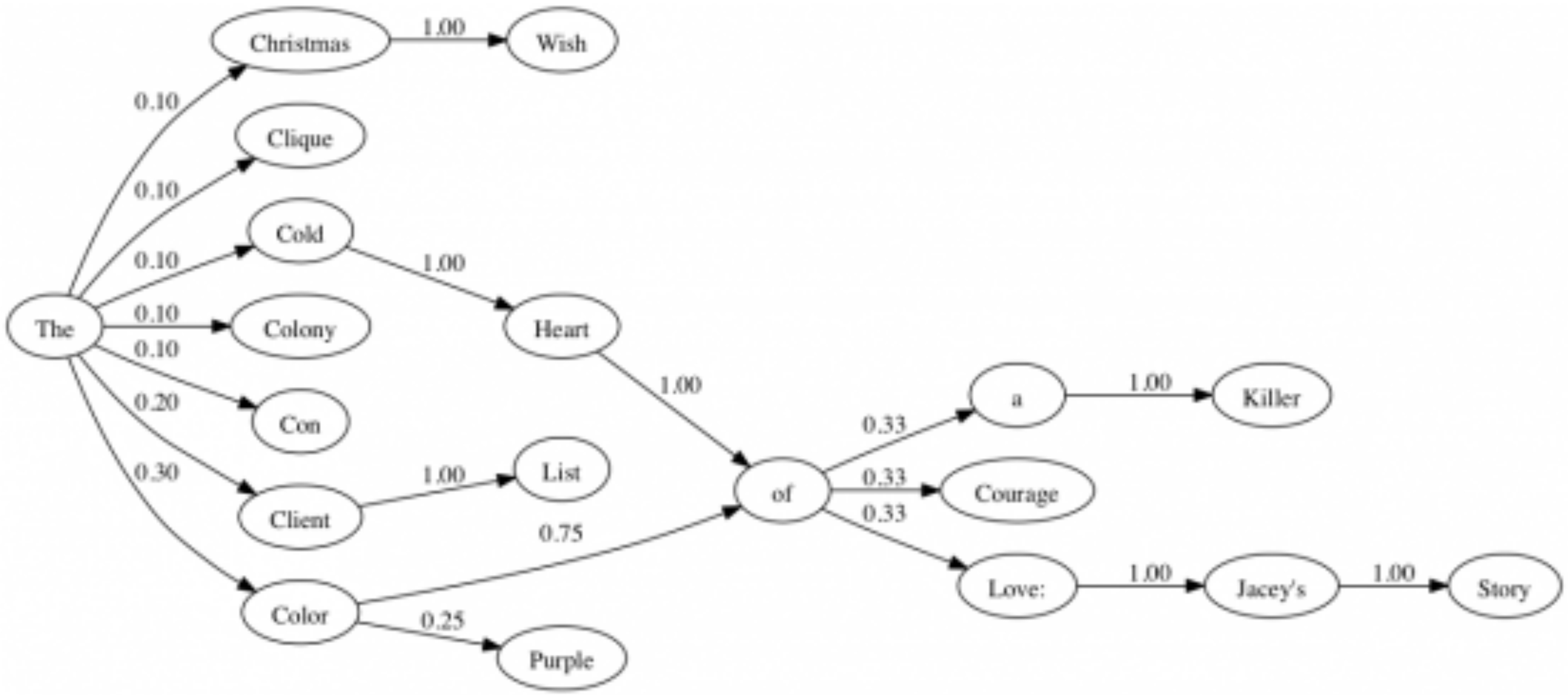
now with a new model GM is trying to

Long n-grams are unique, so we're just memorizing one example.

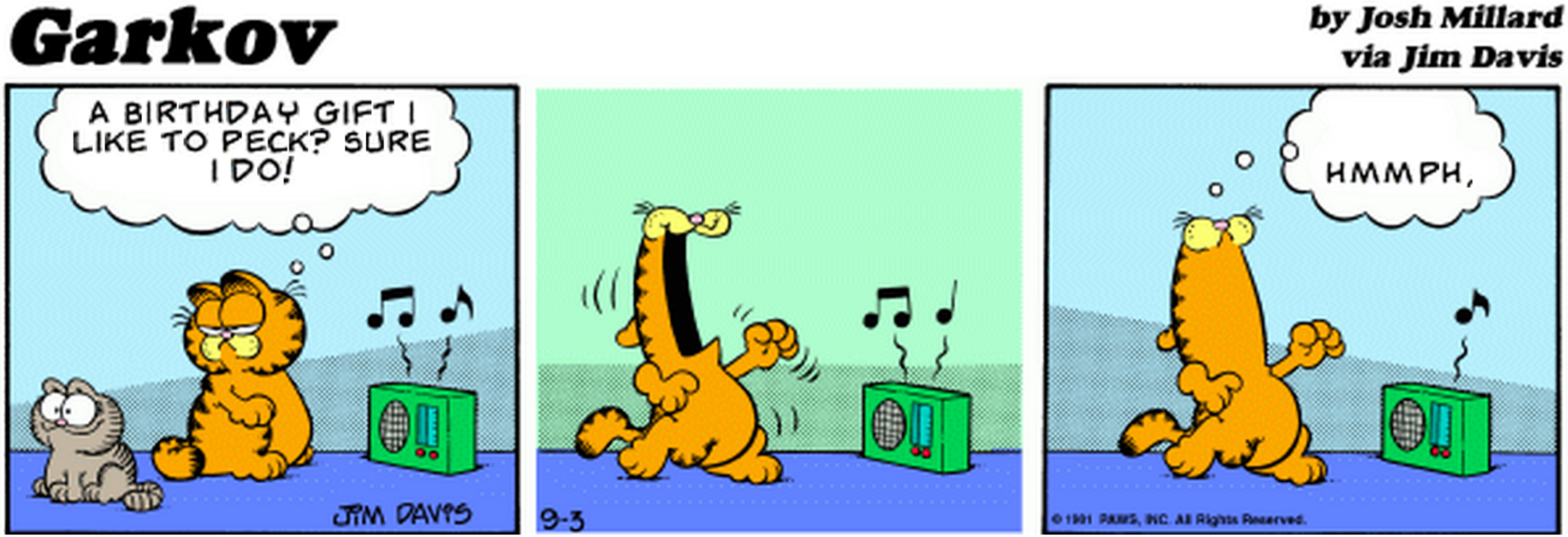


“ prestige. Now, with a new model and a move, G.M. is trying to recapture the swagger. ”

If context is too **short** your sentences don't make sense...



...just because the current word *only* depends on the previous word. (but it usually works *grammatically*)



(this is how most
internet chatbots work)

*checkout Garkov

Idea: replace one-hot coded words with **dense vectors***

$$\begin{aligned} V_{\text{run}} &= [0,\ 0,\ 1,\ 0,\ 0] \\ V_{\text{running}} &= [0,\ 1,\ 0,\ 0,\ 0] \end{aligned}$$

$$\downarrow$$

$$\begin{aligned} V_{\text{run}} &= [0.2,\ 0.3,\ \text{-.1},\ \text{-.9},\ 0.0] \\ V_{\text{running}} &= [0.7,\ 1,\ \text{-.7},\ 0.5,\ 0.1] \end{aligned}$$

*or **distributed representations** Bengio et al.

Idea: replace one-hot coded words with **dense vectors***

$$V_{\text{cat}} = [0.2, 0.3, -.1, -.9, 0.0]$$
$$V_{\text{dog}} = [0.7, 1, -.7, 0.5, 0.1]$$

- *Locality* — properties change gradually from word to word

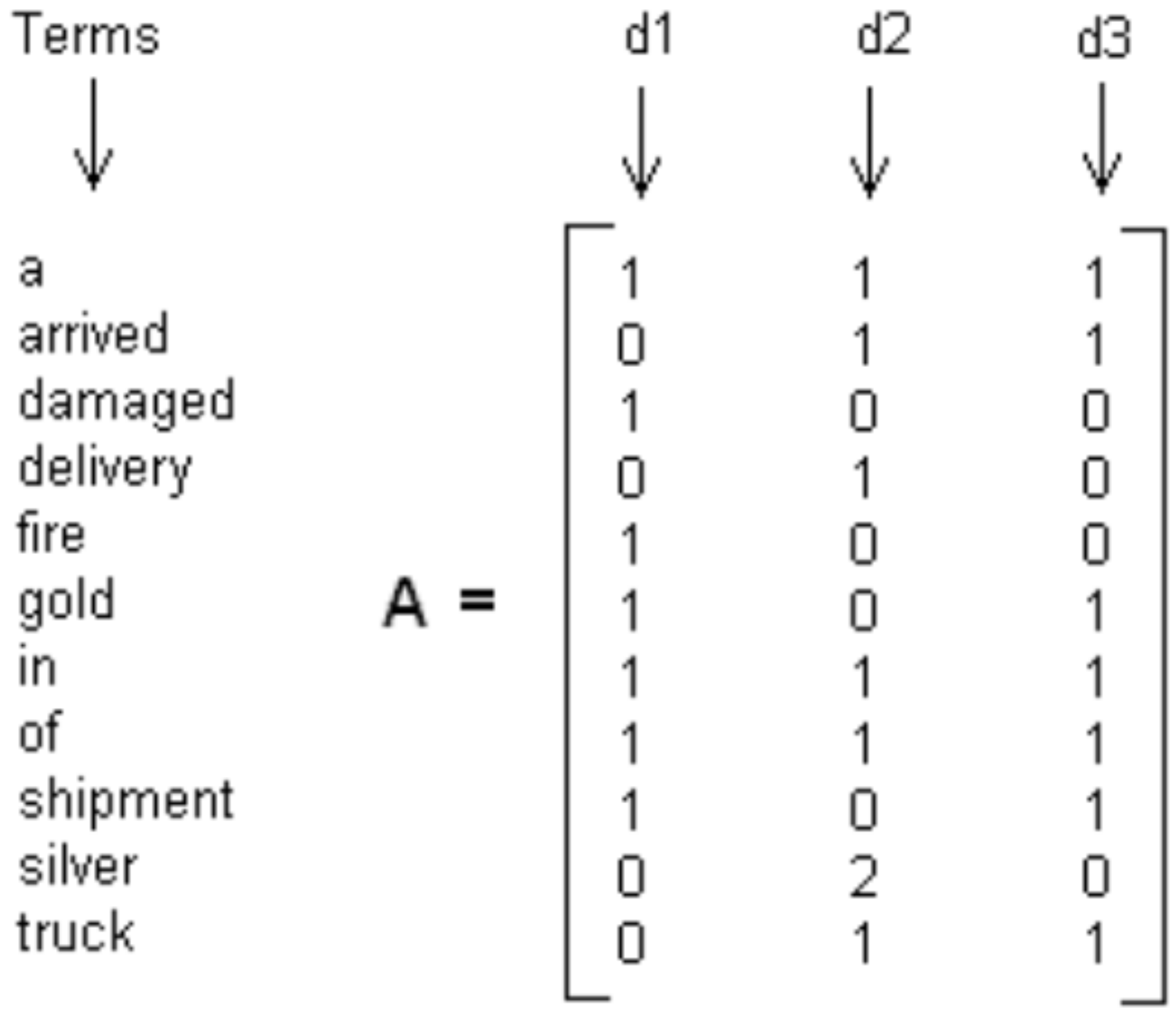
quickly ~ *quick*

- *Regularity* — directions will be meaningful & consistent

king - *man* + *woman* = *queen*

*see more by Rumelhart, Collobert, Bengio

Build a co-occurence matrix.

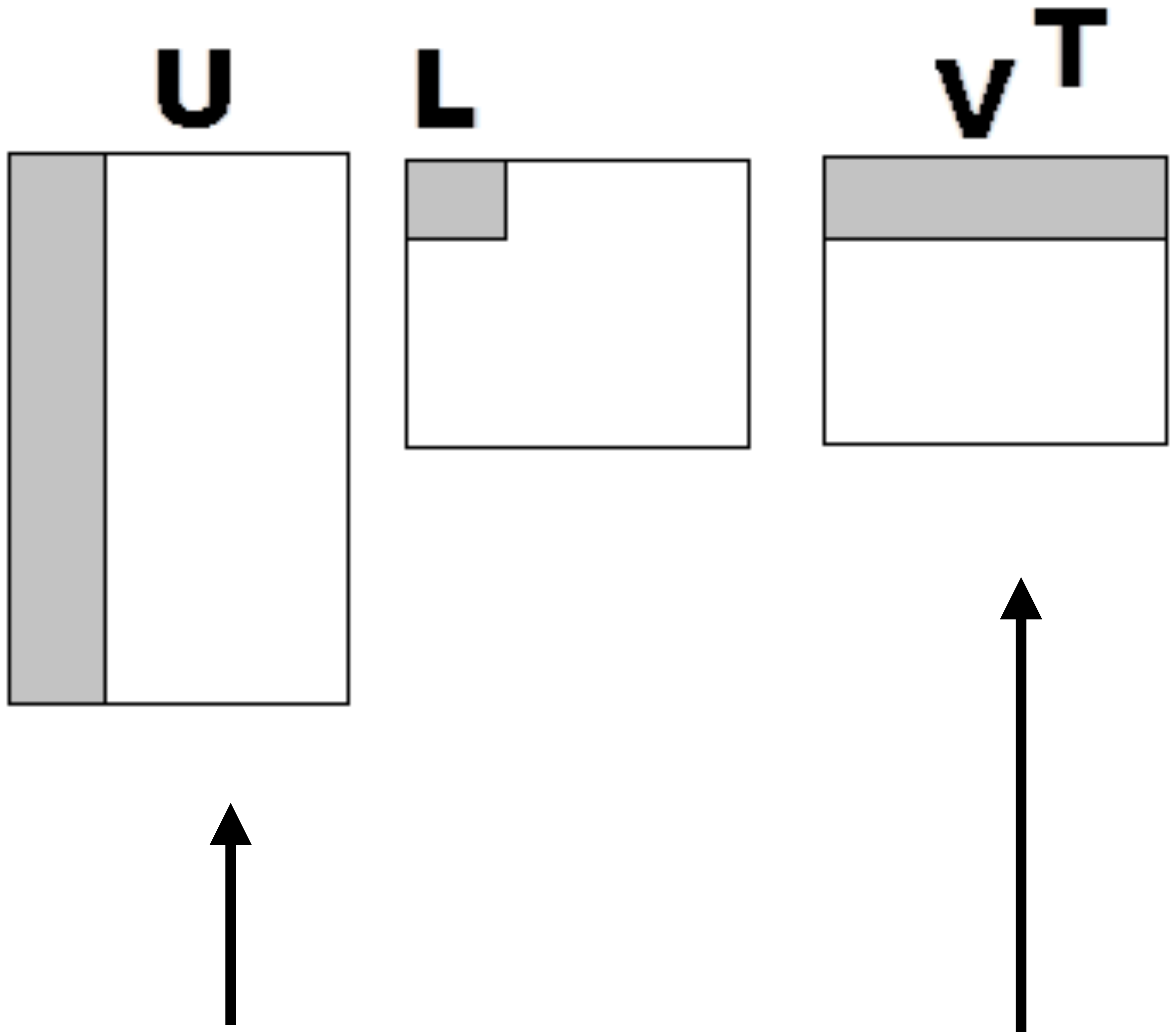


$$\begin{array}{l} \text{Terms} \\ \text{a} \\ \text{arrived} \\ \text{damaged} \\ \text{delivery} \\ \text{fire} \\ \text{gold} \\ \text{in} \\ \text{of} \\ \text{shipment} \\ \text{silver} \\ \text{truck} \end{array} \quad \begin{array}{c} \\ \\ \\ \\ \\ \\ \mathbf{A} = \end{array} \begin{array}{ccc} d1 & d2 & d3 \\ \downarrow & \downarrow & \downarrow \\ \end{array}$$

$$\begin{array}{l|ccc} \text{Terms} & d1 & d2 & d3 \\ \hline \text{a} & 1 & 1 & 1 \\ \text{arrived} & 0 & 1 & 1 \\ \text{damaged} & 1 & 0 & 0 \\ \text{delivery} & 0 & 1 & 0 \\ \text{fire} & 1 & 0 & 0 \\ \text{gold} & 1 & 0 & 1 \\ \text{in} & 1 & 1 & 1 \\ \text{of} & 1 & 1 & 1 \\ \text{shipment} & 1 & 0 & 1 \\ \text{silver} & 0 & 2 & 0 \\ \text{truck} & 0 & 1 & 1 \end{array}$$

Let’s SVD the co-occurence matrix.

$$\mathbf{A} = \begin{array}{l|ccc} \text{Terms} & \text{d1} & \text{d2} & \text{d3} \\ \hline \text{a} & 1 & 1 & 1 \\ \text{arrived} & 0 & 1 & 1 \\ \text{damaged} & 1 & 0 & 0 \\ \text{delivery} & 0 & 1 & 0 \\ \text{fire} & 1 & 0 & 0 \\ \text{gold} & 1 & 0 & 1 \\ \text{in} & 1 & 1 & 1 \\ \text{of} & 1 & 1 & 1 \\ \text{shipment} & 1 & 0 & 1 \\ \text{silver} & 0 & 2 & 0 \\ \text{truck} & 0 & 1 & 1 \end{array} =$$



Item-doc counts

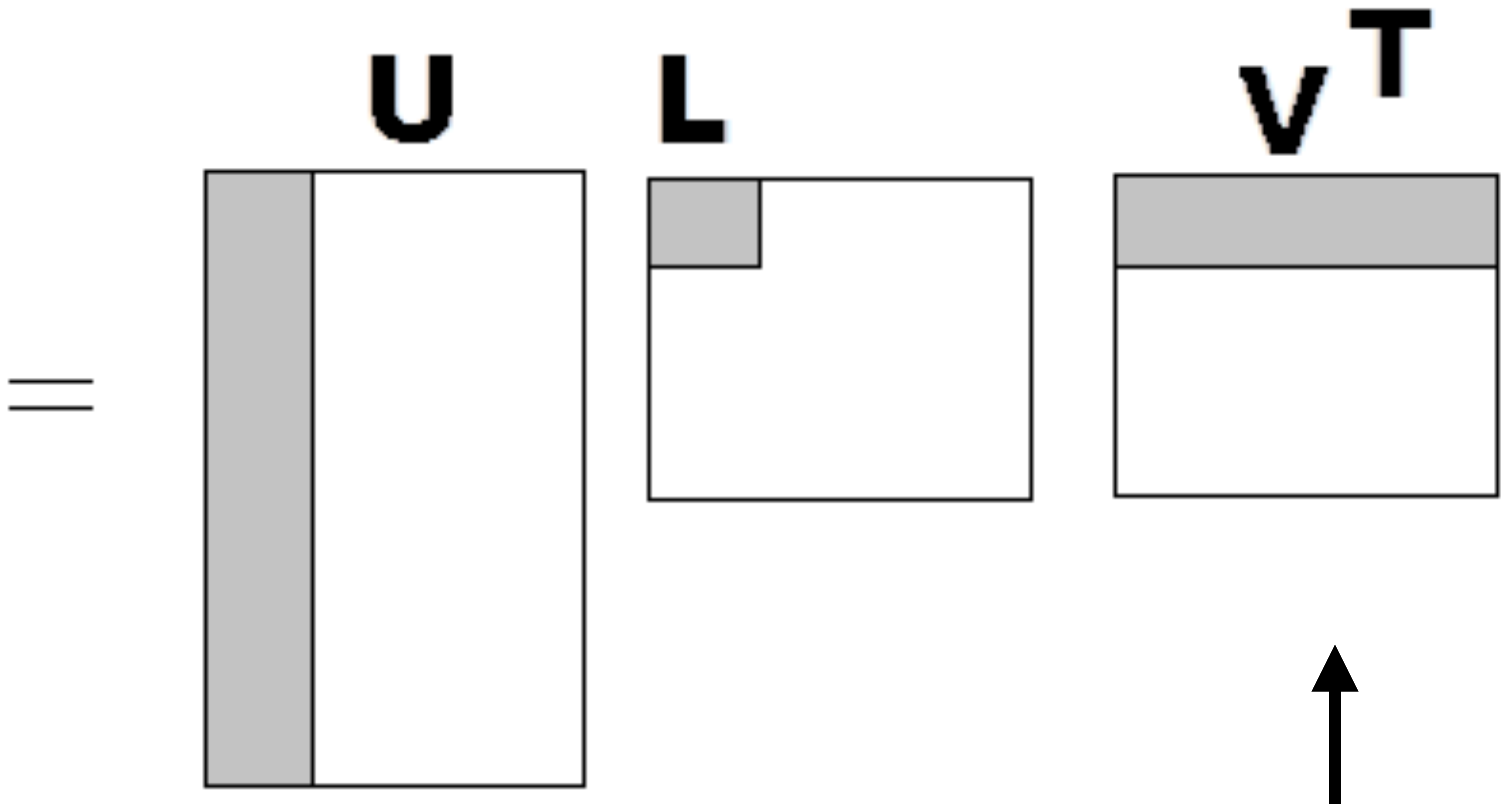
word vectors

document vectors

Let’s SVD the co-occurence matrix.

$$\begin{array}{l|ccc} \text{Terms} & \text{d1} & \text{d2} & \text{d3} \\ \hline \text{a} & 1 & 1 & 1 \\ \text{arrived} & 0 & 1 & 1 \\ \text{damaged} & 1 & 0 & 0 \\ \text{delivery} & 0 & 1 & 0 \\ \text{fire} & 1 & 0 & 0 \\ \text{gold} & 1 & 0 & 1 \\ \text{in} & 1 & 1 & 1 \\ \text{of} & 1 & 1 & 1 \\ \text{shipment} & 1 & 0 & 1 \\ \text{silver} & 0 & 2 & 0 \\ \text{truck} & 0 & 1 & 1 \end{array}$$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} = U\,L\,V^T$$



Item-doc counts
$(10^6, 10^4) \sim 10^{10}$

word vectors
$(10^6, 300) \sim 10^8$

document vectors
$(10^4, 300) \sim 10^6$

Doesn't scale.

$$\sim O(nm^2)$$

Hard to read more than a few millions of documents.

word2vec

1. Set up an objective function
2. Randomly initialize vectors
3. Do gradient descent

word2vec: learn word vector v_{in}
from it's surrounding context

$$v_{in}$$

"The fox jumped **over** the lazy dog"

Maximize the likelihood of seeing this **context** given the word *over*.

$$P(the|over)$$
$$P(fox|over)$$
$$P(jumped|over)$$
$$P(the|over)$$
$$P(lazy|over)$$
$$P(dog|over)$$

...instead of maximizing the likelihood of co-occurrence counts.

What should this be?

$$P(fox|over)$$

Should depend on the word vectors.

$$\begin{tikzcd} P(fox|over) \arrow[d] \\ P(v_{fox}|v_{over}) \end{tikzcd}$$

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$

"The fox jumped **over** the lazy dog"

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$

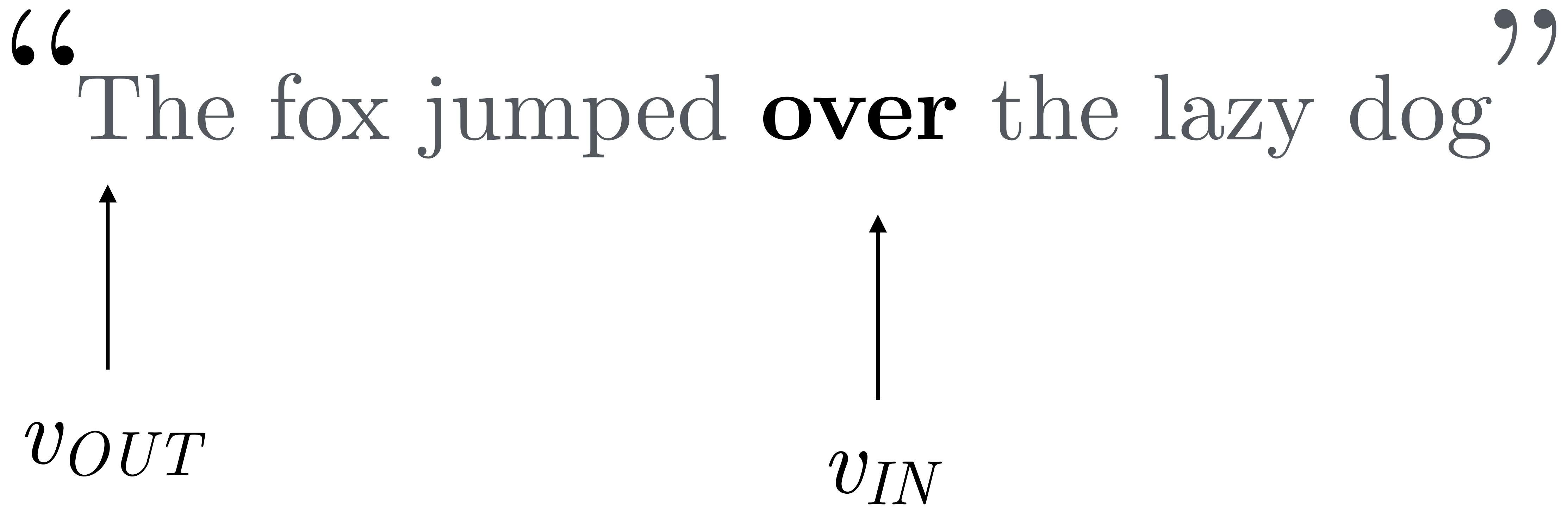
"The fox jumped **over** the lazy dog"

v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



"The fox jumped **over** the lazy dog"

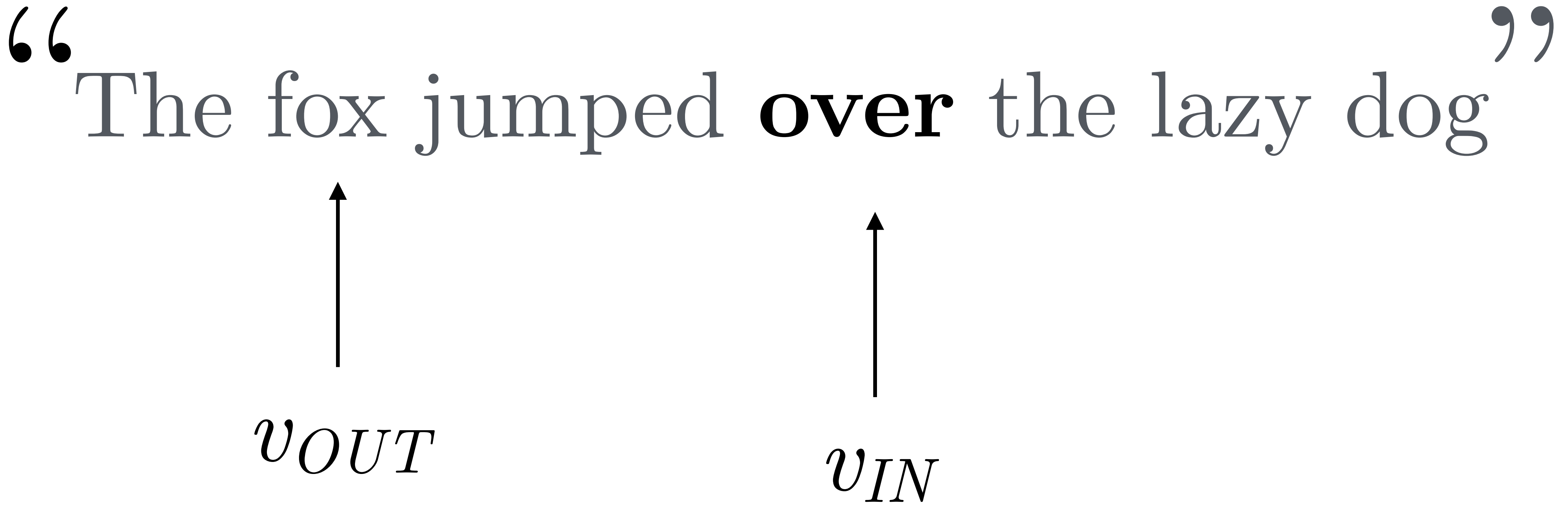
v_{OUT}

v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



"The fox jumped **over** the lazy dog"

v_{OUT}

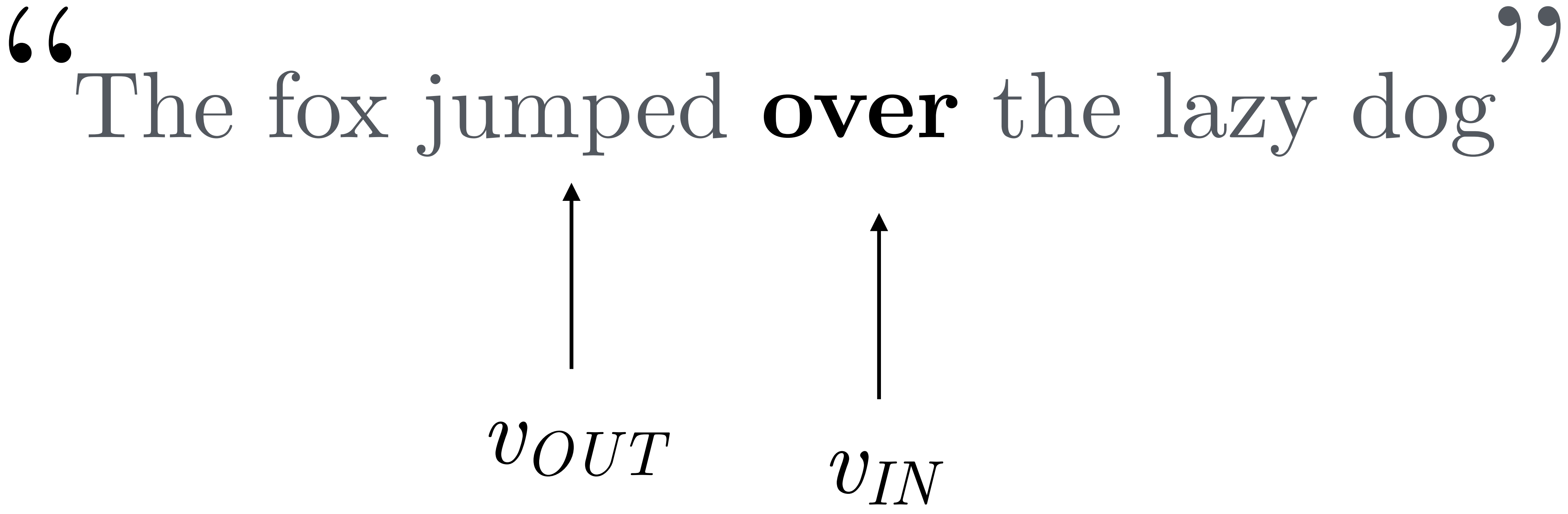
v_{IN}

word2vec

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



“The fox jumped **over** the lazy dog”

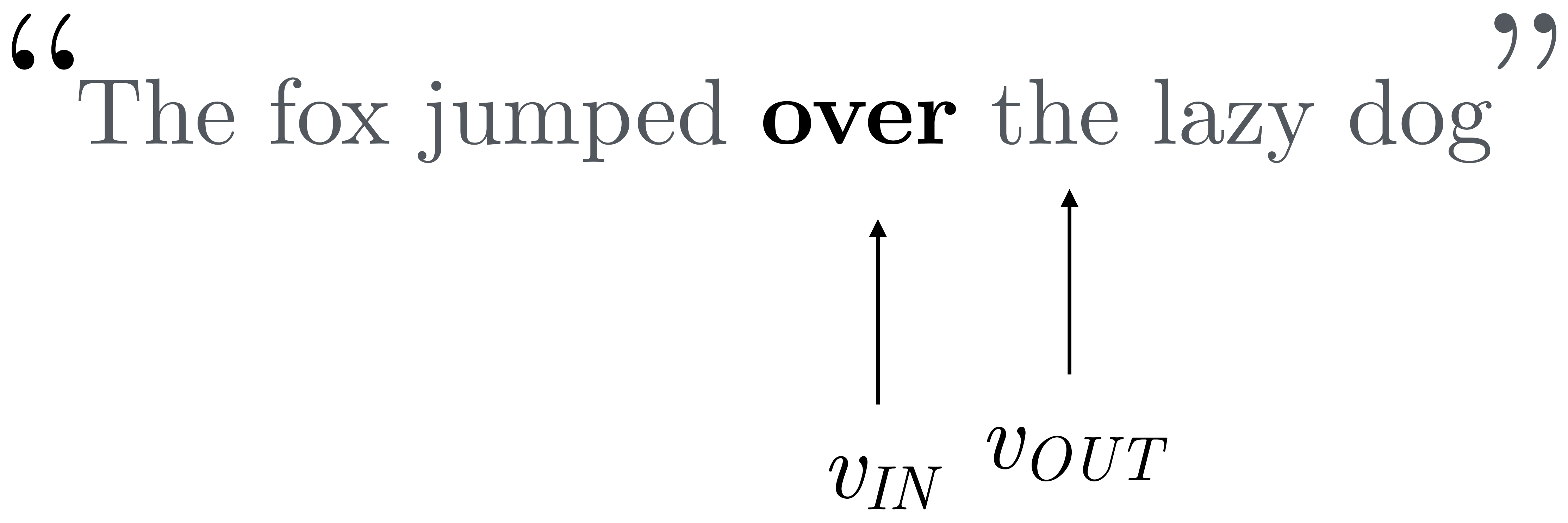
v_{OUT}

v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it’s the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$

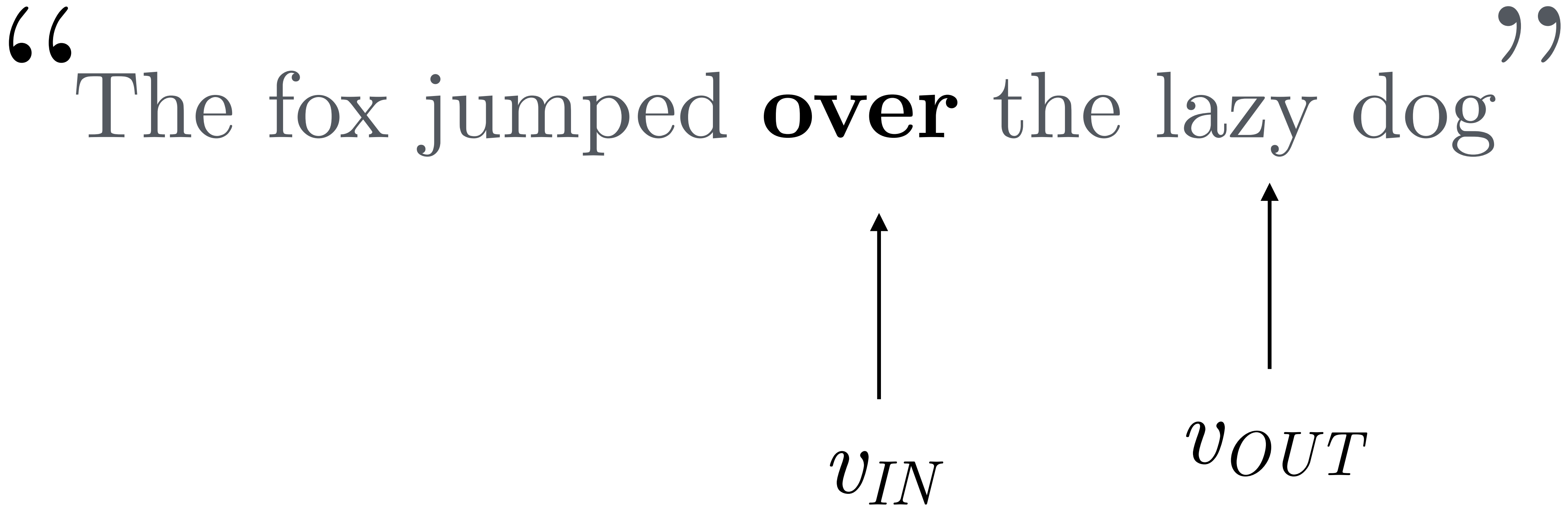


“The fox jumped **over** the lazy dog”

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



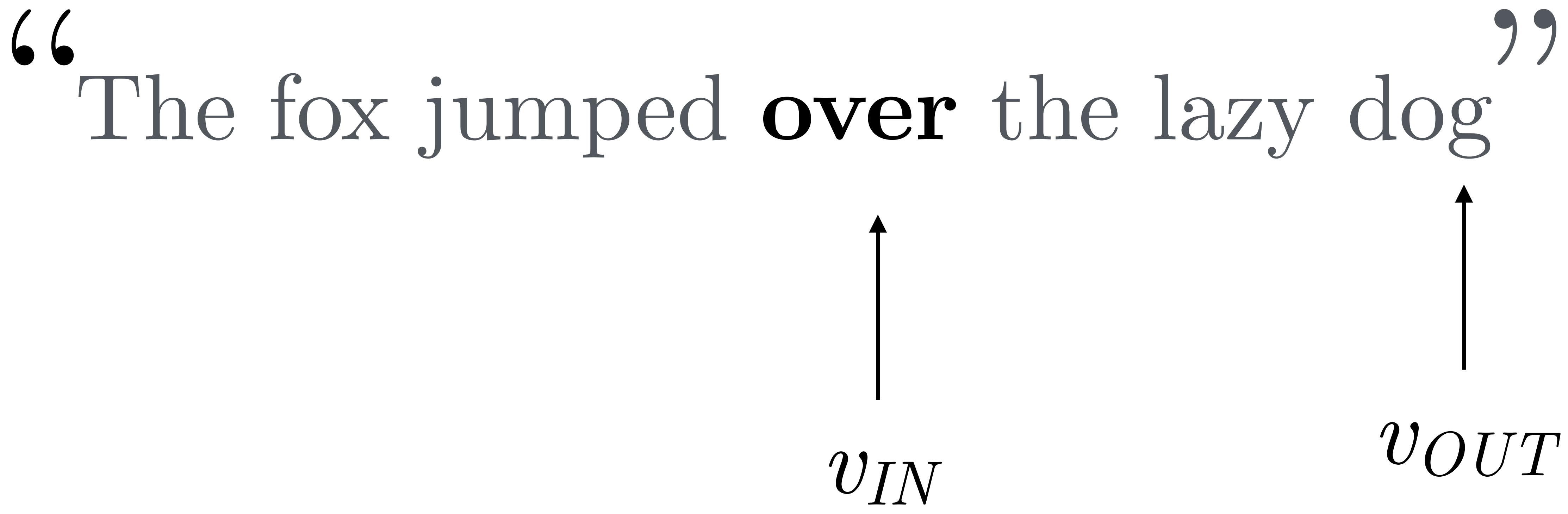
"The fox jumped **over** the lazy dog"

v_{IN} v_{OUT}

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



"The fox jumped **over** the lazy dog"

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$

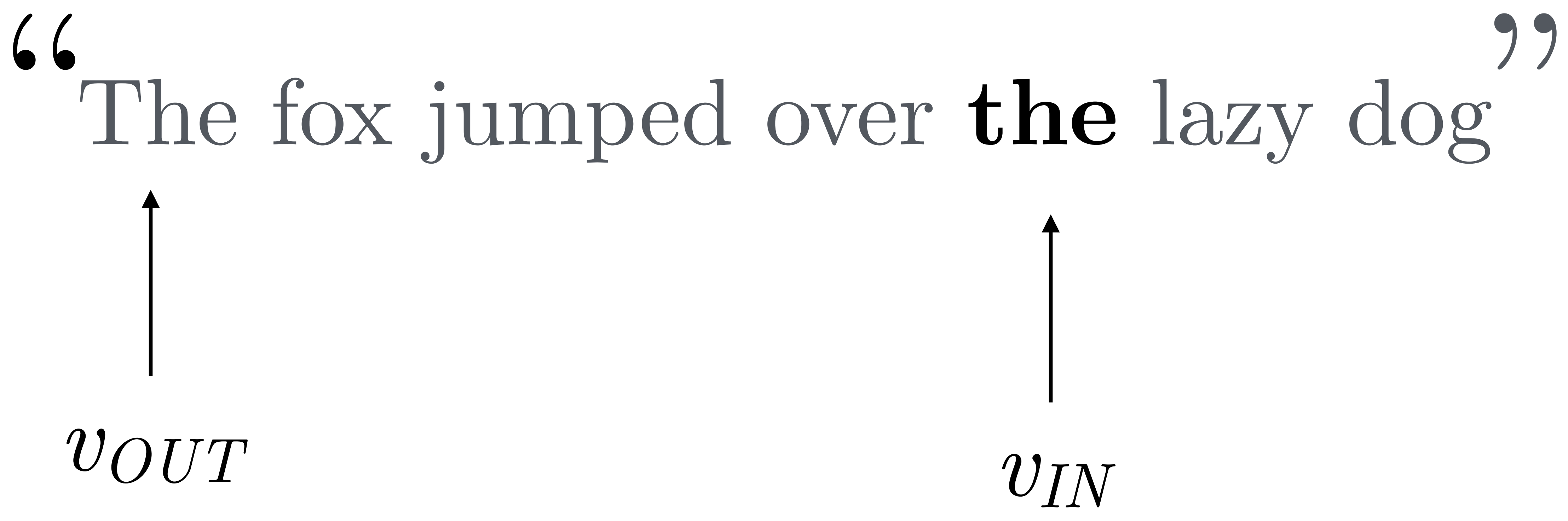
"The fox jumped over **the** lazy dog"

v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it’s the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



“The fox jumped over **the** lazy dog”

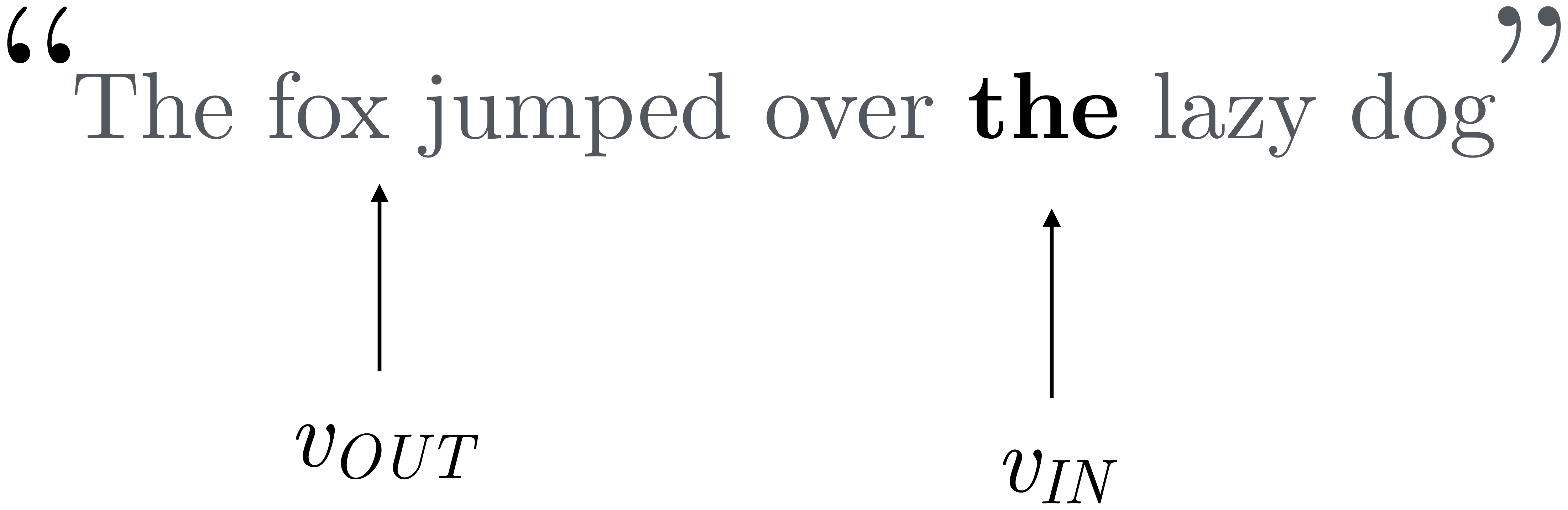
v_{OUT}

v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



"The fox jumped over **the** lazy dog"

v_{OUT}

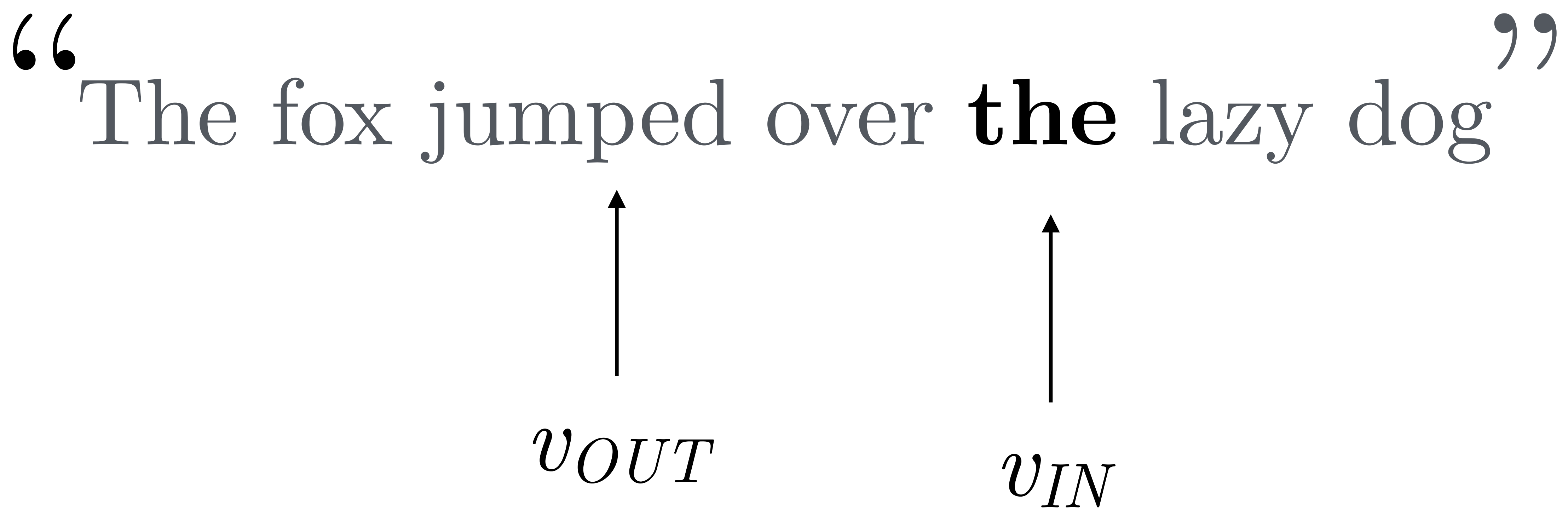
v_{IN}

word2vec

Twist: we have *two* vectors for every word.
Should depend on whether it’s the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



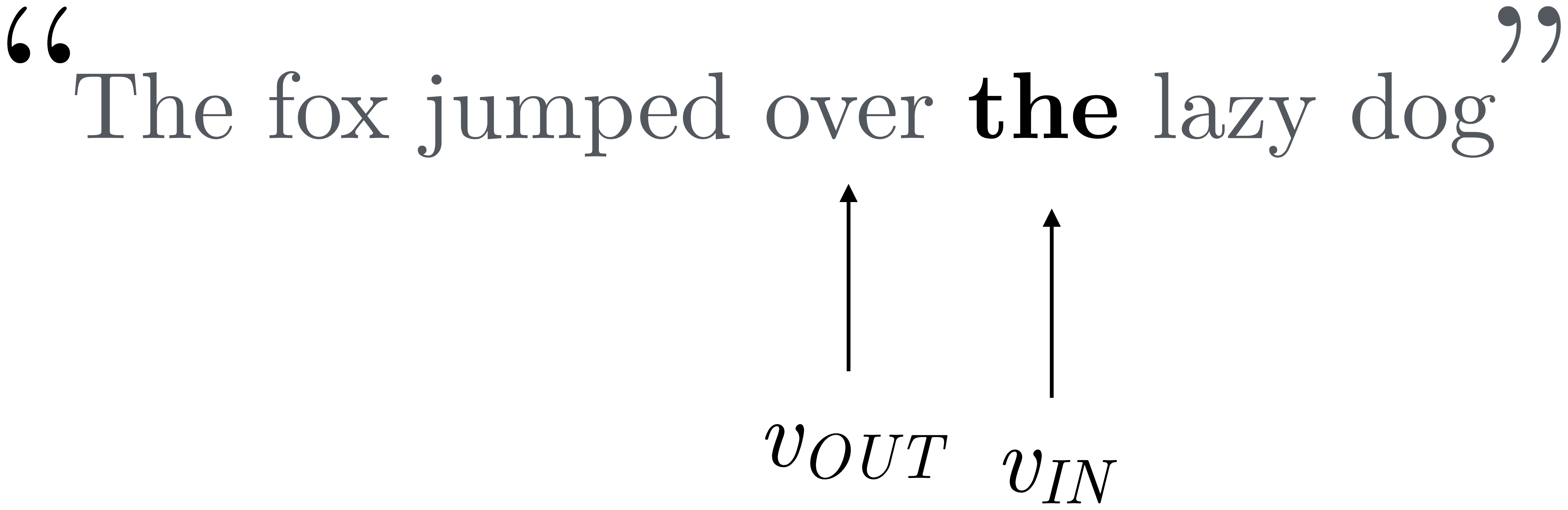
“The fox jumped over **the** lazy dog”

v_{OUT} v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



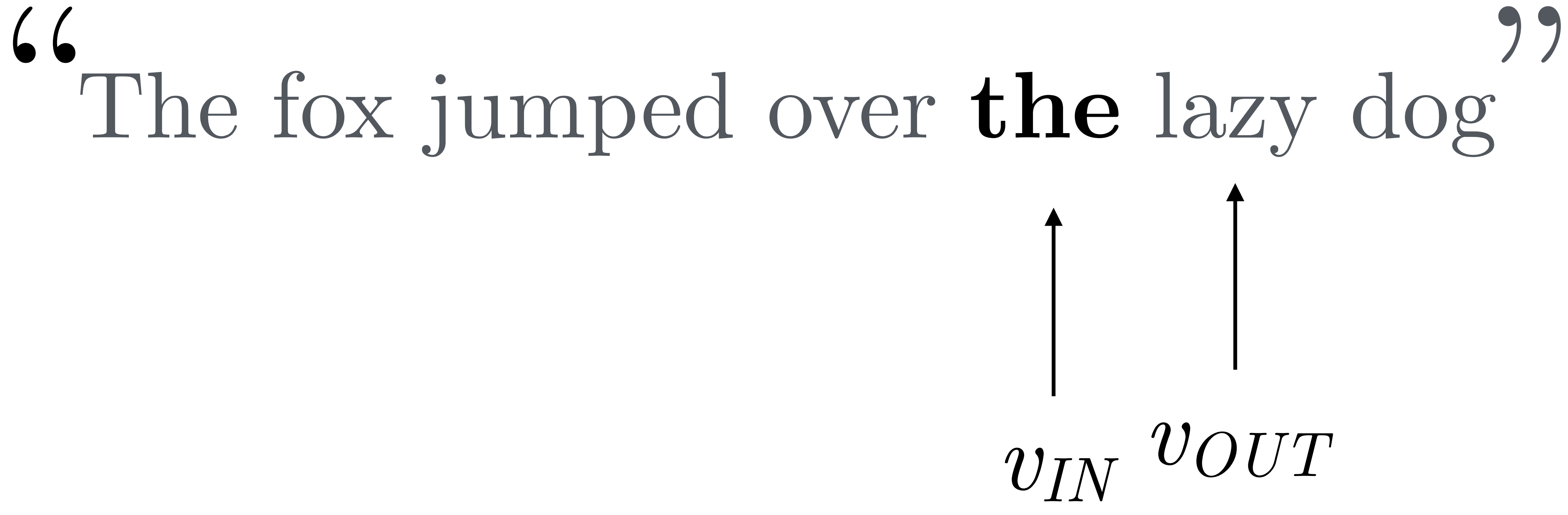
"The fox jumped over **the** lazy dog"

v_{OUT} v_{IN}

Twist: we have *two* vectors for every word.
Should depend on whether it’s the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



“The fox jumped over **the** lazy dog”

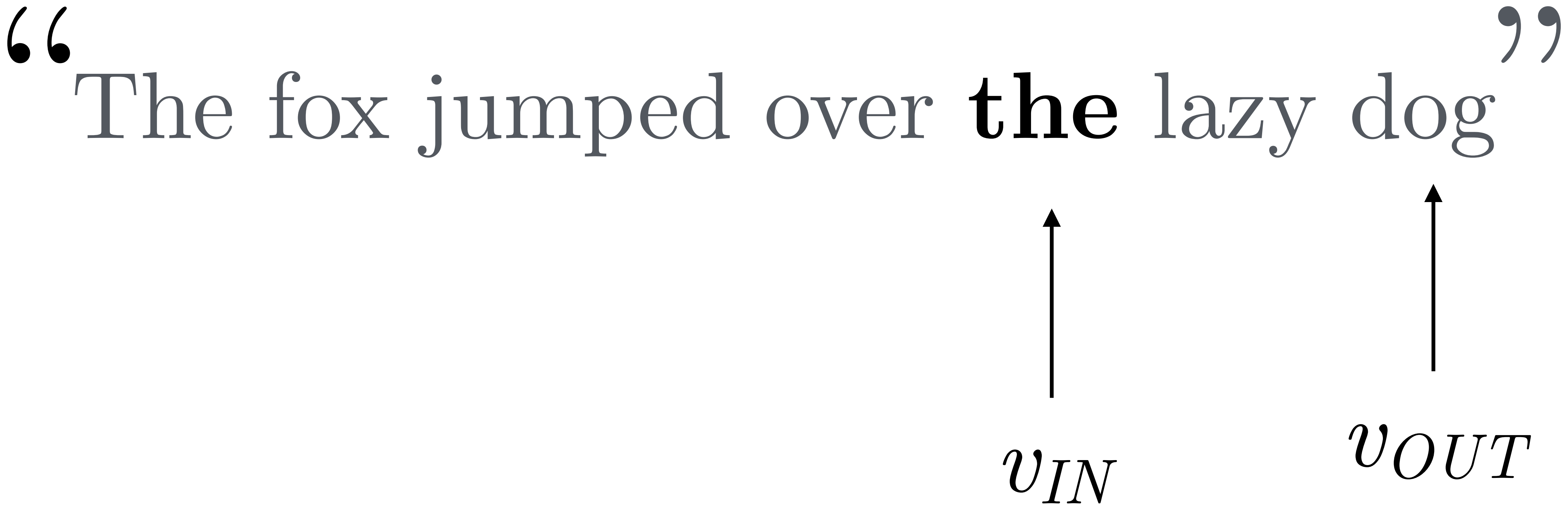
v_{IN} v_{OUT}

word2vec

Twist: we have *two* vectors for every word.
Should depend on whether it's the input or the output.

Also a *context* window around every input word.

$$P(v_{OUT}|v_{IN})$$



“The fox jumped over **the** lazy dog”

objective

How should we define $P(v_{OUT}|v_{IN})$?

Measure loss between
v_{IN} and v_{OUT}?

$$v_{in} \bullet v_{out}$$

objective

But we'd like to measure a probability.

$$v_{in} \bullet v_{out} \in [-1,1]$$

objective

But we'd like to measure a probability.

$$softmax(v_{in} \bullet v_{out}) \in [0,1]$$

objective

But we'd like to measure a probability.

$$softmax(v_{in} \bullet v_{out})$$

Probability of choosing 1 of N discrete items.
Mapping from vector space to a multinomial over words.

But we'd like to measure a probability.

$$softmax = \frac{exp(v_{in} \bullet v_{out})}{\Sigma_{k \in V} exp(v_{in} \bullet v_k)}$$

Normalization term over all words

objective

But we'd like to measure a probability.

$$softmax = \frac{exp(v_{in} \bullet v_{out})}{\sum_{k \in V} exp(v_{in} \bullet v_k)} = P(v_{out}|v_{in})$$

objective

Learn by gradient descent on the softmax prob.

For every example we see update v_{in}

$$v_{in} := v_{in} + \frac{\partial}{\partial v_{in}} P(v_{out}|v_{in})$$

$$v_{out} := v_{out} + \frac{\partial}{\partial v_{out}} P(v_{out}|v_{in})$$

performance

What's our performance?

$$\frac{exp(v_{in} \bullet v_{out})}{\sum_{k \in V} exp(v_{in} \bullet v_k)}$$

What's our performance?

$$\frac{exp(v_{in} \bullet v_{out})}{\underset{k \in V}{\Sigma} exp(v_{in} \bullet v_k)}$$

$O(V)$

What's our performance?

$$\frac{exp(v_{in} \bullet v_{out})}{\underset{k \in V}{\Sigma} exp(v_{in} \bullet v_k)}$$

$$O(V)$$

V operations for every update.

VC operations per input word.

VCN over the whole corpus.

performance

ಠ_ಠ

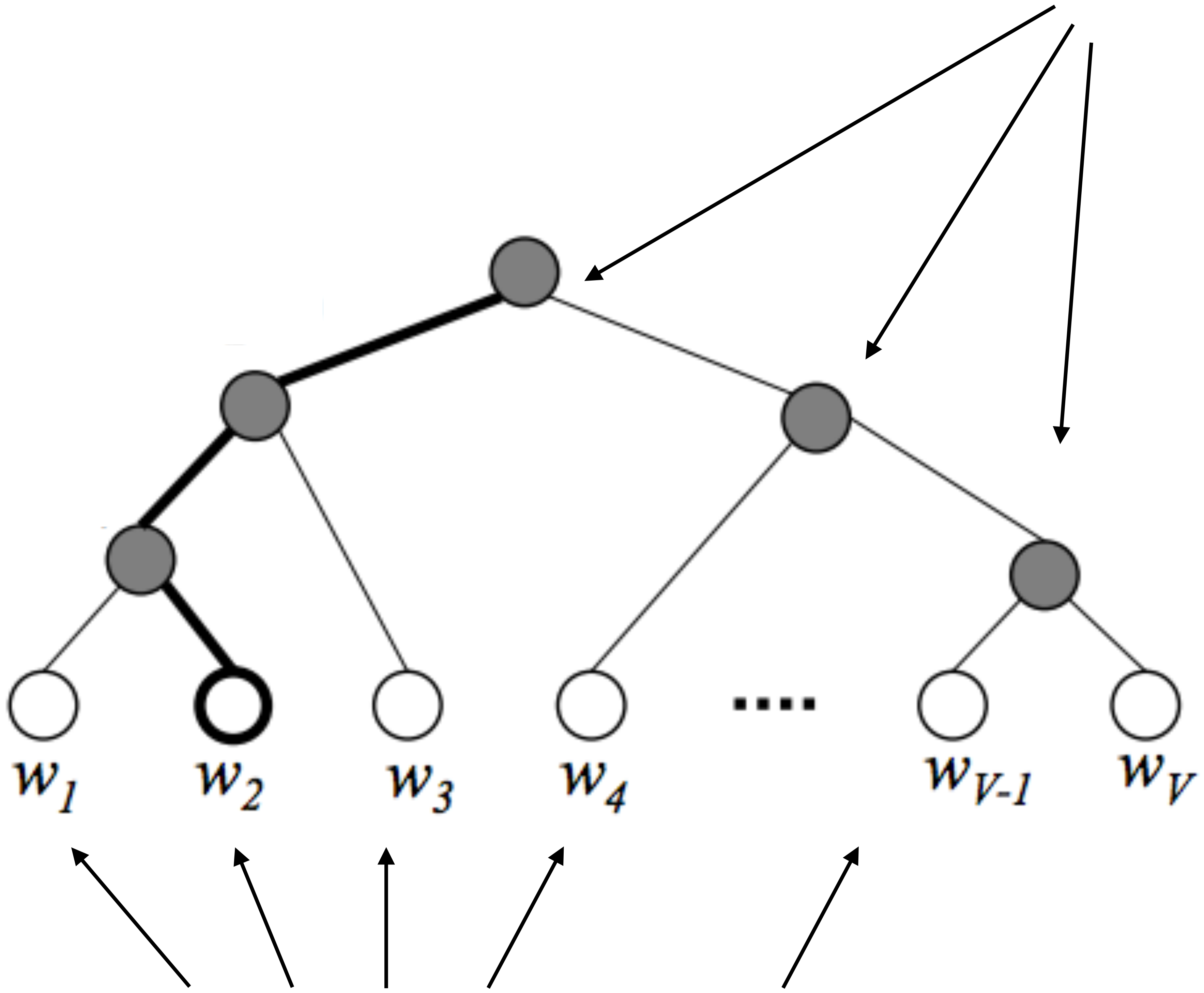
How is $O(VCN)$ supposed to
better than SVD that was $\mathrm{O}(\mathrm{NV}^2)$?

hierchical softmax

Have an $O(V)$ problem?

Build a tree and get a $O(log V)$ problem!

Hierarchical softmax



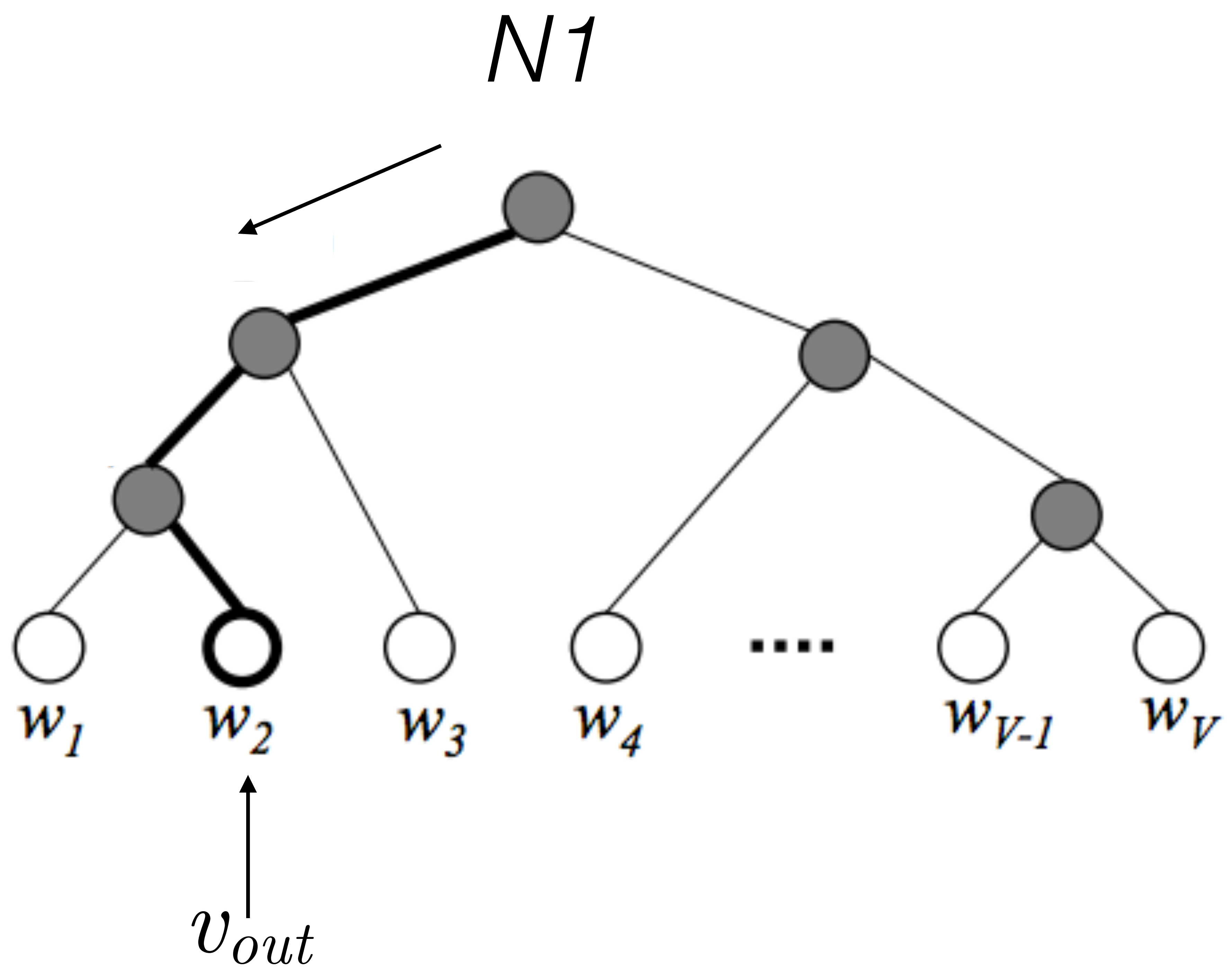
node vectors

V word vectors

*Bengio 2005

hierchical softmax

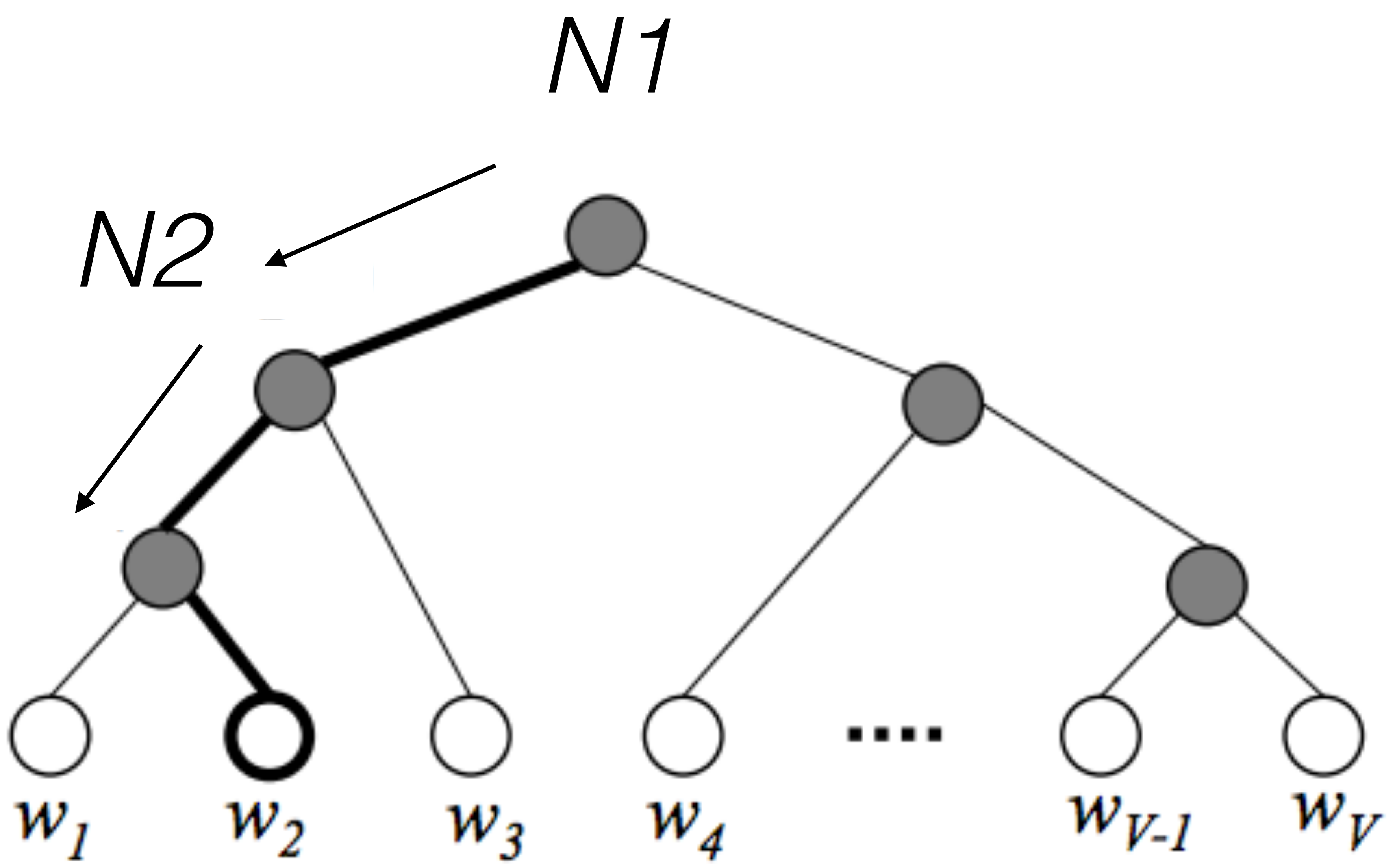
Hierarchical softmax



$$P(v_{out}|v_{in}) = \text{P(going left at N1}|\ v_{out})$$

*Bengio 2005

Hierarchical softmax

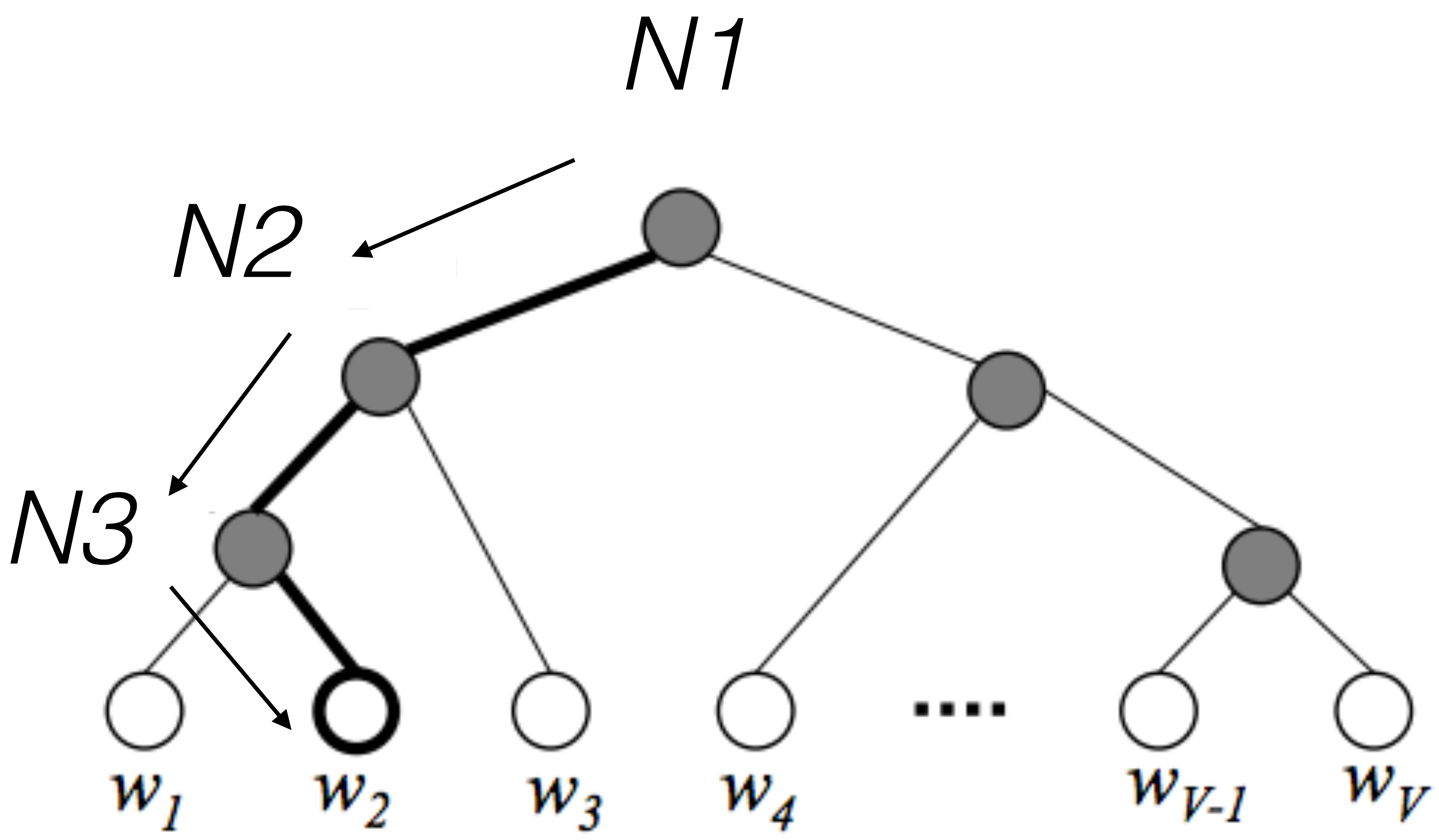


$$P(v_{out}|v_{in}) = \text{P(going left at N1}|\ v_{in})\ \text{P(going left at N2}|\ v_{in})$$

*Bengio 2005

hierchical softmax

Hierarchical softmax



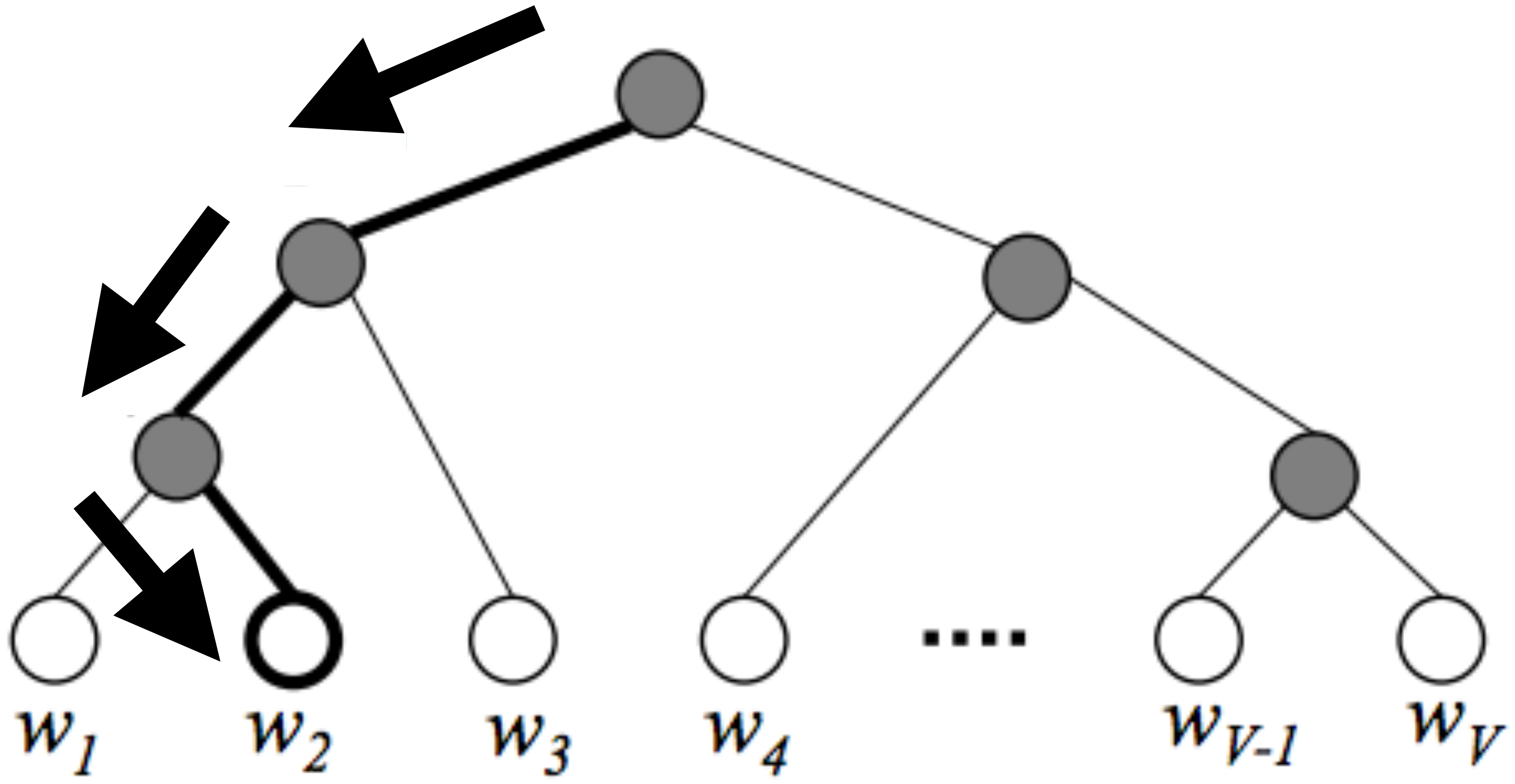
$$P(v_{out}|v_{in}) = \text{P(left at N1}|\ v_{in})\ \text{P(left at N2}|\ v_{in})\ \text{P(right at N3}|\ v_{in})$$

*Bengio 2005

hierchical softmax

Hierarchical softmax

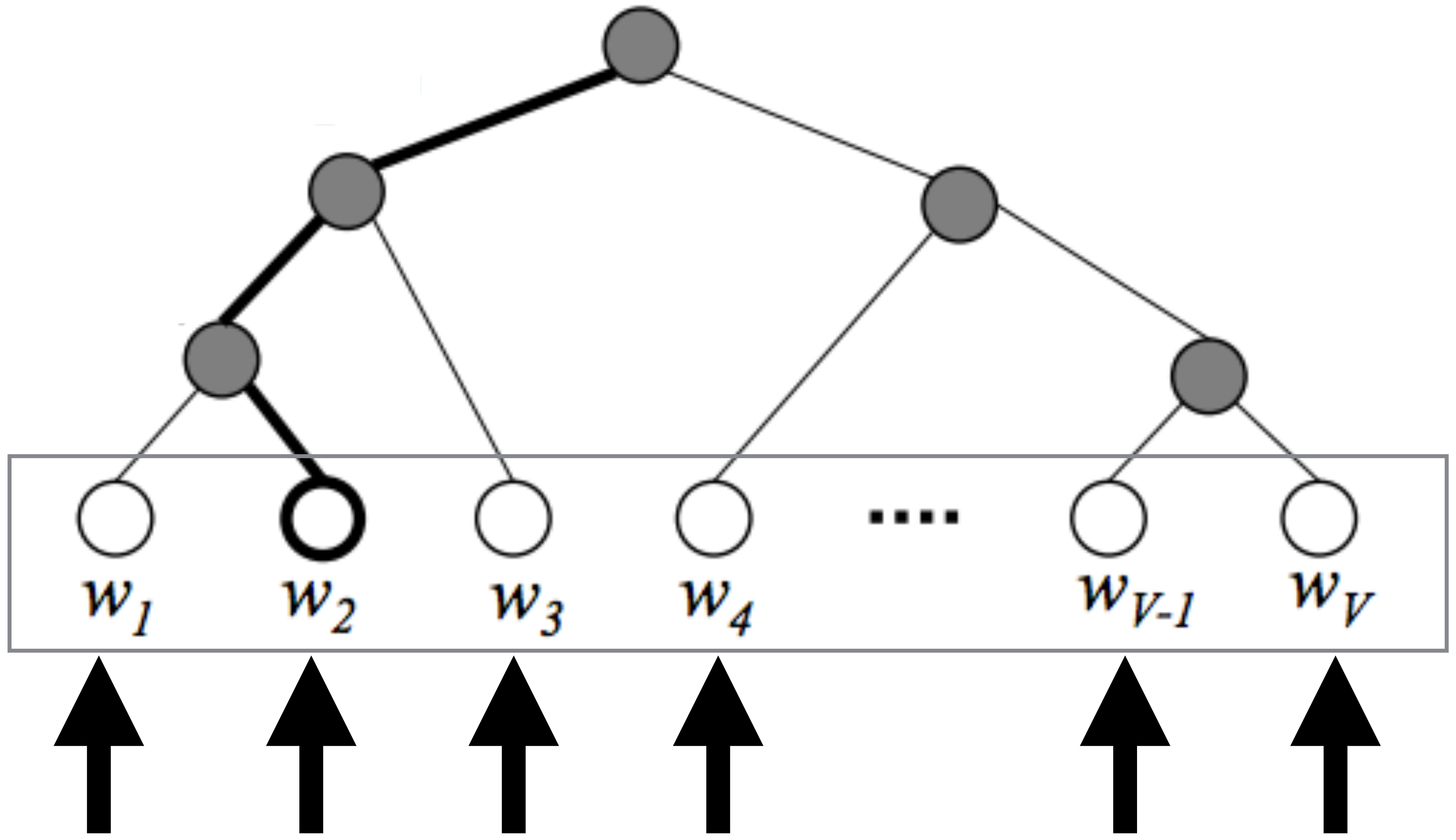
O(*logN*) steps



~10 comparisons

*Bengio 2005

Hierarchical softmax



~O($logN$) steps

vs

O(N) steps

~10 comparisons

~50k comparisons

$$\Sigma exp(v_{in} \cdot v_{out})$$

*Bengio 2005

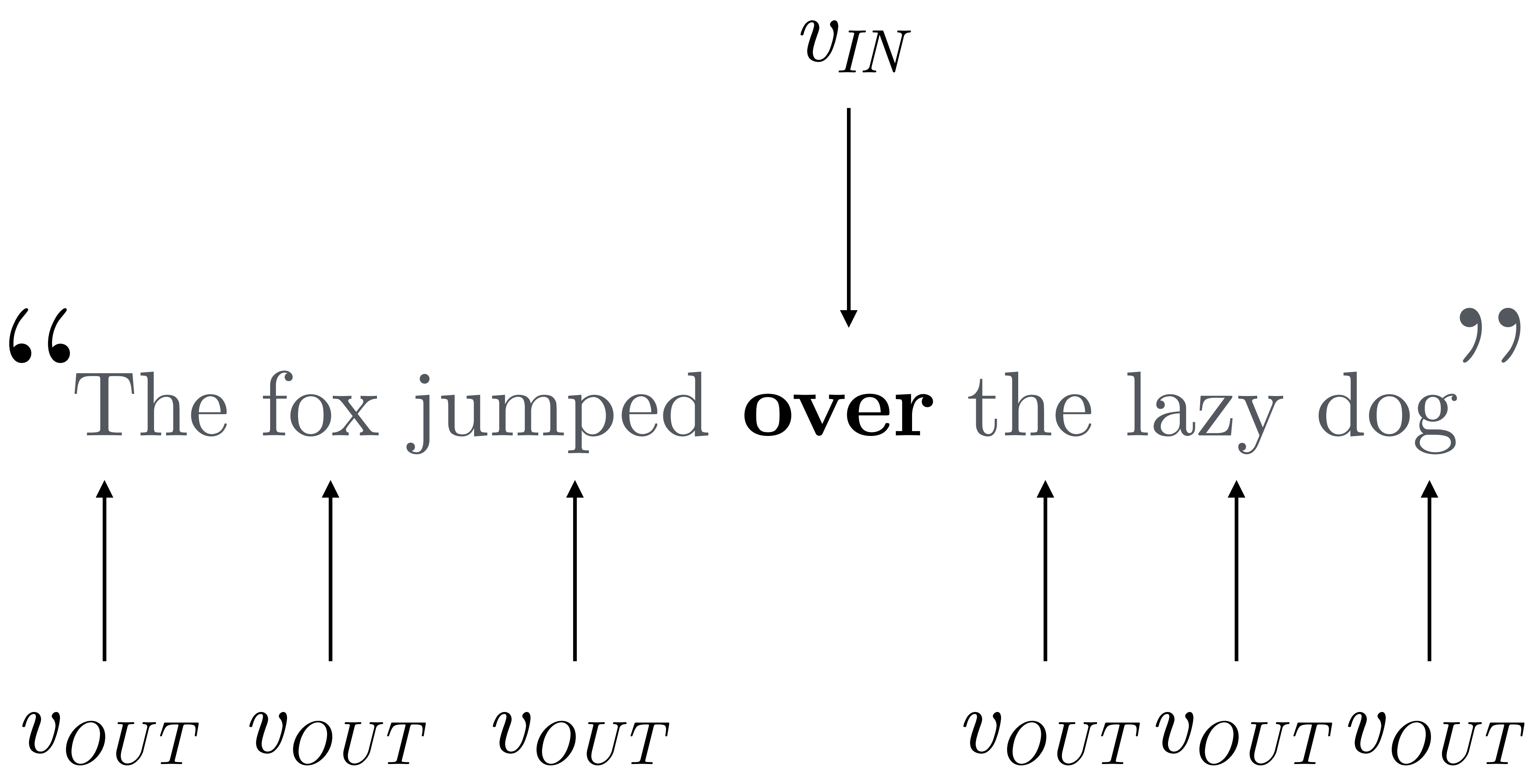
(͡° ͜ʖ ͡°)

Now performance is $O(N\ C\ log V)$!

Now we can scale to a 100 billion word corpus.

SkipGram

Guess the context
given the word



v_{IN}

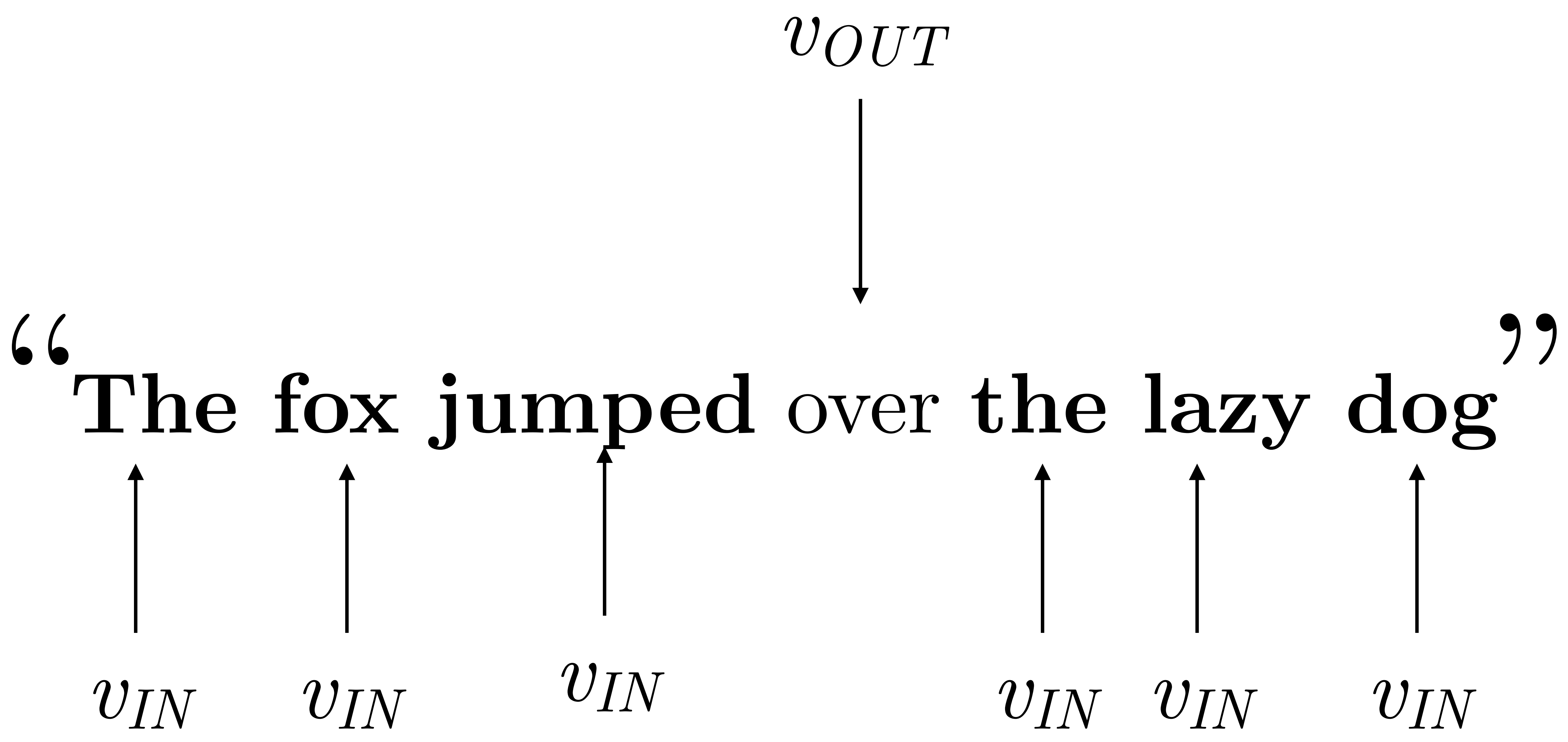
“The fox jumped **over** the lazy dog”

v_{OUT} v_{OUT} v_{OUT} v_{OUT} v_{OUT} v_{OUT}

Better at syntax.
(this is the one we went over)

CBOW

Guess the word
given the context

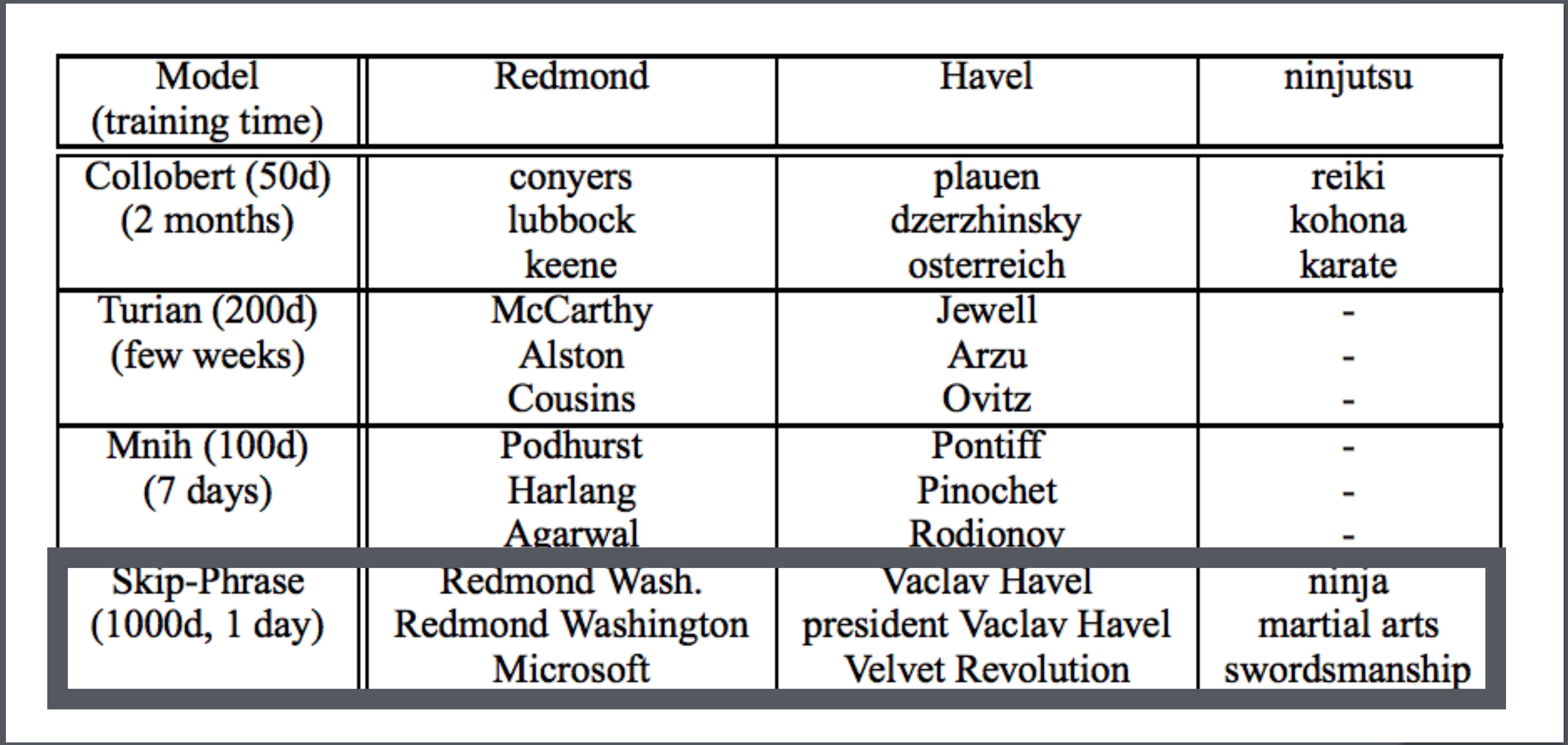


v_{OUT}

“**The fox jumped** over **the lazy dog**”

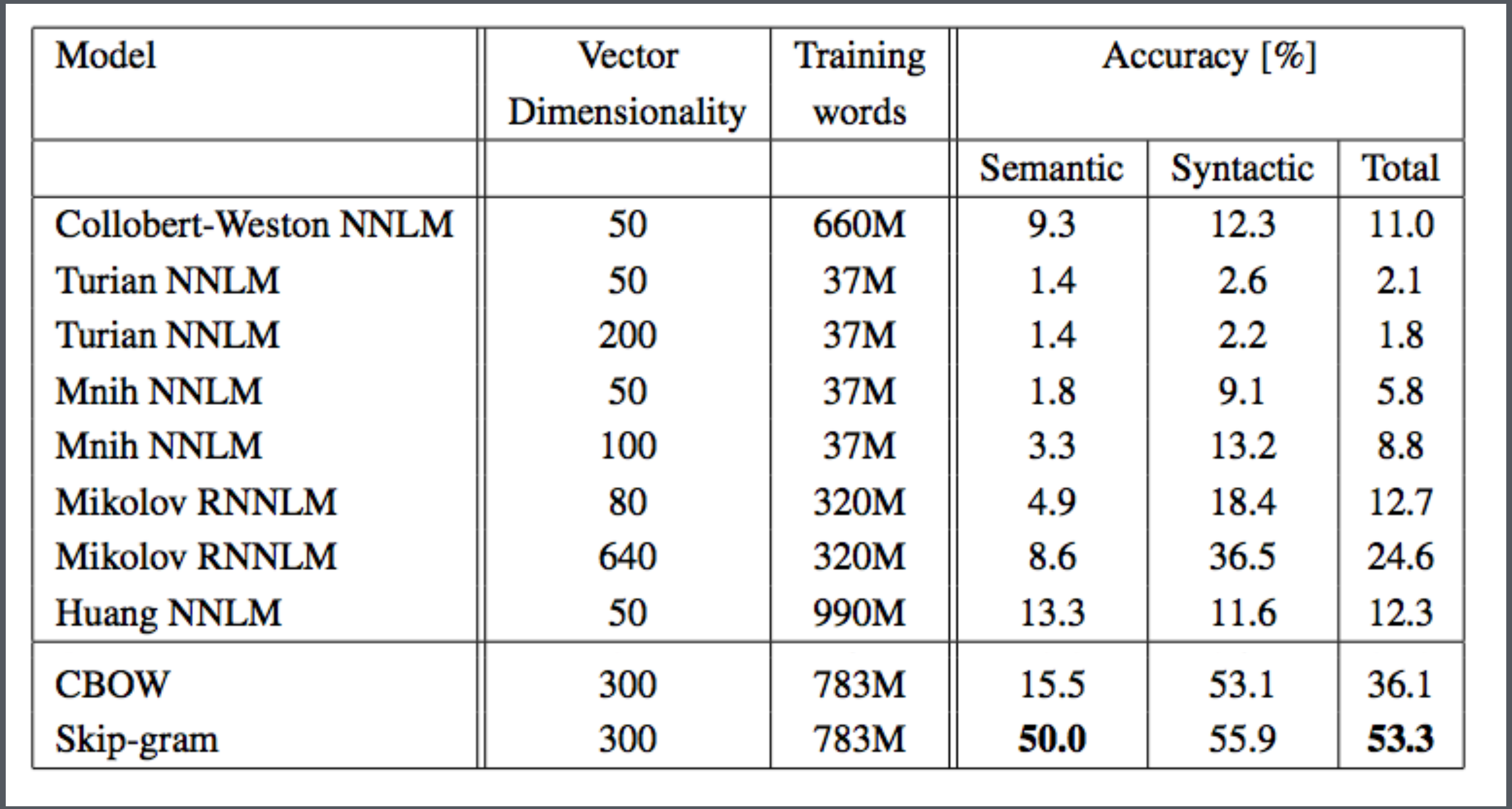
v_{IN} v_{IN} v_{IN} v_{IN} v_{IN} v_{IN}

~20x faster.
(this is the alternative.)



Model (training time)	Redmond	Havel	ninjutsu
Collobert (50d) (2 months)	conyers lubbock keene	plauen dzerzhinsky osterreich	reiki kohona karate
Turian (200d) (few weeks)	McCarthy Alston Cousins	Jewell Arzu Ovitz	- - -
Mnih (100d) (7 days)	Podhurst Harlang Agarwal	Pontiff Pinochet Rodionov	- - -
Skip-Phrase (1000d, 1 day)	Redmond Wash. Redmond Washington Microsoft	Vaclav Havel president Vaclav Havel Velvet Revolution	ninja martial arts swordsmanship

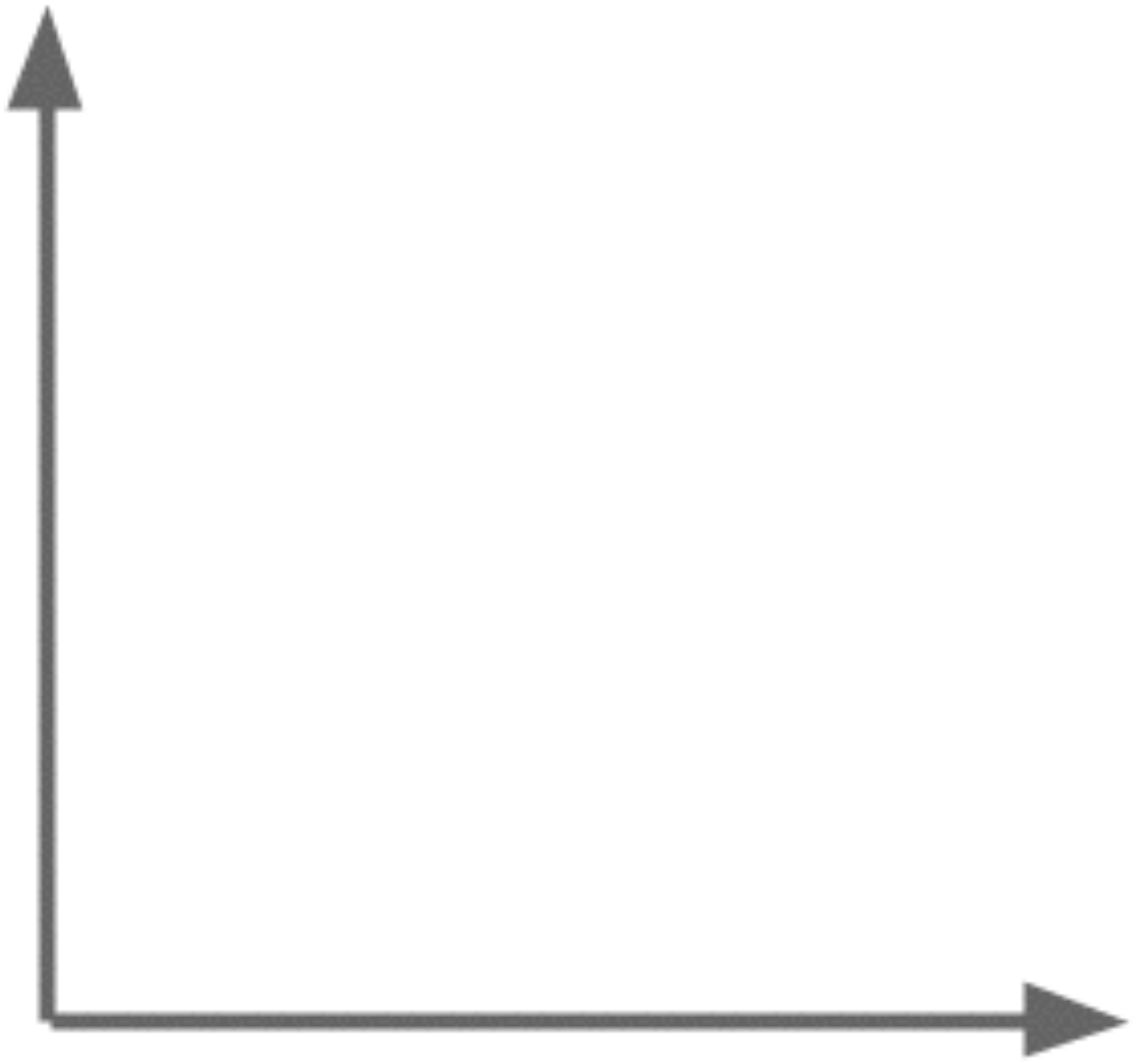
← word2vec



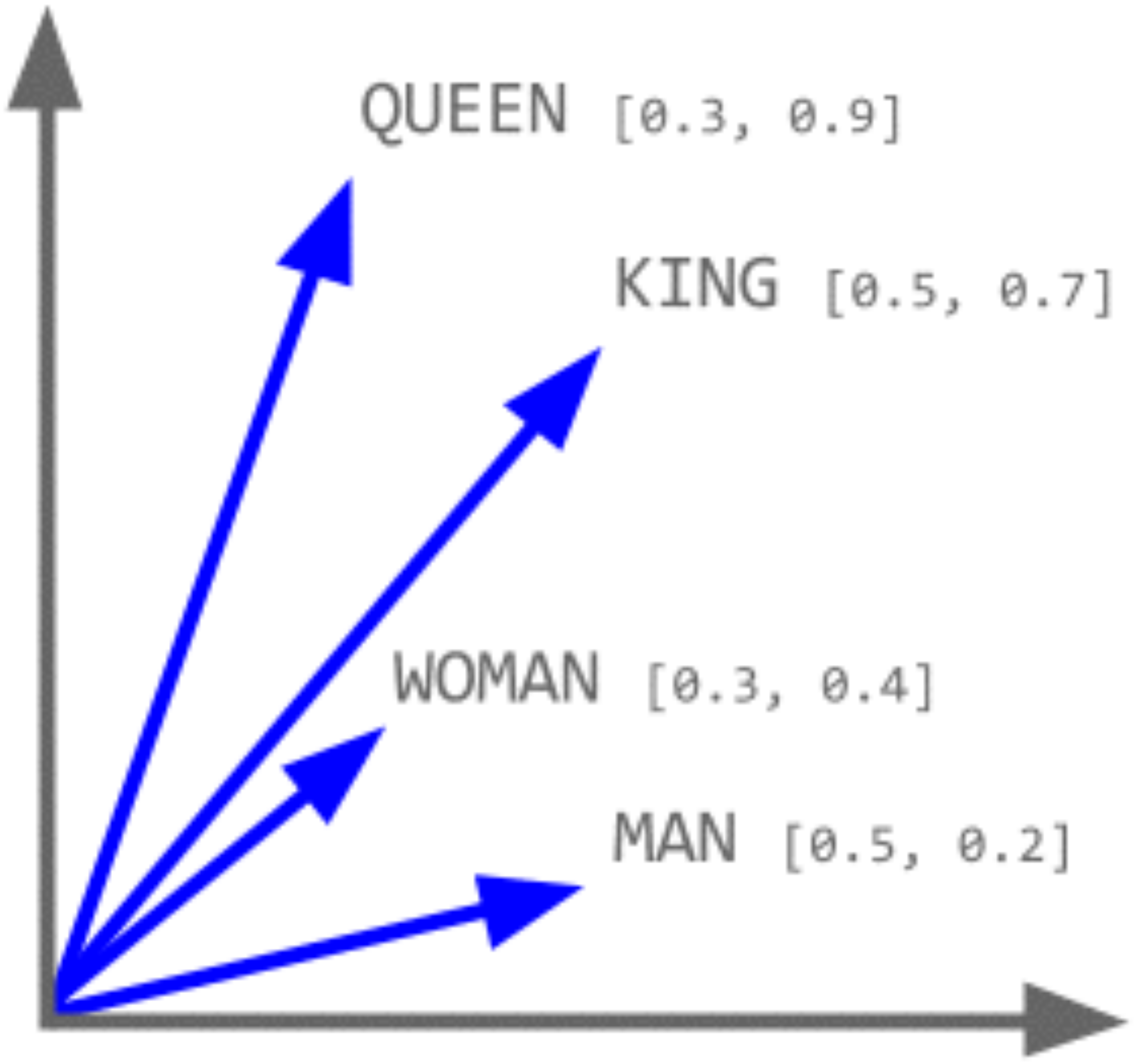
Model	Vector Dimensionality	Training words	Accuracy [%]		
			Semantic	Syntactic	Total
Collobert-Weston NNLM	50	660M	9.3	12.3	11.0
Turian NNLM	50	37M	1.4	2.6	2.1
Turian NNLM	200	37M	1.4	2.2	1.8
Mnih NNLM	50	37M	1.8	9.1	5.8
Mnih NNLM	100	37M	3.3	13.2	8.8
Mikolov RNNLM	80	320M	4.9	18.4	12.7
Mikolov RNNLM	640	320M	8.6	36.5	24.6
Huang NNLM	50	990M	13.3	11.6	12.3
CBOW	300	783M	15.5	53.1	36.1
Skip-gram	300	783M	**50.0**	55.9	**53.3**

← word2vec

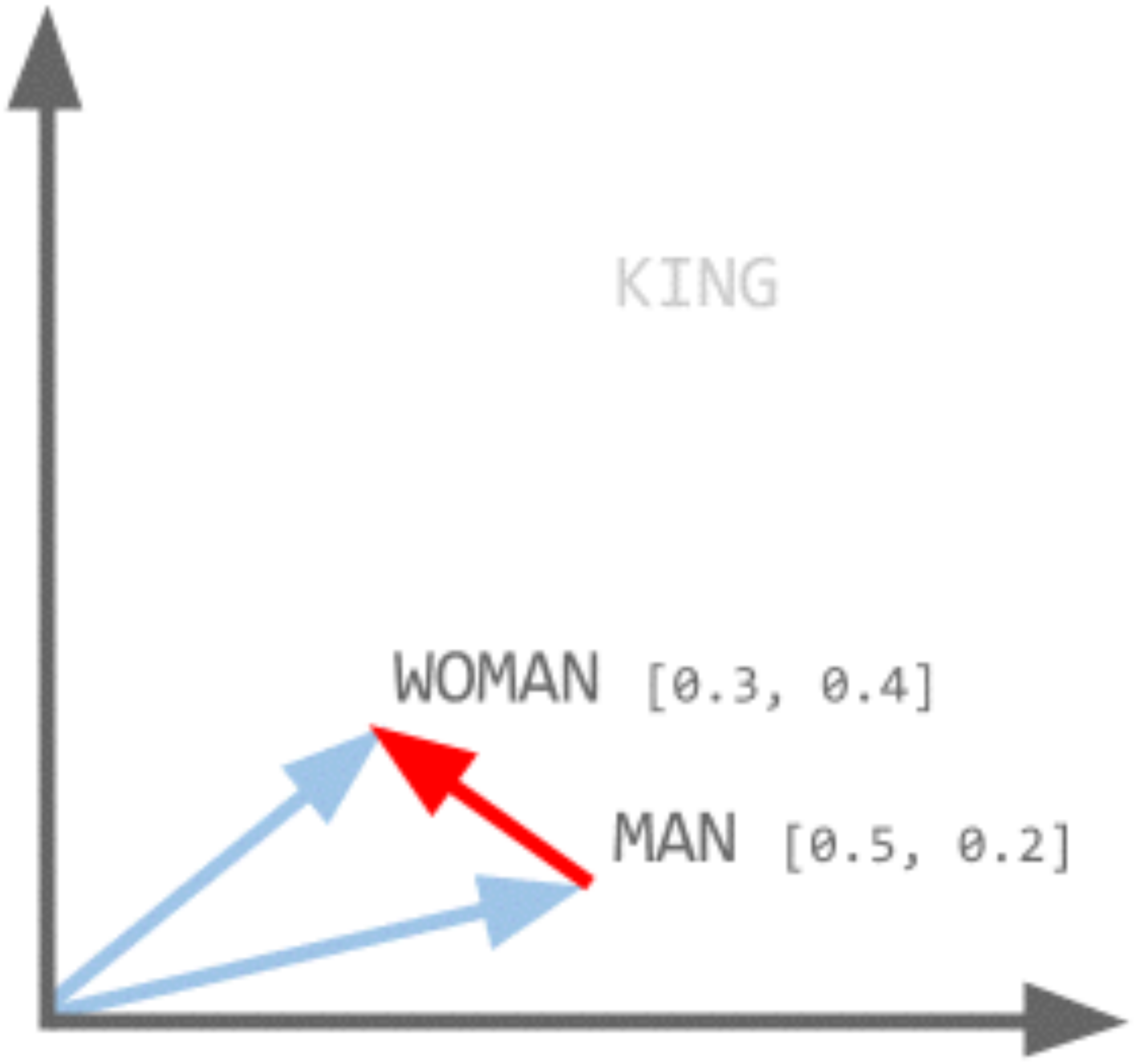
What is `king` + `man` - `woman`?



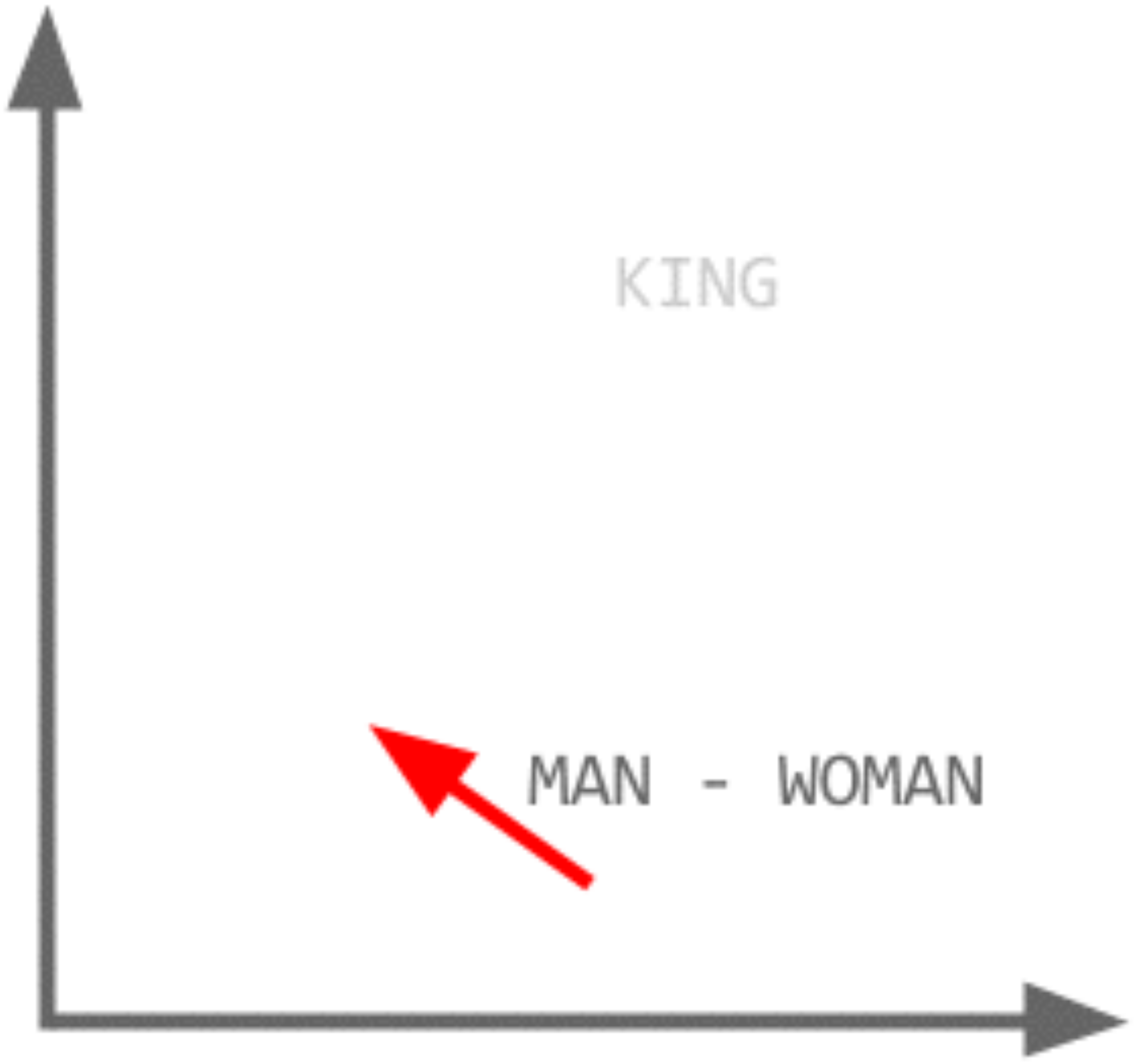
Load up the word vectors



Start with `man - woman`



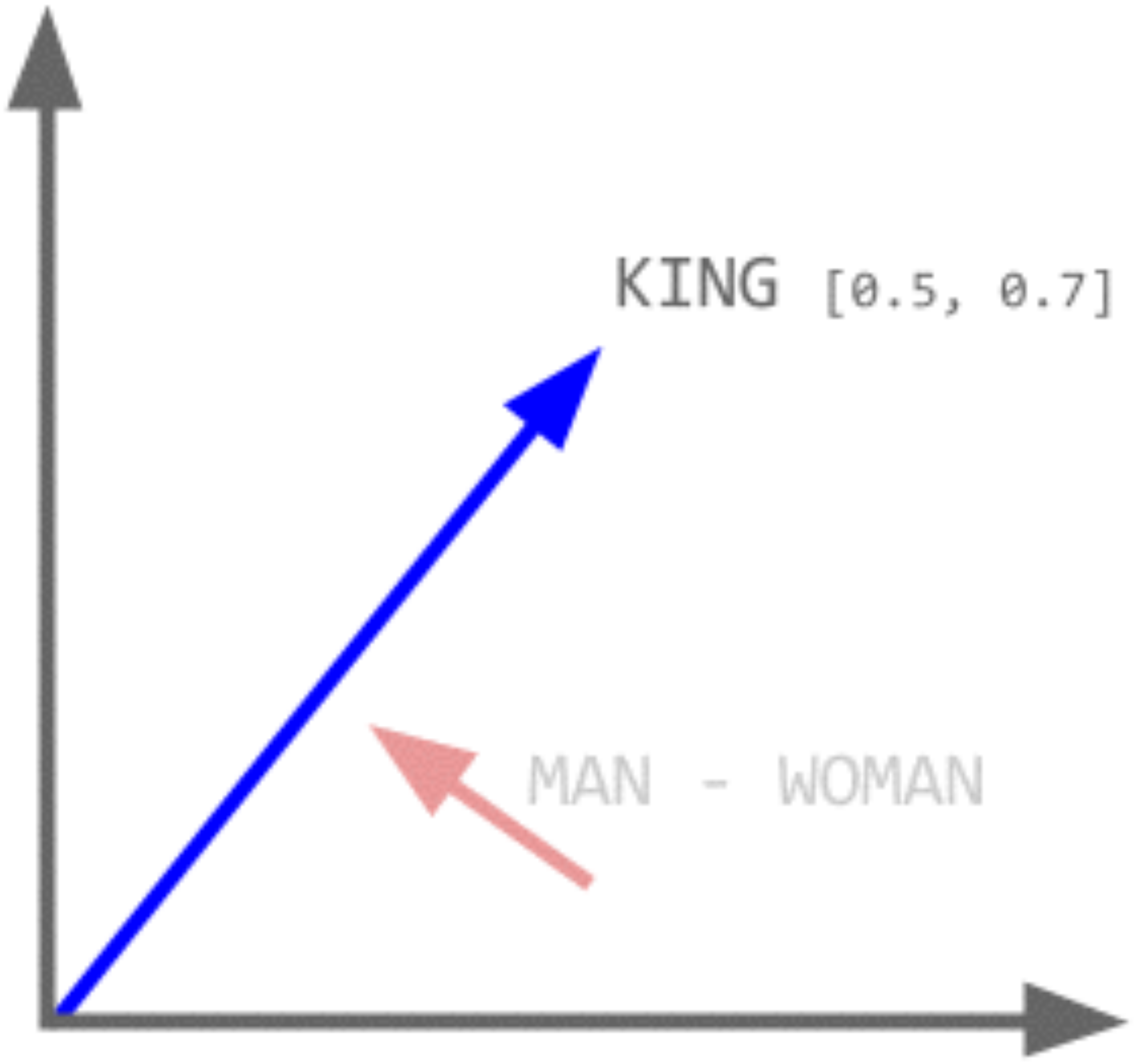
Start with `man - woman`



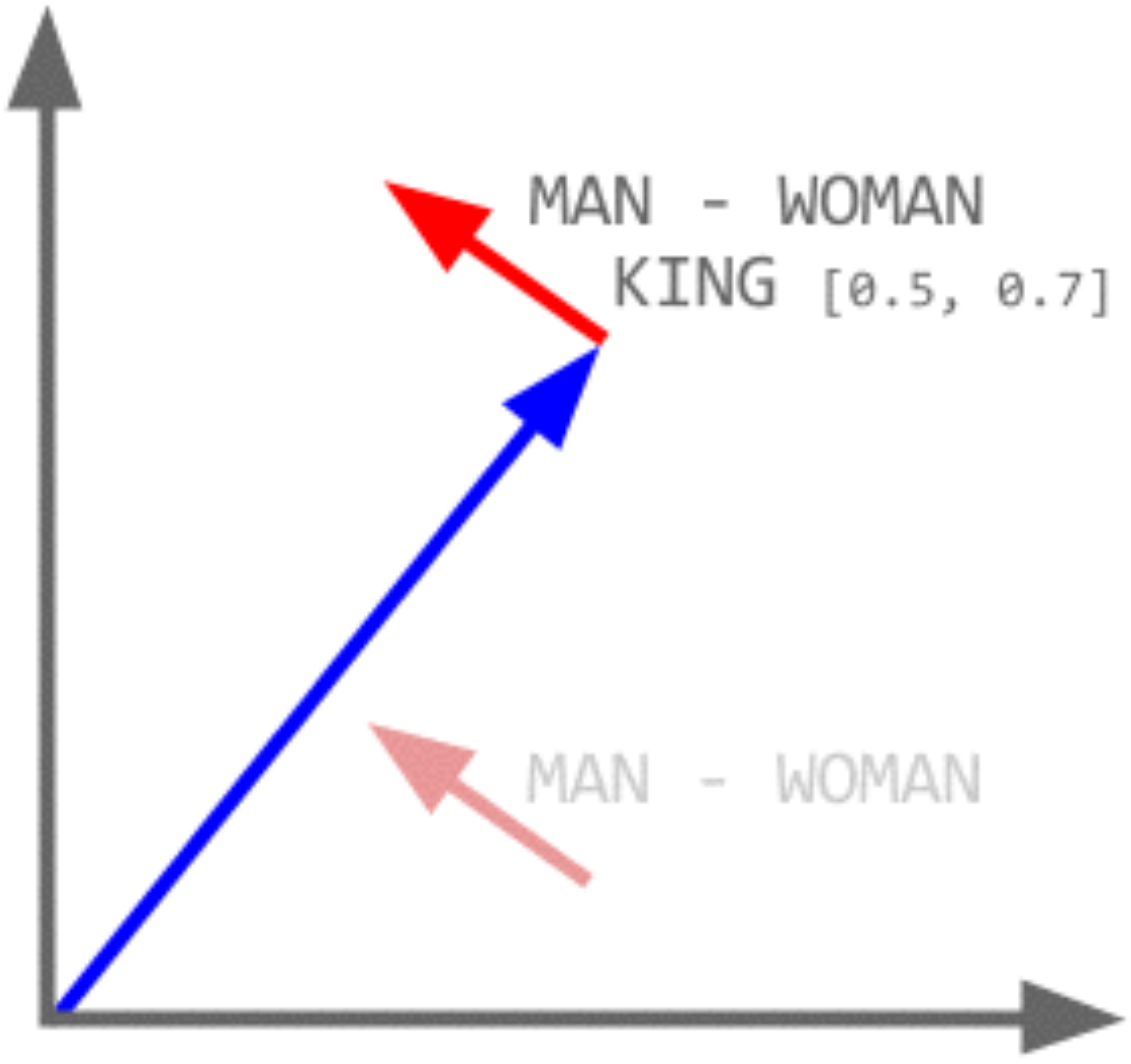
KING

MAN - WOMAN

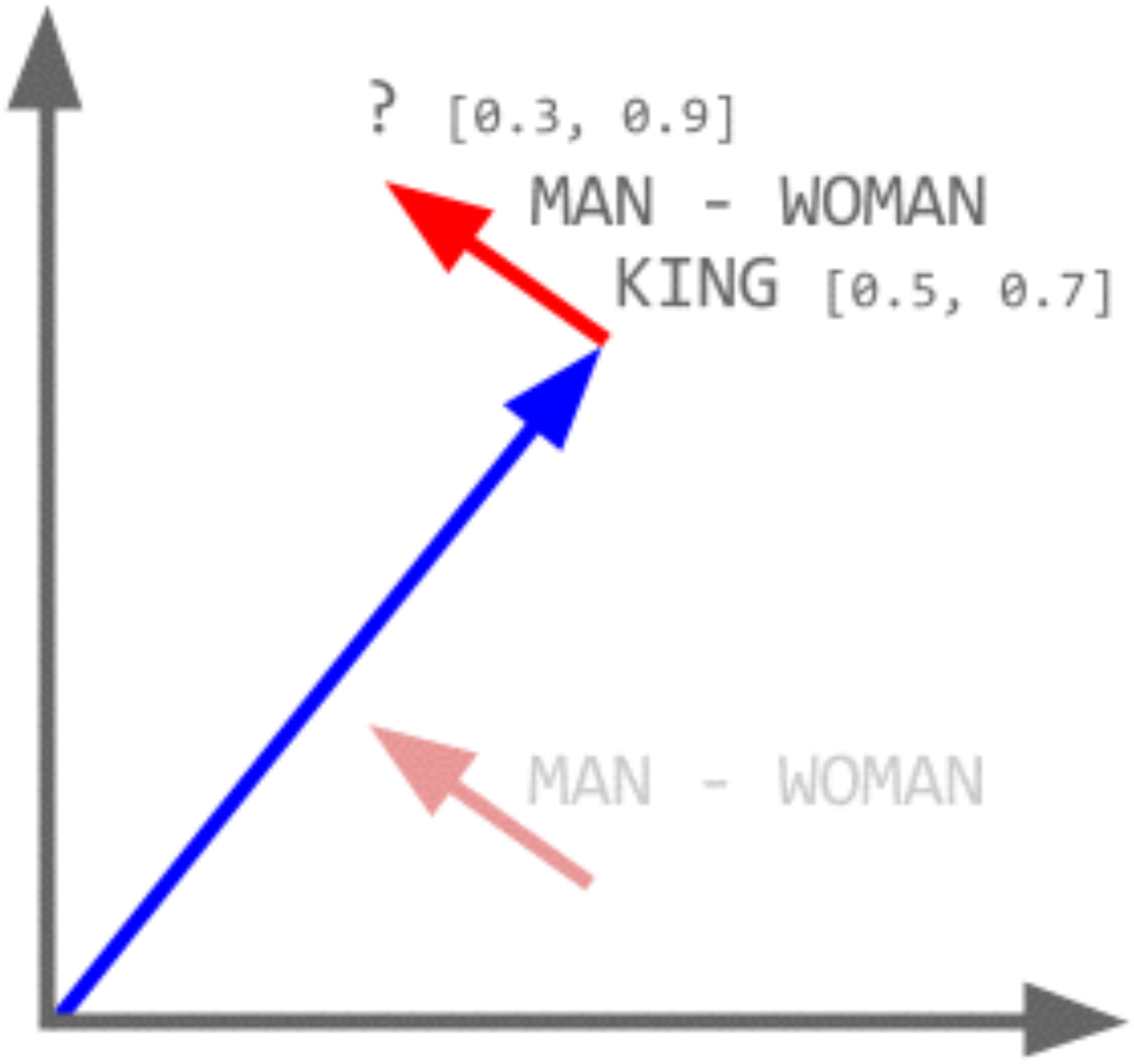
Then take `king`



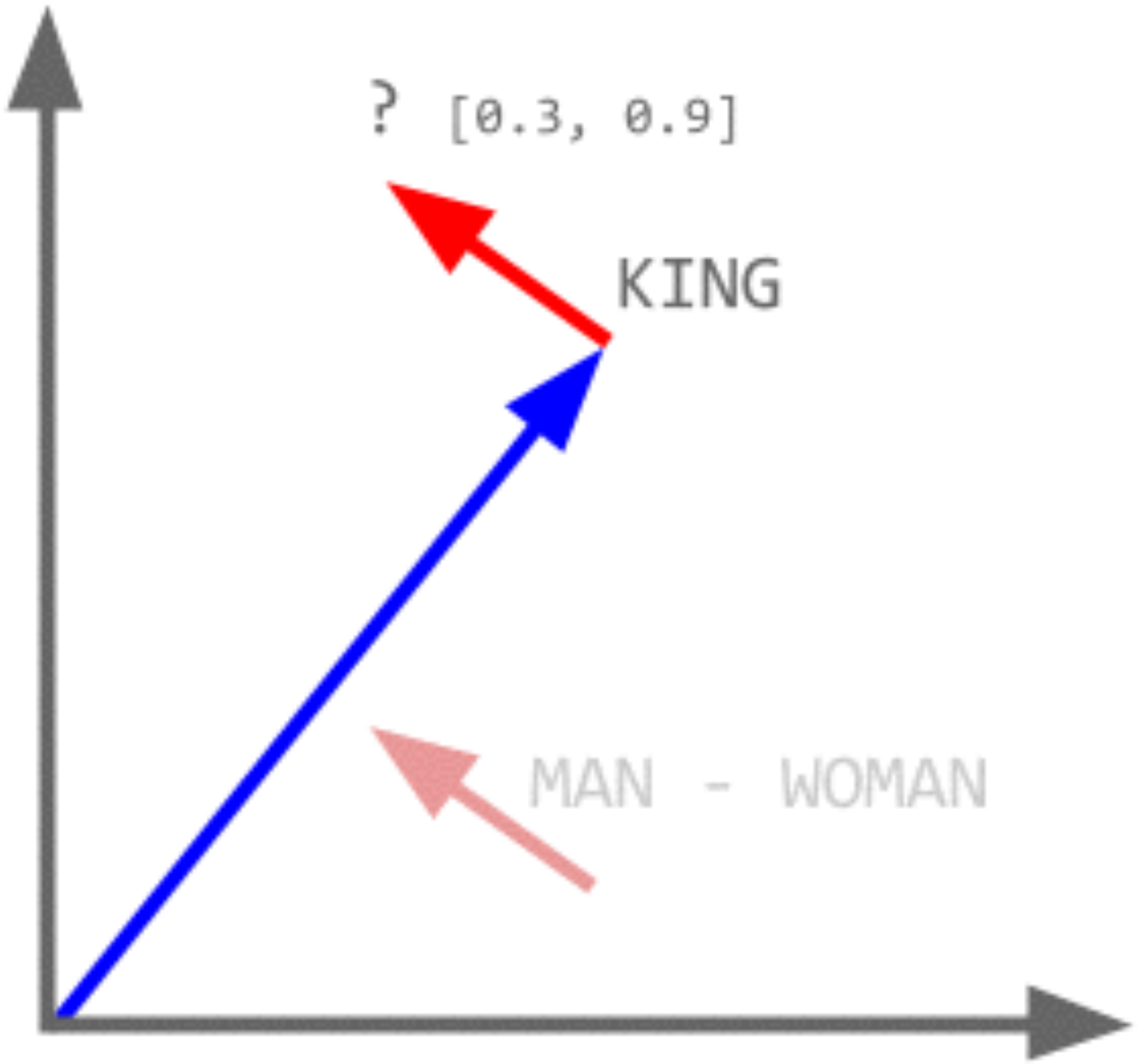
And add `man - woman`



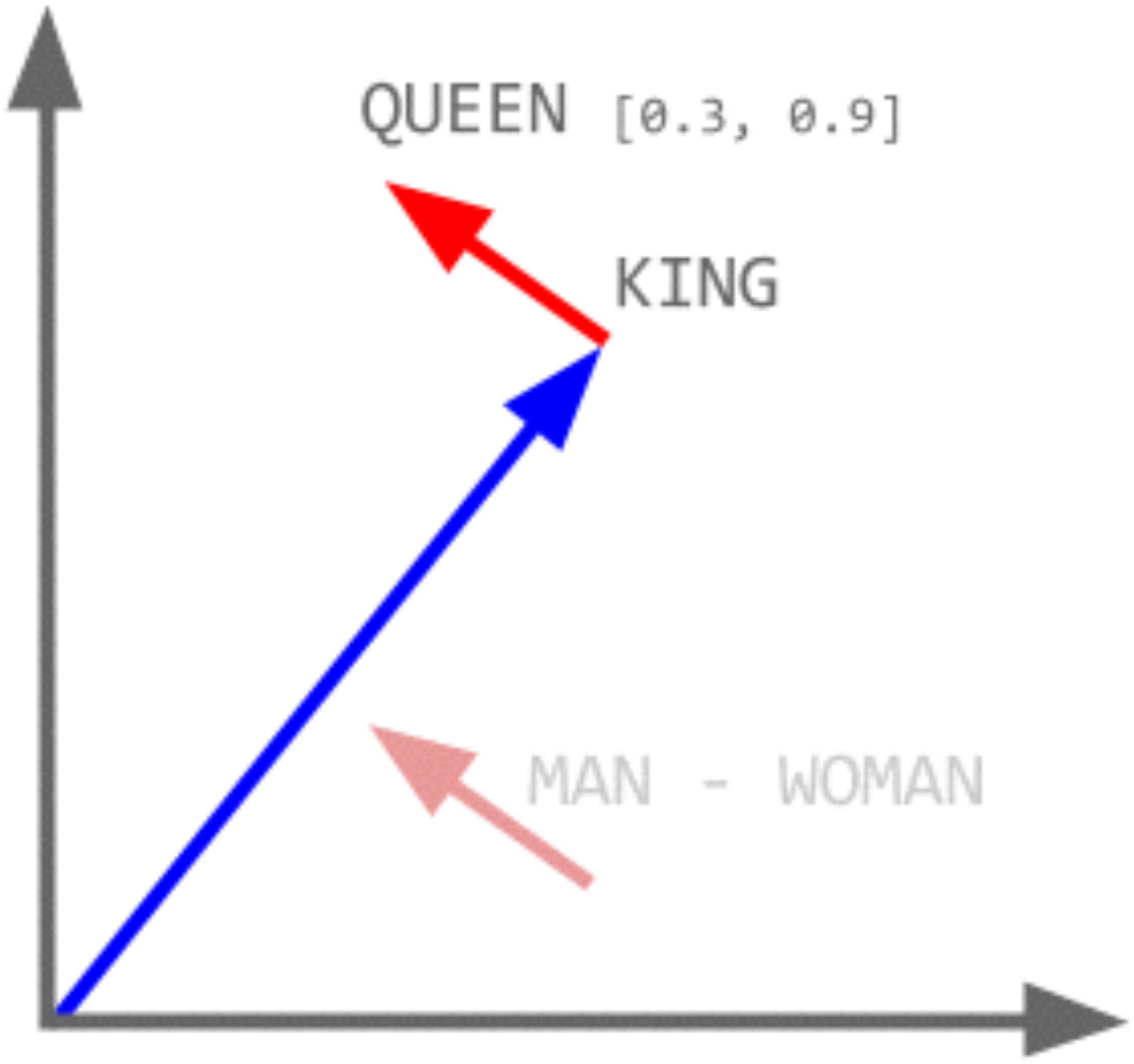
And add `man - woman`



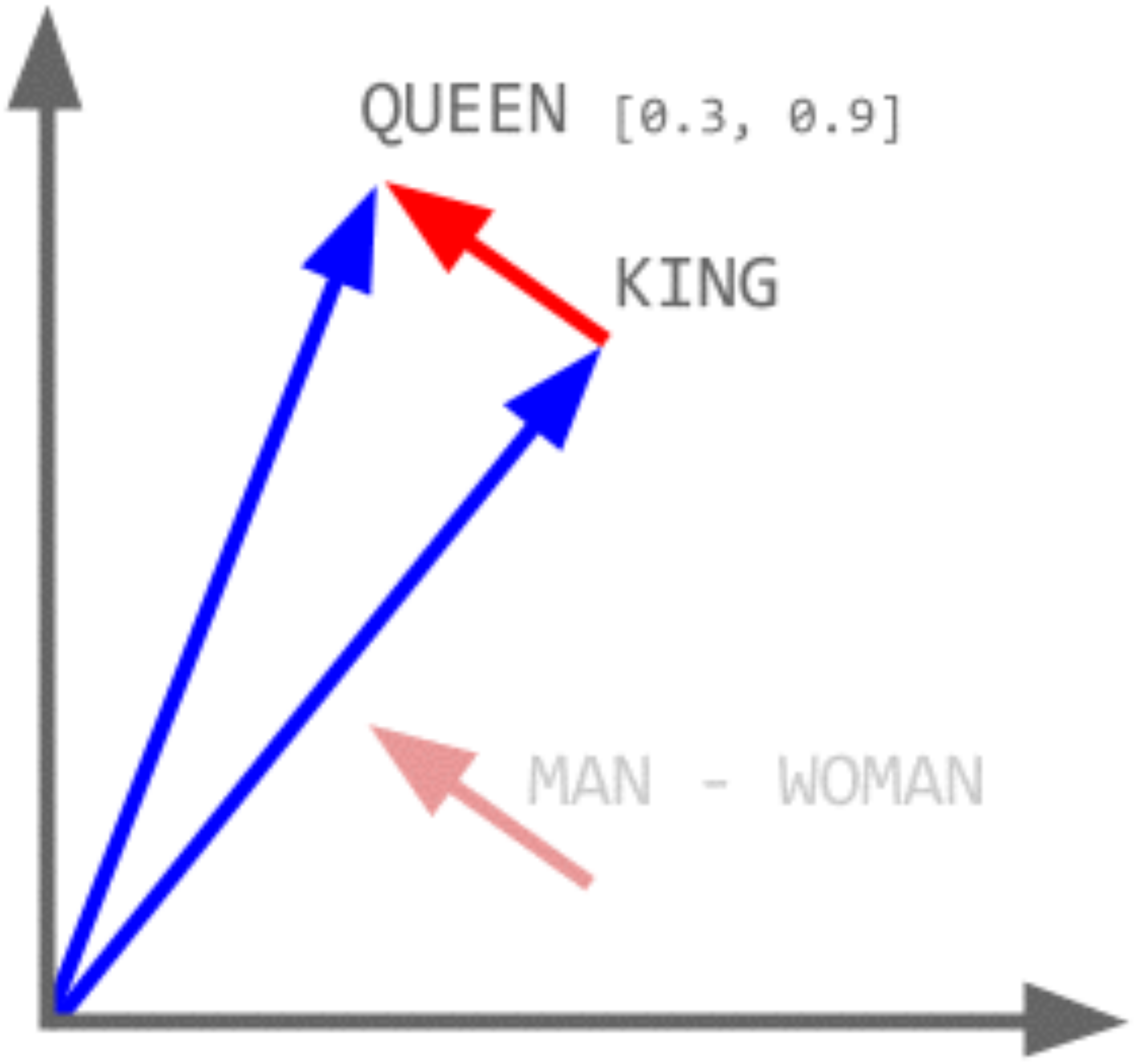
Find nearest word to result



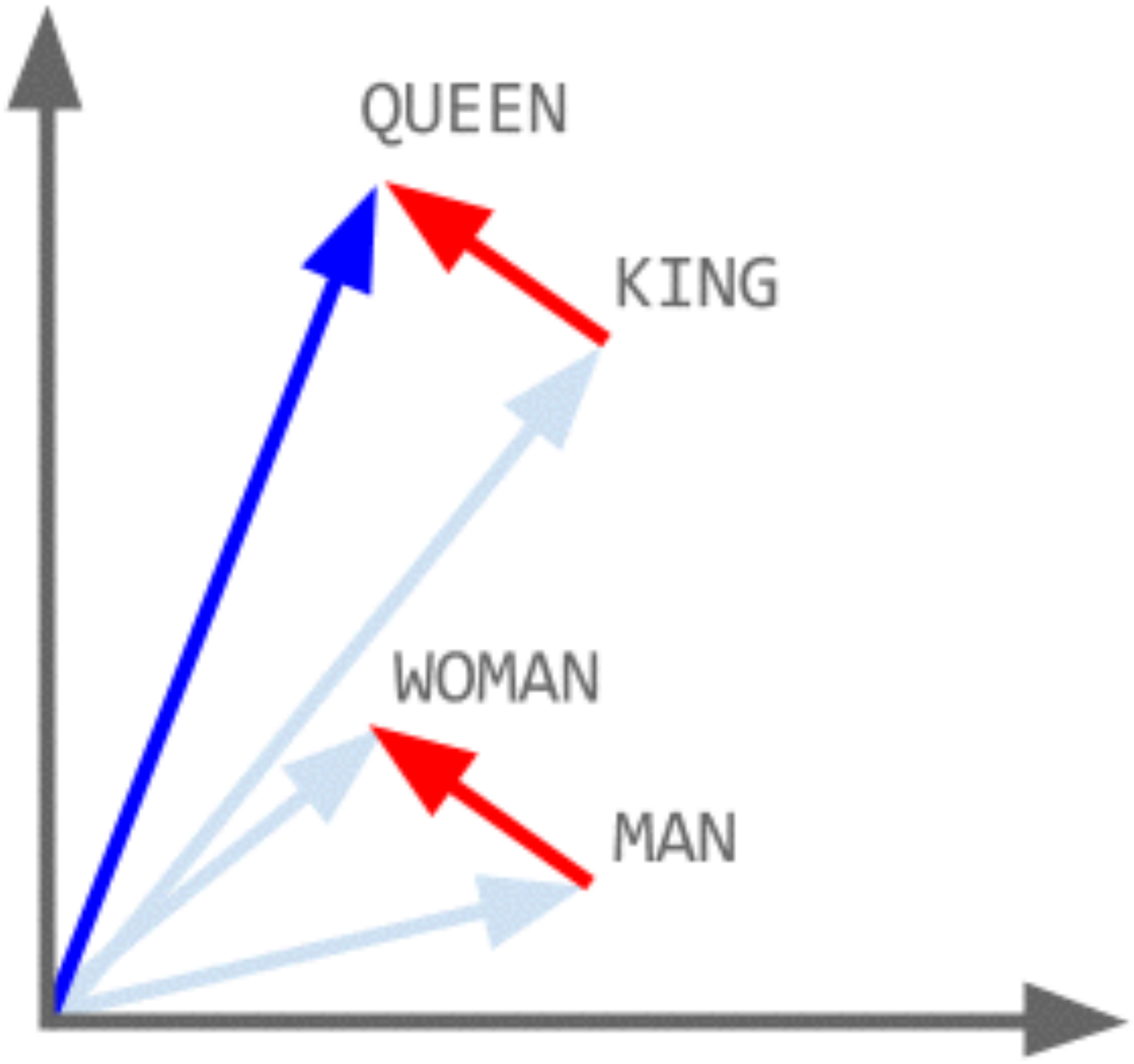
`queen` is closest to resulting vector



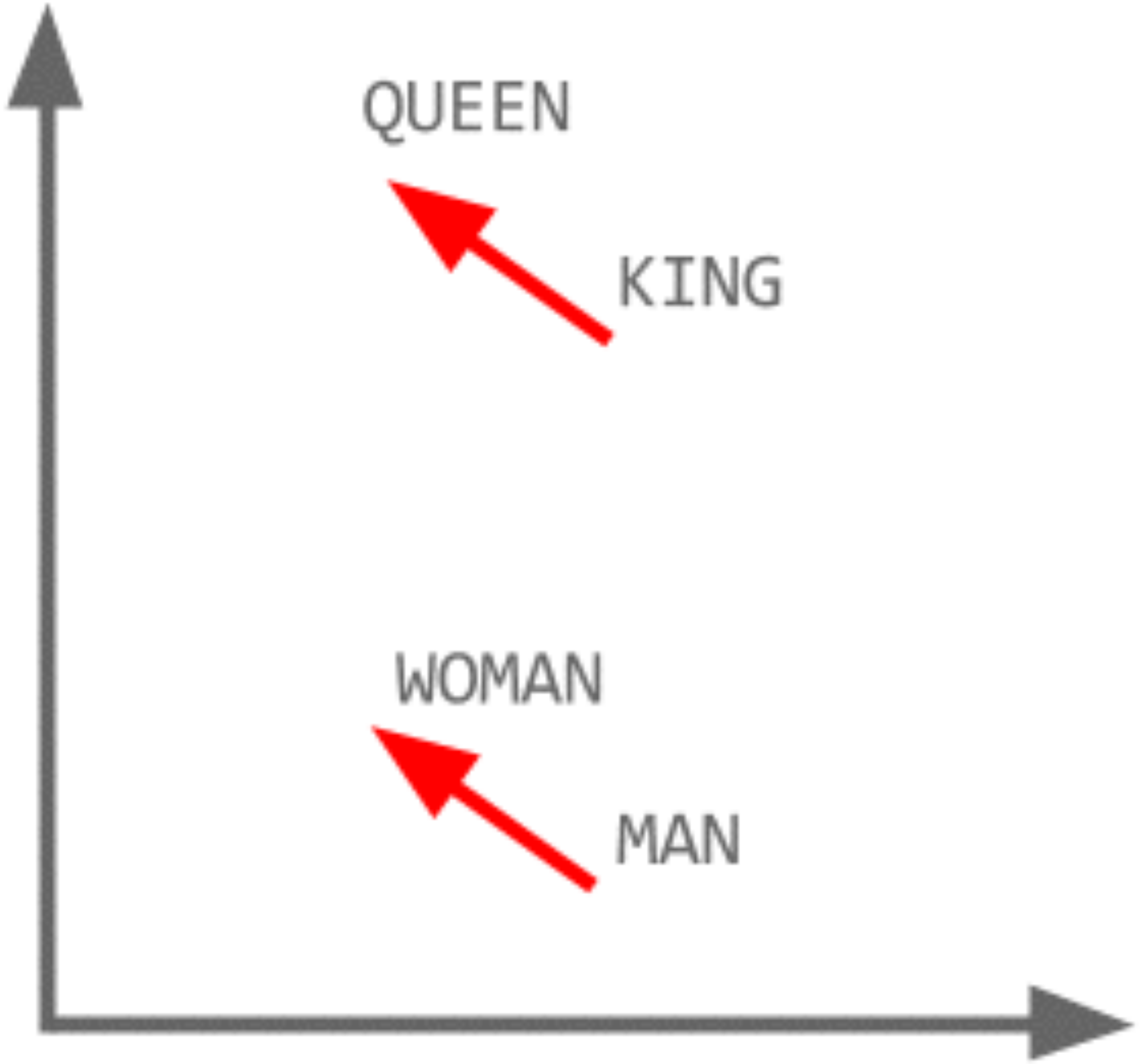
`queen` is closest to resulting vector



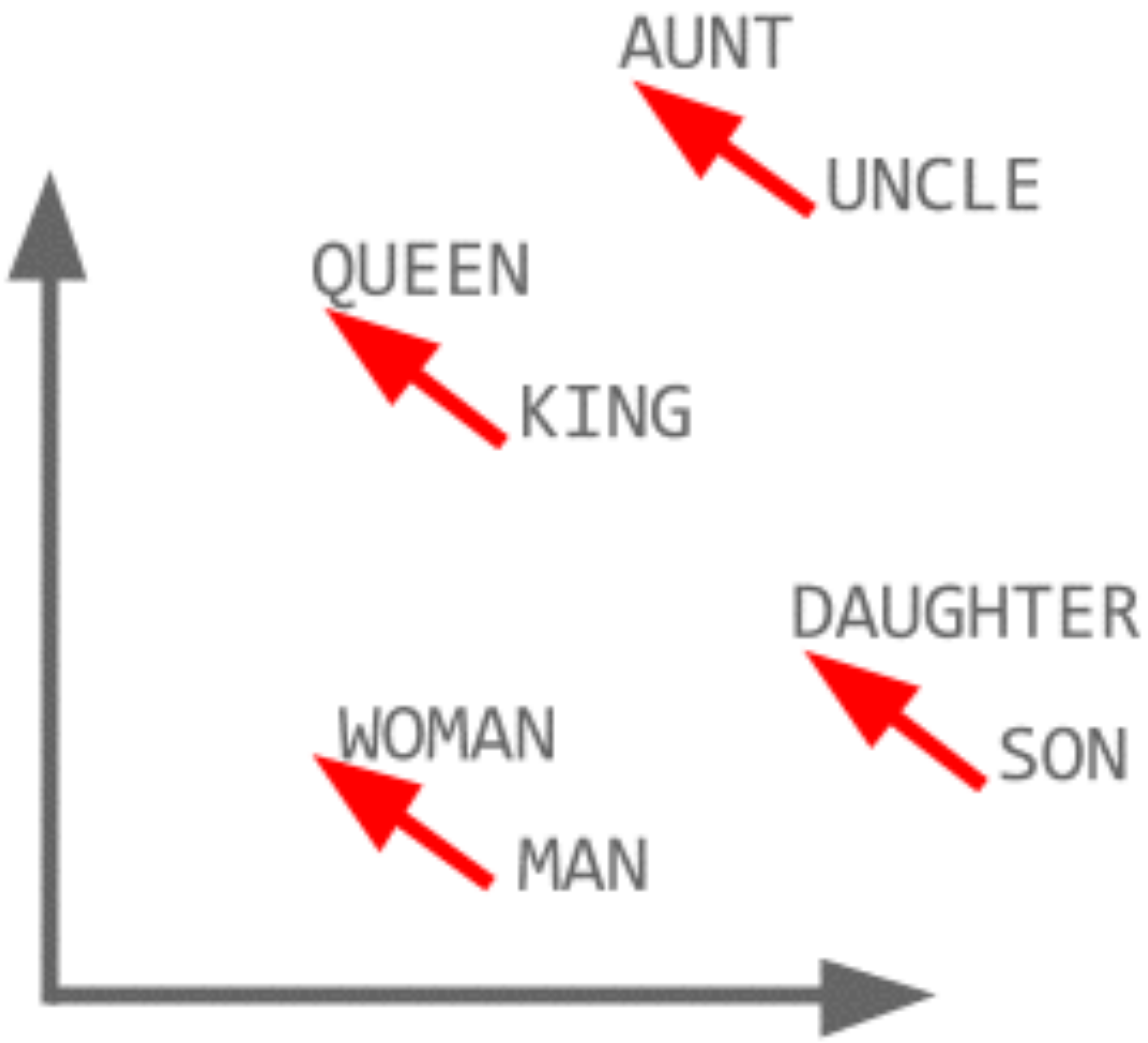
So king + man - woman = queen!



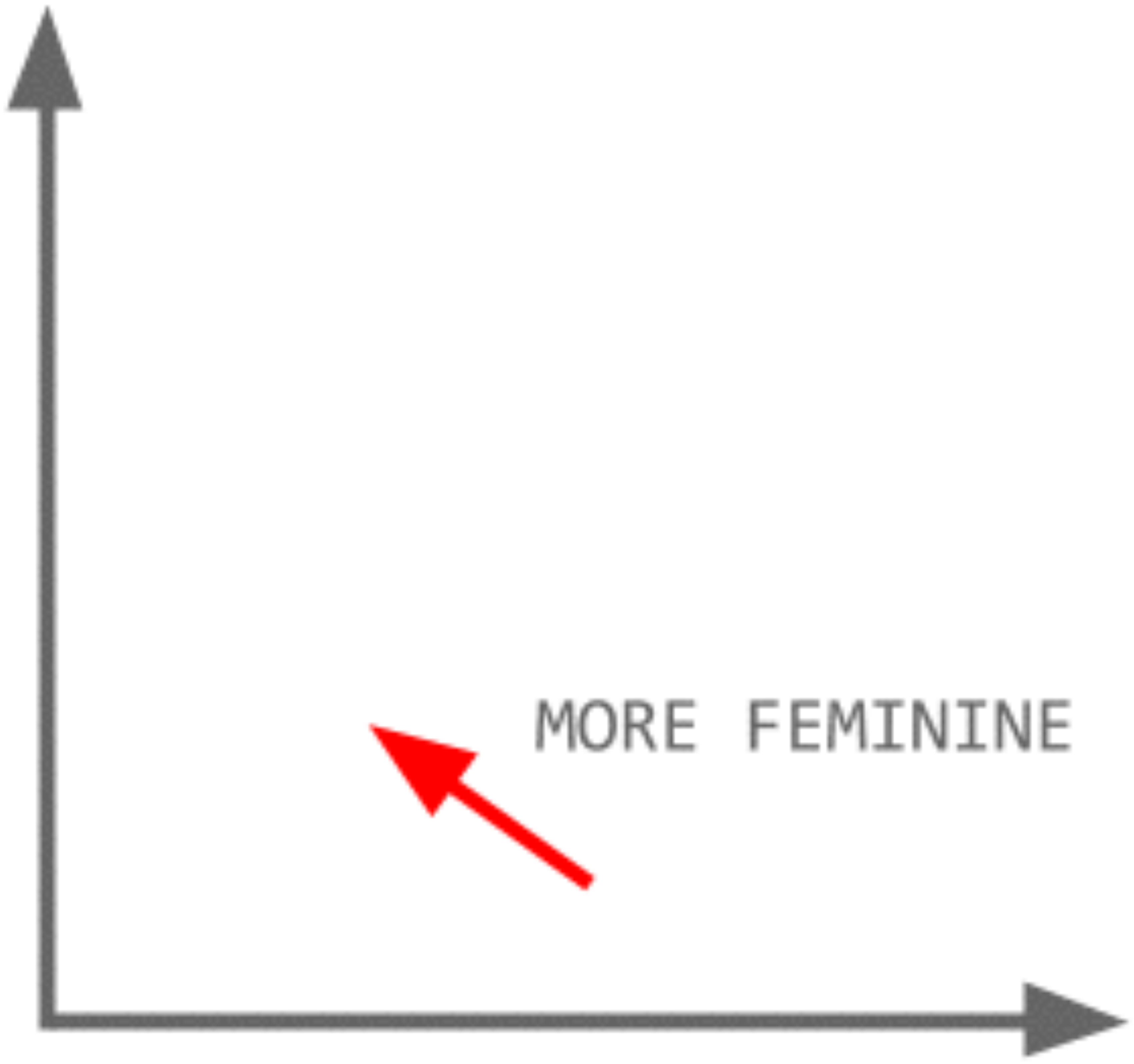
The red direction encodes gender



Which is consistent across all words

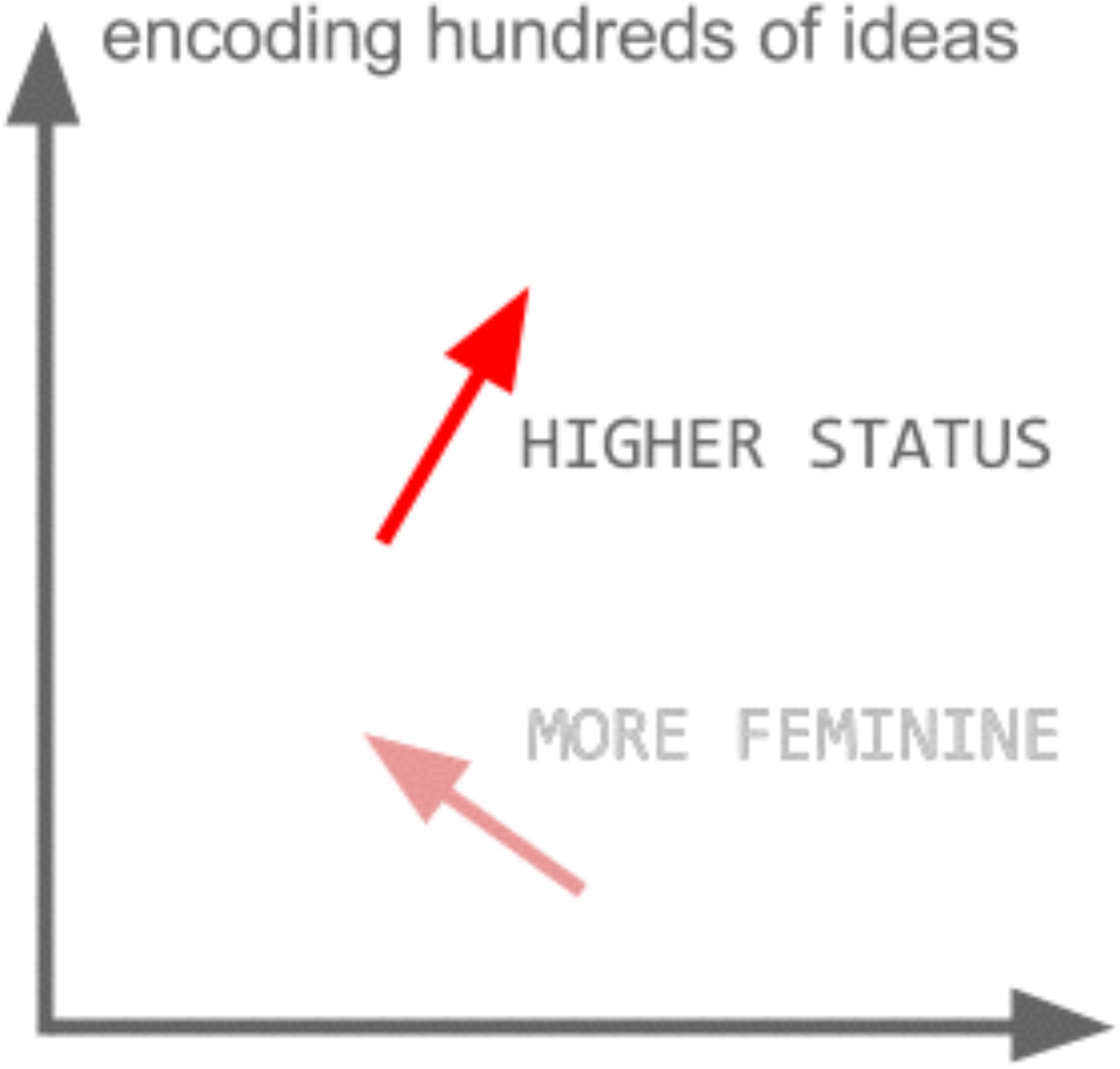


This direction always means gender



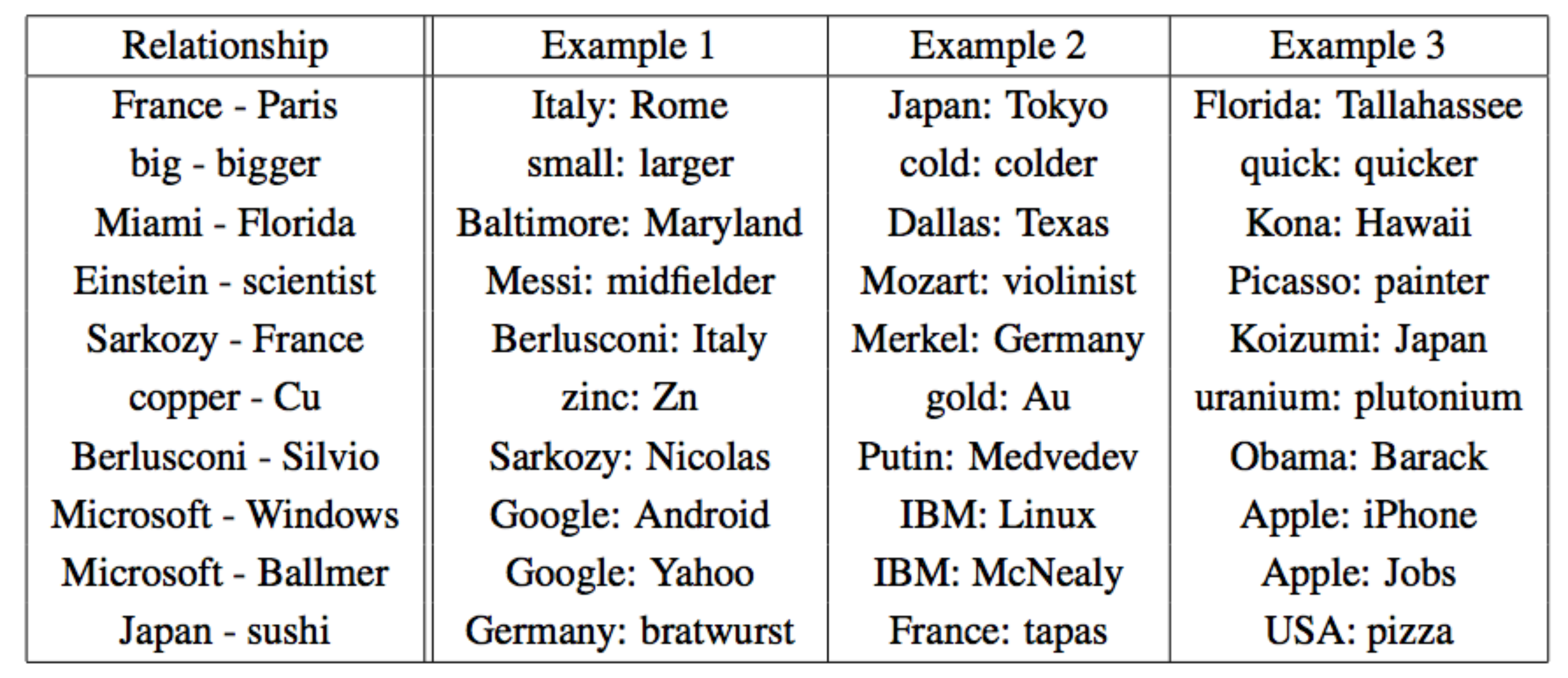
MORE FEMININE

We have hundreds of directions encoding hundreds of ideas

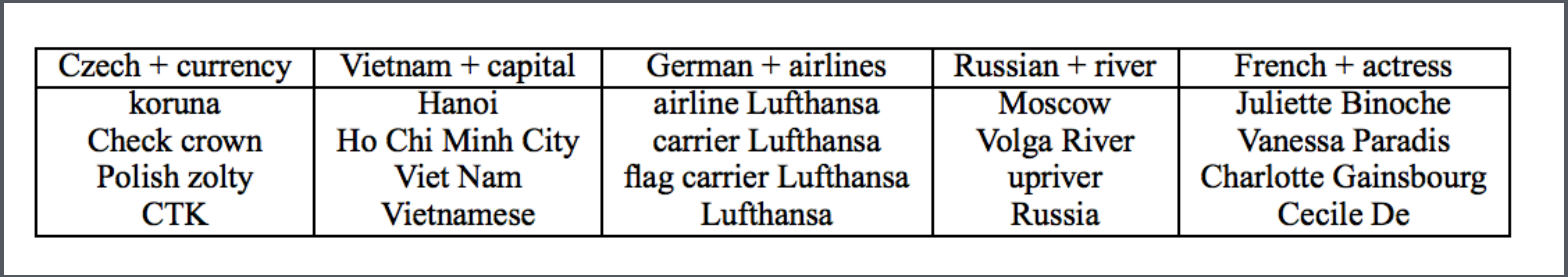


HIGHER STATUS

MORE FEMININE



Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza



Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

ITEM_3469 + ‘Pregnant’

+ ‘Pregnant’


wedding
holidays
month
destinations
vacation
season
bachelorette
shower
cruise
vacationed
vegas
party
luau
birthday
wedding
events
weddings
dinners
holidaying
destination
vacationers
honeymoons
relaxing
outing
getaways
resort
resorts
getaway
brunch
lunch
holidays
thanksgiving
valentines
concert
vacations
easter
outings
trips
dinner
holiday
reunion
attending
waikiki
february
siesta
mustique
oceanfront
nov
oct
dec
headed
evenings
friends
attend
honeymoon
may
feb
june
april
night
evening
fancy
town
engagement
summertime
weekends
planned
traveling
journey
travels
cancun
cozumel
disneyworld
march
th
weekend
days
kids
visiting
springtime
day
trip
beach
st
home
heading
nyc
travel
mexico
march
later
arrived
island
hawaii
business
summery
splurge
planning
moved
sept
arrives
florida
florida
late
arrive
march
california
arriving
approaching
november
october
married
rome
thailand
disney
disneyland
stuff
august
december
options
working
starting
break
family
europe
hawaii
italy
vegas
jamaica
perhaps
april
february
september
july
trend
clothes
especially
beach
resort
spain
greece
bahamas
maui
aruba
congr
surprise
after
before
babymoon
mexico
wonderful
now
rica
caribbean
beginning
june
end
honeymoon
australia
week
coming
birthday
cruise
in
going
i
month
vacay
vaca
destination
belize
to
upcoming
trip
vacations
cancun
here
months
weeks
vacation
focus
focused
early
summer
trip
season
live
spent
spend
summer
fall
winter
spring
weather
warm
getaway
hot
warmer

word2vec

Learns word vectors
Learn doc vectors

Text feature generation is great for ML models

LDA

Learns a topic distribution for every document

Each of the k topics is a great tag

and now for something **completely crazy**

All of the following ideas will change what ‘words’ and ‘context’ represent.

paragraph vector

What about summarizing documents?

On the day he took office, President Obama reached out to America’s enemies, offering in his first inaugural address to **extend** a hand if you are willing to unclench your fist. More than six years later, he has arrived at a moment of truth in testing that

IN

On the day he took office, President Obama reached out to America's enemies,
offering in his first inaugural address to **extend** a hand if you are willing to unclench
your fist. More than six years later, he has arrived at a moment of truth in testing that
The framework nuclear agreement he reached with Iran on Thursday did not provide
the definitive answer to whether Mr. Obama's audacious gamble will pay off. The fist
Iran has shaken at the so-called Great Satan since 1979 has not completely relaxed.

OUT OUT

Normal skipgram extends C words before, and C words after.

paragraph vector

IN

doc_1347

OUT

OUT

On the day he took office, President Obama reached out to America's enemies, offering in his first inaugural address to extend a hand if you are willing to unclench your fist. More than six years later, he has arrived at a moment of truth in testing that

The framework nuclear agreement he reached with Iran on Thursday did not provide the definitive answer to whether Mr. Obama's audacious gamble will pay off. The fist Iran has shaken at the so-called Great Satan since 1979 has not completely relaxed.

OUT

OUT

A document vector simply extends the context to the whole document.

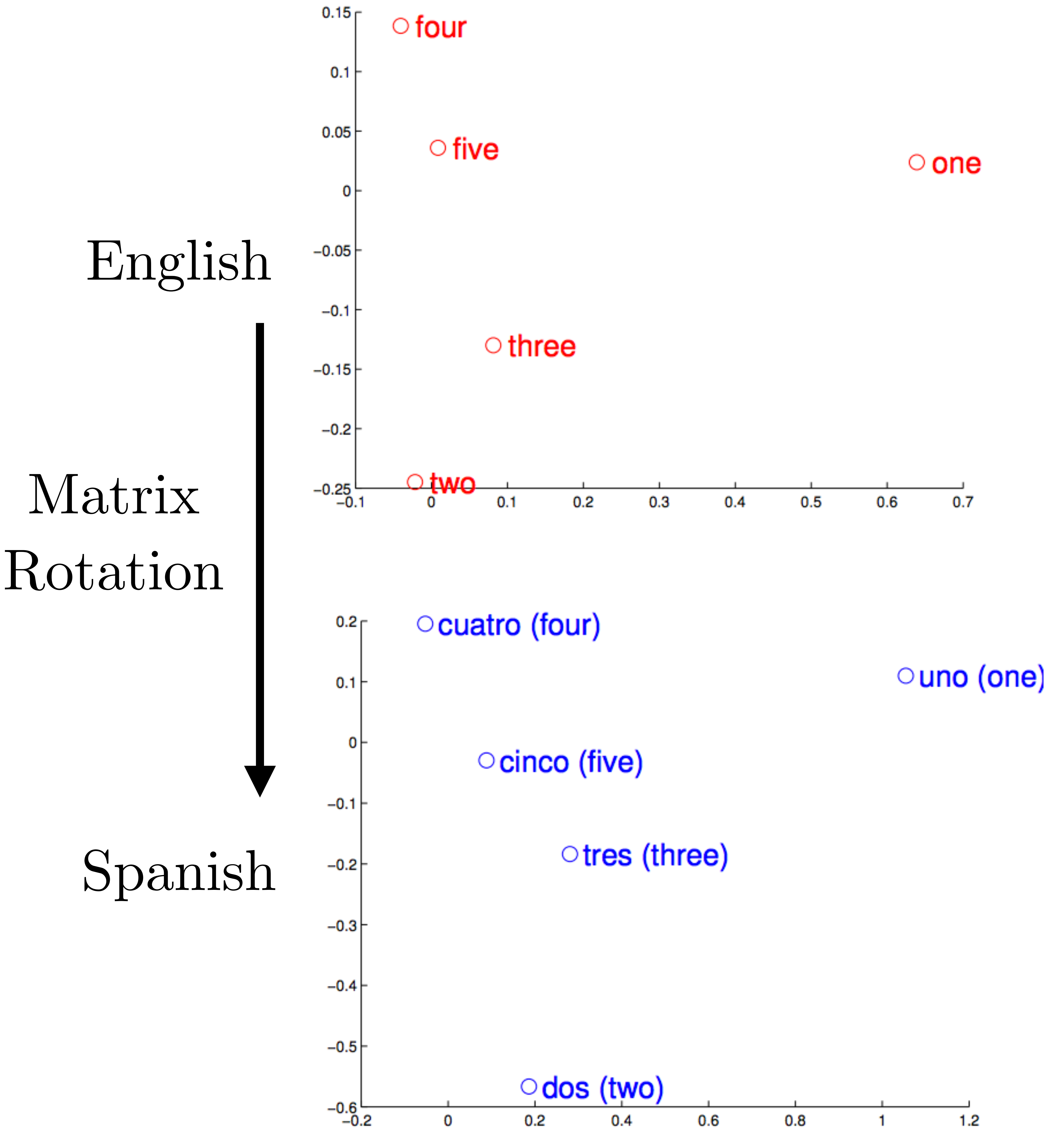
```
from gensim.models import Doc2Vec
fn = “item_document_vectors”
model = Doc2Vec.load(fn)
model.most_similar('pregnant')
matches = list(filter(lambda x: 'SENT_' in x[0], matches))

# ['...I am currently 23 weeks pregnant...',
#  '...I'm now 10 weeks pregnant...',
#  '...not showing too much yet...',
#  '...15 weeks now. Baby bump...',
#  '...6 weeks post partum!...',
#  '...12 weeks postpartum and am nursing...',
#  '...I have my baby shower that...',
#  '...am still breastfeeding...',
#  '...I would love an outfit for a baby shower...']
```

sentence search

translation

(using just a rotation matrix)



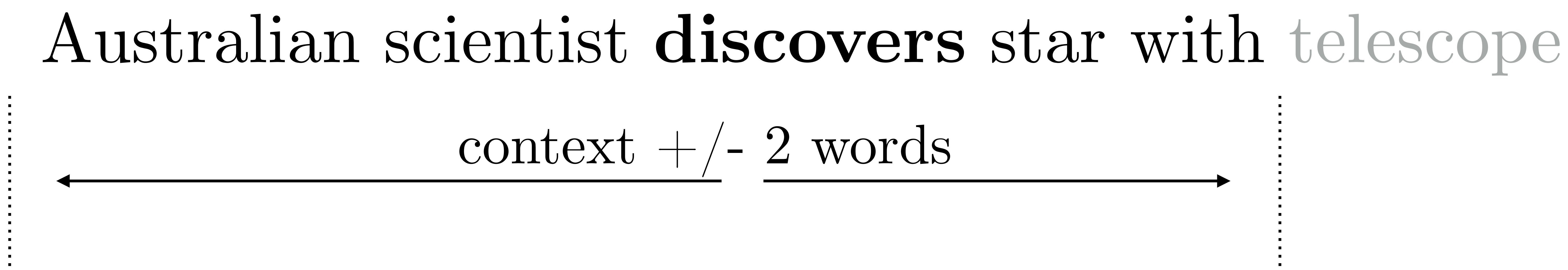
English

Matrix Rotation

Spanish

Mikolov 2013

context dependent



Australian scientist **discovers** star with telescope

context +/- 2 words

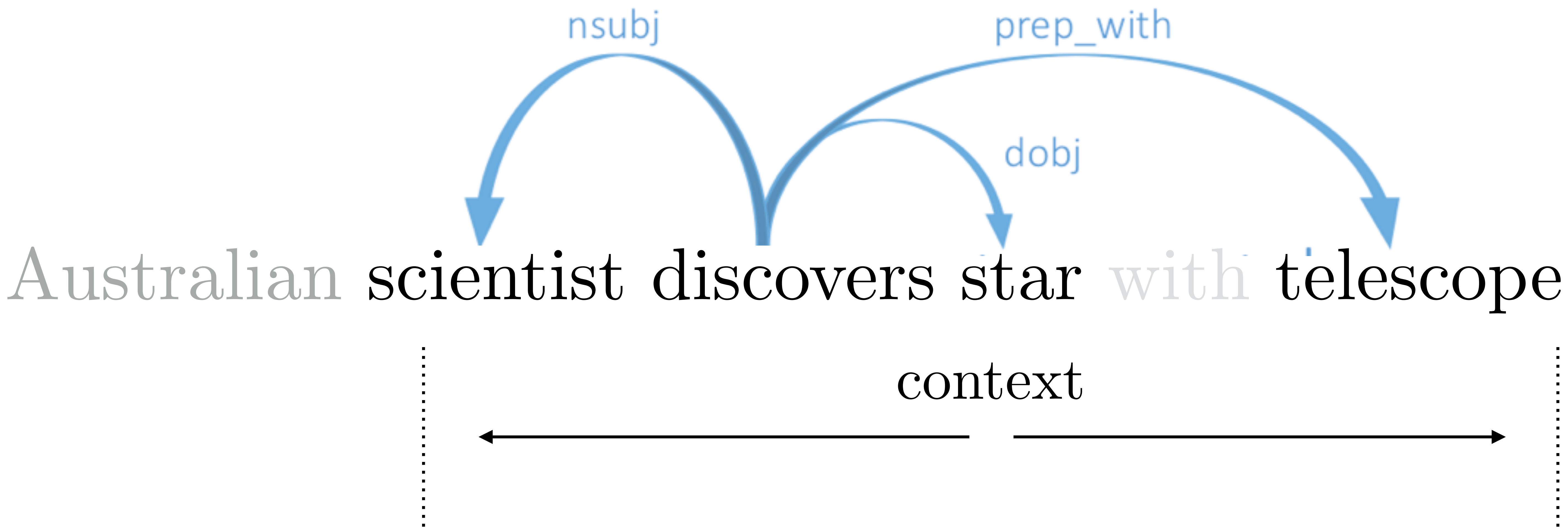
Levy & Goldberg
2014

context dependent

Australian scientist discovers star with telescope

Levy & Goldberg 2014

context dependent



nsubj

prep_with

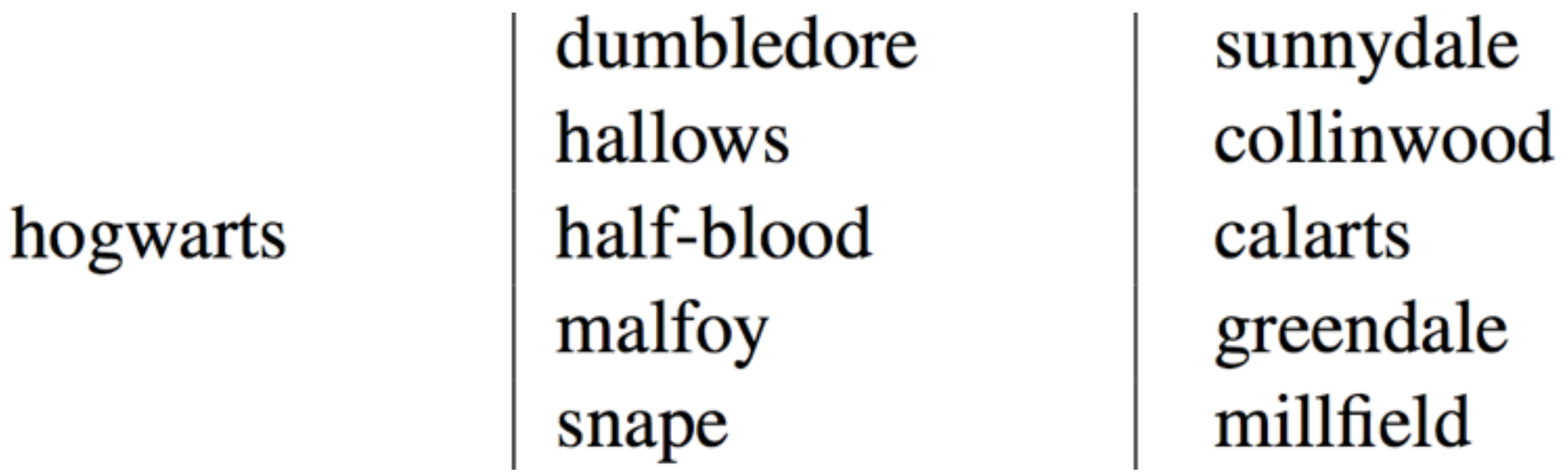
dobj

Australian scientist discovers star with telescope

context

Levy & Goldberg 2014

context dependent



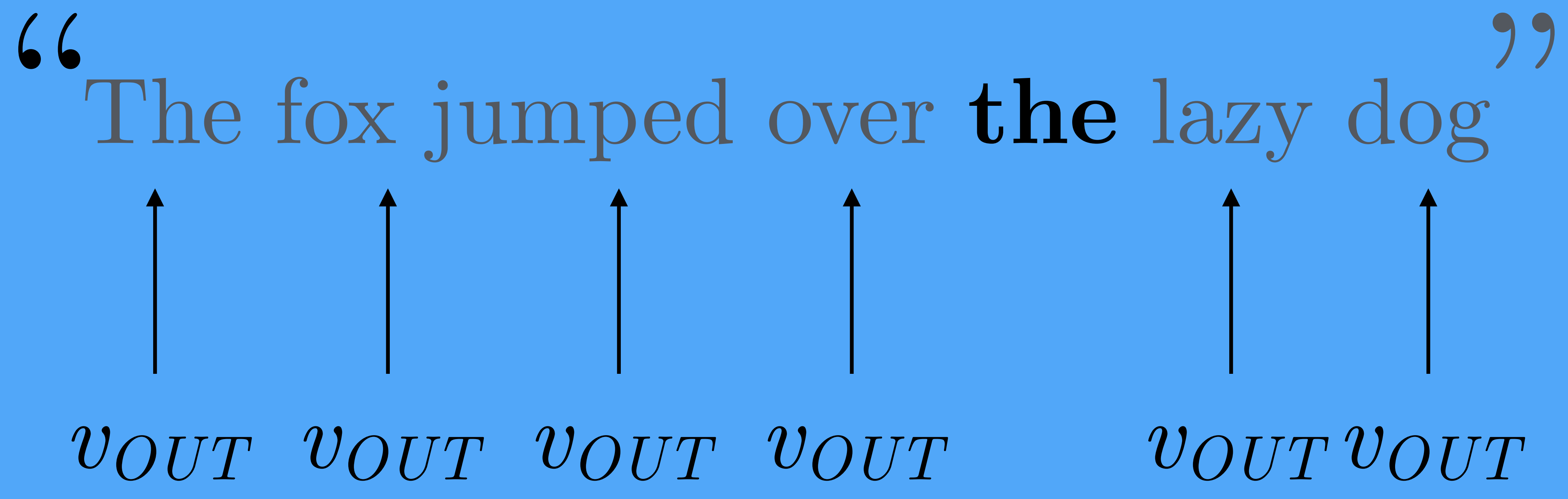
	BoW	DEPS
hogwarts	dumbledore hallows half-blood malfoy snape	sunnydale collinwood calarts greendale millfield

topically-similar vs ‘functionally’ similar

Levy & Goldberg 2014

word2vec

learn word vectors from sentences



“The fox jumped over **the** lazy dog”

v_{OUT} v_{OUT} v_{OUT} v_{OUT} v_{OUT} v_{OUT}

deepwalk

‘words’ are graph vertices

‘sentences’ are random walks on the graph

$$v_{46} \rightarrow v_{45} \rightarrow v_{71} \rightarrow v_{24} \rightarrow v_5$$

Perozzi et al 2014

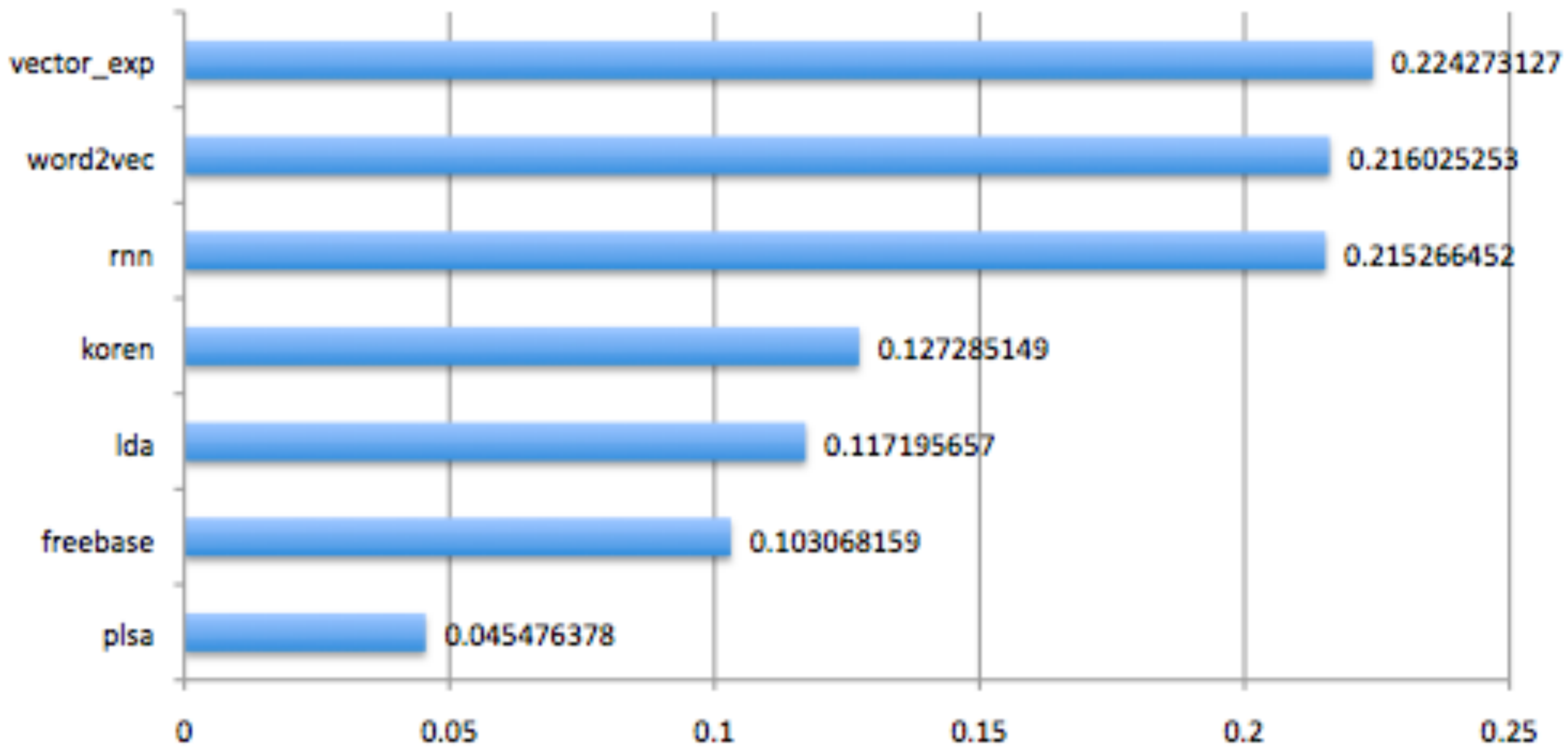
Playlists at Spotify

‘words’ are songs

‘sentences’ are playlists

Playlists at Spotify

Great performance on 'related artists'



Erik Bernhardsson

Fixes at Stitch Fix?

Let’s try:
‘words’ are styles
‘sentences’ are fixes

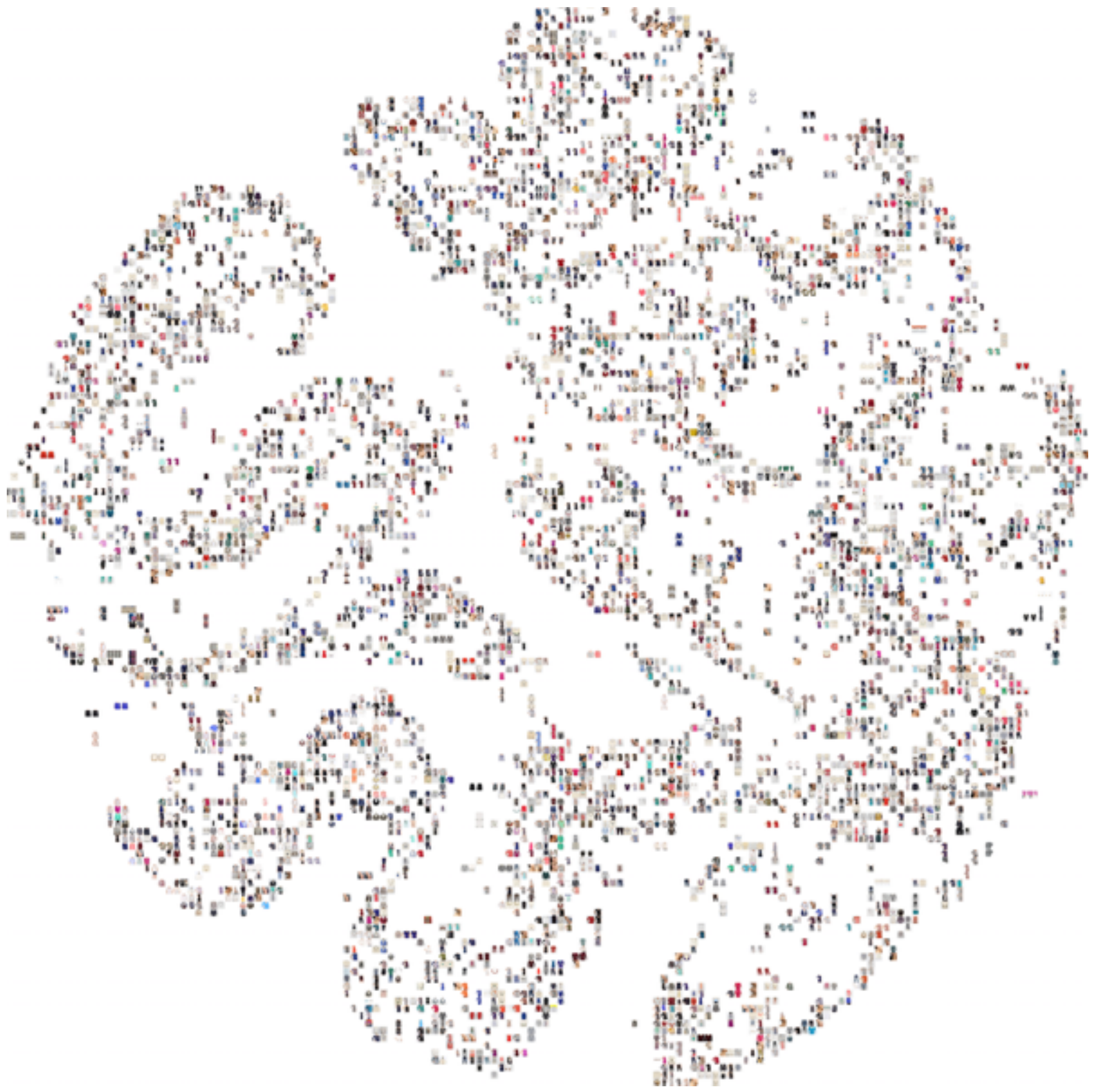
Fixes at Stitch Fix?

Learn similarity between styles because they co-occur

Learn ‘coherent’ styles

sequence learning

Fixes at Stitch Fix?



Got lots of structure!

sequence learning

?

GloVe

sequence
learning