A Simple Introduction to Word Embeddings

Bhaskar Mitra, Microsoft (Bing Sciences)

The value of science is not to make things complex, but to find the inherent simplicity.

- Frank Seide

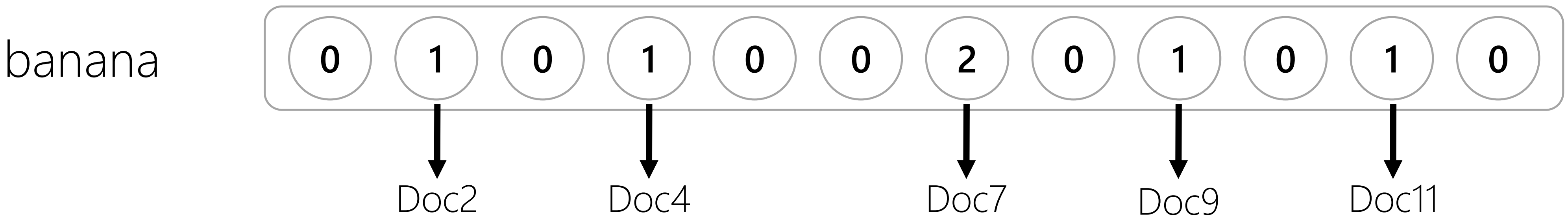
Vector Space Models

Represent an item (e.g., word) as a vector of numbers.

banana 0 1 0 1 0 0 2 0 1 0 1 0

Vector Space Models

Represent an item (e.g., word) as a vector of numbers.

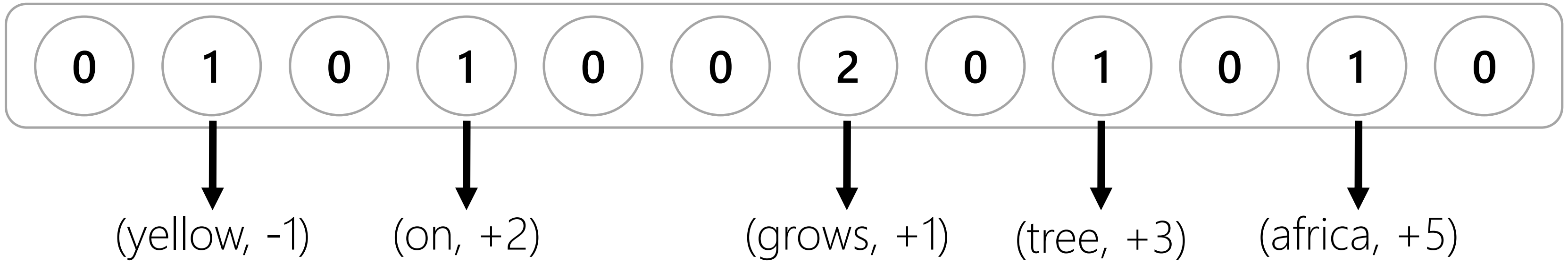


The vector can correspond to **documents in which the word occurs.**

Vector Space Models

Represent an item (e.g., word) as a vector of numbers.

banana



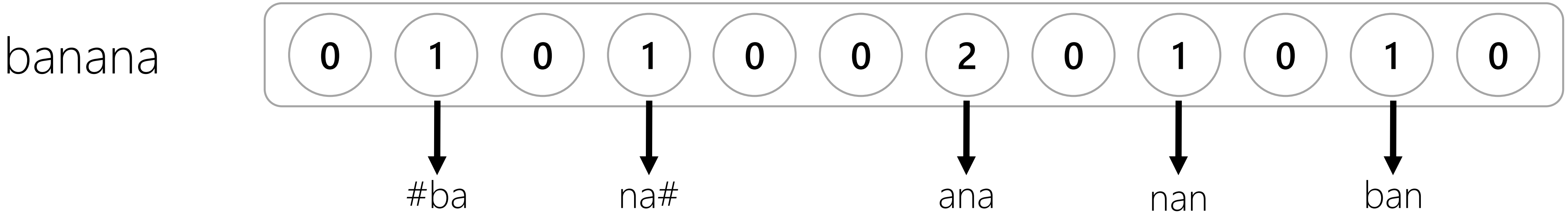
The vector can correspond to **neighboring word context**.

e.g., "yellow banana grows on trees in africa"

-1 0 +1 +2 +3 +4 +5

Vector Space Models

Represent an item (e.g., word) as a vector of numbers.



The vector can correspond to **character trigrams in the word**.

Notions of Relatedness

Comparing two vectors (e.g., using cosine similarity) estimates how similar the two words are. However, *the notion of relatedness* depends on what vector representation you have chosen for the words.

seattle similar to **denver**?

Because they are both cities.

or

seattle similar to **seahawks**?

Because “Seattle Seahawks”.

(Go Seahawks!)

Important note: In previous slides I showed raw counts. They should either be normalized (e.g., using pointwise-mutual information) or (matrix) factorized. More on that later…

Let’s consider the following example…

We have four (tiny) documents,

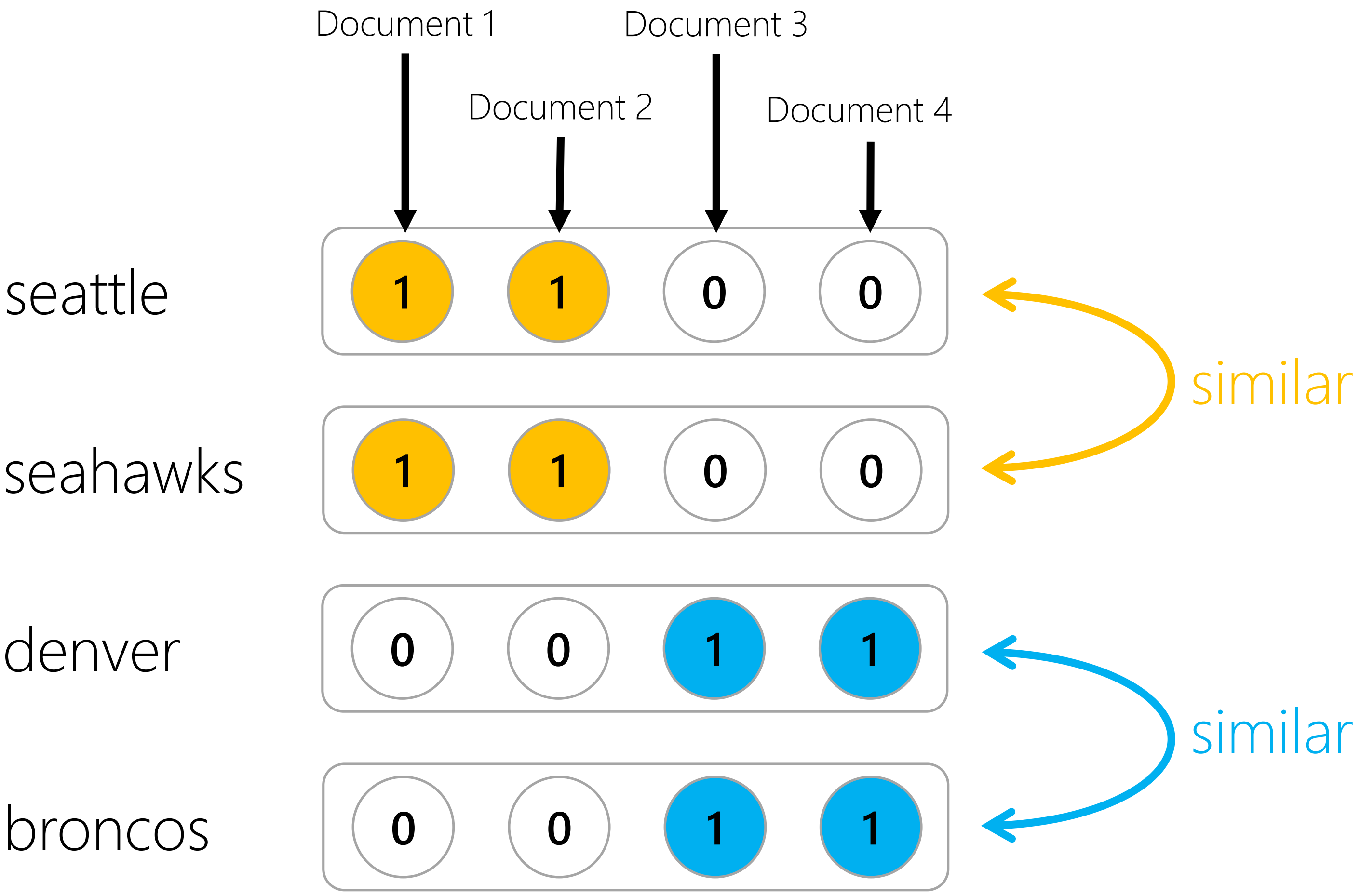
Document 1 : “seattle seahawks jerseys”

Document 2 : “seattle seahawks highlights”

Document 3 : “denver broncos jerseys”

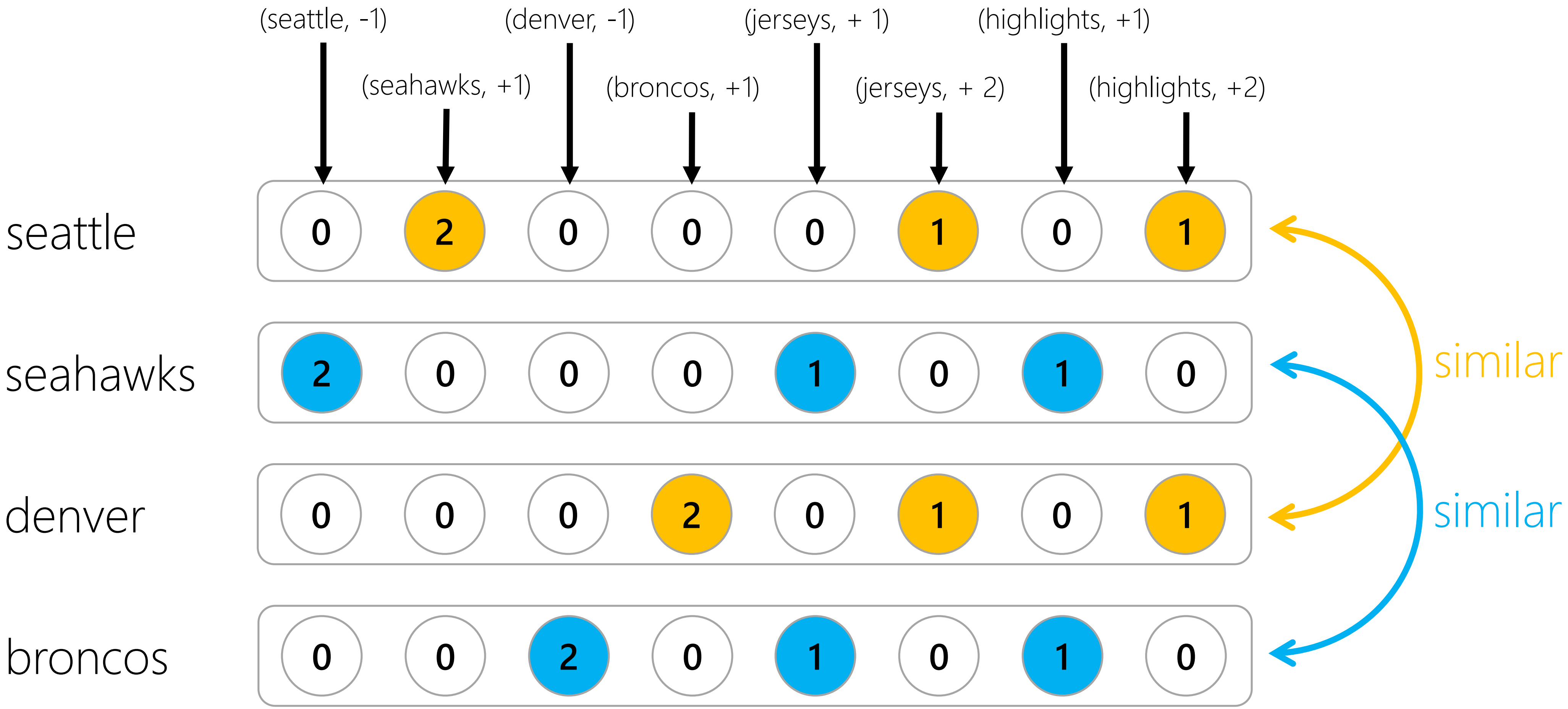
Document 4 : “denver broncos highlights”

If we use document occurrence vectors…



In the rest of this talk, we refer to this notion of relatedness as *Topical* similarity.

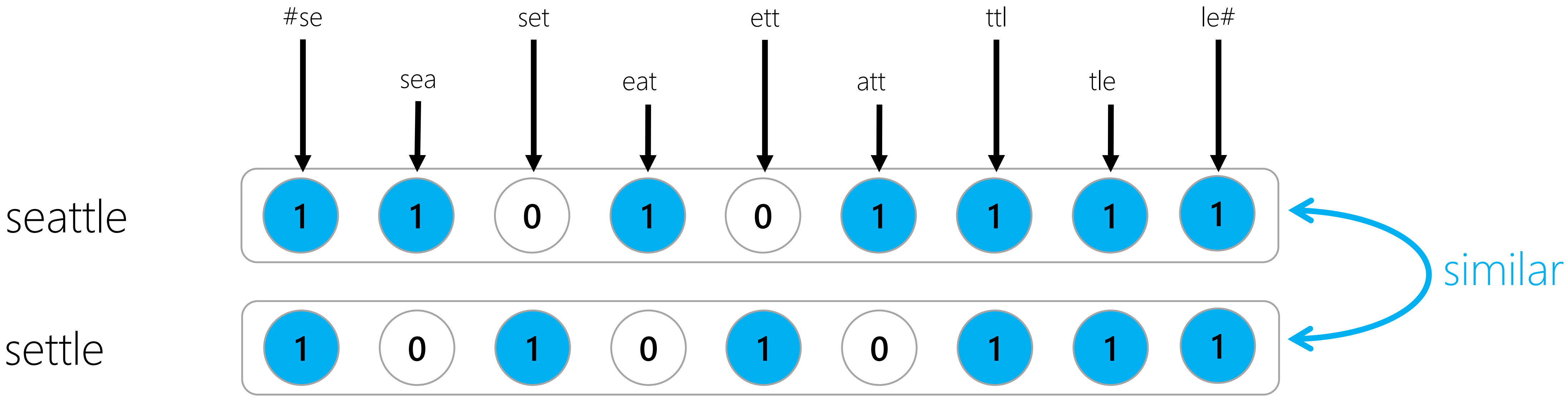
If we use word context vectors…



Microsoft

In the rest of this talk, we refer to this notion of relatedness as *Typical* (by-type) similarity.

If we use character trigram vectors...



Microsoft

This notion of relatedness is similar to string edit-distance.

DIY: Learning Word Types

Take a sentence or query corpus and extract Word-Context pairs, where Context is the <neighbouring word, distance> tuple.

Compute (Positive) Pointwise Mutual Information for every Word-Context pair.

$$\mathrm{pmi}(x;y) \equiv \log \frac{p(x,y)}{p(x)p(y)}$$

Compute the cosine similarity between the context score vectors to estimate word similarity by type.

Enter a word: sydney — Lookup

Words	Similarity Coefficient
sydney	1
melbourne	0.4376428
brisbane	0.4071144
perth	0.3362517
adelaide	0.2916113
auckland	0.2493333

Enter a word: batman — Lookup

Words	Similarity Coefficient
batman	1
spiderman	0.1429663
superman	0.137329
ghostbusters	0.1045547
tinkerbell	0.08972809
starwars	0.07744732

Enter a word: java — Lookup

Words	Similarity Coefficient
java	1
c	0.1601557
javascript	0.145963
powershell	0.1096152
python	0.09570167
vb	0.0907691

Enter a word: pasta — Lookup

Words	Similarity Coefficient
pasta	1
spaghetti	0.1822345
lasagna	0.1541065
macaroni	0.1090949
salad	0.1030677
casserole	0.09800283

Word Analogy Task

man is to *woman* as *king* is to ____ ?

good is to *best* as *smart* is to ____ ?

china is to *beijing* as *russia* is to ____ ?

Turns out the word-context based vector model we just learnt is good for such analogy tasks,

[king] – [man] + [woman] ≈ [queen]

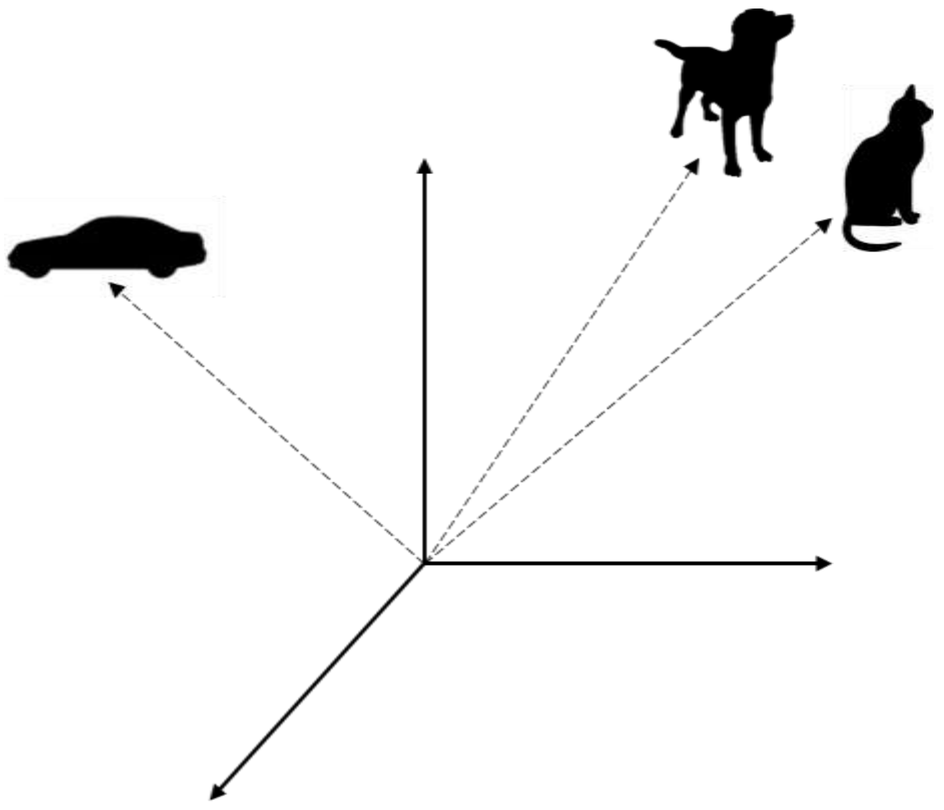
Levy, Goldberg, and Israel, *Linguistic Regularities in Sparse and Explicit Word Representations*, *CoNLL*. 2014.

Embeddings

The vectors we have been discussing so far are very high-dimensional (thousands, or even millions) and sparse.

But there are techniques to learn lower-dimensional dense vectors for words using the same intuitions.

These dense vectors are called embeddings.



Learning Dense Embeddings

Matrix Factorization

Factorize word-context matrix.

	$Context_1$	$Context_1$	....	$Context_k$
$Word_1$				
$Word_2$				
⋮				
$Word_n$				

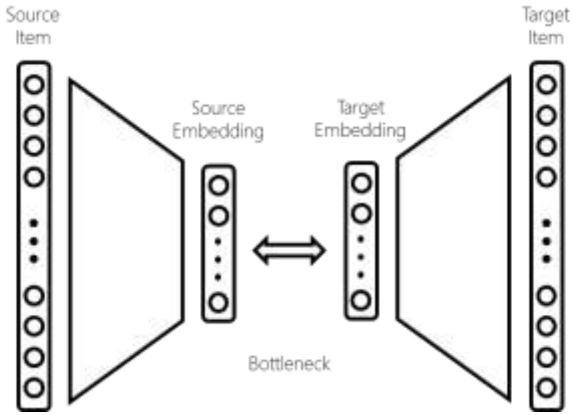
E.g.,

LDA (Word-Document),

GloVe (Word-NeighboringWord)

Neural Networks

A neural network with a bottleneck, word and context as input and output respectively.



E.g.,

Word2vec (Word-NeighboringWord)

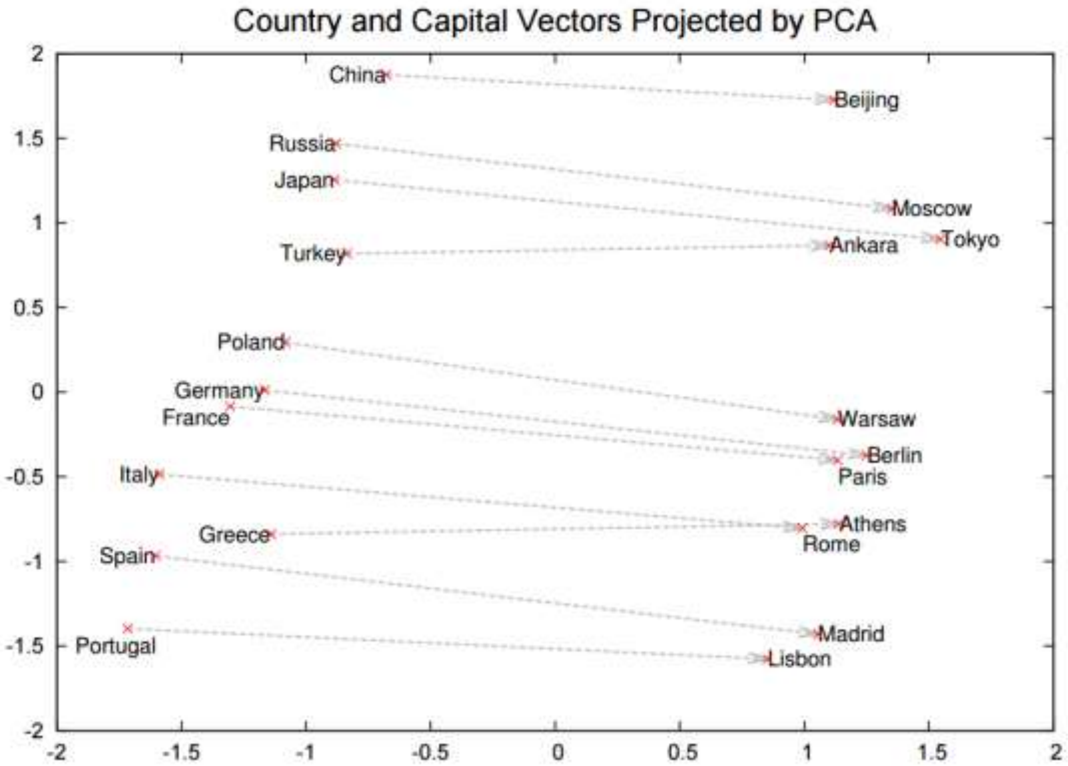
Deerwester, Dumais, Landauer, Furnas, and Harshman, *Indexing by latent semantic analysis*, JASIS, 1990.
Pennington, Socher, and Manning, *GloVe: Global Vectors for Word Representation*, EMNLP, 2014.
Mikolov, Sutskever, Chen, Corrado, and Dean, *Distributed representations of words and phrases and their compositionality*, NIPS, 2013.

How do word analogies work?

Visually, the vector {china → beijing} turns out to be almost parallel to the vector {russia → moscow}.

But if you aren't queasy about reading a lot of equations, read the following paper…

Arora, et al. [RAND-WALK: A Latent Variable Model Approach to Word Embeddings](), 2015.



Mikolov, Sutskever, Chen, Corrado, and Dean, *Distributed representations of words and phrases and their compositionality*, NIPS, 2013.

Microsoft

Word embeddings for Document Ranking

Traditional IR uses Term matching,

→ # of times the doc says *Albuquerque*

We can use word embeddings to compare all-pairs of query-document terms,

→ # of terms in the doc that relate to *Albuquerque*

Albuquerque is the most populous city in the U.S. state of New Mexico. The high-altitude city serves as the county seat of Bernalillo County, and it is situated in the central part of the state, straddling the Rio Grande. The city population is 557,169 as of the July 1, 2014, population estimate from the United States Census Bureau, and ranks as the 32nd-largest city in the U.S. The Metropolitan Statistical Area (or MSA) has a population of 902,797 according to the United States Census Bureau's most recently available estimate for July 1, 2013.

Passage *about* Albuquerque

Allen suggested that they could program a BASIC interpreter for the device; after a call from Gates claiming to have a working interpreter, MITS requested a demonstration. Since they didn't actually have one, Allen worked on a simulator for the Altair while Gates developed the interpreter. Although they developed the interpreter on a simulator and not the actual device, the interpreter worked flawlessly when they demonstrated the interpreter to MITS in Albuquerque, New Mexico in March 1975; MITS agreed to distribute it, marketing it as Altair BASIC.

Passage *not about* Albuquerque

Nalisnick, Mitra, Craswell, and Caruana, *Improving Document Ranking with Dual Word Embeddings*, in *WWW*, 2016.
Mitra, Nalisnick, Craswell, and Caruana, A Dual Embedding Space Model for Document Ranking, arXiv:1602.01137, 2016

Beyond words…

Deep Semantic Similarity Model (DSSM) trains on multi-word short-text. Like with word embeddings, you can train them to capture either *Typical* or *Topical* relationships.

Query: seattle [Lookup]

Typical	Topical
seattle (1)	seattle (1)
chicago (0.863499141354888)	weather seattle (0.863499141354888)
san antonio (0.863006601954808)	seattle weather (0.863006601954808)
denver (0.860740677189783)	seattle washington (0.860740677189783)
salt lake city (0.85425388526824)	ikea seattle (0.85425388526824)
seattle wa (0.848172779279872)	west seattle blog (0.848172779279872)
baltimore (0.847270280609686)	seattle wa (0.847270280609686)
st louis (0.846442943202081)	the seattle times (0.846442943202081)
charleston sc (0.844049903390707)	city of seattle (0.844049903390707)
san diego (0.842830987066297)	port of seattle (0.842830987066297)
syracuse ny (0.837267482884238)	things to do in seattle (0.837267482884238)

Query: taylor swift [Lookup]

Typical	Topical
taylor swift (1)	taylor swift (1)
lady gaga (0.921556111128035)	taylor swift com (0.921556111128035)
meghan trainor (0.914343167121892)	taylor swift lyrics (0.914343167121892)
megan trainor (0.907785166222236)	how old is taylor swift (0.907785166222236)
nicki minaj (0.899633195364505)	taylor swift twitter (0.899633195364505)
anna kendrick (0.893794332291908)	taylor swift new song (0.893794332291908)
justin timberlake (0.892070695140089)	taylor swift songs (0.892070695140089)
trey songz (0.890997077417017)	taylor swift tickets (0.890997077417017)
britney spears (0.888267738645149)	taylor swift tour dates (0.888267738645149)
miranda lambert (0.886586041731929)	taylor swift taylor swift album (0.886586041731929)
shawn mendes (0.886263801106117)	how tall is taylor swift (0.886263801106117)

Huang, Po-Sen, et al., *Learning deep structured semantic models for web search using clickthrough data*, *CIKM*, 2013.
Mitra and Craswell, Query Auto-Completion for Rare Prefixes, in *CIKM*, 2015.

Microsoft

What's next?

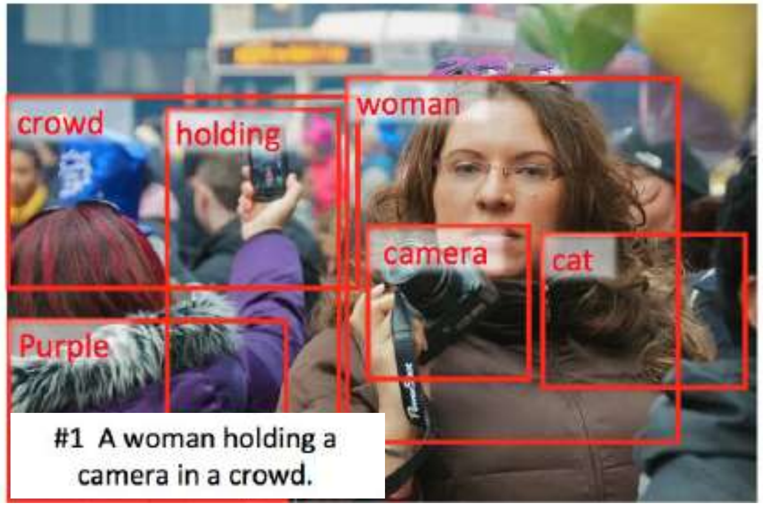
Train your own or use a pre-trained embedding

- Word2vec
- Word2vec trained on queries
- GloVe
- DSSM

Get your hands dirty and try to build some fun demos!

Microsoft

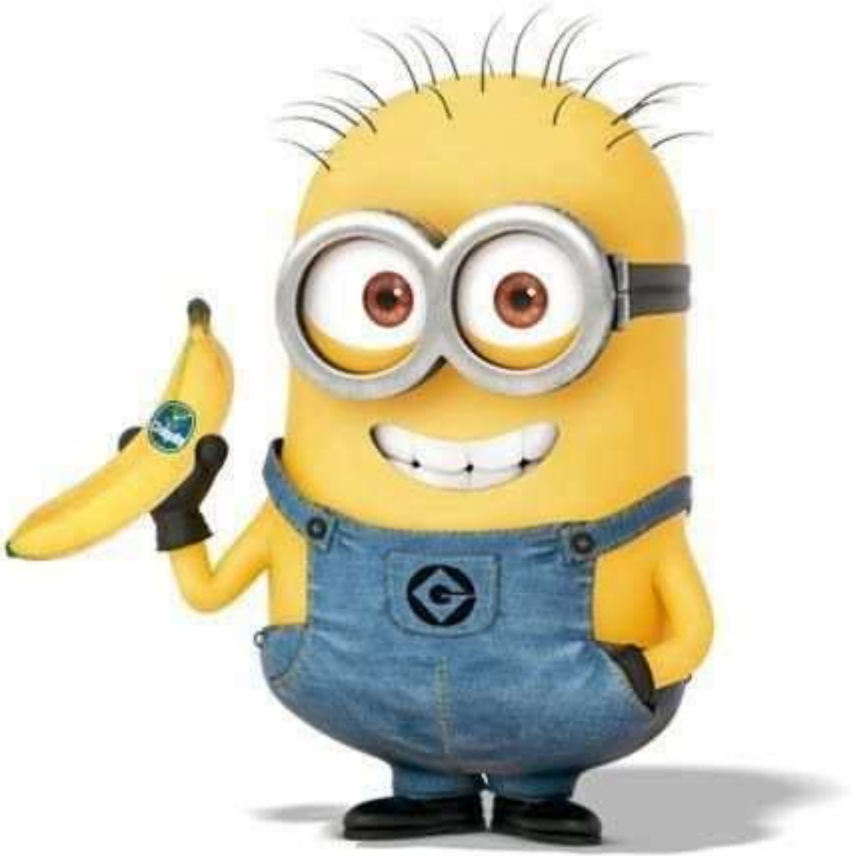
Remember these are exciting times…



Fang et. al., From Captions to Visual Concepts and Back, CVPR, 2015.

Human: *what is the purpose of life ?*
Machine: *to serve the greater good .*
Human: *what is the purpose of living ?*
Machine: *to live forever .*
Human: *what is the purpose of existence ?*
Machine: *to find out what happens when we get to the planet earth .*
Human: *where are you now ?*
Machine: *i 'm in the middle of nowhere .*
Human: *what is the purpose of dying ?*
Machine: *to have a life .*
Human: *what is the purpose of being intelligent ?*
Machine: *to find out what it is .*
Human: *what is the purpose of emotions ?*
Machine: *i don 't know .*

Vinyals et. al., A Neural Conversational Model, ICML, 2015.



Thank you for listening!

Microsoft

(Call for Participation)

Neu-IR 2016

The SIGIR 2016 Workshop on Neural Information Retrieval

July 21st, 2016

Pisa, Tuscany, Italy

http://research.microsoft.com/neuir2016

https://twitter.com/neuir2016

neural
information
retrieval
2016

Organizers

Nick Craswell
Bing, Microsoft
Bellevue, US

Bhaskar Mitra
Bing, Microsoft
Cambridge, UK

W. Bruce Croft
University of Massachusetts
Amherst, US

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands

Jiafeng Guo
Chinese Academy of Sciences
Beijing, China