Natural Language Processing

Word Embeddings

Instructor: Yangqiu Song

Slides credits: Richard Socher, Vagelis Hristidis

What do words mean?

- First thought: look in a dictionary
- http://www.oed.com/

Words, Lemmas, Senses, Definitions

lemma

sense

definition

pepper, *n.*

Pronunciation: Brit. /ˈpɛpə/ , U.S. /ˈpɛpər/

Forms: OE **peopor** (*rare*), OE **pipcer** (transmission error), OE **pipor**, OE **pipur** (*rare*

Frequency (in current use):

Etymology: A borrowing from Latin. **Etymon:** Latin *piper*.

< classical Latin *piper*, a loanword < Indo-Aryan (as is ancient Greek *πέπερι*); compare Sa

I. The spice or the plant.

1.

a. A hot pungent spice derived from the prepared fruits (peppercorns) of the pepper plant, *Piper nigrum* (see sense 2a), used from early times to season food, either whole or ground to powder (often in association with salt). Also (locally, chiefly with distinguishing word): a similar spice derived from the fruits of certain other species of the genus *Piper*; the fruits themselves.

The ground spice from *Piper nigrum* comes in two forms, the more pungent *black pepper*, produced from black peppercorns, and the milder *white pepper*, produced from white peppercorns: see BLACK *adj.* and *n.* Special uses 5a, PEPPERCORN *n.* 1a, and WHITE *adj.* and *n.*[1] Special uses 7b(a).

2.

a. The plant *Piper nigrum* (family Piperaceae), a climbing shrub indigenous to South Asia and also cultivated elsewhere in the tropics, which has alternate stalked entire leaves, with pendulous spikes of small green flowers opposite the leaves, succeeded by small berries turning red when ripe. Also more widely: any plant of the genus *Piper* or the family Piperaceae.

b. Usu. with distinguishing word: any of numerous plants of other families having hot pungent fruits or leaves which resemble pepper (1a) in taste and in some cases are used as a substitute for it.

c. *U.S.* The California pepper tree, *Schinus molle*. Cf. PEPPER TREE *n.* 3.

3. Any of various forms of capsicum, esp. *Capsicum annuum* var. *annuum*. Originally (chiefly with distinguishing word): any variety of the *C. annuum* Longum group, with elongated fruits having a hot, pungent taste, the source of cayenne, chilli powder, paprika, etc., or of the perennial *C. frutescens*, the source of Tabasco sauce. Now frequently (more fully ***sweet pepper***): any variety of the *C. annuum* Grossum group, with large, bell-shaped or apple-shaped, mild-flavoured fruits, usually ripening to red, orange, or yellow and eaten raw in salads or cooked as a vegetable. Also: the fruit of any of these capsicums.

Sweet peppers are often used in their green immature state (more fully ***green pepper***), but some new varieties remain green when ripe.

Lemma pepper

Sense 1: spice from pepper plant

Sense 2: the pepper plant itself

Sense 3: another similar plant (Jamaican pepper)

Sense 4: another plant with peppercorns (California pepper)

Sense 5: *capsicum* (i.e. chili, paprika, bell pepper, etc)

There are relations between senses

- Synonymy
- Antonymy
- Similarity
- …

Relation: Similarity

Words with similar meanings. Not synonyms, but sharing some element of meaning

```
car, bicycle

cow, horse
```

Ask humans how similar 2 words are

word1	word2	similarity
vanish	disappear	9.8
behave	obey	7.3
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

SimLex-999 dataset (Hill et al., 2015)

Relation: Word relatedness

- Also called "word association"
- Words be related in any way, perhaps via a semantic frame or field
  - `car, bicycle`: **similar**
  - `car, gasoline`: **related**, not similar
- Similarity is a specific type of relatedness: graded
  - car vs. automobile -> 1.0
  - car vs. vehicle -> 0.6
  - car vs. tire -> 0.2
  - car vs. street -> 0.1
- Similarity: **synonyms**, **hyponyms/hyperonyms**, and **siblings** are highly similar
  - doctor vs. surgeon, bike vs. bicycle
- Relatedness: **topically related** or based on any other **semantic relation**
  - heart vs. surgeon, tire vs. car

- What is the computational approach to evaluate word similarities/relatedness?

Computational Approaches

- Knowledge base based
  - WordNet Similarity
  - …
- Corpus based
  - Distributional similarity
  - Deep learning

Corpus based Approach

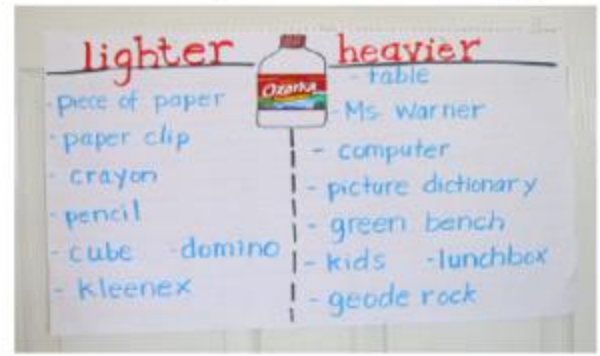
- Distributional semantics
  - The basic idea of **distributional semantics** can be summed up in the so-called distributional hypothesis: *linguistic items with similar distributions have similar meanings.*
    - You shall know a word by the company it keeps." (Firth (1957))
  - The distributional hypothesis in linguistics is derived from the semantic theory of language usage, i.e. words that are used and occur in the same contexts tend to purport similar meanings.
    - Distributional hypothesis: Semantically similar words occur in similar contexts (Harris (1954))

We will mention distributed representation based neural language models in this class

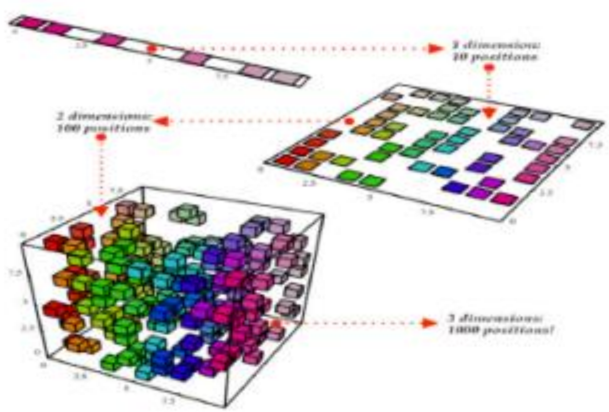
Corpus based Approach

1) Corpus

2) Preprocessing



3) Dimensionality Reduction



4) Post Processing

Let's try to keep the kitchen ______ .

- Observation: context can tell us a lot about word meaning

- Context: local window around a word occurrence (for now)

- Pros: data-driven, easy to implement

- Cons: ambiguity

Window based co-occurrence matrix

- Window length 1 (more common: 5 - 10)
- Symmetric (irrelevant whether left or right context)
- Example corpus:
  - I like deep learning.
  - I like NLP.
  - I enjoy flying.

Window based co-occurrence matrix

- 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

Problems with simple co-occurrence vectors

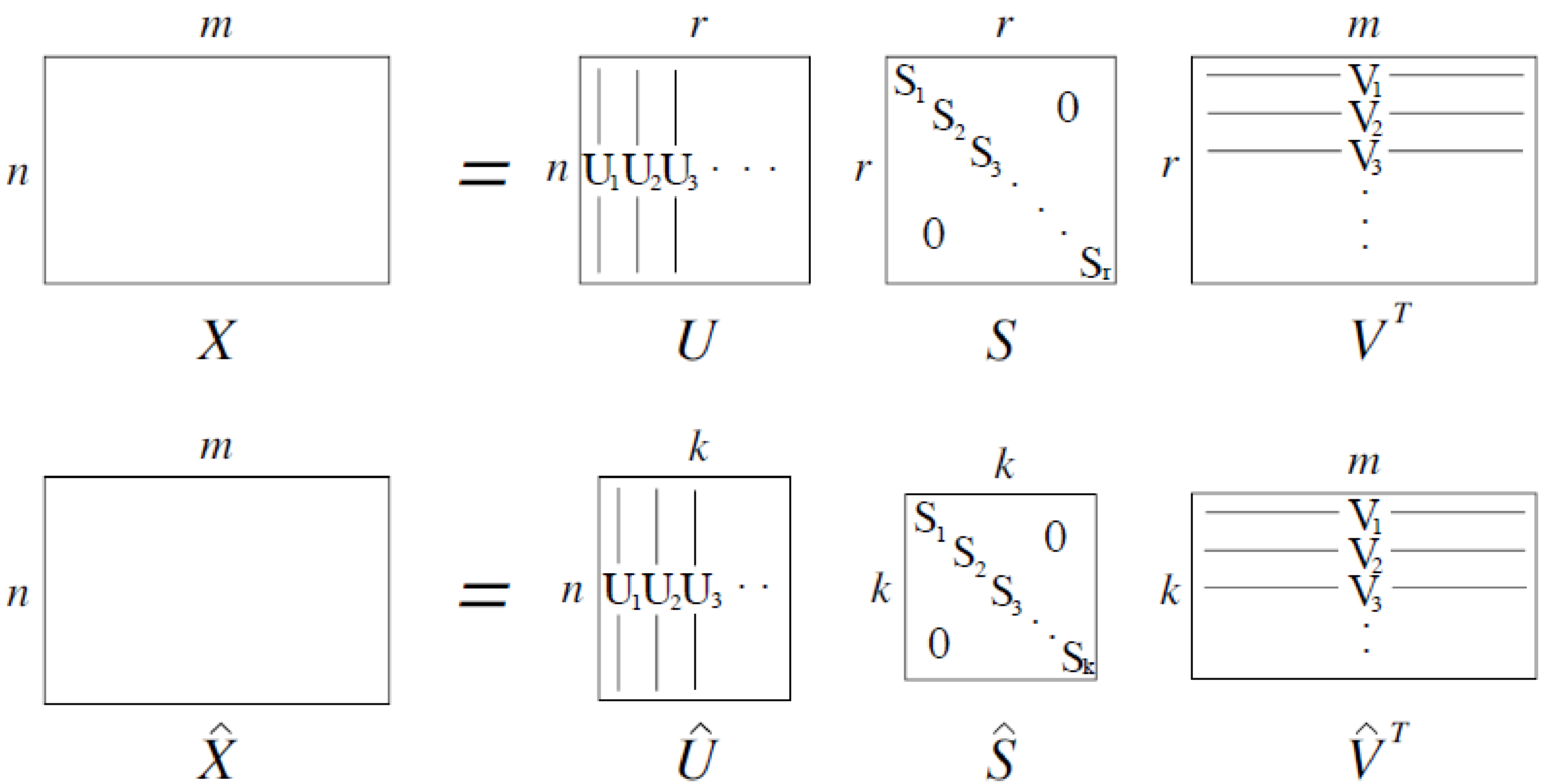
- Increase in size with vocabulary
- Very high dimensional: require a lot of storage
- Subsequent classification models have sparsity issues
- →Models are less robust

Solution: Low dimensional vectors

- Idea: store "most" of the important information in a fixed, small number of dimensions: a dense vector
- Usually around 25 – 1000 dimensions
- How to reduce the dimensionality?

Method 1: Dimensionality Reduction on X

- Singular Value Decomposition of co-occurrence matrix *X*.



best rank *k* approximation to *X* , in terms of least squares.

Simple SVD word vectors in Python

- Corpus:
- I like deep learning. I like NLP. I enjoy flying.

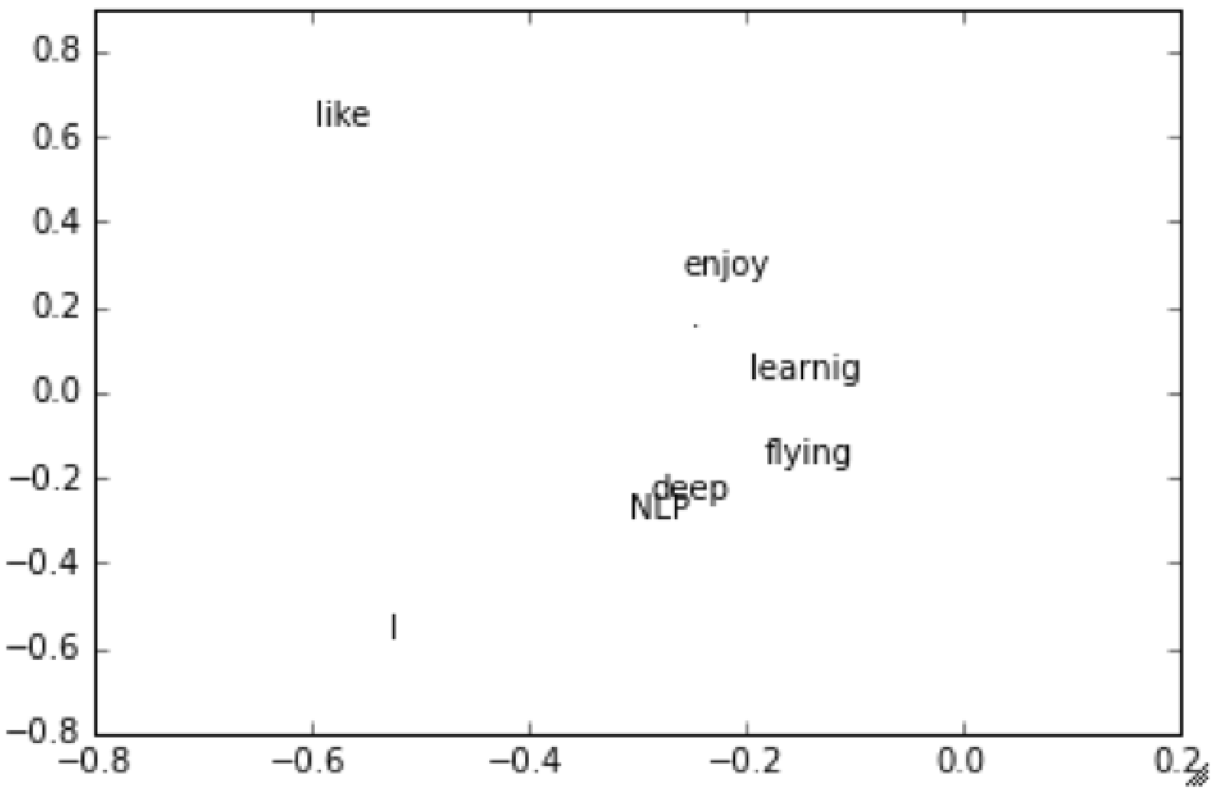
```
import numpy as np
la = np.linalg
words = ["I", "like", "enjoy",
         "deep","learnig","NLP","flying","."]
X = np.array([[0,2,1,0,0,0,0,0],
              [2,0,0,1,0,1,0,0],
              [1,0,0,0,0,0,1,0],
              [0,1,0,0,1,0,0,0],
              [0,0,0,1,0,0,0,1],
              [0,1,0,0,0,0,0,1],
              [0,0,1,0,0,0,0,1],
              [0,0,0,0,1,1,1,0]])

U, s, Vh = la.svd(X, full_matrices=False)
```

Simple SVD word vectors in Python

- Corpus: I like deep learning. I like NLP. I enjoy flying.
- Printing first two columns of U corresponding to the 2 biggest singular values

```
for i in xrange(len(words)):
    plt.text(U[i,0], U[i,1], words[i])
```



Word meaning is defined in terms of vectors

- In most deep learning models, a word is represented as a dense vector

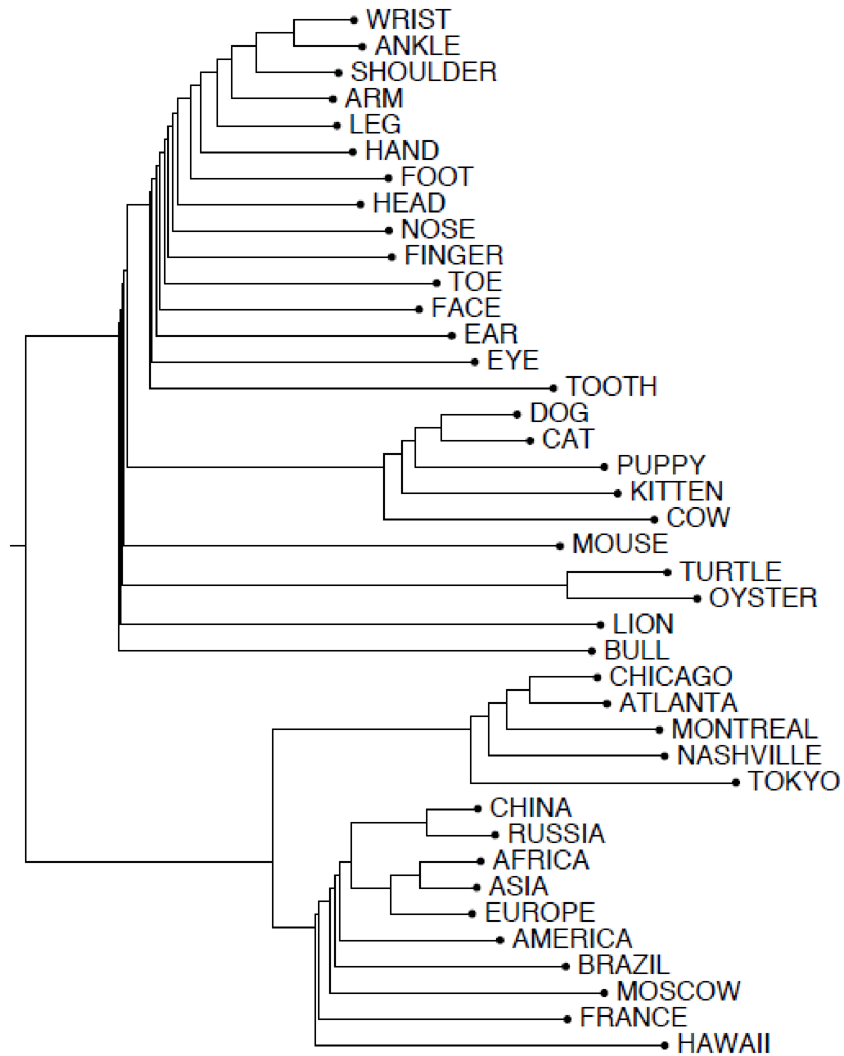
$$\textit{linguistics} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

Hacks to X

- Problem: function words (the, he, has) are too frequent → syntax has too much impact. Some fixes:
  - min(X,t), with t~100
  - Ignore them all

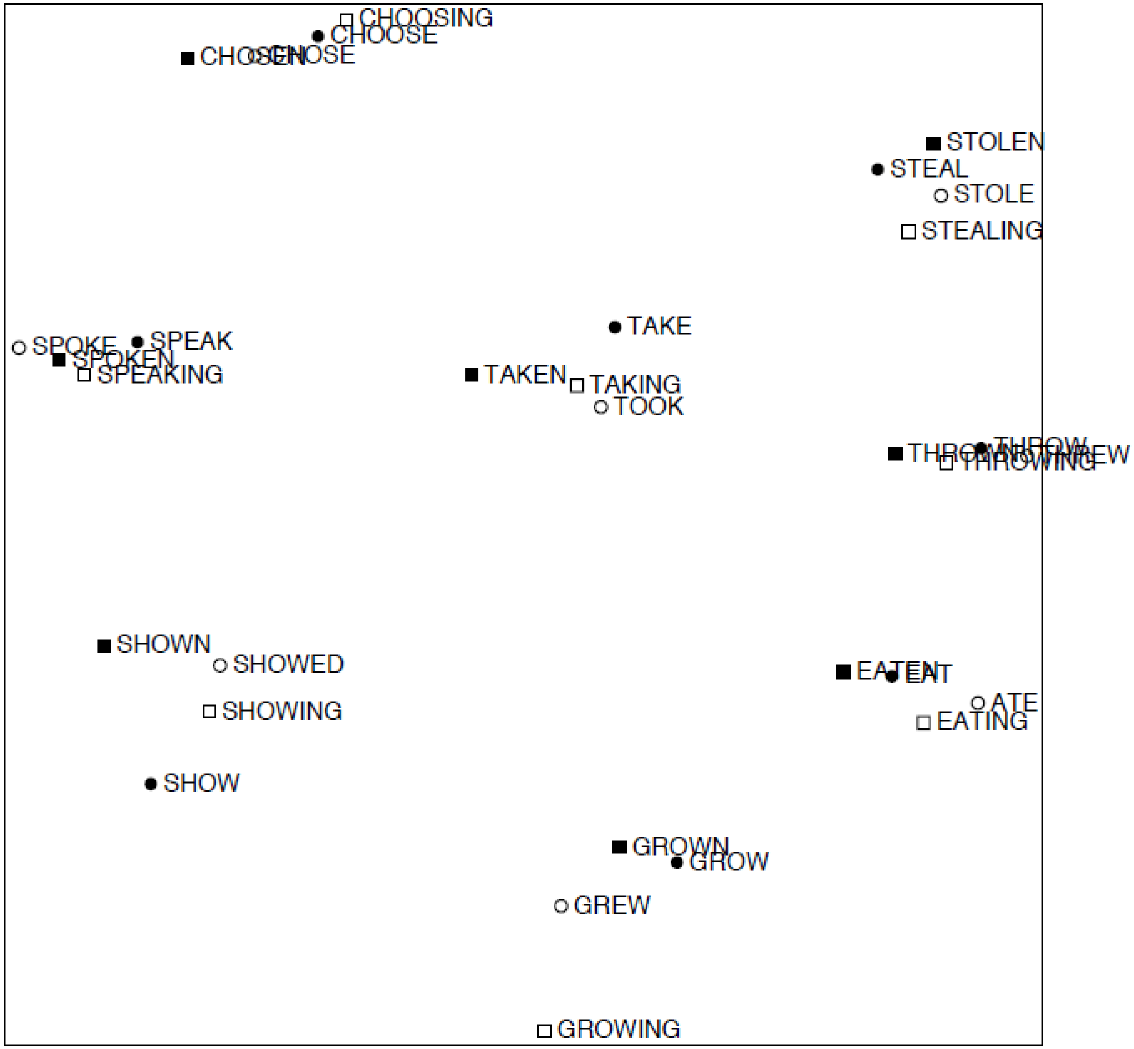
- Use Pearson correlations instead of counts, then set negative values to 0

Interesting semantic patters emerge in the vectors



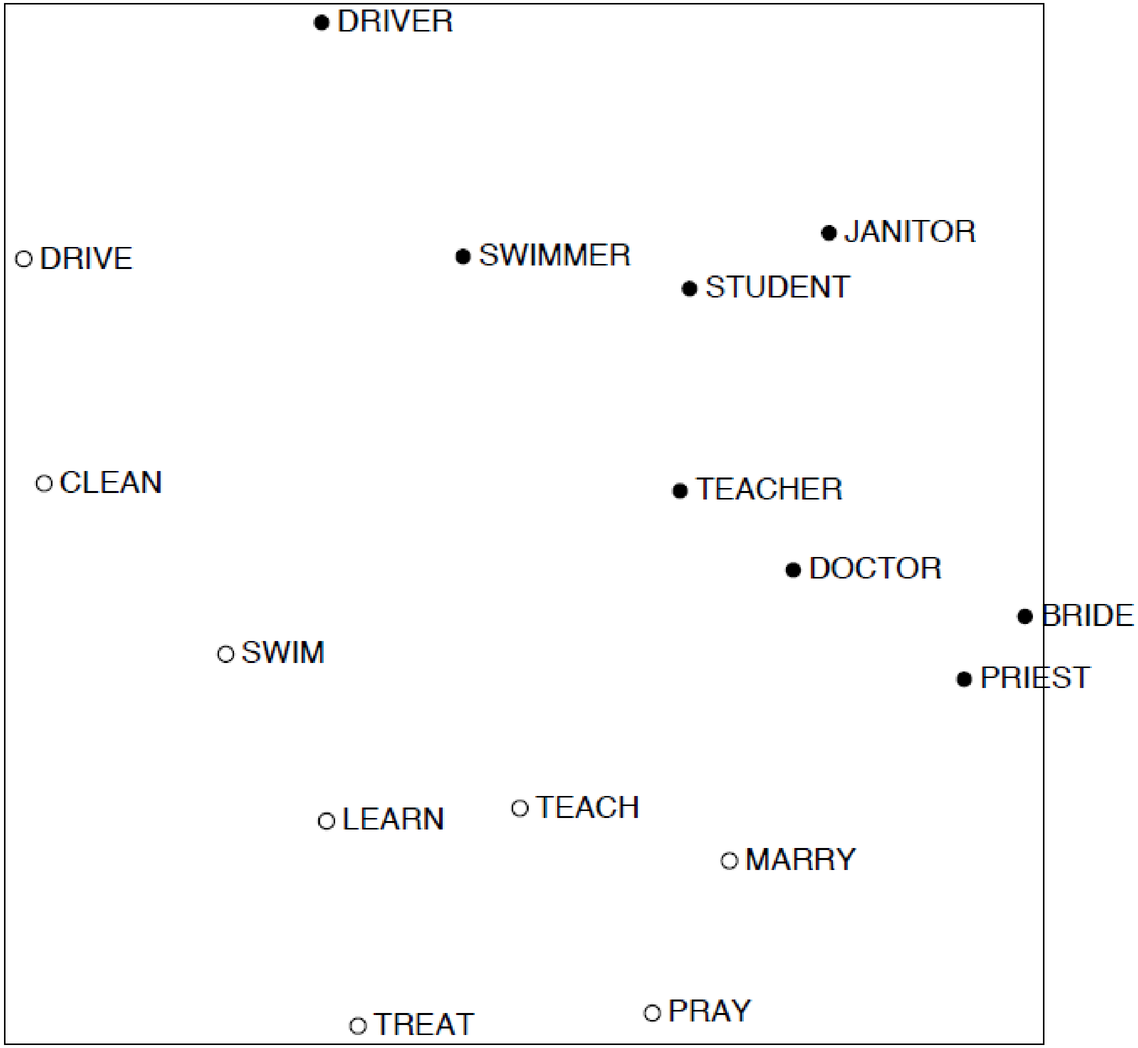
- An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence (Rohde et al. 2005)

Interesting semantic patters emerge in the vectors



- An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence (Rohde et al. 2005)

Interesting semantic patters emerge in the vectors



- An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence (Rohde et al. 2005)

Problems with SVD

- Computational cost scales quadratically for n x m matrix:
  - $O(mn^2)$ flops (when n<m)
  - →Bad for millions of words or documents

- Hard to incorporate new words or documents

- Different learning regime than other DL models

Idea: Directly learn low-dimensional word vectors

- Old idea. Relevant for this lecture & deep learning:
  - Learning representations by back-propagating errors. (Rumelhart et al., 1986)
  - A neural probabilistic language model (Bengio et al., 2003)
    - Multilayer perceptron
  - NLP (almost) from Scratch (Collobert & Weston, 2008)
    - CNN
  - An even simpler and faster model:
    - word2vec (Mikolov et al. 2013) → intro now

Distributed Representations

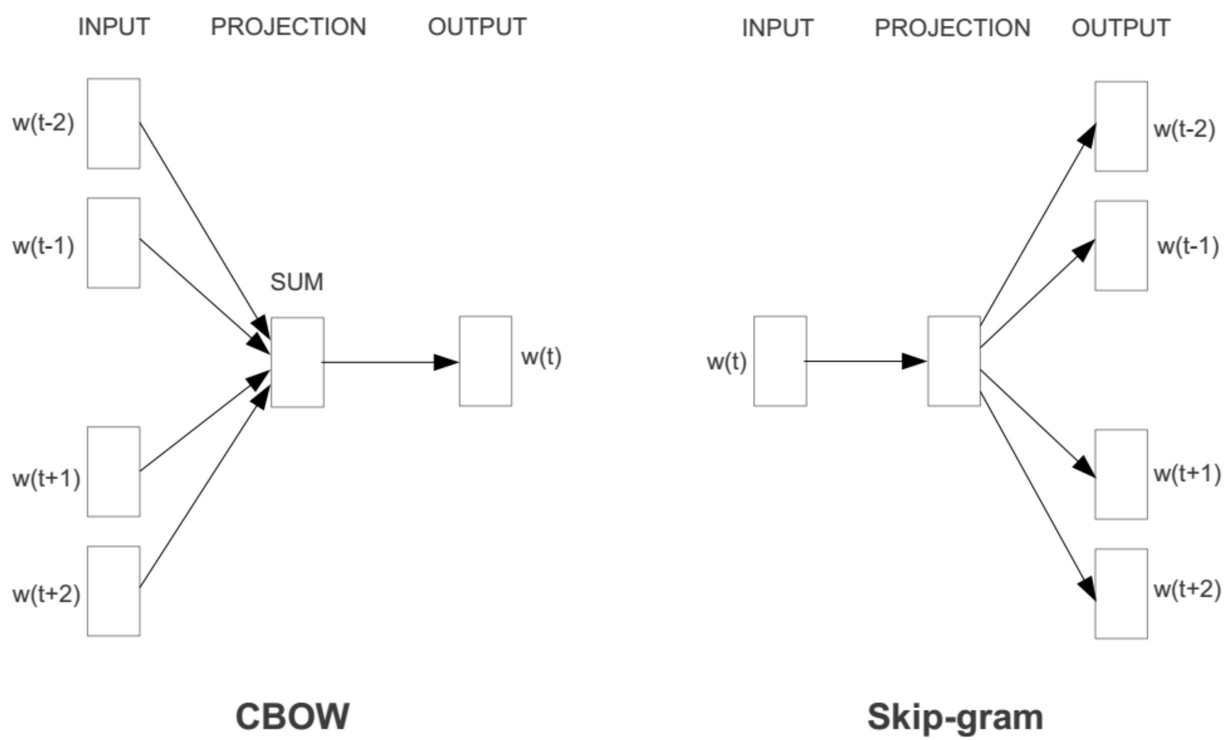
- This is usually called distributed representations in the context of deep learning
  - Vector representation does not represent a distribution, but distributed over the space
  - Term widely used in connectionism (Learning distributed representations of concepts, Hinton (1986))
    - "In the componential approach each concept is simply a set of features and so a neural net can be made to implement a set of concepts by assigning a unit to each feature and setting the strengths of the connections between units so that each concept corresponds to a stable pattern of activity distributed over the whole network."
- Compared to distributional semantics
  - The **distributional hypothesis** in linguistics is derived from the semantic theory of language usage, i.e. words that are used and occur in the same contexts tend to purport similar meanings.

Main Idea of word2vec

- Instead of capturing co-occurrence counts directly,
- Predict surrounding words of every word
- Faster and can easily incorporate a new sentence/document or add a word to the vocabulary

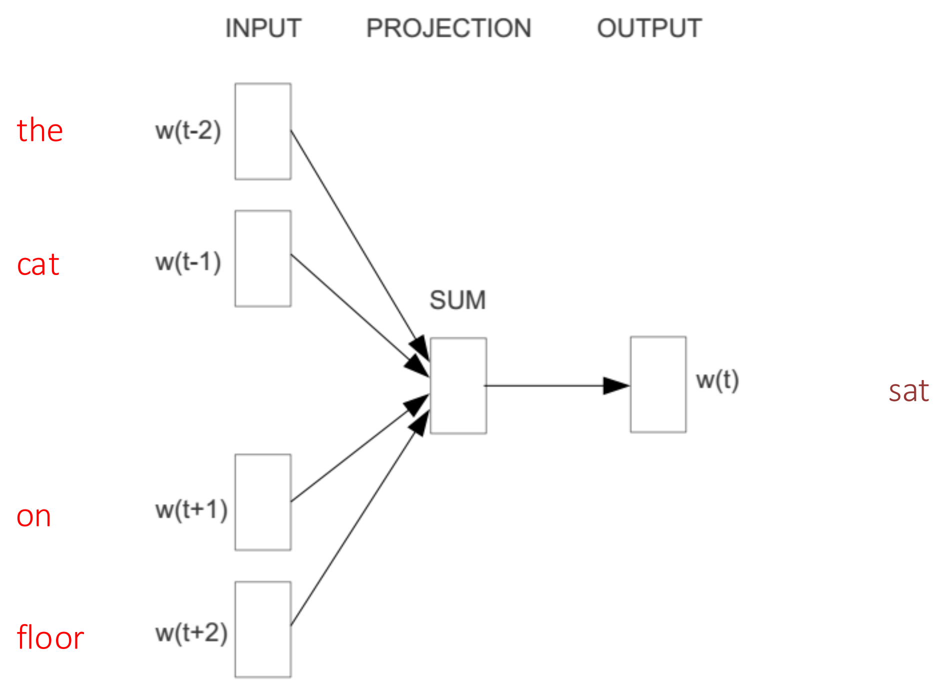
Represent the meaning of word – word2vec

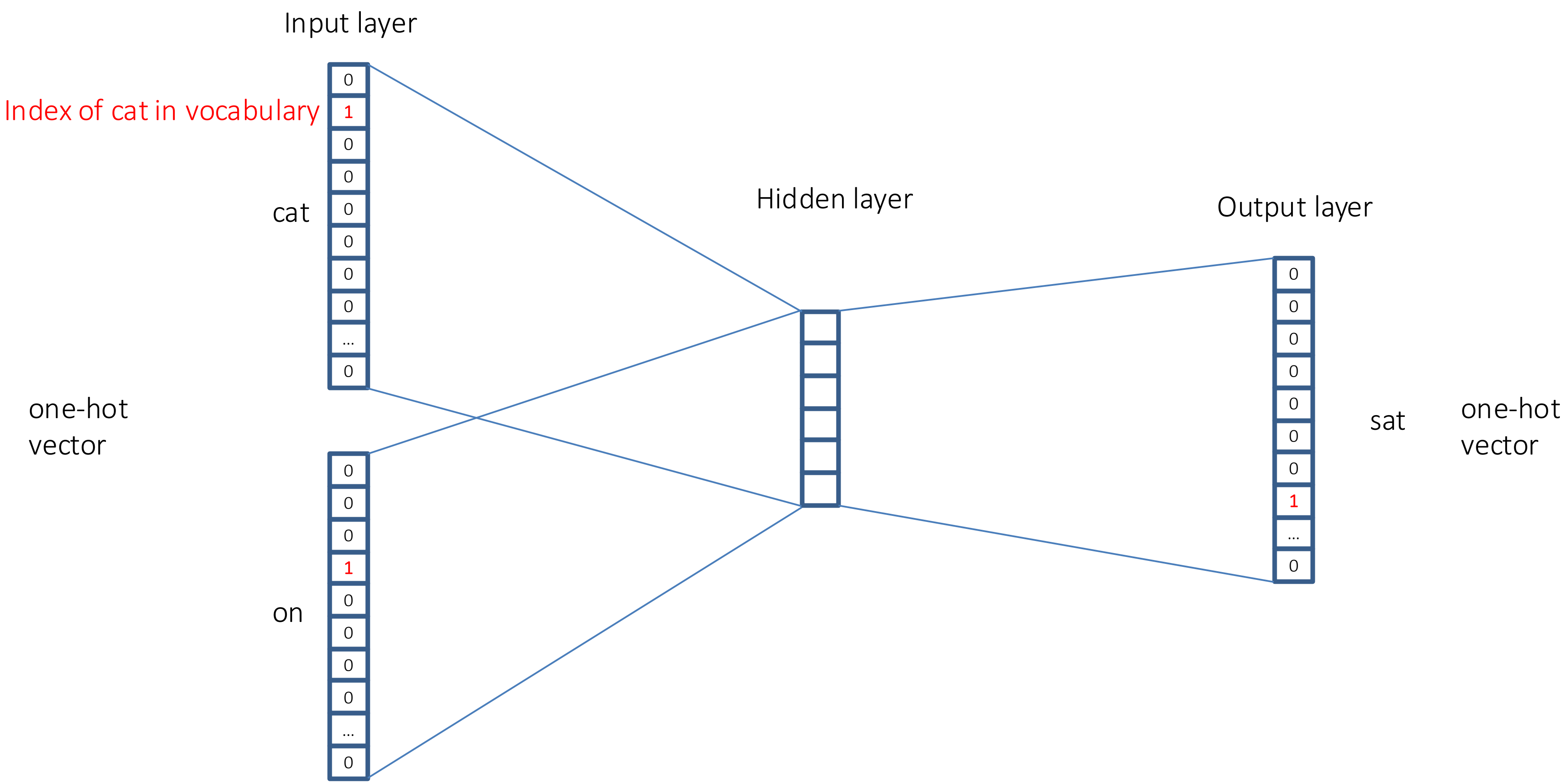
- 2 basic neural network models:
  - Continuous Bag of Word (CBOW): use a window of word to predict the middle word
  - Skip-gram (SG): use a word to predict the surrounding ones in window.

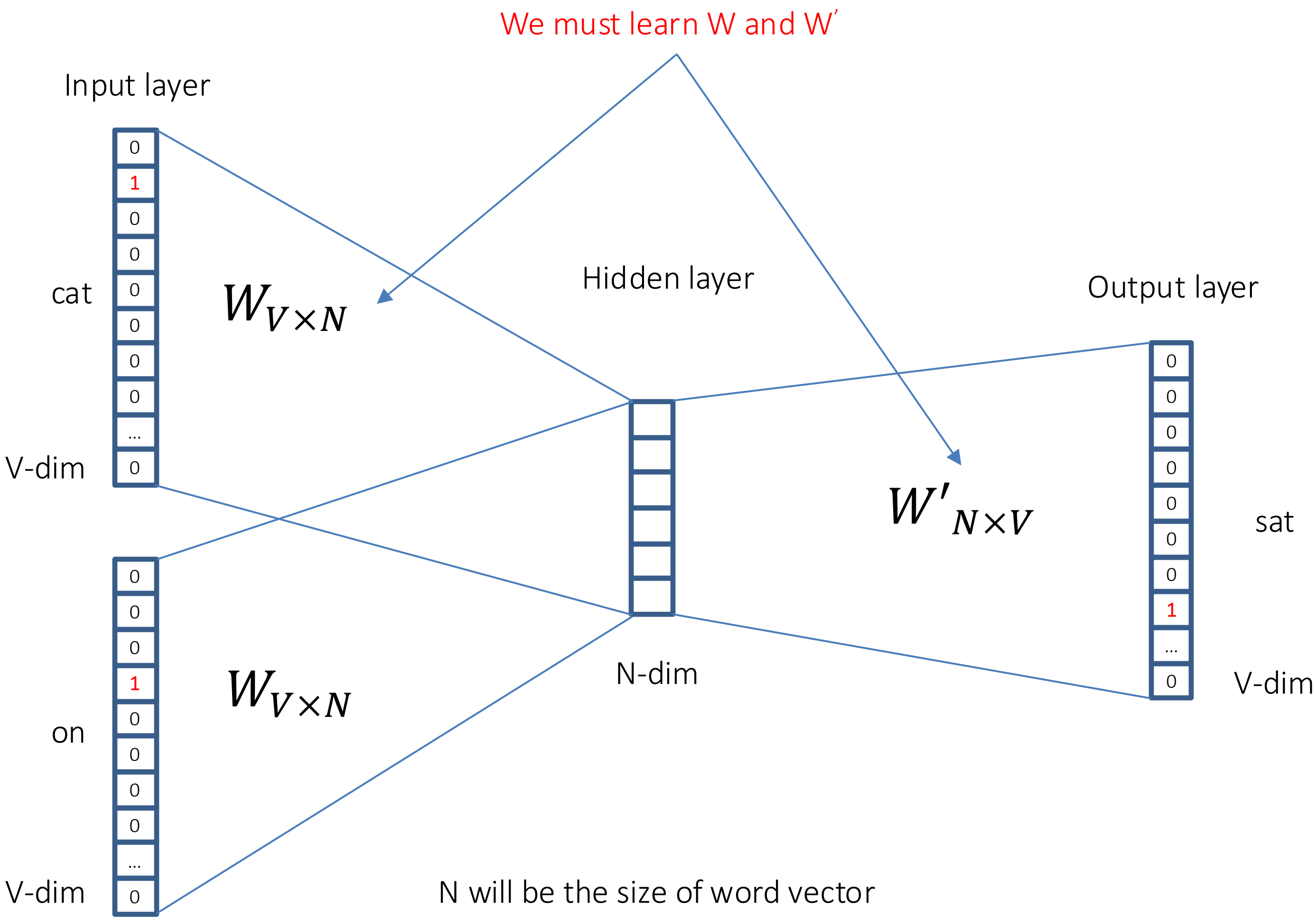


Word2vec – Continuous Bag of Word

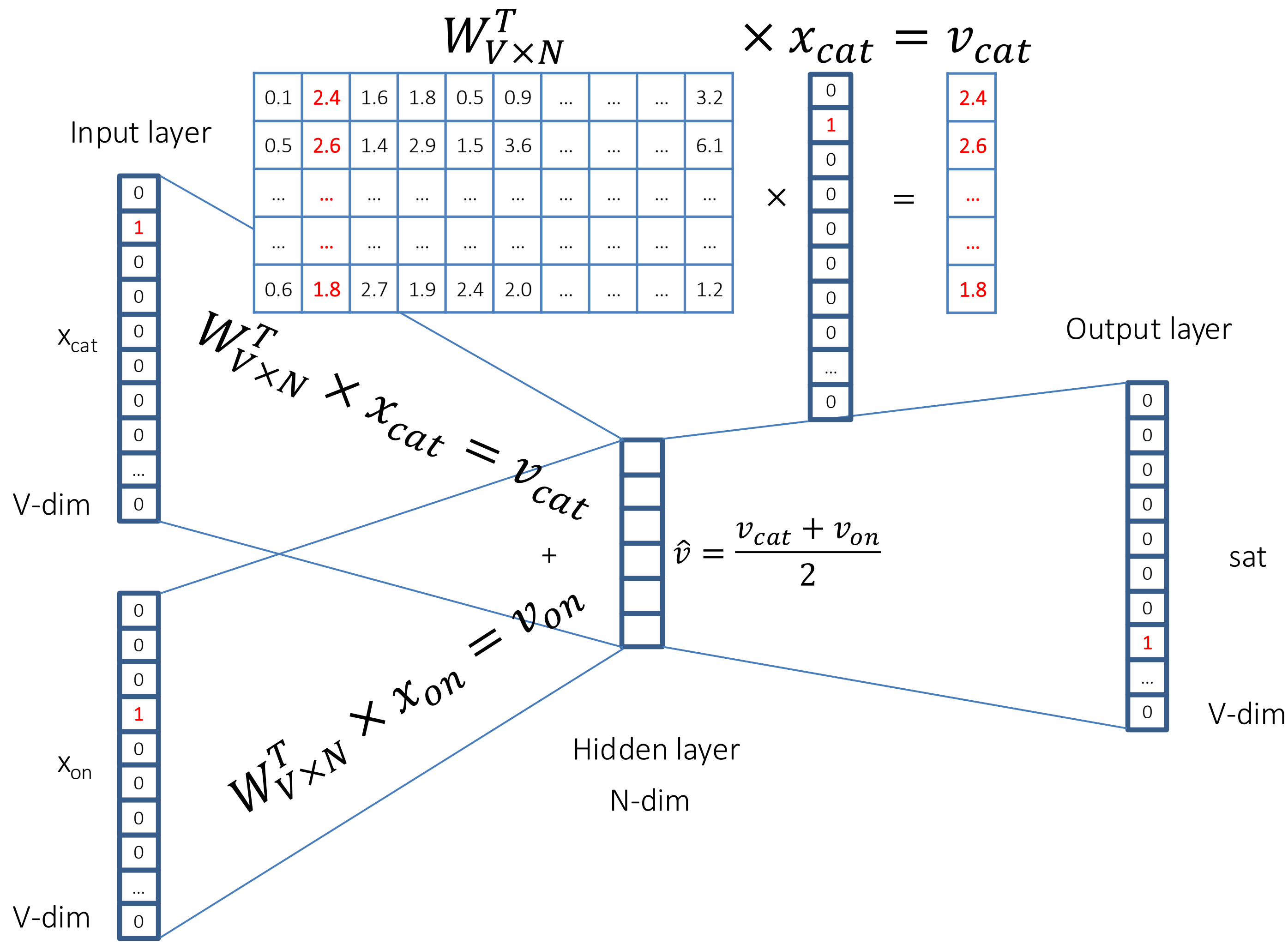
- E.g. “The cat sat on floor”
  - Window size = 2



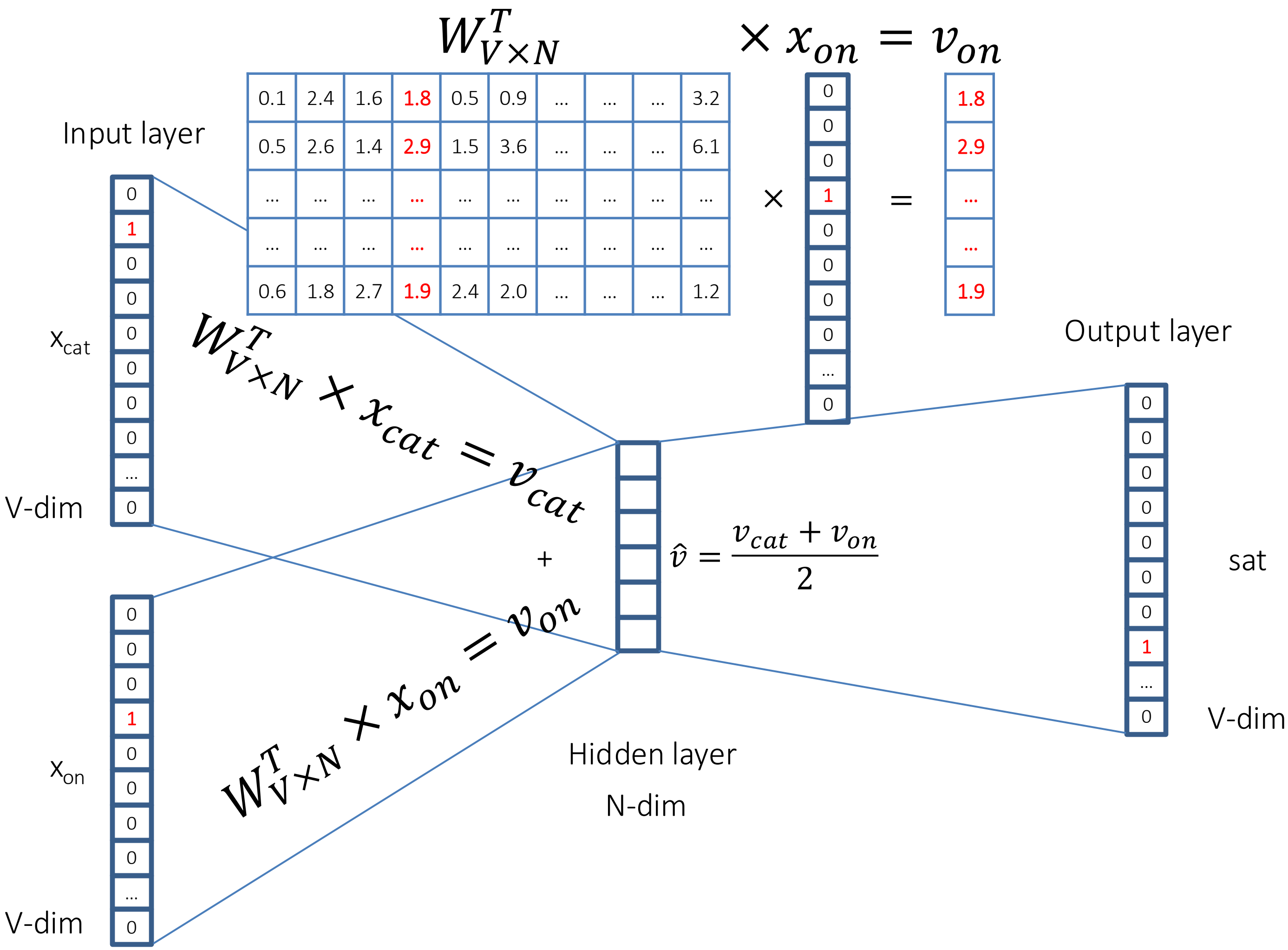

Input layer
Index of cat in vocabulary
cat
one-hot
vector
on
Hidden layer
Output layer
sat
one-hot
vector
0
1
...

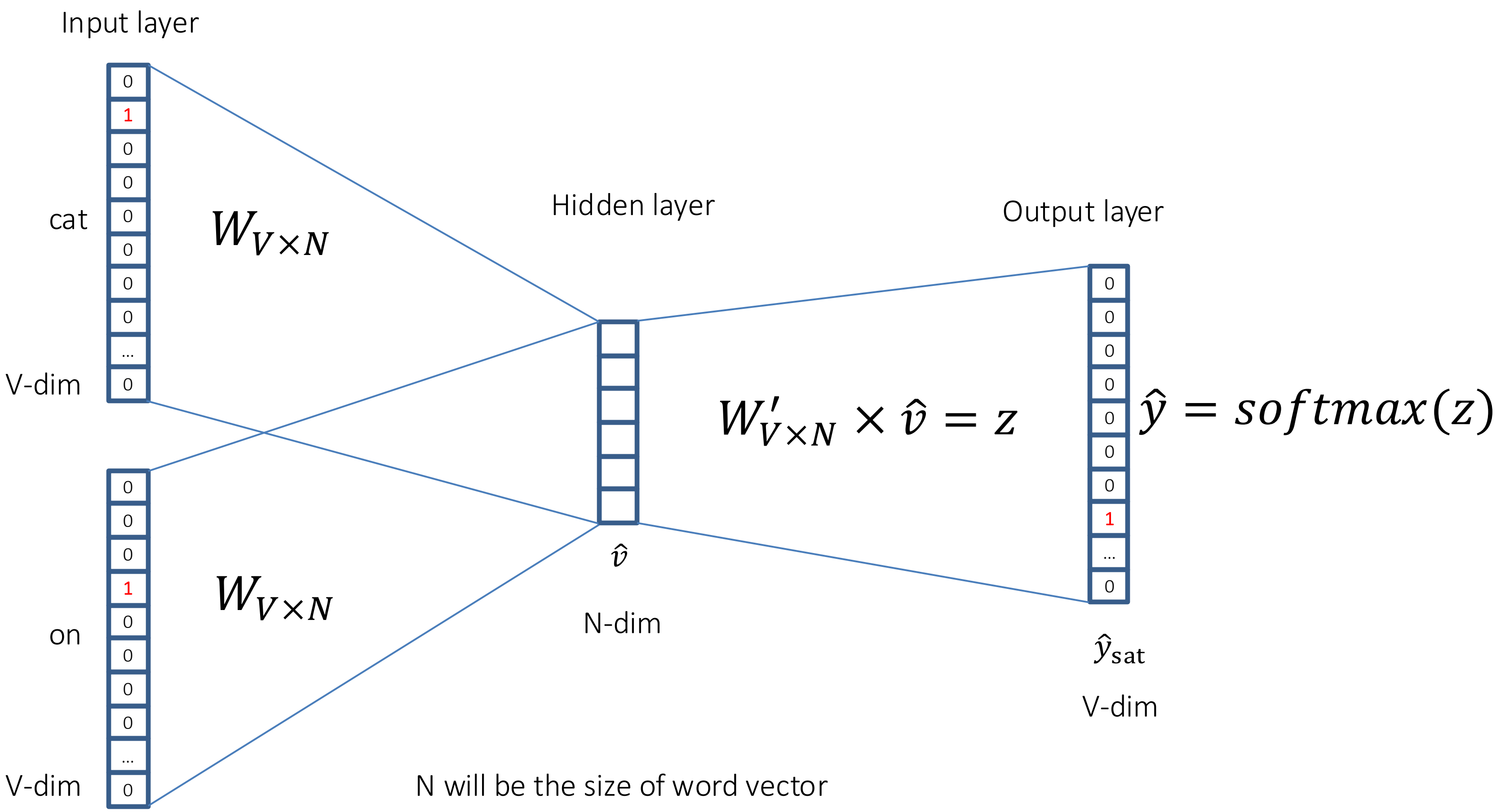


N will be the size of word vector

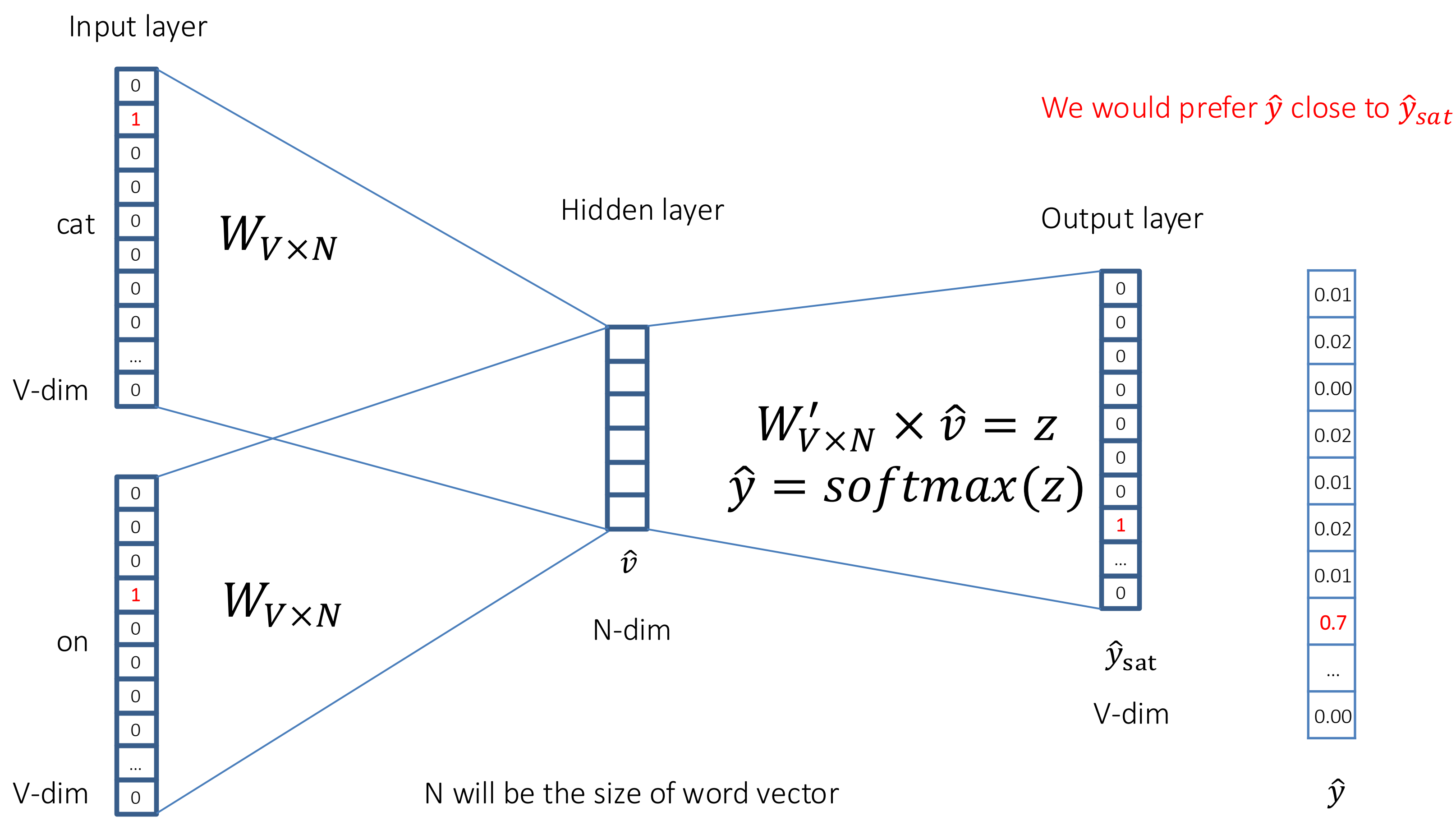


$W^T_{V\times N} \times x_{cat} = v_{cat}$
0.1 2.4 1.6 1.8 0.5 0.9 ... ... 3.2
0.5 2.6 1.4 2.9 1.5 3.6 ... ... 6.1
0.6 1.8 2.7 1.9 2.4 2.0 ... ... 1.2
Input layer
x_{cat}
V-dim
x_{on}
V-dim
$W^T_{V\times N} \times x_{cat} = v_{cat}$
+
$W^T_{V\times N} \times x_{on} = v_{on}$
$\hat{v} = \frac{v_{cat} + v_{on}}{2}$
Hidden layer
N-dim
Output layer
sat
V-dim

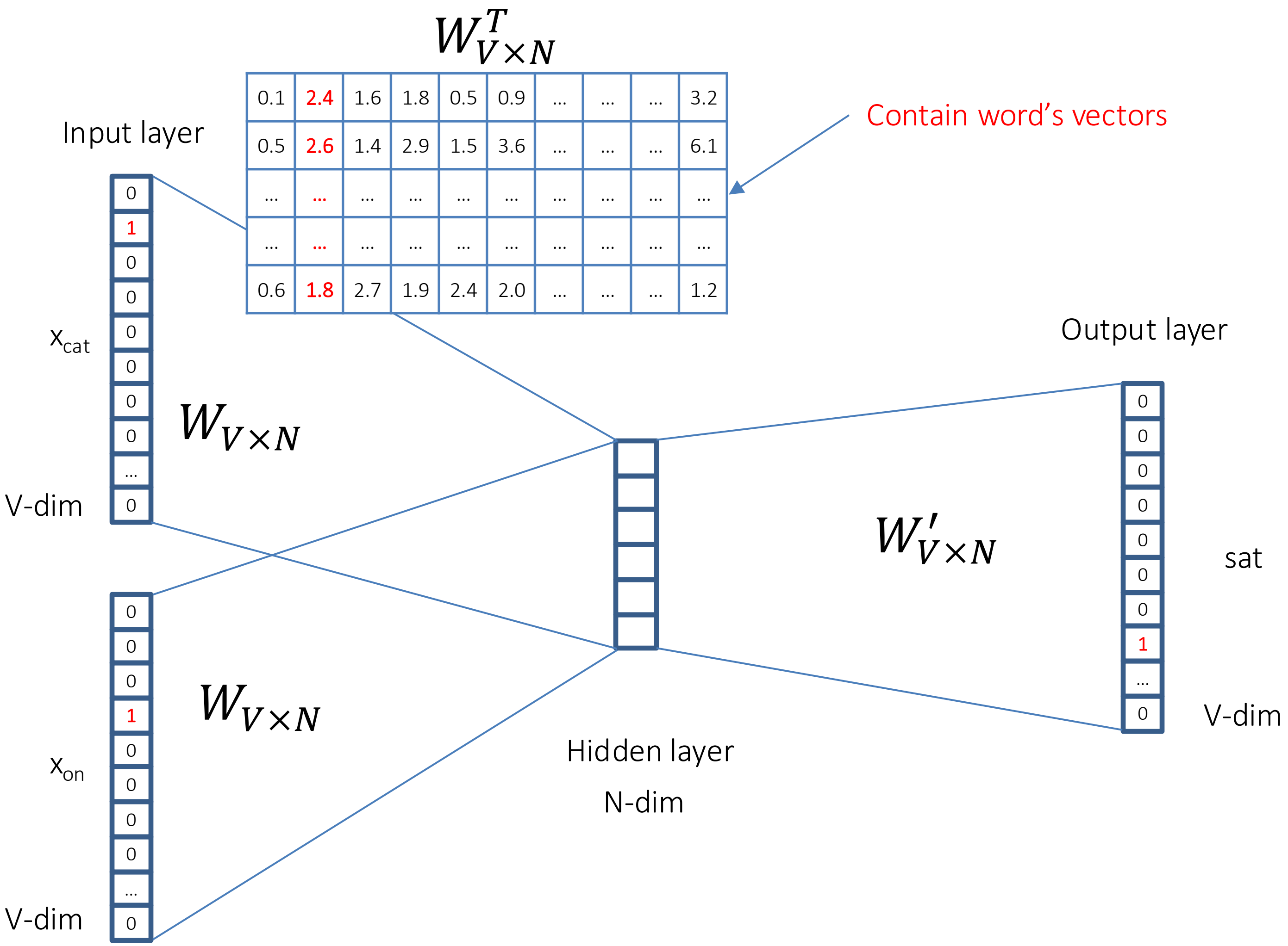

$W_{V \times N}^T$
$\times x_{on} = v_{on}$
Input layer
Output layer
Hidden layer
N-dim
x_{cat}
x_{on}
V-dim
V-dim
V-dim
sat
$W_{V \times N}^T \times x_{cat} = v_{cat}$
+
$W_{V \times N}^T \times x_{on} = v_{on}$
$\hat{v} = \frac{v_{cat} + v_{on}}{2}$
0.1 2.4 1.6 1.8 0.5 0.9 … … … 3.2
0.5 2.6 1.4 2.9 1.5 3.6 … … … 6.1
0.6 1.8 2.7 1.9 2.4 2.0 … … … 1.2
1.8
2.9
1.9



N will be the size of word vector



Input layer
cat
V-dim
on
V-dim
$W_{V\times N}$
$W_{V\times N}$
Hidden layer
$\hat{v}$
N-dim
$W'_{V\times N} \times \hat{v} = z$
$\hat{y} = softmax(z)$
Output layer
$\hat{y}_{sat}$
V-dim
We would prefer $\hat{y}$ close to $\hat{y}_{sat}$
0.01
0.02
0.00
0.02
0.01
0.02
0.01
0.7
...
0.00
$\hat{y}$
N will be the size of word vector



We can consider either W or W' as the word's representation. Or even take the average.

Approximations

- With large vocabularies this objective function is not scalable and would train too slowly! → Why?
- Idea: approximate the normalization or
- Define negative prediction that only samples a few words that do not appear in the context
- Similar to focusing on mostly positive correlations
- More reading
  - https://canvas.ust.hk/courses/16504/files/1444107?module_item_id=214783

Linear Relationships in word2vec

- These representations are *very good* at encoding dimensions of similarity!
- Analogies testing dimensions of similarity can be solved quite well just by doing vector subtraction in the embedding space
  - Syntactically
    - *apple − apples ≈ car − cars ≈ family − families*
    - Similarly for verb and adjective morphological forms
  - Semantically (Semeval 2012 task 2)
    - *shirt − clothing ≈ chair − furniture*
    - *king − man ≈ queen − woman*

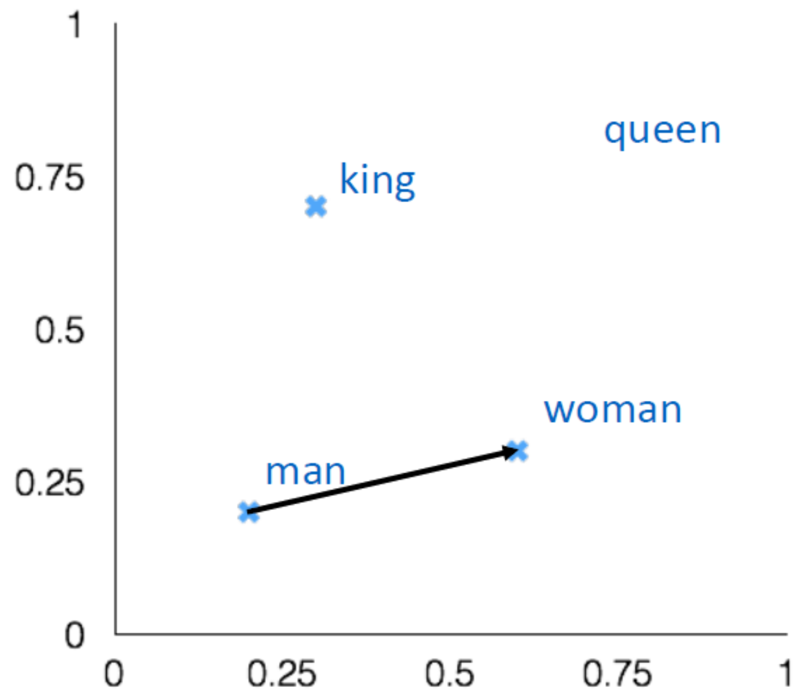
Word Analogies

- Test for linear relationships, examined by Mikolov et al. (2014)

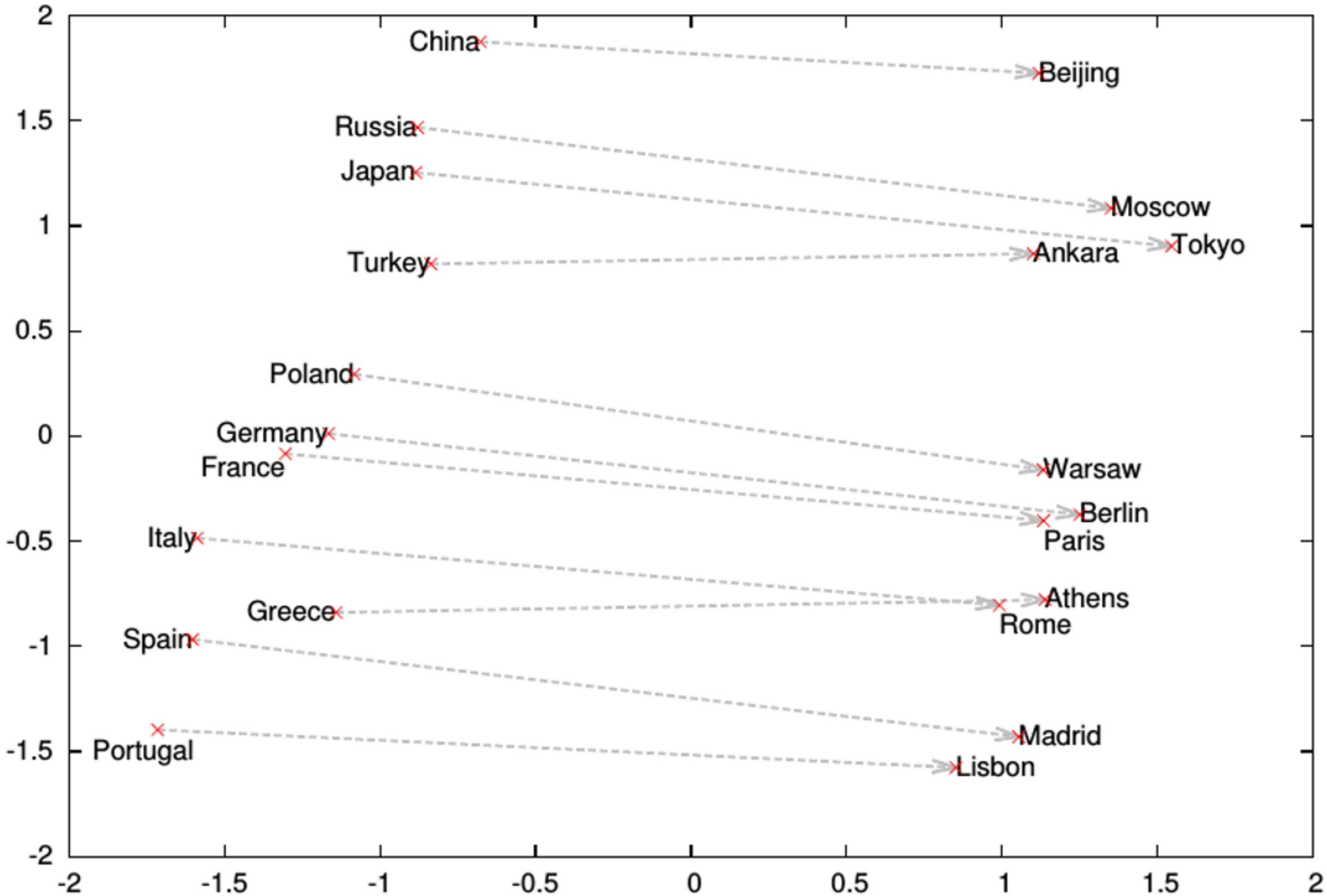
a:b :: c:? $\longrightarrow$ $$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

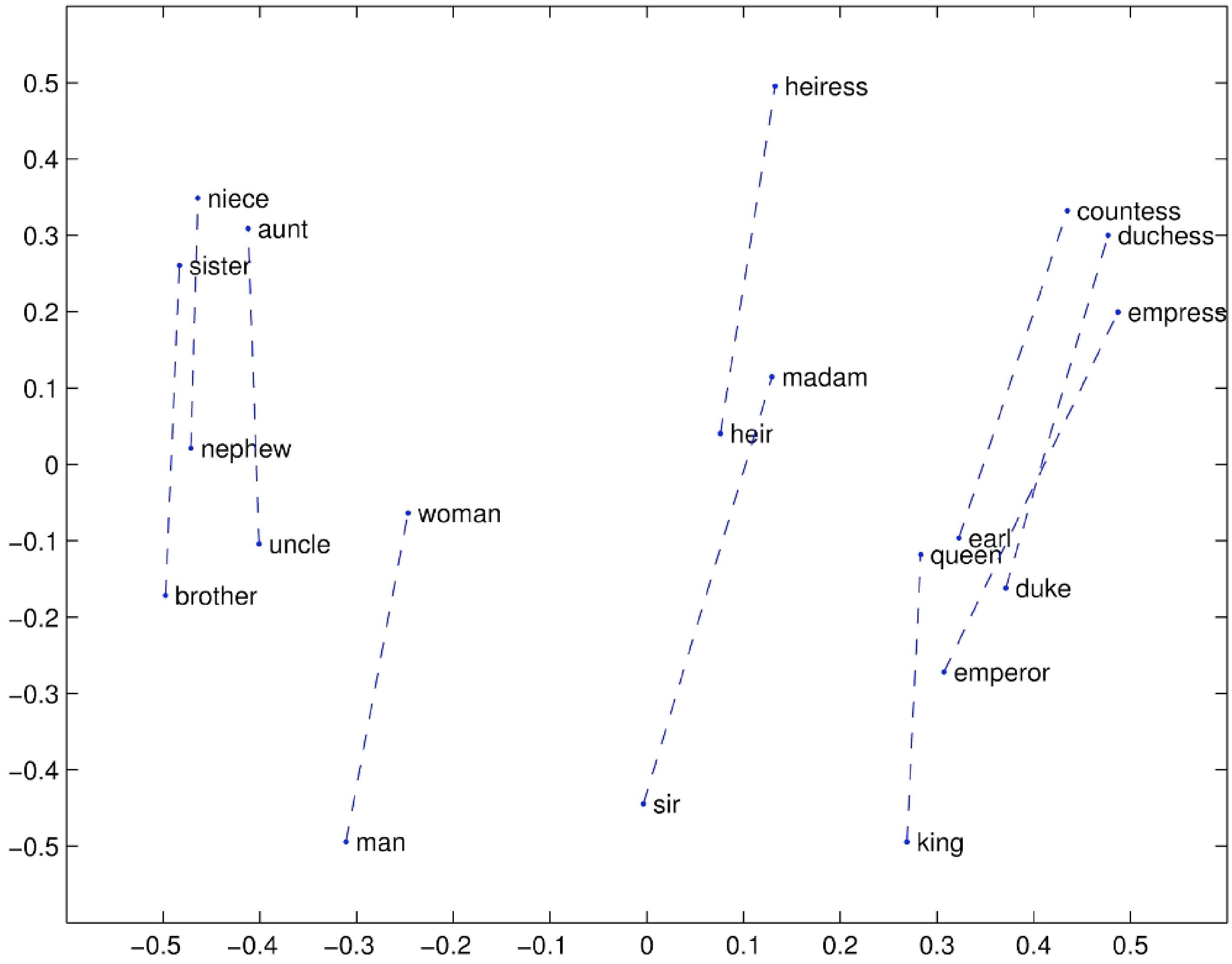
+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



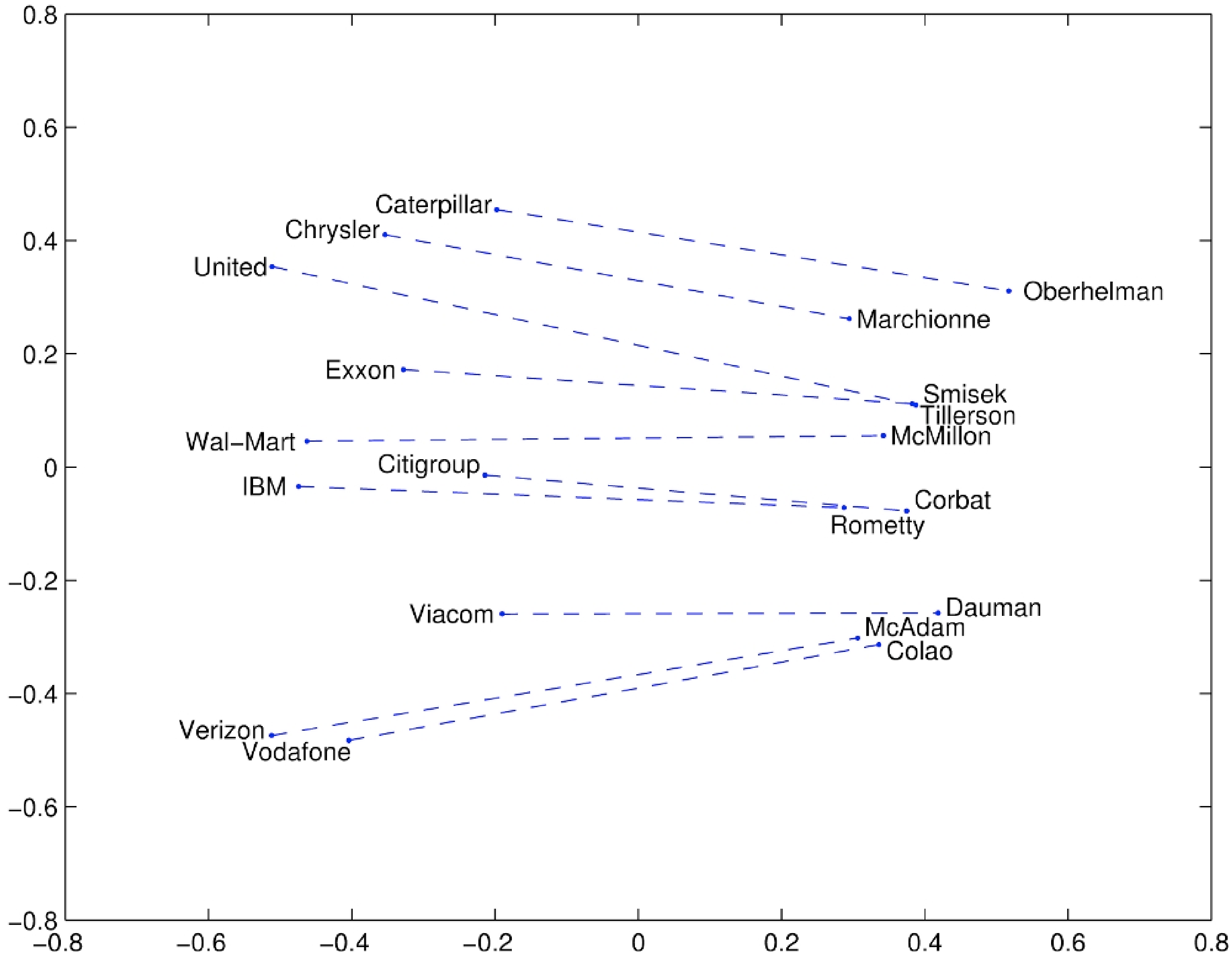
Word analogies



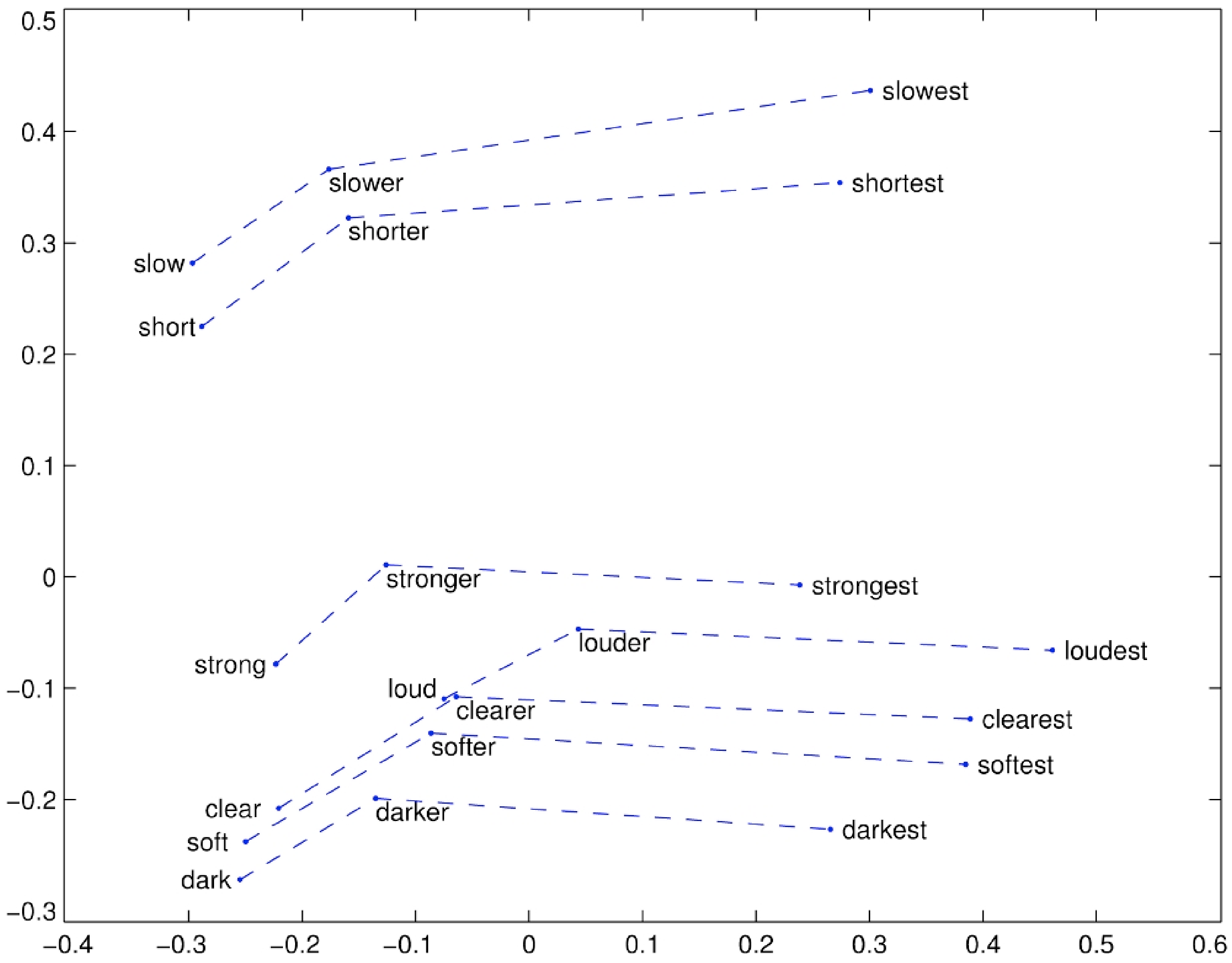
Glove Visualizations



Glove Visualizations: Company - CEO



Glove Visualizations: Superlatives



More Examples

- “word2vec Parameter Learning Explained”, Xin Rong
  - https://ronxin.github.io/wevi/
- Word2Vec Tutorial - The Skip-Gram Model
  - http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/