2. From symbolic to distributed word representations

The vast majority of (rule-based and statistical) natural language processing and information retrieval (NLP/IR) work regarded words as atomic symbols: hotel, conference

In machine learning vector space terms, this is a vector with one 1 and a lot of zeroes

[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

Deep learning people call this a "one-hot" representation

It is a **localist** representation

From symbolic to distributed word representations

Its problem, e.g., for web search:

- If user searches for [Dell notebook battery size], we would like to match documents with “Dell laptop battery capacity”

But

$$
\begin{array}{ll}
\text{size} & [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^T \\
\text{capacity} & [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0] = 0
\end{array}
$$

Our query and document vectors are **orthogonal**

There is no natural notion of similarity in a set of one-hot vectors

Capturing similarity

There are many things you can do to capture similarity:

- Query expansion with synonym dictionaries
- Separately learning word similarities from large corpora

But a word representation that encodes similarity wins:

- Less parameters to learn (per word, not per pair)
- More sharing of statistics
- More opportunities for multi-task learning

A solution via distributional similarity-based representations

You can get a lot of value by representing a word by means of its neighbors

"You shall know a word by the company it keeps"

(J. R. Firth 1957: 11)

One of the most successful ideas of modern NLP

government debt problems turning into	banking	crises as has happened in
saying that Europe needs unified	banking	regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Basic idea of learning neural network word embeddings (Predict!)

We define a model that predicts between a center word w_t and context words in terms of word vectors, e.g.,

$$p(\text{context}|w_t) = \ldots$$

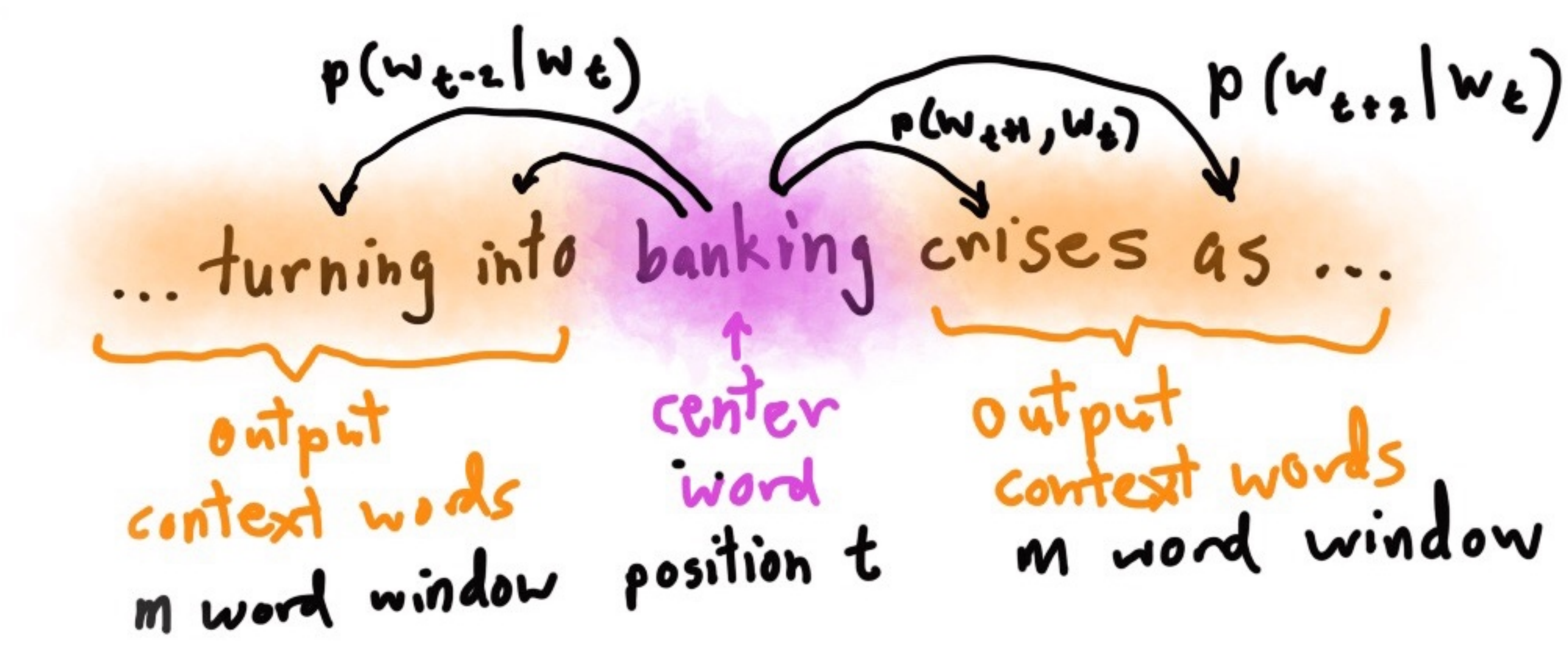
which has a loss function, e.g.,

$$J = 1 - p(w_{-t}|w_t)$$

We look at **many** positions *t* in a big language corpus

We keep adjusting the vector representations of words to minimize this loss

Word2vec skip-gram prediction



Details of Word2Vec

For $p(w_{t+j}|w_t)$ we choose:

$$p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{V} \exp(u_w^T v_c)}$$

where *o* is the outside (or output) word index, *c* is the center word index, v_c and u_o are the "center" and "outside" vectors for word indices *c* and *o*

Softmax using word *c* to obtain probability of word *o*

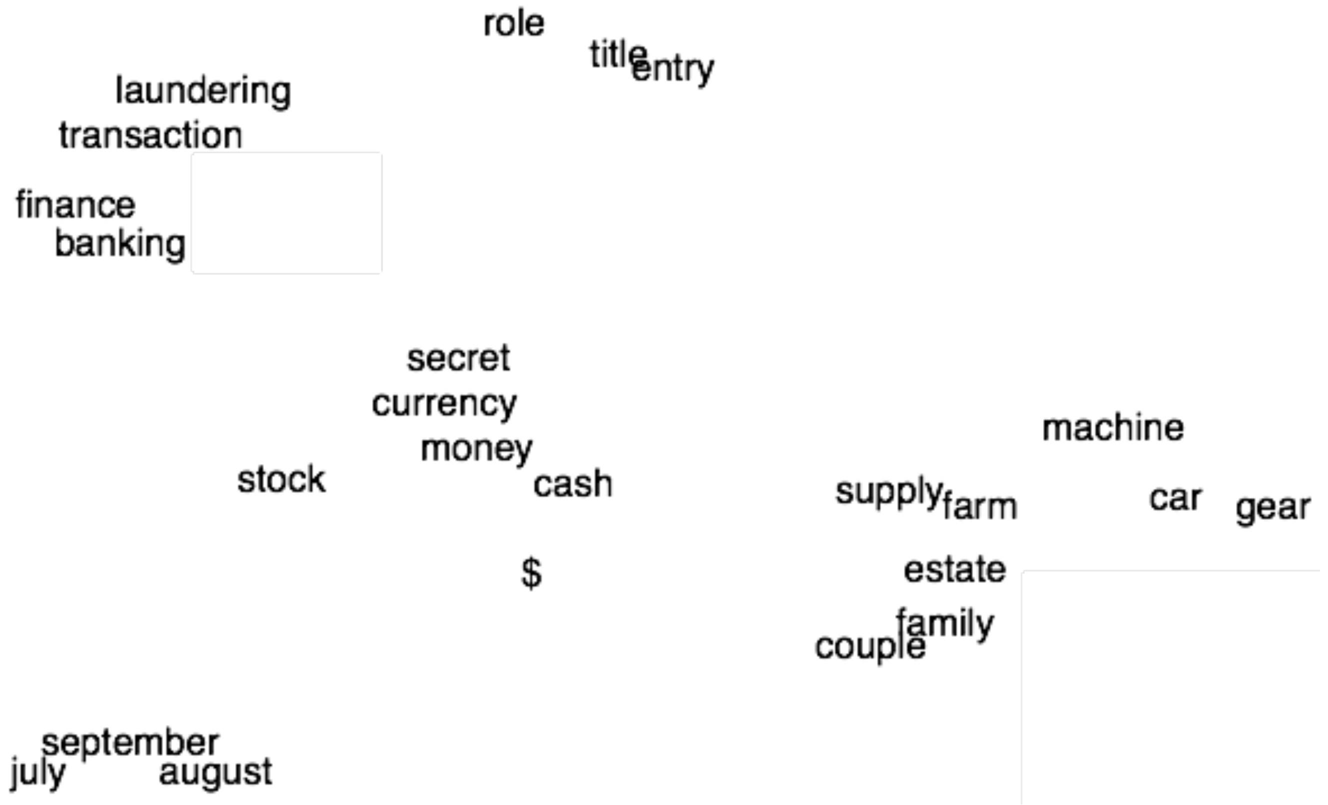
Co-occurring words are driven to have similar vectors

Word meaning as a vector

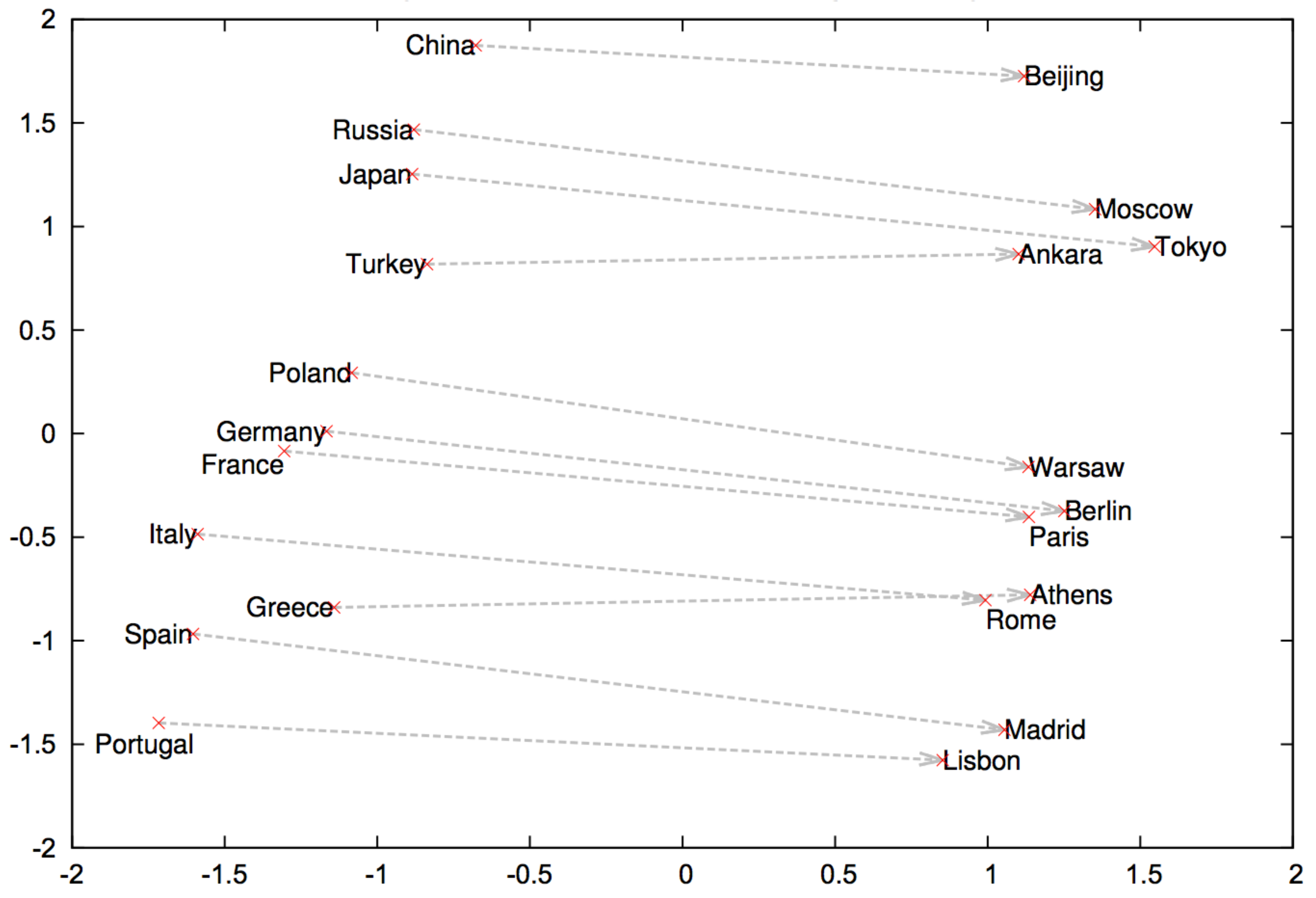
The result is a dense vector for each word type, chosen so that it is good at predicting other words appearing in its context

… those other words also being represented by vectors

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



word2vec encodes semantic components as linear vector differences



Encoding meaning in vector differences

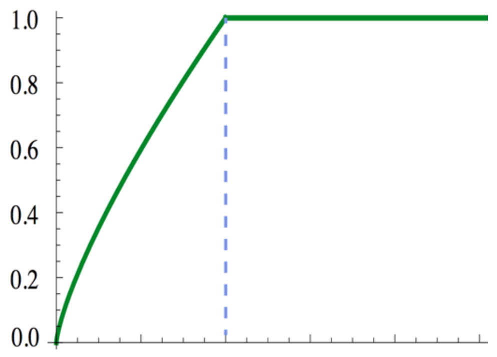
Q: How can we capture ratios of co-occurrence probabilities as meaning components in a word vector space?

A: Log-bilinear model: $w_i \cdot w_j = \log P(i|j)$

with vector differences $w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

$f \sim$



Glove Word similarities

[Pennington et al., EMNLP 2014]

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

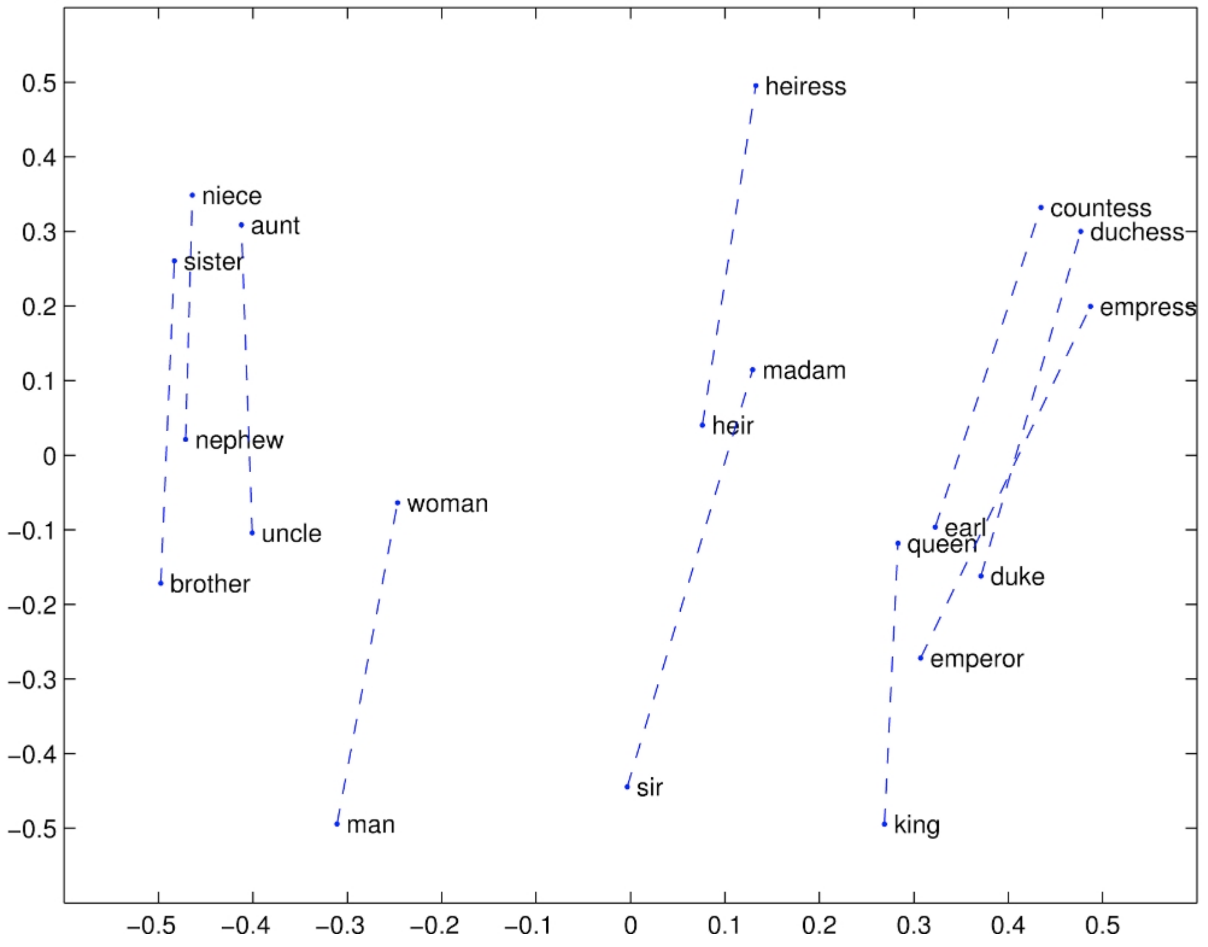
leptodactylidae

rana

eleutherodactylus

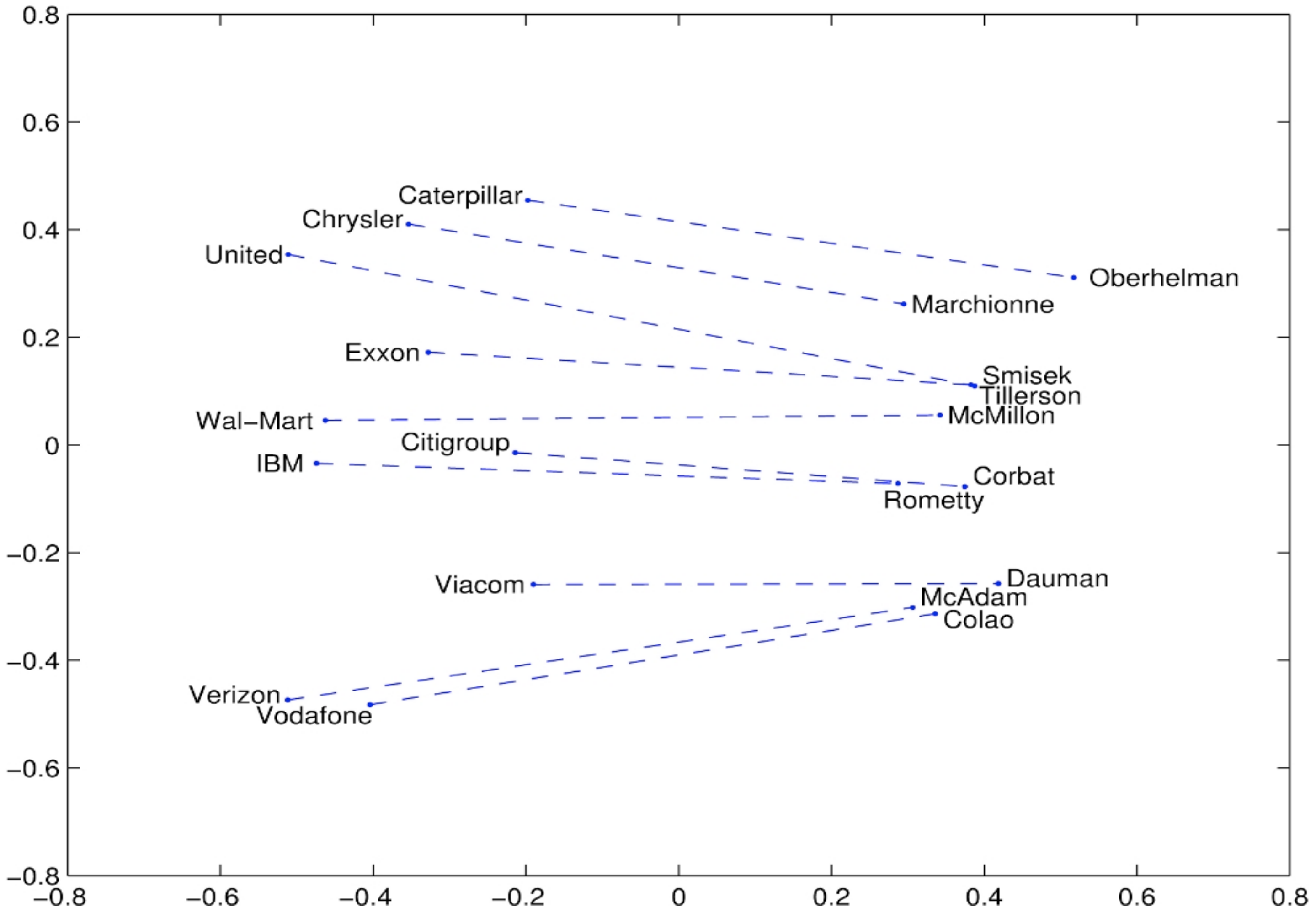
http://nlp.stanford.edu/projects/glove/

Glove Visualizations: Gender pairs



http://nlp.stanford.edu/projects/glove/

Glove Visualizations: Company - CEO



Named Entity Recognition Performance

(finding person, organization names in text)

Model on CoNLL	**CoNLL ’03 dev**	**CoNLL ’03 test**
Categorical CRF	91.0	85.4
SVD (log tf)	90.5	84.8
HPCA	92.6	**88.7**
C&W	92.2	87.4
CBOW	93.1	88.2
GloVe	**93.2**	88.3

F1 score of CRF trained on CoNLL 2003 English with 50 dim word vectors

Named Entity Recognition Performance

(finding person, organization names in text)

Model on CoNLL	**CoNLL '03 dev**	**CoNLL '03 test**	**ACE 2**	**MUC 7**
Categorical CRF	91.0	85.4	77.4	73.4
SVD (log tf)	90.5	84.8	73.6	71.5
HPCA	92.6	**88.7**	81.7	80.7
C&W	92.2	87.4	81.7	80.2
CBOW	93.1	88.2	82.2	81.1
GloVe	**93.2**	88.3	**82.9**	**82.2**

F1 score of CRF trained on CoNLL 2003 English with 50 dim word vectors