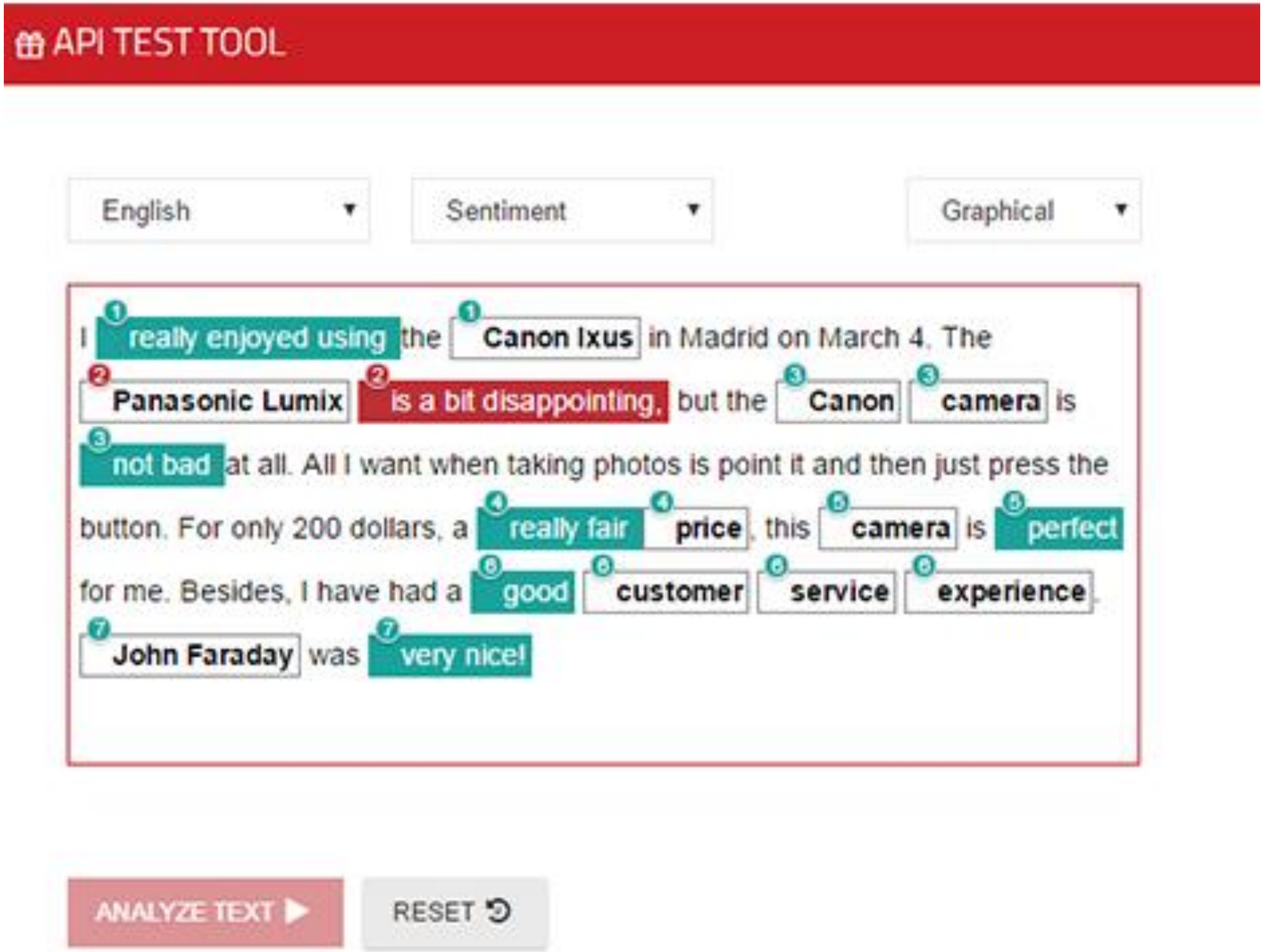
\`king\`+\`woman\`-\`man\`≈\`queen\`? Word Vector in Natural Language Processing

Zhenghui Wang

Shanghai Jiao Tong University

What is NLP?

Sentiment Analysis

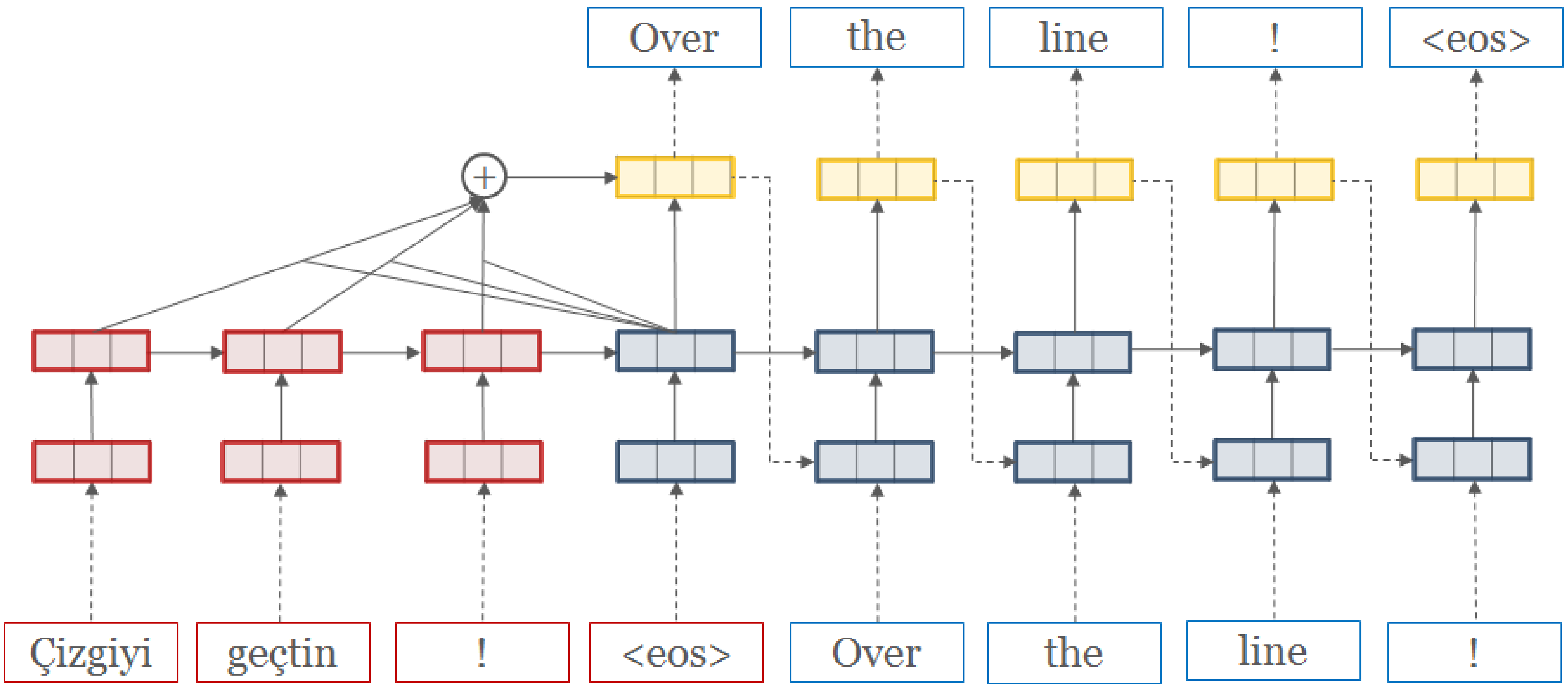


www.bitext.com

Neural Machine Translation

translate.google.cn

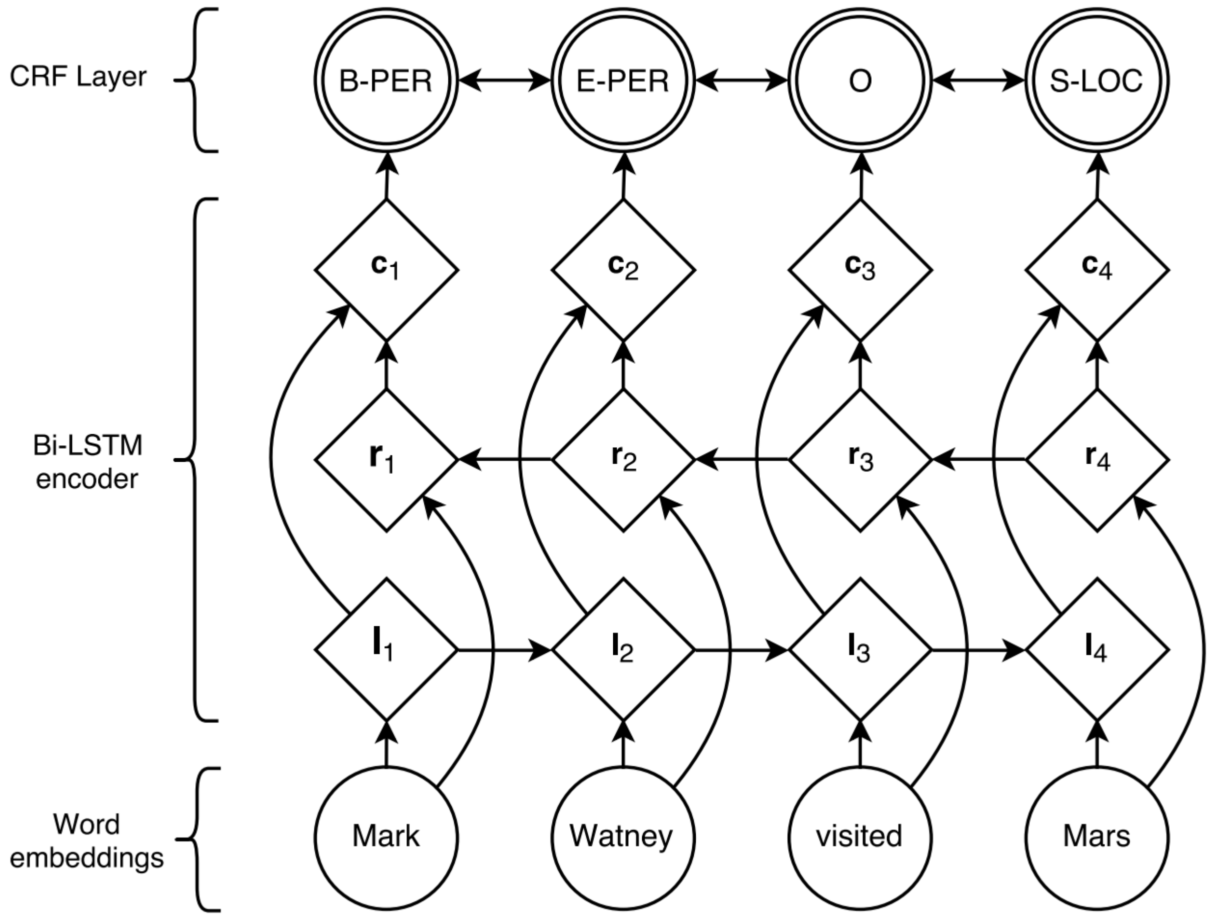
Neural Machine Translation



opennmt.net

Named Entity Recognition

Named Entity Recognition



[Lample G, Ballesteros M, Subramanian S, et al. Neural architectures for named entity recognition[J]. arXiv preprint arXiv:1603.01360, 2016.]

What is word vector?

- A way to represent the meaning of words by fixed-size vectors

"One-hot" Representation

motel $[0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]$

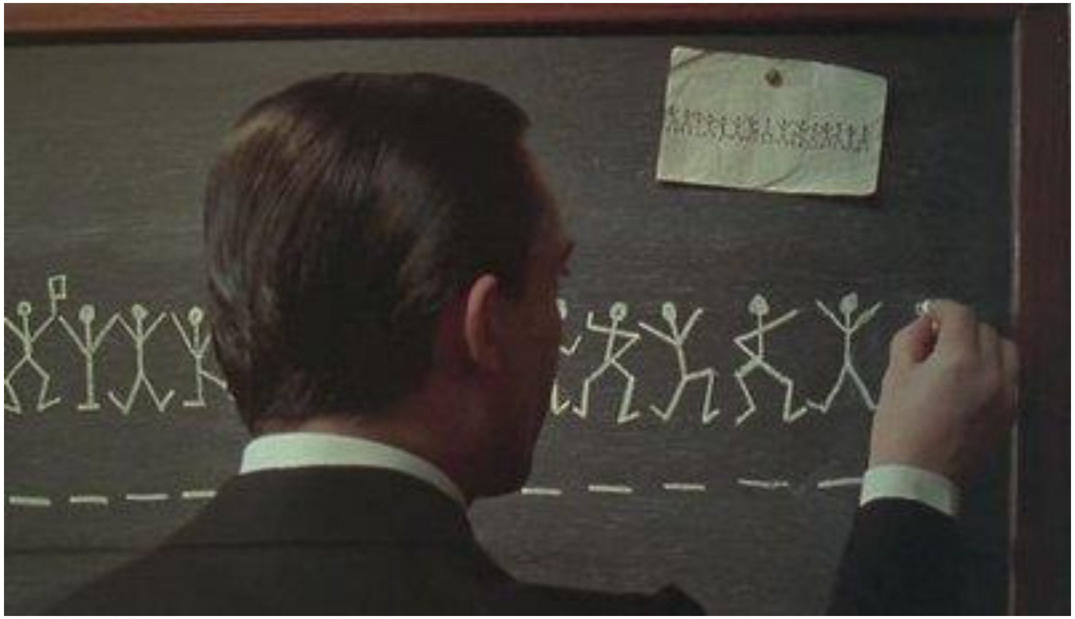
- No natural notion of similarity:

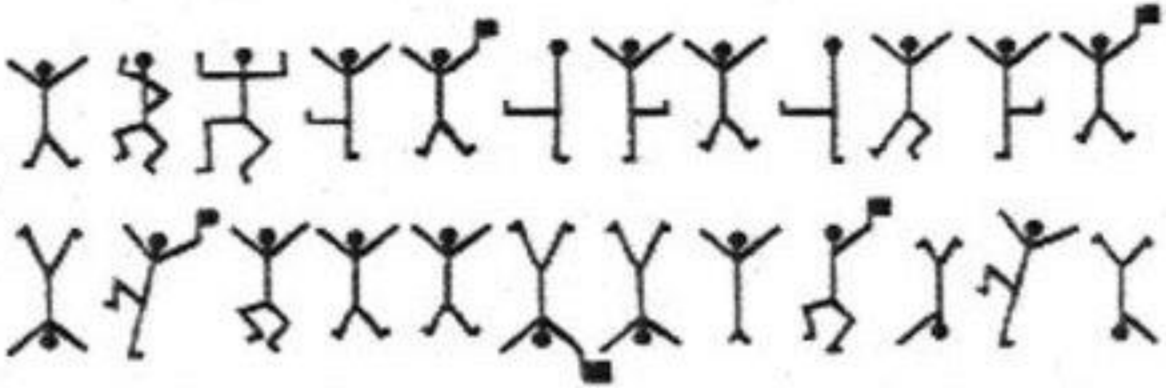
motel $[0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^T$
hotel $[0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0] = 0$

- Scalability: what if we want to add some new words?

How does computer understand natural language?

Holmes: Dancing Man





Main Idea of word2vec

Predict between every word and its context words!

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

(J. R. Firth 1957: 11)

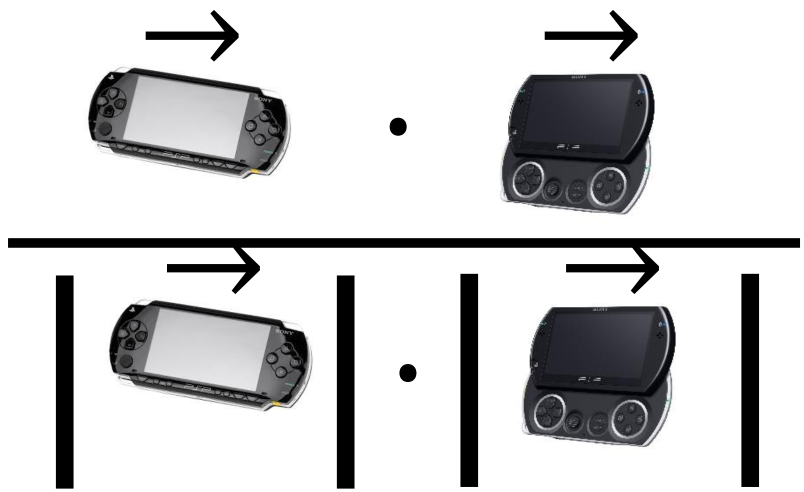
“I like playing X.”

“The X had an attractive library of games I wanted.”

"Many good PS1 games are available on either PS3 or X."

“The Sony X was blessed with some of the biggest names in games.”

$\mathbf{X}$ =  or  ?

 ≈ 0.9

Word2Vec

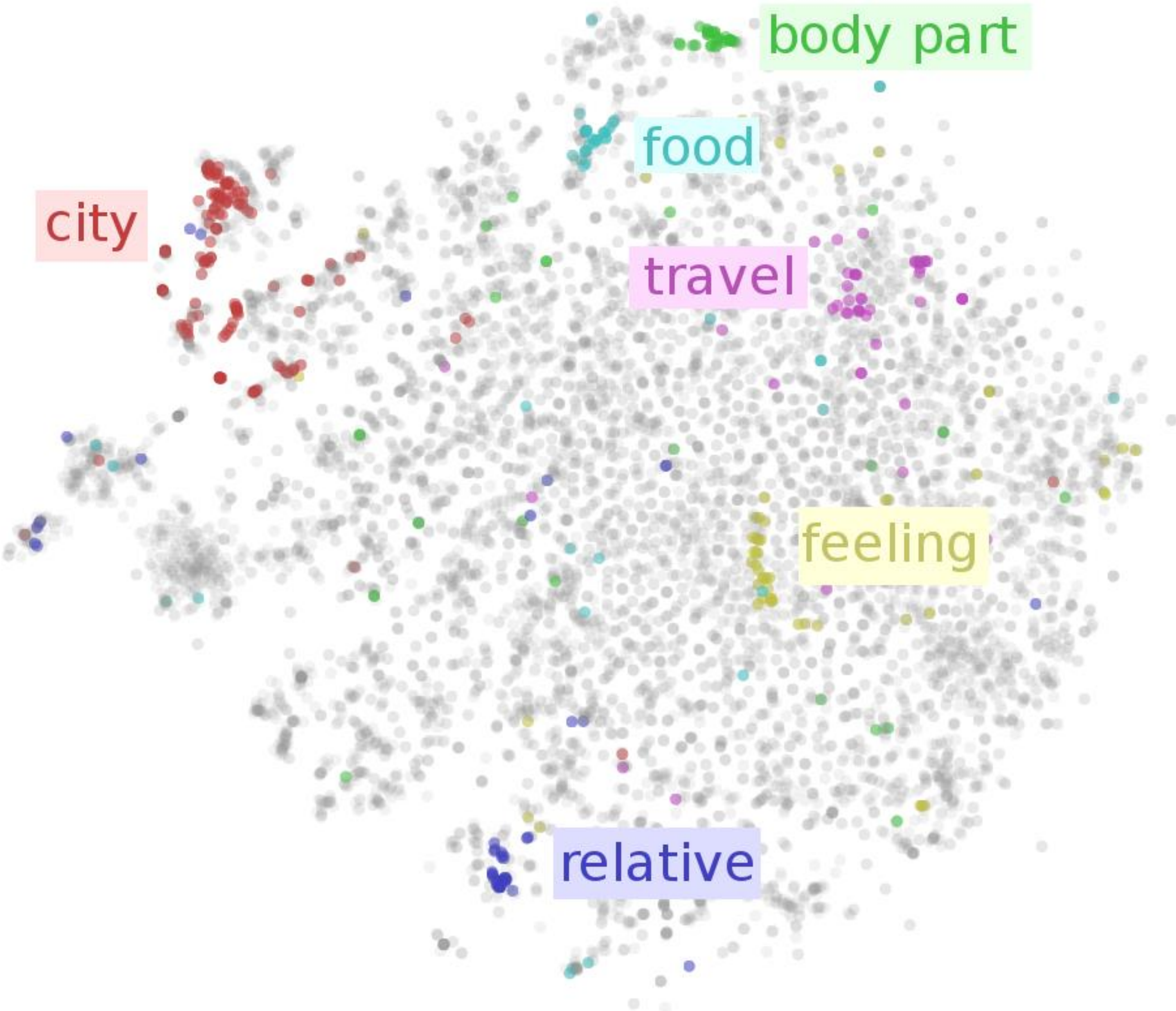
- Distributional similarity based representations

Shanghai = [0.286, 0.792, ⋯ , -0.107, 0.109]

Beijing = [0.178, 0.490, ⋯, - 0.287, 0.201]

…

Washington =[0.334, -0.321, ⋯, -0.233, 0.391]



http://ruder.io/content/images/2016/04/word_embeddings_colah.png

How Do We Calculate?

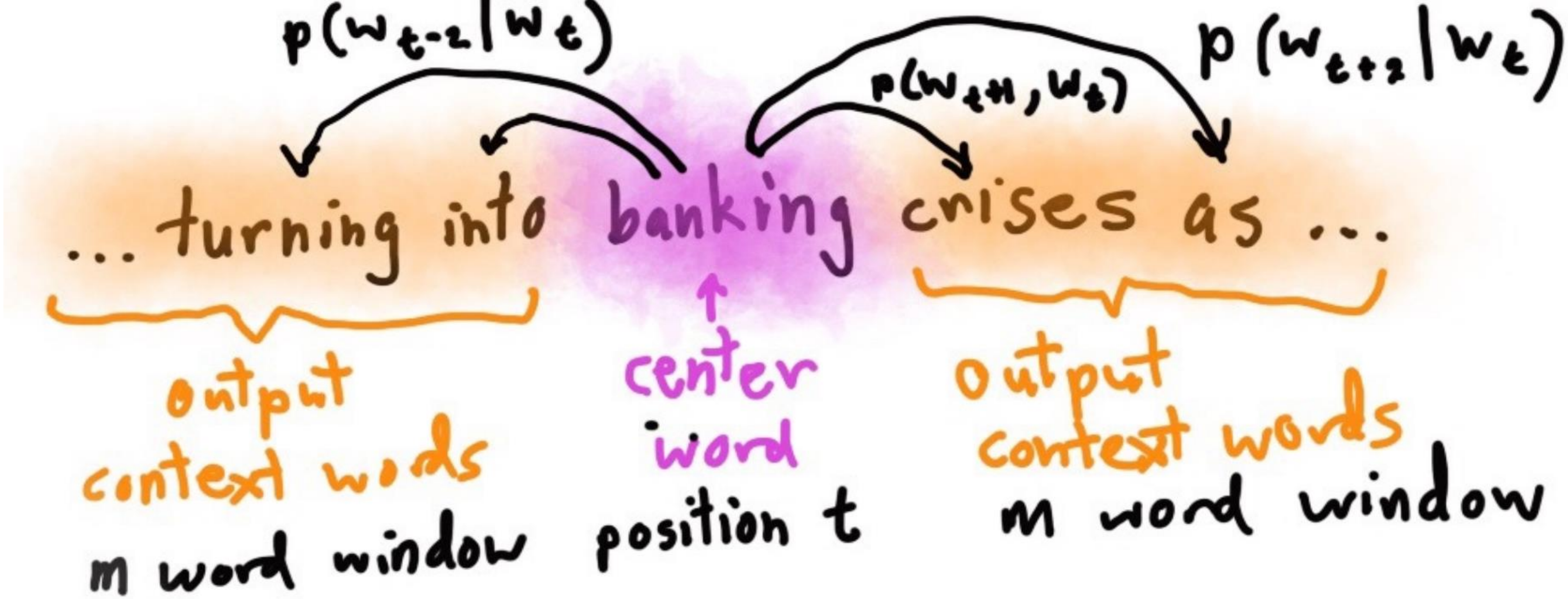
- Two algorithms for word2vec
  - 1. Skip-grams (SG):

    Predict context words given target (position independent)
  - 2. Continuous Bag of Words (CBOW) :

    Predict target word from bag-of-words context

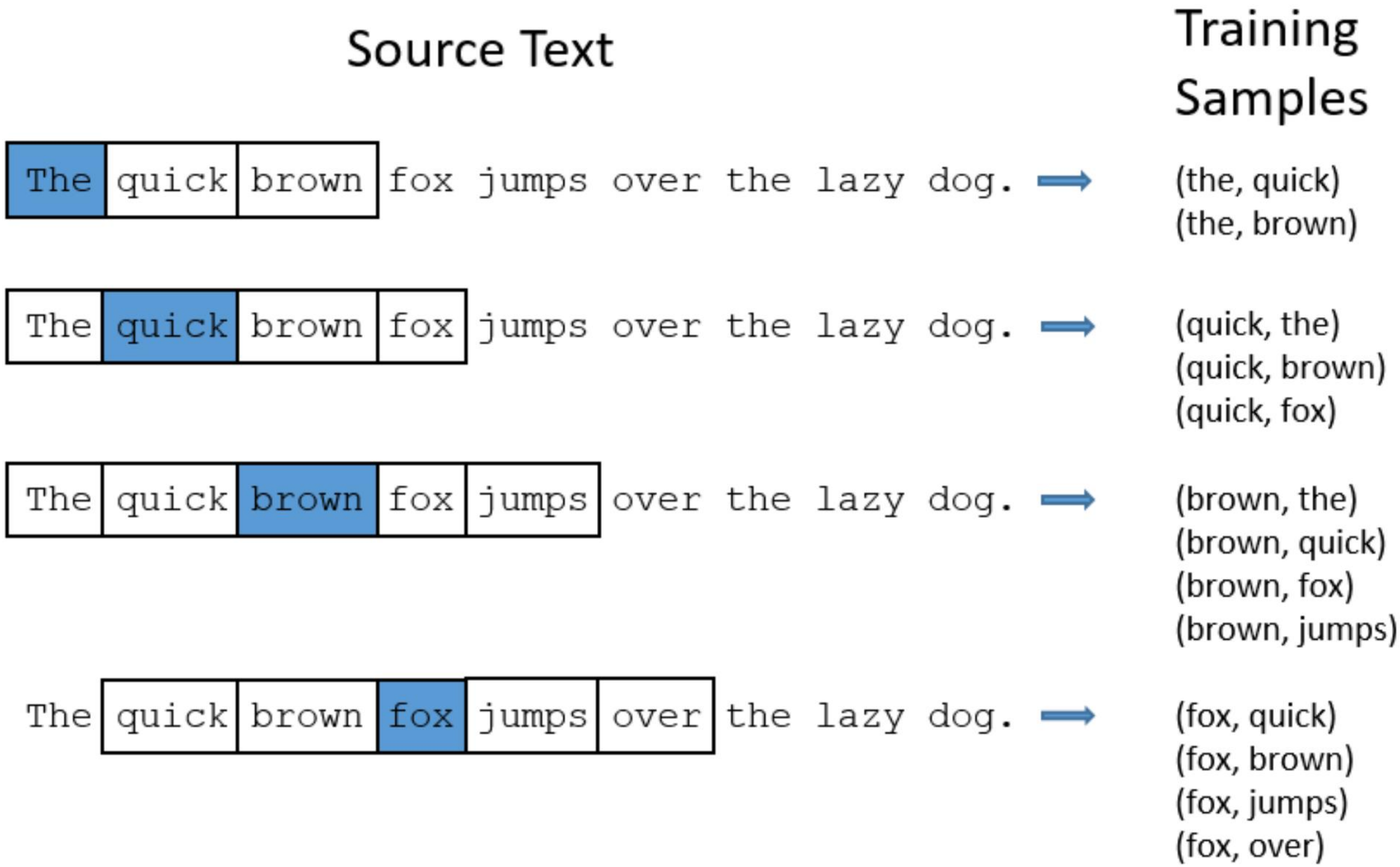
[Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.]

Skip-gram Prediction



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

http://web.stanford.edu/class/cs224n/



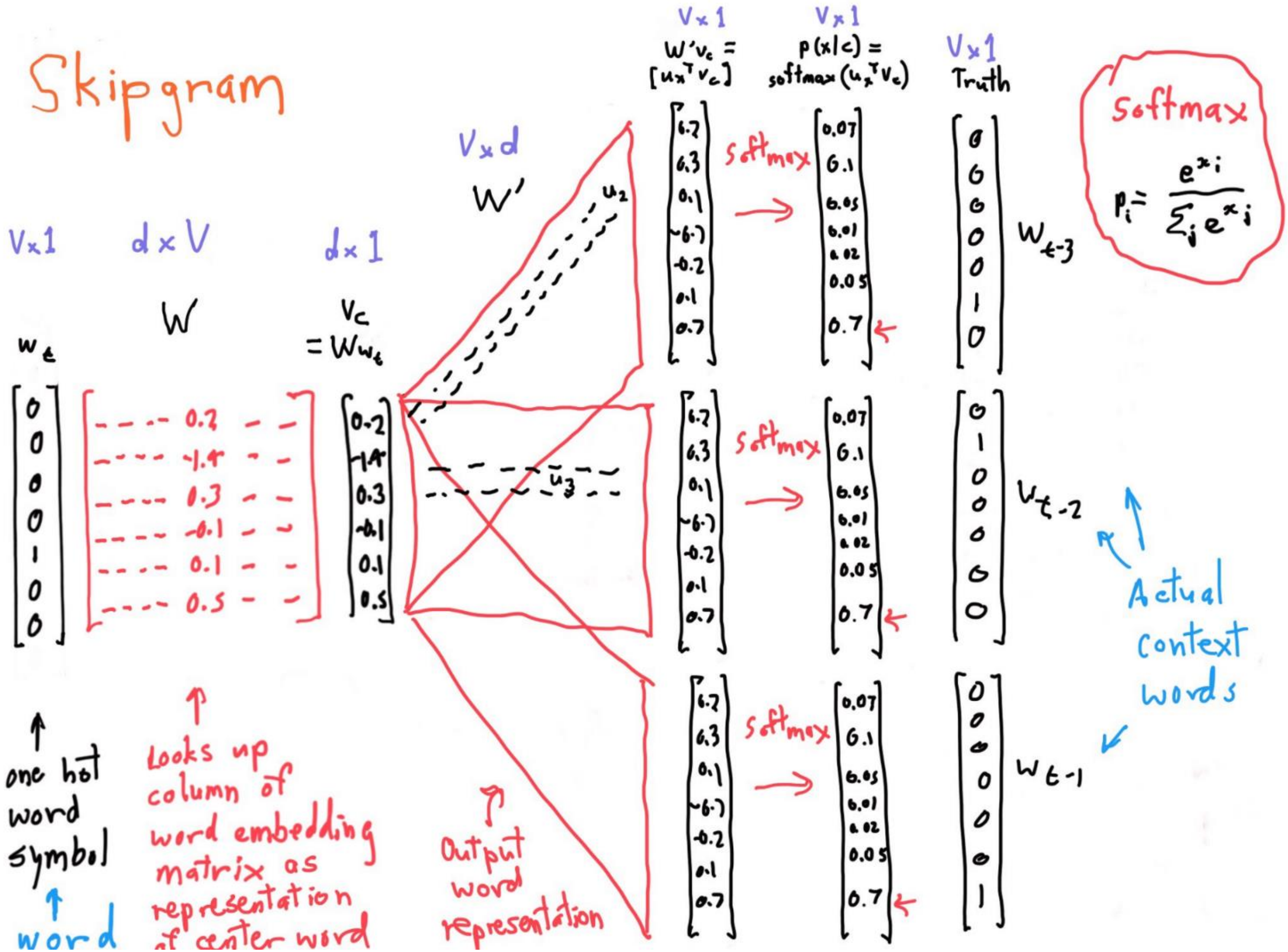
$$\mathrm{p}(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{V} \exp(u_w^T v_c)}$$

- For each word t = 1 … T, predict surrounding words in a window of "radius" m of every word.
- $\mathrm{p}(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{V} \exp(u_w^T v_c)}$

- Objective function: Maximize the probability of any context word given the current center word:

- $J'(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} \mathrm{p}\left(w_{t+j} \middle| w_t\right)$ Maximize

- $J(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log \mathrm{p}\left(w_{t+j} \middle| w_t\right)$ Minimize



http://web.stanford.edu/class/cs224n/

Objective Function

Maximize $J'(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} p(w'_{t+j} \mid w_t; \theta)$

Or minimize neg. log likelihood $J(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log p(w'_{t+j} \mid w_t)$

[negate to minimize; log is monotone]

Text length

window size

where

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

word IDs

Each word type (vocab entry) has two word representations: as center word and context word

We now take derivatives to work out minimum

http://web.stanford.edu/class/cs224n/

$$\frac{\partial}{\partial v_c} \log \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{V} \exp(u_w v_c)}$$

$$= \underbrace{\frac{\partial}{\partial v_c} \log \exp(u_o^T v_c)}_{①} - \underbrace{\frac{\partial}{\partial v_c} \log \sum_{w=1}^{V} \exp(u_w^T v_c)}_{②}$$

① $\frac{\partial}{\partial v_c} \underbrace{\log \exp}_{\text{inverses}} (u_o^T v_c) = \frac{\partial}{\partial v_c} u_o^T v_c = u_o$

Vector!
Not high school single variable calculus

You can do things one variable at a time, and this may be helpful when things get gnarly.

$$\forall j \quad \frac{\partial}{\partial (v_c)_j} u_o^T v_c = \frac{\partial}{\partial (v_c)_j} \sum_{i=1}^{d} (u_o)_i (v_c)_i$$

$$= (u_o)_j$$

Each term is zero except when $i=j$

http://web.stanford.edu/class/cs224n/

2 $$\underbrace{\frac{\partial}{\partial v_c} \log}_{f} \underbrace{\sum_{w=1}^{V} \exp(u_w^T v_c)}_{z = g(v_c)}$$

$$= \underbrace{\frac{1}{\sum_{w=1}^{V} \exp(u_w^T v_c)}}_{} \cdot \underbrace{\frac{\partial}{\partial v_c} \sum_{x=1}^{V} \exp(u_x^T v_c)}_{}$$

Important to change index

$$\frac{\partial}{\partial v_c} f(\overbrace{g(v_c)}^{z}) = \frac{\partial f}{\partial z} \cdot \frac{\partial z}{\partial v_c}$$

Use chain rule

$$= \frac{1}{\sum_{w=1}^{V} \exp(u_w^T v_c)} \cdot \left(\sum_{x=1}^{V} \frac{\partial}{\partial v_c} \underbrace{\exp}_{f} (\underbrace{u_x^T v_c}_{z = g(v_c)}) \right)$$

Move deriv inside sum

$$\left(\sum_{x=1}^{V} \exp(u_x^T v_c) \frac{\partial}{\partial v_c} u_x^T v_c \right)$$

Chain rule

$$\left(\sum_{x=1}^{V} \exp(u_x^T v_c) \, u_x \right)$$

http://web.stanford.edu/class/cs224n/

$$\frac{\partial}{\partial v_c} \log(p(o|c)) = u_o - \frac{1}{\sum_{w=1}^{V} \exp(u_w^T v_c)} \cdot \left(\sum_{x=1}^{V} \exp(u_x^T v_c) u_x \right)$$

$$= u_o - \sum_{x=1}^{V} \frac{\exp(u_x^T v_c)}{\sum_{w=1}^{V} \exp(u_w^T v_c)} u_x$$

Distribute term across sum

$$= u_o - \underbrace{\sum_{x=1}^{V} p(x|c)\, u_x}$$

This an expectation: average over all context vectors weighted by their probability

= observed − expected

This is just the derivatives for the center vector parameters
Also need derivatives for output vector parameters (they're similar)
Then we have derivative w.r.t. all parameters and can minimize

http://web.stanford.edu/class/cs224n/

Finding the degree of similarity between two words

- Vectors trained on ~100M sentences
- word segmentation with **jieba**
  - github.com/fxsjy/jieba
- $sim(u,v) = \frac{u^T v}{|u|\cdot|v|}$
- '不洗头发',
  - ('不洗头', 0.761),
  - ('洗次头', 0.751),
  - ('洗会油', 0.729),
  - ('超油', 0.715),
  - ('就会油', 0.704)

Finding the degree of similarity between two words

- '送病人',
  - ('急救车', 0.673),
  - ('抬走', 0.656),
  - ('妇产科住院', 0.648),
  - ('转送', 0.638),
  - ('救护车', 0.637)
- '胃酸过多',
  - ('胃酸', 0.811),
  - ('胃炎', 0.786),
  - ('反流性胃炎', 0.767),
  - ('烧心', 0.757),
  - ('胃溃疡', 0.756)
- '经济问题',
  - ('断绝关系', 0.645),
  - ('愚孝', 0.593),
  - ('不同意', 0.593),
  - ('撕破脸', 0.591),
  - ('两头跑', 0.586)
- '心不在焉',
  - ('走神', 0.727),
  - ('认真听讲', 0.725),
  - ('一片空白', 0.716),
  - ('专心', 0.713),
  - (‘东张西望', 0.712)

Finding the degree of similarity between two words

Nearest words to frog:

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

litoria

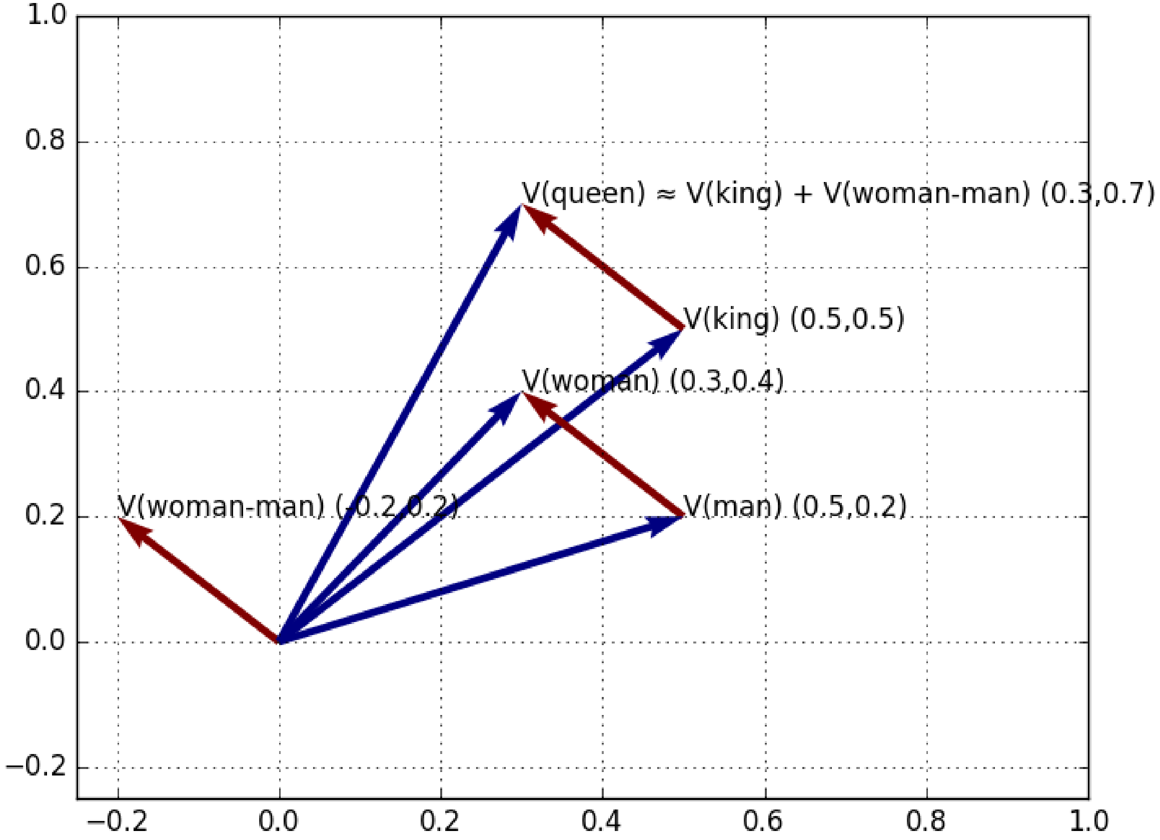
leptodactylidae

rana

eleutherodactylus

[Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.]

`king`-`man`+`woman`≈`queen`

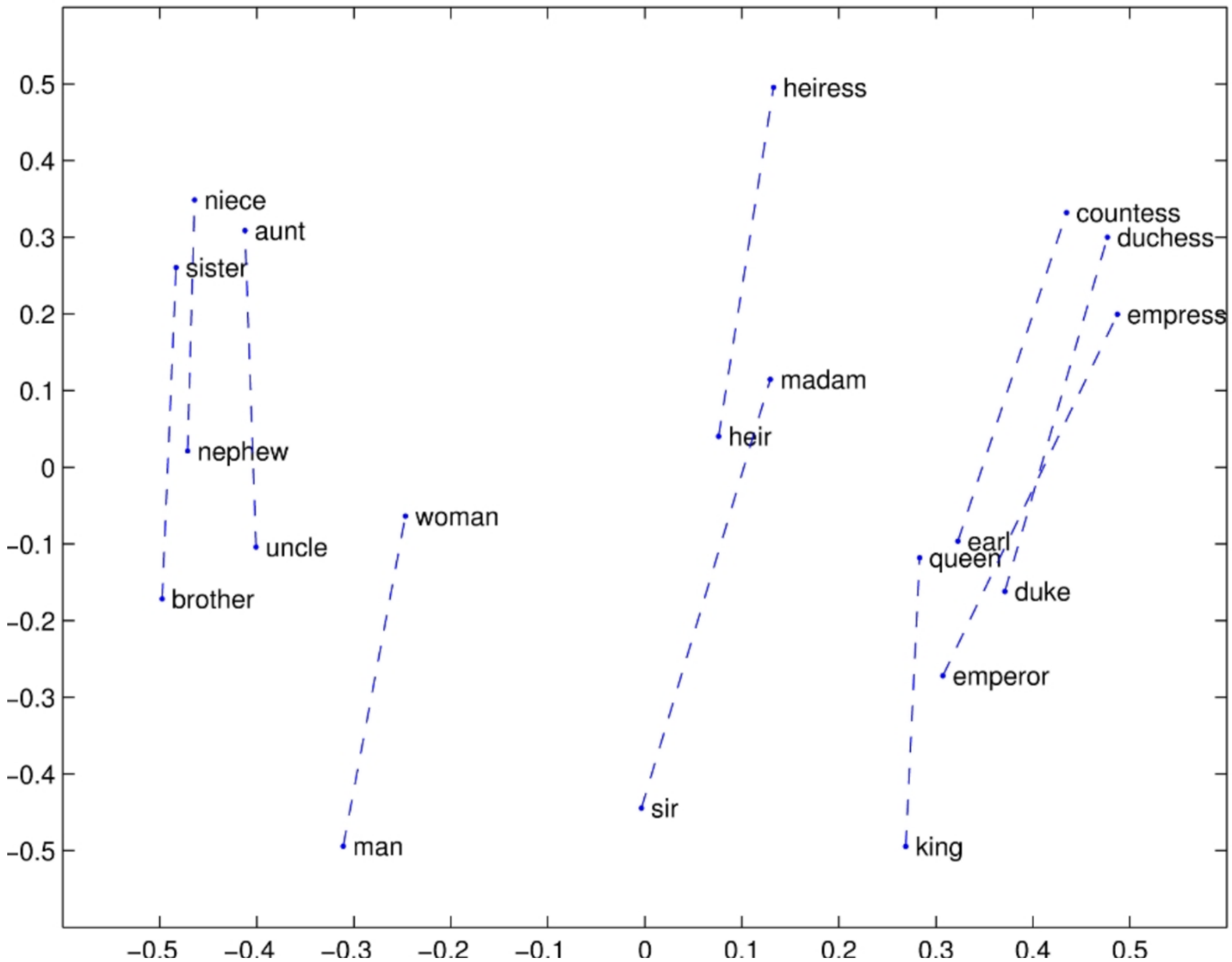


`king`-`man`+`woman`≈`queen`

Expression	*Nearest token*
Paris - France + Italy	Rome
bigger - big + cold	colder
sushi - Japan + Germany	bratwurst
Cu - copper + gold	Au
Windows - Microsoft + Google	Android
Montreal Canadiens - Montreal + Toronto	Toronto Maple Leafs

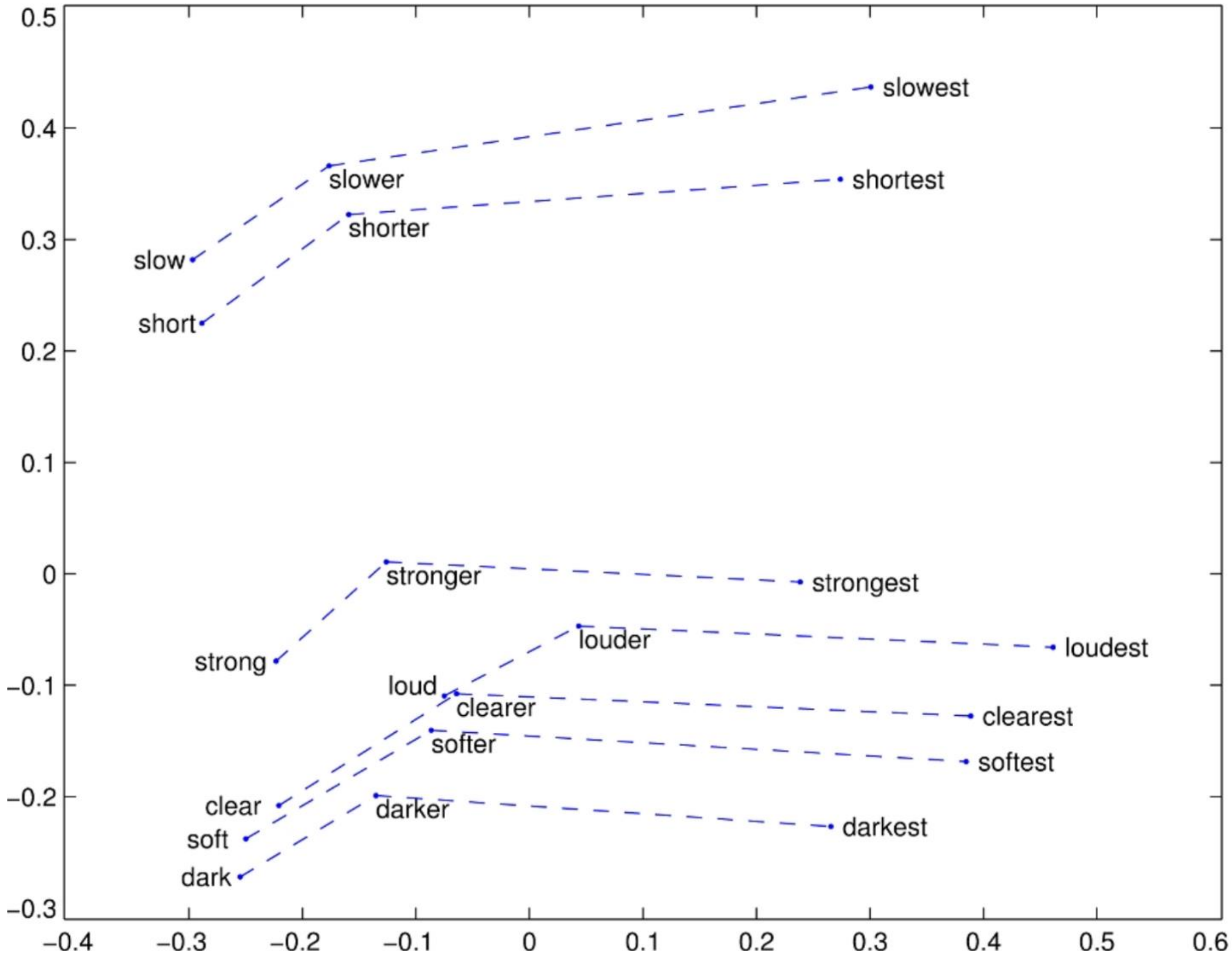
[Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.]

Linear Substructures: man-woman



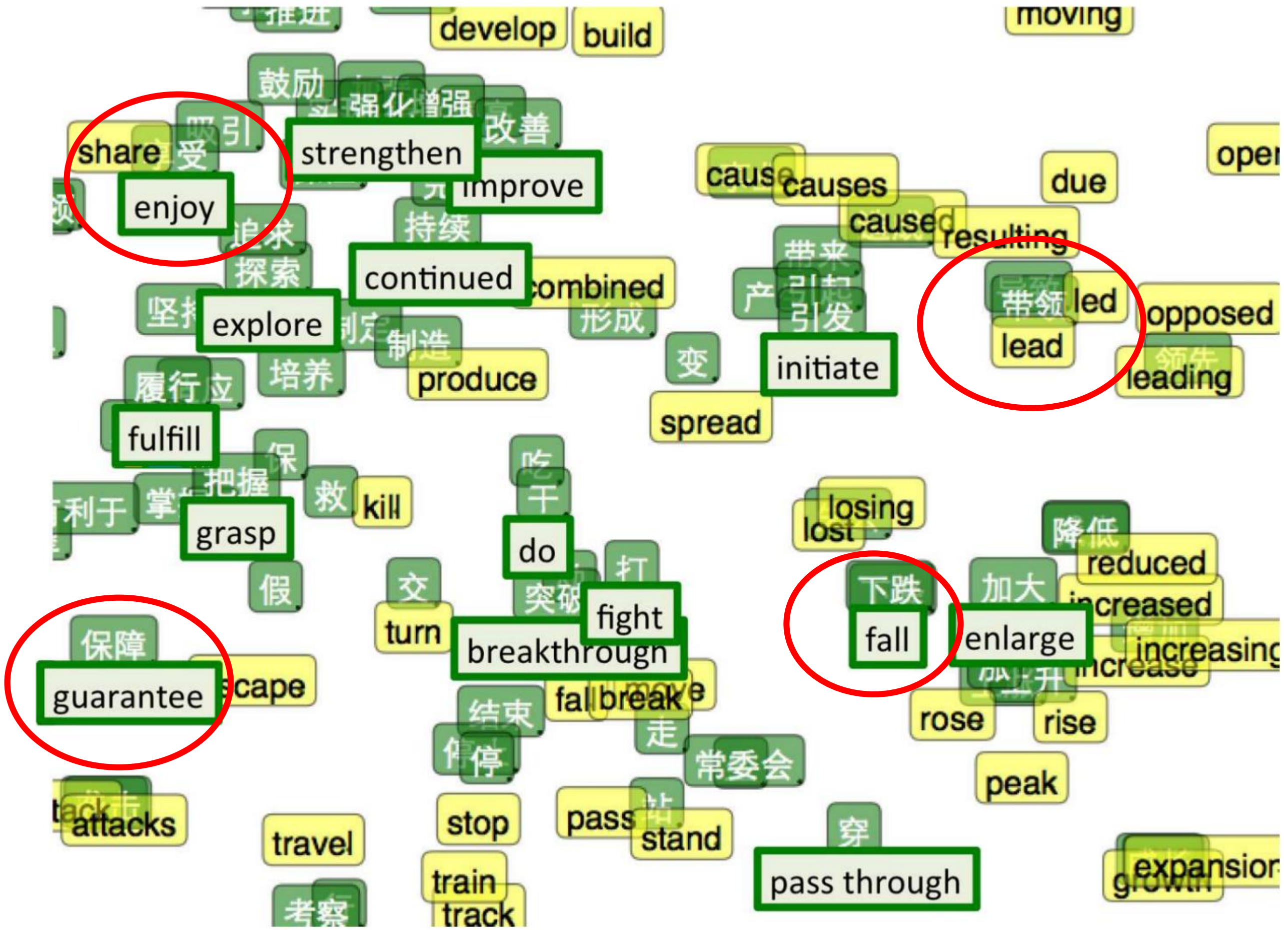
[Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.]

Linear Substructures: comparative-superlative



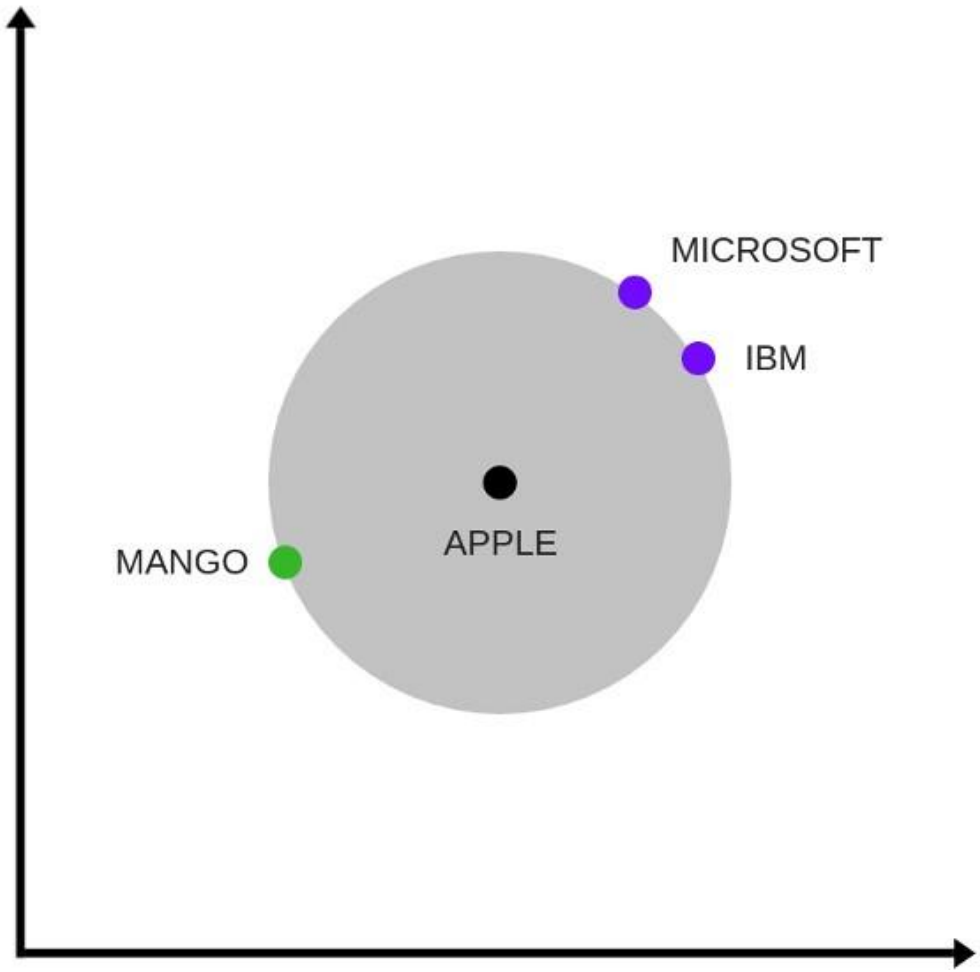
[Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.]

Bilingual Word Vectors



[Zou W Y, Socher R, Cer D, et al. Bilingual word embeddings for phrase-based machine translation[C]//Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013: 1393-1398.]

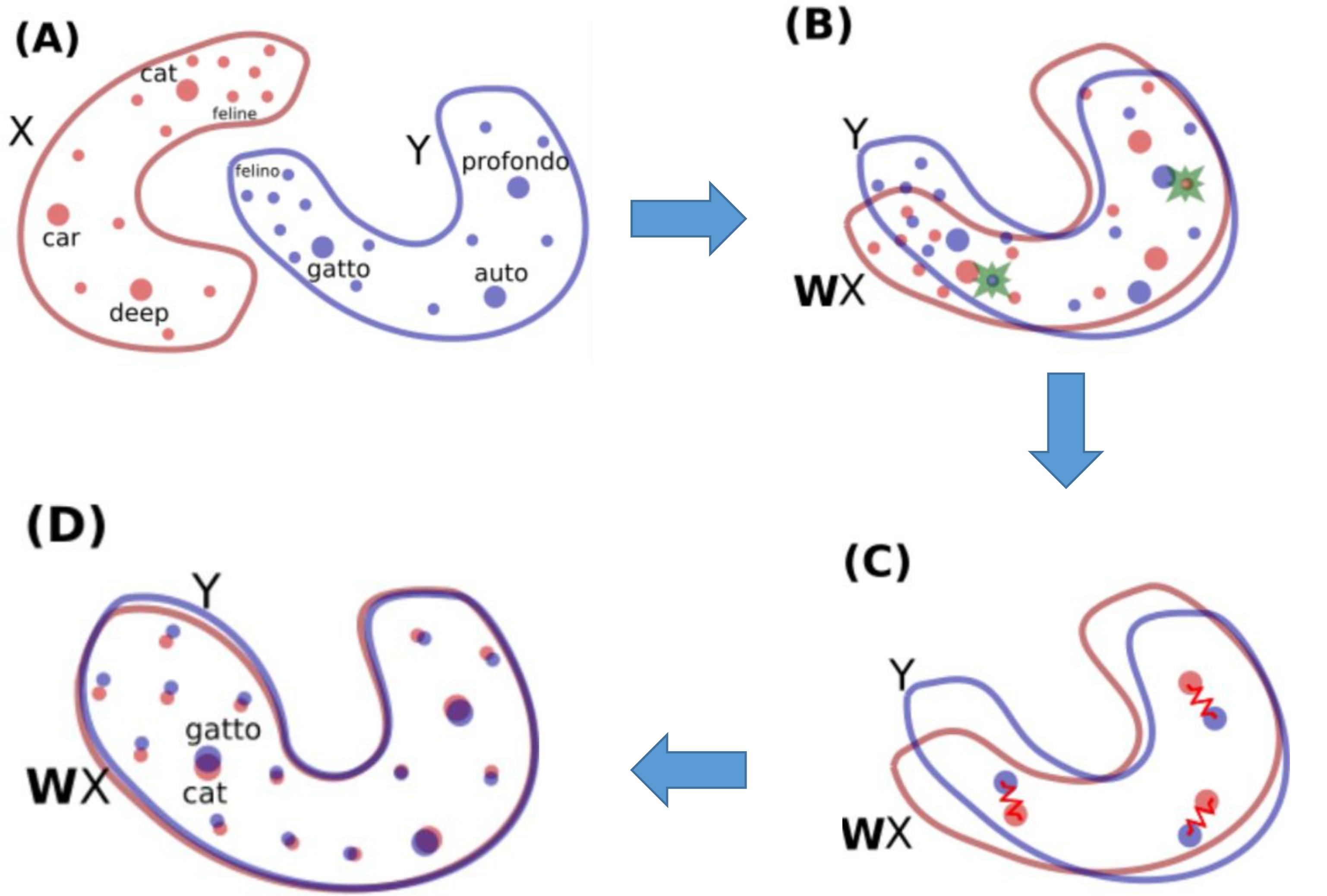
Multiple Contexts of Apple



https://hackernoon.com/word2vec-part-1-fe2ec6514d70

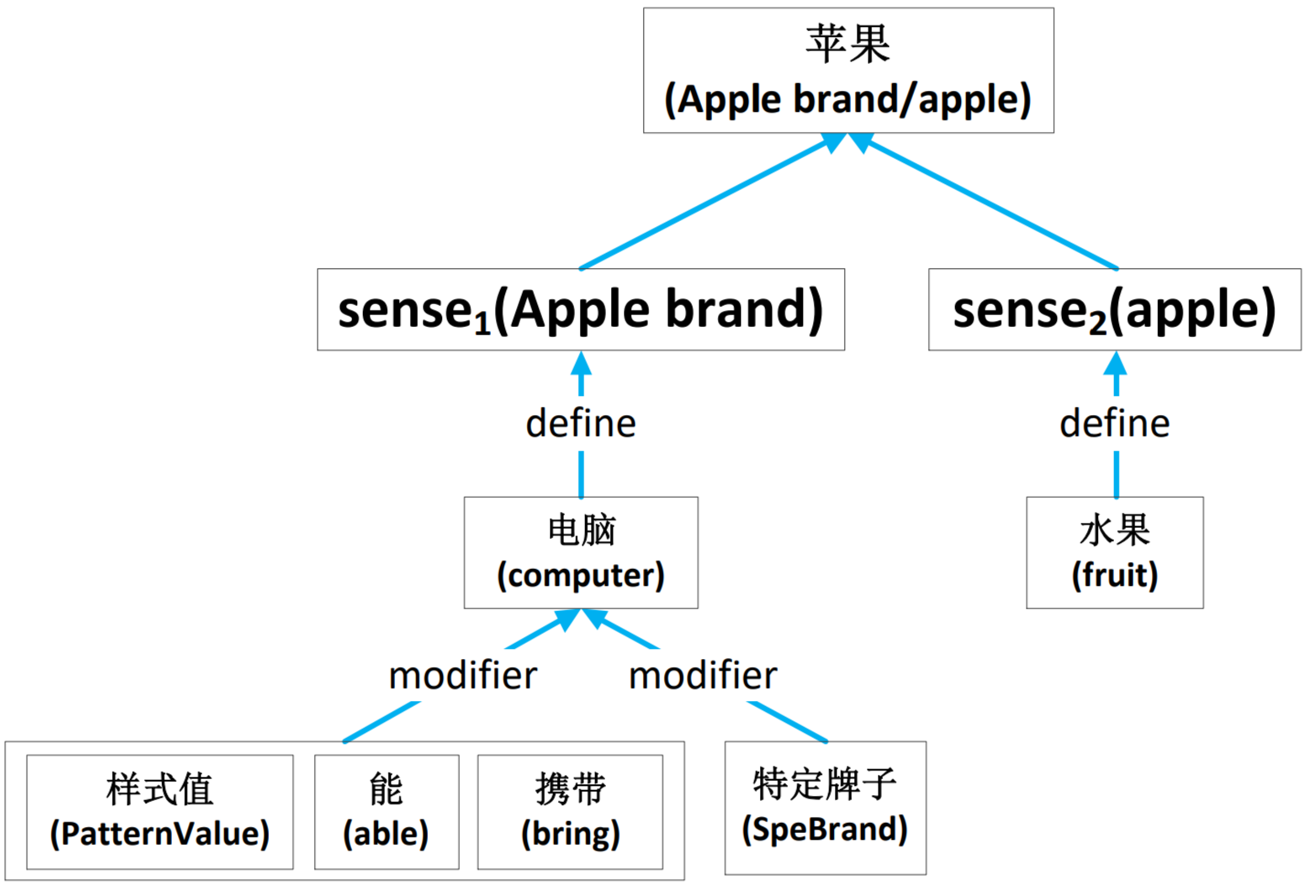
Research Frontier of Word Vector

Word translation without parallel data



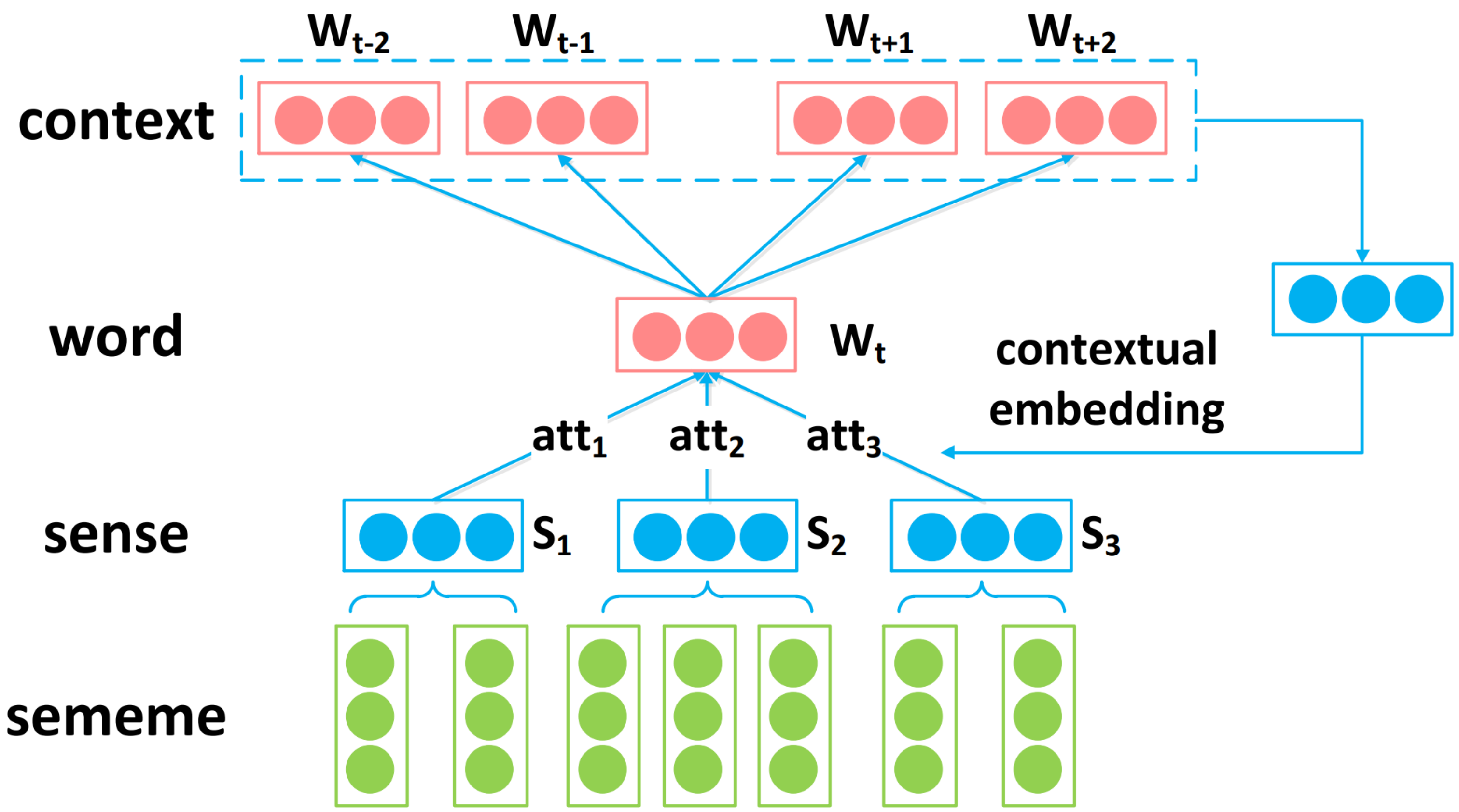
[Anonymous (2018). Word translation without parallel data. *International Conference on Learning Representations*, , .]

Improved Word Representation Learning with Sememes



[Niu Y, Xie R, Liu Z, et al. Improved Word Representation Learning with Sememes[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017, 1: 2049-2058.]

Improved Word Representation Learning with Sememes



[Niu Y, Xie R, Liu Z, et al. Improved Word Representation Learning with Sememes[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017, 1: 2049-2058.]

Q&A

Thanks!