WORD EMBEDDING/WORD2VEC OVERVIEW

Presented by Jieyu Zhao, Zeyu Li and Yichao(Joey) Zhou

OUTLINE

- Motivation
- History
- Word2Vec Algorithms
- Sampling
- Experiments
- Demo

HOW TO REPRESENT A WORD?

- 1. One Hoc Representation

apple [0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0 … 0 0 0 0 0 0]

orange [0 0 0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0 … 0 0 0 0 0 0]

car [0 0 0 0 0 0 0 **1** 0 0 0 0 0 0 0 0 0 … 0 0 0 0 0 0]

Problems?

- Sparsity -> Dimension(Vocabulary) = |V|
- *Motel* [0 0 0 0 0 1 0 0 … 0 0 0] AND *Hotel* [0 0 0 0 0 0 0 0 0 0 … 0 1 0] = 0

“YOU SHALL KNOW A WORD BY THE COMPANY IT KEEPS.”

-- JOHN RUPERT FIRTH

WORD CO-OCCURRENCE

…he curtains open and the moon shining in on the barely…

…ars and the cold , close moon " . And neither of the w…

…rough the night with the moon shining so brightly , it…

…made in the light of the moon . It all boils down , wr…

…surely under a crescent moon , thrilled by ice-white…

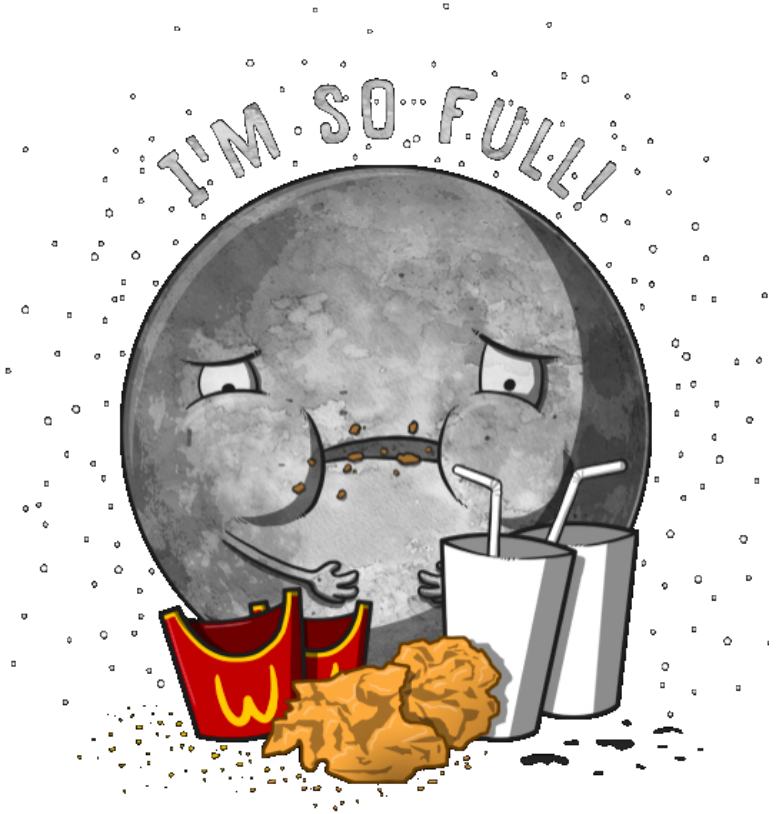
…sun , the seasons of the moon ? Home , alone , Jay pla…

…m is dazzling snow , the moon has risen full and cold…

…un and the temple of the moon , driving out of the hug…

…in the dark and now the moon rises , full and amber a…

…bird on the shape of the moon over the trees in front…



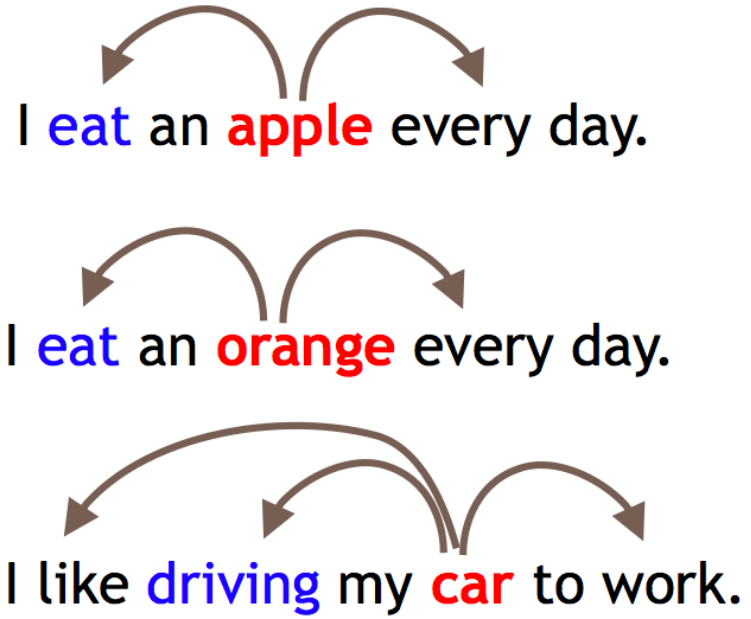
CONTEXT REPRESENTATION

Word is represented by context in use.

But how to use the context?

Create a co-occurrence matrix!

- Full text ❌
- Window ✅



WINDOW BASED CO-OCCURRENCE MATRIX

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

Problem?

- Increase in **size** with vocabulary and require **HUGE storage**.

HOW TO REPRESENT A WORD?

- 2. Word Embedding

What's a word embedding ?

-Word Embeddings aim at quantifying and categorizing **semantic similarities** between linguistic items based on their distributional properties in large samples of language data. *--Wikipedia*

-A parametrized function **W**: words -> $\mathbf{R^n}$ mapping words in some language to a low dimensional vector (typically 50-500)

W("*cat*") = (0.2, −0.4, 0.7, …)

W("*mat*") = (0.0, 0.6, −0.1, …)

DISTRIBUTED REPRESENTATIONS OF WORDS AND PHRASES AND THEIR COMPOSITIONALITY

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean

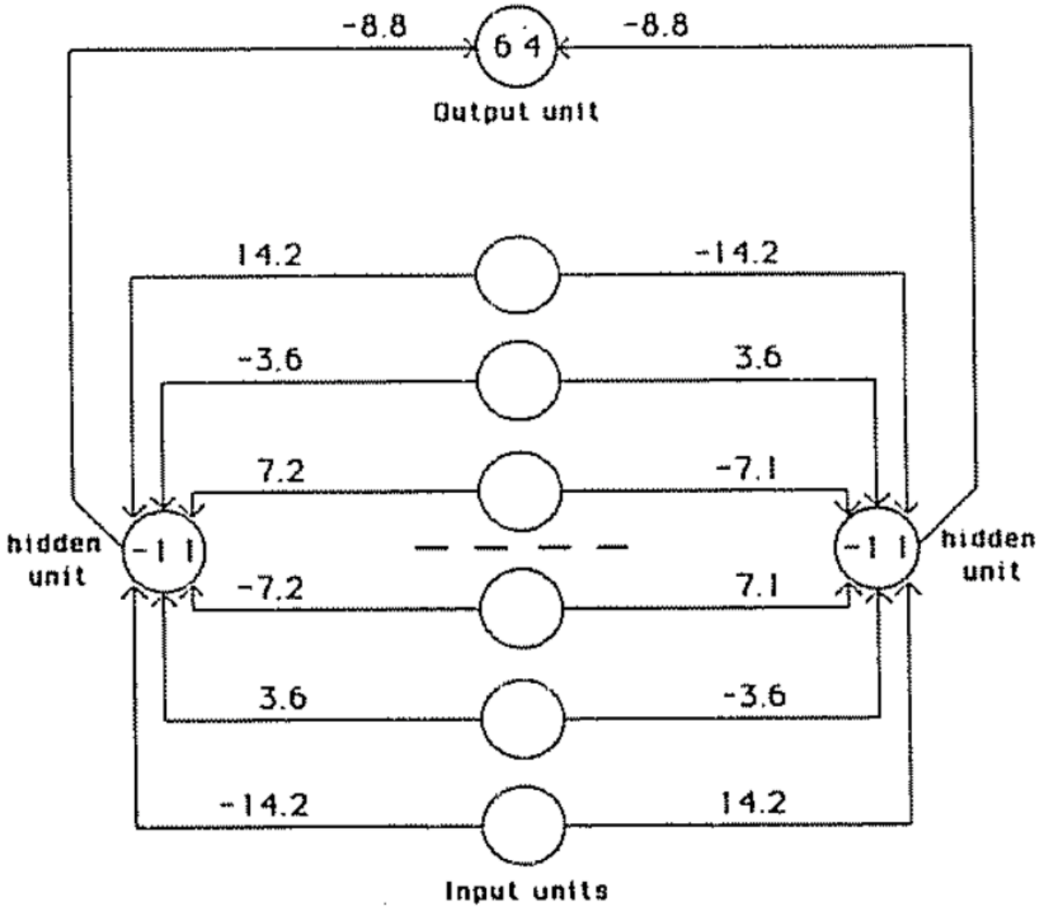
Oct 2013

A BRIEF HISTORY

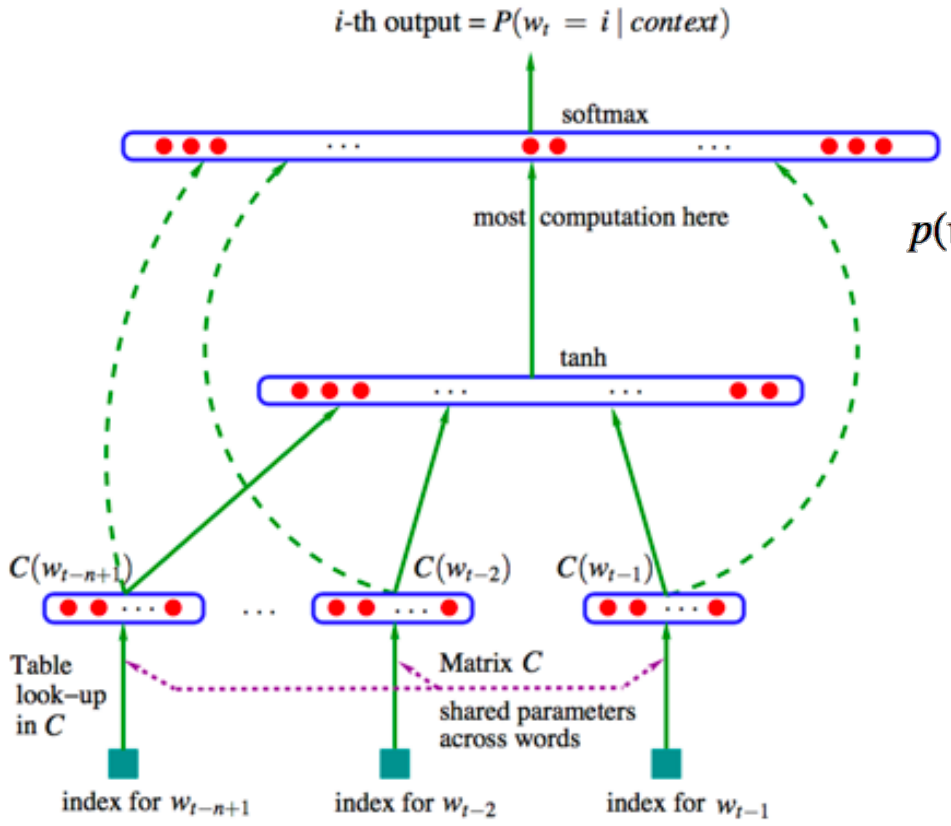
- One of the **earliest** use of word representations dates back to 1986 due to Rumelhart, Hinton, and Williams.
- Distributed word representation has been applied to statistical language modeling with **considerable success** in 2003 by Bengio et al.
- Word2Vec, a toolkit enabling the training and the use of pre-trained embeddings was proposed by Mikolov et al., which has brought embedding to the **forefront** in NLP in 2013.

EARLIEST WORD REPRESENTATION RUMELHART 1986

- This paper introduced the **back-propagation algorithm**.
- As a result of parameter adjustment, hidden units come to represent **important features in specific task domain**.
- The ability to create **new features** distinguish back-propagation from earlier methods like **perceptron-convergence algorithm**.



NEURAL NETWORK LANGUAGE MODEL BENGIO 2003



$$p(w_i|w_1, w_2, \ldots, w_{t-1}) \approx f(w_i, w_{t-1}, \ldots, w_{t-n+1}) = g(w_i, C(w_{t-n+1}), \ldots, C(w_{t-1}))$$

$$L(\theta) = \frac{1}{T} \sum_t \log f(w_t, w_{t-1}, \ldots, w_{t-n+1}) + R(\theta)$$

Problems?

1. Input length is fixed.
2. Too slow

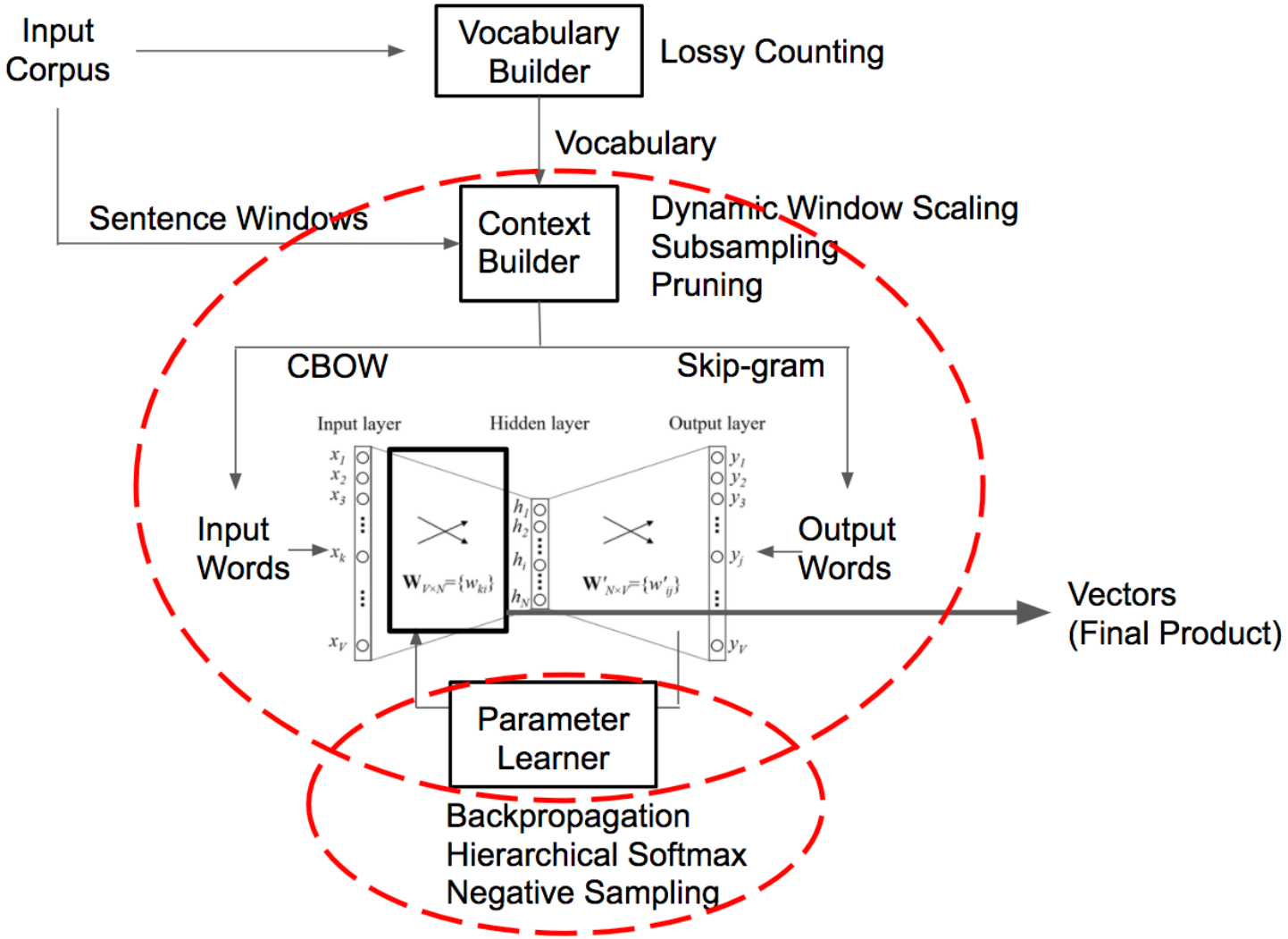
MOTIVATION - I

- Word2vec get better performance in NLP: grouping similar words
- Skip-gram model:
  - Learns word vectors from unstructured text data
  - Does not rely on dense matrix multiplications
  - Efficient training: single machine can train on 100 billion words per day
  - Encodes linguistic regularities and patterns:
    - Linear translations: vec(“Madrid”) – vec(“Spain”) + vec(“France”) → vec(“Paris”)

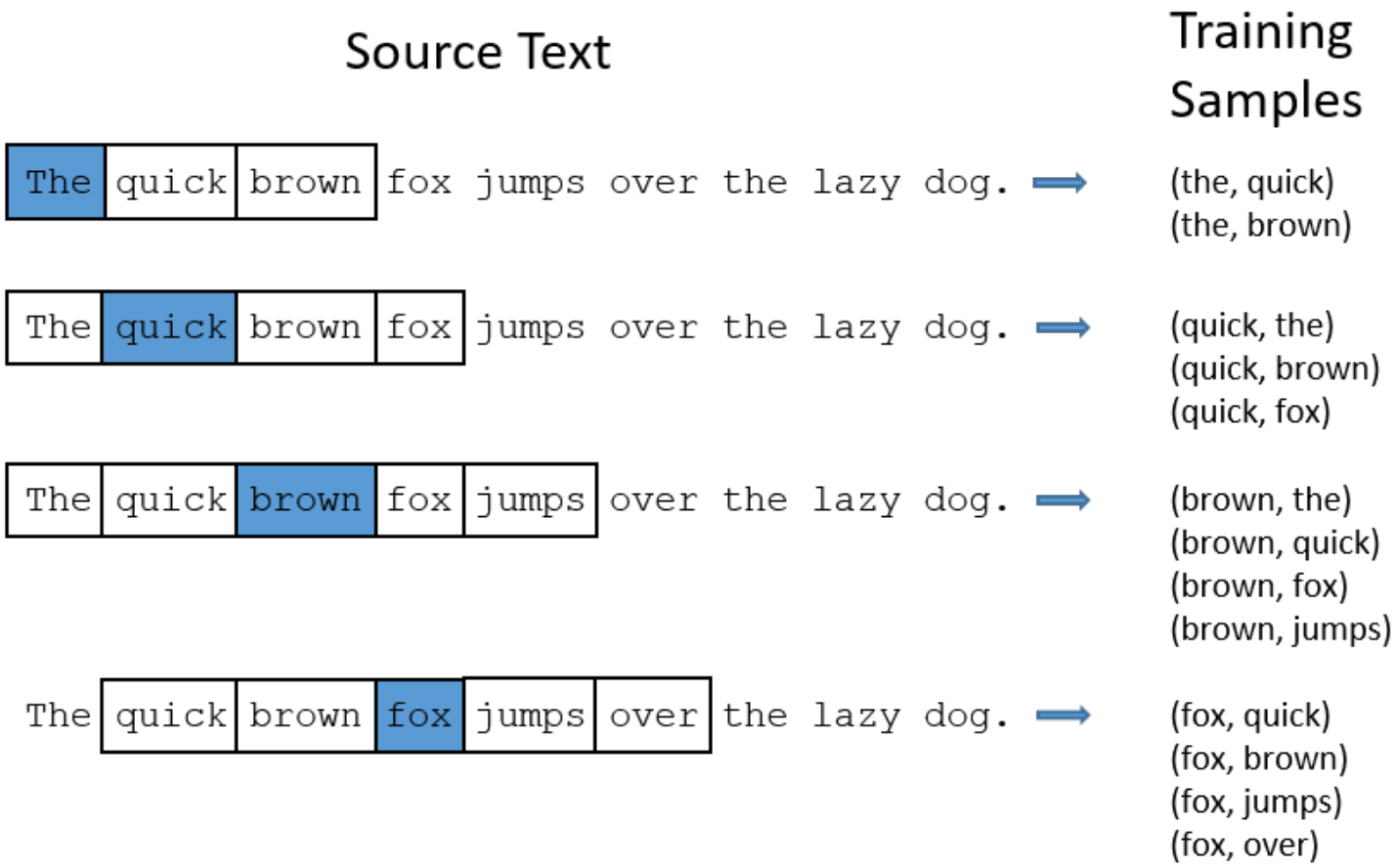
MOTIVATION - II

- Time consuming
  - softmax function: proportional to the vocabulary
- Imbalance between rare and frequent words
  - “the”, “a”, “in”
  - do not change significantly after several million examples
- Limitations in phrases representation
  - Not just simply combining different words

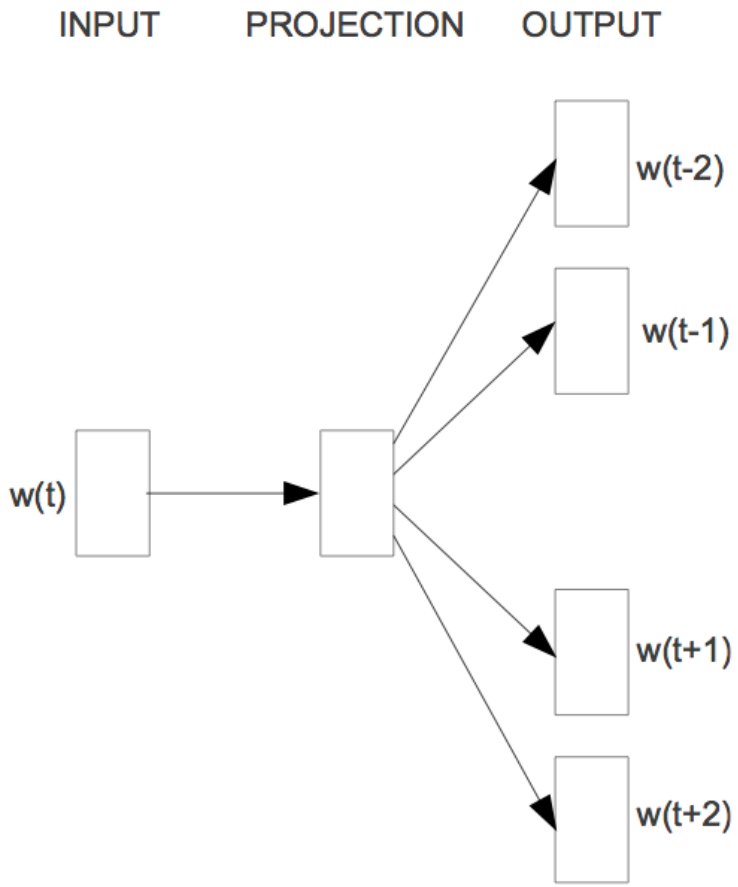
WORD2VEC DECOMPOSED



WORD2VEC ALGORITHM SKIP-GRAM



SKIP-GRAM

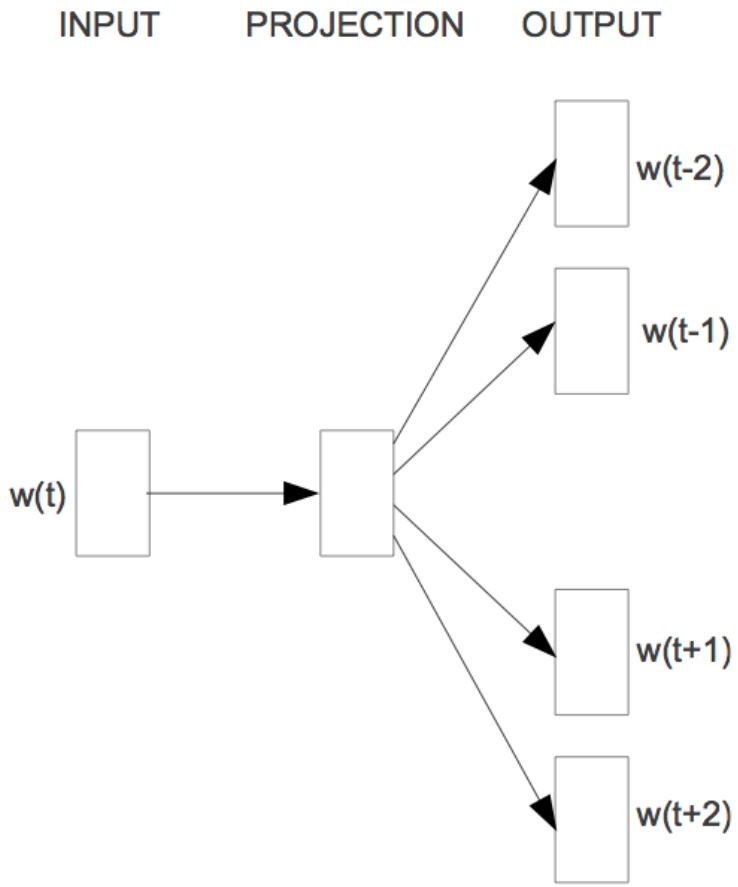


Main idea: maximizing the following probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

Larger c **:** More accuracy, more computational cost

SKIP-GRAM



Main idea: maximizing the following probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

Larger c **:** More accuracy, more computational cost

SKIP-GRAM

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^{\top} v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left({v'_w}^{\top} v_{w_I}\right)}$$

w_o: **Output word**

w_I: **Input word**

W: **Vocabulary**

v'_{wo}: **Output word representation**

v_{wI} : **Input word representation**

$\nabla \log p(w_O|w_I)$ Proportional to W, which is EXPENSIVE

NEGATIVE SAMPLING

$$\arg\max_{\theta} \prod_{w \in Text} \left[\prod_{c \in C(w)} p(c|w;\theta) \right]$$

$$\arg\max_{\theta} \prod_{(w,c) \in D} p(c|w;\theta) \qquad p(c|w;\theta) = \frac{e^{v_c \cdot v_w}}{\sum_{c' \in C} e^{v_{c'} \cdot v_w}}$$

$$\arg\max_{\theta} \sum_{(w,c) \in D} \log p(c|w) = \sum_{(w,c) \in D} \left(\log e^{v_c \cdot v_w} - \log \sum_{c'} e^{v_{c'} \cdot v_w} \right)$$

NEGATIVE SAMPLING

Basic Idea:

- When $D = 1$, (w,c) is from corpus data

$$\arg\max_{\theta} \prod_{(w,c)\in D} p(D=1|w,c;\theta)$$

$$= \arg\max_{\theta} \log \prod_{(w,c)\in D} p(D=1|w,c;\theta)$$

$$= \arg\max_{\theta} \sum_{(w,c)\in D} \log p(D=1|w,c;\theta)$$

$$p(D=1|w,c;\theta) = \frac{1}{1+e^{-v_c \cdot v_w}}$$

$$\arg\max_{\theta} \sum_{(w,c)\in D} \log \frac{1}{1+e^{-v_c \cdot v_w}}$$

NEGATIVE SAMPLING

$$\arg\max_{\theta} \prod_{(w,c)\in D} p(D=1|c,w;\theta) \prod_{(w,c)\in D'} p(D=0|c,w;\theta)$$

$$= \arg\max_{\theta} \prod_{(w,c)\in D} p(D=1|c,w;\theta) \prod_{(w,c)\in D'} (1-p(D=1|c,w;\theta))$$

$$= \arg\max_{\theta} \sum_{(w,c)\in D} \log p(D=1|c,w;\theta) + \sum_{(w,c)\in D'} \log(1-p(D=1|w,c;\theta))$$

$$= \arg\max_{\theta} \sum_{(w,c)\in D} \log \frac{1}{1+e^{-v_c \cdot v_w}} + \sum_{(w,c)\in D'} \log(1-\frac{1}{1+e^{-v_c \cdot v_w}})$$

$$= \arg\max_{\theta} \sum_{(w,c)\in D} \log \frac{1}{1+e^{-v_c \cdot v_w}} + \sum_{(w,c)\in D'} \log(\frac{1}{1+e^{v_c \cdot v_w}})$$

NEGATIVE SAMPLING

$$\arg\max_{\theta} \sum_{(w,c)\in D} \log \frac{1}{1+e^{-v_c \cdot v_w}} + \sum_{(w,c)\in D'} \log\left(\frac{1}{1+e^{v_c \cdot v_w}}\right)$$

$$= \arg\max_{\theta} \sum_{(w,c)\in D} \log \sigma(v_c \cdot v_w) + \sum_{(w,c)\in D'} \log \sigma(-v_c \cdot v_w)$$

D': the set of negative samples, i.e. when $(w,c) \in D'$, (w,c) is not in corpus data.

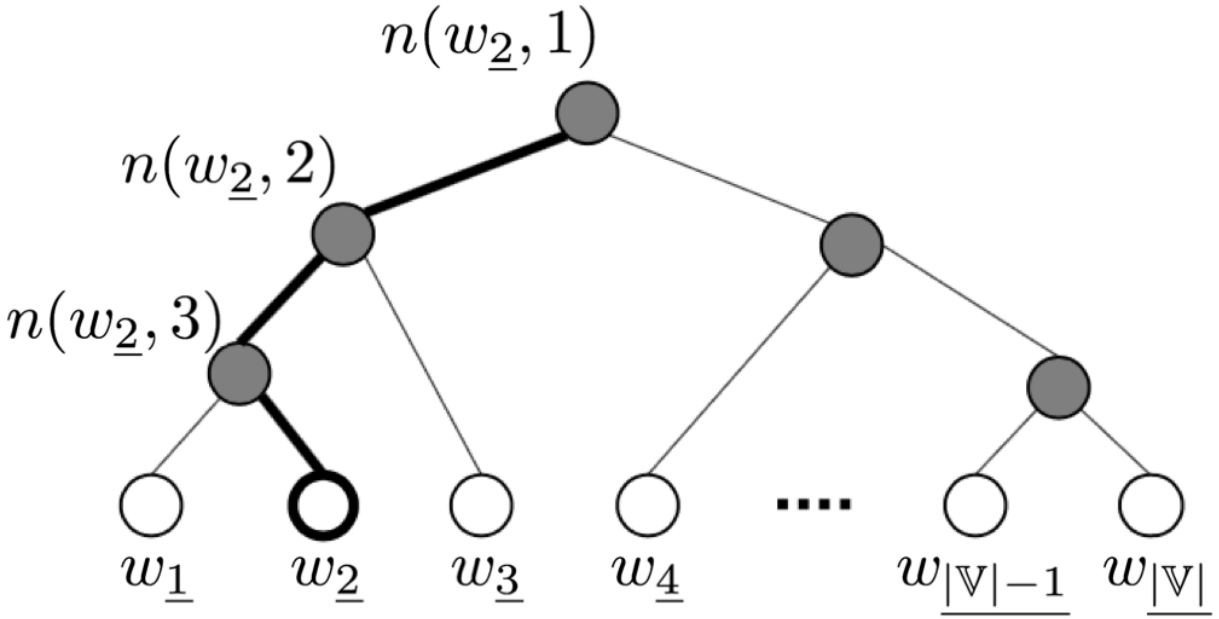
How to draw $(w,c) \in D'$? $(w,c) \sim p_{words}(w)\frac{p_{contexts}(c)^{3/4}}{Z}$

HIERARCHICAL SOFTMAX

$$\arg\max_{\theta} \sum_{(w,c)\in D} \log p(c|w) = \sum_{(w,c)\in D} \left(\log e^{v_c \cdot v_w} - \log \sum_{c'} e^{v_{c'} \cdot v_w}\right)$$

Very large computational cost if calculate the first derivative of this term.

HIERARCHICAL SOFTMAX



Huffman Tree

$$
\begin{aligned}
p(w_2 = w_O) &= p\left(n(w_2, 1), \text{left}\right) \cdot p\left(n(w_2, 2), \text{left}\right) \cdot p\left(n(w_2, 3), \text{right}\right) \\
&= \sigma\left({\mathbf{v}'_{n(w_2,1)}}^T \mathbf{h}\right) \cdot \sigma\left({\mathbf{v}'_{n(w_2,2)}}^T \mathbf{h}\right) \cdot \sigma\left(-{\mathbf{v}'_{n(w_2,3)}}^T \mathbf{h}\right)
\end{aligned}
$$

HIERARCHICAL SOFTMAX

$$p(w|w_I) = \prod_{j=1}^{L(w)-1} \sigma\left([\![n(w,j+1) = \mathrm{ch}(n(w,j))]\!] \cdot {v'_{n(w,j)}}^{\top} v_{w_I}\right)$$

$[\![n(w, j+1) = \mathrm{ch}(n(w,j))]\!] = 1$ if $n(w, j+1) = \mathrm{ch}(n(w,j))$

$[\![n(w, j+1) = \mathrm{ch}(n(w,j))]\!] = -1$ if $n(w, j+1) \neq \mathrm{ch}(n(w,j))$

$\mathrm{ch}(n(w,j))$: left child of $n(w,j)$

$$\sum_{i=1}^{V} p(w_i = w_O) = 1$$

EXPERIMENTS - I

- Accuracy on analogical reasoning task
  - Syntactic: quick : quickly :: slow : slowly
  - Semantic: Berlin: Germany :: Paris: France

Method	Time [min]	Syntactic [%]	Semantic [%]	Total accuracy [%]
NEG-5	38	63	54	59
NEG-15	97	63	58	**61**
HS-Huffman	41	53	40	47
NCE-5	38	60	45	53
The following results use 10^{-5} subsampling				
NEG-5	14	61	58	60
NEG-15	36	61	61	**61**
HS-Huffman	21	52	59	55

  - k negative samples for each positive sample
  - NCE: Noise Contrastive Estimation
  - HS-Huffman: Hierarchical softmax with frequency-based Huffman codes

EXPERIMENTS - II

- Accuracy for phrase analogy task

Method	Dimensionality	No subsampling [%]	10^{-5} subsampling [%]
NEG-5	300	24	27
NEG-15	300	27	42
HS-Huffman	300	19	**47**

	NEG-15 with 10^{-5} subsampling	HS with 10^{-5} subsampling
Vasco de Gama	Lingsugur	Italian explorer
Lake Baikal	Great Rift Valley	Aral Sea
Alan Bean	Rebbeca Naomi	moonwalker
Ionian Sea	Ruegen	Ionian Islands
chess master	chess grandmaster	Garry Kasparov

EXPERIMENTS - III

- Comparison with previous work

Model (training time)	Redmond	Havel	ninjutsu	graffiti	capitulate
Collobert (50d) (2 months)	conyers lubbock keene	plauen dzerzhinsky osterreich	reiki kohona karate	cheesecake gossip dioramas	abdicate accede rearm
Turian (200d) (few weeks)	McCarthy Alston Cousins	Jewell Arzu Ovitz	- - -	gunfire emotion impunity	- - -
Mnih (100d) (7 days)	Podhurst Harlang Agarwal	Pontiff Pinochet Rodionov	- - -	anaesthetics monkeys Jews	Mavericks planning hesitated
Skip-Phrase (1000d, 1 day)	Redmond Wash. Redmond Washington Microsoft	Vaclav Havel president Vaclav Havel Velvet Revolution	ninja martial arts swordsmanship	spray paint grafitti taggers	capitulation capitulated capitulating

LINGUISTIC REGULARITIES IN CONTINUOUS SPACE WORD REPRESENTATIONS

Tomas Mikolov, Wen-tau Yih, Geoffrey Zweig

June 2013

MOTIVATION

- N-gram models discrete units without **inherent relationship** to one another
- Distributed representations achieves **generalization**
- Word vectors can **map** similar words to similar vectors

- Word representations can capture meaningful **syntactic and semantic regularities**
  - constant vector offsets

MEASURE LINGUISTIC REGULARITY - I

- Syntactic Test Set
  - “a is to b as c is to ___”
  - Base/comparative/superlative forms of adjectives
  - Singular/plural forms of common nouns
  - Possessive/non-possessive forms of common nouns
  - Base/ past/ 3rd person present tense forms of verbs
  - 8000 test instances

MEASURE LINGUISTIC REGULARITY - II

- Semantic Test Set
  - Order the target pairs according to the degree to which the relation holds
    - words with same relation
    - Class-Inclusion: Singular_Collective
    - e.g. How valid is “clothing: shirt :: dish: bowl”
  - 79 fine-grained word relations
  - Each relation is exemplified by 3-4 word pairs
  - 10 for training/ 69 testing

VECTOR OFFSET METHOD

- Formulate both syntactic and semantic tasks as **analogy** questions
- **Cosine distance** is remarkably effective
- 1) Suppose given a:b :: c:d , d **unknown**:
  - $y = x_b - x_a + x_c$
  - If not existing:

$$w^* = argmax_w \frac{x_w y}{\|x_w\| \|y\|}$$

- 2) d is given (semantic test set):
  - Calculate $\cos(x_b - x_a + x_c, x_d)$

EXPERIMENTAL RESULTS - I

- Trained with 320M words of Broadcast News data
- In total 82k vocabulary
- RNN vs. LSA

Method	Adjectives	Nouns	Verbs	All
LSA-80	9.2	11.1	17.4	12.8
LSA-320	11.3	18.1	20.7	16.5
LSA-640	9.6	10.1	13.8	11.3
RNN-80	9.3	5.2	30.4	16.2
RNN-320	18.2	19.0	45.0	28.5
RNN-640	21.0	25.2	54.8	34.7
RNN-1600	**23.9**	**29.2**	**62.2**	**39.6**

EXPERIMENTAL RESULTS - II

- Comparison with other methods (Syntactic)
- Trained on 37M words
  - 36k word vocabulary
  - 6632 questions in test set

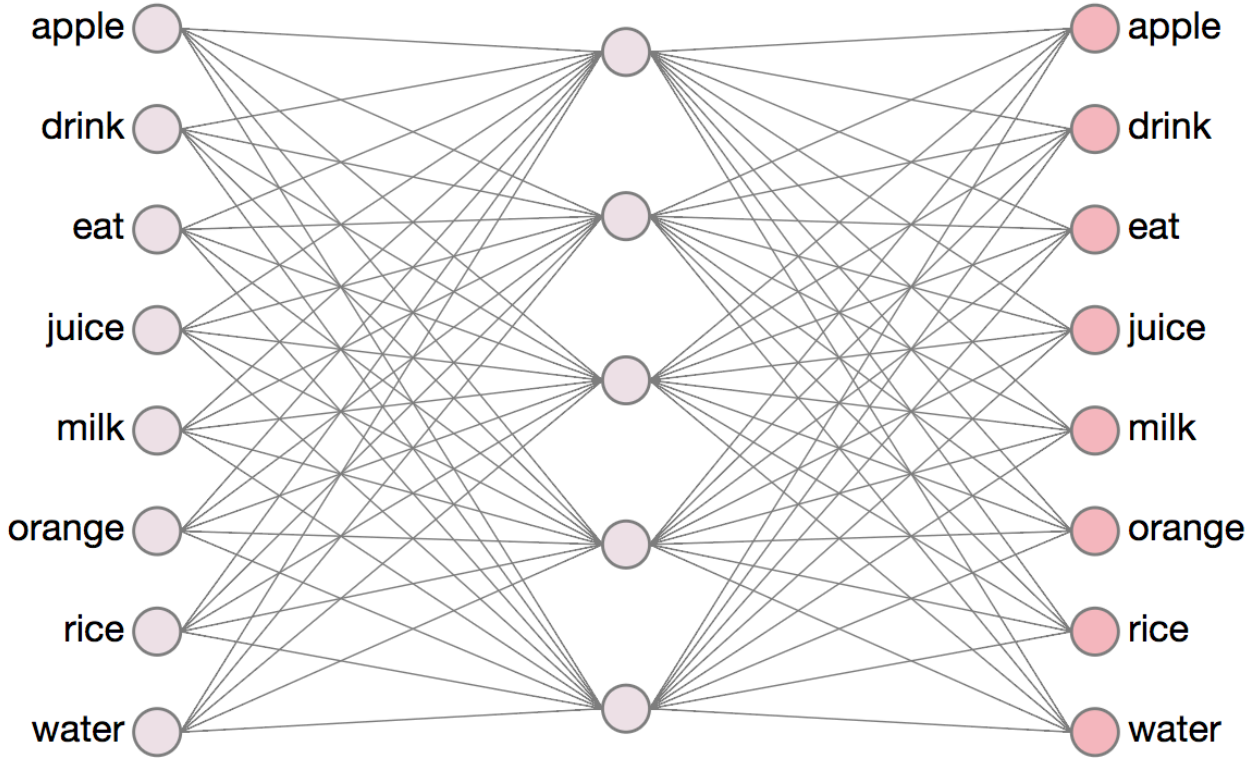
Method	Adjectives	Nouns	Verbs	All
RNN-80	**10.1**	8.1	**30.4**	**19.0**
CW-50	1.1	2.4	8.1	4.5
CW-100	1.3	4.1	8.6	5.0
HLBL-50	4.4	5.4	23.1	13.0
HLBL-100	7.6	**13.2**	30.2	18.7

EXPERIMENTAL RESULTS-III

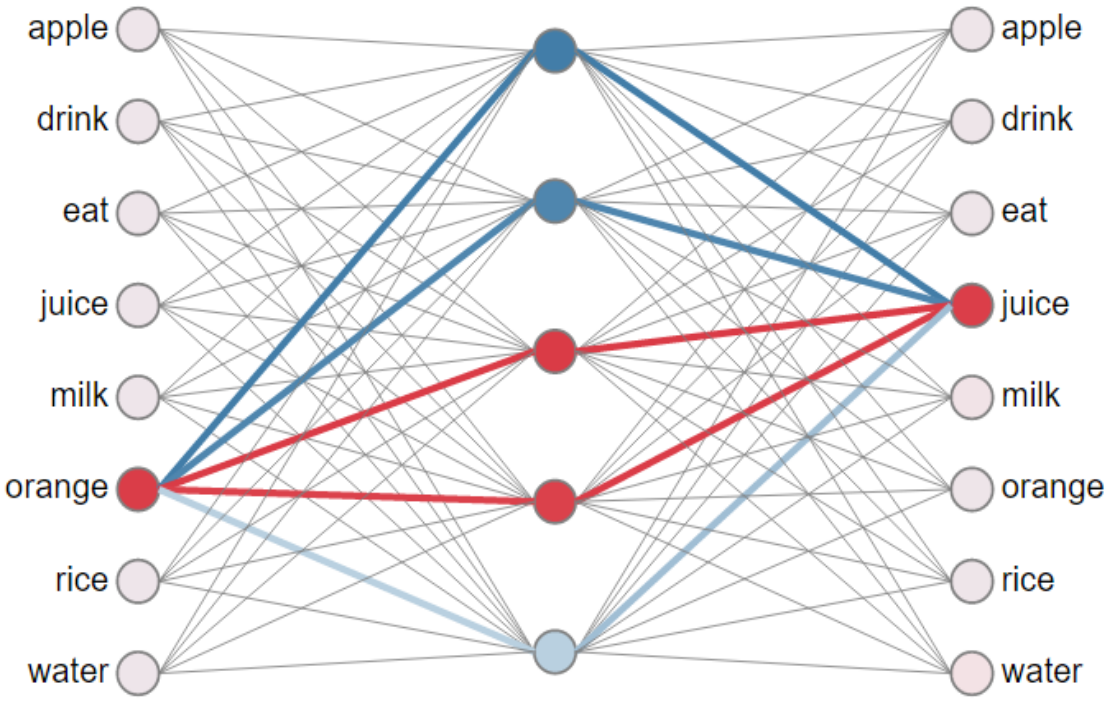
- Semantic Regularities
- Measure the average relational similarity of all the target word pairs
- Two metrics: Spearman's rank correlation coefficient; MaxDiff accuracy (SemEval 2012 Task)
- Average score over all 69 relations
  - UTD-NB: previous best model
  - RNN is not trained or tuned specifically for this task
  - RNN get better results as dimension increases

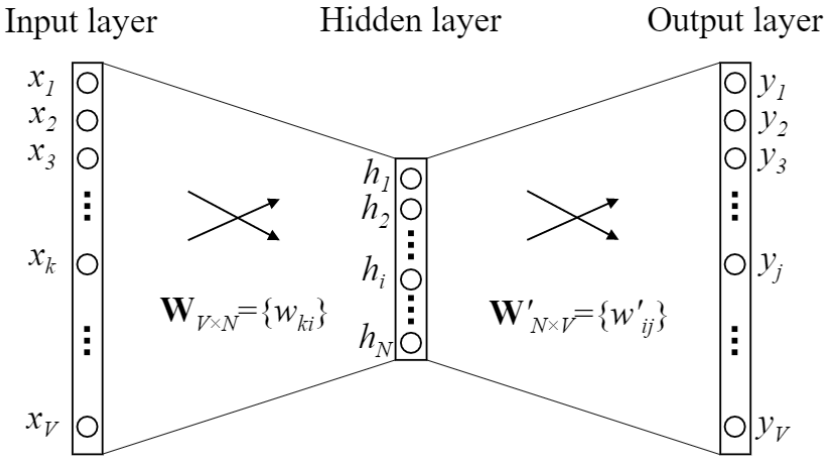
Method	Spearman's ρ	MaxDiff Acc.
LSA-640	0.149	0.364
RNN-80	0.211	0.389
RNN-320	0.259	0.408
RNN-640	0.270	0.416
RNN-1600	**0.275**	**0.418**
CW-50	0.159	0.363
CW-100	0.154	0.363
HLBL-50	0.149	0.363
HLBL-100	0.146	0.362
UTD-NB	0.230	0.395

DEMO – WEVI(WORD2VEC)



WORD2VEC NETWORK



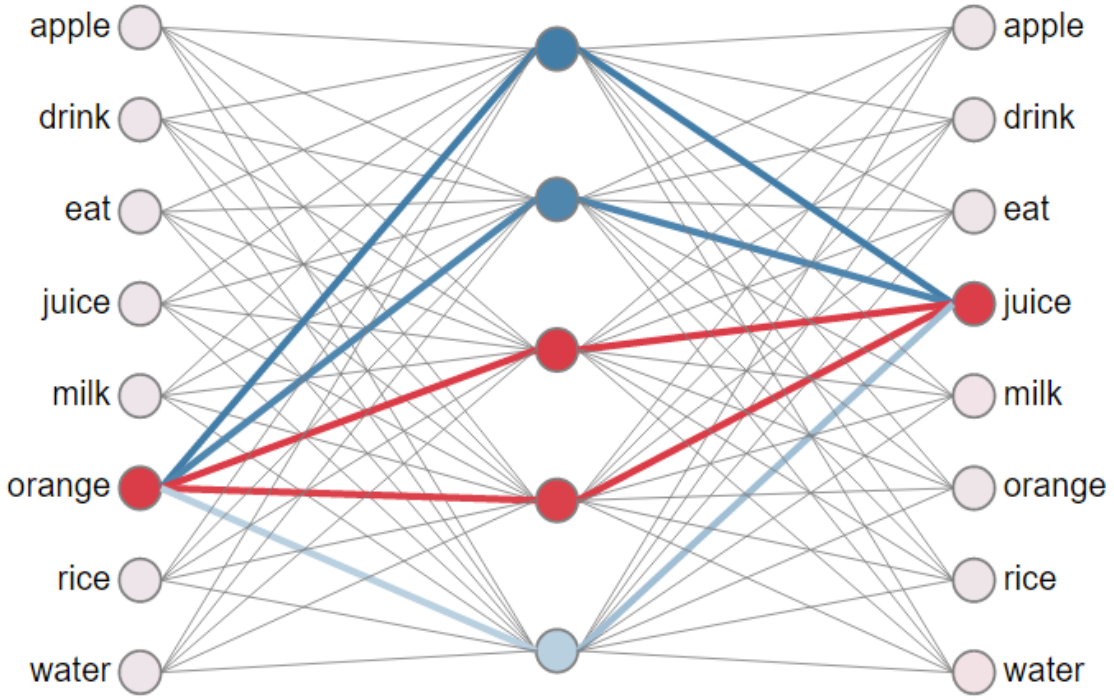


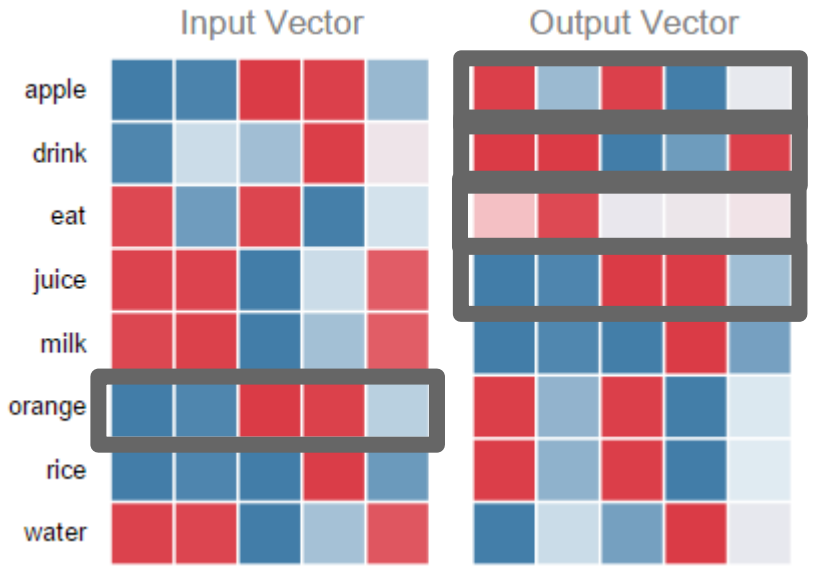
$$\mathbf{h} = \mathbf{x}^T\mathbf{W} := \mathbf{v}_{w_I}$$

$$u_j = \mathbf{v}'_{w_j}{}^T \cdot \mathbf{h}$$

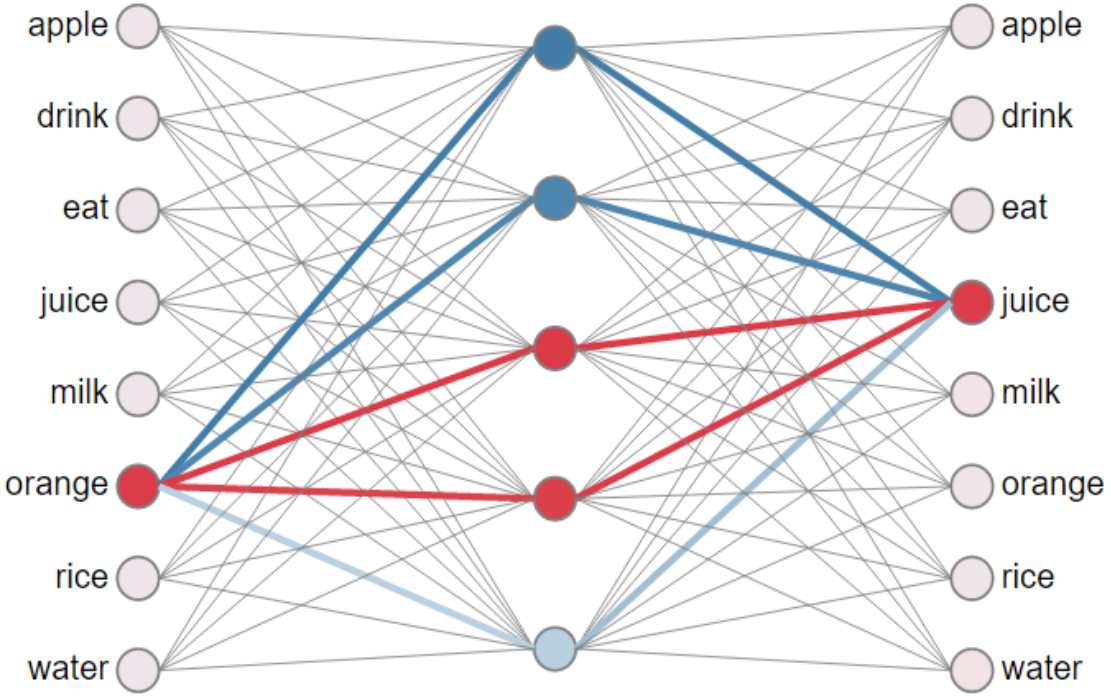
$$p(w_j|w_I) = \frac{\exp\left(\mathbf{v}'_{w_O}{}^T \mathbf{v}_{w_I}\right)}{\sum_{j'=1}^{V} \exp\left(\mathbf{v}'_{w'_j}{}^T \mathbf{v}_{w_I}\right)}$$

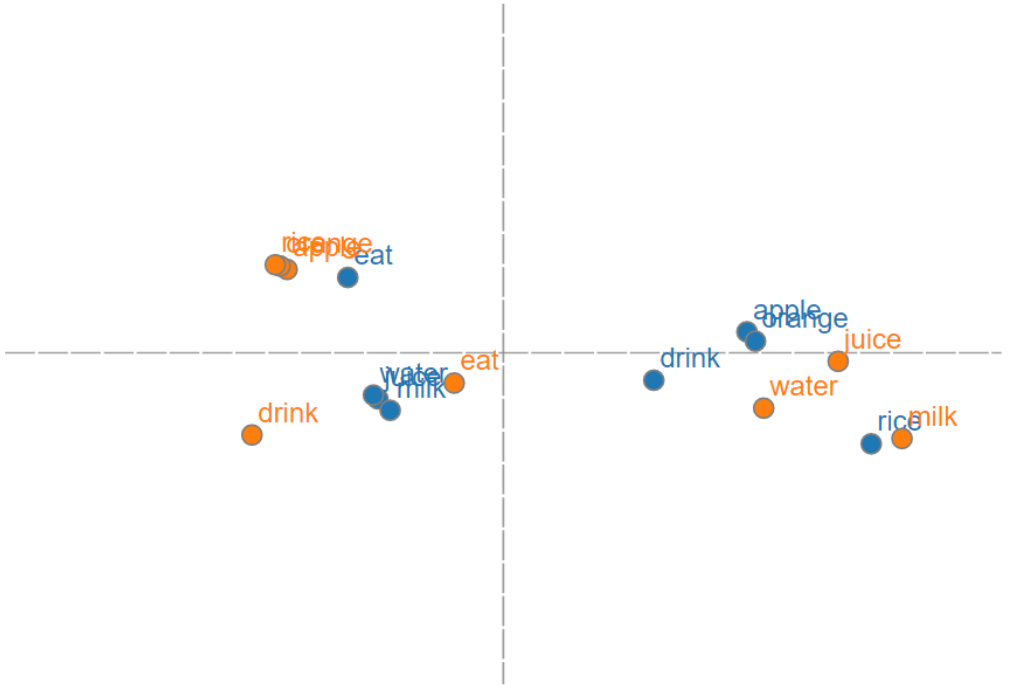
NEURAL NETWORK AND WEIGHT MATRICES



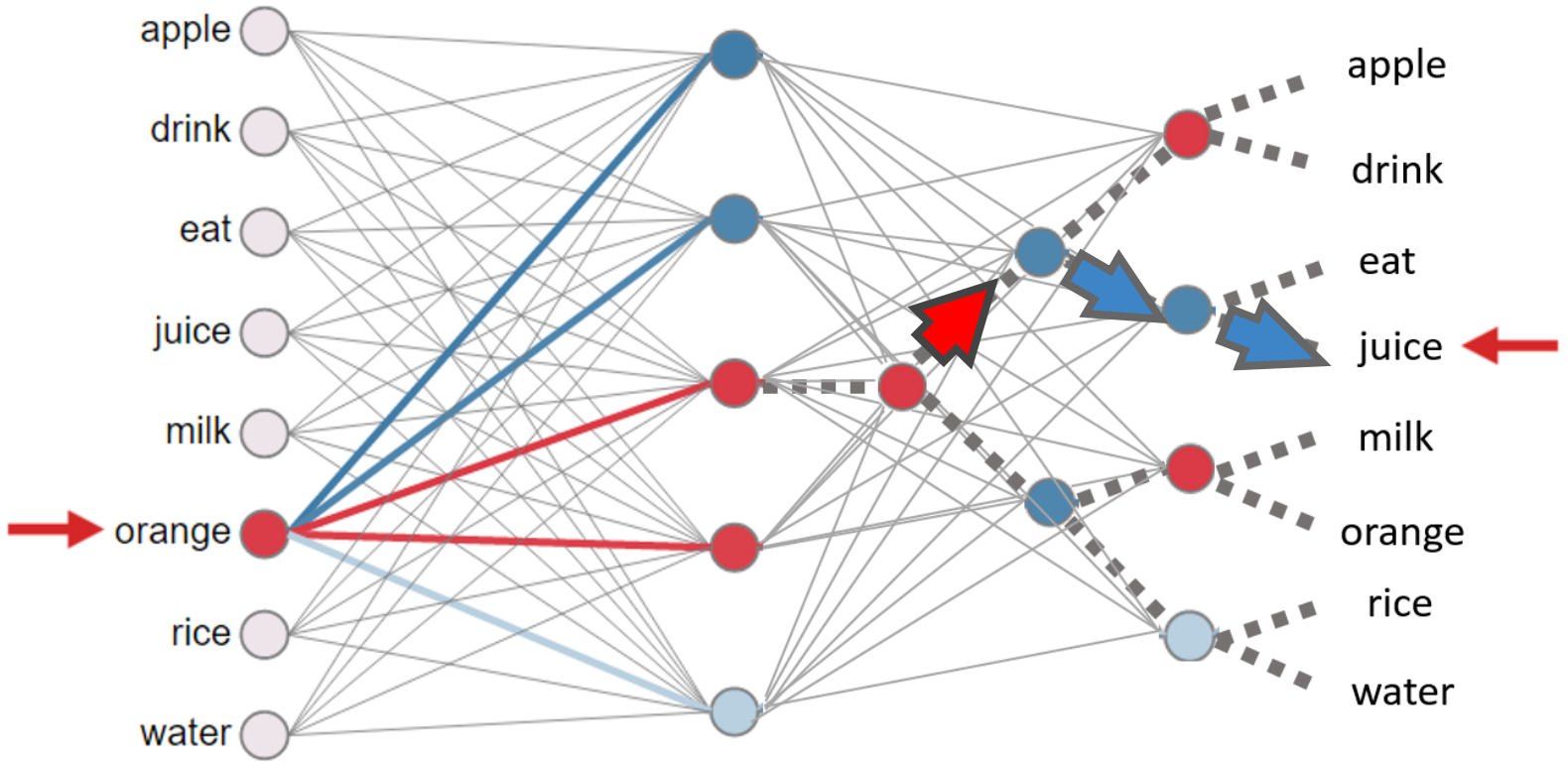


NEURAL NETWORK AND WORD VECTORS

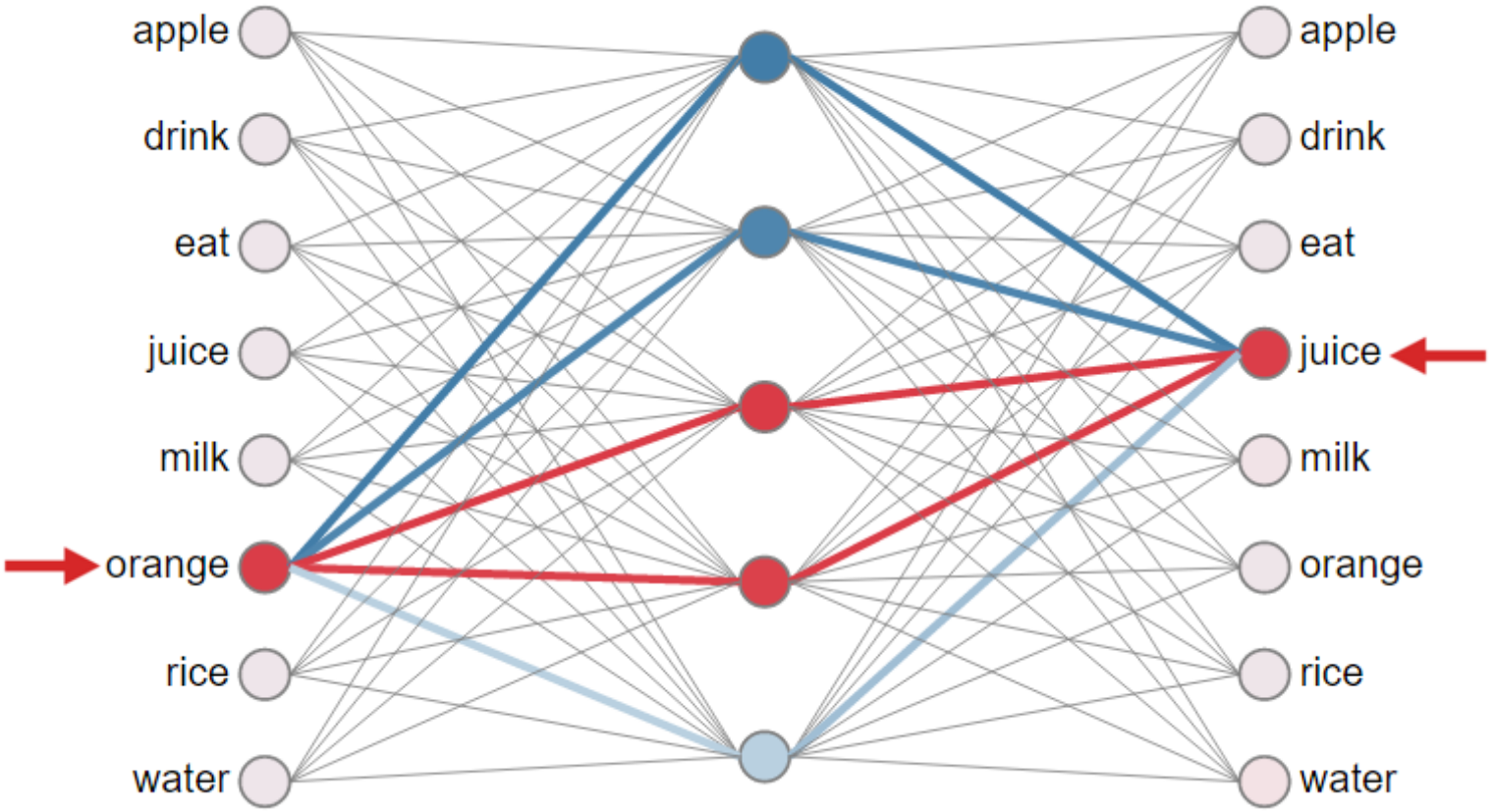




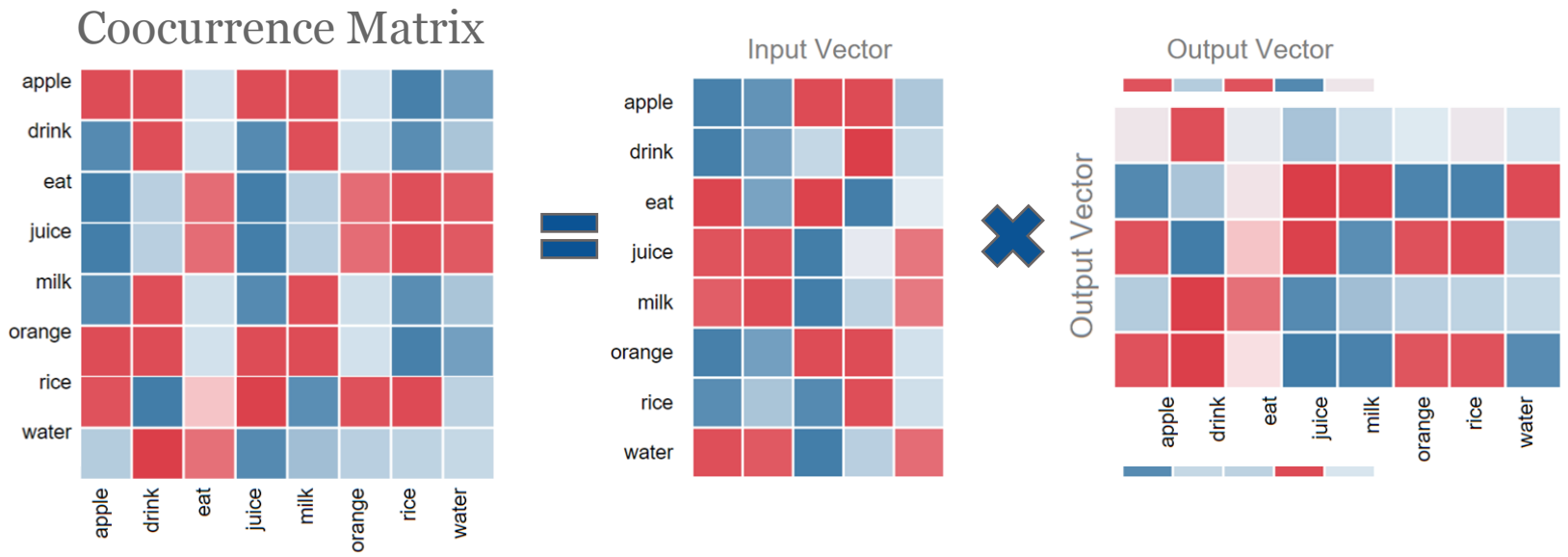
HIERARCHICAL SOFTMAX



NEGATIVE SAMPLING



INTERPRETING WORD EMBEDDING MODEL



REFERENCE

Tomas Mikolov, Wen-tau Yih, Geoffrey Zweig:
Linguistic Regularities in Continuous Space Word Representations. HLT-NAACL 2013: 746-751

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, Jeffrey Dean:
Distributed Representations of Words and Phrases and their Compositionality. NIPS 2013: 3111-3119

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, Christian Janvin:
A Neural Probabilistic Language Model. Journal of Machine Learning Research 3: 1137-1155 (2003)

David E Rumelhart, Geoffrey E Hintont, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Wevi Demo: http://ronxin.github.io/wevi/