Word Embedding

1. How do we represent the meaning of a word?

Definition: **meaning** (Webster dictionary)

- the idea that is represented by a word, phrase, etc.
- the idea that a person wants to express by using words, signs, etc.
- the idea that is expressed in a work of writing, art, etc.

Commonest linguistic way of thinking of meaning:

- signifier ⟺ signified (idea or thing) = denotation

“意思”的意思?

她说：“他这个人怪有**意思（funny）**。”于是人们以为他们有了**意思（wish）**，并让他向她意思**意思（express）**。他火了：“我根本没有那个**意思（thought）**！”她也生气了：“你们这么说是什么**意思（intention）**？”事后有人说：“真**有意思（funny）**。”也有人说：“真**没意思（nonsense）**”

吴尉天，1999］——《统计自然语言处理》

How do we have usable meaning in a computer?

Common answer: Use a taxonomy like WordNet that has hypernyms (is-a) relationships and synonym sets

```
from nltk.corpus import wordnet as wn
panda = wn.synset('panda.n.01')
hyper = lambda s: s.hypernyms()
list(panda.closure(hyper))
```

```
[Synset('procyonid.n.01'),
Synset('carnivore.n.01'),
Synset('placental.n.01'),
Synset('mammal.n.01'),
Synset('vertebrate.n.01'),
Synset('chordate.n.01'),
Synset('animal.n.01'),
Synset('organism.n.01'),
Synset('living_thing.n.01'),
Synset('whole.n.02'),
Synset('object.n.01'),
Synset('physical_entity.n.01'),
Synset('entity.n.01')]
```

(here, for *good*):

```
S: (adj) full, good
S: (adj) estimable, good, honorable, respectable
S: (adj) beneficial, good
S: (adj) good, just, upright
S: (adj) adept, expert, good, practiced,
proficient, skillful
S: (adj) dear, good, near
S: (adj) good, right, ripe
...
S: (adv) well, good
S: (adv) thoroughly, soundly, good
S: (n) good, goodness
S: (n) commodity, trade good, good
```

Problems with this discrete representation

- Great as a resource but missing nuances, e.g., **synonyms**:
  - adept, expert, good, practiced, proficient, skillful?
- Missing new words (impossible to keep up to date): wicked, badass, nifty, crack, ace, wizard, genius, ninja
- Subjective
- Requires human labor to create and adapt
- Hard to compute accurate word similarity →

Problems with this discrete representation

The vast majority of rule-based **and** statistical NLP work regards words as atomic symbols: hotel, conference, walk

In 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]

Dimensionality: 20K (speech) – 50K (PTB) – 500K (big vocab) – 13M (Google 1T)

We call this a "one-hot" representation

It is a **localist** representation

From symbolic to distributed 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"
- If user searches for [Seattle motel], we would like to match documents containing "Seattle hotel"

But

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

hotel $[0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0] = 0$

Our query and document vectors are **orthogonal**

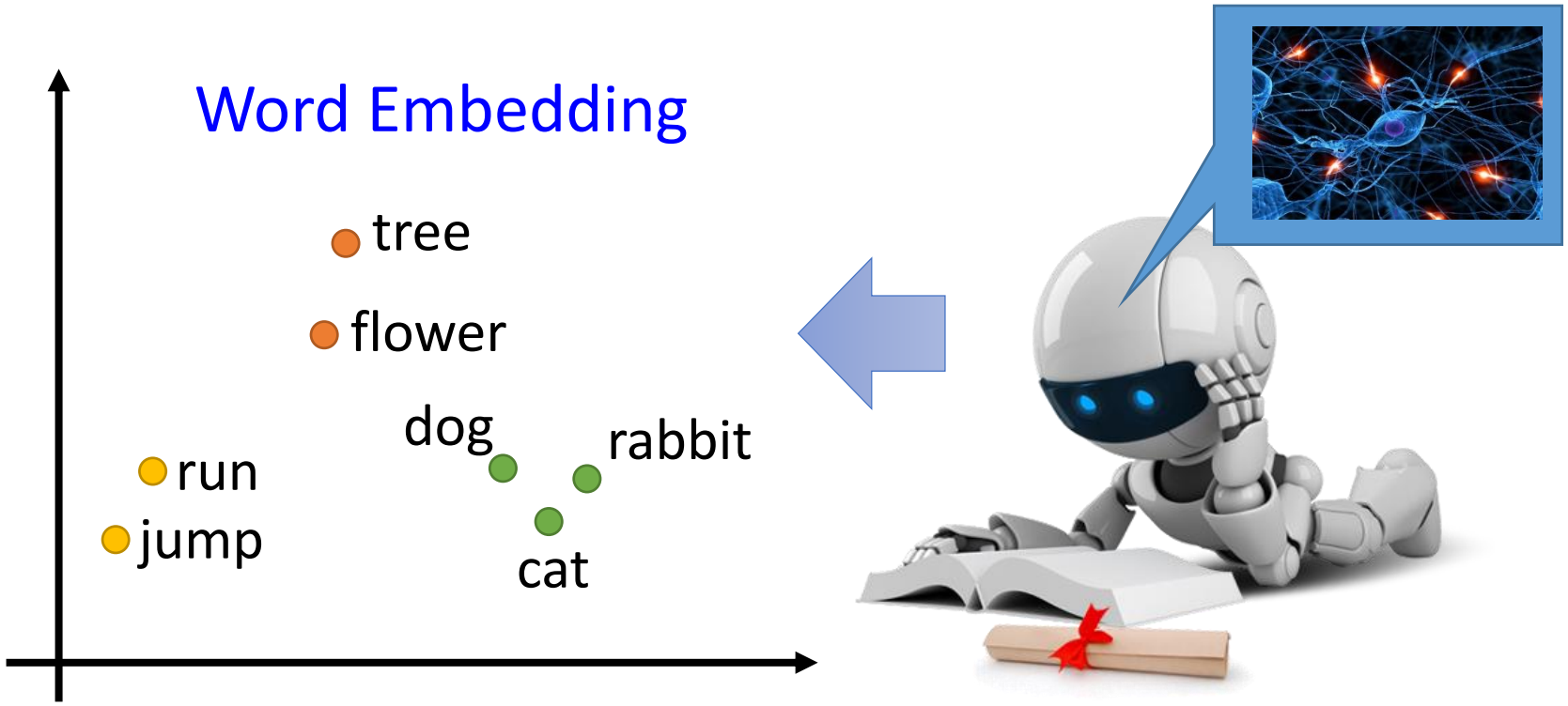
There is no natural notion of similarity in a set of one-hot vectors

Could deal with similarity separately;

instead we explore a direct approach where vectors encode it

Word Embedding

- Machine learns the meaning of words from reading a lot of documents without supervision



1-of-N Encoding

apple = [1 0 0 0 0]

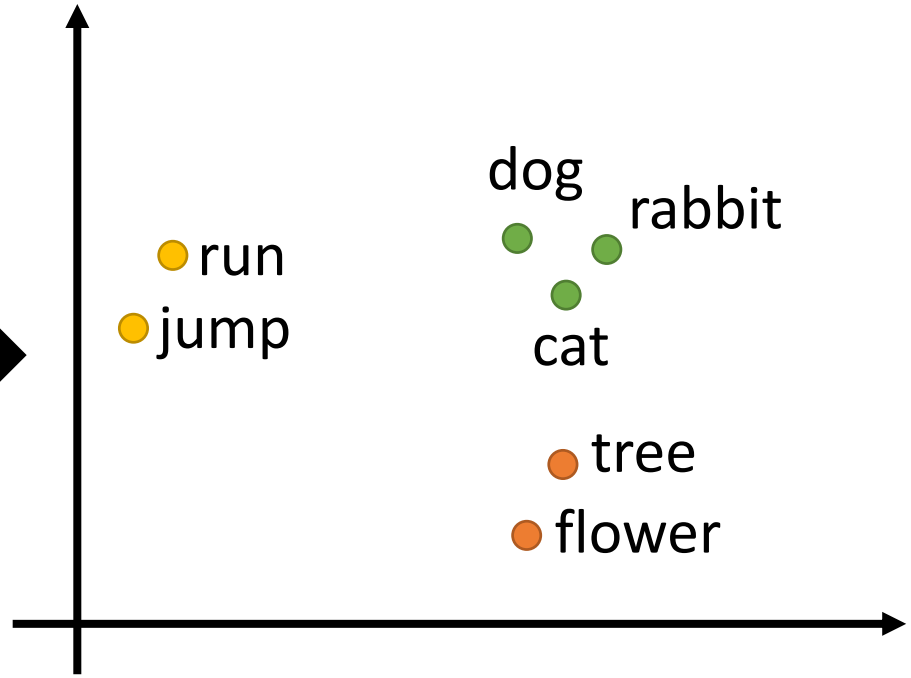
bag = [0 1 0 0 0]

cat = [0 0 1 0 0]

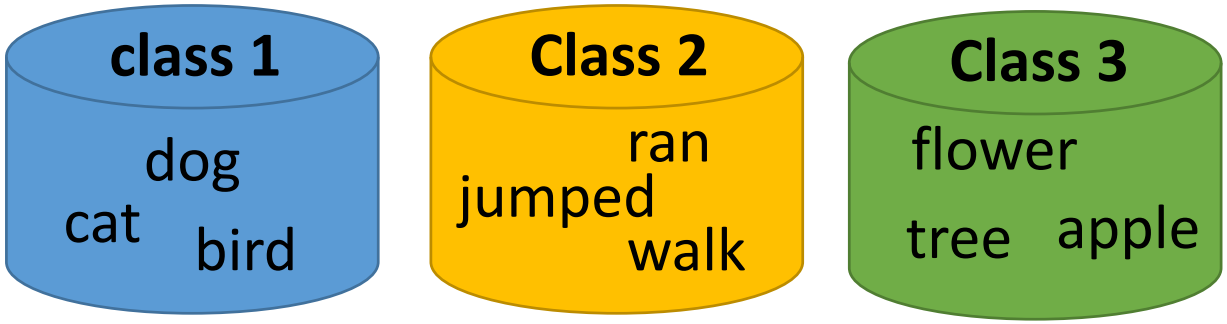
dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

Word Embedding



Word Class



Word Embedding

- Machine learns the meaning of words from reading a lot of documents without supervision
- **A word can be understood by its context**

蔡英文、马英九 are something very similar

You shall know a word by the company it keeps

马英九520宣誓就职

蔡英文 520宣誓就职

How to exploit the context?

- **Count based**
  - If two words w_i and w_j frequently co-occur, $V(w_i)$ and $V(w_j)$ would be close to each other
  - E.g. Glove Vector: http://nlp.stanford.edu/projects/glove/

$V(w_i) . V(w_j)$ ⟷ $N_{i,j}$

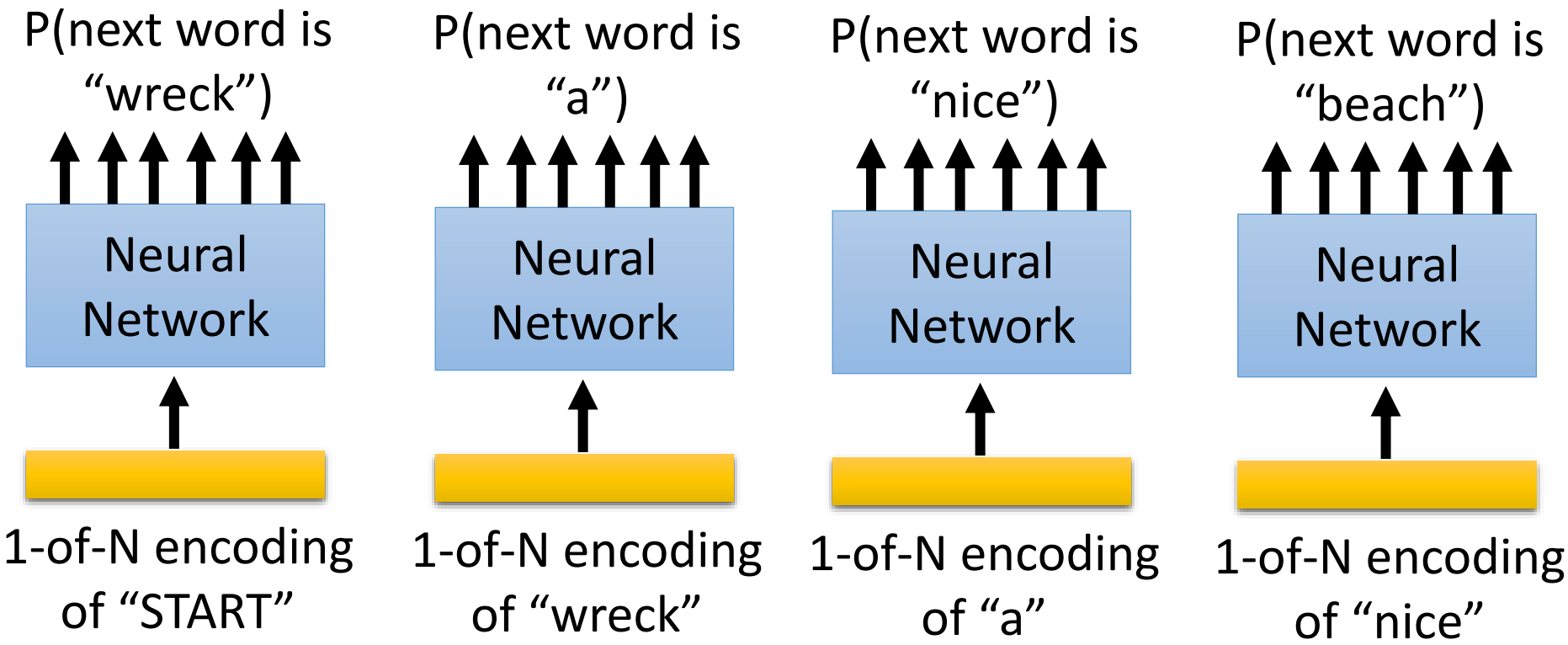
Inner product

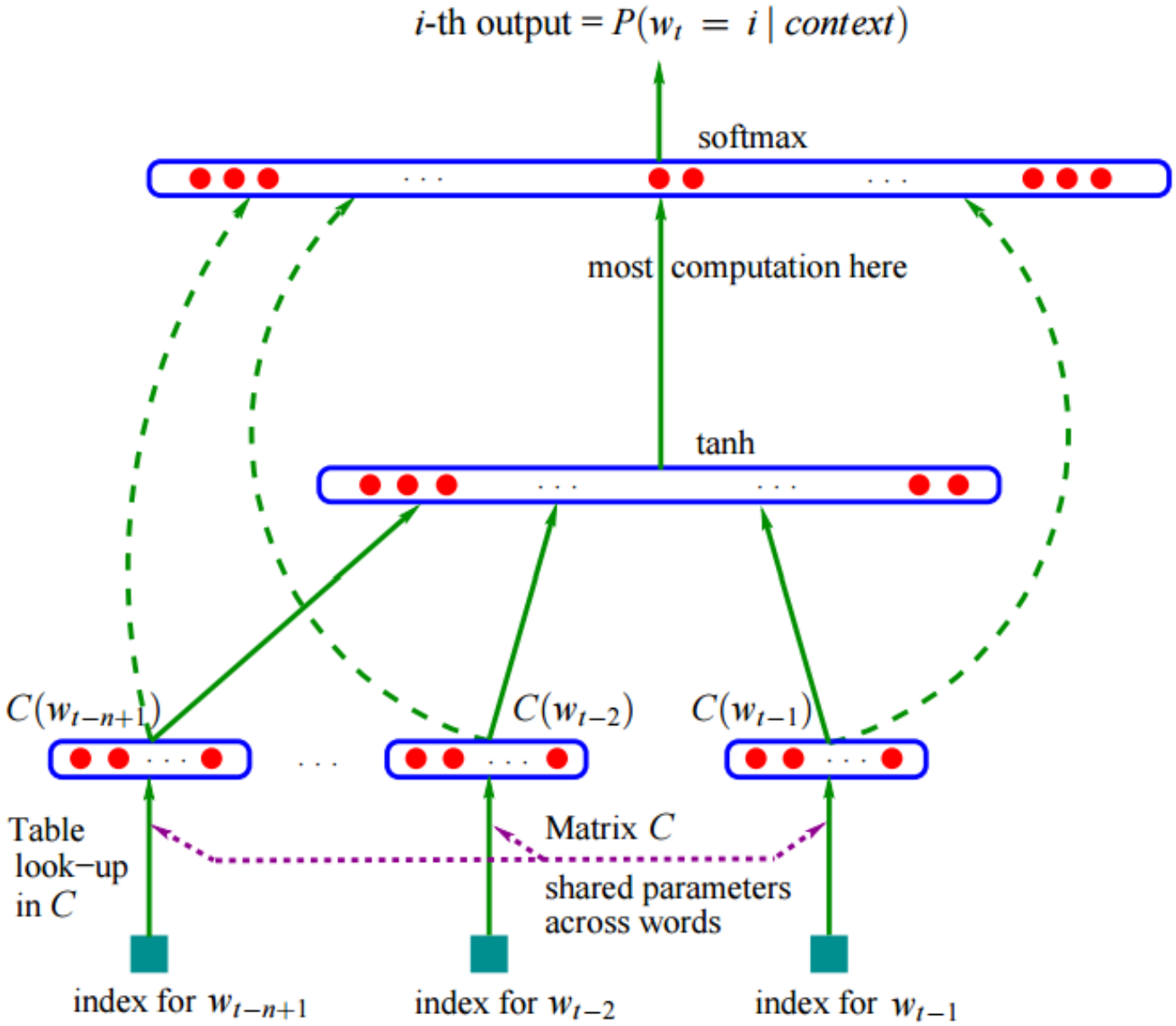
Number of times w_i and w_j in the same document

- **Prediction based**

Prediction-based – Language Modeling

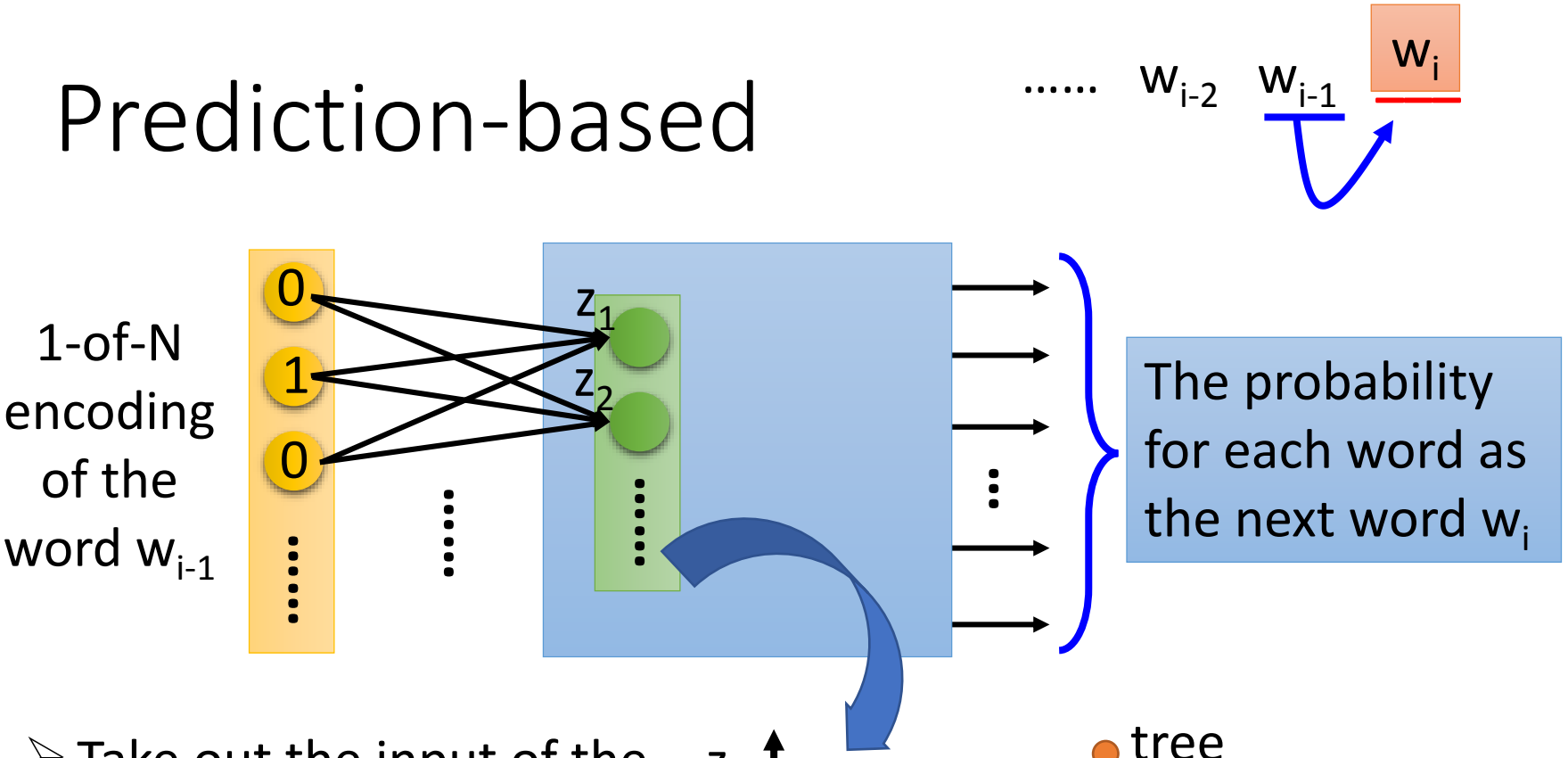
P("wreck a nice beach")
=P(wreck|START)P(a|wreck)P(nice|a)P(beach|nice)
P(b|a): the probability of NN predicting the next word.



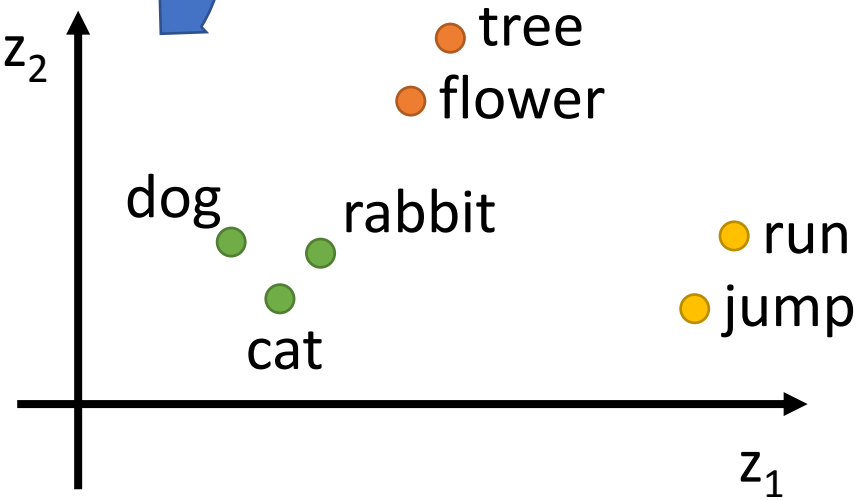


Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(Feb), 1137-1155.

Prediction-based

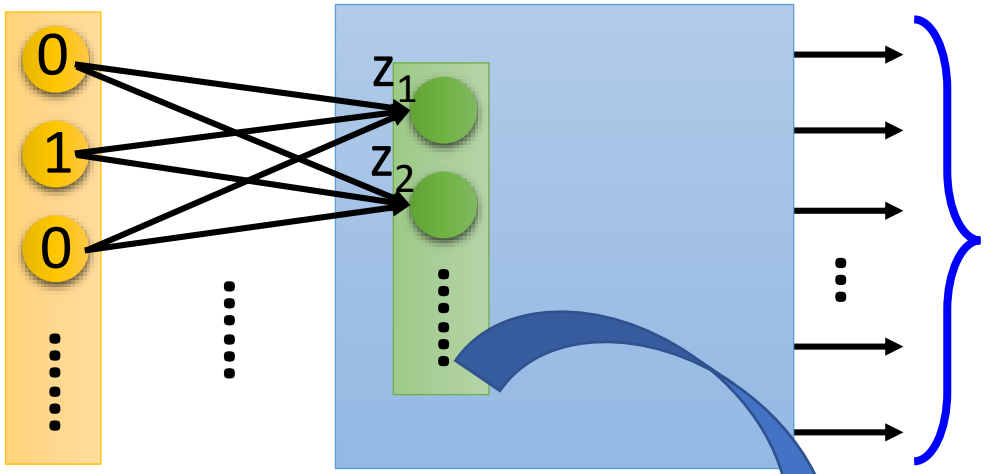


- Take out the input of the neurons in the first layer
- Use it to represent a word w
- Word vector, word embedding feature: V(w)



Prediction-based

You shall know a word by the company it keeps



蔡英文
or
馬英九

The probability for each word as the next word w_i

"宣誓就职" should have large probability

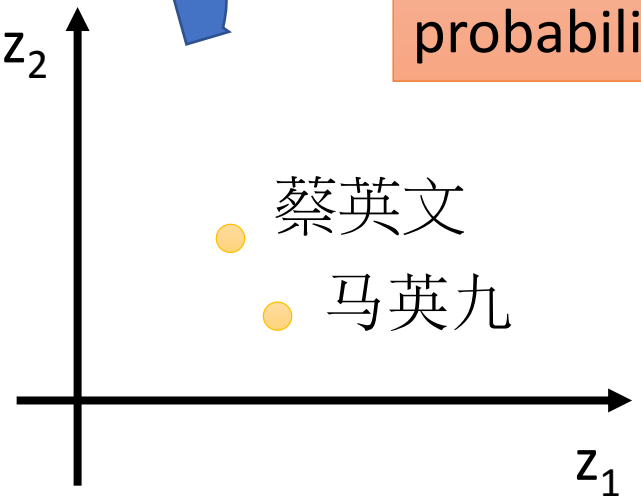
Training text:

……蔡英文 宣誓就职……

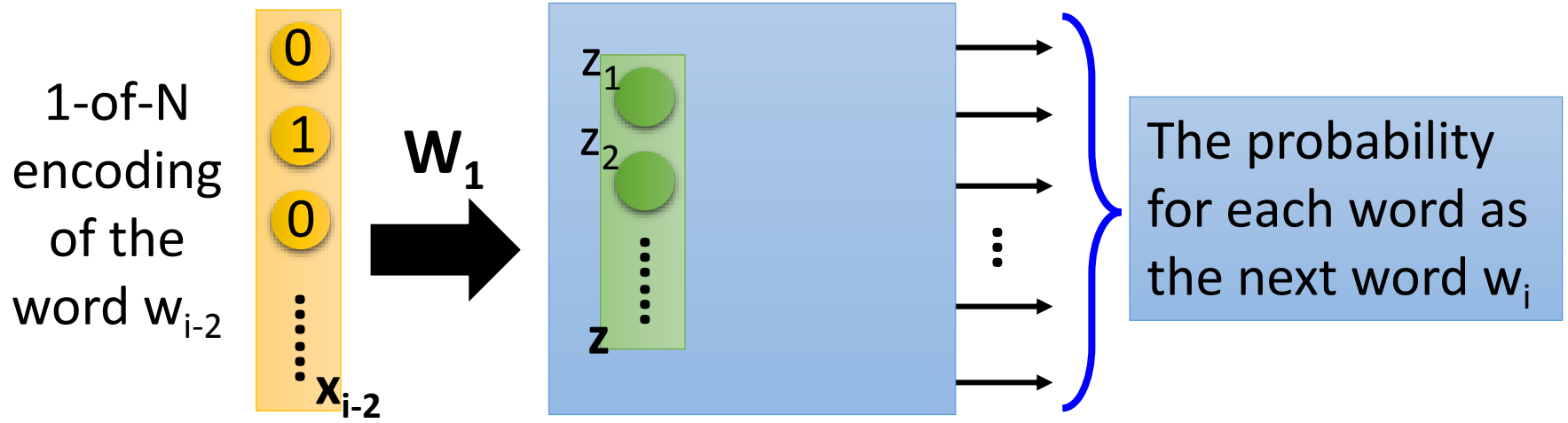
w_{i-1} w_i

……马英九 宣誓就职……

w_{i-1} w_i



Prediction-based – Sharing Parameters



The length of $\mathbf{x}_{i-1}$ and $\mathbf{x}_{i-2}$ are both $|V|$.

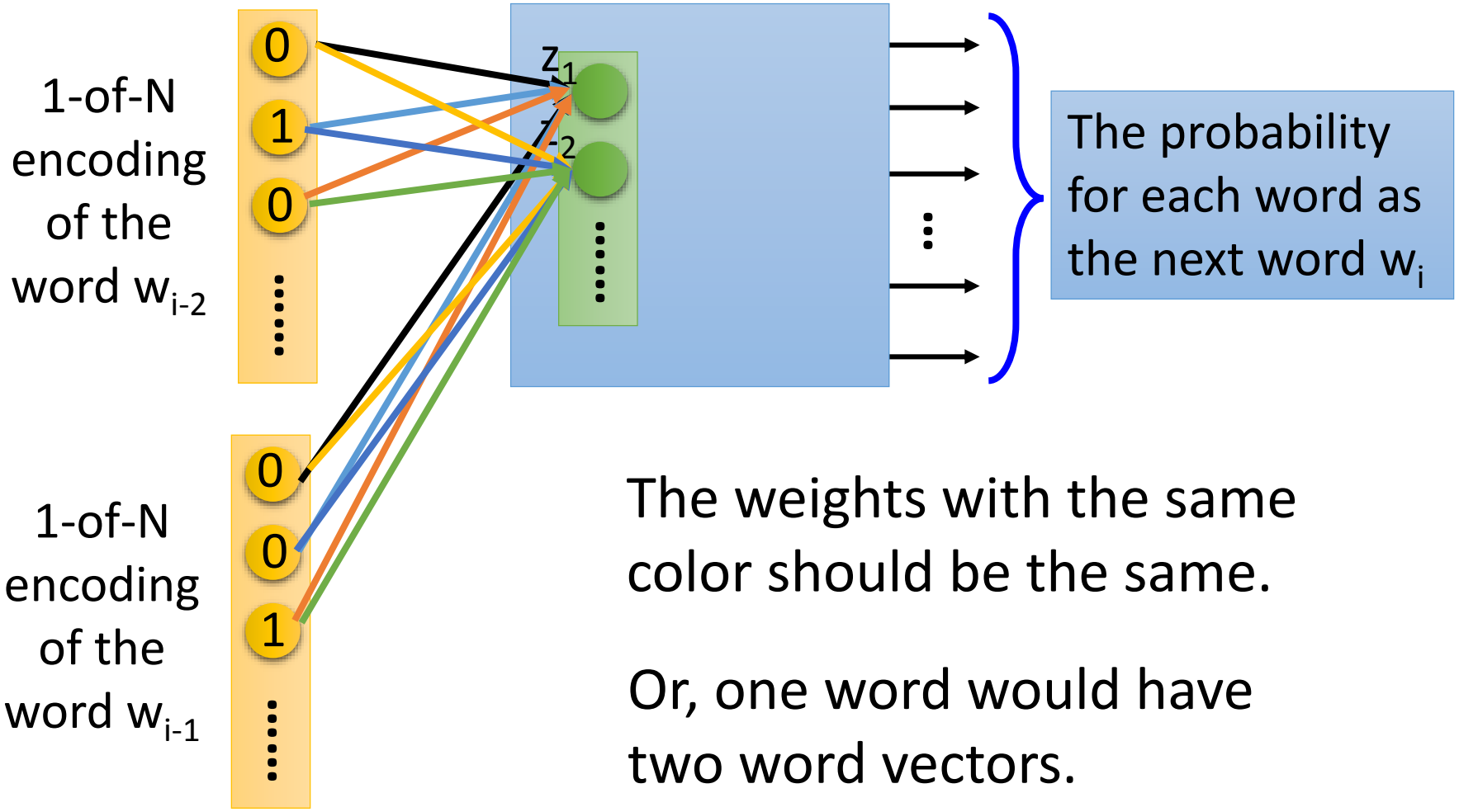
The length of $\mathbf{z}$ is $|Z|$.

$$\mathbf{z} = \mathbf{W}_1 \mathbf{x}_{i-2} + \mathbf{W}_2 \mathbf{x}_{i-1}$$

The weight matrix $\mathbf{W}_1$ and $\mathbf{W}_2$ are both $|Z|X|V|$ matrices.

$$\mathbf{W}_1 = \mathbf{W}_2 = \mathbf{W} \Rightarrow \mathbf{z} = \mathbf{W}(\mathbf{x}_{i-2} + \mathbf{x}_{i-1})$$

Prediction-based – Sharing Parameters

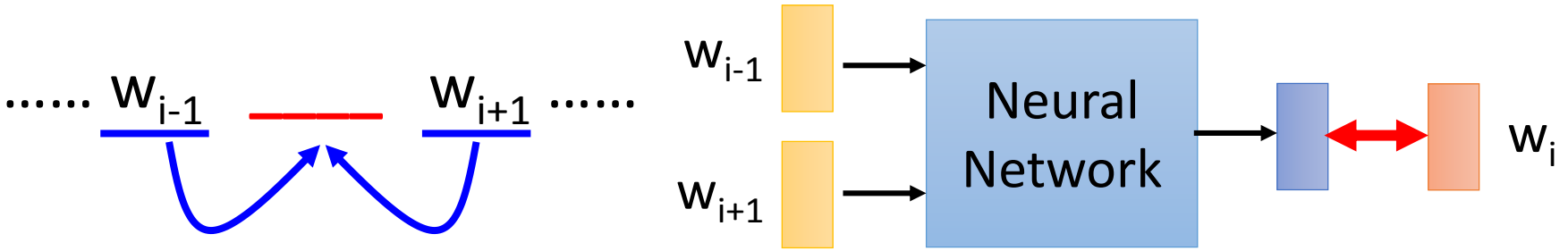


The weights with the same color should be the same.

Or, one word would have two word vectors.

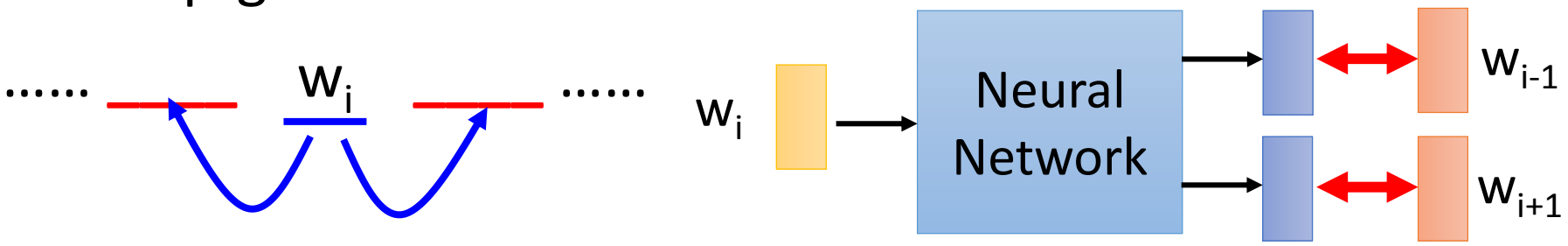
Prediction-based – Various Architectures

- Continuous bag of word (CBOW) model



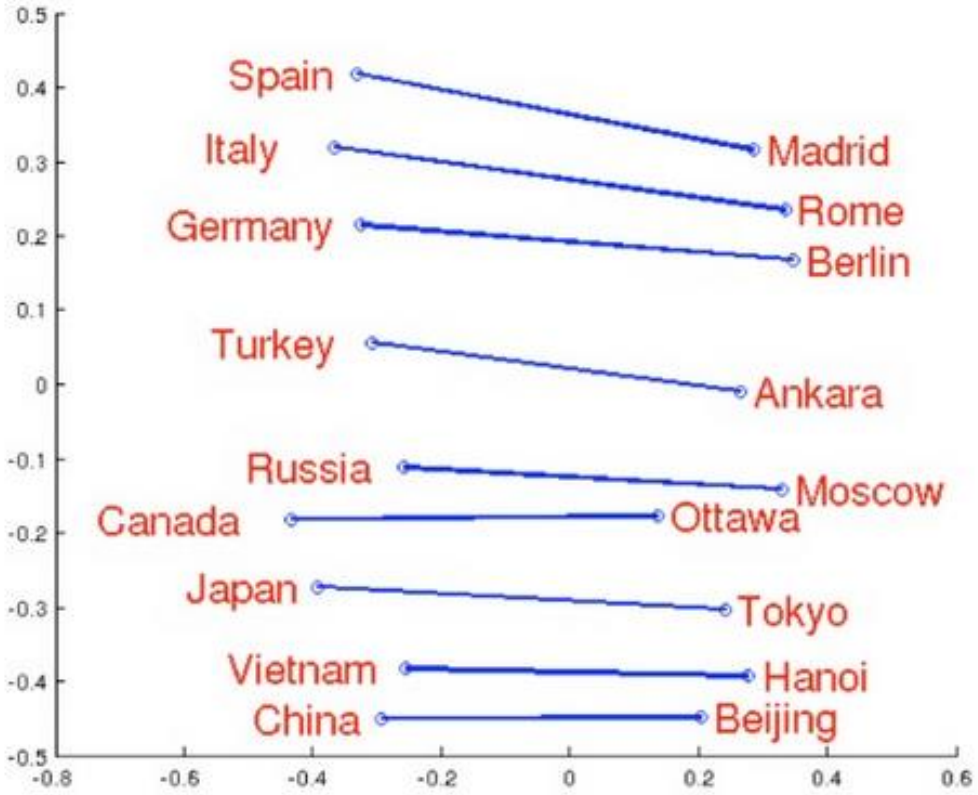
predicting the word given its context

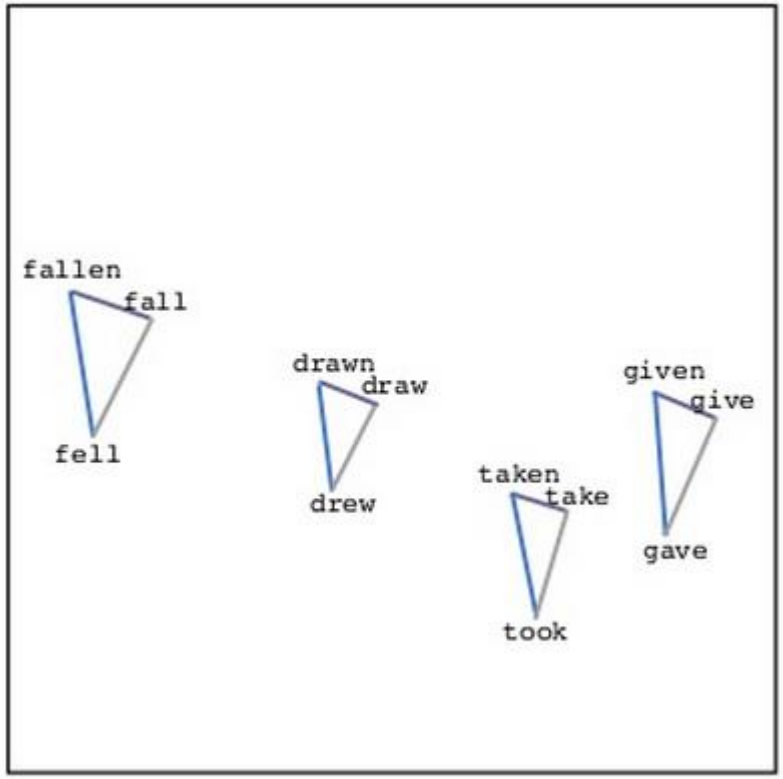
- Skip-gram



predicting the context given a word

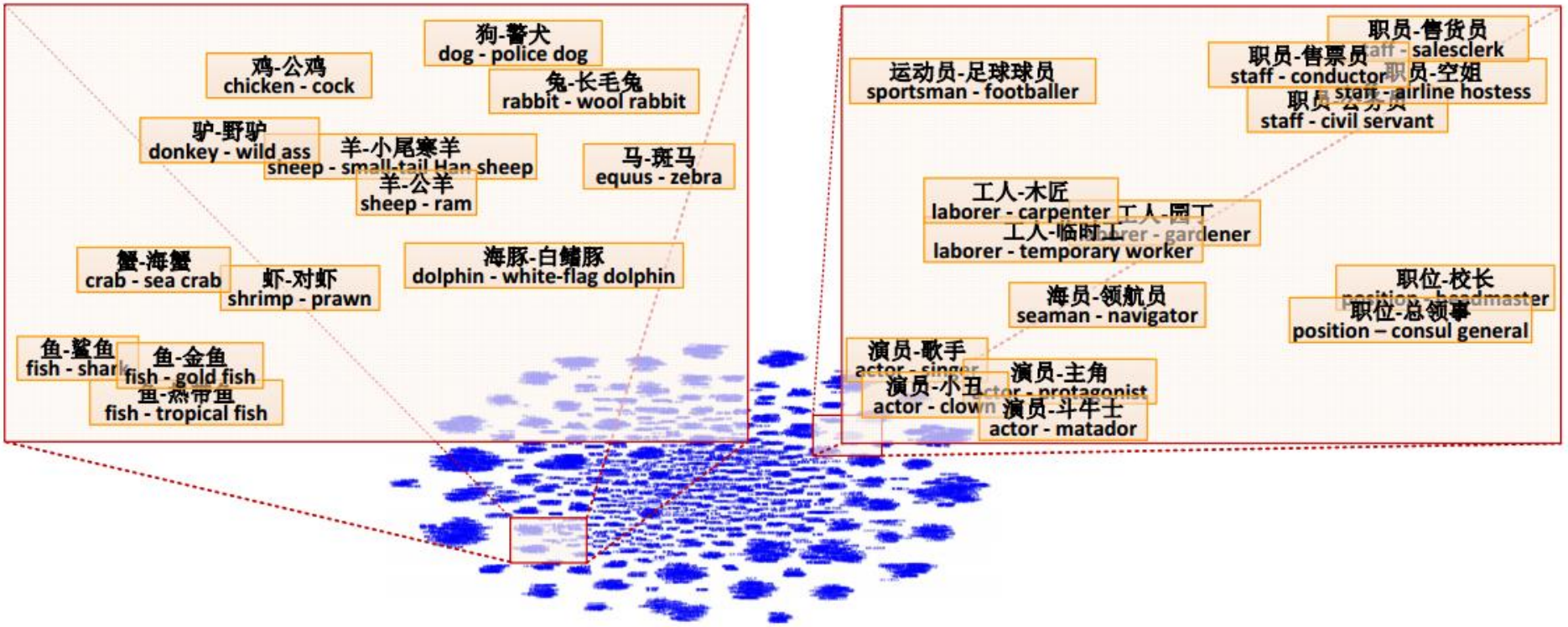
Word Embedding





Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

Word Embedding



Fu, Ruiji, et al. "Learning semantic hierarchies via word embeddings."*Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics: Long Papers*. Vol. 1. 2014.

Word Embedding

- Characteristics

$$V(Germany) \approx V(Berlin) - V(Rome) + V(Italy)$$

$$V(hotter) - V(hot) \approx V(bigger) - V(big)$$

$$V(Rome) - V(Italy) \approx V(Berlin) - V(Germany)$$

$$V(king) - V(queen) \approx V(uncle) - V(aunt)$$

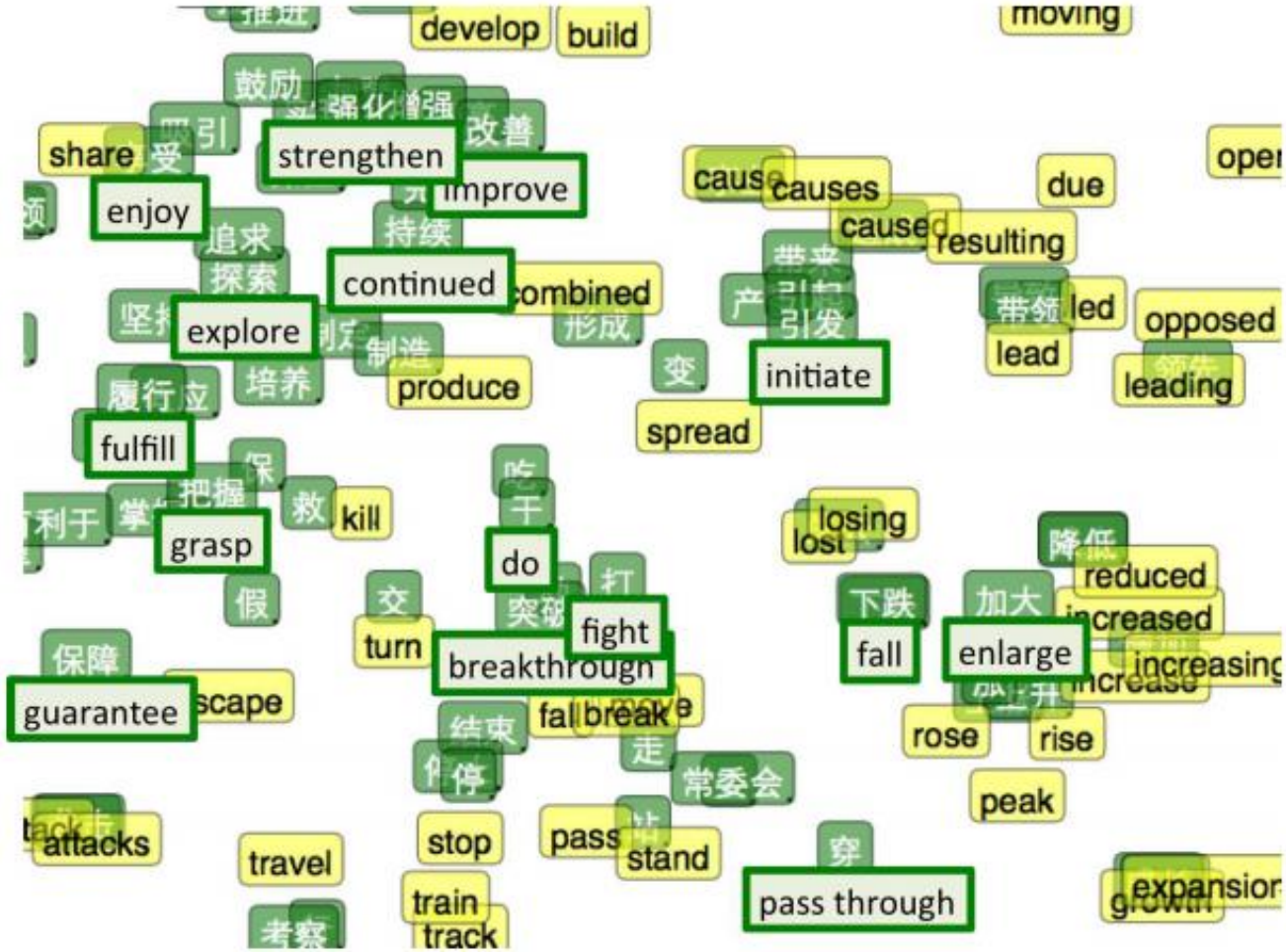
- Solving analogies

Rome : Italy = Berlin : ?

Compute $V(Berlin) - V(Rome) + V(Italy)$

Find the word w with the closest V(w)

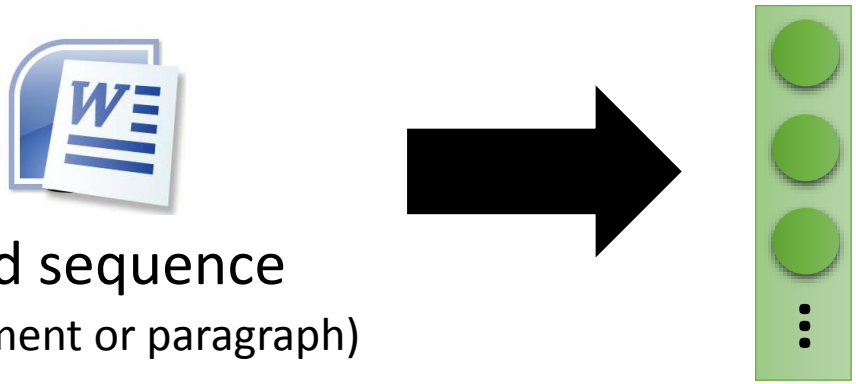
Multi-lingual Embedding



Bilingual Word Embeddings for Phrase-Based Machine Translation, Will Zou, Richard Socher, Daniel Cer and Christopher Manning, EMNLP, 2013

Beyond Bag of Word

- word sequences with different lengths → the vector with the same length
  - The vector representing the meaning of the word sequence
  - A word sequence can be a document or a paragraph



word sequence

(a document or paragraph)

Beyond Bag of Word

- To understand the meaning of a word sequence, the order of the words can not be ignored.