Word Embeddings (I)

CS 485, Fall 2024
Applications of Natural Language Processing

Brendan O'Connor
College of Information and Computer Sciences
University of Massachusetts Amherst

- Last week: Markov N-gram models
- Today: augment with word embeddings
  - 1. Markov model
  - 2. Skip-gram model

- Why?
  - Better LMs
  - Automatically learned word representations ("word embeddings") are interesting & can be used directly (continues Thursday)

Word embeddings

- Today
  - 1. Question: how can we generally represent word meanings?
  - 2. Approach: train a language model with ***word embeddings*** to discover latent meanings of words!
    - ... which exploit the ***distributional hypothesis***

- Key idea: automatically discover aspects of language meaning, from raw textual corpora

What is "asdfasdf"?

“ **asdfasdf**, Most Neglected American Fruit.” — NYTimes 1922

“ **asdfasdf** Recommended by U.S. Food Experts, Along With Persimmon, as War Nutrition” — NYTimes 1942

“ The **asdfasdf** is also pollinated by flies and other insects rather than by honeybees…”— NYTimes 2020

“Many people also cook with ripe **asdfasdf**, making bread, beer, ice cream, or this **asdfasdf** pudding…” — NYTimes 2020

What is a *pawpaw* ?

I. Look it up in a dictionary

https://www.merriam-webster.com/

https://www.oed.com/

https://en.wiktionary.org/

pawpaw noun

Save Word

paw·paw

variants: *or less commonly* **papaw**

Definition of *pawpaw*

1 \ pə-ˈpȯ \ **:** PAPAYA

2 \ ˈpä-(ˌ)pȯ , ˈpȯ- \ **:** a North American tree (*Asimina triloba*) of the custard-apple family with purple flowers and an edible green-skinned fruit

also **:** its fruit

Lemma
pawpaw noun
Save Word
paw·paw
variants: or less commonly papaw
Definition of pawpaw
1 \ pə-ˈpȯ \ : PAPAYA
2 \ ˈpä-(ˌ)pȯ, ˈpȯ- \ : a North American tree (Asimina triloba) of the custard-apple family with purple flowers and an edible green-skinned fruit
also : its fruit
Word
Senses
Definition

II. Look it at how its used

“ **Pawpaw**, Most Neglected American Fruit.” — NYTimes 1922

“ **Pawpaw** Recommended by U.S. Food Experts, Along With Persimmon, as War Nutrition” — NYTimes 1942

“ The **pawpaw** is also pollinated by flies and other insects rather than by honeybees…”— NYTimes 2020

“Many people also cook with ripe **pawpaws**, making bread, beer, ice cream, or this **pawpaw** pudding…” — NYTimes 2020

II. Look it at how its used

“ *Pawpaw*, Most Neglected **American Fruit** .” — NYTimes 1922

“ *Pawpaw* Recommended by U.S. Food Experts, Along With **Persimmon** , as War **Nutrition** ” — NYTimes 1942

“ The *pawpaw* is also **pollinated** by **flies** and other insects rather than by honeybees…”— NYTimes 2020

“Many people also **cook** with **ripe** *pawpaws* , making **bread** , **beer**, **ice cream** , or this *pawpaw* **pudding** …” — NYTimes 2020

$[\![\text{car}]\!] \subset [\![\text{automobile}]\!]$?

Aspects of word meaning

other senses (i)

Synonyms

- couch / sofa
- oculist / eye - doctor
- car / automobile
- water / H_2O

cf. word choice

- draft / draught

Antonyms

- yes / no
- dark / light
- hot / cold
- up / down
- clip / clip

Aspects of word meaning

Similarity

- cat / dog
- cardiologist / pulmonologist
- car / bus
- sheep / goat
- glass / mug

Relatedness

- coffee / cup
- waiter / menu
- farm / cow
- house / roof
- theater / actor

Aspects of word meaning

- Connotation: the affective meaning of a word
- Osgood (1957)'s three-dimensional model:
  - Valence
    - unhappy, annoyed <---------------> happy, satisfied
  - Arousal
    - calm <---------------> excited
  - Dominance
    - awed, influences <---------------> controlling

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24

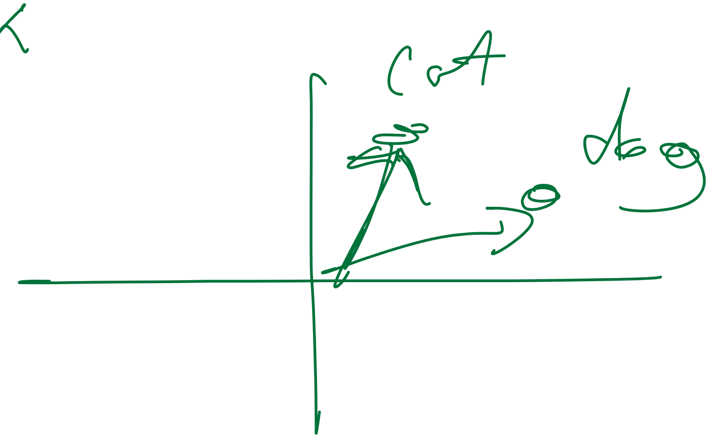
Word embeddings/vectors

- We need a representation of words capable of synonyms, rough similarity, or maybe even other aspects of meaning

$k=100$

- Give each word a k-dimensional **vector**
    - a vector is a list of numbers
    - a vector is a point/direction in k-dimensional space

$w \in V$: $x_w \in \mathbb{R}^k$



Learning word representations

- How to get word meanings?
  - Lexical resources like WordNet: dictionary-like databases of word synonyms & other word-to-word relationships, constructed manually
    - Can sometimes help, but typically don't cover all words or meanings any particular task needs

- OK, can we *learn* the word representations instead?

Distributional Semantics

"You shall know a word by the company it keeps!" — Firth (1957)

Intuitions: Harris (1954)

"If A and B have almost identical environments except chiefly sentences which contain both, we say they are synonyms: *oculist* and *eye- doctor* ."

Learning word representations

- Could we automatically *learn* word meanings?
  - 1. We'd like to generalize word meanings beyond individual words, and
  - 2. Information from nearby words gives information about a word
- What model have we seen, that uses information from nearby words to make inferences about another word?

N-gram LMs!

"road onto the"

- Two word-embedding-based LMs
    - 1. Markovian left-to-right LM (Bengio et al. 2003)
    - 2. "Skip-gram" LM
        - Learns useful standalone embeddings

Left-to-right LM as log. reg.

- Instead of only n-gram count ratios, model the next-word as softmax over the vocabulary.
- We can use anything to help predictions: features (Rosenfeld 1996) or neural networks (Bengio et al. 2003) to compose $\mathbf{v_u}$:

$\beta_w^T u_v$

e

$$p(w \mid u) = \frac{\exp(\boldsymbol{\beta}_w \cdot \boldsymbol{v}_u)}{\sum_{w' \in \mathcal{V}} \exp(\boldsymbol{\beta}_{w'} \cdot \boldsymbol{v}_u)} \qquad \begin{array}{l} \boldsymbol{\beta}_w \in \mathbb{R}^K \\ \boldsymbol{v}_u \in \mathbb{R}^K \end{array}$$

length V

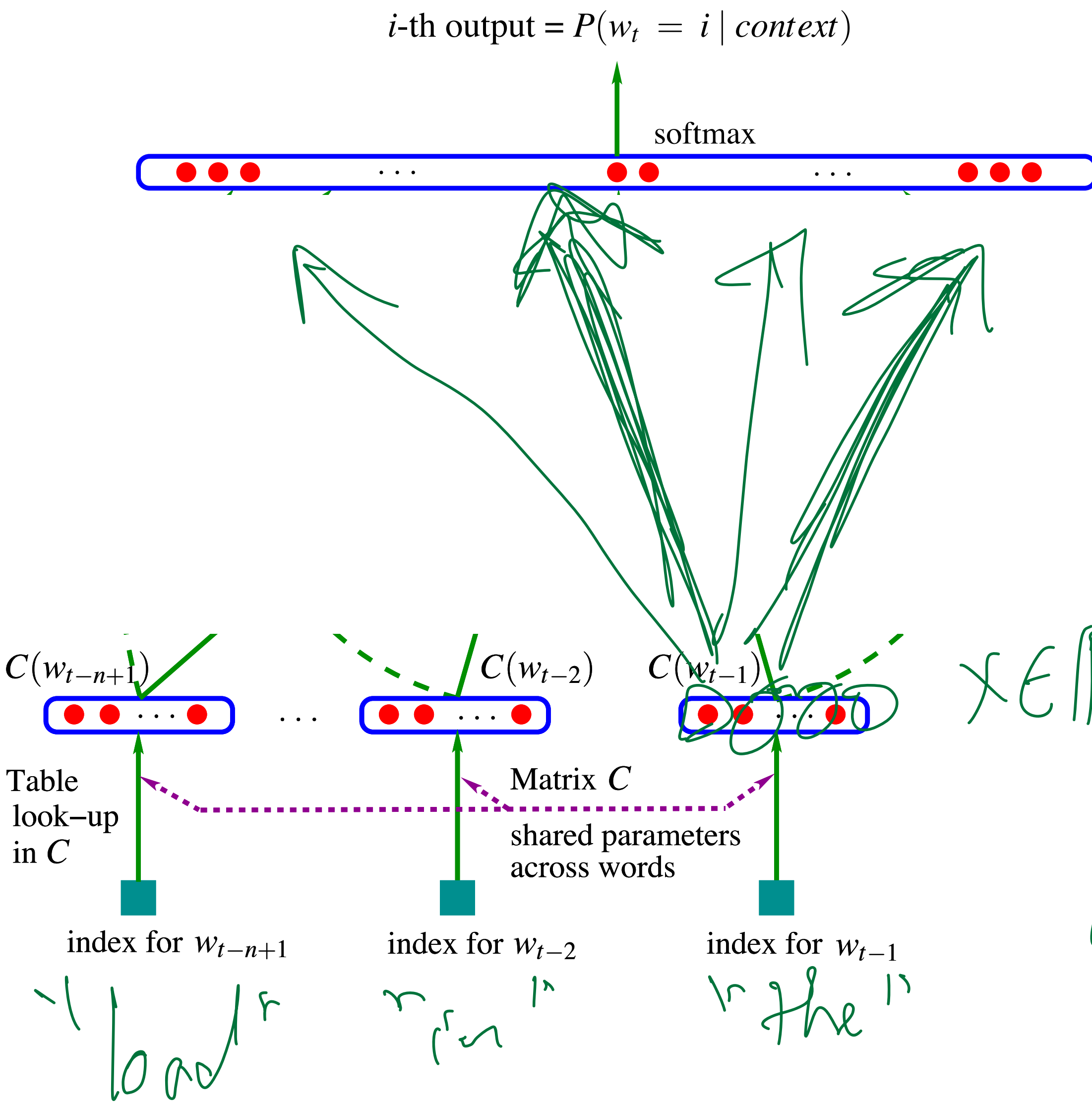
next word

ctx.

- Can use any information from the left context: long-distance topical information, or word vectors!

Bengio et al. 2003: Markov word embedding LM

Key idea: represent words on left as **vectors.** Learn a vector for each word in the vocabulary. Better perplexity than an n-gram LM!



Output layer (softmax)

$$\hat{P}(w_t \mid w_{t-1}, \cdots w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}.$$

$\in \mathbb{R}^V$

linear layer

(ignore today)
hidden layer, size h

$$y = \underbrace{b + Wx}_{\text{linear layer}} + \underbrace{U \tanh(d+Hx)}_{\text{hidden layer, size } h}$$

$x \in \mathbb{R}^{n \cdot k}$

$$x = (C(w_{t-1}), C(w_{t-2}), \cdots, C(w_{t-n+1}))$$

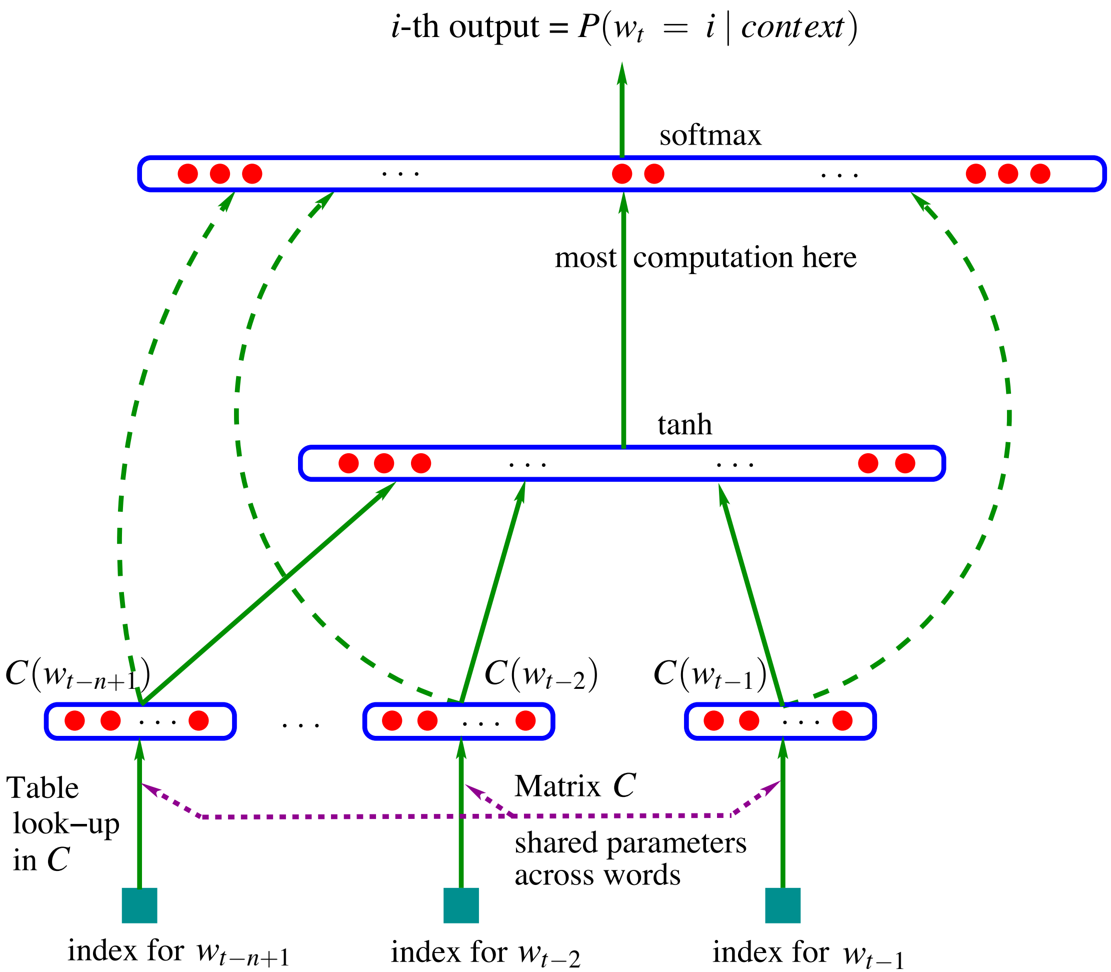
$C \in \mathbb{R}^{V \times k}$

Word vector lookup layer with concatenation

$C(i) \in \mathbb{R}^{\cancel{h}\,k}$ Word embedding parameters

Bengio et al. 2003: Markov word embedding LM

Key idea: represent words on left as **vectors.**
Learn a vector for each word in the vocabulary.



Output layer (softmax)

$$\hat{P}(w_t | w_{t-1}, \cdots w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}.$$

linear layer

(ignore today) hidden layer, size h

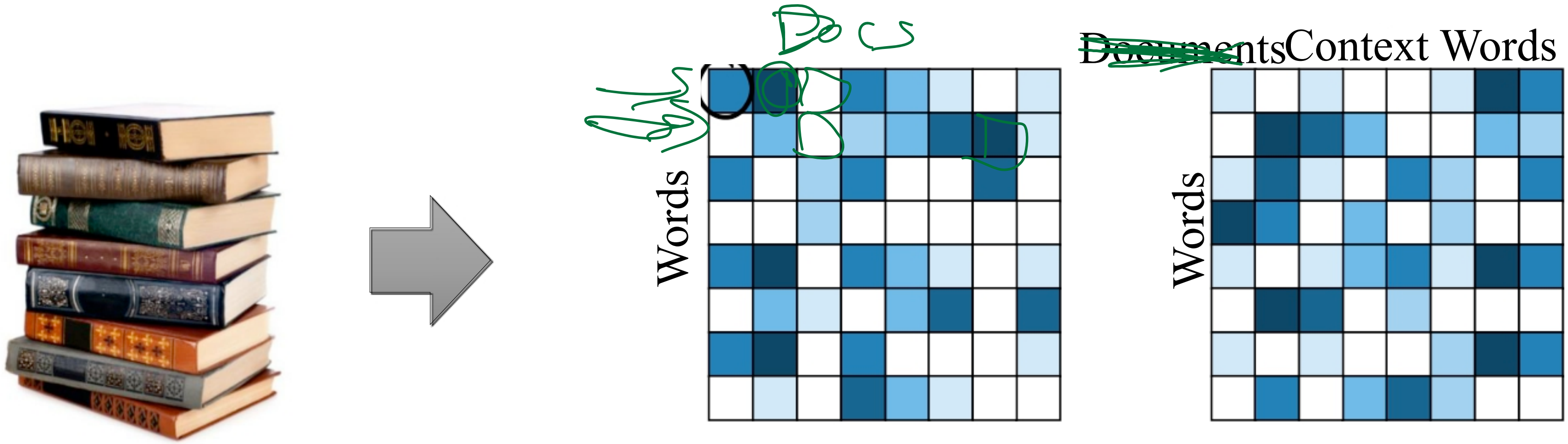
$$y = \underbrace{b + Wx}_{\text{linear layer}} + \underbrace{U \tanh(d+Hx)}_{\text{hidden layer, size } h}$$

$$x = (C(w_{t-1}), C(w_{t-2}), \cdots, C(w_{t-n+1}))$$

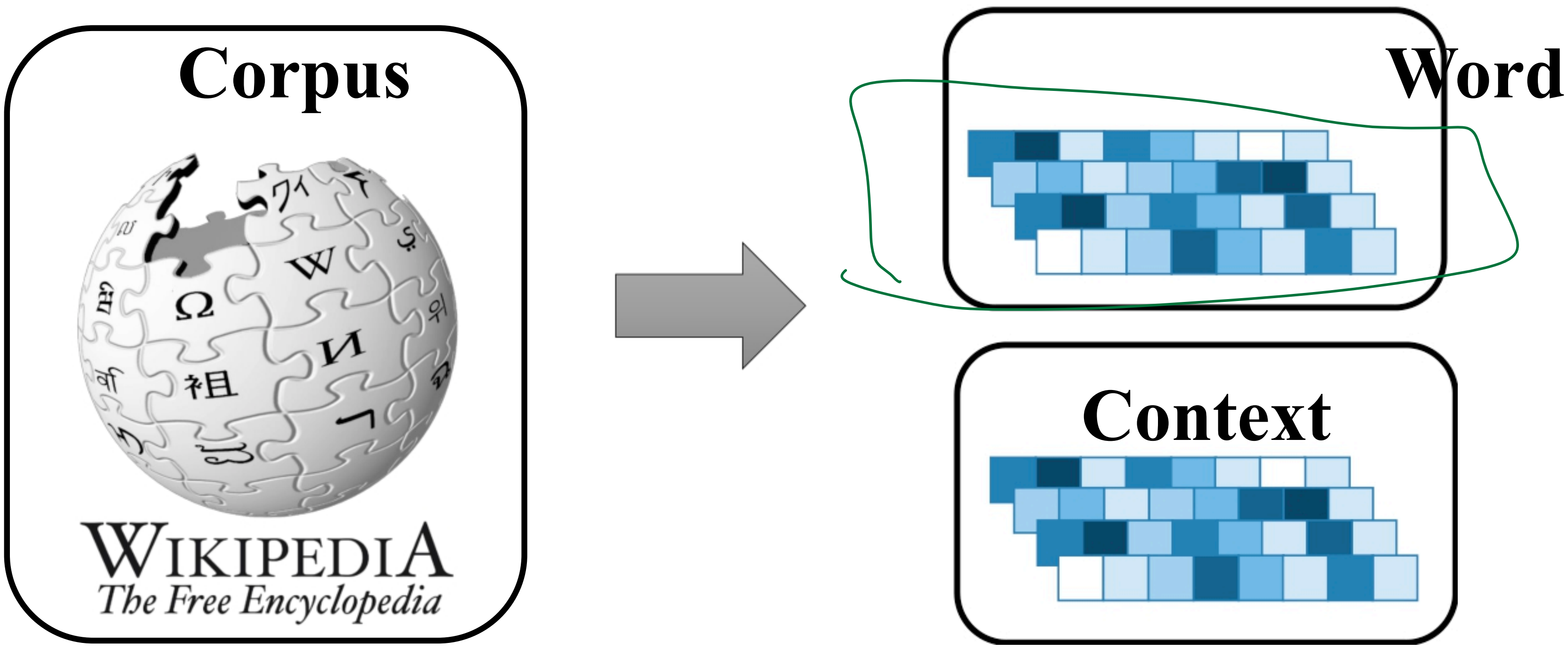
Word vector lookup layer with concatenation

$$C(i) \in \mathbb{R}^m$$ Word embedding parameters

Build vectors based on context



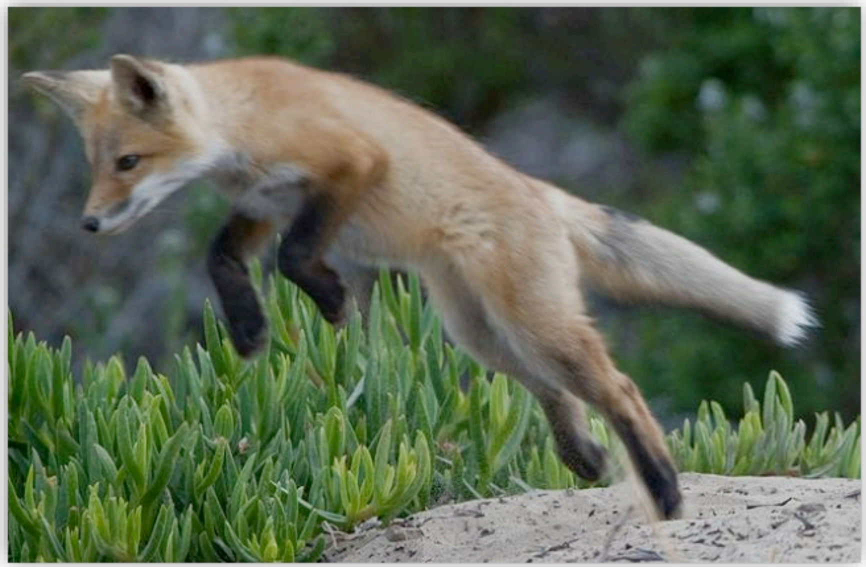
Neural Word Embeddings



Skip- Gram with Negative Sampling (SGNS)

"Word 2 Vec"

The brown fox **jumps** over the lazy dog



SG NS: Skip- Gram Model

context window size 2

The brown fox **jumps** over the lazy dog.

c c w c c

jumps → {brown, fox, over, the}

$P(c = \text{brown} \mid w = \text{jumps})$

<u>SG</u> NS: Skip- Gram Model

The brown fox **jumps** over the lazy dog.

Similar

leaps

Simple idea: from a word, predict its context words!
(A funny type of language model.)
Learn a vector that's good at that. Similar words should get similar vectors.

Key idea: use unlabeled text as *implicitly supervised data*

- A word *s* near *apricot*
  - Acts as gold 'correct answer' to the question
  - "Is word *w* likely to show up near *apricot*?"
- No need for hand-labeled supervision
- The idea comes from **neural language modeling**
  - Bengio et al. (2003)
  - Collobert et al. (2011)

Modeling goal

- Given a (word, context) tuple
  - [+] (apricot, jam) <- observed
  - [–] (apricot, aardvark) <- unseen
- Want binary probability
  - P(c | w) for a real context [+])
  - 1-P(c | w) for a "fake", unseen context [–])
- Let u_t and v_c be their vectors.
- P(c | w) = **σ**(u_w'v_c): logistic in their *affinity/similarity*
- Maximize P(c | w) for all (w, c) pairs

U_w: target word vec

V_c: ctx. word vec

$\sigma(U_{apricot}^T V_{jam}) \rightarrow$ want near 1

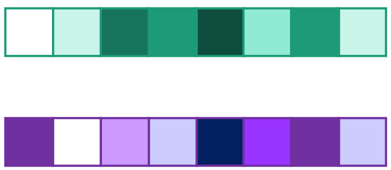
$\sigma(U_{apricot}^T V_{aardvark}) \rightarrow$ want near 0

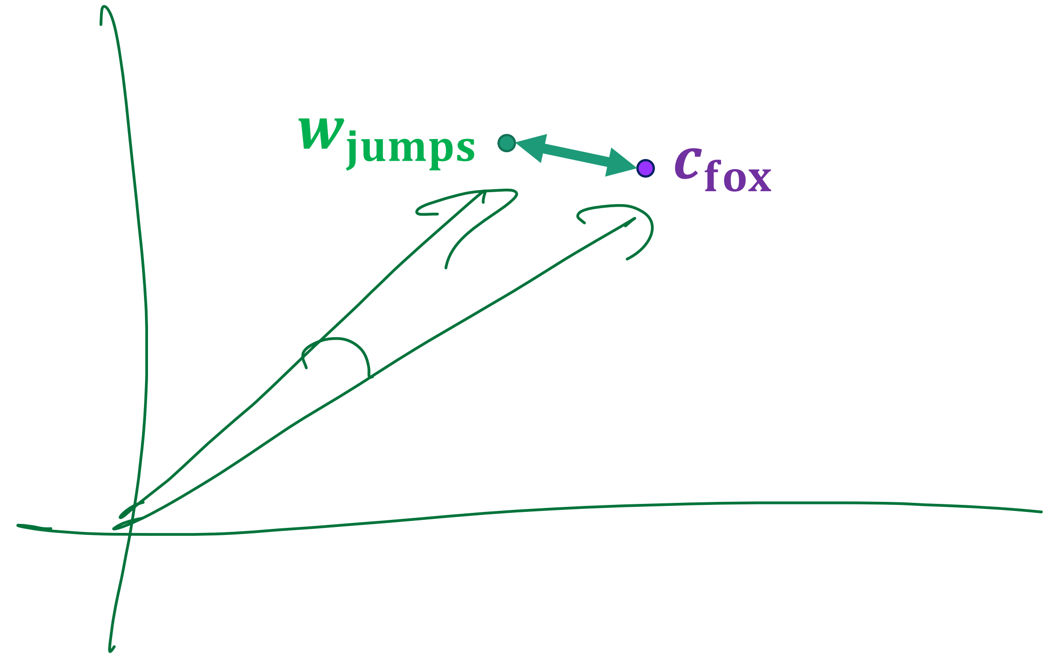
SGNS : Negative Sampling

Co-occurrence **jumps** , **fox**:

w_{jumps}

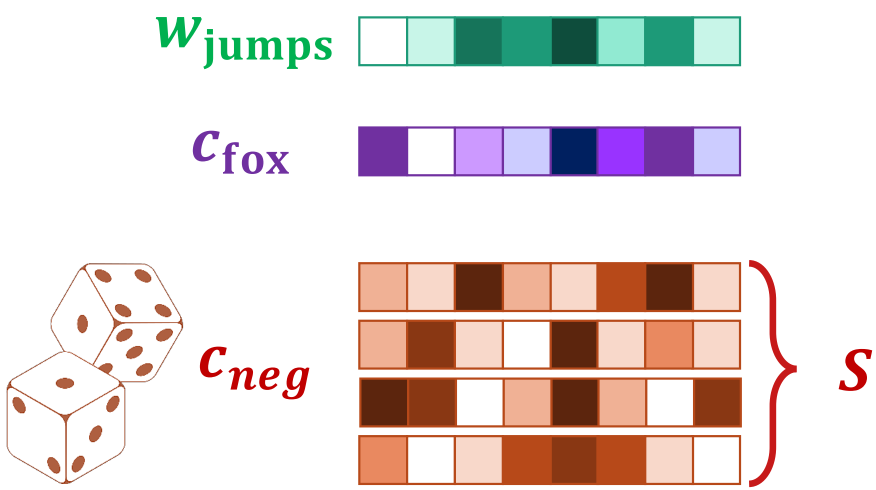
c_{fox}

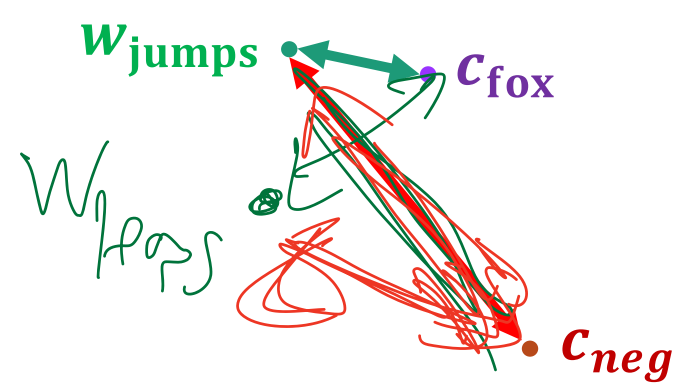




SGNS : Negative Sampling

Co-occurrence **jumps** , **fox**:





Sim(Wjumps, Wleaps)