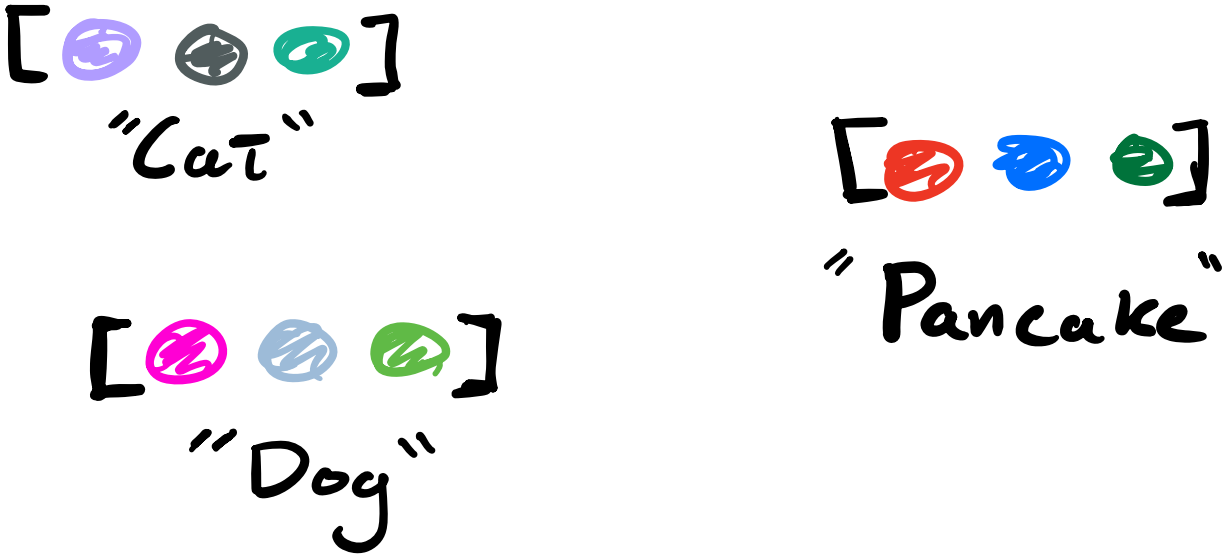
DS 4440 Embeddings

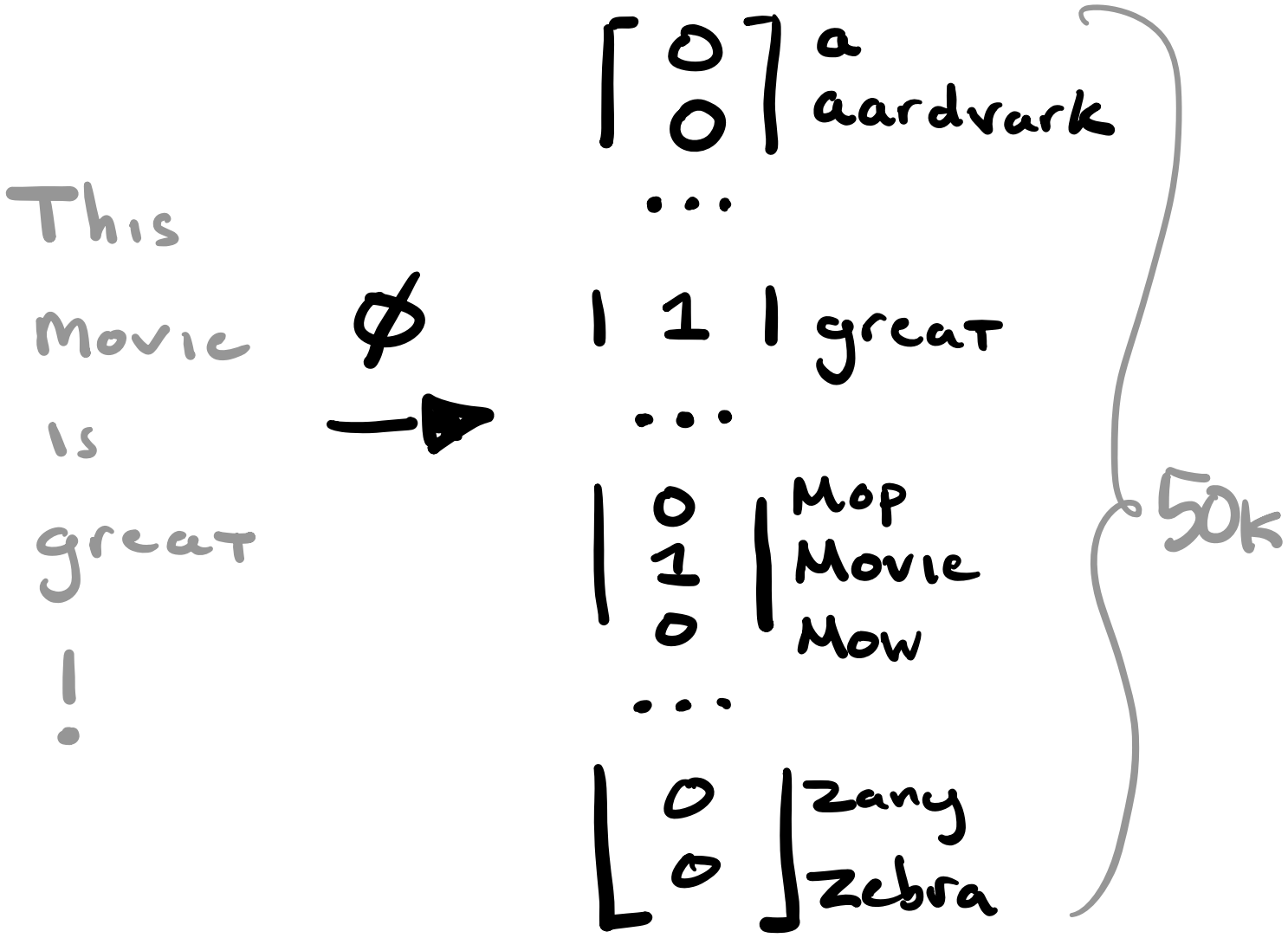
Today

- Our first foray into representation learning
- How to encode text
- Word2Vec (and beyond)



Wayyy back ~2013

Bag-of-Words (BoW)



Word ω → Position ω_{idx}

$$x_{\omega_{idx}} = \begin{cases} 1 & \text{if } \omega \text{ in input} \\ 0 & \text{Otherwise} \end{cases}$$

Long, sparse X vectors.

Problems with BoW?

We often want to measure Text Similarity

$$\text{Cos}(a, b) = \frac{\phi(a)^T \phi(b)}{\|\phi(a)\| \, \|\phi(b)\|}$$

For cat, dog

$$\underbrace{[0 \ldots 1 \ldots 0]}_{\text{"cat"}} \cdot \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} = 0.$$

"dog"

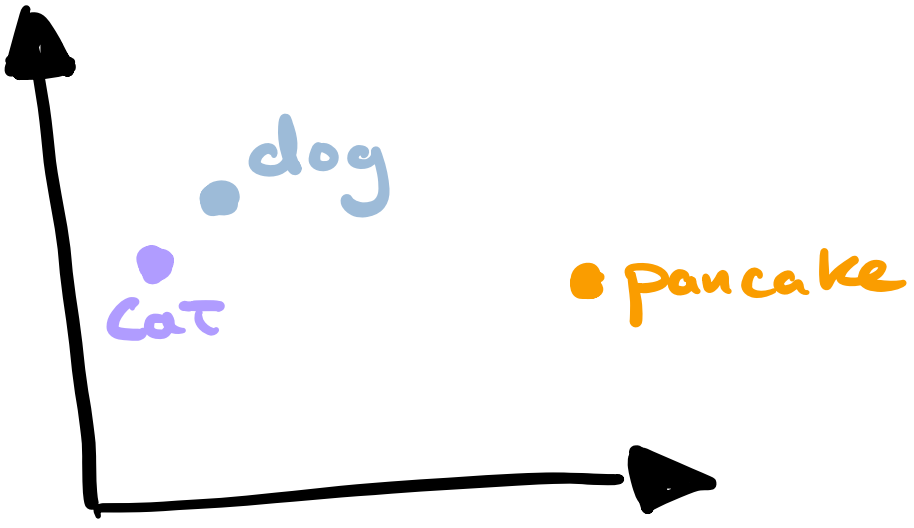
So

Sim(cat, dog) =

Sim(cat, pancake) = 0.

Distributional Semantics

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



Words to "low dimensional" embeddings

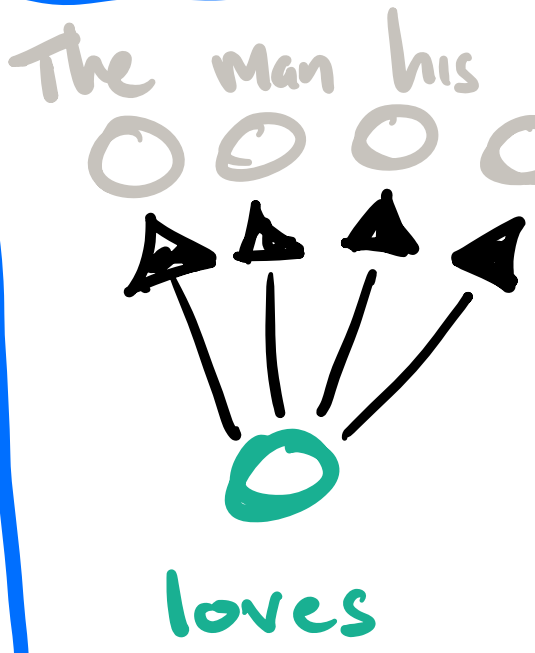
Similar words ~ Nearby

Word2Vec [Mikolov 2013]

Target word and Context

Two variants / objectives

Skip-gram

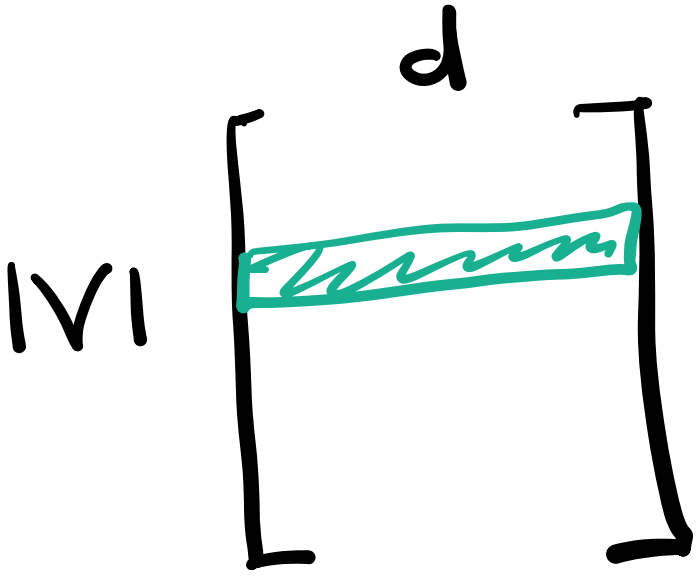


$P(\text{the} \mid \text{loves}) \cdot$
$P(\text{man} \mid \text{loves}) \cdot$
$P(\text{his} \mid \text{loves}) \cdot$
$P(\text{son} \mid \text{loves})$

$$\text{Loss}: \quad -\prod_{w_t} \prod_{w_c \text{ in context}} P(w_c \mid w_t)$$

$$P(w_c \mid w_t) = \frac{\exp\{e_{w_t} \cdot e_{w_c}\}}{\sum_{\tilde{w} \in V} \exp\{e_{w_t} \cdot e_{\tilde{w}}\}}$$

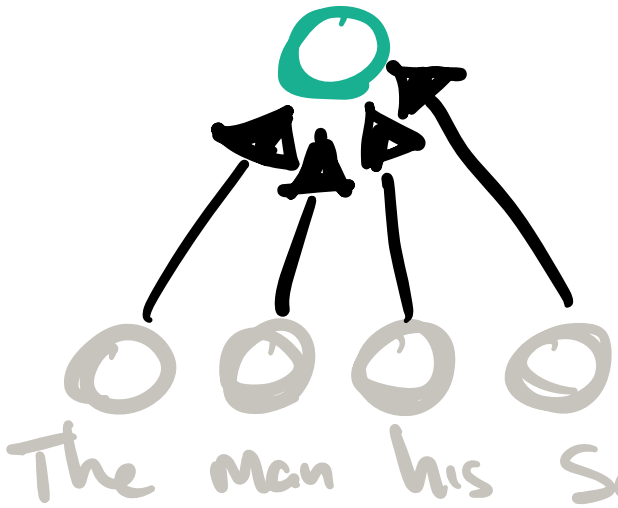
Softmax!



Embedding Matrix

CBoW

$P(\text{loves} \mid \text{the, man, his, son})$



$$\mathcal{L}_{loss}: -P(w_t \mid w_{c1} \dots w_{ck})$$

$$\doteq \frac{\exp\{e_{w_t} \cdot \bar{w}_c\}}{\sum_{\tilde{w} \in V} \exp\{e_{\tilde{w}} \cdot \bar{w}_c\}}$$

Where:

$$\bar{w}_c \doteq \frac{1}{k} \sum_{j=1}^{k} e_{w_{cj}}$$

Ok, BUT V is BIG!

→ Very slow

Idea Negative Sampling

First Sample

$$(w_t, w_{c1} \dots w_{ck})$$

Skip-gram

↑ Maximize Sim b/w w_t, w_{cl} S_1
(for all l)

↓ Minimize Sim b/w w_t and
· Words from other (random)
Contexts S_2

S_1 ↑ $\sigma(e_{w_t} \cdot e_{w_{cl}})$

S_2 ↓ $\sigma(e_{w_t} \cdot e_{\tilde{w}})$

$$Loss = -S_1 + S_2$$

CBoW

↑ Maximize Sim b/w w_t, $\bar{w}_c$

↓ Minimize Sim b/w $\tilde{w}$, $\bar{w}_c$

$\tilde{w}$: random words

$S_1 \uparrow \sigma(e_{w_t} \cdot e_{\bar{w}_c})$

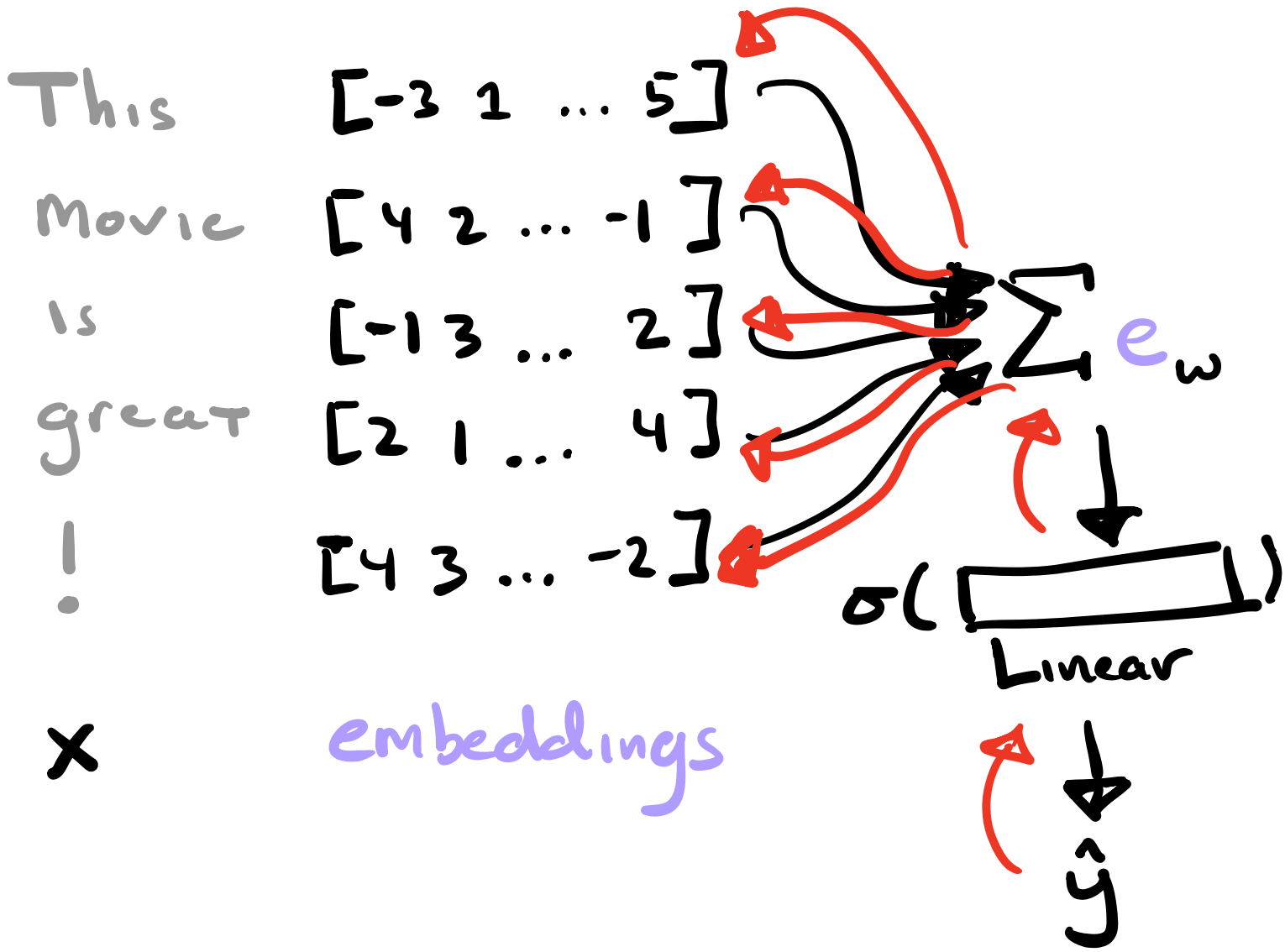
$S_2 \downarrow \sigma(e_{\tilde{w}} \cdot e_{\bar{w}_c})$

$$Loss = -S_1 + S_2$$

Let's see in Colab ...

How to use embeddings (for Text Classification)?

One way: Adopt CBoW



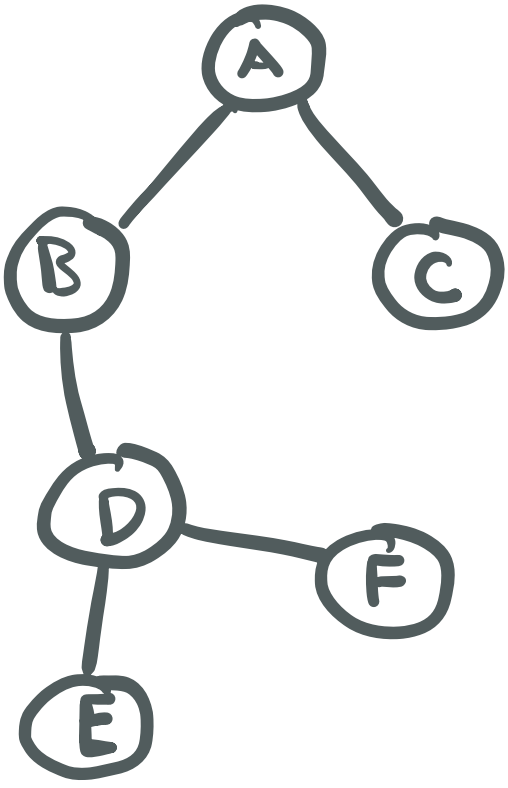
x embeddings

Beyond "Words"

Embeddings not just for NLP!

Example graph nodes

DeepWalk embeds (Perozzi et al.)



- Sample walks randomly
- Treat as "Sentences"

A B D E

A C

...

B D F

Back to NLP, Can Treat Paragraphs or documents as Special "Words"