Word Embeddings

CS 5624: Natural Language Processing

Spring 2025

https://tuvllms.github.io/nlp-spring-2025

Tu Vu

VIRGINIA TECH.

Logistics

- 🚨 Homework 0 & Quiz 0 due tomorrow 🚨

Deep Research

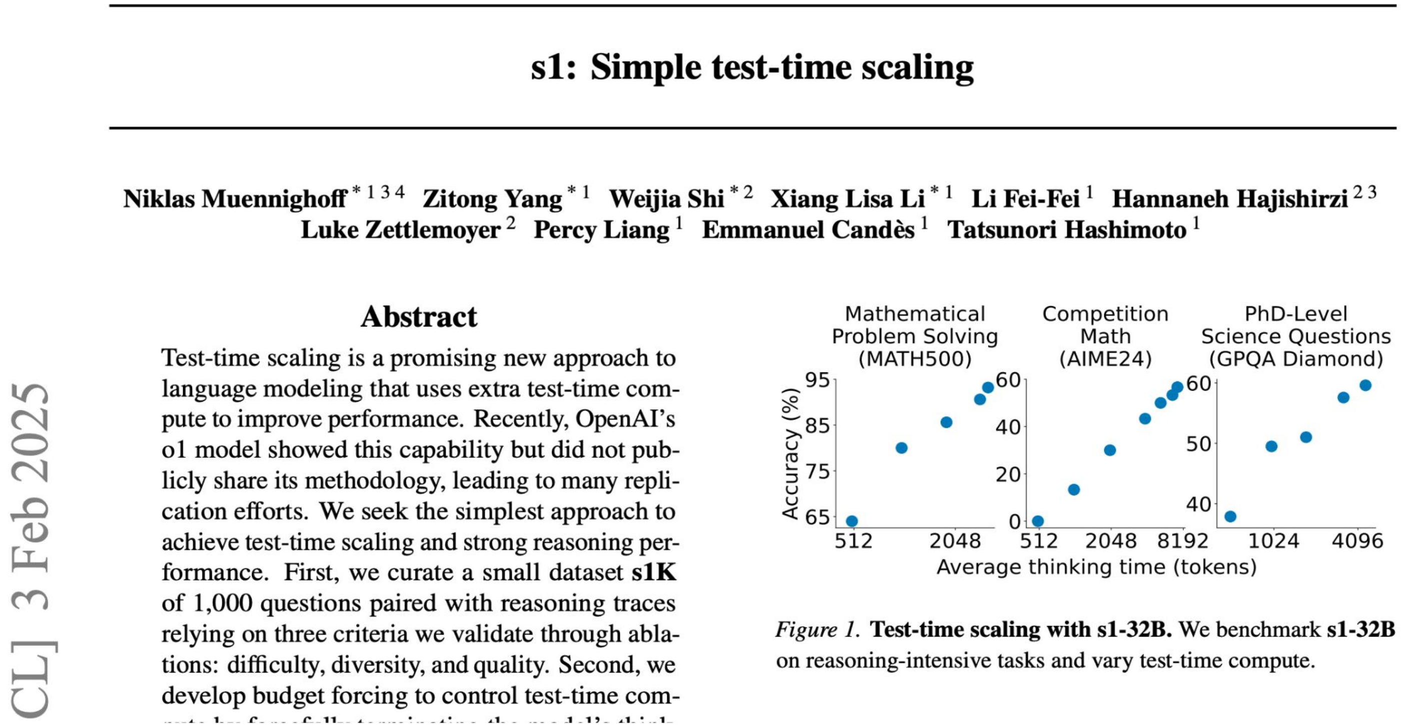
- https://openai.com/index/introducing-deep-research/

OmniHuman-1

- Albert Einstein
  - https://x.com/stillgray/status/1887111258245095508

- Taylor Swift singing Naruto song convincingly
  - https://x.com/bevenky/status/1886840149012607087/video/1

s1: Simple test-time scaling



s1: Simple test-time scaling

Niklas Muennighoff *[1 3 4] Zitong Yang *[1] Weijia Shi *[2] Xiang Lisa Li *[1] Li Fei-Fei[1] Hannaneh Hajishirzi[2 3]
Luke Zettlemoyer[2] Percy Liang[1] Emmanuel Candès[1] Tatsunori Hashimoto[1]

CL] 3 Feb 2025

Abstract

Test-time scaling is a promising new approach to language modeling that uses extra test-time compute to improve performance. Recently, OpenAI's o1 model showed this capability but did not publicly share its methodology, leading to many replication efforts. We seek the simplest approach to achieve test-time scaling and strong reasoning performance. First, we curate a small dataset **s1K** of 1,000 questions paired with reasoning traces relying on three criteria we validate through ablations: difficulty, diversity, and quality. Second, we develop budget forcing to control test-time compute by forcefully terminating the model's think-

Mathematical Problem Solving (MATH500)
Competition Math (AIME24)
PhD-Level Science Questions (GPQA Diamond)
Accuracy (%)
95
85
75
65
512
2048
60
40
20
0
512
2048
8192
60
50
40
1024
4096
Average thinking time (tokens)

Figure 1. **Test-time scaling with s1-32B.** We benchmark **s1-32B** on reasoning-intensive tasks and vary test-time compute.

https://arxiv.org/pdf/2501.19393

RLHF

- https://rlhfbook.com/

Cosine similarity

$$\text{dot product}(\mathbf{v},\mathbf{w}) = \mathbf{v}\cdot\mathbf{w} = \sum_{i=1}^{N} v_i w_i = v_1 w_1 + v_2 w_2 + \ldots + v_N w_N$$

$$\text{cosine}(\mathbf{v},\mathbf{w}) = \frac{\mathbf{v}\cdot\mathbf{w}}{|\mathbf{v}||\mathbf{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

Word2vec

- Continuous bag-of-words (CBoW)
- Skip-gram with negative sampling (SGNS)

Skip-gram with negative sampling (SGNS)

- Simplifies the task: *binary classification* instead of *word prediction*

- Simplifies the architecture: *logistic regression* instead of *multi-layer neural network*

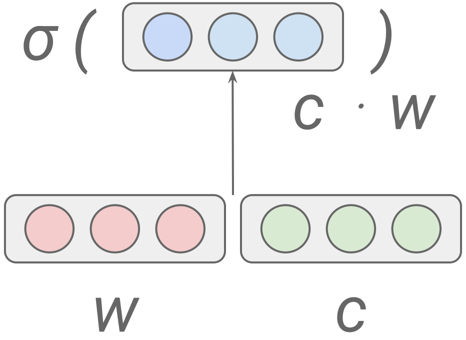
Skip-gram with negative sampling (SGNS) (cont’d)

- Intuition
  - Treat the target word and a neighboring context word as *positive examples*
  - Randomly sample other words in the lexicon to get *negative samples*
  - Train a classifier (**self-supervision**) to distinguish those two cases
  - Use the learned weights as the embeddings

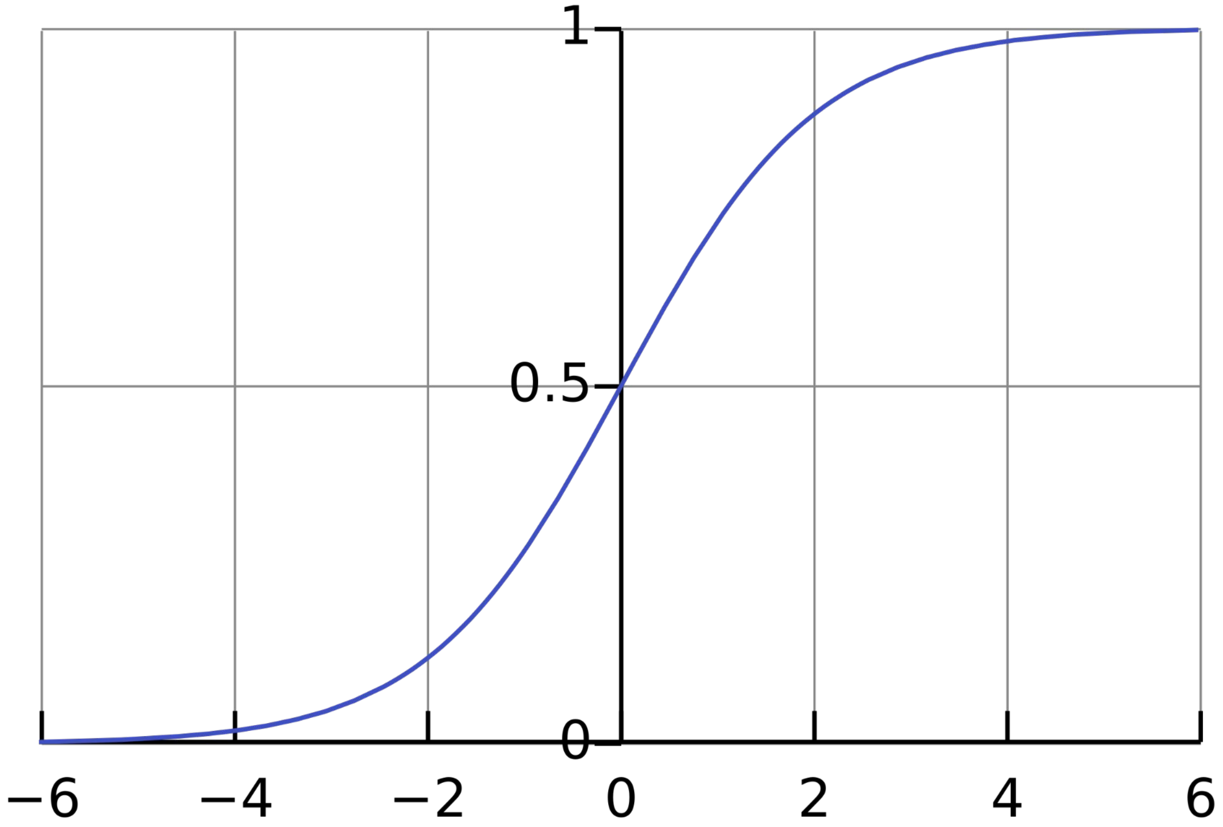
Logistic regression

$$P(+|w,c) \ = \ \sigma(\mathbf{c}\cdot\mathbf{w}) = \frac{1}{1+\exp\left(-\mathbf{c}\cdot\mathbf{w}\right)}$$

the probability that word *c* is a real context word for target word *w*



Sigmoid function



Sigmoid function (cont'd)

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

For $1 - \sigma(x)$:

$$1 - \sigma(x) = \frac{e^{-x}}{1 + e^{-x}}$$

Dividing numerator and denominator by e^{-x}:

$$1 - \sigma(x) = \frac{1}{e^{x} + 1} = \sigma(-x)$$

Logistic regression (cont'd)

$$P(+|w,c) = \sigma(\mathbf{c}\cdot\mathbf{w}) = \frac{1}{1+\exp(-\mathbf{c}\cdot\mathbf{w})}$$

the probability that word ***c*** is a real context word for target word ***w***

$$\begin{aligned} P(-|w,c) &= 1 - P(+|w,c) \\ &= \sigma(-\mathbf{c}\cdot\mathbf{w}) = \frac{1}{1+\exp(\mathbf{c}\cdot\mathbf{w})} \end{aligned}$$

the probability that word ***c*** is ***not*** a real context word for target word ***w***

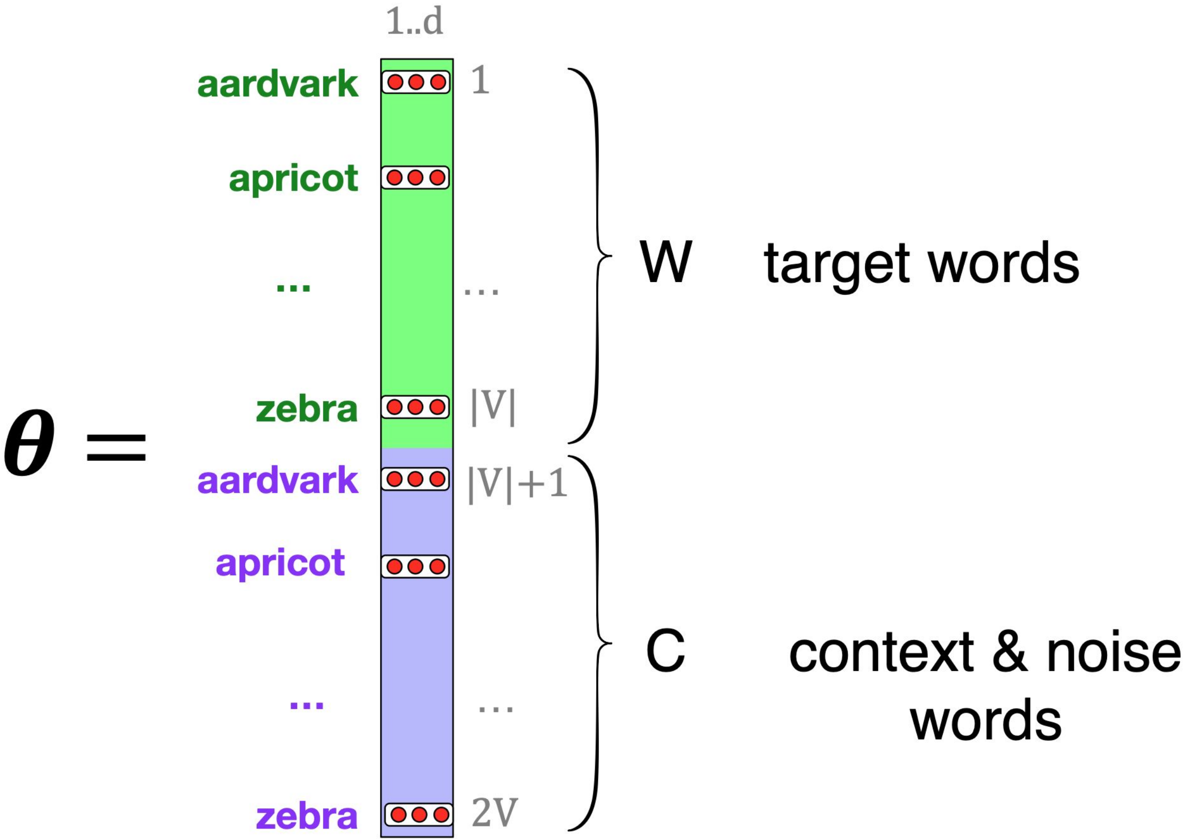
The probability for many context words

Given a target word w and its context window of *L* words $c_{1:L}$, assigns a probability based on how similar this context window is to the target word

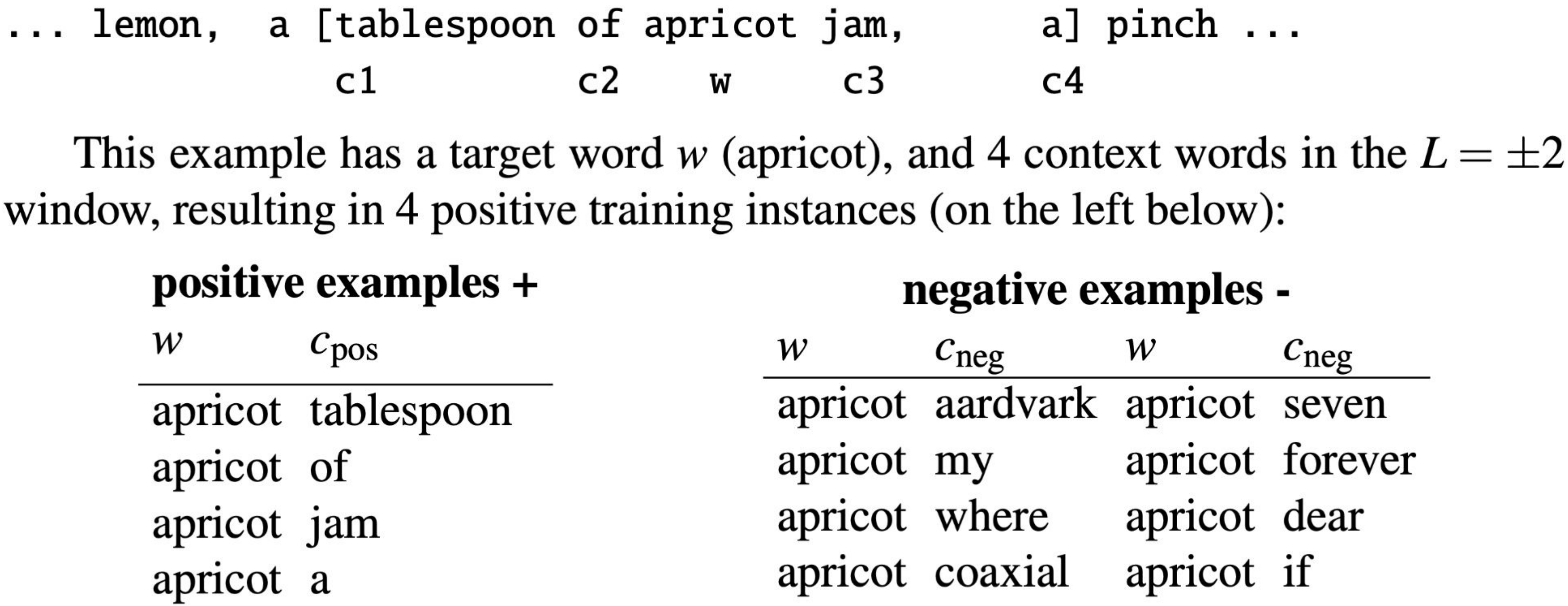
$$P(+|w,c_{1:L}) = \prod_{i=1}^{L} \sigma(\mathbf{c_i} \cdot \mathbf{w})$$

$$\log P(+|w,c_{1:L}) = \sum_{i=1}^{L} \log \sigma(\mathbf{c_i} \cdot \mathbf{w})$$

Two embedding tables



Learning skip-gram embeddings



```
... lemon,   a [tablespoon of apricot jam,      a] pinch ...
                c1          c2     w     c3      c4
```

This example has a target word w (apricot), and 4 context words in the $L = \pm 2$ window, resulting in 4 positive training instances (on the left below):

positive examples +

w	c_{pos}
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

negative examples -

w	c_{neg}	w	c_{neg}
apricot	aardvark	apricot	seven
apricot	my	apricot	forever
apricot	where	apricot	dear
apricot	coaxial	apricot	if

Loss function

$$
\begin{aligned}
L &= -\log\left[\boxed{P(+|w,c_{pos})}\prod_{i=1}^{k}P(-|w,c_{neg_i})\right] \\
&= -\left[\log P(+|w,c_{pos})+\sum_{i=1}^{k}\log P(-|w,c_{neg_i})\right] \\
&= -\left[\log P(+|w,c_{pos})+\sum_{i=1}^{k}\log\left(1-P(+|w,c_{neg_i})\right)\right] \\
&= -\left[\log\sigma(c_{pos}\cdot w)+\sum_{i=1}^{k}\log\sigma(-c_{neg_i}\cdot w)\right]
\end{aligned}
$$

Maximize the similarity of the target word, context word pairs (w, c_{pos}) drawn from the positive examples

Loss function (cont'd)

Minimize the similarity of the (w, c_{neg}) pairs from the negative examples

$$
\begin{aligned}
L &= -\log\left[P(+|w,c_{pos})\prod_{i=1}^{k}P(-|w,c_{neg_i})\right] \\
&= -\left[\log P(+|w,c_{pos})+\sum_{i=1}^{k}\log P(-|w,c_{neg_i})\right] \\
&= -\left[\log P(+|w,c_{pos})+\sum_{i=1}^{k}\log\left(1-P(+|w,c_{neg_i})\right)\right] \\
&= -\left[\log\sigma(c_{pos}\cdot w)+\sum_{i=1}^{k}\log\sigma(-c_{neg_i}\cdot w)\right]
\end{aligned}
$$

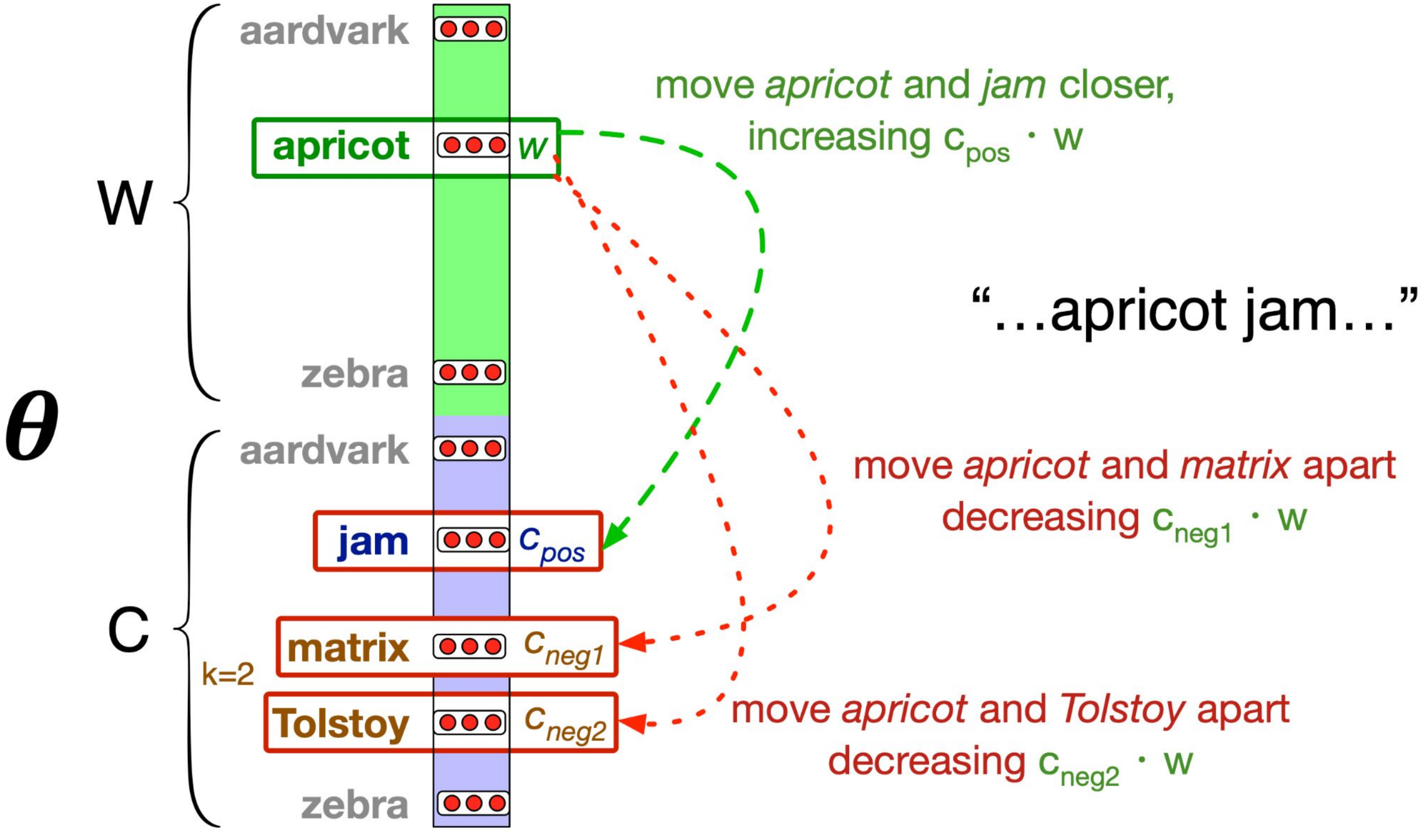
Training

$$
\begin{aligned}
\frac{\partial L}{\partial c_{pos}} &= [\sigma(\mathbf{c}_{pos} \cdot \mathbf{w}) - 1]\mathbf{w} \\
\frac{\partial L}{\partial c_{neg}} &= [\sigma(\mathbf{c}_{neg} \cdot \mathbf{w})]\mathbf{w} \\
\frac{\partial L}{\partial w} &= [\sigma(\mathbf{c}_{pos} \cdot \mathbf{w}) - 1]\mathbf{c}_{pos} + \sum_{i=1}^{k} [\sigma(\mathbf{c}_{\text{neg}_i} \cdot \mathbf{w})]\mathbf{c}_{neg_i}
\end{aligned}
$$

The update equations going from time step t to $t+1$ in stochastic gradient descent are thus:

$$
\begin{aligned}
\mathbf{c}_{pos}^{t+1} &= \mathbf{c}_{pos}^{t} - \eta[\sigma(\mathbf{c}_{pos}^{t} \cdot \mathbf{w}^{t}) - 1]\mathbf{w}^{t} \\
\mathbf{c}_{neg}^{t+1} &= \mathbf{c}_{neg}^{t} - \eta[\sigma(\mathbf{c}_{neg}^{t} \cdot \mathbf{w}^{t})]\mathbf{w}^{t} \\
\mathbf{w}^{t+1} &= \mathbf{w}^{t} - \eta\left[[\sigma(\mathbf{c}_{pos}^{t} \cdot \mathbf{w}^{t}) - 1]\mathbf{c}_{pos}^{t} + \sum_{i=1}^{k} [\sigma(\mathbf{c}_{\text{neg}_i}^{t} \cdot \mathbf{w}^{t})]\mathbf{c}_{neg_i}^{t}\right]
\end{aligned}
$$

Training (cont'd)



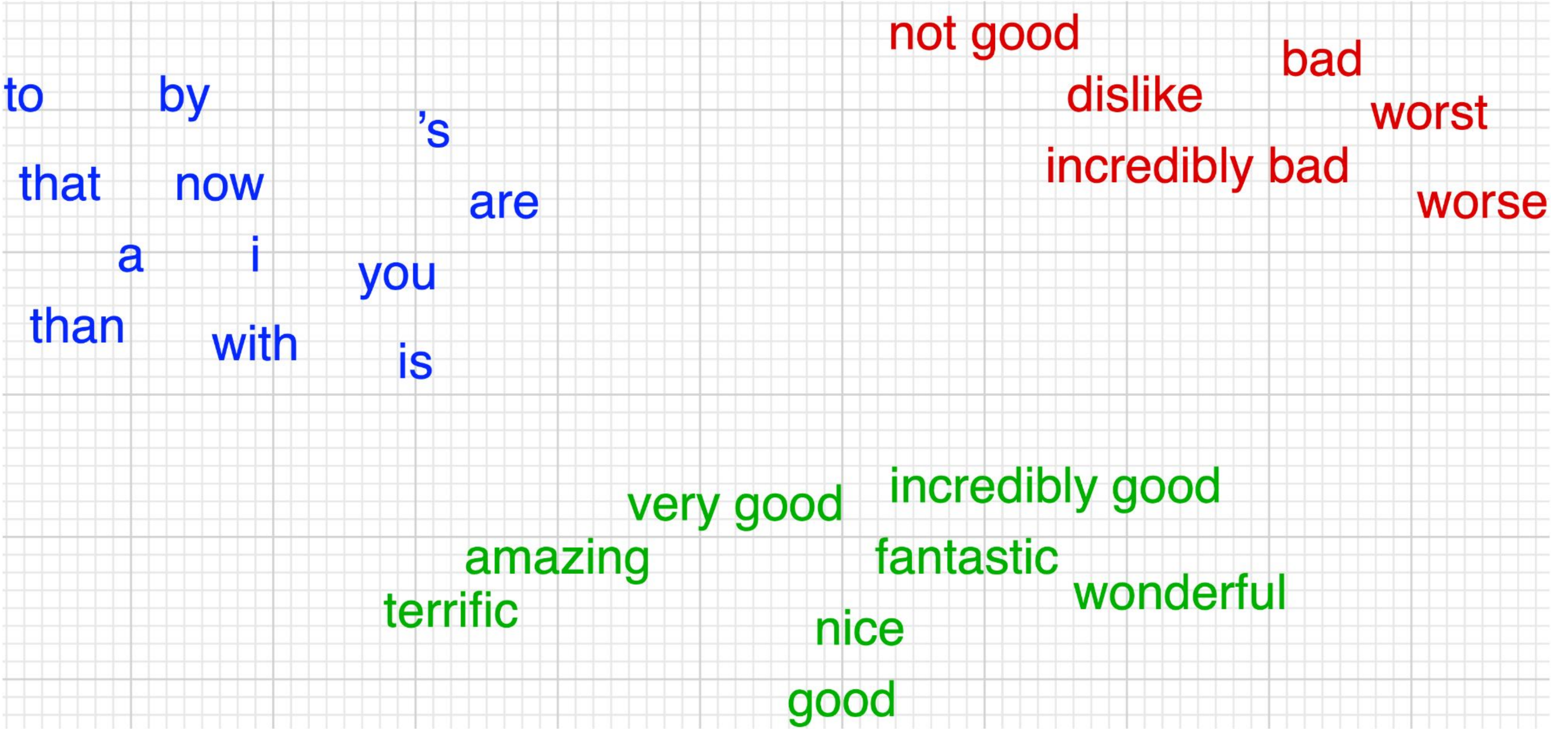
move *apricot* and *jam* closer, increasing $c_{pos} \cdot w$

"…apricot jam…"

move *apricot* and *matrix* apart decreasing $c_{neg1} \cdot w$

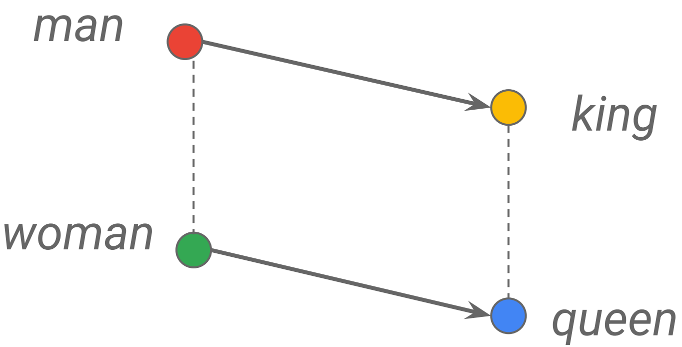
move *apricot* and *Tolstoy* apart decreasing $c_{neg2} \cdot w$

Visualizing word embeddings



A two-dimensional (t-SNE) projection of embeddings for some words and phrases, showing that words with similar meanings are nearby in space.

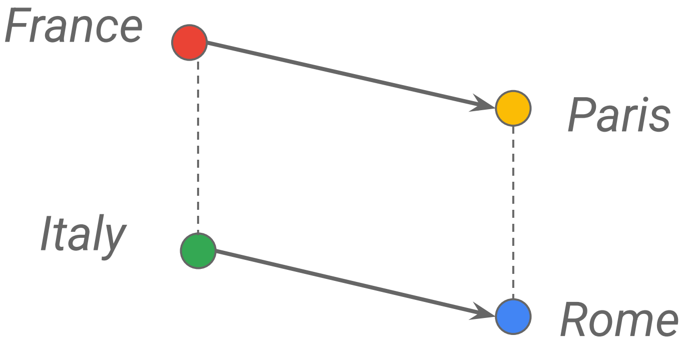
Semantic properties of word embeddings

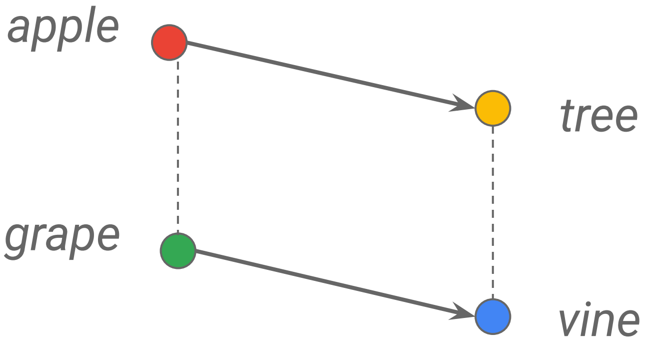


$$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} \approx \overrightarrow{\text{queen}} - \overrightarrow{\text{woman}}$$

$$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} \approx \overrightarrow{\text{queen}}$$

Semantic properties of word embeddings (cont'd)

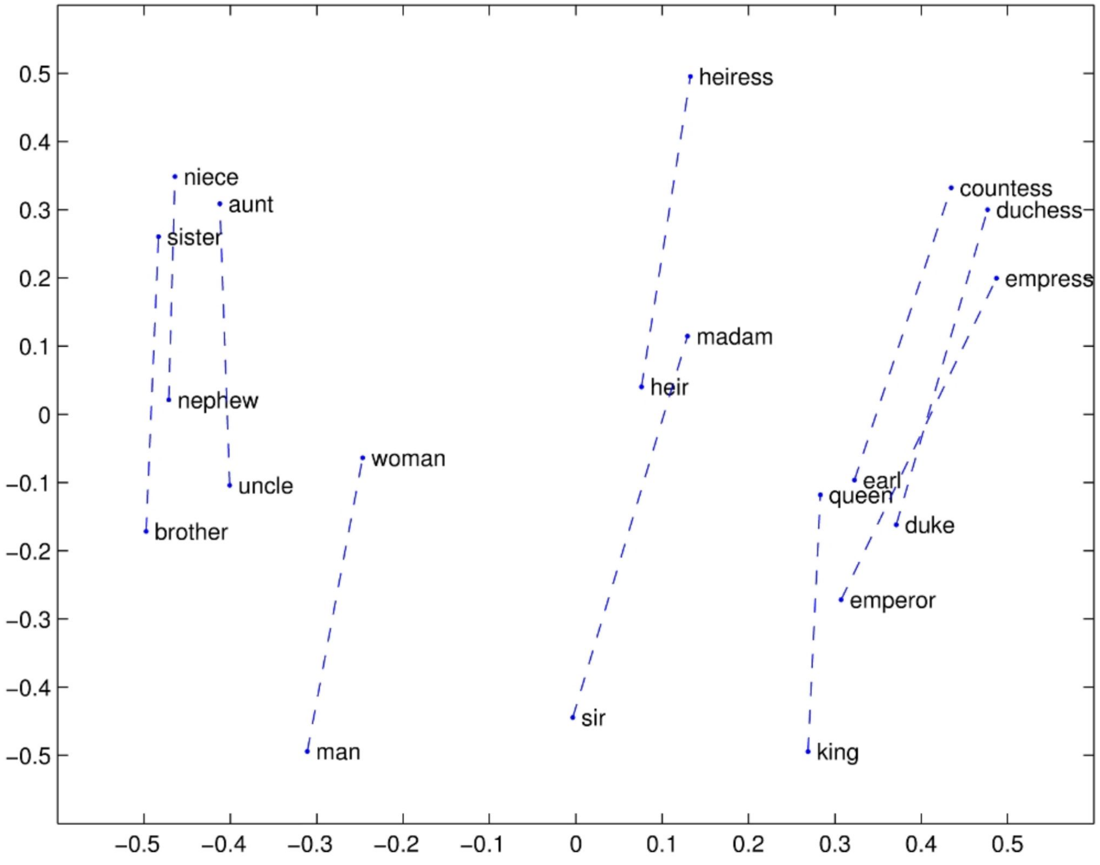




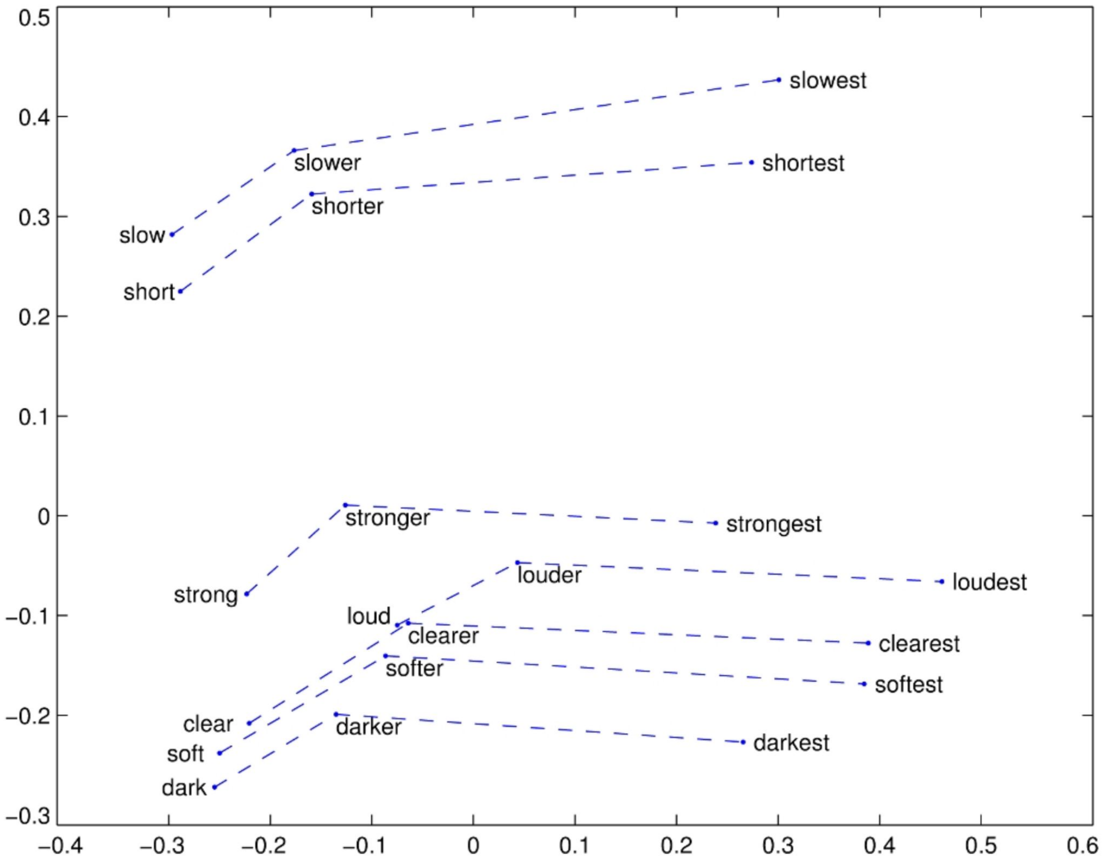
$$\overrightarrow{\text{Paris}} - \overrightarrow{\text{France}} \approx \overrightarrow{\text{Rome}} - \overrightarrow{\text{Italy}} \qquad \overrightarrow{\text{tree}} - \overrightarrow{\text{apple}} \approx \overrightarrow{\text{vine}} - \overrightarrow{\text{grape}}$$

$$\overrightarrow{\text{Paris}} - \overrightarrow{\text{France}} + \overrightarrow{\text{Italy}} \approx \overrightarrow{\text{Rome}} \qquad \overrightarrow{\text{tree}} - \overrightarrow{\text{apple}} + \overrightarrow{\text{grape}} \approx \overrightarrow{\text{vine}}$$

Semantic properties of word embeddings (cont’d)



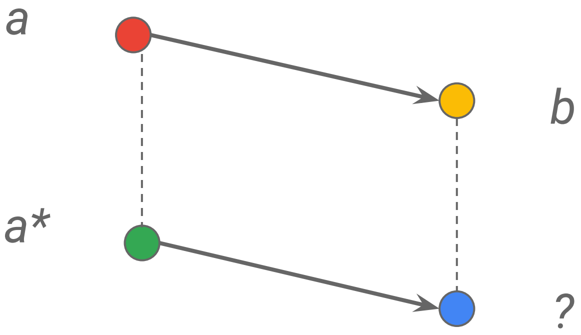
Semantic properties of word embeddings (cont'd)

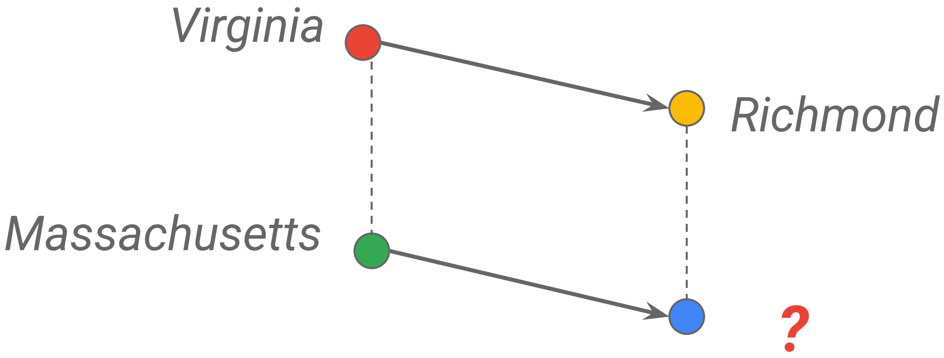


comparative & superlative morphology

Analogy problem $\mathbf{a} : \mathbf{b} :: \mathbf{a}^* : \mathbf{b}^*$

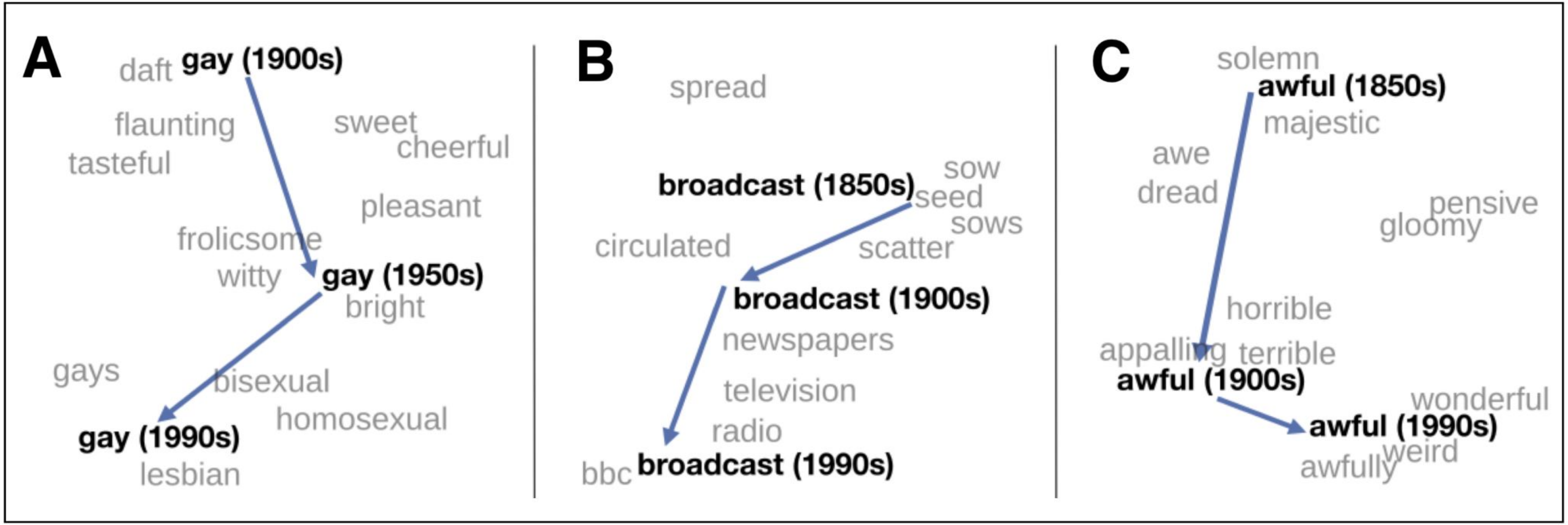
a is to *b* as *a** is to what?





$$\hat{\mathbf{b}}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \; \operatorname{distance}(\mathbf{x}, \mathbf{b} - \mathbf{a} + \mathbf{a}^*)$$

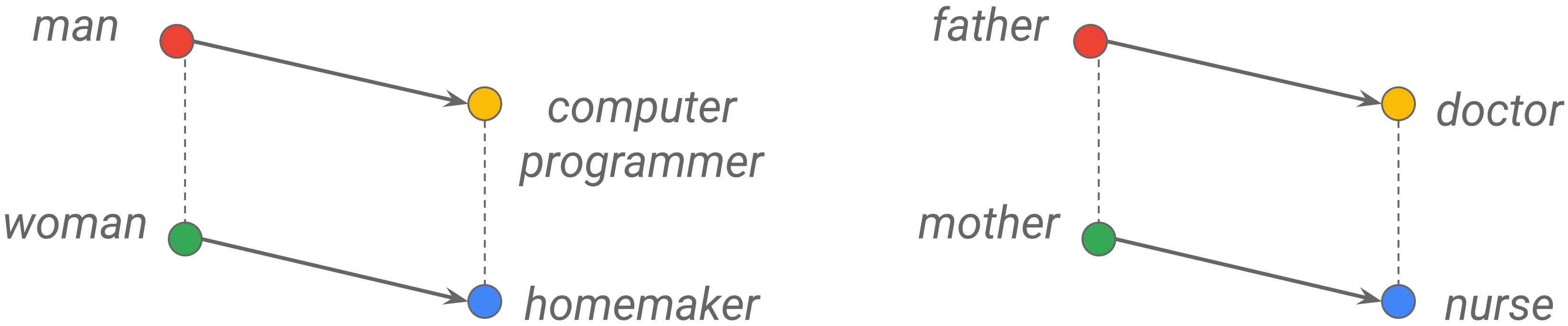
Historical Semantics



A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors.

Biases in word embeddings

- Gender stereotypes



Biases in word embeddings (cont’d)

- People in the US associate:
  - African-American names with unpleasant words (more than European-American names)
  - male names more with mathematics and female names with the arts, and old people’s names with unpleasant words
- …

Word embedding methods

- Word2vec
- Glove: https://nlp.stanford.edu/projects/glove/
- Fasttext: https://fasttext.cc/

Thank you!