

Word Representation

Part II



Tanmoy Chakraborty
Associate Professor, IIT Delhi
<https://tanmoychak.com/>

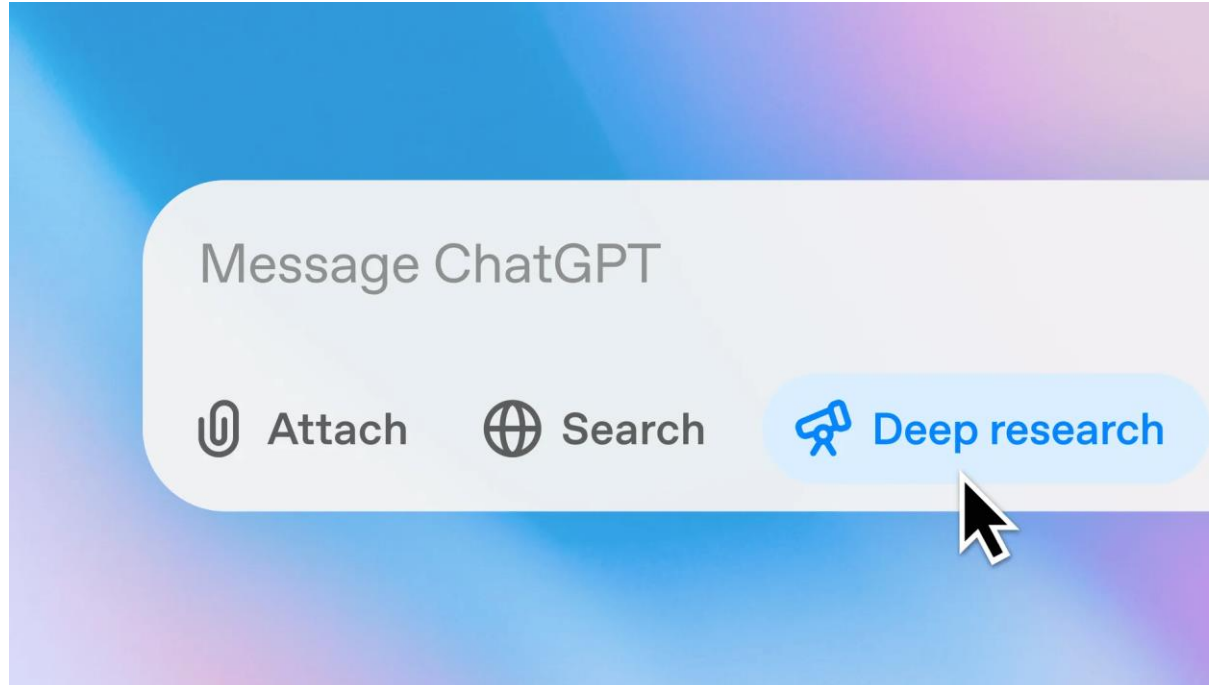


OpenAI releases Deep Research !!

Released on
February 2, 2025

[OpenAI Blog](#)

Deep Research is an agent that uses 'reasoning' to synthesize large amounts of online information and conducts multi-step research on the internet for complex tasks. It is powered by a version of the upcoming OpenAI o3 model that's optimized for web browsing and data analysis.



OpenAI claims that Deep Research is able to accomplish 'complex, time-intensive web research' tasks in tens of minutes what would take a human many hours.

Deep Research was trained using end-to-end reinforcement learning on hard real-world tasks, requiring browser and Python tool use, across a range of domains. Through that training, it learned to plan and execute a multi-step trajectory to find the data it needs, backtracking and reacting to real-time information where necessary.





Count-based vs Prediction-based

Count-based

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to large counts



Count-based vs Prediction-based

Count-based	Prediction-based
<ul style="list-style-type: none">• Fast training• Efficient usage of statistics 	<ul style="list-style-type: none">• Scales with corpus size• Inefficient usage of statistics 
<ul style="list-style-type: none">• Primarily used to capture word Similarity• Disproportionate importance given to large counts 	<ul style="list-style-type: none">• Generate improved performance on other tasks• Can capture complex patterns beyond word similarity 



GloVe – Global Vectors

Crucial insight: Ratios of co-occurrence probabilities can encode word meaning

	$x = \textit{solid}$	$x = \textit{gas}$	$x = \textit{water}$	$x = \textit{random}$
$P(x \mid \textit{ice})$	large	small	large	small
$P(x \mid \textit{steam})$	small	large	large	small
$\frac{P(x \mid \textit{ice})}{P(x \mid \textit{steam})}$	large	small	~ 1	~ 1

Jeffrey Pennington, Richard Socher, Christopher D. Manning, “GloVe: Global Vectors for Word Representation”, 2014



GloVe – Global Vectors

Crucial insight: Ratios of co-occurrence probabilities can encode word meaning

	$x = \textit{solid}$	$x = \textit{gas}$	$x = \textit{water}$	$x = \textit{random}$
$P(x \mid \textit{ice})$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x \mid \textit{steam})$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x \mid \textit{ice})}{P(x \mid \textit{steam})}$	8.9	8.5×10^{-2}	1.36	0.96

Jeffrey Pennington, Richard Socher, Christopher D. Manning, “GloVe: Global Vectors for Word Representation”, 2014



Co-occurrence Matrix

- Let us denote the co-occurrence matrix as **X**.

count	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

Compute $P(j | i)$ from **X**, for two words **i** and **j** in the corpus.

$$P(j|i) = \frac{X_{ij}}{\sum_j X_{ij}} = \frac{X_{ij}}{X_i}$$



Learn Word Vectors Based on These Counts

- For the two words, i and j , assume their corresponding representation vectors are w_i and w_j , respectively.

- $\underbrace{w_i^T w_j}_{\substack{\text{Similarity} \\ \text{between} \\ \text{words } i \text{ and } j}} = \log \underbrace{P(j|i)}_{\substack{\text{How likely is } j \text{ to} \\ \text{occur in the context} \\ \text{of } i}}$

- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i \quad \dots (1)$

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j \quad \dots (2)$



Learn Word Vectors Based on These Counts

- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i \quad \dots (1)$

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j \quad \dots (2)$

- Adding (1) and (2):

$$\begin{aligned} 2 w_i^T w_j &= 2 \log X_{ij} - \log X_i - \log X_j \\ \Rightarrow w_i^T w_j &= \log X_{ij} - \frac{1}{2} \log X_i - \frac{1}{2} \log X_j \end{aligned}$$



Learn Word Vectors Based on These Counts

$$w_i^T w_j = \log X_{ij} - \frac{1}{2} \log X_i - \frac{1}{2} \log X_j$$

- $\log X_i$ and $\log X_j$ depend only on i and j , respectively – can be thought of as word-specific biases
 - These are made learnable (considered as biases)

$$\begin{aligned} w_i^T w_j &= \log X_{ij} - b_i - b_j \\ \Rightarrow w_i^T w_j + b_i + b_j &= \log X_{ij} \end{aligned}$$

- w_i, w_j, b_i, b_j are the learnable parameters
- **Loss function:** $\min_{w_i, w_j, b_i, b_j} \sum_{i,j} (w_i^T w_j + b_i + b_j - \log X_{ij})^2$



Learn Word Vectors Based on These Counts

Loss function: $\min_{w_i, w_j, b_i, b_j} \sum_{i,j} (w_i^T w_j + b_i + b_j - \log X_{ij})^2$

- **Problem:** Gives equal weightage to every co-occurrence
- **Ideally, rare and very frequent co-occurrences should have lesser weightage**
- **Modification:** Add a weighting function $f(x)$.
- **Modified loss function:** $\min_{w_i, w_j, b_i, b_j} \sum_{i,j} f(X_{ij}) (w_i^T w_j + b_i + b_j - \log X_{ij})^2$

What can f possibly be?



Weighting function

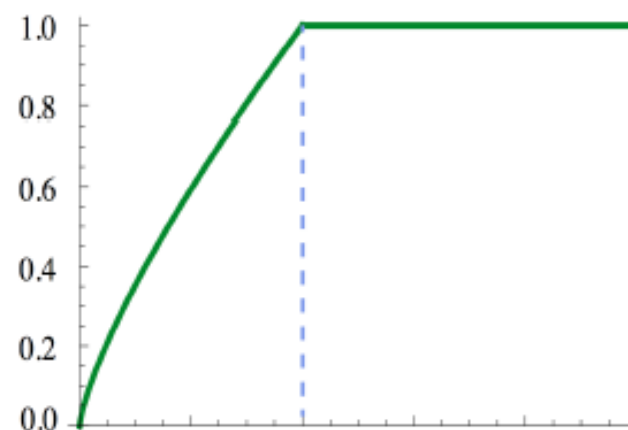
$$f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

α can be chosen empirically for a given dataset.

Properties of f :

1. $f(0) = 0$. If f is viewed as a continuous function, it should vanish as $x \rightarrow 0$ fast enough that the $\lim_{x \rightarrow 0} f(x) \log^2 x$ is finite.
2. $f(x)$ should be non-decreasing so that rare co-occurrences are not overweighted.
3. $f(x)$ should be relatively small for large values of x , so that frequent co-occurrences are not overweighted.

$f \sim$



GloVe: Advantages

- Fast training
- Scalable to huge corpora
- Good performance even with small corpus and small vectors



Details About GloVe

Original paper: <https://nlp.stanford.edu/pubs/glove.pdf>

Blogs with easy explanations:

- <https://medium.com/sciforce/word-vectors-in-natural-language-processing-global-vectors-glove-51339db89639>
- https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/?fbclid=IwAR3-pws3-K-Snfk6aqbixdxS8zFf-uuPDJ_0ipb94kWeygrdCSEqE9HWmNs
- <https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010>



Properties of Embeddings



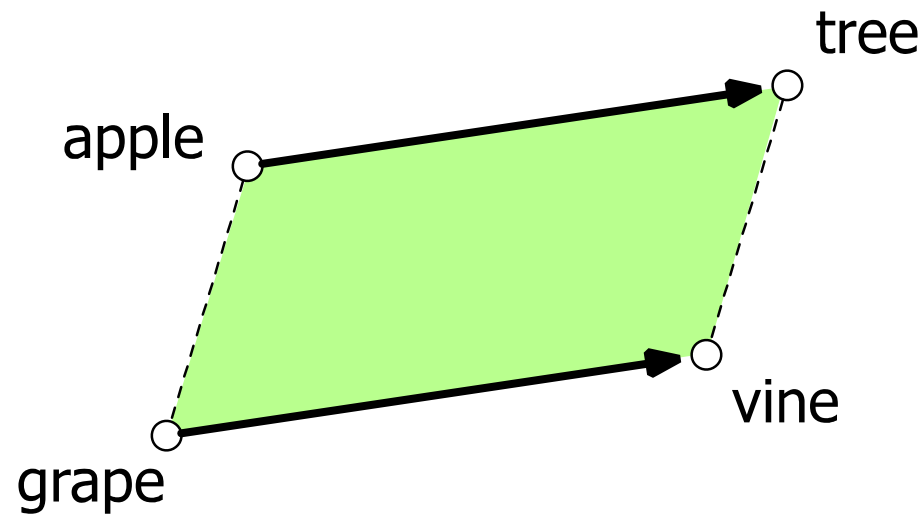
The Kinds of Neighbors Depend on Window Size

- **Small windows** ($C = \pm 2$) : nearest words are syntactically similar words in same taxonomy
 - *Hogwarts* nearest neighbors are other fictional schools
 - *Sunnydale, Evernight, Blandings*
- **Large windows** ($C = \pm 5$) : nearest words are related words in same semantic field
 - *Hogwarts* nearest neighbors are Harry Potter world:
 - *Dumbledore, half-blood, Malfoy*



Analogical Relations

- The classic parallelogram model of analogical reasoning (Rumelhart and Abrahamson 1973)
- To solve: *"apple is to tree as grape is to _____"*
- Add $\overrightarrow{\text{tree}} - \overrightarrow{\text{apple}}$ to $\overrightarrow{\text{grape}}$ to get *vine*



Analogical Relations via Parallelogram

- The parallelogram method can solve analogies with both sparse and dense embeddings (Turney and Littman 2005, Mikolov et al. 2013b)

$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}}$ is close to $\overrightarrow{\text{queen}}$

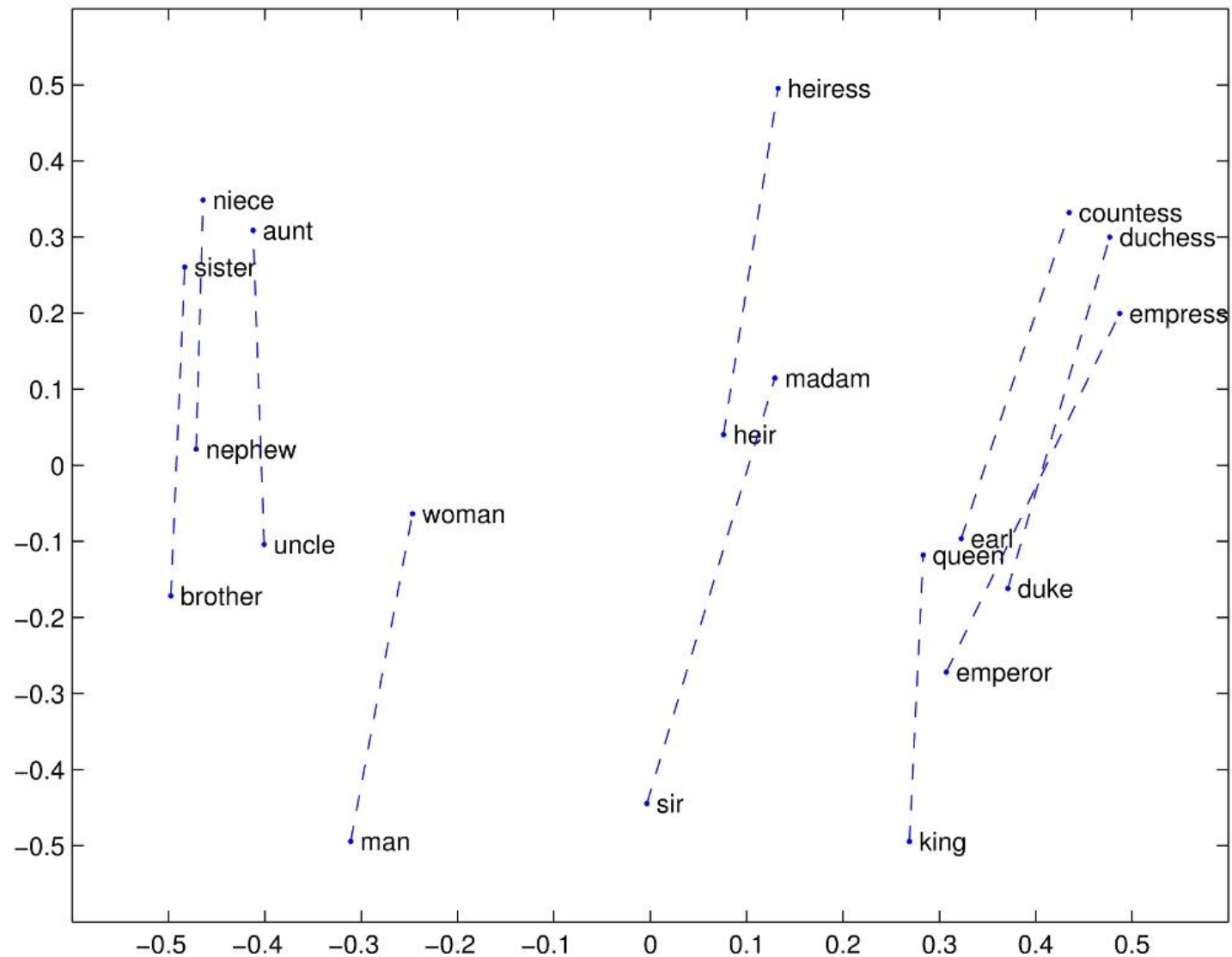
$\overrightarrow{\text{Paris}} - \overrightarrow{\text{France}} + \overrightarrow{\text{Italy}}$ is close to $\overrightarrow{\text{Rome}}$

- For a problem $a:a^*:b:b^*$, the parallelogram method is:

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



Structure in GloVE Embedding space



Caveats With The Parallelogram Method

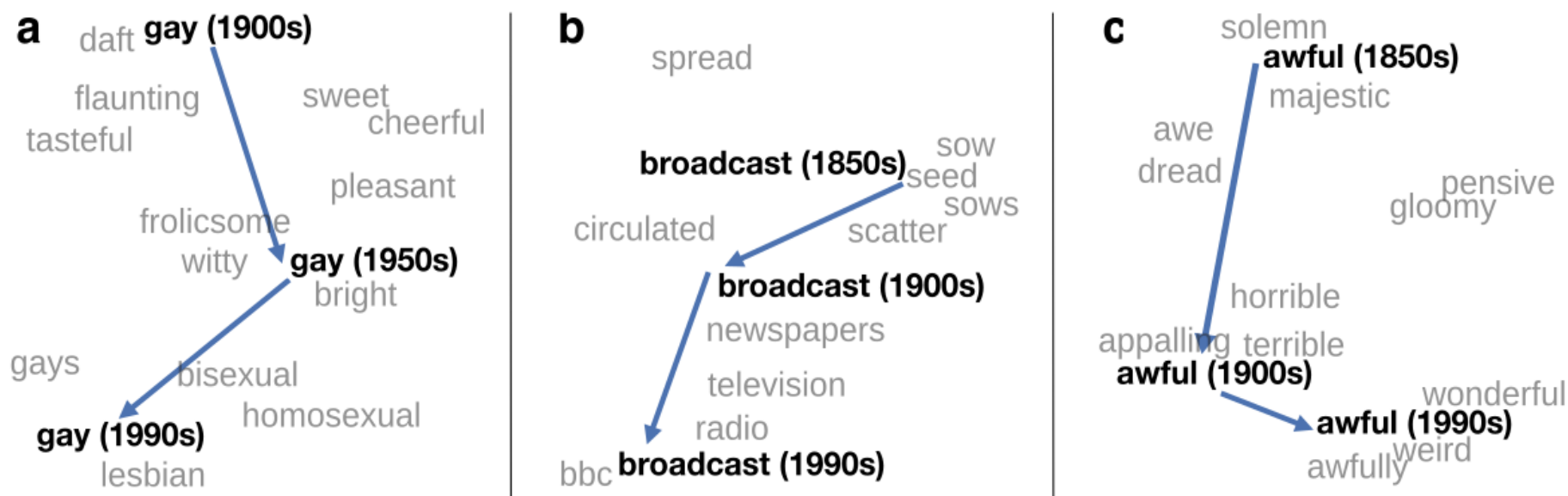
- It only seems to work for frequent words, small distances and certain relations (relating countries to capitals, or parts of speech), but not others. (Linzen 2016, Gladkova et al. 2016, Ethayarajh et al. 2019a)
- Understanding analogy is an open area of research (Peterson et al. 2020)



Embeddings as a window onto historical semantics

Train embeddings on different decades of historical text to see meanings shift

~30 million books, 1850-1990, Google Books data



William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. Proceedings of ACL.



Embeddings Reflect Cultural Bias!

- Ask “Paris : France :: Tokyo : x”
 - x = Japan
- Ask “father : doctor :: mother : x”
 - x = nurse
- Ask “man : computer programmer :: woman : x”
 - x = homemaker

Bolukbasi, Tolga, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. "Man is to computer programmer as woman is to homemaker? debiasing word embeddings." In *NeurIPS*, pp. 4349-4357. 2016.

Algorithms that use embeddings as part of e.g., hiring searches for programmers, might lead to bias in hiring



Historical Embedding as a Tool to Study Cultural Biases

- Compute a **gender or ethnic bias** for each adjective
 - Example: how much closer the adjective is to "woman" synonyms than "man" synonyms, or names of particular ethnicities
 - Embeddings for **competence** adjective (*smart, wise, brilliant, resourceful, thoughtful, logical*) are biased toward men, a bias slowly decreasing 1960-1990
 - Embeddings for **dehumanizing** adjectives (barbaric, monstrous, bizarre) were biased toward Asians in the 1930s, bias decreasing over the 20th century.
- These match the results of old surveys done in the 1930s

Garg, N., Schiebinger, L., Jurafsky, D., and Zou, J. (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. Proceedings of the National Academy of Sciences 115(16), E3635–E3644.



We will see how we can use these separately trained word embeddings (or train/update embeddings on-the-fly) as we perform language modeling using **Neural Nets!**

