Word Vectors

Renan de Luca Avila & Alexandre Cadaval

September 24, 2025

Idea

Basics of word vectorization: Meaning comes from context

Process:
Co-occurrence $\rightarrow$ PMI $\rightarrow$ Dimensionality reduc. $\rightarrow$ Vec. Similarity

Toy case:

- **Tiny synthetic corpus**
- **“Fill in the blank”** question out of corpus
  (which answer relies on logical inference)
- **Word vectorization**
- Answer to the question based on **most similar word** in corpus

Corpus (5 sentences)

- `alice likes cheese and bread`
- `bob likes fish and rice`
- `cheese is dairy`
- `fish is seafood`
- `bread and rice are carbs`

Vocabulary order (rows/columns below use this order):
13 words: alice, and, are, bob, bread, carbs, cheese, dairy, fish, is, likes, rice, seafood

Co-occurrence Matrix C

Definition. $C[w, c]$[1] counts how often word w appears near context c [2] in a window of length 2.

	alice	and	are	bob	bread	carbs	cheese	dairy	fish	is	likes	rice	seafood
alice	0	1	0	0	1	0	1	0	0	0	1	0	0
and	1	0	1	1	**2**	1	1	0	1	0	**2**	2	0
are	0	1	0	0	1	1	0	0	0	0	0	1	0
bob	0	1	0	0	0	0	0	0	1	0	1	1	1
bread	1	**2**	1	0	0	1	1	0	0	0	1	**2**	0
carbs	0	1	1	0	1	0	0	0	0	0	0	1	0
cheese	1	1	0	0	1	0	0	1	0	1	1	0	0
dairy	0	0	0	0	0	0	1	0	0	1	0	0	0
fish	0	1	0	1	0	0	0	0	0	1	1	1	1
is	0	0	0	0	0	0	1	1	1	0	0	0	1
likes	1	**2**	0	1	1	0	1	0	1	0	0	1	1
rice	0	**2**	1	1	**2**	1	0	0	1	0	1	0	1
seafood	0	0	0	1	0	0	0	0	1	1	1	1	0

[1] Column and row headers follow the same order.

[2] w (words) are in the Y axis and c (context) are in the X axis.

From Counts to PMI and PPMI

Probabilities from counts:

$$p(w, c) = \frac{C[w, c]}{\sum_{u,v} C[u, v]}$$

$$p(w) = \sum_{c} p(w, c), \qquad p(c) = \sum_{w} p(w, c).$$

Pointwise Mutual Information (PMI):

$$\mathrm{PMI}(w, c) = \log_2 \frac{p(w, c)}{p(w)\, p(c)}.$$

Intuition: PMI up-weights word pairs that co-occur more often than chance; PPMI discards negative values (less-than-chance).

PMI Matrix

	alice	and	are	bob	bread	carbs	cheese	dairy	fish	is	likes	rice	seafood
alice	0.0	0.0	0.0	0.0	0.0	0.0	2.17	0.0	0.0	0.0	2.17	0.0	0.0
and	0.0	0.0	1.0	0.0	1.585	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0
are	0.0	1.0	0.0	0.0	0.0	**3.17**	0.0	0.0	0.0	0.0	0.0	1.585	0.0
bob	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.17	0.0	2.17	0.0	0.0
bread	0.0	1.585	0.0	0.0	0.0	0.0	1.17	0.0	0.0	0.0	0.0	1.17	0.0
carbs	0.0	0.0	**3.17**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.17	0.0
cheese	2.17	0.0	0.0	0.0	1.17	0.0	0.0	2.17	0.0	1.17	0.585	0.0	0.0
dairy	0.0	0.0	0.0	0.0	0.0	0.0	2.17	0.0	0.0	**2.755**	0.0	0.0	0.0
fish	0.0	0.0	0.0	2.17	0.0	0.0	0.0	0.0	0.0	1.17	0.585	0.585	2.17
is	0.0	0.0	0.0	0.0	0.0	0.0	1.17	**2.755**	1.17	0.0	0.0	0.0	**2.755**
likes	2.17	1.0	0.0	2.17	0.0	0.0	0.585	0.0	0.585	0.0	0.0	0.0	0.0
rice	0.0	1.0	1.585	0.0	1.17	2.17	0.0	0.0	0.585	0.0	0.0	0.0	0.0
seafood	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.17	**2.755**	0.0	0.0	0.0

Column-Drop (Simple Dimensionality Reduction)

Goal. Visualize word vectors in 2D without deep linear algebra background.

Idea. Drop columns (contexts) unrelated to the target question and **keep only 2 informative axes**:

$$\text{keep } \{\textbf{likes},\ \textbf{cheese}\} \quad \text{and drop all other columns.}$$

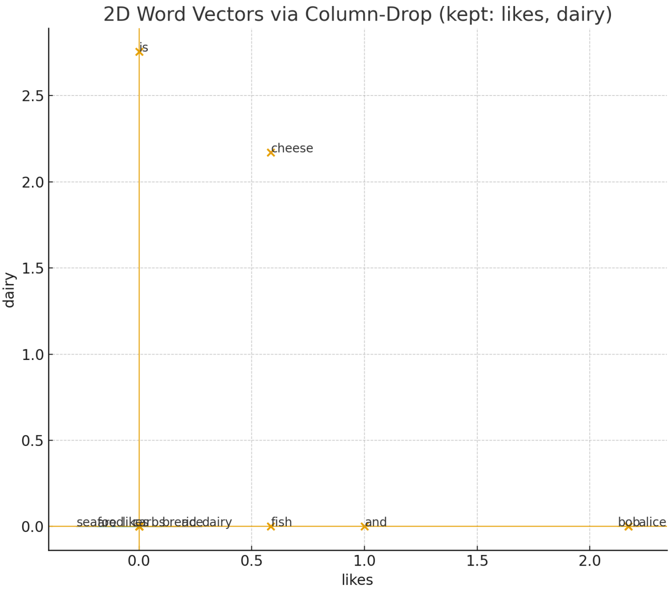
Why these? Our class question is multi-hop: **"___ likes dairy?"** Signal for `dairy` flows through `cheese` (via `cheese is dairy`), and `likes` ties to the subject.

PMI Restricted to Two Axes (kept: likes, dairy)

Each word becomes a 2D vector given by its row in the reduced PMI, resulting in the **Word embedding**:

	likes	dairy
alice	2.17	0.0
and	1.0	0.0
are	0.0	0.0
bob	2.17	0.0
bread	0.0	0.0
carbs	0.0	0.0
cheese	0.585	2.17
dairy	0.0	0.0
fish	0.585	0.0
is	0.0	2.755
likes	0.0	0.0
rice	0.0	0.0
seafood	0.0	0.0

2D Word Vectors (kept axes: likes, dairy)



Smarter Dimensionality Reduction (Context-Expansion)

Goal. Keep the visualization *2D* while preserving multi-hop signal for a query like

$$\text{“\_\_\_ \textbf{likes dairy?}”} \quad \Rightarrow \quad \text{anchors } \mathcal{A} = \{\text{likes}, \text{dairy}\}.$$

Step 1 — Anchor neighborhoods (from PPMI). Given the PPMI matrix M (rows = words w, columns = contexts c), define the *neighbors* of an anchor $A \in \mathcal{A}$ by

$$N(A) \;=\; \{\, c \;|\; M_{c,A} > 0 \,\}.$$

(These are the contexts with positive association to A.)

Step 2 — Bundle each anchor into one axis. Aggregate the columns in $N(A)$ into a single *bundle axis* by a weighted sum:

$$\mathrm{Axis}_A(w) \;=\; \sum_{c \in N(A)} \underbrace{M_{w,c}}_{\text{word} \times \text{context}} \cdot \underbrace{M_{c,A}}_{\text{anchor weight}} .$$

Interpretation: attention-like weighting—contexts more strongly tied to the anchor contribute more.

Smarter Dimensionality Reduction (Context-Expansion)

Step 3 — Normalize (optional). For each bundle axis, apply a z-score across words:

$$\widetilde{\mathrm{Axis}}_A(w) \;=\; \frac{\mathrm{Axis}_A(w) - \mu_A}{\sigma_A}$$

where μ_A, σ_A are the mean and std over w.

Step 4 — 2D word vectors. For anchors $\{\text{likes}, \text{dairy}\}$, define the 2D embedding

$$\mathbf{v}(w) \;=\; \left[\; \widetilde{\mathrm{Axis}}_{\text{likes}}(w),\; \widetilde{\mathrm{Axis}}_{\text{dairy}}(w) \;\right] \in \mathbb{R}^2.$$

Step 5 — Vector composition & decision. Form the query vector

$$\mathbf{q} \;=\; \mathbf{v}(\text{likes}) \;+\; \mathbf{v}(\text{dairy}),$$

and pick the subject s that maximizes cosine similarity:

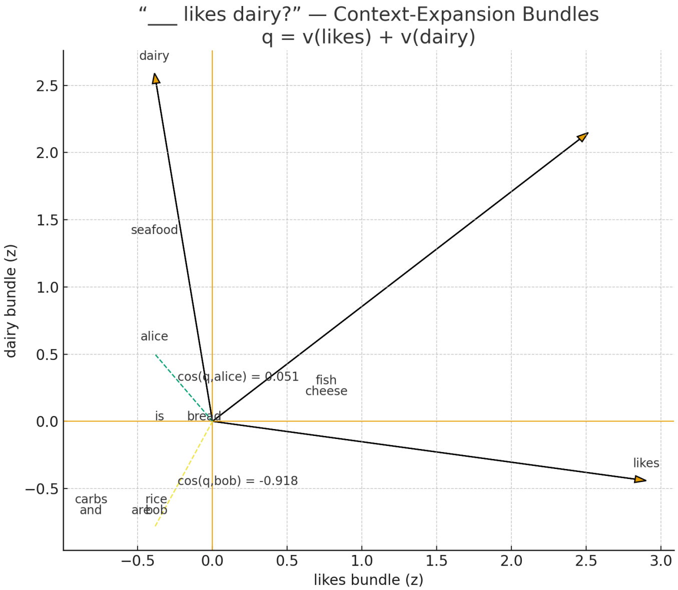
$$\hat{s} \;=\; \arg \max_{s \in \{\text{alice}, \text{bob}, \dots\}} \frac{\mathbf{q} \cdot \mathbf{v}(s)}{\|\mathbf{q}\| \, \|\mathbf{v}(s)\|}.$$

Smarter dimensionality reduction result

Why it helps. Bundling brings informative neighborhood likes $\to$ {alice, bob, cheese, ...}, dairy $\to$ {cheese, is} capturing **likes**$\to$**cheese**$\to$**dairy** in an explainable 2D space.

	likes_{ctx}	dairy_{ctx}
alice	-0.388	0.504
and	-0.813	-0.791
are	-0.478	-0.791
bob	-0.388	-0.791
bread	-0.053	-0.093
carbs	-0.813	-0.791
cheese	0.763	0.095
dairy	-0.388	2.59
fish	0.763	0.095
is	-0.355	-0.093
likes	2.902	-0.442
rice	-0.364	-0.791
seafood	-0.388	1.296

2D Word Vectors (context-expansion)



Conclusion

- Question "___ likes dairy" $\rightarrow$ Answer: **Alice**! Because Alice has higher cosine similarity.
- Co-occurrence $\Rightarrow$ PMI produces **transparent count-based signals**.
- **Dropping columns** gives a simple 2D view, **multi-hop inference** may emerge via vector addition, but depending on the corpus it might not keep enough information.
- A better dimensionality reduction technique is the **context-expansion**, which provides more information, enough to produce logical inference.
- In practice, state-of-the-art techniques prefer **automatic dimensionality reduction** (SVD/PCA) instead of manual column selection or feature combination.

Appendix: SVD and PCA (High-Level)

SVD (Singular Value Decomposition). Any matrix M can be factored as

$$M \; = \; U \Sigma V^\top,$$

where columns of U/V are orthonormal and Σ has nonnegative *singular values*. Truncating to k largest singular values (*rank-k SVD*) gives a low-dimensional approximation that preserves most variance (energy).

PCA (Principal Component Analysis). Finds orthogonal directions of maximum variance in the data, projecting to a few principal components. PCA on word-context features is closely related to SVD on the (centered) data matrix.

Connection. In large vocabularies we often apply SVD/PCA to PPMI (or related) matrices to obtain dense, low-dimensional embeddings automatically (instead of manually dropping columns).

Appendix: Modern Generalizations (Word2Vec & beyond)

Word2Vec (SGNS/CBOW). Trains a simple neural model to predict contexts from words (skip-gram) or words from contexts (CBOW). It *implicitly* factorizes a shifted PMI/PPMI matrix, but **without explicitly building it**, making it efficient at scale.

GloVe. Minimizes a weighted loss over word–context co-occurrence counts, explicitly relating embedding dot-products to log-co-occurrences.

Contextual embeddings (BERT, GPT). Instead of one static vector per word, produce **context-dependent** embeddings for each token in its sentence. These models subsume distributional signals and multi-hop reasoning in larger learned representations.

Takeaway: Our column-drop/context-expansion 2D demo is the simplest transparent case; SVD/PCA generalize it linearly; Word2Vec/GloVe scale it; modern transformers go beyond static co-occurrence to context-sensitive meaning.