

Vector Semantics and Embeddings

Robert Minneker

2026-01-27

Sources

Content derived from: J&M Ch. 5

Reminders

! Important

- The first deliverable for Project 1 is **due today**
- Only one person from each group needs to submit the deliverable
- A1 due Thursday

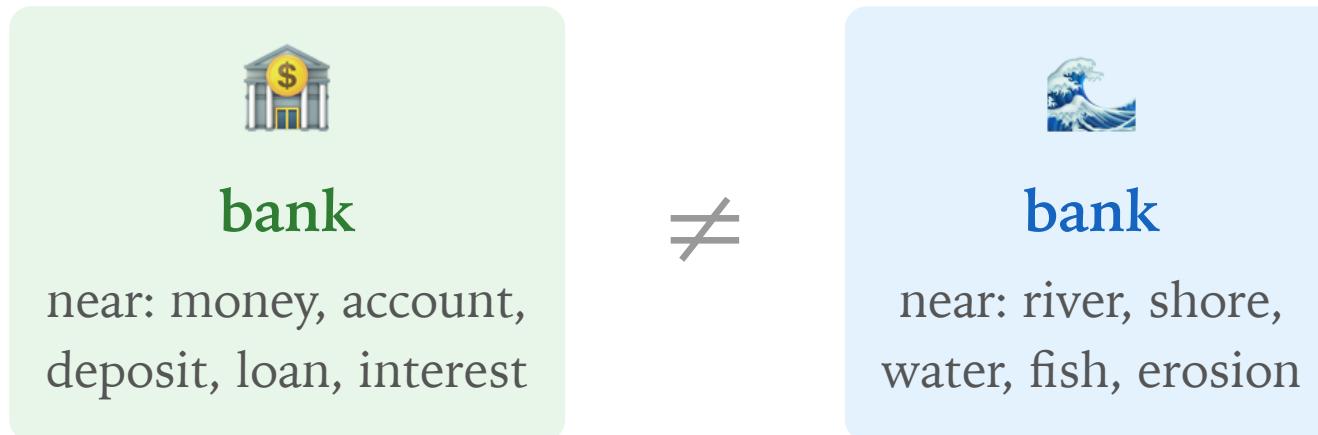
Project Team Dynamics

- All of your group members name's must be on the submission: **Check Canvas and make sure you know who is in your group!**
 - If your name is not included in a project submission and you cannot provide evidence of you reaching out to your group, **you get no credit.**
 - If your submission is missing the name of someone in your group and you cannot provide evidence that you reached out to them, **you will get no credit.**
 - If your group is not responding, make every effort to complete the first checkpoint to the best of your ability, and we will grade leniently based on how many people worked on it.

Part 1: Foundations of Vector Semantics

“You shall know a word by the company it keeps” — Firth

- Words are characterized by their distributional properties, not in isolation
- “Bank” near “river” vs. “bank” near “money” reveals context-dependent meaning



The distributional hypothesis

- Harris (1954): If two words appear in similar contexts, their meanings are likely similar

$$\text{contexts}(w_1) \approx \text{contexts}(w_2) \implies \text{meaning}(w_1) \approx \text{meaning}(w_2)$$

- No dictionary definitions needed—meaning emerges from usage patterns

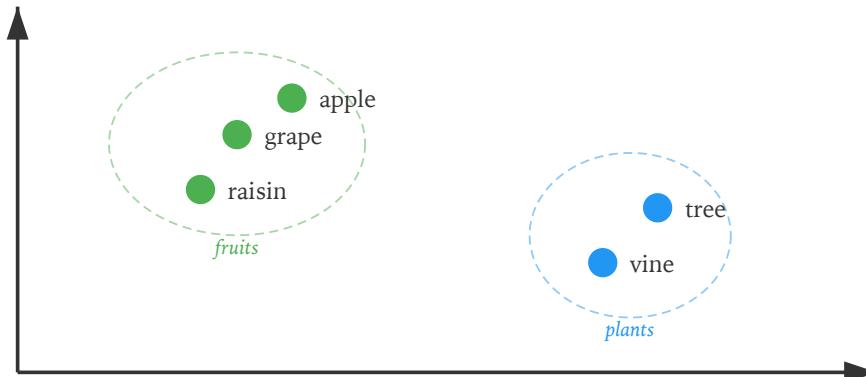
The antonyms problem: A limitation

- “Hot” and “cold” appear in **very similar contexts**:
 - “The water was ____”
 - “It’s ____ outside today”
 - “____ temperature”, “____ weather”
- But they have **opposite meanings**!

Key limitation: Distributional similarity captures *topical relatedness*, not all aspects of meaning. Antonyms, hypernyms, and other semantic relations need additional modeling.

Each word maps to a vector in semantic space

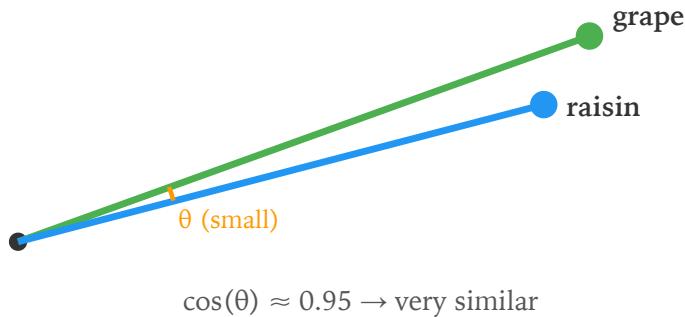
- Vocabulary V ; each word $w \in V$ maps to $\mathbf{v}_w \in \mathbb{R}^d$
- Similar contexts \rightarrow nearby vectors \rightarrow similar meanings



Cosine similarity measures semantic closeness

$$\text{sim}(w_1, w_2) = \cos(\theta) = \frac{\mathbf{v}_{w_1} \cdot \mathbf{v}_{w_2}}{\|\mathbf{v}_{w_1}\| \|\mathbf{v}_{w_2}\|}$$

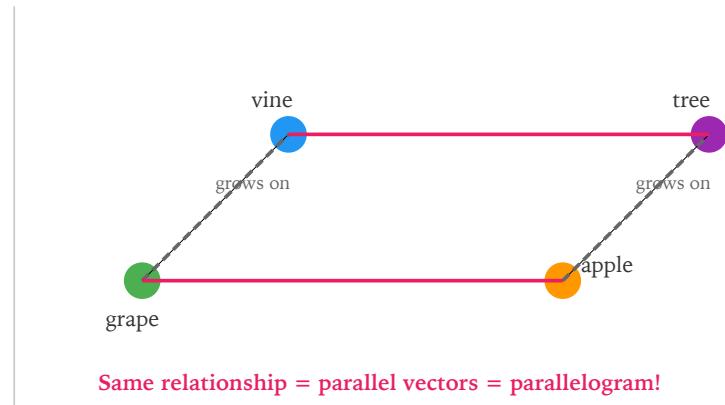
- **Direction** matters more than magnitude for meaning



Vector arithmetic captures relationships

- Relationships encoded as **directions** in embedding space

$$\vec{v}_{\text{grape}} - \vec{v}_{\text{vine}} + \vec{v}_{\text{tree}} \approx \vec{v}_{\text{apple}}$$



The Conceptual Leap: Meaning Becomes Geometry

A Paradigm Shift in Linguistics

Traditional View

Meaning = symbolic definitions
"cat" defined by features:
+animate, +furry, +feline...



Vector Semantics

Meaning = position in space
"cat" = [0.2, -0.5, 0.8, ...]
300 learned dimensions

Why this matters:

- Semantic operations become **mathematical operations**
- Similarity → distance/angle; Analogy → vector arithmetic
- Meaning can be **computed**, not just looked up
- Enables **generalization** to unseen combinations

Part 2: Classical Count-Based Methods

TF-IDF: The Document Retrieval Baseline

- **Term Frequency (TF):** How often does word w appear in document d ?
- **Inverse Document Frequency (IDF):** How rare is word w across all documents?

$$\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \log \frac{|D|}{\text{DF}(w)}$$

High TF-IDF

"photosynthesis" in a biology paper
→ Frequent here, rare elsewhere
→ **Discriminative**

Low TF-IDF

"the" in any document
→ Frequent everywhere
→ **Not informative**

Limitation: TF-IDF captures document-level topicality, not fine-grained word similarity

Term-document matrices

- Matrix $\mathbf{X} \in \mathbb{R}^{|V| \times |D|}$ where X_{wd} counts word w in document d
- Each word = high-dimensional, sparse vector

	Doc 1 wine review	Doc 2 botany text	Doc 3 recipe
grape	15	8	3
wine	25	2	5
tree	0	18	1

"grape" vector: [15, 8, 3] → similar to "wine" [25, 2, 5]

Word-context matrices with sliding windows

- Matrix $\mathbf{C} \in \mathbb{R}^{|V| \times |V|}$: C_{wv} = count of v near w
- Window size controls what “near” means

The **fresh** **grape** **juice** tastes great

Window ± 1 : fresh, juice \rightarrow syntactic neighbors

Window ± 5 : The, fresh, juice, tastes, great \rightarrow topical neighbors

PMI: Measuring association strength

Pointwise Mutual Information: How much more often do words co-occur than expected?

$$\text{PMI}(w, c) = \log_2 \frac{P(w, c)}{P(w) \cdot P(c)}$$

Worked example

Corpus: 1000 word pairs

$$P(\text{grape, wine}) = 8/1000 = 0.008$$

- “grape” appears in 20 pairs
- “wine” appears in 50 pairs
- “grape-wine” co-occurs 8 times

$$P(\text{grape}) \times P(\text{wine}) = 0.02 \times 0.05 = 0.001$$

$$\text{PMI} = \log_2(0.008/0.001) = \log_2(8) = 3 \text{ bits}$$

Interpretation: 8× more likely than chance → strong association!

PPMI: Fixing negative infinity

Problem with PMI: If $P(w, c) = 0$, then $\text{PMI} = -\infty$

Solution: Positive PMI (PPMI) — clip negative values to zero

$$\text{PPMI}(w, c) = \max(0, \text{PMI}(w, c))$$

Why clip at zero?

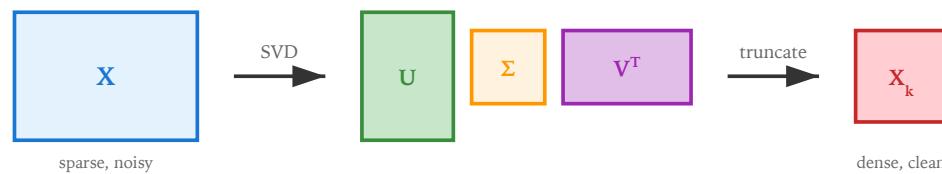
- Negative PMI often unreliable (sparse data)
- “Never co-occurred” \neq “semantically opposite”
- Zeros are easier to handle (sparse matrices)

PPMI Matrix

- Sparse (mostly zeros)
- High-dimensional ($|V| \times |V|$)
- Better than raw counts
- Foundation for SVD/LSA

LSA: Dimensionality reduction reveals latent structure

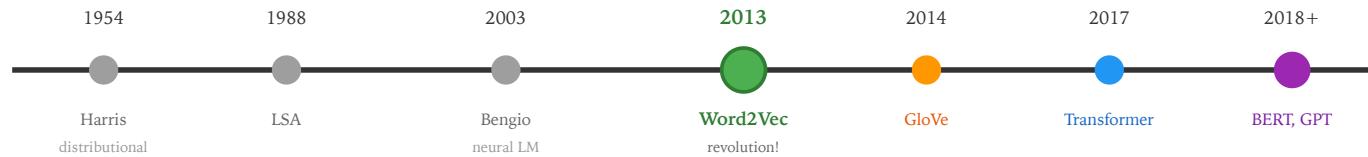
- SVD factorizes the matrix: $X = U\Sigma V^T$
- Truncate to k dimensions: keeps most important patterns



- “Grape” and “vineyard” become close even without direct co-occurrence
- Shared contexts (“wine”, “harvest”) create latent similarity

Part 3: Neural Embedding Methods

Historical timeline: The evolution of embeddings



Word2Vec: A Framework, Not a Single Algorithm

Word2Vec = Framework with Multiple Components

Architectures

- **SGNS**: Skip-gram + Negative Sampling
- **CBOW**: Continuous Bag of Words

Training Tricks

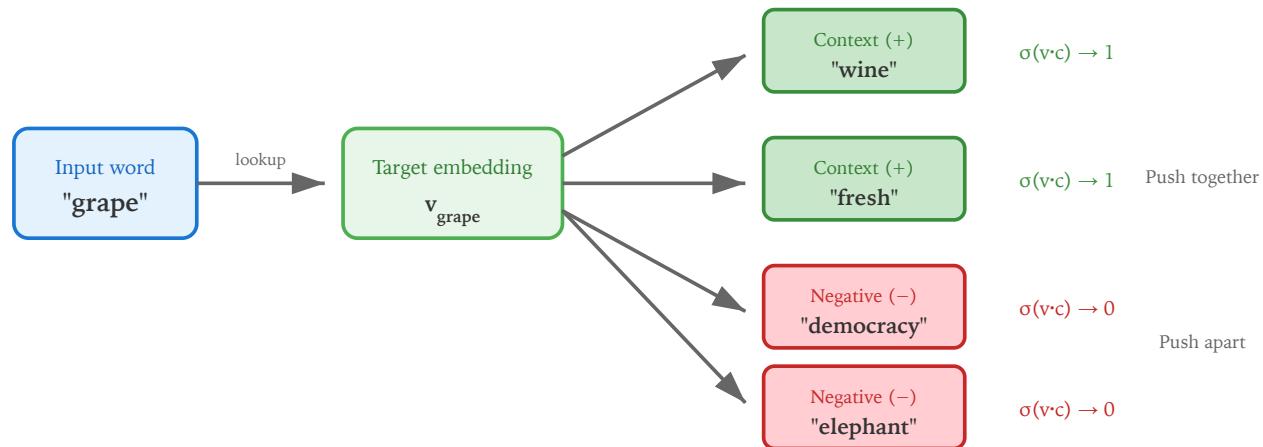
- Negative sampling
- Hierarchical softmax
- Subsampling frequent words

Key insight: Mikolov et al. (2013) introduced a family of methods, not one algorithm

- “Word2Vec” often refers to SGNS specifically (the most popular variant)
- Both architectures learn by **predicting** rather than counting

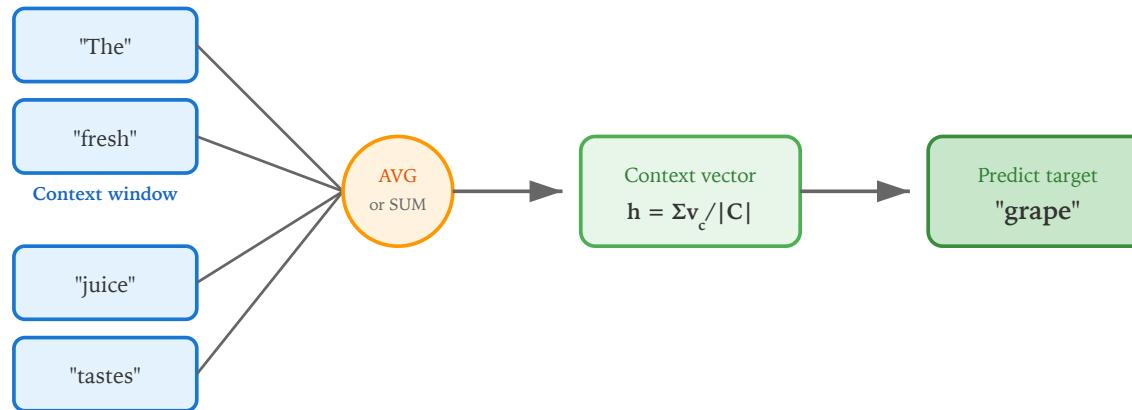
Skip-Gram with Negative Sampling (SGNS)

Core idea: Given a target word, predict its context words



Continuous Bag of Words (CBOW)

Core idea: Given context words, predict the target word



CBOW vs Skip-gram:

Aspect	CBOW	Skip-gram (SGNS)
Input	Context words	Target word
Output	Target word	Context words
Speed	Faster (1 prediction)	Slower (k predictions)
Rare words	Worse	Better

The Deep Connection: SGNS \approx Implicit PMI Factorization

! Levy & Goldberg (2014): A Landmark Discovery

SGNS implicitly factorizes: $\mathbf{W} \cdot \mathbf{W}'^T \approx \text{PMI}(w, c) - \log k$

What this means:

- Skip-gram with negative sampling learns embeddings whose dot product approximates **shifted PMI**
- The “prediction” objective recovers the same statistical information as “counting”
- Neural and count-based methods are **two sides of the same coin**

Methods Comparison: The Full Picture

Method	What it captures	Matrix factorized	Training
TF-IDF	Document-level topicality	Weighted term-doc	Direct computation
PMI/PPMI + SVD	Word co-occurrence strength	PPMI matrix	Count → SVD
Word2Vec (SGNS)	Shifted PMI (implicitly)	PMI – $\log k$	SGD prediction
GloVe	Log co-occurrence (explicit)	$\log X$ weighted	SGD regression

GloVe: Explicit Global Optimization

GloVe's insight: If SGNS implicitly factorizes PMI, why not do it explicitly?

Word2Vec (SGNS)

- Learns by **predicting** context
- Stochastic: samples (word, context) pairs
- **Implicitly** factorizes PMI
- Online learning possible

GloVe

- Learns by **regression** on log-counts
- Batch: uses full co-occurrence matrix
- **Explicitly** minimizes reconstruction error
- Weighted by $f(X_{ij})$ to handle frequent pairs

Result: Similar embeddings, different training dynamics

Static vs contextualized embeddings

Static (Word2Vec, GloVe)

$$f : V \rightarrow \mathbb{R}^d$$

“The **bank** was steep”

$$\rightarrow [0.2, -0.5, \dots]$$

“The **bank** was closed”

$$\rightarrow [0.2, -0.5, \dots]$$

Same vector!

Contextualized (BERT)

$$f : (w, C) \rightarrow \mathbb{R}^d$$

“The **bank** was steep”

$$\rightarrow [0.3, 0.1, \dots]$$

“The **bank** was closed”

$$\rightarrow [-0.2, 0.4, \dots]$$

Different vectors!

Transformers: Self-attention for contextualization

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V$$

- Every token attends to every other token
- Multiple layers refine representations
- Pre-training: Masked LM (BERT), autoregressive (GPT)

Concept Check

Why might word analogy tasks (grape - vine + tree = apple) work BETTER with static embeddings than contextualized ones?

Part 4: Evaluation

Similarity vs. relatedness: Know what you're measuring

Word Pair	WordSim-353 (relatedness)	SimLex-999 (similarity)
car - gasoline	HIGH	LOW
coffee - cup	HIGH	LOW
car - automobile	HIGH	HIGH

- Relatedness (WordSim): Are these words associated?
- Similarity (SimLex): Are these words interchangeable?

When analogies fail

- “king - man + woman = queen” is the famous success story
- But many analogies don’t work:

doctor - man + woman = ?

Often returns "nurse" (bias!)

Paris - France + Japan = ?

Sometimes "Tokyo", often noise

bigger - big + small = ?

Rarely returns "smaller"

Reality check: Google analogy dataset accuracy is ~60-75%, not 95%.
Analogy arithmetic is a useful probe, not a reliable tool.

Extrinsic evaluation: The real test

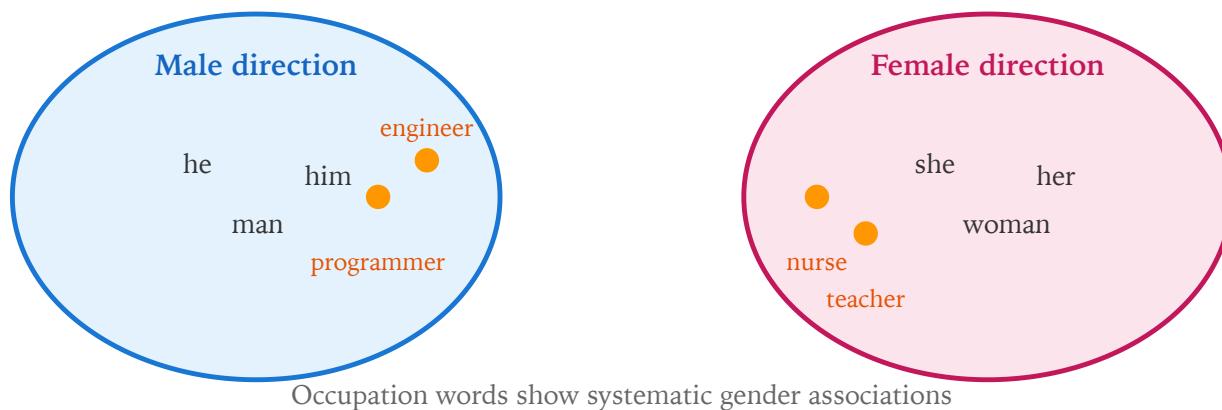
- **Intrinsic:** How good are the embeddings themselves?
- **Extrinsic:** How much do embeddings help downstream tasks?

Task	Without pretrained	With Word2Vec	With BERT
Sentiment	78%	84%	93%
NER	81%	88%	95%
Question Answering	65%	72%	89%

Part 5: Ethics and Bias

Embeddings encode societal biases

- Training data reflects historical biases
- Embeddings learn and amplify these patterns



WEAT: Measuring embedding bias

$$s(X, Y, A, B) = \frac{1}{|X|} \sum_{x \in X} s(x, A, B) - \frac{1}{|Y|} \sum_{y \in Y} s(y, A, B)$$

- Compare associations between word sets and attribute sets
- Parallels the psychological Implicit Association Test (IAT)

Example:

X = {programmer, engineer, scientist}

Y = {nurse, teacher, librarian}

A = {he, him, man}

B = {she, her, woman}

Finding: X more associated with A; Y more associated with B → gender bias

Debiasing: Partial solutions

Projection method

1. Identify "gender direction"
2. Project it out of all word vectors
3. "programmer" moves to neutral position

Limitations

- May hide rather than remove bias
- Some words "should" be gendered
- Bias can reappear in fine-tuning

Discussion

If occupation words like “engineer” are closer to “man” than “woman” in embedding space:

1. What downstream harms might this cause? (Think: hiring systems, search engines)
2. Can we ever create a truly “unbiased” language model?
3. Who should decide what counts as bias?

Summary

The Paradigm Shift

Meaning → Geometry
Semantic ops → Vector ops

Word2Vec = Framework

SGNS, CBOW architectures
Implicitly factorizes PMI

Methods Unified

TF-IDF → PMI → Word2Vec → GloVe
Count ↔ Prediction equivalence

Limitations & Ethics

Antonyms problem, analogy failures
Embeddings encode societal bias

Key takeaway: Vector semantics transforms meaning into geometry—powerful but imperfect.