Word Embeddings: A Visual Guide

Teaching Computers to Understand Language

Prof. Dr. Jörg Osterrieder

Methods and Algorithms — MSc Data Science

Spring 2026

How Do You Teach a Computer to Read?

How do you teach a computer to ***understand words****?*

Today: **Word Embeddings** — turning text into numbers that capture meaning.

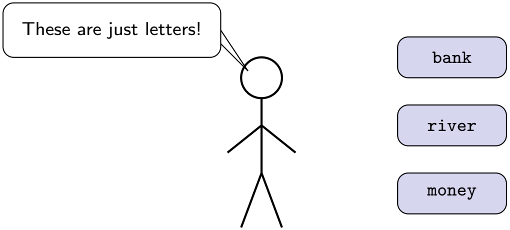
XKCD #1838 by Randall Munroe (CC BY-NC 2.5)

What You Will Learn Today

1. **Explain what word embeddings do** — in plain English
2. **Show why simple numbering fails** and embeddings succeed
3. **Use embeddings for real finance tasks**

Everything is explained with pictures and analogies.

The Problem: Computers Don't Understand Words!



A computer sees text as a string of characters. It has no concept of meaning.

Computers need numbers, not words. But which numbers capture meaning?

Why Not Just Number Every Word?

Simple Numbering

- cat = 1, dog = 2, car = 3
- dog is NOT "between" cat and car
- Numbers imply an order that doesn't exist

Embeddings

- cat = [0.9, 0.1]
- dog = [0.8, 0.2]
- car = [0.1, 0.7]

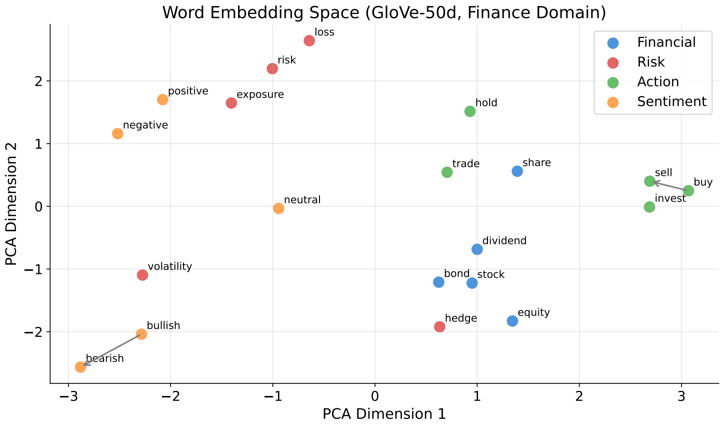
cat and dog are close (both animals). car is far away (different concept).

Key Insight

Embeddings capture relationships. Simple numbers don't.

Embeddings are learned from data — the computer discovers which words are similar by seeing them in context.

Words Can Become Points in Space!

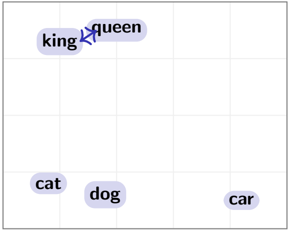


https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/01_word_embedding_space

- Each word is placed at a specific location (coordinates)
- Similar words end up close together
- The computer learned these positions from reading text

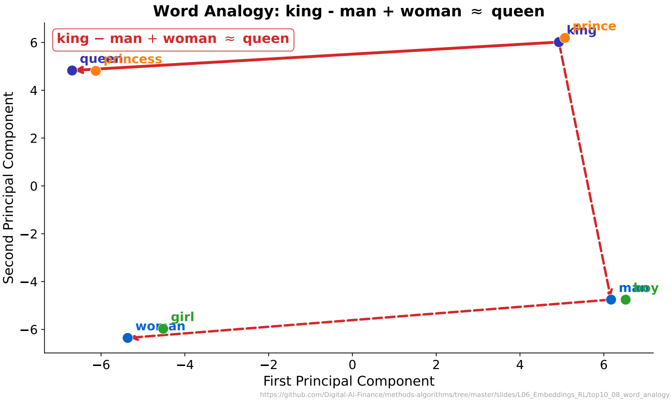
Word embeddings turn words into vectors — lists of numbers that capture meaning.

Embeddings = Giving Words a Home Address



Similar words live in the same neighborhood.

In a city, nearby addresses mean nearby locations. In embeddings, nearby vectors mean similar meanings.



- King − Man + Woman = Queen
- Paris − France + Germany = Berlin
- Embeddings capture analogies as vector arithmetic

This is the "aha moment" of embeddings: meaning has direction. Gender, country, tense — all are vectors.

How Embeddings Are Made

1. Read Text
Feed the model
billions of words

→

2. Learn Context
Which words appear
near each other?

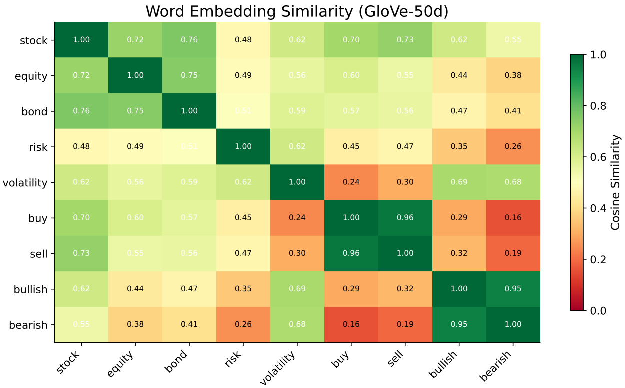
→

3. Assign Coordinates
Similar context
= similar numbers

That's it. The computer reads text and figures out which words are similar.

Word2Vec, GloVe, and BERT all follow this principle: learn from context.

Who Is Similar to Whom?

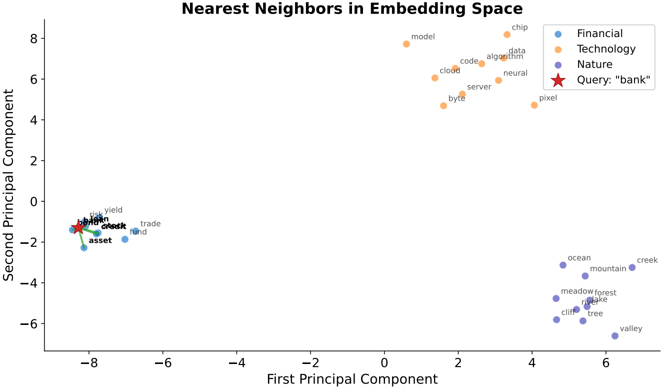


https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/02_similarity_heatmap

- Dark squares = highly similar words
- Light squares = unrelated words
- The computer learned this from reading text alone — no human labeled it

Cosine similarity measures the angle between two word vectors. Small angle = similar meaning.

What Words Live Nearby?

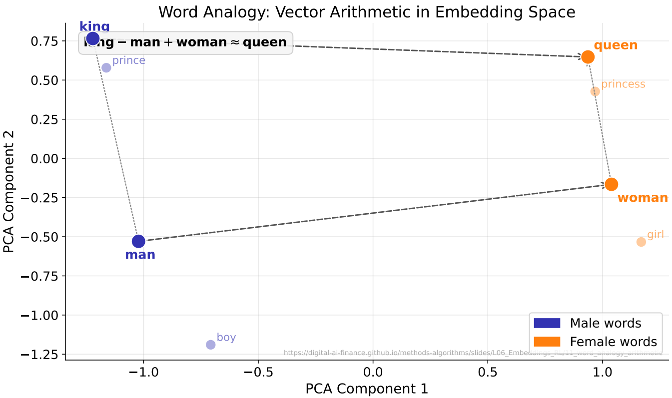


https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/top10_13_embedding_neighbors

- Shows the closest words to a query word
- Validates that the embedding captured real meaning
- Similar words = similar contexts in training data

The nearest-neighbor test is a quick way to check if embeddings captured meaning.

Vector Math Preserves Meaning



- Visual proof that vector math preserves relationships
- Directions in the space encode concepts like gender and geography
- Not magic — learned from co-occurrence patterns

Analogy arithmetic works because embeddings encode relational structure.

Why Bankers Care About Embeddings

- **Sentiment Analysis**: Is this news article positive or negative for the stock?
- **Fraud Detection**: Does this transaction description look suspicious?
- **Document Search**: Find contracts similar to this one

Bottom Line

Embeddings let machines understand financial text — reports, news, filings.

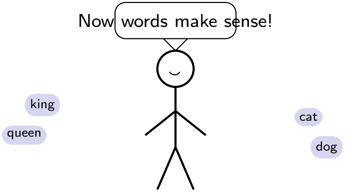
JPMorgan, Goldman Sachs, and Bloomberg all use NLP with embeddings for financial analysis.

Try It Yourself: 3 Lines of Python

```
from gensim.models import Word2Vec
model = Word2Vec(sentences, vector_size=100)
similar = model.wv.most_similar("bank")
```

That's it — three lines to train word embeddings on your own text.

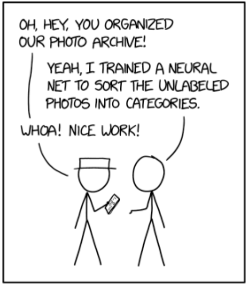
pip install gensim. For pre-trained embeddings, try gensim.downloader.



1. Embeddings turn words into numbers that capture meaning
2. Similar words get similar numbers
3. Finance uses: sentiment, fraud detection, document search

Embeddings: giving every word a home address in number space.

Go Teach a Machine Something



ENGINEERING TIP:
WHEN YOU DO A TASK BY HAND, YOU CAN TECHNICALLY SAY YOU TRAINED A NEURAL NET TO DO IT.

XKCD #2173 by Randall Munroe (CC BY-NC 2.5). Next: try the Jupyter notebook with real financial text data!