Word Embeddings

Mini-Lecture: Teaching Computers to Understand Words

Methods & Algorithms

MSc Data Science – Spring 2026

The Machine Learning Pipeline

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

Words as Vectors: From Symbols to Numbers

The Representation Problem

- Computers need numbers, not words – how do we encode "bank", "risk", "default"?
- **One-hot encoding**: each word is a sparse vector with a single 1. Vocabulary of 50k = 50k-dimensional vectors.
- Problem: one-hot treats "bank" and "finance" as equally different from "banana"
- **Dense embeddings**: map each word to a short vector (50–300 dims) where similar words are nearby

One-hot (sparse, 50k dims):

```
bank = [0,0,1,0,...,0]
risk = [0,0,0,0,...,1]
```

Embedding (dense, 300 dims):

```
bank = [0.2, -0.1, ...]
risk = [0.3, -0.2, ...]
```

Similar words → nearby vectors

Distributional Hypothesis

"You shall know a word by the company it keeps" (Firth, 1957). Words in similar contexts have similar meanings.

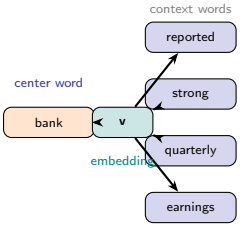
Embeddings compress meaning into geometry – distance = semantic similarity

Word2Vec: Learning Embeddings from Context

The Skip-gram Model

- Given a center word, predict its surrounding context words
- Training objective: maximize $P(\text{context} \mid \text{center})$
- The hidden layer weights become the word embeddings
- Trained on billions of words of text – embeddings emerge from co-occurrence patterns

Example: "The bank reported strong quarterly earnings" – "bank" and "earnings" become neighbors in embedding space.



Insight

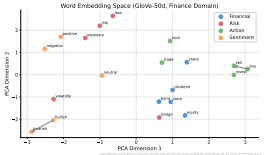
Word2Vec does not understand language – it learns statistical co-occurrence. But this is enough to capture remarkable semantic structure.

Word2Vec (Mikolov et al., 2013) trained on Google News: 100 billion words, 3 million word vectors

Semantic Similarity: Words That Cluster Together

Measuring Word Similarity

- **Cosine similarity**: $\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$
- Range: -1 (opposite) to $+1$ (identical meaning)
- Finance words cluster: "stock", "equity", "shares" are neighbors
- Unrelated words are distant: "stock" vs "elephant"



Insight

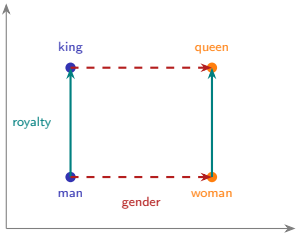
Cosine similarity ignores vector magnitude – it measures direction only. Two documents about finance point the same way, regardless of length.

In embedding space, semantic relationships become geometric relationships – similarity = proximity

Word Analogies: Vector Arithmetic on Meaning

The Famous Analogy Test

- $\vec{\text{king}} - \vec{\text{man}} + \vec{\text{woman}} \approx \vec{\text{queen}}$
- The "gender direction" is captured as a consistent vector offset
- Finance analogies work too: $\vec{\text{stock}} - \vec{\text{equity}} + \vec{\text{debt}} \approx \vec{\text{bond}}$



Insight

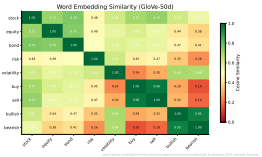
Analogies reveal that embeddings encode relational structure, not just similarity. The same vector offset maps "man:woman" as "king:queen".

Analogy accuracy is a standard benchmark for embedding quality – good embeddings score 60–75% on the Google analogy test

Cosine Similarity Between Finance Words

Similarity Heatmap

- Real cosine similarities from pre-trained Word2Vec
- Finance terms cluster by semantic field: risk words together, asset words together
- Cross-domain pairs (e.g., “volatility” vs “dividend”) show moderate similarity
- This structure enables document classification without labeled data



Insight

The heatmap reveals that embeddings capture domain-specific structure – finance words form a coherent subspace.

Pre-trained embeddings transfer knowledge from general text to domain-specific tasks – fine-tuning adapts them further

Pre-trained Models: Standing on the Shoulders of Giants

From Static to Contextual Embeddings

Model	Type	Dimensions	Key Feature
Word2Vec	Static	100–300	Fast, simple, one vector per word
GloVe	Static	50–300	Global co-occurrence statistics
ELMo	Contextual	1024	Different vector per context
BERT	Contextual	768	Bidirectional, fine-tunable

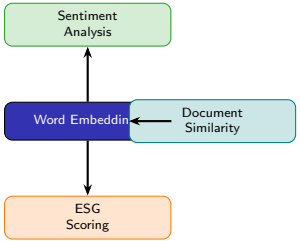
- **Static**: "bank" has ONE vector regardless of context (river bank vs investment bank)
- **Contextual**: "bank" gets different vectors depending on surrounding words
- For most finance NLP tasks, fine-tuned BERT outperforms static embeddings

FinBERT (Araci, 2019) is BERT fine-tuned on financial text – state of the art for financial sentiment analysis

Finance Application: NLP for Financial Text

Where Embeddings Meet Finance

- **Sentiment analysis**: classify earnings calls, news, analyst reports as positive/negative/neutral
- **Document similarity**: find SEC filings similar to a reference document using cosine similarity of averaged embeddings
- **ESG scoring**: embed company reports and measure proximity to ESG-related concept vectors



Insight

Embeddings turn unstructured text into structured features. Once text is a vector, any ML algorithm can process it.

Loughran & McDonald (2011) showed that general-purpose sentiment lexicons fail on financial text – domain-specific embeddings fix this

Summary: Word Embeddings in 4 Takeaways

1. **Representation**: Dense embeddings map words to short vectors where semantic similarity = geometric proximity, replacing sparse one-hot encoding
2. **Word2Vec**: Skip-gram learns embeddings by predicting context words – co-occurrence statistics produce semantic structure
3. **Analogies**: Vector arithmetic captures relational meaning (king − man + woman = queen), enabling zero-shot reasoning
4. **Finance**: Embeddings power sentiment analysis, document similarity, and ESG scoring – turning unstructured text into ML-ready features

Next Steps

Explore the deep dive slides for negative sampling, GloVe derivation, and contextual embeddings. Try the notebook to train Word2Vec on financial news.

Embeddings are the bridge between human language and machine learning – the foundation of modern NLP