L06: Embeddings & RL

Text Representations and Sequential Decision Making

Methods and Algorithms

MSc Data Science

Spring 2026

Outline

Three zones: Introduction, Core Content (PMSP), and Wrap-Up

When All You Have Is Linear Algebra...

"You pour the data into this big pile of linear algebra, then collect the answers on the other side."

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

The Text Data Challenge

Financial text is everywhere—but machines cannot read

- Earnings calls, analyst reports, and news articles contain market-moving signals
- Text is unstructured: no rows, no columns, no obvious features
- We need to capture semantic meaning—"bullish" should be close to "positive outlook"

The core question: How do we convert words into numbers that preserve meaning?

Embeddings turn text into numbers that ML models can process

The Sequential Decision Challenge

Trading is not a one-shot prediction—it is a sequence of decisions

- A portfolio manager makes buy, sell, and hold decisions every day
- Each action has delayed consequences: today's trade affects next week's P&L
- The explore-vs-exploit dilemma: stick with what works or try something new?

The core question: How do we learn an optimal strategy from trial and error?

Reinforcement learning: learning from trial and error with delayed rewards

Why Banks and Asset Managers Care

Embedding Applications in Finance

- **Sentiment analysis:** Classify news as positive/negative for trading signals
- **Document similarity:** Find related filings, detect duplicate reports
- **Entity recognition:** Extract company names, tickers, and financial events

Reinforcement Learning Applications in Finance

- **Algorithmic trading:** Learn buy/sell/hold policies from market data
- **Portfolio rebalancing:** Optimize allocation over time with transaction costs
- **Optimal execution:** Minimize market impact when filling large orders

---

Both techniques address real problems in quantitative finance and risk management

Learning Objectives

By the end of this lecture, you will be able to:

1. **Derive** the Skip-Gram objective and analyze the negative sampling approximation
2. **Evaluate** static vs. contextual embeddings for domain-specific NLP tasks (e.g., FinBERT)
3. **Analyze** the convergence properties of Q-learning and the role of the exploration–exploitation tradeoff
4. **Critique** RL-based trading strategies and their limitations (transaction costs, non-stationarity, overfitting)

Finance Application: Sentiment-driven trading signals and sequential portfolio optimization

---

Bloom's taxonomy levels 4–5: Analyze, Evaluate, Derive, Critique

The Business Problem

Text Data: One-Hot Encoding Wastes Dimensions

- A vocabulary of 50,000 words $\Rightarrow$ 50,000-dimensional sparse vectors
- No notion of similarity: “bank” and “financial institution” are equally distant
- Embeddings compress words into dense vectors of 100–768 dimensions

Sequential Decisions: Delayed Reward Signals

- Supervised learning needs immediate labels; trading profit is realized days later
- The agent must learn which past actions led to eventual gains or losses
- RL explicitly models the credit-assignment problem via reward discounting

Embeddings solve the text problem; RL solves the sequential decision problem

Key Equations

Embeddings — Skip-Gram Objective:

$$\max \sum_{t=1}^{T} \sum_{\substack{-c \leq j \leq c \\ j \neq 0}} \log p(w_{t+j} \mid w_t)$$

Cosine Similarity:

$$\text{sim}(u, v) = \frac{u \cdot v}{\|u\| \, \|v\|}$$

Reinforcement Learning — Bellman Equation:

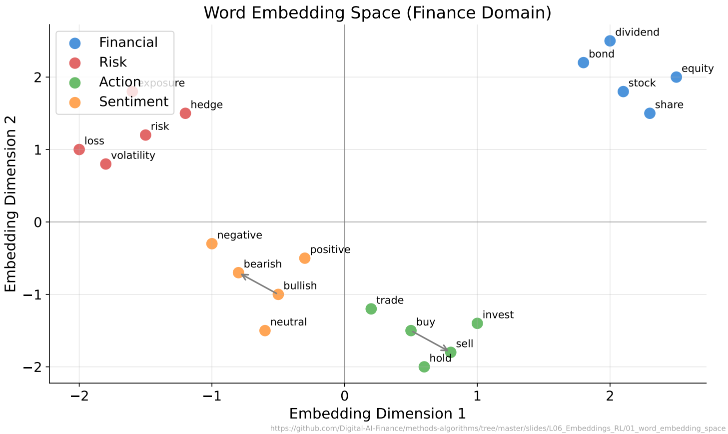
$$Q^*(s, a) = \mathbb{E}\left[r + \gamma \max_{a'} Q^*(s', a') \mid s, a\right]$$

TD Update (Q-Learning):

$$Q(s, a) \leftarrow Q(s, a) + \alpha\left[r + \gamma \max_{a'} Q(s', a') - Q(s, a)\right]$$

These four equations are the mathematical backbone of this lecture

Word Embedding Space



Similar words cluster together in embedding space

The Distributional Hypothesis

"You shall know a word by the company it keeps." — J.R. Firth (1957)

- Words appearing in similar contexts have similar meanings
- **Dense** vectors (100–300 dims) replace **sparse** one-hot vectors (50,000+ dims)
- **Word2Vec core idea:** Train a shallow neural network to predict context words from a target word (Skip-Gram) or vice versa (CBOW)

Word2Vec: the breakthrough that made word embeddings practical (Mikolov et al., 2013)

Static vs. Contextual Embeddings

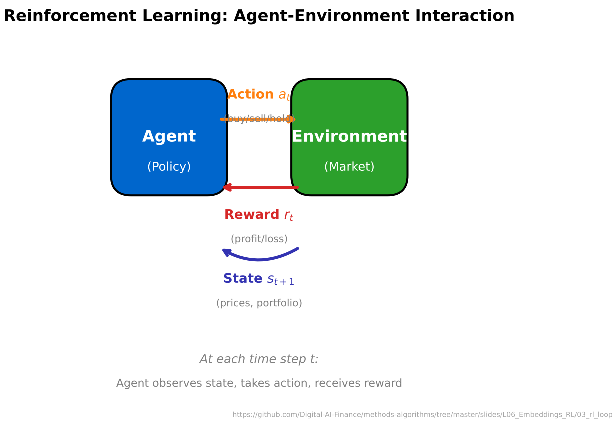
Static Embeddings (Word2Vec, GloVe)

- One fixed vector per word, regardless of context
- "bank" always maps to the same point—whether river bank or investment bank

Contextual Embeddings (BERT, FinBERT)

- Different vector for each occurrence, depending on surrounding sentence
- "The **bank** approved the loan" vs. "We walked along the river **bank**"
- FinBERT: BERT fine-tuned on financial text—captures domain nuance

Static: one meaning per word. Contextual: meaning adapts to context.



Five core components:

- **Agent:** The decision maker (e.g., trading algorithm)
- **Environment:** The market or simulator
- **State:** Current portfolio, prices, indicators
- **Action:** Buy, sell, or hold
- **Reward:** P&L after each action

Agent takes actions, receives rewards, learns optimal policy

Q-Learning at a Glance

What is Q(s, a)?

- The expected total discounted reward from taking action a in state s, then acting optimally

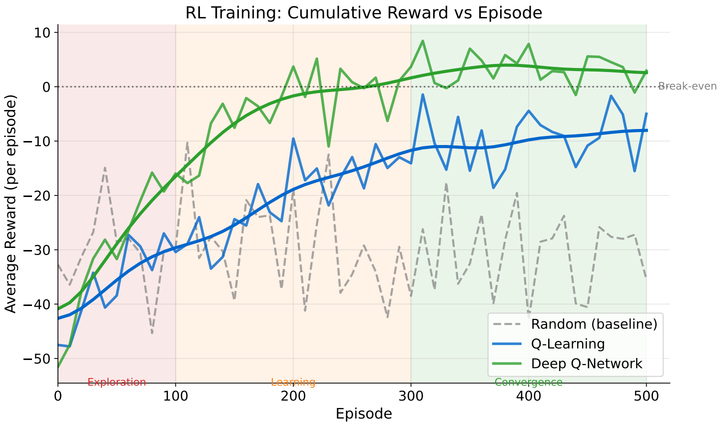
How does the agent learn?

- **Update rule:** Blend old estimate with new evidence (controlled by learning rate α)
- **Target:** Immediate reward r plus discounted best future value $\gamma \max_{a'} Q(s', a')$
- Old and new are mixed: $Q_{\text{new}} = (1 - \alpha)\, Q_{\text{old}} + \alpha\, [\text{target}]$

Exploration vs. Exploitation

- ϵ-greedy: with probability ϵ, choose a random action (explore); otherwise, choose the best known action (exploit)

Q-learning: the foundation of value-based reinforcement learning



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/05_reward_curves

RL agents improve through exploration and exploitation over many episodes

Embeddings in Finance: Sentiment Analysis

Worked Example

- Input headline: "Fed signals aggressive rate hike amid inflation fears"
- Embed headline into a 768-dimensional vector using FinBERT
- Compute cosine similarity to positive/negative anchor phrases
- Classification: **Negative sentiment** (bearish for equities)

Why FinBERT?

- General BERT struggles with financial jargon ("hawkish", "dovish")
- FinBERT achieves 87% accuracy on financial sentiment (Araci, 2019)

Simplified example — real embeddings are 300–768 dimensions

RL in Finance: Trading Agents

Formulation

- **State:** Price history, technical indicators, current position
- **Action:** Buy, sell, hold (possibly with position sizing)
- **Reward:** Risk-adjusted return (e.g., Sharpe ratio per step)

Key Challenges

- **Non-stationarity:** Market regimes shift—policies trained on bull markets fail in crashes
- **Overfitting:** Agent memorizes historical patterns that do not repeat
- **Partial observability:** The agent never sees all relevant information

---

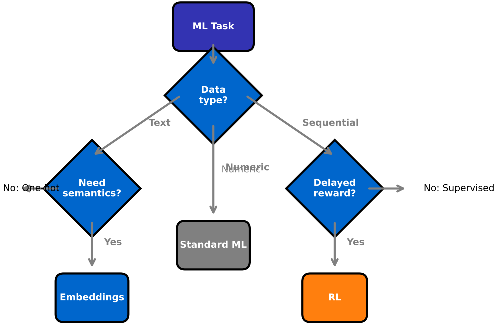
RL for trading is an active research area; not a solved problem

Embeddings vs. Reinforcement Learning

	Embeddings	**Reinforcement Learning**
Input	Text data (words, sentences, docs)	Sequential decisions with delayed feedback
Learns	Semantic representations (dense vectors)	Optimal policy (state $\rightarrow$ action mapping)
Finance Use	Sentiment analysis, document similarity	Trading strategies, portfolio optimization

Both transform complex inputs into learnable representations

When to Use Embeddings vs RL



Embeddings: Text, categorical -> dense vectors (Word2Vec, BERT)

RL: Sequential decisions with delayed rewards (trading, games)

https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L06_Embeddings_RL/07_decision_flowchart

Embeddings for text, RL for sequential decisions with delayed rewards

Practical Tips

Embeddings

- Start with pre-trained models (Word2Vec, GloVe, or FinBERT)—training from scratch requires massive corpora
- Match the domain: financial text needs financial embeddings
- Visualize with t-SNE to sanity-check that clusters make sense

Reinforcement Learning

- Start simple: tabular Q-learning before DQN or policy gradient
- Design rewards carefully—reward shaping prevents sparse-signal problems
- Normalize states: RL is sensitive to feature scales

Both domains: start simple, iterate, validate thoroughly

Hands-on Exercise

Open the Colab Notebook

1. **Exercise 1:** Explore word embeddings with Word2Vec—find similar words and visualize clusters
2. **Exercise 2:** Implement basic Q-learning on a grid-world environment
3. **Exercise 3:** Apply RL to a simple trading environment and analyze the learned policy

Link: `https://colab.research.google.com/github/Digital-AI-Finance/methods-algorithms/blob/master/notebooks/L06_embeddings_rl.ipynb`

Notebooks available on the course GitHub page — see the L06 folder

Key Takeaways

Embeddings

- Words become dense vectors where proximity encodes semantic similarity
- Pre-trained models (Word2Vec, GloVe, FinBERT) give you a strong starting point

Reinforcement Learning

- An agent interacts with an environment, learning from rewards over time
- Q-learning and DQN provide practical algorithms for value-based control

Finance Applications

- Embeddings power sentiment analysis, document search, and entity extraction
- RL enables algorithmic trading, portfolio rebalancing, and optimal execution

Key Insight: Both methods transform raw, complex inputs into structured representations that machines can learn from.

Course complete! Apply these methods in your capstone project

Closing Thought

*"We poured financial news into a pile of linear algebra
and got sentiment scores on the other side.*

*Then we poured the sentiment scores into a reinforcement learner
and got a trading strategy on the other side.*

The strategy said: 'Buy and hold.'"

— Adapted from XKCD #1838 "Machine Learning" by Randall Munroe

References

- Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv:1301.3781.
- Sutton, R.S. & Barto, A.G. (2018). *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press. Free at `http://incompleteideas.net/book/the-book-2nd.html`
- Jurafsky, D. & Martin, J.H. (2024). *Speech and Language Processing*, 3rd ed. `https://web.stanford.edu/~jurafsky/slp3/`
- Araci, D. (2019). *FinBERT: Financial Sentiment Analysis with Pre-Trained Language Models*. arXiv:1908.10063.

Sutton & Barto: the definitive RL textbook (free at incompleteideas.net)