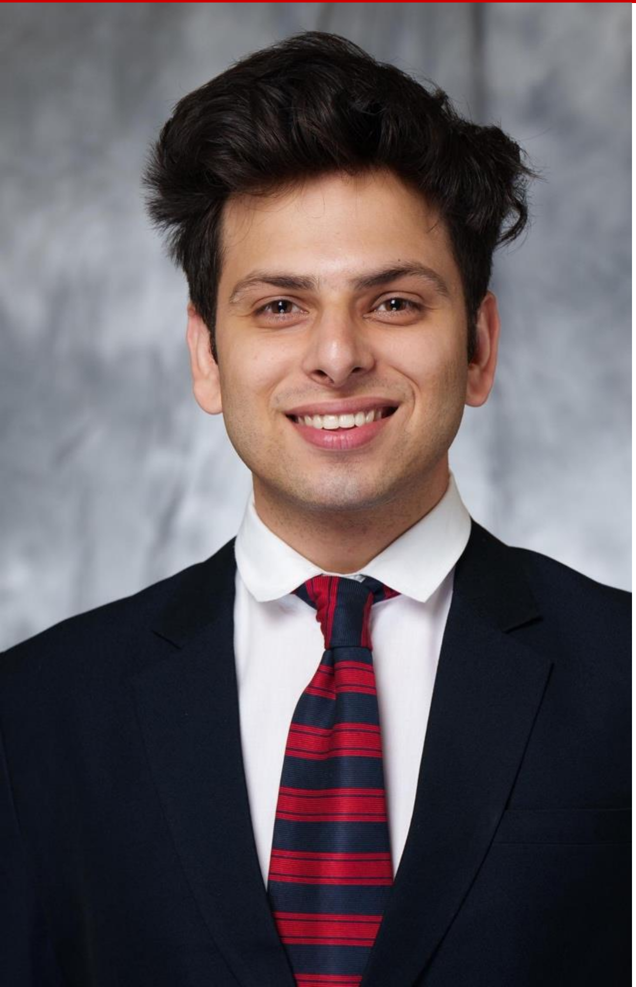
NC STATE UNIVERSITY



AlphaPortfolio: Generating Alpha for Direct Portfolio Optimization Using Deep Reinforcement Learning

By: Dev Kewlani

Date: January 10th, 2025

Contents

Approach

This approach uses multi-sequence attention-based neural networks and economic distillation for explainable AI.

Multi-Sequence Attention-Based Neural Networks:

- **Definition:** These are neural network architectures designed to process time-series data for multiple assets simultaneously.
- **Purpose:** They capture the dependencies and interactions both over time (temporal relationships) and across assets (cross-sectional relationships).
- **Example:** For a stock portfolio, the model might analyze how the past returns of one stock influence others or how macroeconomic variables affect multiple assets in a given period.
- **Benefit:** This structure helps in identifying patterns and relationships that traditional models might overlook, leading to better portfolio optimization.

Economic Distillation for Explainable AI:

- **Definition:** Economic distillation translates complex AI model outputs into simpler, human-understandable forms, such as linear models or feature sensitivity scores.
- **Process:**
  - Identifies which factors (e.g., price-to-book ratio, volatility) most influence the model's decisions.
  - Highlights how these factors interact or evolve over time.
- **Example:** If the model allocates a large weight to a particular stock, economic distillation explains whether this is due to its profitability, momentum, or other features.
- **Importance:** It ensures transparency and builds trust, as investors and stakeholders can see the rationale behind each portfolio decision.

Data Management

- Set up access to CRSP and Compustat databases using Wharton Research Data Services (WRDS)
- Collect monthly stock return data from CRSP (1980-2016)
- Gather firm balance sheet data from Compustat
- Construct 51 firm characteristics and market signals with 12-month lookback
- Clean and normalize the data
- Training period: 1965-1999
- Testing period: 1999-2016

Model Architecture

- Sequence Representation Extraction Module (SREM)
- Cross-Asset Attention Network (CAAN)
- Portfolio Generator

Sequence Representation Extraction Module (SREM)

Purpose:

- The SREM processes the historical time-series data of each asset (e.g., stock prices, financial ratios) to capture trends, patterns, and dependencies over time.

Key Features:

- **Handles Sequential Data:** Uses **Transformer Encoder or LSTM (Long Short-Term Memory)** to extract representations from historical data.
- **Captures Long-Range Dependencies:** Identifies relationships across time, such as how past performance affects future returns.
- **Shared Parameters:** The same SREM is applied to all assets, ensuring consistent feature extraction across the portfolio.

Example:

- For a stock, the SREM analyzes data from the past 12 months (e.g., returns, volatility) and converts it into a compact representation (vector) that summarizes the stock’s state over that period.

Why It’s Important:

- Financial time-series data is noisy and non-linear. The SREM ensures that relevant patterns are captured while filtering out noise.

Cross-Asset Attention Network (CAAN)

Purpose:

- The CAAN identifies and models the interrelationships between assets in the portfolio, such as correlations or interactions between different stocks or sectors.

Key Features:

- **Attention Mechanism:** Assigns weights to the importance of one asset relative to others, based on their historical states.
- **Self-Attention for Relationships:** For example, if stock A and stock B are highly correlated, CAAN uses this relationship to influence portfolio construction.
- **Flexible and Scalable:** Can model complex interdependencies across a large number of assets.

Example:

- Suppose stock A (a tech company) and stock B (a semiconductor supplier) are closely linked. If stock A shows strong momentum, CAAN might increase the weight of stock B due to their relationship.

Why It's Important:

- Asset interactions are crucial in portfolio optimization. Ignoring them can lead to suboptimal decisions, especially in diversified portfolios.

Portfolio Generator

Purpose:

- Converts the outputs of the CAAN (winner scores) into actionable portfolio weights for long and short positions.

Key Features:

- **Winner Scores:** Each asset is assigned a score based on its likelihood of contributing positively to the portfolio's performance.
- **Dynamic Allocation:** Assets with high winner scores are given higher weights in the long portfolio, while those with low scores are shorted.
- **Handles Constraints:** Can incorporate transaction costs, liquidity constraints, and other real-world factors.

Process:

- Assets are ranked by their winner scores.
- Top-ranked assets are assigned to the long portfolio, and bottom-ranked assets to the short portfolio.
- Portfolio weights are normalized to ensure balance and compliance with constraints.

Why It's Important:

- The Portfolio Generator ensures that the final portfolio is not only theoretically optimal but also practical and implementable in real-world trading.

Reinforcement Learning for Portfolio Optimization

Core Idea

- Optimize portfolio weights by maximizing financial performance metrics through trial-and-error learning.

RL Components

- States: Historical market and portfolio data.
- Actions: Portfolio weights for all assets.
- Rewards: Sharpe ratio or risk-adjusted returns for holding period.

Training Process

- Initialize random weights.
- Simulate performance using historical data.
- Update policy network using RL algorithms like PPO.

Benefits

- Dynamic and adaptive to market changes.
- Directly aligns with portfolio goals.
- Transparent and interpretable with modern AI tools.

Implementation Timeline

Phase 1: Data Pipeline (3 weeks)
- WRDS API integration and data collection
- Feature engineering
- Data cleaning

Phase 2: Model Development (4 weeks)
- SREM implementation
- CAAN development
- Portfolio generator

Phase 3: Training (3 weeks)
- RL framework setup
- Parameter optimization
- Performance monitoring

Phase 4: Testing (2 weeks)
- OOS validation
- Robustness checks
- Economic Interpretation

Results Expectations

- **Key Metrics Expectations**
  - Sharpe ratio > 2.0
  - Annual returns ~17%
  - Low turnover (~0.26)
  - Maximum drawdown < 10%
- **Robustness**
  - Consistent across market caps
  - Transaction cost resilient
  - Industry-neutral performance

Validating Economic Interpretability

Analysis Methods

- Feature sensitivity analysis
- Polynomial projections
- Economic distillation

Key Findings

- Important features rotation
- Non-linear effects
- Market condition adaptability

WHY THIS PROJECT?

- TENSOR FLOW and PYTORCH
- Building Attention-based NNs and Transformers
- End To End System Design
- Explainable AI methods
- Project Catered for interviews with IBs, QR/QT and DS

Thankyou! Any Questions?