Graph Machine Learning for Asset Pricing: Traversing the Supply Chain

Agostino Capponi

IEOR and Columbia Business School

Joint work with

Jose Sidaoui Jiacheng Zou

Supply chain data is hard to use

- **Lack of systematic approach** to use supply chain data
  - Noisy, missing, endogenous - linkages are commercial secrets
  - Dynamic - firms remove linkages and add new linkages
  - High-dimensional - 1000 firms may have $\binom{1000}{2}$ linkages at any time

- How to use supply chain data for asset pricing?

⇒ **This paper:** data-driven framework to extract asset pricing factors integrating supply chain and firm-specific information

Methodological Contributions

1. First formulation of graph learning grounded on financial economic principles

   $\Rightarrow$ Directly extracts latent asset pricing factor from supply chain graph

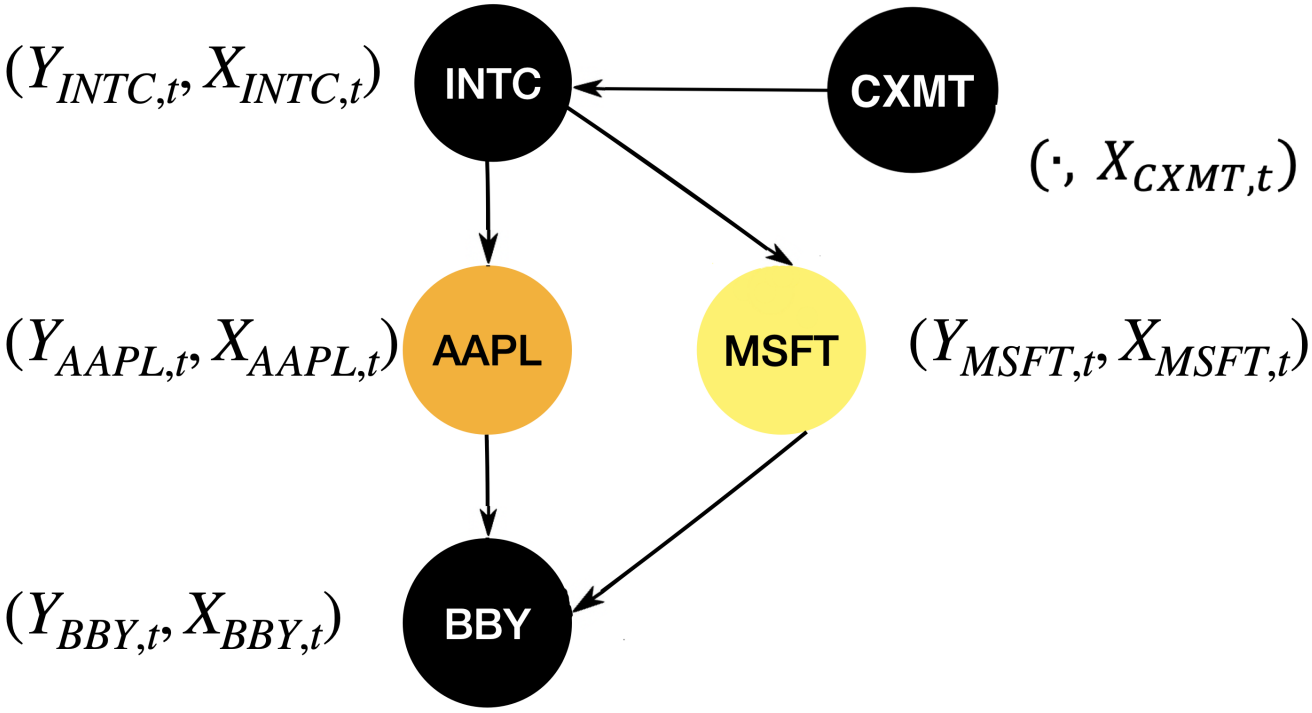
   $\Rightarrow$ Avoids hand-crafting structural signals from graph

2. Tackling noisy, dynamic, high-dimensional supply chain graph data

   $\Rightarrow$ Dynamic SCG provides variations for analyzing effects of the graph structures (e.g. firm centrality) on returns

   $\Rightarrow$ Leveraging *message passing*, well-designed for graphs with rich contexts

Supply chain graph (SCG) representation



$\Rightarrow$ **Objective:** estimate Y_{MSFT} and Y_{AAPL}'s co-movement from the structure of graph, accounting for the rich set of firm-level information

Research Questions

1. Does the supply chain graph improve return predictability beyond characteristic-managed portfolio returns alone?
2. Do supply chain graphs produce new risk factors, uncorrelated with existing benchmarks, unspanned by them, and generating economically large Sharpe ratios?

Supply chain network as a structural source of cross-asset return predictability

Empirical observations

Cross-asset return predictability is large — but its transmission mechanism is not understood. [Kelly, Kuznetsov, Malamud, and Xu, 2025, Kelly, Malamud, and Pedersen, 2023]

Economic mechanism

Input-output shocks propagate along buyer–supplier links, generating forecastable return co-movement. [Cohen and Frazzini, 2008, Ding, Levine, Lin, and Xie, 2021]

- Own-firm models discard graph topology entirely
  $\Rightarrow$ network-transmitted co-movement goes undetected
- Conditioning on direct links only misses multi-tier transmission
  $\Rightarrow$ higher-order supply chain effects remain unpriced
- Observed buyer–supplier edges make the channel empirically testable
  $\Rightarrow$ graph topology is the structural source of cross-asset predictability

Notations and benchmarks

Data:

- Y_{it}: Returns of firms in NYSE, NASDAQ, AMEX [Gu, Kelly, and Xiu, 2020]
- X_{it}: two parts, firm information and macro information

  Part 1: 64 lagged firm characteristic-managed portfolios [Freyberger, Neuhierl, and Weber, 2020]

  Part 2: 28 lagged macro variables used in Federal Reserve Bank's Dodd-Frank stress test (including: US / Europe / UK / Japan / Asia GDP growth rates and inflation rates, VIX, US unemployment growth rate, etc.)

  $\Rightarrow$ End-to-end `Python` code for (Y_{it}, X_{it}), **available on GitHub**
- SCG_t: FactSet Revere relationships data ⏩ Factset Details

Frequency:

- From 2003-01 to 2025-06. (Y_{it}, X_{it}) – monthly
- SCG_t – (regularly updated by Factset)

Notations and benchmarks (cont'd)

Benchmarks with factor model but without SCG:

- Fama-French 5 factor (FF5) [Fama and French, 2015]
- Principal Components factor model (PCA)
- Multi-Layer Perceptron Neural Network (NN) [Lettau and Pelger, 2020]

Benchmarks with supply chain information without graph:

- Benchmarks "nearest neighbors" only consider characteristics of direct suppliers/customers [Agrawal and Osadchiy, 2024]
- By contrast, we **adaptively consider higher-order** relationships

Benchmarks with higher order network effects in SCG:

- Modified regression with Leontief inverse based on adjancency matrices of the SCG monthly observations [Leontief, 1986, Acemoglu et al., 2012]
- We consider a nonlinear architecture that learns node-specific and feature-specific weights, enabling it to capture multi-hop relationships

Graph Neural Networks (GNNs) for Supply Chain Data

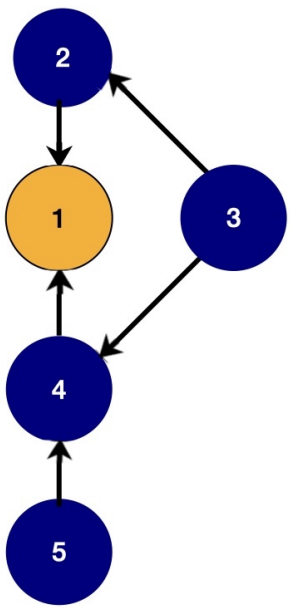
- **Goal**: Capture interactions in the supply chain using non-linear functions.
- **How it works:**
  - Implements *message passing* to propagate information through the supply chain graph.
  - Update each node firm i by aggregating information from its **upstream** neighbors $j \in \mathcal{N}_t(i)$ and combining it with its own features. $\Rightarrow$ Firm i passes a message to all customers j; at the same time, receives messages from all suppliers j'.
  - Estimate firm i's outcome variable Y_{it} based on network-integrated features.
- **Why GNNs?**
  - Specifically designed for **graph-based learning**, unlike traditional ML.

Intuitions on the "message passing" algorithm

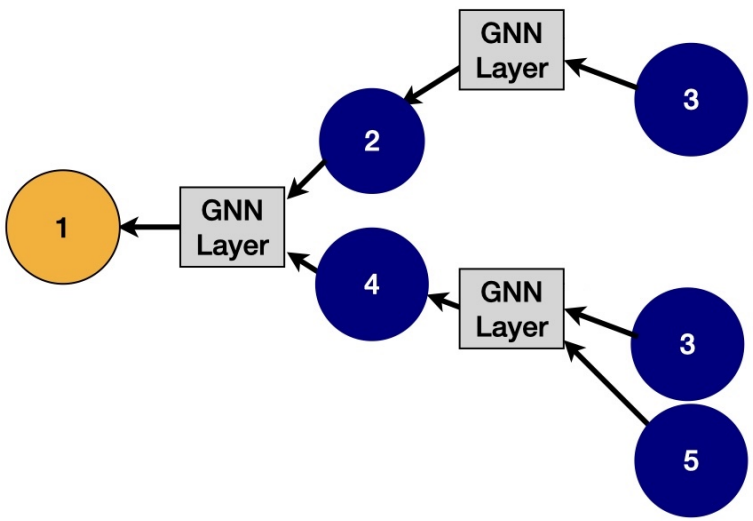
- GNN takes **"weighted non-linear averages"** of X_{it} in node i's local graph
- Number of layers is the size of the local sub-graph

  $\Rightarrow$ E.g. 2nd layer gets $X_{i't}$ of i'th nodes that are 2-hops away

  $\Rightarrow$ Size of local sub-graph d $\Leftrightarrow$ **number of layers**
- Back-propagation updates the weights of these "weighted averages" so that the last layer's output fits the returns
- Embedding is the output of message passing



(a) Example graph



(b) $d = 2$ message passing

GNN computation with message passing

In each GNN layer:

- **Message Calculation:**

$$m_{ij}^{(l)} = f_{\text{msg}}(h_i^{(l-1)}, h_j^{(l-1)}, \mathbf{x}_{it}, \mathbf{x}_{jt})$$

  Computes messages $m_{ij}^{(l)}$ from neighbor j to node i with function f_{msg}.

- **Aggregation:**

$$m_i^{(l)} = AGG(\{m_{ij}^{(l)} : j \in \mathcal{N}_t(i)\})$$

  Summarizes messages from all neighbors using a permutation-invariant function (e.g., sum, mean, max) as AGG.

- **Update:**

$$h_i^{(l)} = f_{\text{upd}}(h_i^{(l-1)}, m_i^{(l)})$$

  Combines aggregated messages with the node's current features to produce updated features with function f_{upd}.

- **Output layer:** The output of message passing (i.e. the last hidden layer's neurons) $h_i^{(d)}$ is the **embedding**

GNN computation with message passing

In each GNN layer:

- **Message Calculation:**

$$m_{ij}^{(l)} = f_{\text{msg}}(h_i^{(l-1)}, h_j^{(l-1)}, \mathbf{x}_{it}, \mathbf{x}_{jt})$$

  Computes messages $m_{ij}^{(l)}$ from neighbor j to node i with function f_{msg}.

- **Aggregation:**

$$m_i^{(l)} = AGG(\{m_{ij}^{(l)} : j \in \mathcal{N}_t(i)\})$$

  Summarizes messages from all neighbors using a permutation-invariant function (e.g., sum, mean, max) as AGG.

- **Update:**

$$h_i^{(l)} = f_{\text{upd}}(h_i^{(l-1)}, m_i^{(l)})$$

  Combines aggregated messages with the node's current features to produce updated features with function f_{upd}.

- **Output layer:** The output of message passing (i.e. the last hidden layer's neurons) $h_i^{(d)}$ is the **embedding**

GNN computation with message passing

In each GNN layer:

- **Message Calculation:**

$$m_{ij}^{(l)} = f_{\text{msg}}(h_i^{(l-1)}, h_j^{(l-1)}, \mathbf{x}_{it}, \mathbf{x}_{jt})$$

  Computes messages $m_{ij}^{(l)}$ from neighbor j to node i with function f_{msg}.

- **Aggregation:**

$$m_i^{(l)} = AGG(\{m_{ij}^{(l)} : j \in \mathcal{N}_t(i)\})$$

  Summarizes messages from all neighbors using a permutation-invariant function (e.g., sum, mean, max) as AGG.

- **Update:**

$$h_i^{(l)} = f_{\text{upd}}(h_i^{(l-1)}, m_i^{(l)})$$

  Combines aggregated messages with the node's current features to produce updated features with function f_{upd}.

- **Output layer:** The output of message passing (i.e. the last hidden layer's neurons) $h_i^{(d)}$ is the **embedding**

GNN computation with message passing

In each GNN layer:

- **Message Calculation:**

$$m_{ij}^{(l)} = f_{\text{msg}}(h_i^{(l-1)}, h_j^{(l-1)}, \mathbf{x}_{it}, \mathbf{x}_{jt})$$

  Computes messages $m_{ij}^{(l)}$ from neighbor j to node i with function f_{msg}.

- **Aggregation:**

$$m_i^{(l)} = AGG(\{m_{ij}^{(l)} : j \in \mathcal{N}_t(i)\})$$

  Summarizes messages from all neighbors using a permutation-invariant function (e.g., sum, mean, max) as AGG.

- **Update:**

$$h_i^{(l)} = f_{\text{upd}}(h_i^{(l-1)}, m_i^{(l)})$$

  Combines aggregated messages with the node's current features to produce updated features with function f_{upd}.

- **Output layer:** The output of message passing (i.e. the last hidden layer's neurons) $h_i^{(d)}$ is the **embedding**

Message passing: linear illustration

One-layer case ($d = 1$):

$$h_{it}^{(1)} = W_1 h_{it}^{(0)} + W_2 \left(\frac{1}{|\mathcal{N}_t(i)|} \sum_{j \in \mathcal{N}_t(i)} h_{jt}^{(0)} \right)$$

$\Rightarrow$ expected return depends on firm i's own characteristics *and* the average characteristics of its direct suppliers

$\Rightarrow$ first-order cross-firm dependence through direct supply chain exposure

Two-layer case ($d = 2$) adds a second-order term:

$$\frac{1}{|\mathcal{N}_t(i)|} \sum_{j \in \mathcal{N}_t(i)} \frac{1}{|\mathcal{N}_t(j)|} \sum_{k \in \mathcal{N}_t(j)} h_{kt}^{(0)}$$

$\Rightarrow$ supplier shocks that first affect firm j can then affect firm i through j's own suppliers

$\Rightarrow$ indirect bottlenecks and financing stress are priced even without a direct contractual link

Our model: f_{msg} and f_{upd} are nonlinear

d and supply chain risks

Small d:

- Captures only immediate supply chain dependencies
- Omits higher-order transmission — the focus of previous literature

Large d:

- Captures higher-order disruptions propagating through the SCG
- But triggers *oversmoothing* [Chen et al., 2020]: repeated aggregation homogenizes firm embeddings, reducing the cross-sectional heterogeneity needed for return prediction
  $\Rightarrow$ this is a *structural* constraint imposed by the graph topology, *not* a standard bias-variance tradeoff

Optimal d:

- Balances higher-order signal capture against oversmoothing

GNN computation: design

- **Design question:** What are good model implementation choices of GNN layers?

[PDF] Attention is all you need

A Vaswani - Advances in Neural Information Processing Systems, 2017 - user.phil.hhu.de

The dominant sequence transduction models are based on complex recurrent orconvolutional neural networks in an encoder and decoder configuration. The best …

☆ Save 99 Cite Cited by 147292 Related articles »

- **Transformers** are behind most modern neural network applications: ChatGPT, Llama, Gemini, Claude, etc.

GNN estimation: direct asset pricing

- End-to-end GNN formulation that takes both firm-level information $\boldsymbol{X}_t$ and supply chain graph SCG_t, to directly estimate Y_{it}
- Our novelty: estimate GNN by optimizing

$$\inf_{\Theta} \mathbb{E}_{(i,t)\sim\mathcal{D}_{\text{train}}} \left[\left(Y_{it} - \underbrace{g_i(\Theta; \mathbf{X}_t, SCG_t)}_{\hat{Y}_{it}} \right)^2 \right] + \ell(\Theta),$$

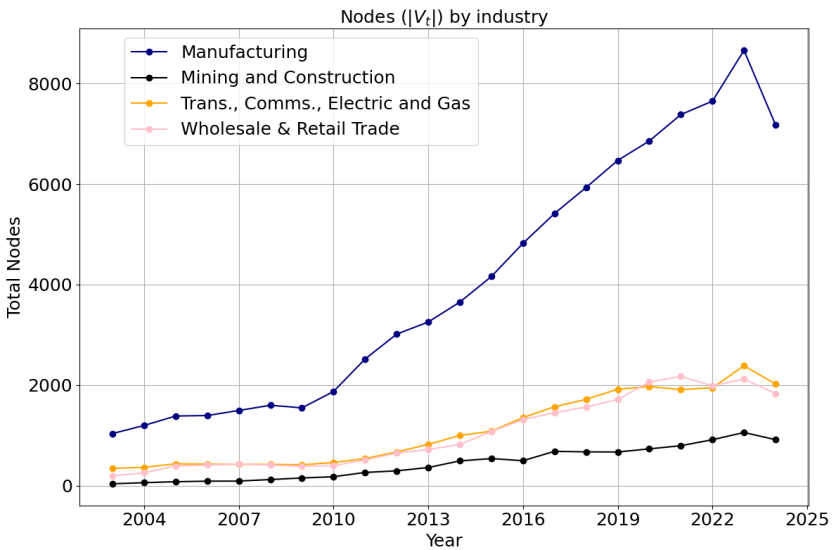
  where g_i represents the GNN model that conditions on SCG and characteristic-managed portfolio returns
- Θ is **not retrained** for each firm i
  $\Rightarrow$ Would be prohibitively computationally expensive
- Instead, we juxtaposed the graph differently for each firm by treating itself as the target firm for message passing
  $\Rightarrow$ GNN's $\hat{Y}_{it}$ **depend on firm i's location in the graph** SCG_t

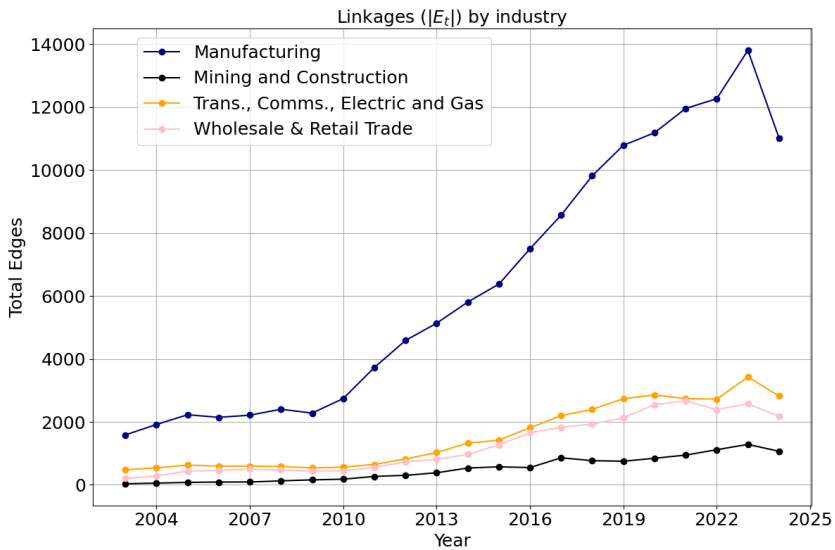
Descriptive Statistics of the SCG_t data

- Both the number of nodes and edges increase

  $\Rightarrow$ We split the train vs test data in "edge space": 2003-2016; 2017-2025

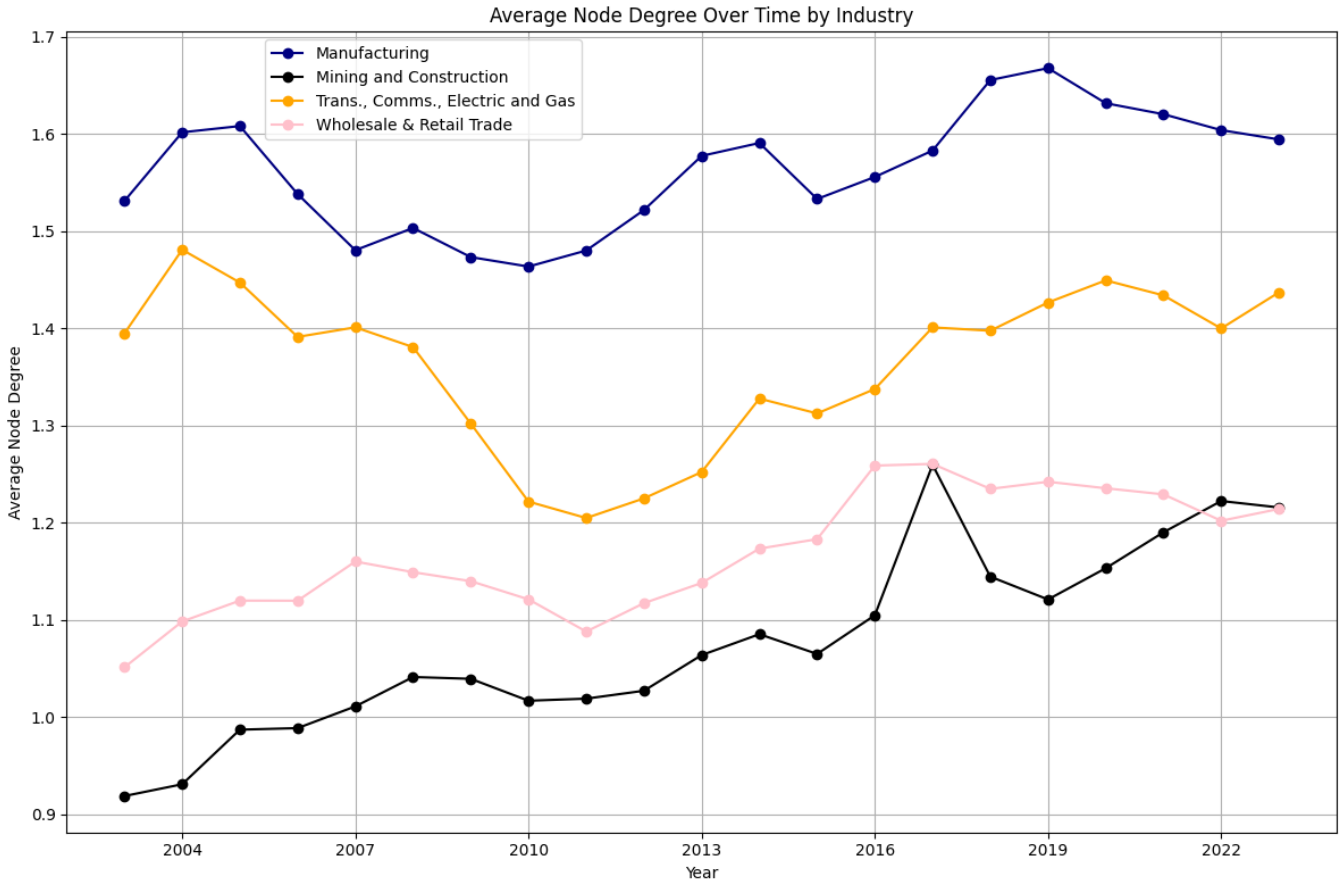
  This break-point yields train and test set that roughly have the same number of supply chain relationships





Descriptive Statistics of the SCG_t data (cont'd)

- SCG is sparse: avg degree of 0.9∼1.7 across years
- Manufacturing (orange) and Sales (purple) are the better connected industries



Benchmark: nearest neighbor firms' characteristics

- Question: what if we use supply chain data but not GNN?

  $\Rightarrow$ Measure the gain of GNN adaptively learning higher-order relationships

- We consider linear model using nearest neighbor firms' level information

$$\bar{\mathbf{x}}_{it} = \frac{1}{|\mathcal{N}_t(i)|} \sum_{j \in \mathcal{N}_t(i)} \mathbf{x}_{jt}.$$

  - $\bar{\mathbf{x}}_{it}$ is 92-dimensional, same as $\mathbf{x}_{it}$

- Benchmarks: LASSO and Ridge neighboring $\bar{\mathbf{x}}_{it}$ and firm's own $\mathbf{x}_{it}$

$$\hat{\theta}^{\text{LASSO}} = \text{argmin}_\theta \mathbb{E}_{(i,t)\sim\mathcal{D}_{\text{train}}} \left[\left(Y_{it} - [\mathbf{x}_{it} \quad \bar{\mathbf{x}}_{it}]^\top \theta \right)^2 \right] + \lambda_1 \|\theta\|_1,$$

$$\hat{\theta}^{\text{Ridge}} = \text{argmin}_\theta \mathbb{E}_{(i,t)\sim\mathcal{D}_{\text{train}}} \left[\left(Y_{it} - [\mathbf{x}_{it} \quad \bar{\mathbf{x}}_{it}]^\top \theta \right)^2 \right] + \lambda_2 \|\theta\|_2^2.$$

  $\Rightarrow$ These benchmarks uses supply chain, **but not higher-order supply chain relationships**

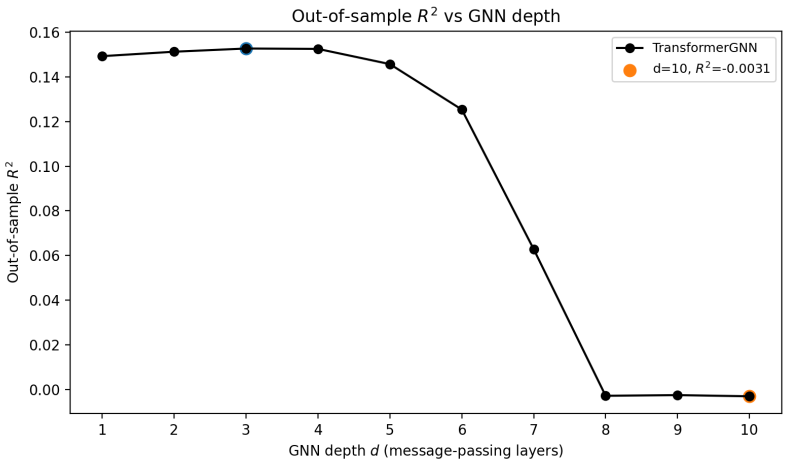
Research Questions

1. Does the supply chain graph improve return predictability beyond characteristic-managed portfolio returns alone?
2. Do supply chain graphs produce new risk factors, uncorrelated with existing benchmarks, unspanned by them, and generating economically large Sharpe ratios?

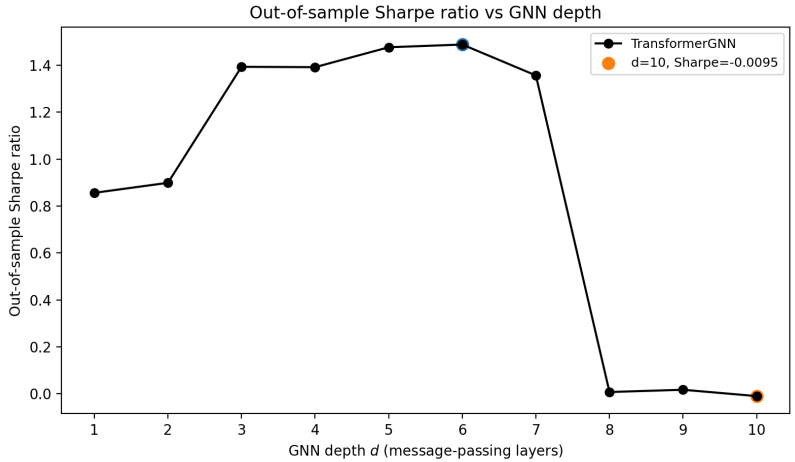
Empirical asset pricing results: conditional mean return

Model	Out-of-sample R^2		
	Value	Improvement vs FF5	Improvement vs NN3
Graph Neural Network Models			
TransformerGNN ($d = 1$)	0.1493	+131.6%	+17.3%
TransformerGNN ($d = 2$)	0.1512	+132.0%	+18.8%
TransformerGNN ($d = 3$)	**0.1527**	**+132.3%**	**+20.0%**
TransformerGNN ($d = 4$)	0.1525	+132.3%	+19.8%
TransformerGNN ($d = 6$)	0.1253	+126.5%	-1.6%
TransformerGNN ($d = 10$)	-0.0030	+99.4%	-102.4%
Neural Network Benchmarks			
NN2	0.1131	+124.0%	-11.2%
NN3	0.1273	+127.0%	Baseline
Linear / Neighborhood Benchmarks			
FF5	-0.4722	Baseline	-470.9%
PCA ($k = 5$)	-0.4659	+1.3%	-466.0%
NC Ridge	0.0663	+114.0%	-47.9%
NC LASSO	0.0660	+114.0%	-48.1%

Graph depth: performance peaks, then oversmooths



(a) OOS R^2 by depth



(b) OOS Sharpe ratio by depth

- Predictive R^2 rises from $d = 1$ to $d = 3$ and then declines.
- Sharpe ratio follows the same pattern: the trading value peaks and then declines for $d \geq 8$.
- Interpretation: $d = 3$ captures higher-order supply-chain transmission before distant/redundant neighborhoods dilute firm heterogeneity.

Do higher-order network effects matter?

Shock propagation via Leontief inverse:

$$L_t(\rho) = (I - \rho A_t)^{-1} = \sum_{k=0}^{\infty} \rho^k A_t^k$$

- A_t^k captures k-hop indirect supply chain links
  $\Rightarrow$ a shock at firm j attenuates at rate ρ^k across k hops
- **Main limitation**: decay rate ρ and loading $\mathbf{b}$ are homogeneous across all firms and distances
  $\Rightarrow$ GNN learns node-specific and distance-specific weights instead

Empirical asset pricing results: Leontief-inverse

Using firm characteristic (firm char.)-managed portfolio returns $\mathbf{X}_t$ we estimate

$$\mathbf{y}_t = \sum_{k=0}^{\infty} A_t^k \rho^k \mathbf{X}_t \mathbf{b} + \epsilon_t,$$

We implement the estimation via Leontief inverse $L_t(\rho) = (I - \rho A_t)^{-1}$:

$$\min_{\rho, b} \sum_{t \in T_{Train}} \|\mathbf{y}_t - L_t(\rho)\mathbf{X}_t b\|_2^2 + \lambda \|b\|_1 \quad s.t. \quad 0 \leq \min(1, \lambda_{\max}(A_t))$$

	Out-of-sample R^2	
Model	Value	Improvement vs Leontief
Graph Neural Network Models		
TransformerGNN ($d = 1$)	0.1493	+71.0%
TransformerGNN ($d = 2$)	0.1512	+73.2%
TransformerGNN ($d = 3$)	**0.1527**	**+74.9%**
Leontief Network Regression		
Full Leontief Regression	0.0873	Baseline
Partial Leontief Network Regression		
First-Order	0.0165	-81.1%
Up to Second-Order	0.0229	-73.8%
Up to Third-Order	0.0284	-67.5%
Up to Fourth-Order	0.0336	-61.5%
Up to Fifth-Order	0.0382	-56.2%

Incremental predictive content: GNN signal beyond the benchmark menu

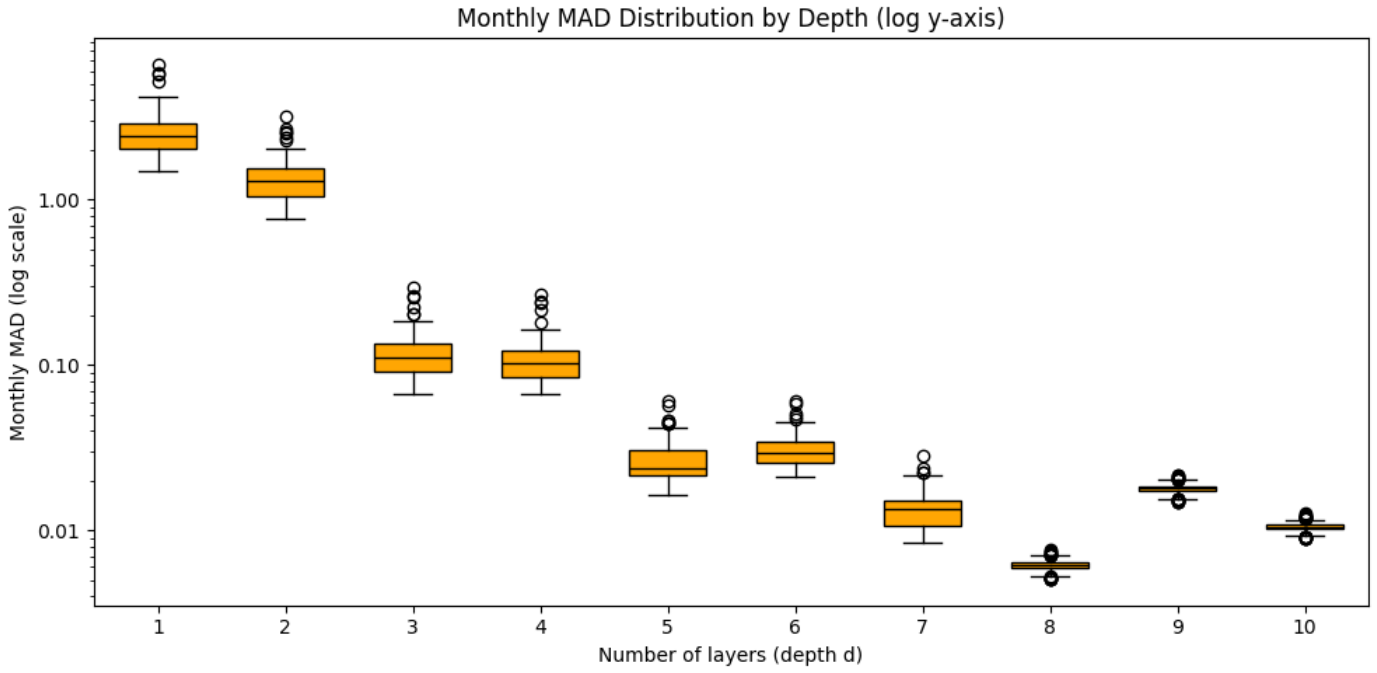
$$\hat{Y}_{it}^{\mathrm{GNN}(d)} = a_d + \boldsymbol{\beta}_d^{\top} \mathbf{z}_{it} + \varepsilon_{it}^{(d)},$$

$\mathbf{z}_{it}$ includes NN2, NN3, FF5, PCA, NC Ridge, and NC LASSO predicted returns.

Specification	$\hat{a}_d$	$t(\hat{a}_d)$	R^2
GNN ($d = 1$)	0.0218	44.6751	0.8782
GNN ($d = 2$)	0.0197	49.6512	0.8661
GNN ($d = 3$)	**0.0235**	**72.4248**	**0.8988**
GNN ($d = 4$)	0.0268	86.0512	0.9018
GNN ($d = 6$)	0.0252	125.2749	0.8855
GNN ($d = 10$)	0.0013	348.6447	0.0007

- The positive intercept indicates incremental graph-conditioned predictive content beyond the benchmark menu.

Oversmoothing mechanism: embedding dispersion collapses



- Mean Average Distance (MAD) measures cross-sectional dispersion of final-layer firm embeddings.
- MAD falls from 2.61 at $d = 1$ to below 0.02 for $d \geq 8$.
- If all embeddings converge to the same vector, all firms receive the same predicted return, cross-sectional dispersion in $\hat{Y}_{it}$ disappears, and the long-short portfolio earns zero return.

Research Questions

1. Does the supply chain graph improve return predictability beyond characteristic-managed portfolio returns alone?
2. Do supply chain graphs produce new risk factors, and generating economically large Sharpe ratios?

Does the supply chain graph produce a new priced risk factor?

We answer this question in 2 steps:

→ **Does sorting on GNN predictions generate return separation?**

Step 1. **Construct a tradable long-short factor from GNN predictions**

→ *Is the factor unspanned by existing benchmarks?*

Step 2. Alpha-test regressions, correlation with FF3, and time-series regressions on FF25, FF30, JKP13

Long-short portfolio managed by GNN: construction

Step 1: Estimate conditional mean returns

- For each firm i at month t, the trained GNN produces a predicted excess return:

$$\hat{Y}_{it} = a^\top \mathbf{e}_{it}^{(d)} + b$$

  where $\mathbf{e}_{it}^{(d)}$ is the final-layer embedding

Step 2: Sort firms into deciles

- At the end of month t, rank all firms by $\hat{Y}_{it}$ and assign to ten equal groups
  $\Rightarrow$ Decile 10 contains firms with highest predicted returns, Decile 1 the lowest

Step 3: Form the long-short factor

- Go long the top decile and short the bottom decile, equal-weighted within each decile:

$$F_t = R_{10,t} - R_{1,t}$$

- Portfolio is rebalanced monthly; GNN is estimated once on training data and held fixed out-of-sample

Long-short portfolio managed by GNN: performance

Model Name	Out-of-sample	
	ER	SR
GNN ($d = 1$)	0.0648	0.8562
GNN ($d = 2$)	0.0682	0.8988
GNN ($d = 3$)	**0.1106**	**1.3923**
GNN ($d = 4$)	0.1078	1.3914
GNN ($d = 10$)	-0.0003	-0.0095
NC Ridge	0.1156	1.1513
NC LASSO	0.1148	1.1946

- $d = 3$ GNN commands the **highest Sharpe ratio** across all benchmarks; expected returns are broadly superior, with NC Ridge the sole exception at a narrow margin (0.1156 vs. 0.1106)
  $\Rightarrow$ Higher-order supply chain structure is the marginal driver of risk-adjusted performance
- GNN too deep ($d = 10$) triggers overfitting

Residualized GNN factor: priced graph-specific component

Residualized portfolio returns	Mean	Volatility	Sharpe Ratio
$\hat{Y}_t^{\mathrm{GNN}(1),\perp}$	0.0525	0.0037	0.8660
$\hat{Y}_t^{\mathrm{GNN}(2),\perp}$	0.0546	0.0034	0.9354
$\hat{\mathbf{Y}}_{\mathbf{t}}^{\mathrm{GNN}(\mathbf{3}),\perp}$	**0.0633**	**0.0035**	**1.0682**
$\hat{Y}_t^{\mathrm{GNN}(4),\perp}$	0.0568	0.0039	0.9115
$\hat{Y}_t^{\mathrm{GNN}(6),\perp}$	0.0431	0.0045	0.6422
$\hat{Y}_t^{\mathrm{GNN}(10),\perp}$	0.0024	0.0010	0.0766

- After projecting out NN2, NN3, FF5, PCA, NC Ridge, and NC LASSO, the $d = 3$ graph-specific component still earns Sharpe ratio 1.07.
- The pattern again peaks at $d = 3$ and collapses at $d = 10$, consistent with oversmoothing.

Does the supply chain graph produce a new priced risk factor?

We answer this question in 2 steps:

$\rightarrow$ *Does sorting on GNN predictions generate return separation?*

Step 1. Construct a tradable long-short factor from GNN predictions

$\rightarrow$ *Is the factor unspanned by existing benchmarks?*

Step 2. Alpha-test regressions, correlation with FF3, and time-series regressions on FF25, FF30, JKP13

Does the supply chain graph produce a new priced risk factor?

We answer this question in 2 steps:

→ *Does sorting on GNN predictions generate return separation?*

Step 1. Construct a tradable long-short factor from GNN predictions ✓

→ **Is the factor unspanned by existing benchmarks?**

Step 2. **Alpha-test regressions, correlation with FF3, and time-series regressions on FF25, FF30, JKP13**

Test-asset regressions: setup and interpretation

For each test asset p, estimate

$$r_{pt} = \alpha_p + \beta_p F_t + \eta_p^\top \mathbf{z}_t + \varepsilon_{pt},$$

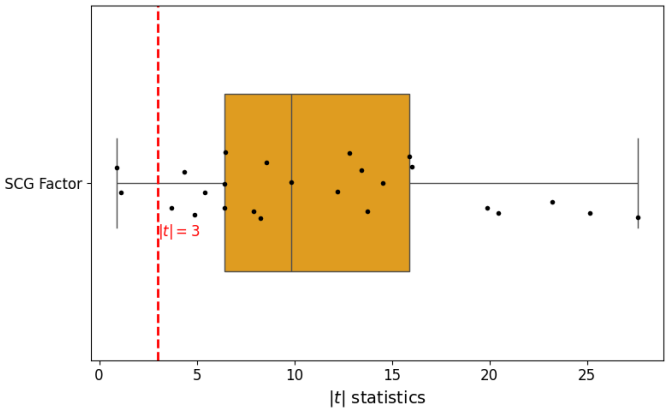
where F_t is the $d = 3$ GNN supply-chain factor and $\mathbf{z}_t$ contains Fama-French controls.

- **FF25:** size-value portfolios.
- **FF30:** industry portfolios.
- **JKP13:** anomaly theme factors.

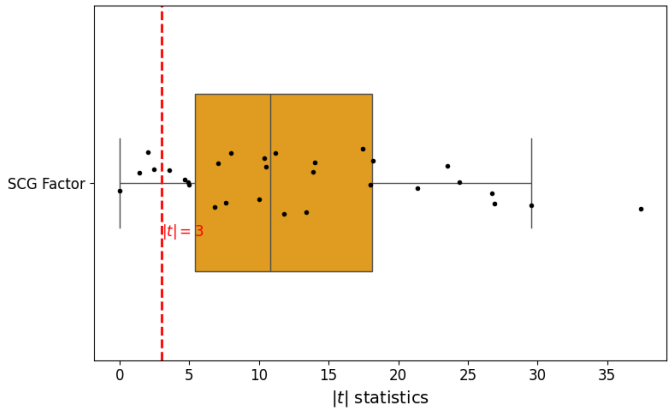
Evidence is strongest when test assets are naturally aligned with production-network and industry risk.

Time-series regressions on test assets

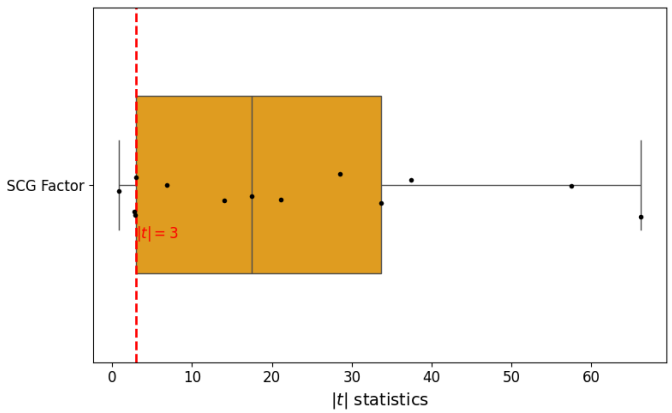
$$r_{pt} = \alpha_p + \beta_p F_t + \eta_p^\top \mathbf{z}_t + \varepsilon_{pt}.$$



(a) FF25



(b) FF30



(c) JKP13

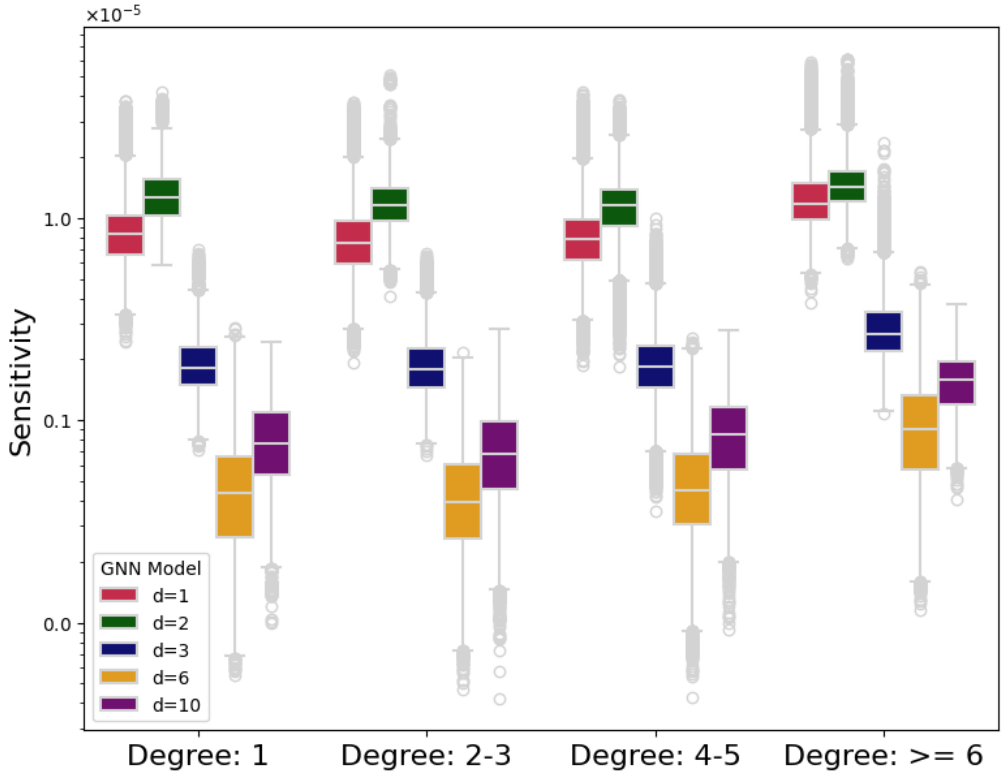
Interpretability

- Does the GNN price firms based on their *position* in the supply chain network, or is it simply proxying for firm size and other characteristics?
  $\Rightarrow$ if the GNN learns from graph topology, sensitivity to missing links should concentrate in *high-degree hubs*, not in large-cap firms
- **Challenge:** neural network embeddings are not directly interpretable
  $\Rightarrow$ we design a **novel Monte Carlo edge-deletion experiment:** for firm i at time t,

$$\tilde{S}_{it}^{(b)} = \left| \hat{g}_i(\mathbf{X}_t, SCG_t^{(b)}) - \hat{g}_i(\mathbf{X}_t, SCG_t) \right|,$$

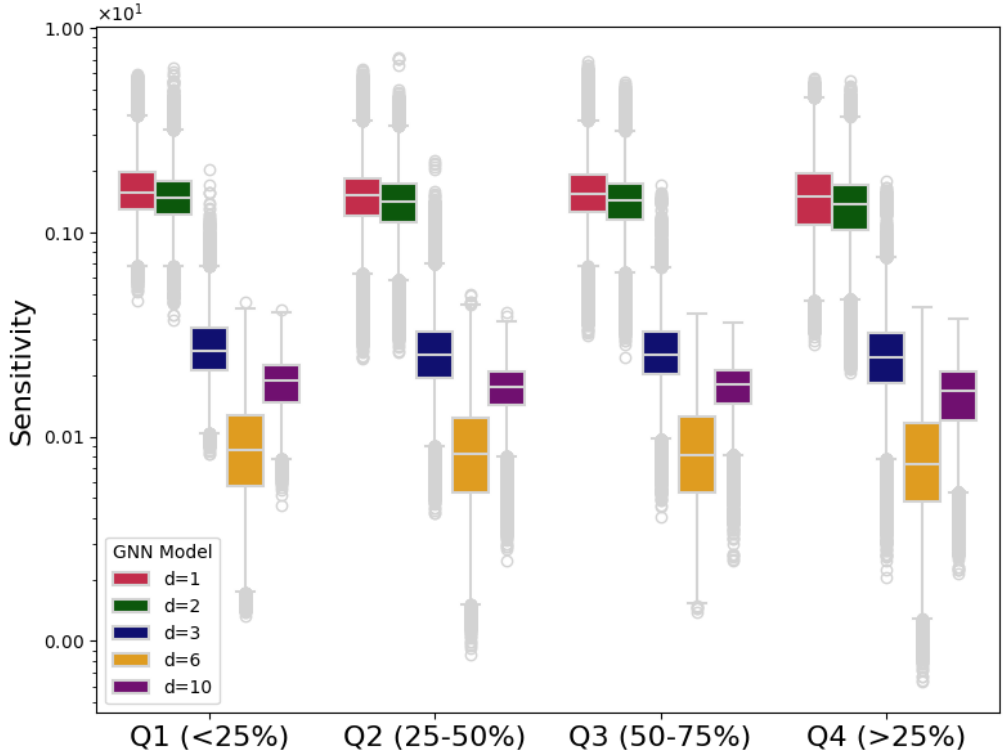
  as the absolute difference between the original return estimate and the perturbed estimate $SCG_t^{(b)}$ after an edge deletion
- We average $\tilde{S}_{it}^{(b)}$ over multiple Monte-Carlo runs ($b = 1, 2, ..., 100$), and obtain estimate of the sensitivity for each group
- We introduce randomness to marginalize over variations of specific edge or node choices within each group

More central firms have higher sensitivity $\tilde{S}$



- **Firm centrality**: as the degree of perturbed nodes increases, the GNN estimates show higher average sensitivity $\tilde{S}$ and increased variance.
- With 6-layer GNN (yellow boxes), the average sensitivity of firms with ≥ 6 degrees is 41% higher than that of firms with 1 degree.
- **Economic mechanism**: deleting an edge of a high-degree hub disrupts the message-passing path of every firm within d hops

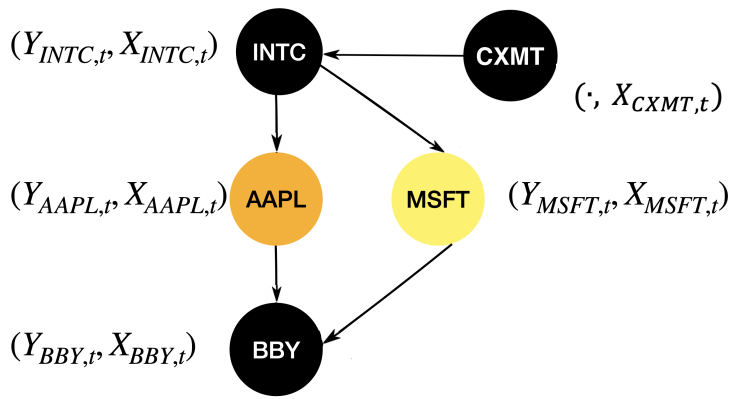
Network, not firm size, drives results

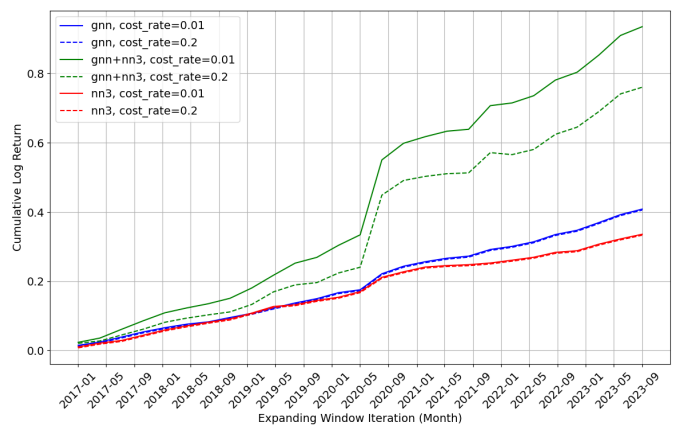


- For $d = 3$, mean sensitivity varies by only 7% across the four market-cap quartiles
- Market cap and graph degree need not move together: a mid-size manufacturer with many counterparties occupies a high-degree position, while a large conglomerate with few direct links does not
- Central firms are priced more heavily through graph topology, *not* firm size

Conditional mean return estimation and identification of a new priced risk factor from supply chain data – Capponi, Sidaoui, and Zou. "Graph Machine Learning for Asset Pricing." (2024).

- New multi-modal ML handling **noisy, dynamic, high-d** supply chain data
- Number of layers d interpretable $\Rightarrow$ **Size of local subgraph**
- **Improvements** over NN, FF, Ridge, LASSO, PCA, Leontief-inverse in estimation of stock returns
- Empirically GNN + characteristic portfolios outperforms characteristic portfolios alone





Appendix

Individual test-asset loadings: headline counts

Test asset set	Significant loadings	Interpretation
FF25 size-value	23/25	broad cross-sectional pricing relevance
FF30 industries	26/30	strongest for supply-chain-intensive industries
JKP13 themes	9/13	strongest for investment/debt/seasonality themes

- The FF30 result is especially consistent with the network channel because industries aggregate firms connected by input-output relationships.
- JKP13 results aggregate well-known cross-sectional anomalies into representative long-minus-short portfolios. The largest exposures in "Investment" ($|t| = 66.2$), "Debt Issuance" ($|t| = 57.5$), and "Seasonality" ($|t| = 37.4$).

HJD: the joint pricing-error metric

We approximate a linear SDF

$$m_t(\mathbf{b}) = 1 - \mathbf{b}^\top \mathbf{z}_t,$$

and evaluates out-of-sample pricing errors via

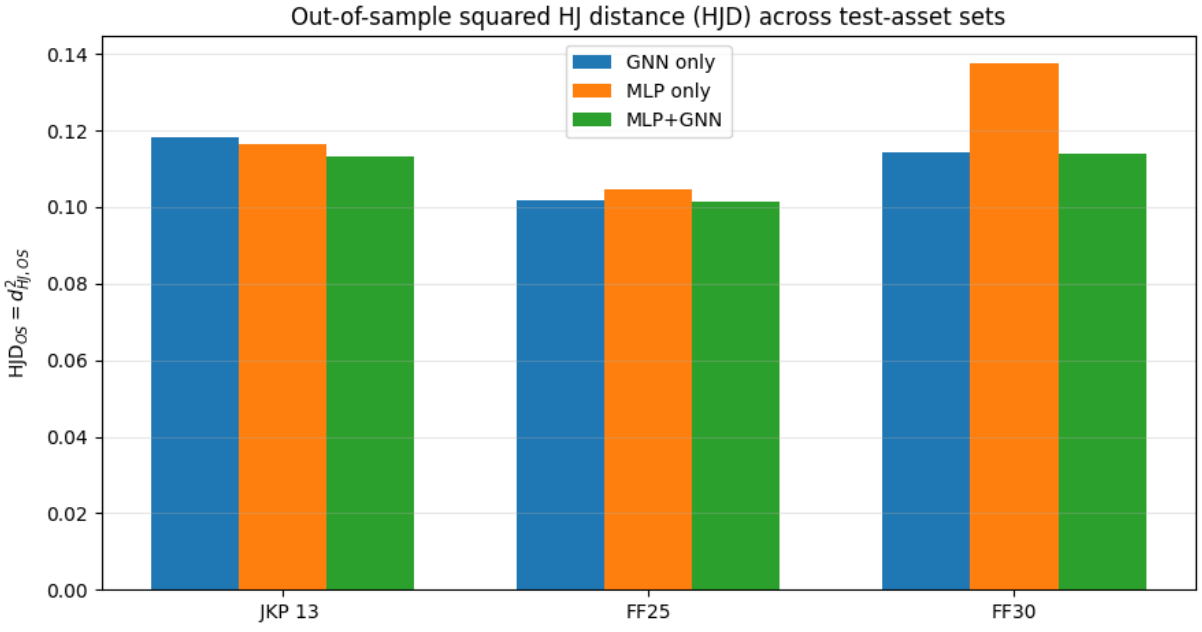
$$\widehat{\mathrm{HJD}}_{OS} = \hat{\mathbf{e}}_{OS}^\top \hat{\mathbf{B}}_{OS}^\dagger \hat{\mathbf{e}}_{OS},$$

where

$$\hat{\mathbf{e}}_{OS} = T_{OS}^{-1} \sum_{t \in OS} \hat{m}_t \mathbf{r}_t, \qquad \hat{\mathbf{B}}_{OS} = T_{OS}^{-1} \sum_{t \in OS} \mathbf{r}_t \mathbf{r}_t^\top.$$

- Lower HJD means smaller joint pricing errors across all test portfolios.
- This complements individual t-statistics because it accounts for the covariance structure of test assets.

Hansen-Jagannathan distance: joint pricing-error test



Test assets	NN3 HJD	GNN HJD	GNN+NN3 HJD
JKP13 anomaly themes	0.1163	0.1182	0.1131
FF25 size-value	0.1046	0.1017	0.1013
FF30 industries	0.1377	0.1142	0.1138

- Lower HJD means smaller joint pricing errors. Largest reduction is for FF30 industries: about 17% versus NN3.

HJD tables by test-asset set

JKP13

Model	HJD	Red.
NN3	.1163	–
GNN	.1182	-1.63%
GNN+NN3	.1131	2.75%

FF25

Model	HJD	Red.
NN3	.1046	–
GNN	.1017	2.77%
GNN+NN3	.1013	3.15%

FF30

Model	HJD	Red.
NN3	.1377	–
GNN	.1142	17.07%
GNN+NN3	.1138	17.36%

- The combined model improves HJD for all three test-asset sets.
- The largest gain is for FF30 industry portfolios, where supply-chain structure and industry exposure are naturally linked.

Correlation between GNN-managed factor and FF3

	SCG	MKT	SMB	HML
SCG	1			
Market (MKT)	**0.179**	1		
Size (SMB)	**0.187**	0.392	1	
Book-to-Market (HML)	**0.103**	0.157	0.284	1

- **First column** reports the correlation of FF vs the SCG asset (GNN-managed L-S portfolio, expanding window trained)
- SCG correlation with FF3 factors are on average 0.16

  $\Rightarrow$ Lower than the internal correlations among FF3

  $\Rightarrow$ But still correlated with FF3: FF3 themselves are learned by our GNN model via $\mathbf{x}_{it}$

From One-Month Prediction to Multi-Step Forecasts

Does the graph-based signal remains economically useful beyond the one-month horizon?

We therefore examine whether multi-step GNN forecasts:

- remain informative at longer horizons;
- reduce turnover through less frequent rebalancing;
- improve net performance after transaction costs.

Tradeoff: A signal with slightly lower statistical accuracy may still be more valuable if it is more persistent and cheaper to trade.

Empirical asset pricing results: Leontief-inverse

Notice that **b** enters homogeneously across all orders of the network effect terms

$$\mathbf{y}_t = \sum_{k=0}^{\infty} A_t^k \rho^k \mathbf{X}_t \mathbf{b} + \epsilon_t, \tag{1}$$

Our empirical results provide evidence of network effects —

- GNN (1st row) vs Leontief (2nd row) shows:
  $\Rightarrow$ GNN's adaptiveness performs better
- Leontief regression (2nd row) vs "up to t-order network terms" (3rd to 7th rows) shows:
  $\Rightarrow$ Higher-order effects in SCG are useful, but GNN wins because it can adaptively weigh these higher-order effects

Multi-step forecasting design

Hold the information set fixed at month t:

$$\mathcal{F}_t = \sigma(\mathbf{X}_t, SCG_t).$$

The baseline one-step model is

$$\hat{Y}_{it} = \hat{g}_i(\mathbf{X}_t, SCG_t),$$

and the h-step extension is

$$\hat{Y}^{(h)}_{i,t+h} = \hat{g}^{(h)}_i(\mathbf{X}_t, SCG_t), \qquad h \in \{0, 1, 3, 6\}.$$

where $\hat{g}^{(h)}_i(\mathbf{X}_t, SCG_t)$ is the trained GNN mapping for forecast horizon h

- No post-t information enters $\hat{Y}^{(h)}_{i,t+h}$.

Transaction cost table: net return and Sharpe

Net return (%)

κ	$h = 0$	$h = 1$	$h = 3$	$h = 6$
0.0000	1.139	0.773	0.562	0.734
0.0003	1.041	0.673	0.531	0.717
0.0010	0.811	0.439	0.457	0.678
0.0030	0.156	-0.228	0.247	0.568
0.0100	-2.138	-2.563	-0.489	0.181

Sharpe ratio

κ	$h = 0$	$h = 1$	$h = 3$	$h = 6$
0.0000	0.156	0.116	0.155	0.241
0.0003	0.143	0.101	0.147	0.236
0.0010	0.111	0.066	0.126	0.223
0.0030	0.021	-0.034	0.068	0.187
0.0100	-0.278	-0.366	-0.136	0.060

- At high costs, **longer** horizons dominate despite **weaker forecast accuracy**.

Turnover explains why horizon rankings change

$h = 0$	$h = 1$	$h = 3$	$h = 6$
3.277	3.336	3.154	3.316

Average turnover per rebalance block

For a proportional cost rate κ, effective annualized cost drag is approximately

$$\kappa \times \text{turnover} \times \frac{12}{h},$$

with $h = 0$ interpreted as monthly rebalancing.

- Per-event turnover is similar across horizons.
- The cost difference comes mainly from how often the portfolio is rebalanced.
- Thus, high-cost settings favor low-turnover horizons even when raw predictability is weaker.

Backtest interpretation under costs

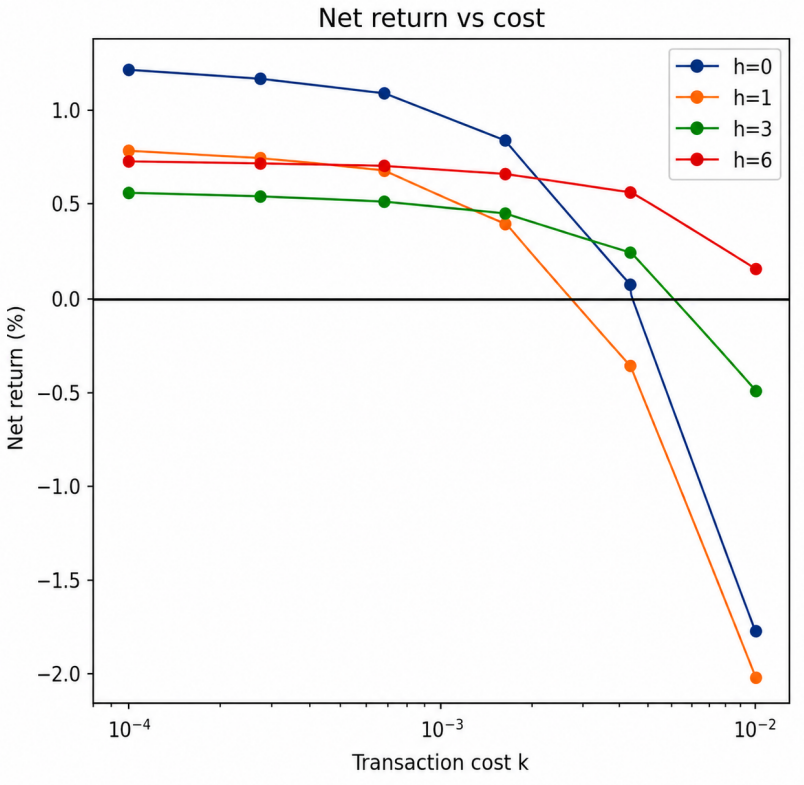
- At $\kappa = 0$, monthly rebalancing earns the highest net return because it exploits the strongest short-run signal.
- As κ rises, monthly rebalancing suffers the steepest cost drag because it trades most frequently.
- A crossover occurs around $\kappa \approx 0.002$: after that, the 6-month horizon produces higher net returns.
- The Sharpe-ratio comparison is even more favorable to $h = 6$, because longer holding periods reduce return volatility in the block-rebalancing construction.

Multi-step forecast horizons before costs

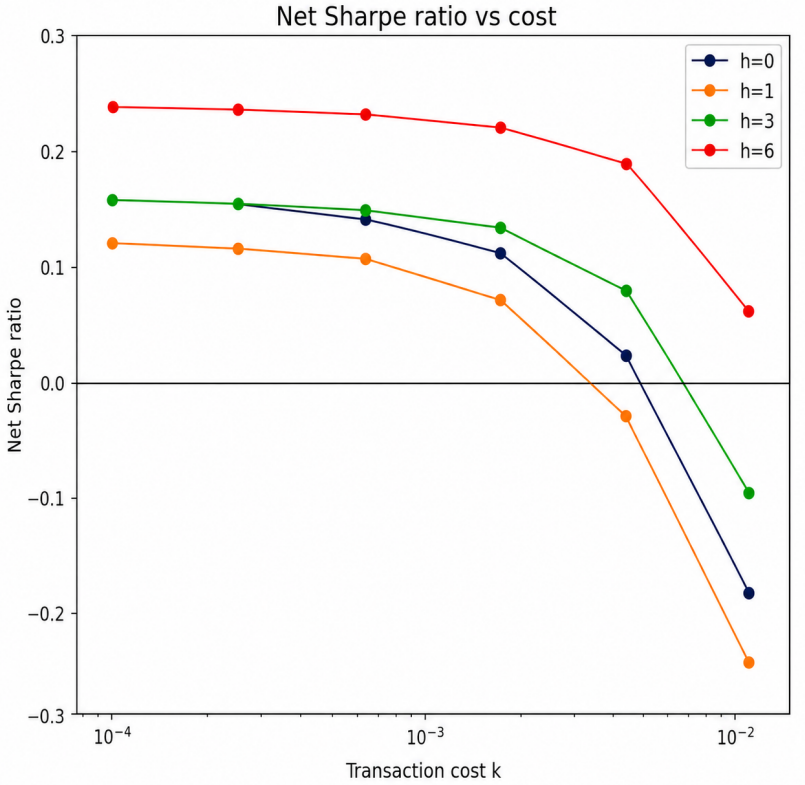
Forecast horizon	ER	Volatility	Sharpe ratio
Baseline ($h = 0$)	0.1106	0.0047	1.3923
$h = 1$ month	0.0096	0.0037	0.1214
$h = 3$ months	0.0068	0.0032	0.0921
$h = 6$ months	0.0066	0.0028	0.0895

- Return predictability is concentrated in the one-step cross-sectional signal.
- Extending the forecast horizon substantially reduces pre-cost expected return and Sharpe ratio.
- Longer horizons lower volatility, which becomes useful when trading costs are introduced.

Transaction costs reshape horizon rankings



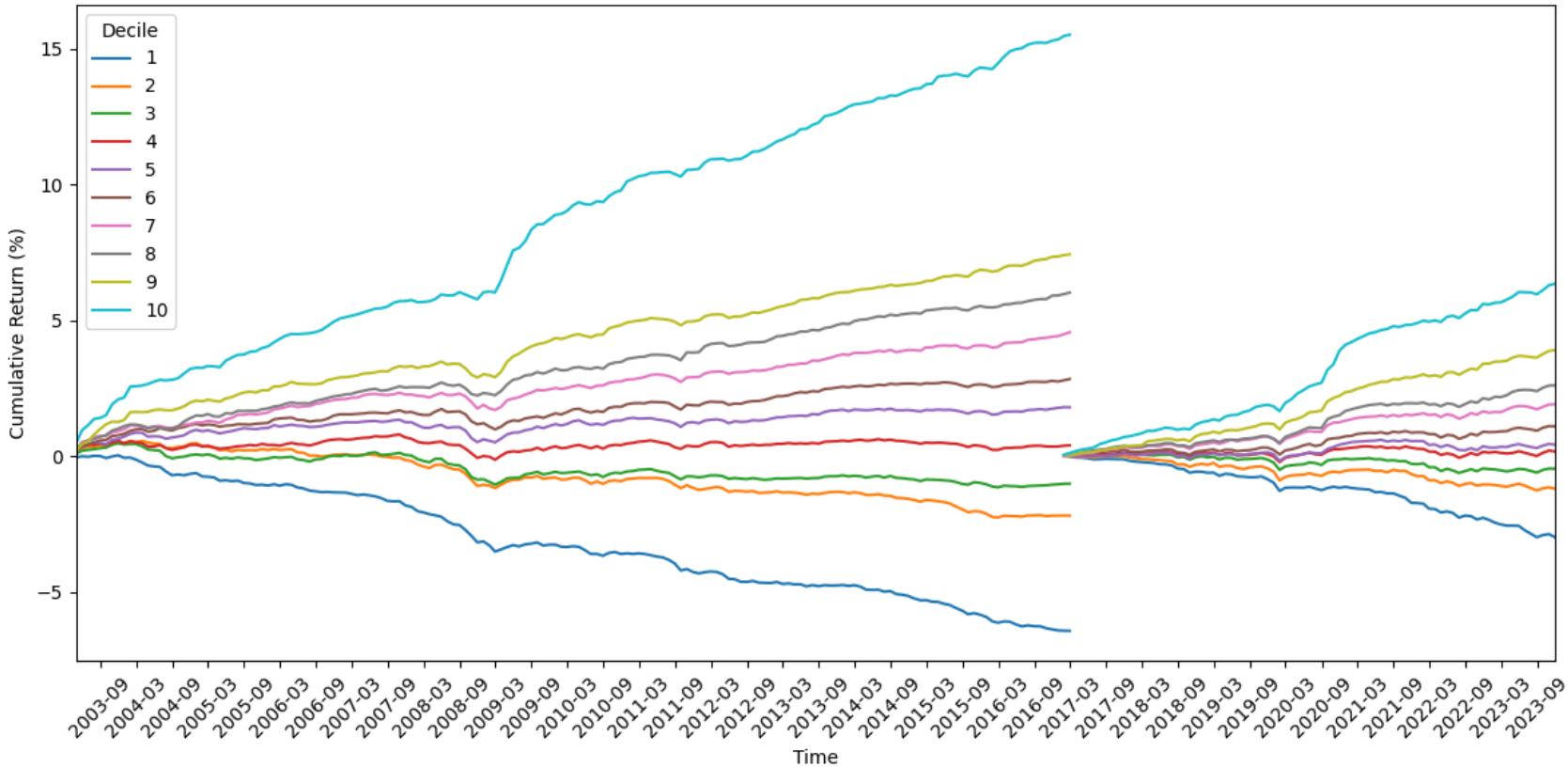
(a) Net return



(b) Sharpe ratio

- Monthly rebalancing has high gross return but is most sensitive to costs.
- The 6-month strategy has the best Sharpe ratio at all cost levels in the (κ, h) block-rebalancing experiment.

Deciles from GNN sorting, trained only once



- Sort firms based on GNN estimation of monthly returns into deciles, calculate equal-weighted portfolio cumulative returns of each decile
- GNN estimated only once and held constant out-of-sample
  $\Rightarrow$ **Clear pattern of top and bottom deciles'** returns out-of-sample remaining positive and negative, respectively

GNN computation: Transformer for f_{msg}, AGG, f_{upd}

- **Message with Multi-Head Attention** (f_{msg}): message from j to i in the lth layer and hth head is:

$$\mathbf{m}_{ij}^{(l,h)} = \text{softmax}\left(\frac{(\mathbf{W}_Q^{(l,h)}\mathbf{h}_i^{(l)})^\top(\mathbf{W}_K^{(l,h)}\mathbf{h}_j^{(l)})}{\sqrt{d_k}}\right)\mathbf{W}_V^{(l,h)}\mathbf{h}_j^{(l)},$$

  with $\mathbf{W}_Q^{(l,h)}, \mathbf{W}_K^{(l,h)}, \mathbf{W}_V^{(l,h)}$: weights for query, key, and value. d_k: dimension of key.

- **Aggregation Across Attention Heads** (AGG): concatenating the outputs of all attention heads and applying a learnable linear transformation:

$$\mathbf{m}_i^{(l)} = \mathbf{W}_O^{(l)} \cdot \text{concat}\left(\mathbf{m}_i^{(l,1)}, \mathbf{m}_i^{(l,2)}, \ldots, \mathbf{m}_i^{(l,H)}\right),$$

  with H: number of attention heads, $\mathbf{W}_O^{(l)}$: weight of heads.

- **Update** (f_{upd}): a feedforward neural network (FFN) with common regularization techniques:

$$\mathbf{h}_i^{(l+1)} = \text{LayerNorm}\left(\mathbf{h}_i^{(l)} + \text{FFN}\left(\text{LayerNorm}(\mathbf{h}_i^{(l)} + \mathbf{m}_i^{(l)})\right)\right).$$

Oversmoothing table: embedding dispersion by depth

Depth d	Average MAD	Median MAD
1	2.6110	2.4210
2	1.3727	1.2897
3	0.1202	0.1101
4	0.1103	0.1028
5	0.0270	0.0237
6	0.0312	0.0294
8	0.0061	0.0061
10	0.0105	0.0105

- Dispersion collapses after moderate depth, showing that firm embeddings become increasingly similar.
- This provides a mechanism for why $d = 10$ loses predictive and trading value.

Out-of-Sample Evaluation of Tangency Portfolios

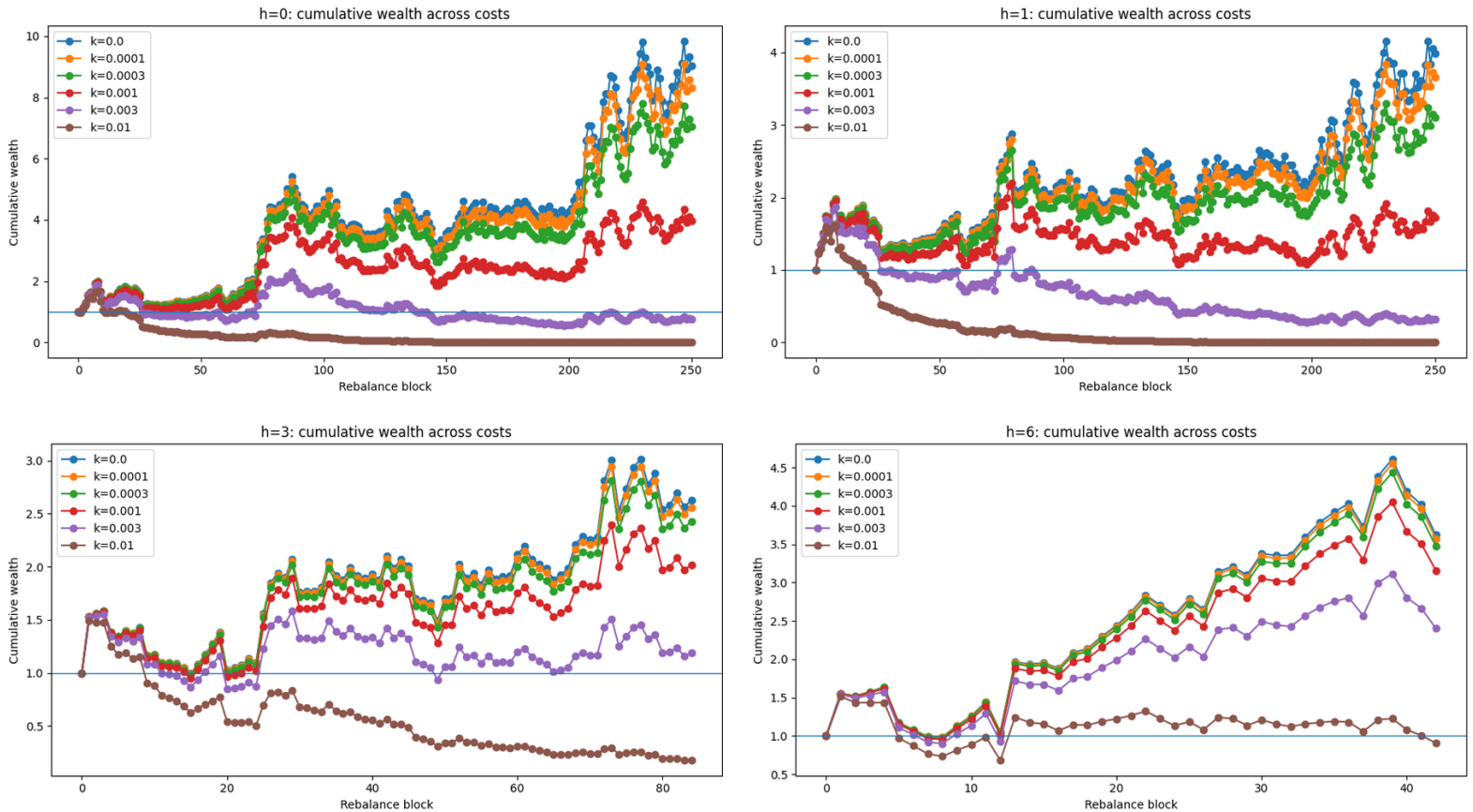
Procedure:

1. **Expanding Window Backtest**
   - At each evaluation period, split the data into:
     - *In-sample*: All historical observations up to the current period.
     - *Out-of-sample*: Returns over the next test window.
   - Standardize factor returns based on in-sample volatility.
2. **Tangency Portfolio Estimation**
   - Compute the tangency (maximum Sharpe ratio) portfolio weights:

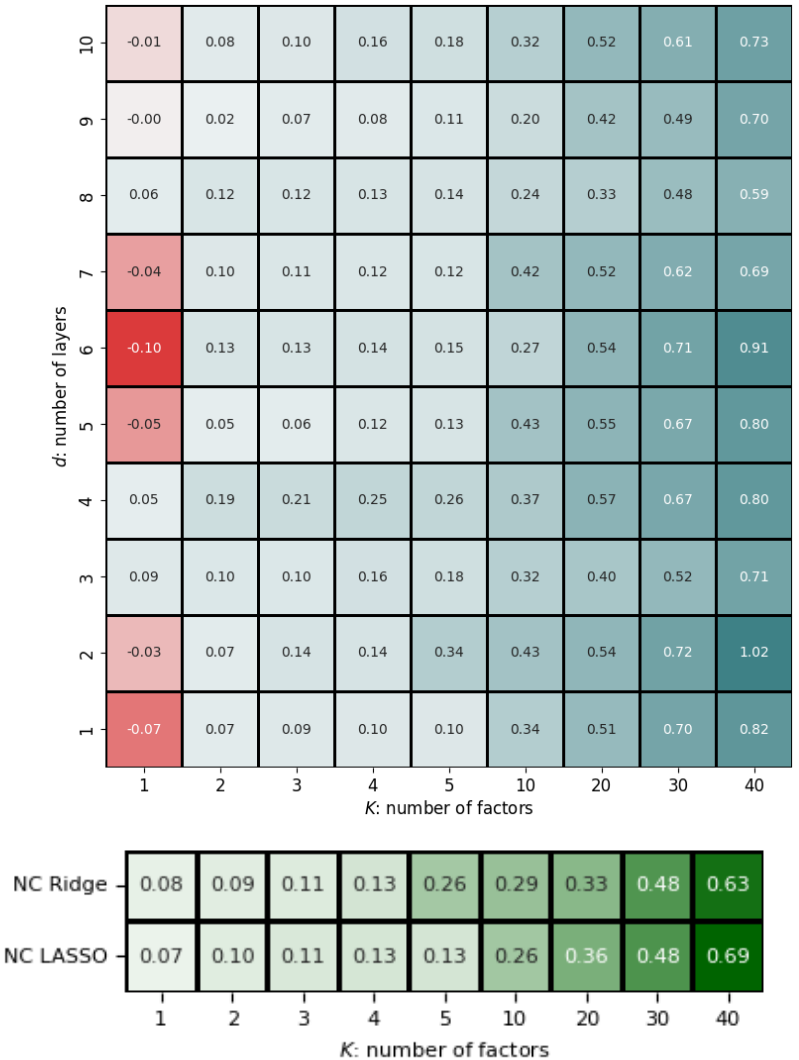
     $$w^* \propto \Sigma^{-1}\mu,$$

     where Σ is the covariance matrix of in-sample factor returns and μ the mean in-sample returns.
   - Calculate out-of-sample portfolio return and Sharpe ratio.
   - Compute *turnover* as the change in portfolio weights from the previous period.
3. **Performance Measurement**
   - Evaluate cumulative net returns after applying transaction cost penalties proportional to turnover.
   - Repeat across different factor configurations and number of factors (K).

Cumulative returns under multi-step horizons



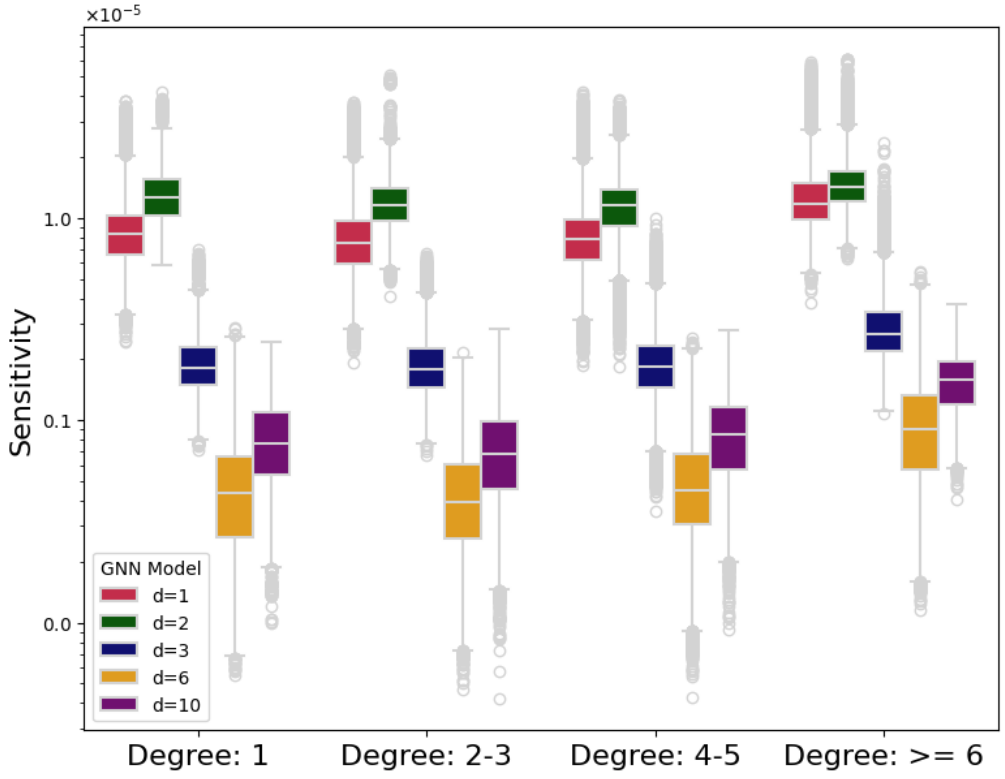
OOS Sharpe ratio, LS on GNN Embeddings+PCA



- Recommended $d = 3$ is better than NC for typical K (e.g. $K = 5$)

  $\Rightarrow$ GNN successfully picks up higher-order relationships, with SR gain

Finding 2: trade-off on complexity of the GNN model



Distributional results of sensitivity in model performance across degree groups and models when we randomly remove edges from SCG's specific group of firms.

- **Network Depth**: model with $d = 6$ exhibits the lowest sensitivity to edge perturbations
- Among the most central firms (degree ≥ 6) in the SCG:
  - 65% lower avg. and 62% lower standard error than the $d = 2$ model
  - $d = 10$ may be overfitting: 74% higher average than the $d = 6$ model

Do Network Features Explain GNN Embeddings?

- Construct clusters of the GNN embeddings via k-means clustering ($k = 2$).
- For each firm characteristic, rank their importance in forming clusters by their significance (ANOVA F-tests).

Characteristic	F-stat	p-val
Avg. Total Degree	449.037	$< .0001$
Avg. Out-Degree	448.895	$< .0001$
Avg. In-Degree	387.5177	$< .0001$

$\Rightarrow$ F-tests show embeddings are differentiated based on **network centrality (degree)** and on firm-specific characteristics (ROC: return on capital, LME: Size as in total market capitalization, Free CF: Cash flow to book value of equity). ⏩ Top 20 Characteristics

Supply chain graph representation is efficient

Graph at time t is $SCG_t = (V_t, E_t)$ with

- Firms as **nodes** V_t (firms with supply chain data)
- Supply chain linkages as **edges** E_t (e_{ijt}: link between firms i and j at t)

Associate each node with 64 characteristics X_{it} and returns Y_{it}

- Joint representation of data and relationships:

$$D_t^{\text{Graph}} = (64 + 1) \times |V_t| + |E_t|.$$

Efficiency:

- Naive alternative is to store pairs of firms $(Y_{it}, \mathbf{x}_{it}, Y_{jt}, \mathbf{x}_{jt}) \in \mathbb{R}^{2(p+1)}$,

$$D_t^{\text{Naive}} = 2 \times (64 + 1) \times |E_t|.$$

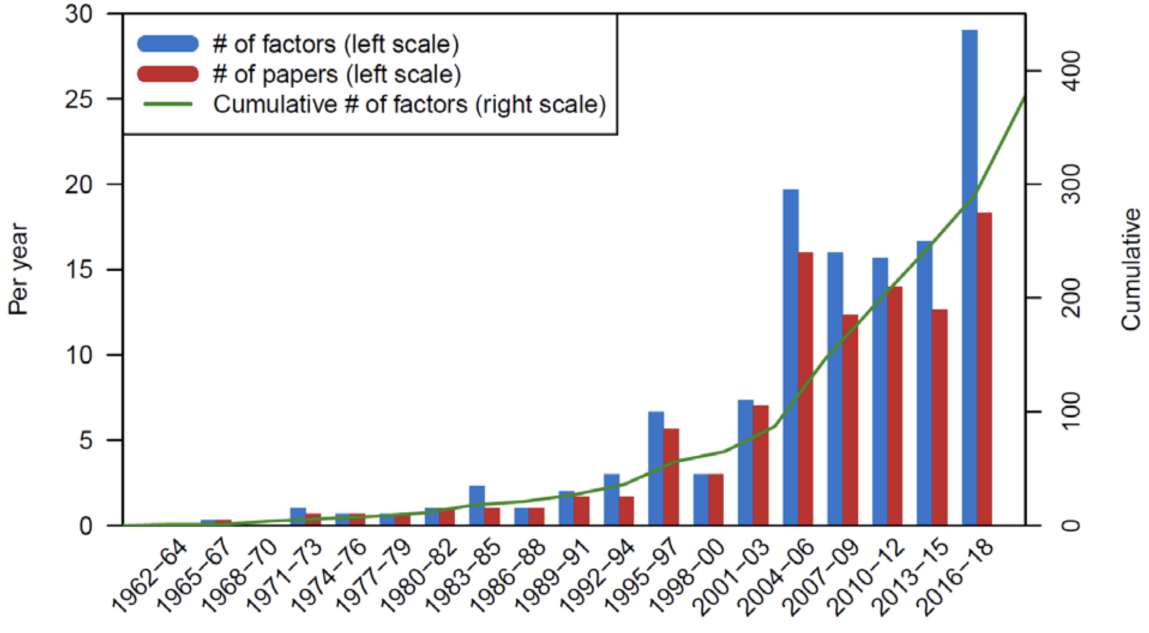
- D_t^{Graph} is more efficient since $|E_t| > |V_t|$
- Worst case: $|E_t| \sim |V_t|^2$, so dimensional gap scales as $|V_t|^2$

The factor zoo in empirical asset pricing

- 3 risk factors of Fama and French [1993] won 2013 Nobel Economics Prize

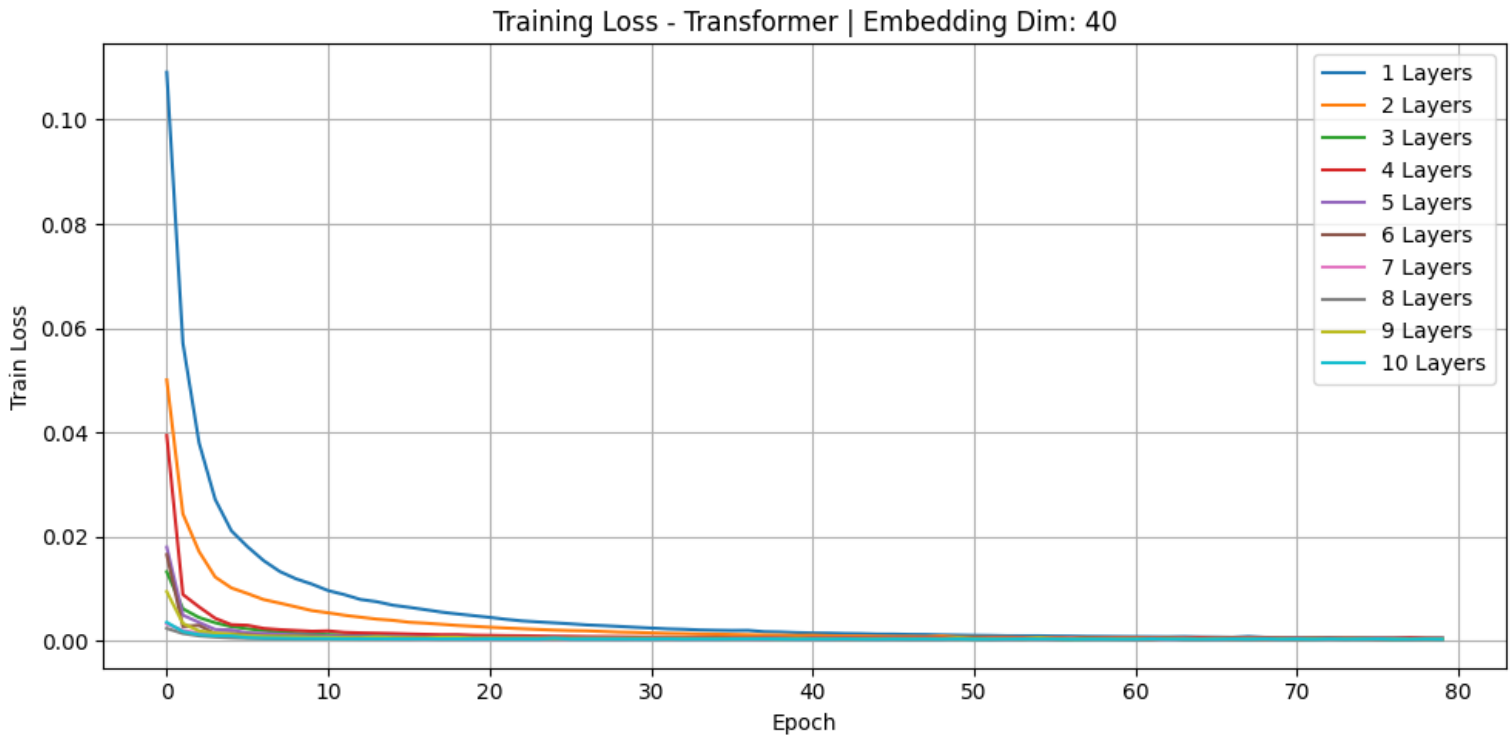
- Harvey and Liu [2022] found 386 factors by end of 2016

Back



Transformer GNN Training Loss

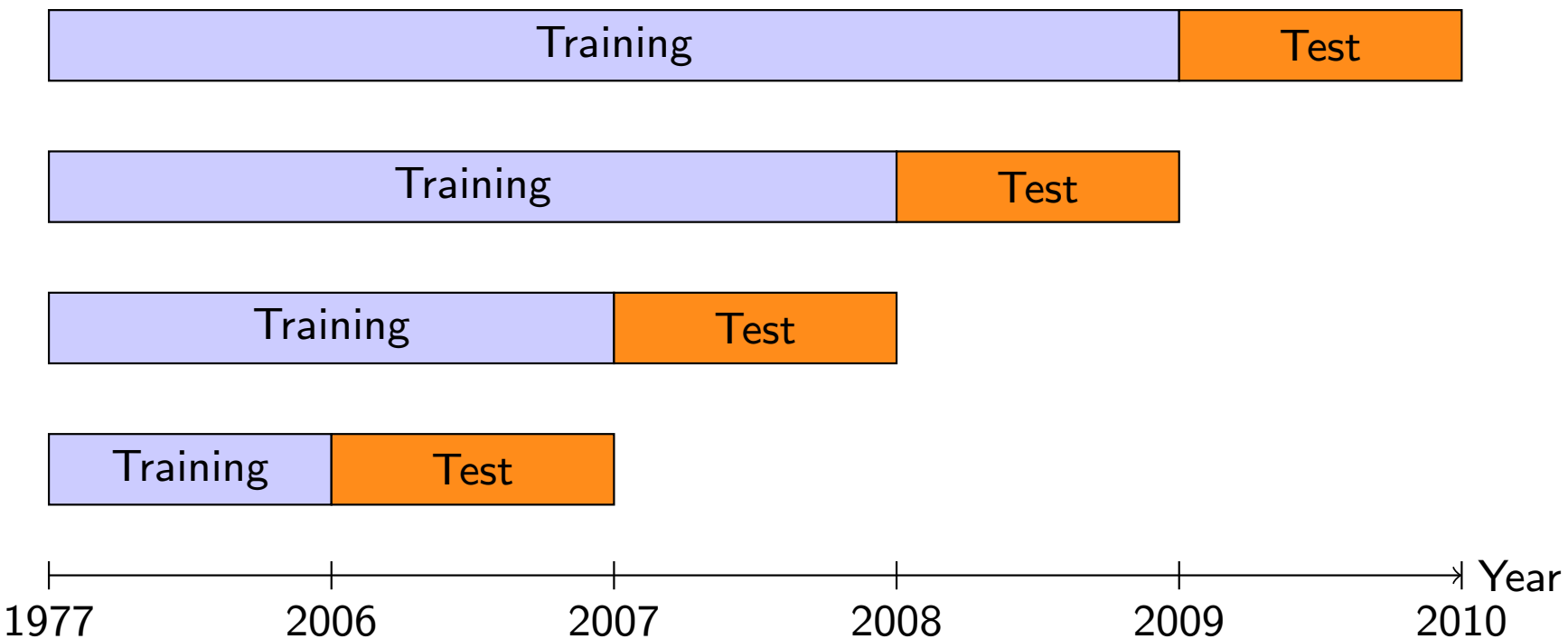
- Our empirical results train for each d a TransformerGNN with the following **hyper-parameter combination:** 80 epochs, dropout of .3, learning rate of .001, and L2 regularization of $1e-4$:



Expanding Window Training Procedure

Training and Test Data Setup:

- Initial training set: January 1977 to December 2006.
- Expands by adding 1 additional year of $(\mathbf{Y}_t, \mathbf{X}_t, \text{SCG}_t)$ data.
- OOS test data: next 1 year. E.g. test on 2007 after training up to 2006.



Benefits of the Expanding Window Approach

Estimation Objective:

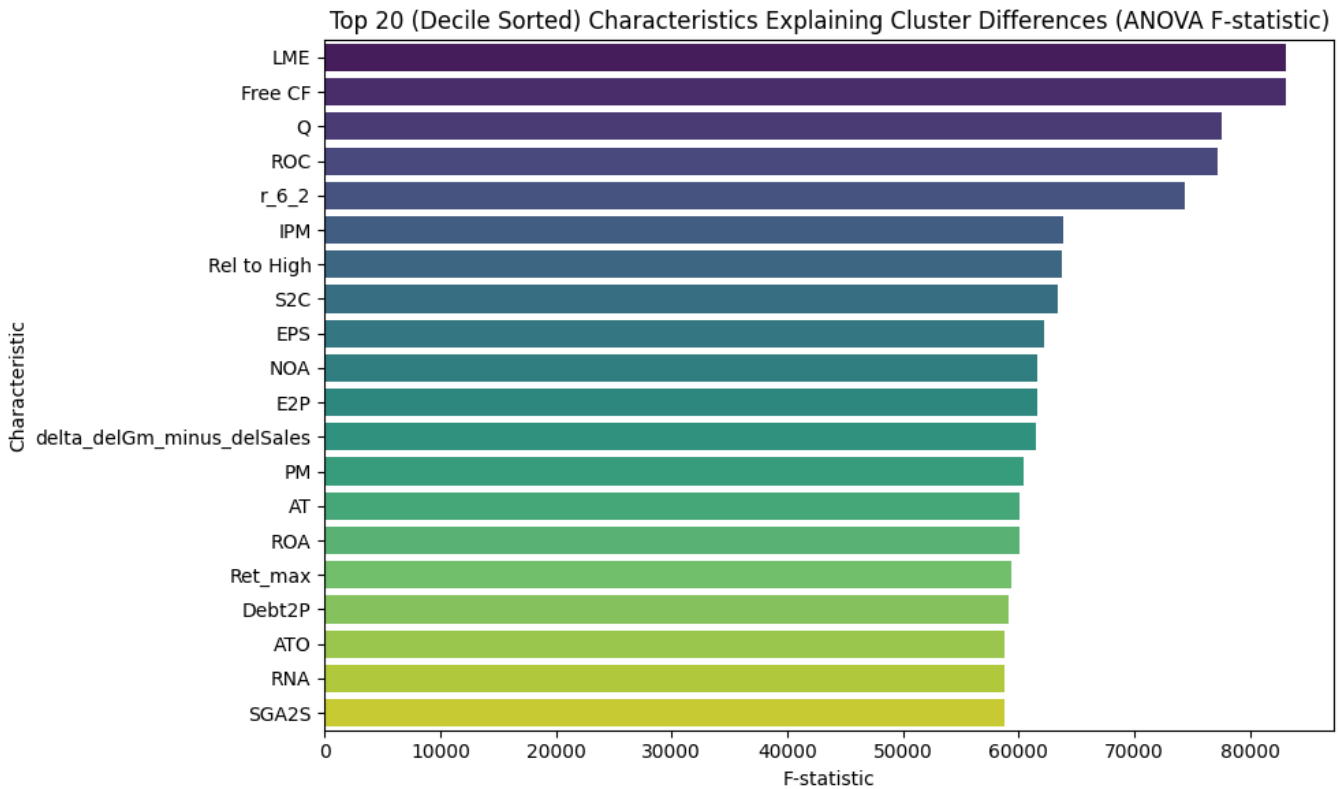
$$\min_{\Theta_t} \ \mathbb{E}_{(i,t')\sim\mathcal{D}_{\text{train},t}} \left[\left(Y_{it'} - g_i(\Theta_t; \mathbf{X}_{t'}, \text{SCG}_{t'}) \right)^2 \right]$$

- Θ_t: Model trained up to year $t-1$.
- Expanding window captures evolving supply chain structure over time.

Performance Metrics:

- Mean Squared Error (MSE) on rolling, out-of-sample basis
- Sharpe Ratio (SR) of portfolios using supply chain factors and PCA factors with trading costs

What do the GNN Embeddings Mean?



All are significant firm-level characteristics, ranked by cross-cluster F-statistics.
LME: Size as in total market capitalization. Free CF: Cash flow to book value of equity. Q: Tobin's Q

Factset Dataset Description

- We use the **Factset relationships data**, and focus on supplier/customer relationship types.
- Factset reports the start and end dates for the relationship between two counterparties $\Rightarrow$ **we focus on monthly relationships**.
- Factset reports the revenue percentage that the source company derived from its relationship in the reporting period.
- Factset does not have records that start before April 1, 2003.
- A single pair of unique companies (nodes) can be connected by more than one edge. $\Rightarrow$ for example two pharmaceutical companies (x and y) with development partnerships for two different drugs (A and B) would result in an x-y edge for drug A, and an x-y edge for drug B.

D. Acemoglu, V. M. Carvalho, A. Ozdaglar, and A. Tahbaz-Salehi. The network origins of aggregate fluctuations. *Econometrica*, 80(5):1977–2016, 2012.

D. Agrawal and N. Osadchiy. Inventory productivity and stock returns in manufacturing networks. *Manufacturing & Service Operations Management*, 26(2): 573–593, 2024.

D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 3438–3445, 2020.

L. Cohen and A. Frazzini. Economic links and predictable returns. *The Journal of Finance*, 63(4):1977–2011, 2008.

W. Ding, R. Levine, C. Lin, and W. Xie. Corporate immunity to the covid-19 pandemic. *Journal of Financial Economics*, 141(2):802–830, 2021.

E. F. Fama and K. R. French. Common risk factors in the returns on stocks and bonds. *Journal of financial economics*, 33(1):3–56, 1993.

E. F. Fama and K. R. French. A five-factor asset pricing model. *Journal of Financial Economics*, 116(1):1–22, 2015. ISSN 0304-405X. doi: https://doi.org/10.1016/j.jfineco.2014.10.010. URL https://www.sciencedirect.com/science/article/pii/S0304405X14002323.

J. Freyberger, A. Neuhierl, and M. Weber. Dissecting characteristics nonparametrically. *The Review of Financial Studies*, 33(5):2326–2377, 2020.

S. Gu, B. Kelly, and D. Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273, 2020.

C. R. Harvey and Y. Liu. Luck versus skill in the cross section of mutual fund returns: Reexamining the evidence. *The Journal of Finance*, 77(3):1921–1966, 2022.

B. T. Kelly, S. Malamud, and L. H. Pedersen. Principal portfolios. *The Journal of Finance*, 78(1):347–387, 2023. doi: 10.1111/jofi.13199.

B. T. Kelly, B. Kuznetsov, S. Malamud, and T. A. Xu. Artificial Intelligence Asset Pricing Models. NBER Working Paper 33351, National Bureau of Economic Research, 2025.

W. Leontief. *Input-output economics*. Oxford University Press, 1986.

M. Lettau and M. Pelger. Factors that fit the time series and cross-section of stock returns. *The Review of Financial Studies*, 33(5):2274–2325, 2020.