**ANL 488 PROJECT PROPOSAL**

**Optimising Portfolio Allocation with Machine Learning Techniques**

**Submitted by**

**Kam Jia Kai, Brian Isaac**

**SCHOOL OF BUSINESS**

**Singapore University of**

**Social Sciences**

**Presented to Singapore University of Social**

**Sciences in partial fulfillment of the**

**requirements for the**

**Degree of Bachelor of Science**

**in Business Analytics**

**2025**

**Table of Contents**

[Chapter 1: Introduction 3](#_Toc207923954)

[1.1 Motivation 3](#_Toc207923955)

[1.2 Project Aim 4](#_Toc207923956)

[1.3 Objectives 5](#_Toc207923957)

[1.4 Deliverables 6](#_Toc207923958)

[Chapter 2: Literature Review 7](#_Toc207923959)

[2.1 Classical Foundations and Practical Limits 7](#_Toc207923960)

[2.2 Network Filtering 7](#_Toc207923961)

[2.3 Hierarchical Risk Parity (HRP): Clustering-Aware Risk Allocation 8](#_Toc207923962)

[2.4 Predictive Deep Learning for Allocation via LSTM 8](#_Toc207923963)

[2.5 GNNs and GATs for Direct Weight Optimisation 8](#_Toc207923964)

[2.6 Cross Study Analysis 9](#_Toc207923965)

[Chapter 3: Data Understanding and Preparation 11](#_Toc207923966)

[3.1 Data Understanding 11](#_Toc207923967)

[3.1.1 Data Collection (with Python Script) 11](#_Toc207923968)

[3.1.2 Data Exploration (with Python Notebook) 12](#_Toc207923969)

[3.1.3 Data Quality 16](#_Toc207923970)

[3.2 Data Preparation 17](#_Toc207923971)

[3.2.1 Data Selection 17](#_Toc207923972)

[3.2.2 Data Cleaning 18](#_Toc207923973)

[3.2.3 Feature Engineering and Graph Construction 18](#_Toc207923974)

[Chapter 4: Proposed Modelling and Evaluation 21](#_Toc207923975)

[4.1 EW and MV (Baselines) 22](#_Toc207923976)

[4.2 HRP 23](#_Toc207923977)

[4.3 LSTM 24](#_Toc207923978)

[4.4 Graph Attention Network (GAT) 25](#_Toc207923979)

[4.5 Evaluation Pipeline and Metrics 25](#_Toc207923980)

[Chapter 5: Summary 28](#_Toc207923981)

[Chapter 6: Proposed Schedule 29](#_Toc207923982)

[References 30](#_Toc207923983)

[Appendix A: Building Membership from Wikipedia 32](#_Toc207923984)

[Appendix B: Downloading Stooq Ticker Data 34](#_Toc207923985)

[Appendix C: Downloading and Augmenting Stooq Data with Yahoo Finance Data 37](#_Toc207923986)

[Appendix D: Data Exploration Notebook Code Blocks 41](#_Toc207923987)

# Chapter 1: Introduction

## 1.1 Motivation

Portfolio Optimisation is a fundamental problem in finance: “How to allocate capital among a set of assets to maximise return for a given level of risk?”. A classical approach, the Markowitz mean-variance model, seeks to optimise trade-off between expected return and variance of portfolio returns (Markowitz, 1952). This however poses a few practical challenges, one of which assumes normally distributed returns and requires estimating expected returns and covariances accurately, leading to poor out of sample performance (Siegel & Woodgate, 2007). In large-scale portfolios with many assets, estimation errors and high dimensionality, this further undermines the effectiveness of traditional models. In response to these challenges, researchers and practitioners have shown growing interest in data-driven and machine learning approaches for portfolio allocation. Rather than relying purely on static estimates of means and covariances, these approaches leverage algorithms and richer data structures to improve diversification and resilience.

One such approach is Hierarchical Risk Parity (HRP), a clustering-based allocation technique introduced by López de Prado in 2016. HRP is a graph-theoretic and machine learning inspired alternative to Markowitz mean-variance optimisation. Instead of inverting the covariance matrix, it builds a hierarchy of assets using correlation distance and allocates capital according to a tree structure, avoiding the need for a positive-definite covariance matrix and leads to more stable allocations (Lopez de Prado, 2016). This demonstrates the power of incorporating unsupervised learning – hierarchical clustering, into portfolio construction.

Another promising approach is the use of deep neural networks like Long Short-Term Memory (LSTM) to model temporal patterns in asset returns. LSTMs are a type of Recurrent Neural Network (RNN) specifically designed to capture long-term dependencies in sequential data (Hochreiter & Schmidhuber, 1997). Financial time series, such as asset prices and returns, are inherently sequential and often non-linear, making LSTMs a natural choice to forecast future market movements or risk. By capturing complex temporal structures that span multiple time horizons, an LSTM model can inform a dynamic allocation strategy, tilting portfolio toward assets predicted to outperform. This data-driven forecasting approach could overcome the backward-looking bias of MPT, which uses historical average returns, by adaptively learning from more recent and higher-frequency data.

A more recent and advanced approach involves Graph Neural Networks (GNNs), a class of deep learning methods designed to perform inference on data described by graphs. GNNs can learn representations of each node, in our case an asset, that incorporate information from its neighbours, a related asset. Potentially uncovering non-linear and higher-order relationships that would be missed by traditional Euclidean data models (Merritt, 2022). Graph Attention Networks (GATs), a type of GNN makes use of a self-attention mechanism on graph nodes, allowing the model to learn the relative importance of neighbouring nodes adaptively, rather than using a fixed aggregation scheme (Veličković et al., 2018). Their ability to handle high dimensionality input for a corpus of assets with numerous features, and to incorporate custom layers make them appealing for large-scale portfolio problems.

All in all, the motivation for this project stems from both the limitations of classical portfolio theory and the opportunities modern machine learning techniques offer. We aim to explore a broader range of techniques for portfolio allocation, improving portfolio performance in terms of risk-adjusted returns.

## 1.2 Project Aim

The aim of this project is to develop and evaluate machine learning based portfolio allocation strategies that improve risk-adjusted returns over classical approaches while satisfying real-world constraints. The project will investigate the following three methods:

1. Hierarchical Risk Parity
2. Long Short-Term Memory (LSTM)
3. Graph Attention Networks (GATs)

We investigate the application of these methods to construct equity portfolios that maximise the Sharpe ratio under practical restrictions such as long-only positions, limited turnover, transaction cost awareness and sparse asset holding. We aim to identify a modelling approach that delivers superior out-of-sample performance and robust diversification for investors. By the end of the project, we will deliver the results of portfolio allocation frameworks from the three approaches to determine the applicability of them.

## 1.3 Objectives

To fulfil the project aim, objectives are defined and shown as below.

Literature Review

Conduct literature review on portfolio optimisation, covering the limitations of classical methods, as well as the recent advances in machine learning methods. The review will establish and justify the selection of HRP, LSTM and GATs as candidate techniques, establishing possible baseline metrics.

Dataset Creation

We will assemble a comprehensive dataset for backtesting portfolio. The key task would be to obtain a survivorship-bias-free universe of assets for the chosen index or market, consisting of a time-indexed panel of asset returns, with dynamic membership reflecting real index history, ready for use in simulation and modelling.

Model Implementation

Developing implementations for the selected machine learning approaches as mentioned earlier, and incorporating realistic constraints into portfolio construction and rebalancing process, aiding the backtesting and evaluation process.

Outcome Analysis and Model Refinement

Based on the backtest results, we will analyse the approaches. If one method underperforms we will diagnose potential reasons. This phase may involve iterating of model design, and the objective is to not just find the best winner, but also to gain insights into the conditions in which each technique excels.

Business Implications and Recommendations

Lastly, we will translate the technical findings into a business context, outlining the possible implications for portfolio managers or investors of a particular ML approach does show promising results. This includes the feasibility of the implementation, robustness and the risks identified. The deliverable will include a clear recommendation of the selected approach for deployment in a practical portfolio management scenario, or a discussion of why an ML approach may or may not be worth pursuing further in this domain.

In summary, the result will be a comprehensive assessment of machine learning techniques for portfolio allocation.

## 1.4 Deliverables

By the end of this project, we expect the following deliverables

1. Project Report: documenting all aspects of the work. This includes the motivation, a literature review, data description and preparation steps, methodology and model design. A compilation of results from backtest and business recommendations.
2. Reproducible Code and Dataset: A codebase containing the code developed for data collection, processing and model implementation.

# Chapter 2: Literature Review

This review is organised to ground our choices and claims. We will begin with the foundations of portfolio construction via classical mean-variance and robust baselines to anchor evaluation. Then we will examine network filters that justify building sparse, denoised correlation graphs. Next, we cover clustering aware allocation via HRP and temporal sequence models such as LSTMs. We proceed to end-to-end deep learning that optimises Sharpe directly, focusing on GNNs – GATs for relation aware allocation with neighbour specific attention. Finally, we review application-scale GAT portfolio studies to de-risk these design choices and set comparative baselines.

## 2.1 Classical Foundations and Practical Limits

Modern Portfolio Theory (MPT), introduced by Harry Markowitz in 1952, provides the groundwork for quantitative portfolio selection. Markowitz approach formalised the idea of an efficient frontier, a set of portfolios offering the maximum expected return for a given level of risk (Markowitz, 1952). The mean-variance model has been widely adopted due to its intuitive appeal but have been critiqued on its practical use, (Siegel & Woodgate, 2007) found that naïve equal-weighted portfolios often outperformed mean-variance optimised portfolios out of sample, showcasing the latters sensitivity to estimation error. Similarly, (Guidolin & Ria, 2010) highlighted how real world issues like non-normal return distributions, regime shifts, and transaction costs violate MPTs assumptions. These findings exemplify a need for an improved or alternative optimisation technique that can better handle real market complexities.

## 2.2 Network Filtering

One branch of research focuses on the use of network science and graph theory in portfolio construction. Representing assets as a network helps captures the dependence structure between asset returns. Minimum Spanning Tree (MST) introduced by Mantegna, builds on the idea of filtering stock correlation matrices to extract a meaningful network that can guide diversification (Mantegna, 1999). Information filtering methods like Triangulated Maximally Filtered Graph (TMFG) create sparse graphs that retain the most significant connections from a dense correlation matrix (Massara et al., 2017). These networks highlight clusters of closely related stocks and isolated peripheral stocks. Building onto this, studies like (Pozzi et al., 2013) show that investing in these peripheral nodes with low connectivity in the stock network reduces risk concentration and improves returns. This shows that network metrics can be crucial signals for portfolio allocation.

## 2.3 Hierarchical Risk Parity (HRP): Clustering-Aware Risk Allocation

In the paper *Building diversified portfolios that outperform out-of-sample* (Lopez de Prado, 2016)*.* Lopez introduced HRP, built to address key weaknesses of quadratic optimisers, leading to instability, concentration and underperformance. It clusters assets by correlation distance, quasi-diagonalises the covariance, and allocates risk top-down via recursive bisection, avoiding unstable matrix inversion. Out of sample HRP has been shown to deliver lower realised risk than minimum variance despite its heuristic nature.

## 2.4 Predictive Deep Learning for Allocation via LSTM

In the paper *Deep learning with long short-term memory networks for financial market predictions* (Fischer & Krauss, 2018)*.* LSTM was applied to S&P 500 constituents, reporting ~0.46% daily return (Sharpe ≈ 5.8) before transaction costs, outperforming classic classifiers and yielded economically meaningful performance (pre-cost), illustrating the value of learned temporal structure for portfolio tilts.

## 2.5 GNNs and GATs for Direct Weight Optimisation

Advances in machine learning and deep learning have opened new avenues for portfolio optimisation. Instead of using hand-crafted metrics from networks, we can train models to learn the optimal patterns from data. Within such models, GNNs have gained traction for financial applications due to them accommodating the relational structure between entities like companies or assets. Such applications include a Temporal Graph Convolutional Network to rank stocks for portfolio selection, constructing an evolving graph based on firms return correlations (Uddin et al., 2023). However, such applications primarily aimed at forecasting returns or rankings rather than directly optimising portfolio weights, with our focus on the latter to use GNNs to determine allocations. GATs, introduced by Veličković, is particularly well-suited for this task due to its ability to learn attention weights for each edge based on the data.

The most directly relevant study to our project *is Large-scale Time-Varying Portfolio Optimisation using Graph Attention Networks* by (Korangi et al., 2022), which applied GATs to a large-scale portfolio of US mid-cap stocks. They constructed graphs using a distance correlation measure and TMFG filtering to connect stocks and trained a GAT model to output portfolio weights. Notably, they incorporated custom layers in the network to enforce portfolio constraints and used a loss function based on the Sharpe ratio. By training the GAT to maximise risk-adjusted return, their model achieved an annualised Sharpe ratio of 1.08 on test data, outperforming benchmarks including a network-based heuristic portfolio, a mean-variance efficient portfolio, and an equal-weight portfolio (Korangi et al., 2022). This provides proof that learning directly from a relational graphical data with an objective can yield superior portfolios. Our project builds on these literature insights, aiming to contribute by applying similar GAT approaches to a different dataset and by exploring any necessary adjustments to improve performance.

## 2.6 Cross Study Analysis

For the three approaches, although Sharpe values were reported in papers, they usually are compared only against specific, differing baselines. Furthermore, Sharpes across papers arent like-for-like, with universes, periods, constraints, costs and objectives all differing. Hence, it is imperative during evaluation and iteration to do a side-by-side comparison with the same settings to yield a fair, decision-relevant comparison.

All in all, the literature suggests the following:

1. Traditional mean-variance optimisation, while foundational, is fragile to estimation error and can be improved by using robust baselines and richer information.
2. Network based analysis emphasise the importance of asset relationships, tilting towards less-connected assets can enhance diversification.
3. Machine learning models like HRP and LSTM can also be applied as a framework for portfolio allocation, improving over traditional means.
4. GNNs, and GATs provide a powerful framework to capture complex dependency structures in financial data and have been successfully applied in recent research to portfolio optimisation.

This project’s methodology is grounded in these findings. We will compare HRP, LSTM, and GAT under a unified, cost-aware evaluation. All models will be long-only, top-k constrained, and turnover-controlled with their performance assessed via rolling backtests with net-of-cost Sharpe, CAGR, volatility and drawdown. This design lets us quantify the incremental value of all the models. The next sections detail the data and modelling setup that implement this comparative framework.

# Chapter 3: Data Understanding and Preparation

## 3.1 Data Understanding

## 3.1.1 Data Collection (with Python Script)

A diagram of a flowchart

AI-generated content may be incorrect.

***Fig 1: Data Collection Pipeline***

Our dataset is built to create a dynamic equity universe sourced from Wikipedias historical membership of the S&P MidCap 400 (SP400) (“List of S&P 400 Companies,” 2025); *Fig. 1* shows the data collection pipeline as a graph for better depiction. From the scraped membership table we retain, for each ticker, its first inclusion and any subsequent entry and exit dates, allowing for constituents to enter and leave the universe over time, avoiding survivorship bias and enables time-varying graphs in which nodes, as assets, genuinely appear and disappear.

Market data are compiled at daily frequency from two sources. We first download wide parquet panels – dates and respective tickers of adjusted close and volume from Stooq (*An Introduction to Stooq Pricing Data | QuantStart*, n.d.). We then augment coverage with Yahoo Finance (yfinance), filling Stooq gaps only where Stooq is missing. In overlapping segments, the Yahoo price path is level-aligned to the Stooq baseline via a median ratio over the common window to prevent sticking artefacts, volumes are filled directly. The output of this pipeline is a pair of merged artefacts, `prices.parquet` and `volume.parquet`, which preserves membership dynamics without imposing continuous trading histories on every constituent. Throughout the experiment phase, the risk-free rate will be set to 0 for Sharpe computations – this can be swapped for a short-tenor Treasury series. The main code functions for universe sourcing, downloading OHLCV from Stooq and Yahoo Finance can be seen across *Appendix A, B and C*.

Below *Table 1* summarises the merged panels and the membership file, generated from *Code 13 in Appendix D.*

|  |  |
| --- | --- |
| **Item** | **Value** |
| Membership rows | **752** |
| Merged prices shape (dates × tickers) | **3,774 × 853** |
| Merged volume shape (dates × tickers) | **3,774 × 853** |
| Date span | **2010-01-04 → 2024-12-31** |
| Unique tickers (merged) | **853** |

***Table 1: Panel Summary (Post-merge)***

## 3.1.2 Data Exploration (with Python Notebook)

Using the dynamic SP400 membership file with `ticker`, `start` and `end` and the merged market panels, we derive a daily returns panel and a monthly rebalance calendar, then profile universe dynamics, coverage, and basic distribution shape.

As shown in *Fig 2*, the investable universe expands materially over the sample. Active constituents rise from near zero prior to 2015 to approximately 290 by end of 2024, reflecting recorded entries into the SP400 index. Generated from *Code 14 in Appendix D.*

A graph of growth in a graph

AI-generated content may be incorrect.

***Fig 2: Universe Size Through Time***

Annual entries and exits shown in *Figs 3 and 4* confirm non-trivial churn of 752 entries and 458 exits overall, with notable waves of rotation in 2016, 2017 and again in 2023. This churn is what drives the time-varying node set used in our graph snapshots later. Generated from *Code 16 in Appendix D.*

A graph of blue bars

AI-generated content may be incorrect.

***Fig 3: Membership Entries Per Year***

A graph of blue bars

AI-generated content may be incorrect.

***Fig 4: Membership Exits Per Year***

The availability heatmap for a random subset *Fig 5* , yellow highlight being present, shows that most series are contiguous, with the gaps concentrated around late entries and early exits – a show of membership dynamics instead of data loss. Furthermore, *Fig 6* shows the aggregate daily coverage share for prices, trending upwards from 0.55 to 0.78 by 2024, with transient dips around calendar effects and membership transitions. The merged panels therefore offer high completeness without imposing continuous histories on every constituent. Generated from *Code 15 and 18 in Appendix D.*

A yellow and purple lines

AI-generated content may be incorrect.

***Fig 5: Availability Heatmap for Tickers***

A graph showing the growth of a company

AI-generated content may be incorrect.

***Fig 6: Daily Coverage Share***

We focus on liquidity since it constrains deployability. *Fig 7*, the raw daily volume distribution is extremely right skewed with a skew of 42.24, indicating that a small subset of names accounts for a disproportionate share of trading activity. This concentration motivates reporting turnover and capacity alongside performance in later sections. Generated from *Code 17 in Appendix D.*

A graph with numbers and letters

AI-generated content may be incorrect.

***Fig 7: Distribution of Daily Volume***

In summary, these confirms that the dataset captures a realistic, dynamic universe with sufficient coverage for building time-varying correlation graphs and training allocation models.

## 3.1.3 Data Quality

We verify the dataset at two layers to surface issues that require remediation.

Firstly, membership integrity, ticker symbols are normalised across sources (share-class dot vs dash). Normalisation is handled consistently by the symbol mappers in the ingestion scripts under the functions `\_to\_stooq\_symbol` in `download\_stooq.py` and `\_yahoo\_symbol\_map` in `augment\_with\_yfinance.py` for Stooq and Yahoo respectively. Sanity-check the membership table for chronology and uniqueness: for each ticker, start ≤ end, ranges do not overlap, and duplicates are removed. Finally, we confirm that the cleaned membership symbols exist in at least one market panel (merged Stooq+Yahoo).

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 1: QA Checks on Membership***

From *Code 1*, we can see that our audit has 15 interval that were inverted. These intervals will be corrected during the Data Preparation stage.

Next, panel hygiene. Series are aligned to a master trading calendar `ingest\_stooq\_to\_hydra.py` scripts `\_read\_panel` function and `data.py` scripts `\_align\_calendar` functions. For analysis, returns are computed only where consecutive prices exist with no fabricated returns across long gaps, and extreme one-day moves are flagged, not dropped, winsorisation remains off by default with hooks living in `make\_graphs.py` script. We also confirm minimum observation overlap before correlation estimation to avoid spurious edges under `make\_graphs.py` script using `\_prune\_min\_overlap` function.

A screenshot of a computer screen

AI-generated content may be incorrect.

***Code 2: QA Checks on Panel Hygiene***

From Code 2 we can see that the merged price panels use a `DatetimeIndex` which is increasing and contains no duplicate timestamps. The distribution of index day gaps concentrates at 1 day with small masses at 3-5 days, consistent with weekends and market holidays.

## 3.2 Data Preparation

## 3.2.1 Data Selection

We work from the merged daily panels `prices.parquet` and `volume.parquet` and the dynamic membership file. On each trading day, the active universe is defined by the membership intervals start to end, and the analysis window is restricted to 2016–2024 (inclusive). From adjusted closes we compute daily returns only where both and exist ensuring no returns are fabricated across gaps, and persist an aligned returns panel as `returns\_daily.parquet`. A monthly rebalance calendar, the last trading day of each month, anchors both the feature windows and the graph construction. All of this is implemented in `make\_dataset.py` based on selection, returns and rebalance dates.

## 3.2.2 Data Cleaning

Cleaning consists of symbol normalisation, examples shown, calendar alignment to the trading calendar without synthetic return creation, and reliance on source adjusted close for corporate actions to avoid double adjustments. Outlier handling via winsorisation is available but off by default so the modelling pipeline would absorb the tail risks. The canonical analysis-ready panels are thus the merged `prices.parquet`, `volume.parquet`, and the derived `returns\_daily.parquet`.

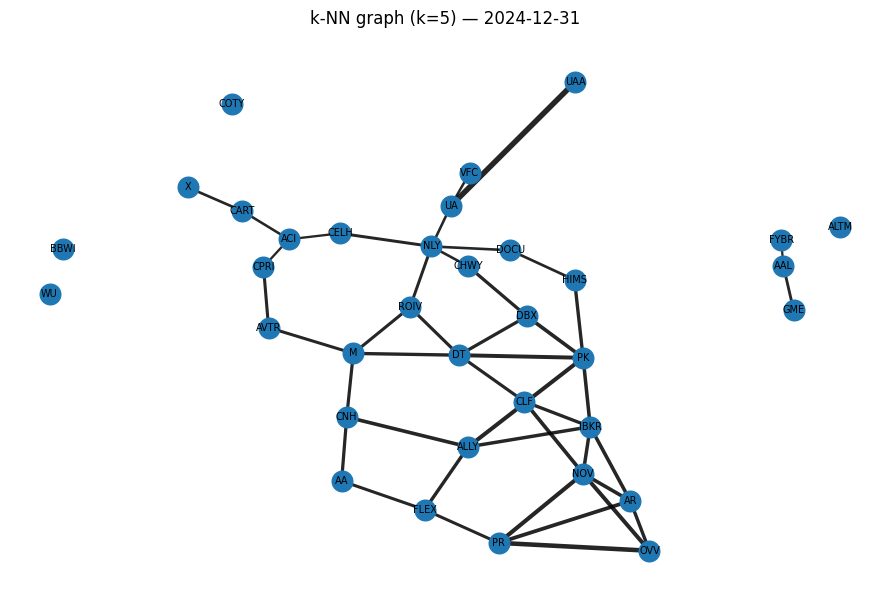
## 3.2.3 Feature Engineering and Graph Construction

At each rebalance date 𝑡, we build node features from a backward-looking 60-trading-day window of daily returns that ends at 𝑡. For each active asset, the last 60 returns are stacked into a vector and then standardised cross-sectionally within the window to stabilise training and avoid scale effects.

Using the same lookback, we estimate a robust correlation matrix for the active universe and prune sparse histories before estimation (minimum observation overlap). Correlations are produced via `graph.py` script and `corr\_from\_returns` function which delegates to shrinkage routines in `cov.py` script to reduce sampling noise in short windows.

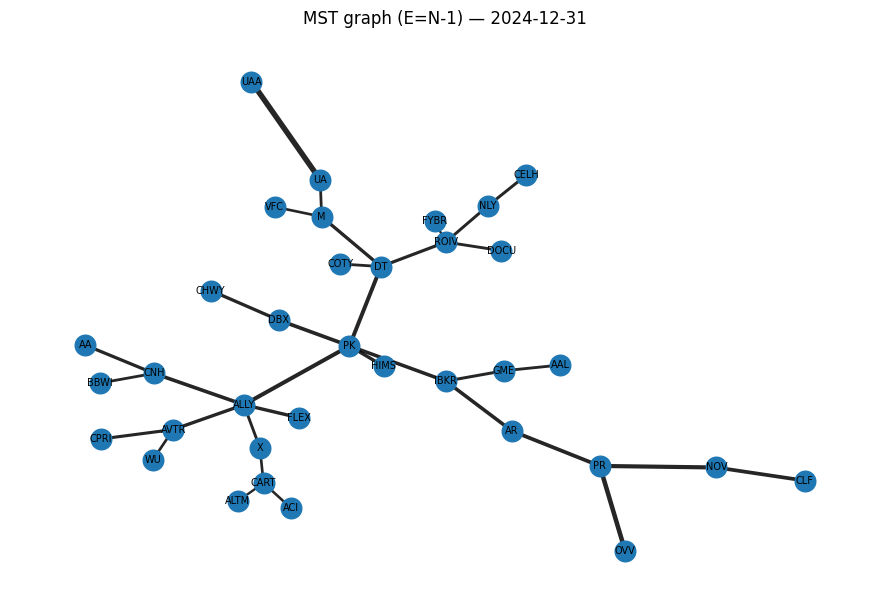
We then filter this dense correlation into a sparse, time-varying graph, choosing one of three implemented methods in `graph.py` script with `edges\_from\_corr` function, all operating on ∣𝜌∣ and producing an undirected graph with edge attributes [𝜌, ∣𝜌∣, sign], the illustrated charts were generated using *Code 22 and 23 in Appendix D*:

* k-NN (mutual ∣𝜌∣-nearest neighbours): for each node, keep its top-k neighbours by∣𝜌∣, then mutualise (edge kept only if 𝑖 is in 𝑗s top-k and vice-versa). Degrees concentrate near 𝑘; the graph can be disconnected, which preserves local cluster structure.



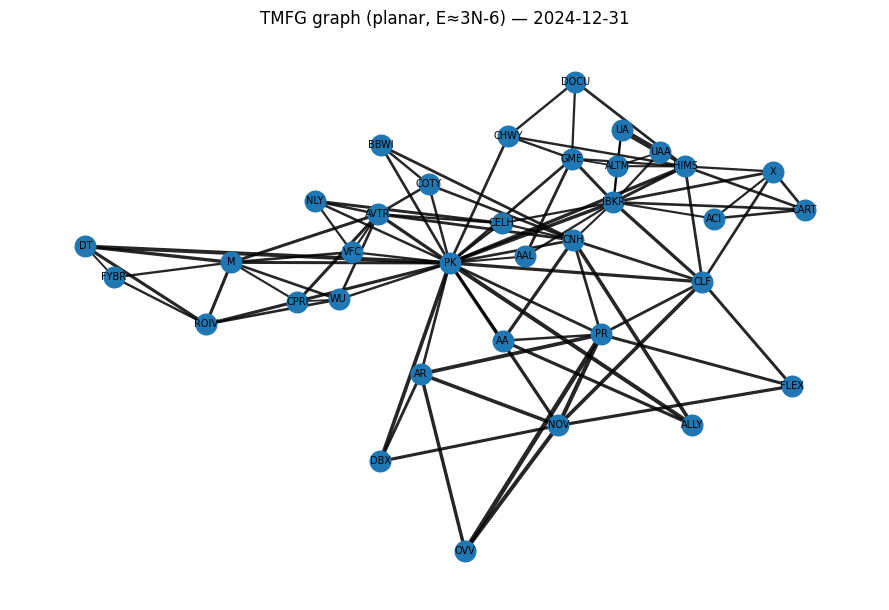
***Fig 8: Mutual k-nearest neighbour graph, k = 5***

* MST (Minimum Spanning Tree): convert correlation to a distance =2(1−∣∣) and run a Prim-style MST. This yields exactly 𝑁−1 edges and is always connected, providing a conservative “backbone” of relationships.



***Fig 9: Minimum Spanning Tree on correlation-derived distance***

* TMFG (Triangulated Maximally Filtered Graph): a greedy planar triangulation on ∣𝜌∣ (seed ; iterative face insertion) that returns 3𝑁−6 edges for 𝑁≥3. TMFG preserves more structure than MST while avoiding dense hairballs and typically captures sector/cluster geometry well.



***Fig 10: Triangulated Maximally Filtered Graph***

Because membership is dynamic, the node set varies month-to-month; each snapshot is built only on assets active at 𝑡 and with sufficient data in the lookback window. Leakage is explicitly avoided: feature windows end before 𝑡, and forward labels are the next period compounded arithmetic returns from 𝑡+1 up to the trading day before the next rebalance. The end-to-end construction (feature windows, robust correlations, and the chosen k-NN/MST/TMFG filter) is implemented in `make\_graphs.py`, which writes per-snapshot graph objects (node features, edge\_index, and edge\_attr) for downstream GAT training.

# Chapter 4: Proposed Modelling and Evaluation

In line with the business objective, which is to optimise portfolio allocation on a dynamic US mid-cap universe, we design the modelling and evaluation framework to compare three advanced allocation approaches (HRP, LSTM, GAT) under realistic constraints. Our modelling phase is designed to improve risk-adjusted return – Sharpe, while staying practical for live deployment.

We will benchmark our proposed models against simple baselines that a real investment desk might use to validate that the new methods add value, all as seen below:

1. Equal-Weight (EW). Ultra-robust and capacity-friendly; diversifies by c`onstruction but ignores information.
2. Minimum-/Mean-Variance (MV). Brings risk estimates into the decision yet is estimation-sensitive and can produce unstable weights.
3. Hierarchical Risk Parity (HRP). A clustering-aware heuristic that respects cross-asset structure (clusters, peripheries) typically, steadier than MV, but not objective-aligned.
4. LSTM (sequence model with allocation rule). Learns temporal signals from return histories, predictions drive a long-only, top-k tilt with turnover controls.
5. Graph Attention Network (GAT). Relation-aware and objective-aligned: ingests time-varying correlation graphs and outputs valid portfolio weights, trained end-to-end on Sharpe with embedded constraints.

This progression lets us test incremental value, moving from EW to MV introducing explicit risk, to HRP to introduce structure, to LSTM for temporal learning, and ultimately to GAT which is relation-aware for Sharpe optimisation. With the universe, constraints and cost model held constant, we isolate how each added capability translates into deployable, higher-Sharpe portfolios.

## 4.1 EW and MV (Baselines)

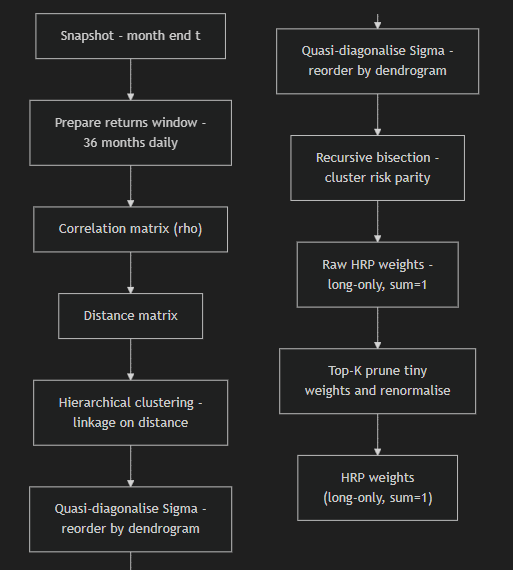
A diagram of a diagram

AI-generated content may be incorrect.

***Fig 11: Proposed EW and MV Pipeline***

The above diagram – *Fig 11* shows our proposed baseline model’s pipeline. From the month-end snapshot we apply liquidity hygiene to fix the eligible universe, then build a 36-month return window. The path splits: the equal-weight baseline selects the top-k liquid names and assigns 1/K to each, producing a long-only, fully invested portfolio; the mean- or minimum-variance branch estimates a covariance matrix with Ledoit–Wolf shrinkage and a simple return vector, solves a capped long-only optimisation, then removes tiny allocations to respect the top-k limit and renormalises. These steps keep the baselines stable, capacity-aware, and directly comparable within the shared harness.

## 4.2 HRP



***Fig 12: Proposed HRP Pipeline***

The above diagram – *Fig 12* shows our proposed HRP pipeline. Starting at month end, correlations are computed over a 36-month window and transformed into distances. Assets are grouped by hierarchical clustering, and the covariance is reordered by the resulting dendrogram. Allocation proceeds top down through recursive bisection, so sibling clusters carry balanced risk. The output is a long-only weight vector; small positions are set to zero to meet the top-k limit and weights are renormalised. The pipeline preserves the market’s correlation structure while avoiding unstable matrix inversion and yields deployable weights for the backtest.

## 4.3 LSTM

A diagram of a program

AI-generated content may be incorrect.

***Fig 13: Proposed LSTM Pipeline***

The above diagram – *Fig 13* shows our proposed LSTM pipeline. From each month-end snapshot we build per-asset return sequences with a fixed lookback, scale features, and train the network on the rolling training window with no look-ahead. The checkpoint with the highest validation Sharpe is retained. At each rebalance in the test year the model predicts next-period returns, assets are ranked, the top k is selected, and signals are mapped to long-only, fully invested weights using a softmax or proportional rule. Trading bands ensure small forecast changes do not churn positions, keeping turnover and costs contained while letting temporal patterns drive the allocation under the common constraints.

## 4.4 Graph Attention Network (GAT)

A screenshot of a computer flowchart

AI-generated content may be incorrect.

***Fig 14: Proposed GAT Pipeline***

The above diagram – *Fig 14* shows our proposed model GAT model pipeline. It begins with the graph snapshots, built based on the data creation mentioned in previous sections. The GAT ingests these graphs, learns which neighbours matter under current market structure, and produces long only, fully invested weights via a simplex projection head.

Two deployment-minded choices are built in. First, we train with a Sharpe-ratio loss over realised portfolio returns, so the model optimises the KPI directly. Second, we enforce top-k sparsity at inference: tiny positions are zeroed, and the remainder renormalised. This keeps transaction costs and operational complexity in check without materially degrading performance in our experiments.

## 4.5 Evaluation Pipeline and Metrics

The evaluation phase would be strongly tied to backtesting results. The following pipeline in *Fig 15* depicts the evaluation pipeline:

A screenshot of a computer

AI-generated content may be incorrect.

***Fig 15: Proposed Evaluation Pipeline***

We run a rolling, out-of-time protocol that mirrors live use: 36 months train / 12 months validation / 12 months test, stepping forward by a year. Each roll rebuilds features and graphs using only information available up to each rebalance, trains the model, selects the checkpoint with the best validation Sharpe, and then deploys monthly through the test year (trade at 𝑡+1, hold to next month-end). All comparators use the same universe dynamics, same rebalancing calendar, and the same transaction-cost schedule applied to realised turnover.

The metrics used are as follows:

Primary metric: Annualised Sharpe (net of costs)

Let be the daily portfolio return after trading cost (defined below), for . With risk-free set to 0.

Secondary metrics:

CAGR

With equity curve and trading days.

Annualised Volatility

Max Drawdown (MDD)

Let the running peak be .

With the business problem as well as evaluation metrics in place, the decision rule for business adoption would be to adopt the best candidate among HRP, LSTM, GAT only if, under the unified backtest, it beats both EW and MV on net Sharpe by a clear, statistically supported margin across rolls, keeps drawdown and volatility no worse, stays within turnover or cost limits with robustness to sensible cost/hyperparameter tweaks, and respects long-only, top-k, caps. If several qualify, pick the best trade-off of Sharpe, drawdown, turnover and simplicity. If none qualify, deploy the strongest benchmark.

# Chapter 5: Summary

In summary, this proposal sets out a practical, cost-aware path to improving portfolio allocation in a dynamic US mid-cap universe. We will compare three candidate models—HRP, LSTM, and GAT—against simple EW/MV benchmarks, all under the same long-only, top-k, turnover-controlled constraints and a rolling 36/12/12 backtest. Data are built to avoid survivorship bias; features and graphs are rebuilt strictly with information available at each rebalance. Performance will be judged primarily on net Sharpe, with CAGR, volatility, drawdown and turnover as guardrails. Adoption is model-agnostic: whichever candidate delivers a consistent, statistically supported uplift over both benchmarks—without worse downside or excessive trading—advances to a limited pilot with clear guardrails and monitoring; if none qualifies, we proceed with the strongest benchmark and document follow-ups. Deliverables include the report, reproducible code and data pipeline, figures (including model pipelines and evaluation), and a brief business recommendation ready for stakeholder review.

# Chapter 6: Proposed Schedule

A screenshot of a computer

AI-generated content may be incorrect.

# References

*An Introduction to Stooq Pricing Data | QuantStart*. (n.d.). Retrieved September 2, 2025, from https://www.quantstart.com/articles/an-introduction-to-stooq-pricing-data/

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654–669. https://doi.org/10.1016/j.ejor.2017.11.054

Guidolin, M., & Ria, F. (2010). *Regime Shifts in Mean-Variance Efficient Frontiers: Some International Evidence* (SSRN Scholarly Paper 1698153). Social Science Research Network. https://doi.org/10.2139/ssrn.1698153

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Korangi, K., Mues, C., & Bravo, C. (2022). *Large-scale Time-Varying Portfolio Optimisation using Graph Attention Networks*. Ar5iv. https://ar5iv.labs.arxiv.org/html/2407.15532

List of S&P 400 companies. (2025). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=List\_of\_S%26P\_400\_companies&oldid=1308022568

Lopez de Prado, M. (2016). *Building Diversified Portfolios that Outperform Out-of-Sample* (SSRN Scholarly Paper 2708678). Social Science Research Network. https://doi.org/10.2139/ssrn.2708678

Mantegna, R. N. (1999). Hierarchical structure in financial markets. *The European Physical Journal B - Condensed Matter and Complex Systems*, *11*(1), 193–197. https://doi.org/10.1007/s100510050929

Markowitz, H. (1952). Portfolio Selection. *The Journal of Finance*, *7*(1), 77–91. https://doi.org/10.2307/2975974

Massara, G. P., Di Matteo, T., & Aste, T. (2017). Network Filtering for Big Data: Triangulated Maximally Filtered Graph. *Journal of Complex Networks*, *5*(2), 161–178. https://doi.org/10.1093/comnet/cnw015

Merritt, R. (2022, October 24). What Are Graph Neural Networks? *NVIDIA Blog*. https://blogs.nvidia.com/blog/what-are-graph-neural-networks/

Pozzi, F., Di Matteo, T., & Aste, T. (2013). Spread of risk across financial markets: Better to invest in the peripheries. *Scientific Reports*, *3*(1), 1665. https://doi.org/10.1038/srep01665

Siegel, A., & Woodgate, A. (2007). Performance of Portfolios Optimized with Estimation Error. *Management Science*, *53*, 1005–1015. https://doi.org/10.1287/mnsc.1060.0664

Uddin, A., Tao, X., & Yu, D. (2023). Attention based dynamic graph neural network for asset pricing. *Global Finance Journal*, *58*, 100900. https://doi.org/10.1016/j.gfj.2023.100900

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). *Graph Attention Networks* (arXiv:1710.10903). arXiv. https://doi.org/10.48550/arXiv.1710.10903

# Appendix A: Building Membership from Wikipedia

A screenshot of a computer program

AI-generated content may be incorrect.

***Code 3: Membership Building Function***

A screen shot of a computer code

AI-generated content may be incorrect.

***Code 4: Membership Building Main Kwargs and Orchestration***

# Appendix B: Downloading Stooq Ticker Data

A computer screen shot of a program code

AI-generated content may be incorrect.

***Code 5: Stooq Data Fetching Function from API***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 6: Membership Loading Function***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 7: Main Orchestration Function for Download***

A screen shot of a computer code

AI-generated content may be incorrect.

***Code 8: Stooq Downloading Main Kwargs and Orchestration***

# Appendix C: Downloading and Augmenting Stooq Data with Yahoo Finance Data

A computer screen shot of a program code

AI-generated content may be incorrect.

***Code 9: Stooq Loading Function***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 10: Yahoo Finance Data Fetching Function from yfinance API***

A computer screen shot of a program code

AI-generated content may be incorrect.

***Code 11: Panel Merging Function***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 12: Yahoo Finance Data Downloading and Merging Main Kwargs and Orchestration***

# Appendix D: Data Exploration Notebook Code Blocks

A computer screen shot of text

AI-generated content may be incorrect.

***Code 13: Panel Summaries Printing***

A computer screen with text

AI-generated content may be incorrect.

***Code 14: Universe Size Through Time Chart Plotting***

A screen shot of a computer code

AI-generated content may be incorrect.

***Code 15: Availability Heatmap for Tickers Chart Plotting***

A computer screen shot of a program code

AI-generated content may be incorrect.

***Code 16: Membership Entries and Exits Per Year Charts Plotting***

A computer screen shot of text

AI-generated content may be incorrect.

***Code 17: Daily Volume Distribution Chart Plotting***

A computer screen with text on it

AI-generated content may be incorrect.

***Code 18: Daily Coverage Share Chart Plotting***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 19: TMFG Corr Edges***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 20: KNN Corr Edges***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 21: MST Corr Edges***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 22: Plotting Sample Graphs (1)***

A screen shot of a computer program

AI-generated content may be incorrect.

***Code 23: Plotting Sample Graphs (2)***