Incorporating Knowledge Graphs for Return Prediction

1. Problem Statement

Let $r_t \in \mathbb{R}^N$ denote the vector of asset returns at time t (with N assets) and $m_t \in \mathbb{R}^M$ denote a vector of macroeconomic indicators at time t (with M indicators). Our goal is to predict the asset returns at the next time step, r_{t+1}, by incorporating both historical asset information and external views from macro indicators. We allow the portfolio manager (PM) to tweak the asset–macro interrelationships, which we encode in a heterogeneous knowledge graph.

2. Model Setup

2.1 Heterogeneous Graph Construction

We construct a block adjacency matrix $\mathcal{A} \in \mathbb{R}^{(N+M)\times(N+M)}$ that integrates three types of relationships:

$$\mathcal{A} = \begin{pmatrix} A_{aa} & B_{am} \\ B_{ma} & A_{mm} \end{pmatrix}$$

where:

- **$A_{aa} \in \mathbb{R}^{N\times N}$:** Asset–asset similarity matrix (e.g. historical correlation).
- **$A_{mm} \in \mathbb{R}^{M\times M}$:** Macro–macro similarity (or self-relations among macro indicators).
- **$B_{am} \in \mathbb{R}^{N\times M}$ and $B_{ma} \in \mathbb{R}^{M\times N}$:** Asset–macro linkages that encode how sensitive an asset is to a given macro factor.
  Typically, the PM's views are encoded by adjusting the entries in B_{am} (with $B_{ma} = B_{am}^\top$ if we assume symmetry).

A **baseline** with no external views is obtained by setting B_{am} (and hence B_{ma}) to the zero matrix.

2.2 Combined Signal Vector

We combine the asset returns and macro indicators into one vector:

$$x_t = \begin{pmatrix} r_t \\ m_t \end{pmatrix} \in \mathbb{R}^{N+M}$$

This vector represents the full state of the market at time t.

2.3 Graph Filtering via the Laplacian

First, compute the degree matrix $\mathcal{D}$ with diagonal entries

$$\mathcal{D}_{ii} = \sum_{j=1}^{N+M} \mathcal{A}_{ij}$$

Then, the combinatorial Laplacian is given by:

$$\mathcal{L} = \mathcal{D} - \mathcal{A}$$

Perform an eigen-decomposition of $\mathcal{L}$:

$$\mathcal{L} = \mathcal{U}\Lambda\mathcal{U}^\top$$

where $\mathcal{U} \in \mathbb{R}^{(N+M)\times(N+M)}$ is an orthonormal matrix of eigenvectors and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{N+M})$ contains the eigenvalues.

Apply the **Graph Fourier Transform (GFT)** to x_t:

$$\tilde{x}_t = \mathcal{U}^\top x_t$$

Next, define a spectral filter function $h(\lambda)$ (for example, an exponential low-pass filter)

$$h(\lambda) = \exp(-\gamma\,\lambda), \quad \gamma > 0$$

Then, the filtered spectral coefficients are:

$$\tilde{x}_t^{\text{filtered}}(i) = h(\lambda_i)\,\tilde{x}_t(i)$$

Finally, recover the filtered signal by inverting the transform:

$$x_t^{\text{filtered}} = \mathcal{U}\,\tilde{x}_t^{\text{filtered}}$$

Because x_t stacks both r_t and m_t, the influence of the macro indicators is now propagated into the filtered asset signals. In particular, let:

$$r_t^{\text{filtered}} = \left[x_t^{\text{filtered}}\right]_{1:N}$$

which denotes the first N entries corresponding to the assets.

3. Prediction Model

We propose a simple prediction model where the filtered asset returns drive the next timestep's returns. For example, a linear autoregressive model may be used:

$$r_{t+1} = \alpha + \beta\,r_t^{\text{filtered}} + \epsilon_t,$$

where $\alpha \in \mathbb{R}^N$, $\beta \in \mathbb{R}^{N\times N}$ are parameters and ϵ_t is an error term.

The key point is that r_t^{filtered} incorporates the effects of both asset–asset relationships and the PM's external views (via B_{am}). Thus, by adjusting B_{am}, the PM can express views such as "asset i is more sensitive to macro factor j" which in turn affects the filtering and the final prediction.

4. A Toy Example

Consider a toy example with: - $N = 2$ assets, with returns $r_{1,t}$ and $r_{2,t}$. - $M = 1$ macro indicator, m_t.

4.1 Block Adjacency Matrix

Let

$$A_{aa} = \begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}, \quad A_{mm} = \begin{pmatrix} a_{mm} \end{pmatrix}$$

and

$$B_{am} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \quad B_{ma} = \begin{pmatrix} b_1 & b_2 \end{pmatrix}$$

The full block adjacency matrix is

$$\mathcal{A} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{12} & a_{22} & b_2 \\ b_1 & b_2 & a_{mm} \end{pmatrix}$$

In the baseline (no external view), we set $b_1 = b_2 = 0$

4.2 Graph Laplacian and Filtering

Compute the degree matrix $\mathcal{D}$ with diagonal entries:

$$\mathcal{D}_{ii} = \sum_{j=1}^{3} \mathcal{A}_{ij}$$

Then, the Laplacian is:

$$\mathcal{L} = \mathcal{D} - \mathcal{A}$$

Perform eigen-decomposition:

$$\mathcal{L} = \mathcal{U}\Lambda\mathcal{U}^\top$$

Stack the combined signal at time t:

$$x_t = \begin{pmatrix} r_{1,t} \\ r_{2,t} \\ m_t \end{pmatrix}$$

Apply the GFT:

$$\tilde{x}_t = \mathcal{U}^\top x_t$$

then filter:

$$\tilde{x}_t^{\text{filtered}}(i) = \exp(-\gamma\, \lambda_i)\, \tilde{x}_t(i)$$

and invert:

$$x_t^{\text{filtered}} = \mathcal{U}\, \tilde{x}_t^{\text{filtered}}$$

Extract the asset part:

$$r_t^{\text{filtered}} = \begin{pmatrix} \left[x_t^{\text{filtered}}\right]_1 \\ \left[x_t^{\text{filtered}}\right]_2 \end{pmatrix}$$

4.3 Prediction

Set up the prediction model:

$$\hat{r}_{t+1} = \alpha + \beta\, r_t^{\text{filtered}}$$

and estimate the parameters α and β (e.g. via least squares) using historical data.

---

5. Summary

- **Data Input:**
  We combine asset returns r_t and macro indicators m_t into one vector x_t.
- **Knowledge Graph Construction:**
  A block adjacency matrix $\mathcal{A}$ is formed with asset–asset relationships (A_{aa}), macro–macro relationships (A_{mm}), and asset–macro interactions (B_{am}). The PM's views can be expressed by tweaking B_{am}.
- **Graph Filtering:**
  The Laplacian $\mathcal{L} = \mathcal{D} - \mathcal{A}$ is computed, and the combined signal x_t is filtered in the spectral domain. The filtered asset returns r_t^{filtered} are then used for prediction.
- **Prediction Model:**
  A linear model (or other appropriate regression) is set up to predict r_{t+1} as a function of the filtered returns.
- **Baseline Comparison:**
  Setting $B_{am} = 0$ creates a baseline model using only asset–asset relations, allowing you to test the improvement from incorporating external macro views.

This framework gives you a mathematically rigorous yet flexible way to integrate heterogeneous data (assets and macro indicators) via a knowledge graph for return prediction.