Software Requirements Specification (SRS)

Project: Cognitive Graph Portfolio Optimizer (CGPO)

Version: 1.0
Date: September 13, 2025

1. Introduction

1.1 Purpose

This document specifies the requirements for the Cognitive Graph Portfolio Optimizer (CGPO), a novel decision-support system for financial markets. The system aims to provide advanced risk forecasting and dynamic portfolio allocation recommendations. It achieves this by fusing firm-specific insights from multimodal data (text, audio, news) with a systemic, graph-based view of asset interdependencies. The intended audience includes quantitative researchers, portfolio managers, and data scientists.

1.2 Scope

The CGPO system will be a research platform designed to:

- Ingest and process diverse financial data sources: earnings call transcripts, audio recordings, financial news, and time-series market data.
- Generate firm-specific risk and sentiment signals using a multimodal Large Language Model (LLM).
- Model the entire asset ecosystem as a dynamic graph, where nodes are assets and edges represent their interdependencies.
- Utilize a Reinforcement Learning (RL) agent to suggest optimal portfolio rebalancing actions based on the dynamic graph's state.
- Provide a robust back testing framework and visualization dashboard to evaluate strategy performance against standard benchmarks.

Out of Scope:

- Automated, live trade execution. The system is a decision-support tool, not an automated trading platform.
- User account management and multi-tenancy.

- A commercial-grade, public-facing user interface.

1.3 Definitions, Acronyms, and Abbreviations

- **LLM:** Large Language Model
- **GNN:** Graph Neural Network
- **GL-STN:** Graph-Temporal Network
- **RL:** Reinforcement Learning
- **API:** Application Programming Interface
- **SRS:** Software Requirements Specification
- **CGPO:** Cognitive Graph Portfolio Optimizer (this project)
- **Node Feature:** A vector of data representing an individual asset (node) in the graph.

1.4 References

1. *RiskLabs: How to Use Large Language Models for Risk Management* (arXiv, April 2024)
2. *Graph-LSTM: A new model for learning representations of dynamic graphs* (Inspiration for GL-STN architecture)

2. Overall Description

2.1 Product Perspective

The CGPO is a self-contained research and analysis platform. It will operate independently, pulling data from external APIs and producing analytical outputs. It is not an extension of an existing trading suite, but a novel system designed to test the hypothesis that combining deep qualitative analysis (from the LLM) with systemic network analysis (from the GNN) can lead to superior investment strategies.

2.2 Product Functions

The system's core functions are:

1. **Data Ingestion:** Collect and preprocess data from various sources.
2. **Signal Generation:** Analyze multimodal data to extract rich features for each asset.

3. **Systemic Modeling:** Construct and analyze a dynamic graph of assets to understand systemic risk.
4. **Optimized Decisioning:** Recommend portfolio actions using an RL agent.
5. **Evaluation & Visualization:** Back test strategies and present results through an intuitive dashboard.

2.3 User Characteristics

- **Quantitative Analyst/Researcher:** The primary user, who will interact with model configurations, analyze raw outputs, and develop new hypotheses. Requires access to logs, model parameters, and detailed performance metrics.
- **Portfolio Manager:** The end-consumer of the analysis, who will use the dashboard to understand current portfolio risks and evaluate rebalancing suggestions. Requires high-level, interpretable visualizations.

2.4 Constraints

- **Data Availability:** The system's performance is contingent on access to high-quality, timely, and potentially costly financial data APIs.
- **Computational Resources:** Training the LLM, GNN, and RL models will require significant GPU capacity.
- **Regulatory:** The system provides analytical outputs and suggestions, not financial advice. All investment decisions remain the responsibility of the user.
- **Technology Stack:** The proposed stack includes Python, PyTorch/TensorFlow, Hugging Face Transformers, DGL/PyG (for GNNs), and a web framework (like Dash or Streamlit) for the dashboard.

2.5 Assumptions and Dependencies

- **Core Hypothesis 1:** Vocal cues (tone, hesitation) in earnings calls contain predictive financial information not fully captured in the text transcripts.
- **Core Hypothesis 2:** Modeling asset interdependencies as a dynamic graph provides a more accurate representation of systemic risk than traditional correlation matrices.
- **Dependency:** The system relies on the continued availability and consistent formatting of third-party data APIs.

3. Specific Requirements

3.1 Functional Requirements

FR-1: Data Ingestion Module

- **FR-1.1:** The system shall connect to specified APIs to fetch historical and real-time market data (price, volume) for a defined universe of assets.
- **FR-1.2:** The system shall ingest earnings conference call transcripts (text) and audio recordings (e.g., MP3, WAV).
- **FR-1.3:** The system shall connect to news APIs to retrieve relevant articles for the asset universe.
- **FR-1.4:** All ingested data must be cleaned, preprocessed, and stored in a structured database or data lake.

FR-2: FinRisk-LLM Signal Generation Module

- **FR-2.1:** The system shall use an LLM to analyze text from transcripts and news to extract sentiment, identify key topics (e.g., supply chain risk, competition), and quantify risk factors.
- **FR-2.2:** The system shall employ audio processing models to analyze vocal features from earnings calls, extracting metrics for emotional tone, sentiment, and speaker hesitation.
- **FR-2.3:** The system must fuse the outputs from text, audio, and market data into a unified feature vector (e.g., predicted volatility, credit risk score) for each asset at each time step.

FR-3: GL-STN Systemic Modeling Module

- **FR-3.1:** The system shall construct a graph where assets are nodes. The node features will be the feature vectors generated by the FinRisk-LLM module.
- **FR-3.2:** Edge weights between nodes shall be dynamically computed based on a combination of market correlation and LLM-derived relationships (e.g., supply chain links identified in news).
- **FR-3.3:** The GL-STN model shall process sequences of these dynamic graphs to forecast future graph states and identify propagating risk clusters.

FR-4: RL Optimization Module

- **FR-4.1:** The system shall define an RL environment where the "state" is the output of the GL-STN model.
- **FR-4.2:** The "action space" shall be defined as a set of possible portfolio weight adjustments (e.g., buy, sell, hold; or percentage allocation changes).
- **FR-4.3:** The "reward function" shall be configurable, based on metrics like risk-adjusted return (e.g., Sharpe Ratio) or portfolio drawdown.
- **FR-4.4:** The system will train an RL agent (e.g., PPO, A2C) to learn an optimal policy for rebalancing the portfolio.

FR-5: Backtesting and Visualization Module

- **FR-5.1:** The system shall include a backtesting engine to simulate the RL agent's learned policy on historical data.
- **FR-5.2:** The system must calculate and display key performance indicators (KPIs), including total return, volatility, Sharpe ratio, and maximum drawdown.
- **FR-5.3:** The system's performance must be benchmarked against at least two standard strategies (e.g., a market-cap-weighted index like the S&P 500 and an equal-weight portfolio).
- **FR-5.4:** A web-based dashboard shall visualize the portfolio's performance, current asset allocations, systemic risk map (from the GNN), and the key signals from the LLM for individual assets.

3.2 Non-Functional Requirements

- **NFR-1: Performance:** Model inference for a new data point (e.g., a single earnings call) should be completed in under 5 minutes to be considered near real-time. Full back testing runs on several years of data should complete within 24 hours.
- **NFR-2: Scalability:** The architecture must support scaling the asset universe from an initial 50 stocks to at least 500 stocks (e.g., the S&P 500) without requiring a complete redesign.
- **NFR-3: Modularity:** Each module (Ingestion, LLM, GNN, RL, Visualization) shall be developed with clear interfaces to allow for independent testing, upgrading, and replacement.
- **NFR-4: Reliability:** Data ingestion pipelines must be fault-tolerant, with logging and retry mechanisms.
- **NFR-5: Usability:** The visualization dashboard must be intuitive and interpretable by users who are not experts in machine learning.