Golden Retriever

An AI Rail Network Brain for Golden-Run Conflict Resolution

Project Title & Overview

Golden Retriever is an AI-powered decision-support platform designed to assist rail network operators in resolving both current and emerging traffic conflicts. The system leverages operational memory and similarity search to retrieve historically successful disruption resolutions—referred to as *golden runs*—and adapts them to the current network context.

For each detected conflict, Golden Retriever retrieves similar past situations, extracts effective resolution strategies, and evaluates these strategies using a **digital twin simulation** of the rail network. The outcomes of the simulation are combined with historical performance metrics to produce a ranked, explainable list of recommended actions.

In addition to reactive conflict resolution, the platform includes a **predictive capability** that identifies conflicts before they materialize by detecting similarity to **historical pre-conflict patterns** and proactively suggesting preventive measures.

—

Problem Statement

Rail networks are complex, tightly coupled systems where minor disruptions—such as small delays, infrastructure constraints, or temporary capacity reductions—can quickly propagate and cause large-scale network instability.

Current operational approaches often rely on static rules, manual operator experience, or optimization techniques that recompute solutions without learning from past outcomes. These approaches make it difficult to:

- Rapidly identify the most effective resolution strategy,
- Anticipate secondary and cascading conflicts,
- Reuse institutional knowledge embedded in historical operations,
- Provide transparent, evidence-backed explanations for decisions.

As traffic density increases and networks become more constrained, there is a growing need for systems that can learn from historical experience and support operators with fast, explainable, and context-aware recommendations.

—

Use Case Being Solved

This project addresses **Use Case 1: AI Rail Network Brain – Real-Time Conflict Resolution via Multimodal Memory**.

Golden Retriever focuses on:

- Detecting current and predicted operational conflicts (track, platform, headway, capacity),
- Retrieving similar historical situations and their successful resolution strategies,
- Validating candidate strategies using a digital twin simulation of the current network state,
- Ranking and explaining resolution options based on both historical outcomes and simulated performance,
- Continuously learning from operational outcomes to improve future recommendations.

An additional predictive feature extends the use case by identifying early signals of future conflicts through similarity search over historical pre-conflict states and recommending preventive actions before disruptions escalate.

—

How Qdrant Is Used

Qdrant serves as the core vector database and operational memory of the Golden Retriever platform.

Conflict Memory

Each detected conflict is represented as a vector embedding that captures the operational context, including:

- Conflict type and severity,
- Network topology and location,

- Traffic density and time-of-day,
- Active infrastructure constraints,
- Delay distribution and system state.

These embeddings are stored in Qdrant along with structured payloads describing the resolution actions taken and their measured outcomes, such as delay reduction and recovery time. Similarity search over this collection enables retrieval of historically successful *golden runs* relevant to the current situation.

Pre-Conflict Memory

A second Qdrant collection stores embeddings of short operational windows preceding historical conflicts. These vectors allow the system to:

- Detect emerging conflicts by similarity to known precursors,
- Anticipate potential disruptions before they occur,
- Learn which early interventions successfully prevented conflict formation.

Decision Support and Learning Loop

Retrieved historical cases are used to generate candidate resolution strategies. Each strategy is evaluated through digital twin simulation, and the resulting performance metrics are combined with historical outcomes to produce a final ranked recommendation.

After execution, actual outcomes are fed back into Qdrant as new memory entries, enabling continuous improvement of the system over time.

—