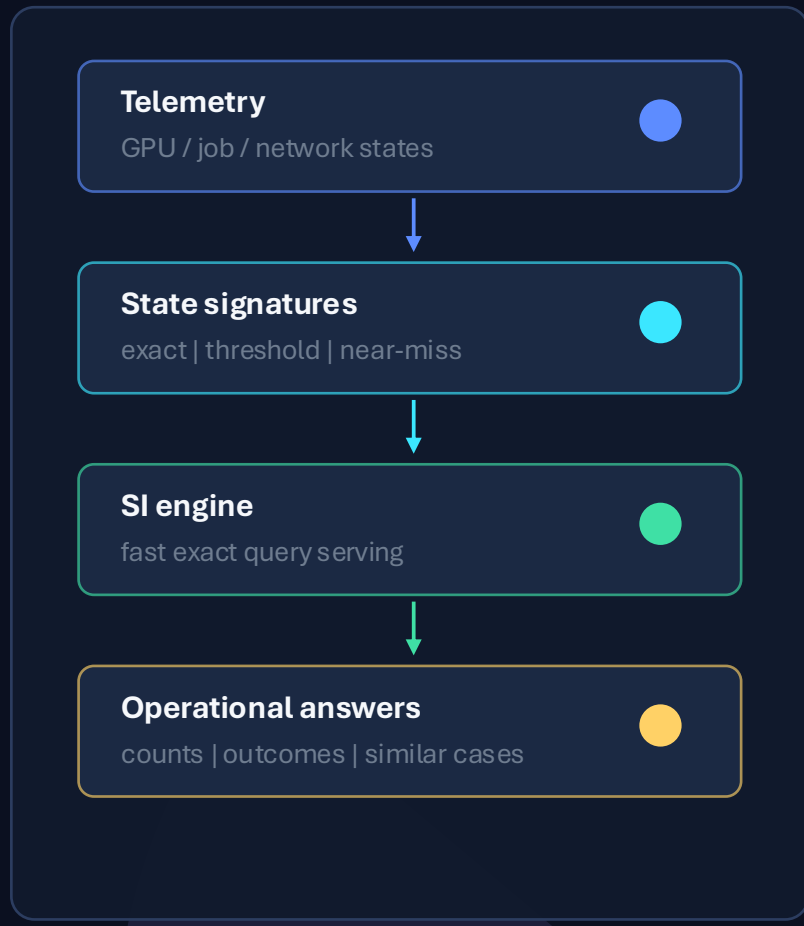
AI INFRASTRUCTURE | GPU CLUSTERS | TAIL-RISK MEMORY

Signature Index for AI Infrastructure Risk Memory

An exact, high-speed query layer for recurring telemetry-state signatures: tail bottlenecks, fail-slow states, near-miss incidents and multi-outcome historical counts.



WHY THIS MATTERS

The problem: AI clusters create tail-risk signatures

Synchronized workloads

Large training jobs can be slowed by one delayed transfer, one degraded path, or one fail-slow node.

Telemetry is rich but fragmented

GPU metrics, job metadata, network state, placement, storage and incident logs live in different layers.

Average metrics hide the real problem

p99/p100 step latency, stragglers and near-miss incidents matter more than normal-case averages.

Teams need historical memory

The key operational question is often: have we seen this state before, and what happened next?

SI is positioned above observability: not as a monitor, scheduler or network protocol, but as an exact historical query layer for recurring telemetry-state signatures.

POSITIONING

What SI does in AI infrastructure

Signature Index turns GPU-cluster telemetry history into a fast, exact, queryable memory of recurring states and incident outcomes.

Exact incident signatures

Have we seen this exact combination of GPU, job, network and I/O states before?

Near-miss incident neighborhoods

Which historical states were almost the same, and were they risky?

Broad-to-specific diagnostics

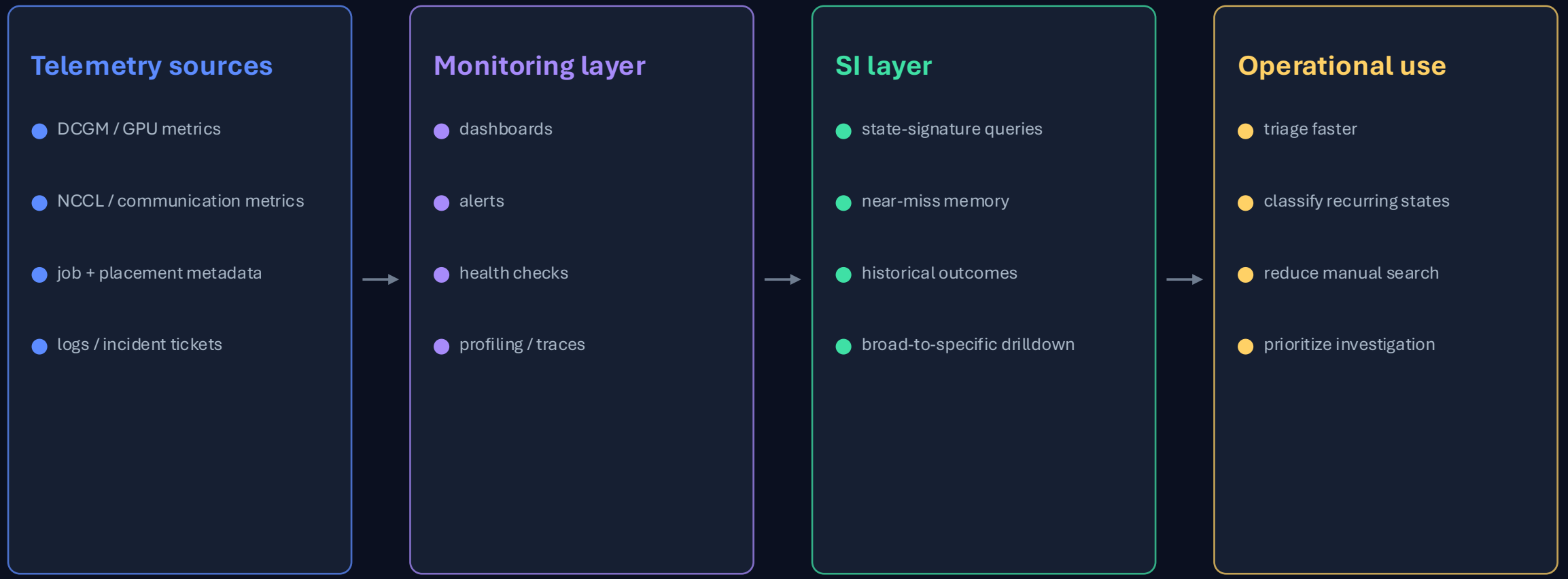
Inside a broad alert, which precise sub-states mattered most?

Multi-outcome counts

For a given state, count tail spikes, retries, failures, rollbacks or incident tickets.

NOT A REPLACEMENT FOR TELEMETRY

Where SI sits in the stack



SI complements GPU telemetry and observability. It does not replace DCGM, Prometheus, profiling, scheduling or routing.

OPERATOR-FACING QUERIES

Example questions SI is built to answer

Seen before?

Have we seen this exact GPU/job/network/I/O state before?

What happened next?

How often did it lead to tail spikes, retries, rollbacks or failures?

Near-miss risk?

Are similar states also dangerous, or is this exact state special?

Why this alert?

Which precise sub-states inside a broad alert class carry the real operational risk?

Where is it concentrated?

Does the signature concentrate in a rack, job shape, GPU type, placement pattern or time segment?

EVIDENCE

Public-trace benchmark: Alibaba GPU Cluster Trace

Benchmark setup

3,033,232
telemetry-state rows

300
hidden queries

0
SI mismatches vs reference

DuckDB + Polars
public baselines present

Public data source

Alibaba GPU Cluster Trace v2020: public production AI/ML workload trace from Alibaba PAI, covering GPU-cluster job and telemetry data.

Benchmark task

Answer hidden finite-state telemetry-signature queries exactly: exact, threshold, near-miss, broad-to-specific and segment-conditioned counts.

The test is a query-engine benchmark, not a prediction or root-cause benchmark.

BENCHMARK RESULTS

Exact correctness first

0

SI mismatches vs reference across 300 hidden queries

The engine returns exact counts. It is not a probabilistic model and does not approximate the answer.

SI vs reference

0 / 300 mismatches

DuckDB baseline

0 / 300 mismatches

Polars baseline

0 / 300 mismatches

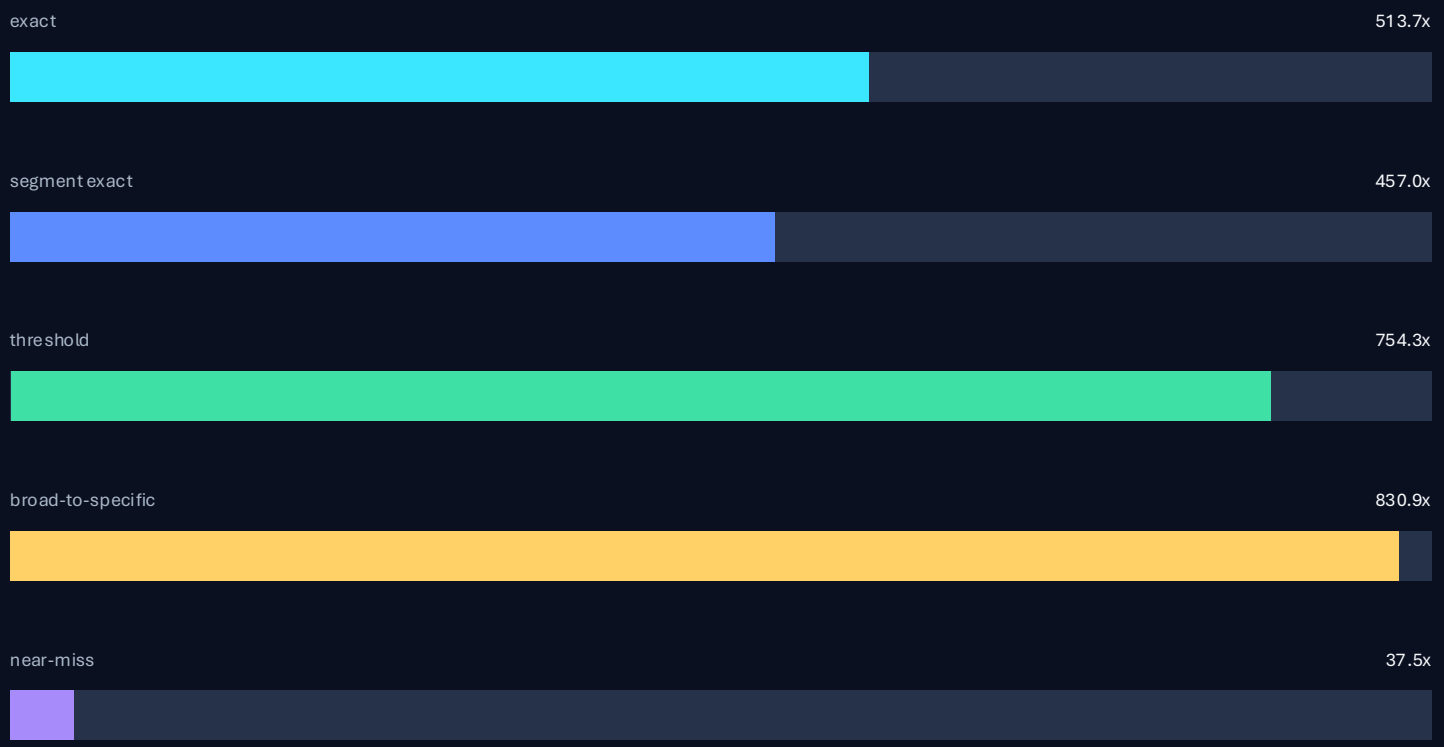
Baseline errors

none

BENCHMARK RESULTS

Performance: speedup vs fastest public baseline

Speedup by query family



513.7x

median speedup vs fastest public baseline

37.5x

minimum speedup across query families

Public baselines: DuckDB and Polars. Fastest public baseline is selected per query family.

MEMORY STORY

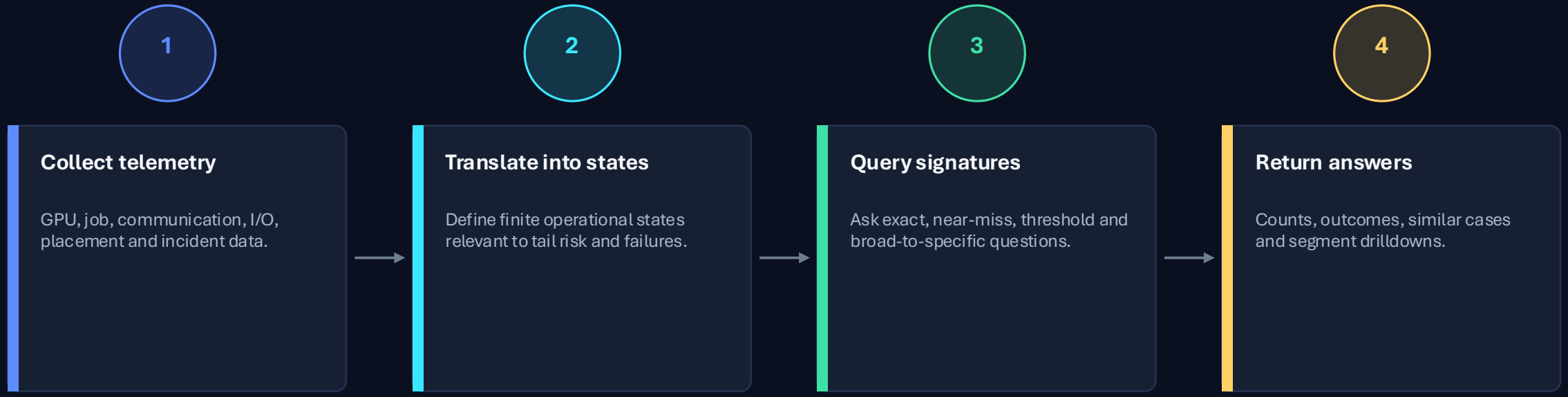
Resource frontier: where flat materialization hits a wall

Flat materialization can work for simple exact equality, but rich catalogs quickly exceed practical memory budgets.

Query family	Flat materialized catalog	Fits 24 GB?
exact	1.77 GB	YES
segment exact	1.77 GB	YES
near-miss	**58.26 GB**	NO
threshold / roll-up	**2,694.27 GB**	NO
broad-to-specific	**2,694.27 GB**	NO

At 3.03M public-trace rows and a 5,000-query catalog.

How SI changes the operational workflow



The result: telemetry becomes an exact, queryable memory of recurring operational states - not just a collection of dashboards.

USE CASES

Where SI can create value

Tail-bottleneck memory

Retrieve historical states similar to a p99/p100 slowdown and count outcomes.

Fail-slow triage

Classify states that look like previous fail-slow / straggler situations.

Near-miss incident search

Find almost-similar incidents even when no exact match exists.

Broad alert refinement

Break broad alerts into specific sub-states with different historical outcomes.

Segment drilldown

Ask whether signatures concentrate in job shapes, racks, clusters or hardware groups.

Multi-outcome history

Count tail spike, retry, rollback, failure and incident-ticket outcomes together.

BOUNDARY

What SI is not

SI is intentionally positioned as a specialized query layer, not as a replacement for existing infrastructure tools.

Not a GPU monitor

It can sit above DCGM/Prometheus-style telemetry, not replace it.

Not a scheduler

It does not allocate jobs or optimize cluster placement.

Not a network protocol

It does not route packets or replace congestion/failure mitigation mechanisms.

Not a predictor by itself

It can power predictive or diagnostic systems by serving exact state/outcome queries.

NEXT STEP

Evaluation path

A practical evaluation should focus on exact state-signature queries over historical telemetry, using the partner's own data and validation criteria.

Case pack

Telemetry windows, states, segments, outcomes and query families agreed upfront.

Black-box run

SI can be evaluated as a controlled engine without exposing its internal implementation.

Validation

Correctness checked by reference counts, hashes and agreed output schema.

Metrics

Latency, throughput, memory, exactness and rich-query capacity under budget.

Goal: determine whether SI adds a high-speed state-signature memory layer to an existing observability or AI infrastructure platform.

SUMMARY

Bottom line

SI makes recurring telemetry-state questions directly answerable.

For AI infrastructure teams, the opportunity is to move from metric-by-metric investigation to exact historical memory of complex operational states: what matched, what nearly matched, and what happened next.

Public-trace result: 0 mismatches, DuckDB/Polars baselines present, minimum speedup 37.5x, median speedup 513.7x across hidden query families.