What to learn before you train

Kenneth Pernyér, Reflective Labs
Stockholm, Sweden
kenneth.pernyer@reflective.se · reflective.se · ORCID 0009-0006-4543-1156

This paper reports no trained model. It reports a formulation: what an organizational trajectory model should predict, what data it needs, which of several machine-learning approaches — a Joint-Embedding Predictive Architecture among them — actually fits the shape of that data, and how such a model would have to be made operational inside the substrate described in my first report. I argue the target is not a model of people. It's a model of coordination — the unit of analysis is an initiative, not an employee — and I give the contrast that makes that distinction operational rather than aspirational. I connect the design to an argument I made earlier, outside this series, about software becoming an intent codec: a system that ships less because its decoder understands more. A predictive model of organizational state is the same move, applied to coordination instead of video. I lay out a four-stage build plan in which JEPA is stage three, not stage one, because the earlier stages are the same-information comparison a JEPA claim needs to survive before it's worth believing. Four independent adversarial reviews of the first hypothesis draft each found a distinct, real problem — a possible label-timing circularity, a feedback loop specific to models that inform the reviewers who generate their own training labels, a missing information-parity control, and a training-data population that, on inspection, probably doesn't exist yet in usable volume. None of these findings are fatal. All of them are reasons to publish the formulation and postpone the training, which is what this paper does.

1. Introduction

Software used to ship everything it needed the receiver to have. A JPEG carries every pixel. Early video codecs carried every frame. The receiver did no work beyond decompression, because the receiver was assumed to understand nothing.

That assumption stopped being true, and the encoders got smaller because of it. H.264 ships keyframes and motion vectors and lets the decoder reconstruct the frames in between, because the decoder now has a motion model. NVIDIA's Maxine ships facial keypoints — a few dozen numbers — and a generative model on the receiving end reconstructs a face, because the decoder now understands faces well enough to regenerate one from a sparse description. I made the general version of this argument in an earlier essay: the smarter the decoder's model of the world, the less the encoder has to ship, and **"the last piece of software is an intent codec: it minimizes what humans must specify and maximizes what systems can safely reconstruct"** (Pernyér, 2025a). That argument was about product software — interfaces, specifications, the gap between what a person says and what a system does. This paper asks whether the same move applies to something else entirely: whether an organization's coordination state can be treated as the thing a decoder reconstructs, rather than the thing every stakeholder has to fully specify by hand.

A Joint-Embedding Predictive Architecture is a concrete, machine-learned instance of exactly that move. It doesn't reconstruct raw input — the missing week of project history, the exact sequence of messages — the way an autoencoder or a language model asked to fill in a blank would. It predicts a **representation** of what's missing, in a space shaped by what recurs across many examples (Assran et al., 2023, Bardes et al., 2024). That is the encoder/decoder relationship from video codecs, restated as a training objective. Whether it's the right tool for organizational data specifically is a question this paper takes seriously and does not resolve — because resolving it requires data this organization does not yet have in usable volume, a claim I defend with evidence rather than assert.

This paper is the formulation that precedes that resolution: what to collect, what to train, what to learn, and how to make it operational, once the prerequisites exist. §2 argues for the right target. §3 develops the intent-codec connection properly. §4 states what needs to be collected. §5 lays out a staged build plan in which self-supervised representation learning is the third stage, not the first, and says why. §6 works out the type discipline a learned prediction needs once it enters the substrate from my earlier report (Pernyér, 2025b), and gives one worked example — an unrecorded rank override — of the kind of coordination failure this is for. §7 states

plainly why training was postponed, with the four specific problems that caused it.

2. The wrong target is a model of people

Ask what a predictive model of an organization should learn, and the easy answer is: whether a given person is performing well, disengaged, at risk of leaving. That answer is easy because performance-management data is the data most readily at hand, and it's wrong for this program, for a reason sharper than privacy.

The target I want is a model of **coordination**, not behavior. Call it an Initiative Trajectory Model. Its unit of analysis is a project, a Formation, a decision process, a cross-team initiative — never an employee. The question it answers:

> **Framing question.** Given everything legitimately known about an initiative today, what coordination state is it in, what is likely to become constrained next, and what evidence supports that view?

$$\textit{events} + \textit{decisions} + \textit{dependencies} + \textit{work flow} + \textit{time}$$
$$\downarrow$$
$$\textit{coordination state}$$
$$\downarrow$$
$$\textit{trajectory, uncertainty, precedents}$$

Useful outputs at this grain: the likelihood a milestone needs revision in the next thirty days; the likelihood a dependency stays unresolved past its required date; **decision debt** — decisions needed, their age, their blocking impact, a named and trackable quantity rather than a vibe; whether an initiative is converging, drifting, or stalled; which historical initiatives had a genuinely similar trajectory; and, usefully, what additional evidence would most reduce the model's own uncertainty — a question a model can answer honestly without answering the harder question first.

Organizations don't usually fail because one person underperformed. They fail through a structural pattern:

$$\textit{ambiguous ownership} + \textit{dependency churn} + \textit{unresolved decisions} + \textit{scope movement} + \textit{slow feedback loops} + \textit{unrecorded rank override} = \textit{coordination failure}$$

I return to the last term in §6.3; it's the sharpest of the six and it connects directly back to the substrate. The pattern as a whole plausibly connects to coordination theory's treatment of coordination failure as structural rather than personnel-driven (Malone and Crowston, 1994) — I haven't read that paper closely enough yet to cite it as more than a lead, and I say so rather than dress up an unread reference as a foundation.

What makes "coordination, not behavior" an actual constraint, rather than a value statement I can violate the first time it's inconvenient, is a concrete test any output has to pass:

> **Test (Safe output).** Not: *"Alice is disengaged."* Instead: *"This initiative resembles prior stalled initiatives: three unresolved cross-team decisions are now older than the normal resolution range, and its main dependency has changed twice. Coverage is incomplete; inspect these source events."*

The second form names structural facts, cites its own evidence, and states its own uncertainty. The first form is a judgment about a person the model has no business making, however accurate it might turn out to be. Any output shaped like the first sentence is a defect in the system, not a difference of style — and it's a defect a model trained on organizational data can produce by accident if nothing in its design rules it out. Ruling it out is a design decision, not a consequence that follows automatically from choosing the right training objective.

3. Coordination as a codec problem

The intent-codec argument is worth developing here rather than gesturing at, because it's doing real work: it's the reason a **predictive** model, rather than a **complete** one, is the right shape for this problem at all.

A complete model of an initiative — every message, every meeting, every private judgment that shaped a decision — is not available, would not fit in anything tractable if it were, and would answer questions nobody with legitimate access is entitled to ask. That's the MJPEG position: ship everything, because the receiver understands nothing and must be given nothing to reconstruct.

The codec insight is that a receiver with a good enough model of the domain needs far less. H.264's decoder understands motion well enough that shipping motion vectors is sufficient; Maxine's decoder understands faces well enough that shipping keypoints is sufficient. The organizational analogue: a decoder that understands **coordination** well enough — what unresolved decisions typically do to a timeline, what dependency churn typically precedes — needs only the structural signal, not the full record, to reconstruct a useful estimate of where an initiative stands and where it's headed.

This reframes what "training a model" means here. It is not "teach a system to know what happened." It's "teach a decoder to reconstruct a compressed state from partial, legitimate signal" — precisely JEPA's training objective, applied outside the domain it was built for. Predicting a masked window's **representation**, rather than its raw content, is the machine-learning-native version of shipping keypoints instead of pixels. That parallel is suggestive, not dispositive

— video has spatial and temporal redundancy that organizational event streams may or may not share, a question I return to in §7 — but it is the actual reason JEPA belongs in this paper's staged plan and not simply "a technique I looked into."

The intent-codec essay's second claim matters just as much here as the first: **"the winners won't be whoever renders the prettiest app from a prompt. They'll be whoever owns the decoder for trust"** (Pernyér, 2025a). Applied to this program: the value isn't in collecting more organizational data. It's in building a decoder — a trained model plus the evidence and provenance machinery around it — that an organization actually trusts to reconstruct its own state honestly, including trusting it to say when it doesn't know.

4. What to collect

The instinct to design a bespoke "organizational event" schema should be resisted. My first report already guarantees one exists for anything that runs through the substrate: the Observability invariant requires that **"every effect is recorded in the ledger"** (Pernyér, 2025b), and every promoted fact carries its producing Suggestor's identity and version and the context it read. That is an organizational event stream — proposals, promotions, blocks, escalations, timestamped, per fact type — without inventing a new one. The right instinct is to consume the ledger that already exists rather than build a parallel one that will drift from it.

Two things must exist as governed facts before anything downstream is trainable, and neither currently does in a confirmed way.

Plans, as inspectable artifacts. Detecting drift from a plan requires the plan to be a fact the ledger can reference, revised through a recorded decision — not a document that lives in a deck or a meeting nobody transcribed. This is a genuine gap to name, not an assumption to build around: if strategic plans live only informally today, drift-from-plan detection is unbuildable until they don't, and that's a prerequisite worth stating before it's discovered the hard way.

Labels, with their timing pinned exactly. An outcome label — delayed, blocked, recovered, delivered — has to be computed strictly from facts that postdate whatever window a model is being evaluated against. Getting this wrong doesn't produce a slightly weaker result; it produces a number that looks like evidence and isn't, because the model was scored on information it was, in effect, allowed to see.

Data volume needs honest arithmetic, not a hopeful estimate. If a model consumes six to twelve weeks of history and is evaluated against the following two to four weeks, one fully evaluable example takes roughly eight to sixteen weeks after reliable logging begins to exist at all — and meaningfully longer again to accumulate enough **independent** examples to say anything general. That last qualifier matters more than it looks: one initiative sampled at thirty weekly windows is one trajectory, richly observed, not thirty proofs that anything generalizes. Windows are not cases, and treating them as interchangeable is the fastest way to manufacture false confidence out of a single long project.

Waiting for volume doesn't mean waiting idly. Six things are worth doing in parallel, each aimed at a specific way this program could otherwise go wrong later: build the canonical event schema and immutable history snapshots; census what data already exists, including through whatever path preceded the current one and at what fidelity; define labels and their exact timestamps, on paper, before there's real data to get the definition wrong on; run data-quality and coverage reports as the census returns; build a shadow-mode serving path that logs predictions without ever showing them to a human reviewer — the reason for this is developed in §6.2; and prototype against synthetic or substitute-path data strictly to validate plumbing, never to validate the hypothesis a real result would need to support.

5. What to train

The build plan is staged, and the order is the argument.

1. **First.** A calibrated risk/trajectory model over explicit ledger features — gradient-boosted or a small tabular classifier. Not JEPA yet.
2. **Then.** A temporal graph or sequence model reading the full event history: initiatives, teams, decisions, dependencies, timestamps, without the summary step the first stage takes.
3. **Later.** JEPA-style self-supervised pretraining, to produce a reusable organizational-state embedding — if the earlier stages show the data is rich enough to reward it.
4. **Much later.** An intervention model: given this condition, which action tended to shorten resolution time historically. Flagged explicitly as needing real causal evidence or controlled experiments, not correlational inference — this is the most valuable model in the sequence and the easiest one to get dangerously wrong, and it does not proceed without that evidence.

5.1. Why the order matters, not just the roster

A representation-learning claim needs a same-information comparison to mean anything. If a self-supervised sequence model beats a hand-featurized baseline, the result is uninterpretable unless both models saw comparable information — otherwise a win proves only that more raw information helps, which nobody doubted, rather than that representation-space prediction specifically pays off on this data. Stages one and two are not consolation prizes on the

way to the real model. They are the controlled comparison a JEPA claim in stage three would otherwise lack, and they are useful in their own right regardless of whether stage three ever ships.

This also mostly resolves a question that would otherwise stall the whole program: where does this live architecturally. Stage one — a calibrated classifier over explicit features — is close to a direct fit for the fitted-model extension in this stack, which already ships a tabular classifier trained by classical optimization and already carries the provenance discipline this needs (Pernyér, 2025c). Stage two and three need a training regime that extension's current interfaces don't shape-match: self-supervised, three networks including a non-gradient-trained EMA target, a representation-space loss rather than a labelled classification. That's a real architectural question — new capability in an existing extension, or a new one entirely — and it only has to be answered once the program reaches stage two. Nothing about stage one requires deciding it now.

5.2. Where JEPA sits in the wider landscape

JEPA descends from a broader argument that predictive world models, not generative ones, are the right foundation for machine intelligence (LeCun, 2022), and from the older observation that grounding symbols in something outside the symbol system is what makes them mean anything (Harnad, 1990). Its concrete lineage is I-JEPA for images (Assran et al., 2023), V-JEPA for video (Bardes et al., 2024), and V-JEPA 2, which extends the architecture to action-conditioned prediction and was demonstrated zero-shot on robotic manipulation (Assran et al., 2025) — evidence that the architecture generalizes past its original domain, though not evidence about **this** domain specifically.

Non-vision applications exist and are worth naming honestly rather than implying either "well-trodden" or "untried." T-JEPA applies the same predict-in-representation-space objective to tabular data by masking one feature subset and predicting another within the same row — directly relevant, since it solves the same "no natural augmentation" problem organizational event data shares with tables (Thimonier et al., 2024). Graph-JEPA extends the approach to masked-subgraph prediction over graph-structured data (Skenderi et al., 2023) — relevant if dependency and team structure end up modelled as a graph rather than a sequence. I searched specifically for JEPA applied to organizational, business-process, or enterprise-event data and found nothing, positive or negative — this looks like open territory rather than a line other researchers tried and abandoned, though my search was not exhaustive and industry-lab work that isn't indexed publicly could exist without my having found it.

6. Making a prediction mean something once it's in the loop

Collecting the right data and training the right model doesn't by itself say what a prediction is **allowed to do** once it reaches the substrate. Three things have to be settled, and I take a position on each.

6.1. Confidence is not one thing, and this model adds a category to keep straight

My first report already establishes that certificates, fuzzy memberships, and statistical probabilities are three distinct quantities that must never be averaged together. A learned trajectory prediction adds a genuine fourth question rather than fitting cleanly into one of the three: its parameters are learned, which rules out both certificate (requires a proof procedure) and membership (requires an authored function). It reduces to two honest cases that must be reported separately. Where no calibrated probe sits on top of it, the only honest signal is a bare distance or anomaly score — structurally identical to a z-score, a normalized distance and not a probability. Where a calibrated probe **is** trained on the frozen representation, its output is an ordinary statistical probability, subject to the ordinary discipline that category already carries: it must be calibrated, and it belongs to the fitted-model category regardless of what features it consumes.

JEPA	distance / anomaly score
calibrated probe	statistical probability, if validated
fuzzy logic	authored degree, for policy and explanation
ledger	factual evidence and provenance
precedent index	nearest comparable cases

Fuzzy logic deserves one clarification, because the temptation it exists to resist is specific to this kind of model. A latent vector is not naturally readable, and assigning meaning to one of its dimensions — "dimension seventeen means stakeholder alignment" — is invented interpretation, not discovered structure. Fuzzy logic's legitimate role is downstream and separate: expressing human-authored, inspectable concepts over **observable facts** — decision latency is high, dependency exposure is medium — for policy, simulation, and explanation. It never explains what an embedding dimension means, and its output must never be relabeled as the model's own confidence. The right way to make a prediction inspectable isn't to decode the embedding at all. It's to surface evidence around it: retrieve comparable historical initiatives from an index pinned to the same model version; show the actual observed events behind a score rather than the score alone; ask what would change the

score under a counterfactual — if this unresolved decision cleared, how much would the estimate move; and where a specific named property matters enough to justify it, train a narrow, validated probe for that property specifically, rather than reading tea leaves from raw dimensions.

6.2. The feedback loop this kind of model creates, and why serving has to start silent

Every fitted model in this stack so far has had an evaluation that stays honest indefinitely, because the thing being predicted is exogenous to the prediction — a loan default doesn't care what a classifier said about it. A trajectory model placed as evidence in front of a human reviewer does not have that property. A reviewer shown **"this resembles a project that was later blocked"** is more likely to escalate or block the current one. Once that happens routinely, the model's own past outputs are shaping the human decisions that generate the very labels used to retrain it — which means a "temporally held out" evaluation quietly stops being independent of the thing it's evaluating, without anyone having to do anything wrong.

The rule that guards against this already exists in my second report — learning flows backward into priors and never into authority — and this is the case where obeying the rule's letter while missing its spirit is easy to do by accident: a model that never touches the promotion gate directly can still reach into a decision through the humans who read its output. The fix I take is structural rather than statistical: any pilot serves in shadow mode first — predictions logged, never shown to a reviewer — until there's a clean, uncontaminated evaluation on record. Trying to control for the contamination after the fact, by freezing model versions and hoping a temporal split holds, is a weaker guarantee than simply not creating the loop until the model has earned the right to be seen.

6.3. A worked example: the unrecorded rank override

One coordination-failure pattern is worth working through concretely, because it connects this program back to the substrate's own philosophy rather than sitting beside it.

Plans get overridden informally. Someone with rank says something in a meeting, and the organization follows it, even though a documented strategic plan said otherwise and was never formally revised. This has a name in analytics culture — **HiPPO**, the highest-paid person's opinion — and a looser academic cousin in the strategy literature's treatment of incremental, unreconciled drift from a stated position. Neither is load-bearing here; the mechanism is what matters.

My first report states the relevant principle already, about a different channel: **"a system that resolves contradictions silently has chosen, on the organization's behalf, which of two conflicting claims about it is true"** (Pernyér, 2025b). A committed plan and a senior redirection are two conflicting claims about what the organization is doing. When the org follows the redirection without anyone formally superseding the plan, that contradiction has been resolved — silently, by rank, rather than by escalation. The substrate refuses this move for anything code-mediated. Nothing currently refuses it when the mediation is a conversation in a room.

Detecting it needs no learned model at all — it's a structural query, closer to policy-consistency checking than to trajectory modelling: given a committed plan and the decisions that followed it, is there a recorded revision event in between, or does the trajectory simply diverge with nothing accounting for it. Output in the same register as the safe-output test in §2:

> "This initiative's last three decisions diverge from the plan committed on [date]. No formal revision is recorded in between. Inspect these decisions for their actual authorization."

I place this example here rather than in §5′s staged plan because it belongs to stage one, not stage three — it needs plans-as-facts, established in §4, and nothing else. It's also the clearest illustration in this paper of what "operational" is supposed to mean: not a prediction about a person, a structural gap between what was decided and what happened, with a pointer back to the evidence.

7. Why the research is postponed

Four independent adversarial reviews of the first hypothesis draft for this program each returned a distinct, substantive objection, not a shared one — a sign the review process worked rather than that the idea failed.

Data volume. The ledger path capable of producing genuine, complete, end-to-end initiative histories is itself new — new enough that the population of real, trainable examples right now is close to zero, not merely "not yet large." §4′s arithmetic makes this precise rather than hand-waved.

A possible label-timing circularity. Nothing in the first draft pinned down whether an outcome label could lean on facts inside or immediately after the window a model is scored on predicting. Unresolved, this doesn't produce a weaker result — it produces a number shaped like evidence that isn't.

A missing information-parity control. The first draft let a JEPA-style model read the full event stream while its baseline saw five hand-picked summary numbers. A win under those conditions wouldn't distinguish "representation-space prediction is the right objective here" from "more raw information helps,"

which nobody doubted. §5.2's staged plan exists specifically to produce that comparison honestly, by making the simpler models real deliverables instead of strawmen built to lose.

The feedback loop in §6.2. Specific to this class of model, and not addressed by anything the earlier fitted models in this series needed.

None of these four findings says the idea is wrong. Each says the first attempt to test it would not have produced evidence worth trusting, in either direction. Publishing a paper that reported a result built on any of the four would have been the wrong kind of paper — the kind that states an answer this program hasn't yet earned the right to state. Publishing the formulation, and naming plainly what has to be true before the next attempt is worth running, is the version of this paper that's actually honest about where the work stands.

8. Limitations

Everything in §3's codec argument is a suggestive parallel, not a proof. Video's redundancy — spatial, temporal, learned once across an enormous training distribution — may or may not have a real analogue in sparse, discrete, typed organizational event streams, and I haven't established that it does. That's precisely the question stage two's same-information comparison is built to answer, and until it runs, the codec framing in §3 is an argument for why the question is worth asking, not evidence that the answer is yes.

The coordination-failure formula in §2 is my own synthesis, not a derivation from a literature I've read closely. The one citation I attached to it is flagged as unverified in this paper's own reference list, which is an unusual thing to admit in a bibliography and the honest one.

The plans-as-facts prerequisite in §4 is stated as a requirement without confirming whether it's currently true or false in this codebase. That census is prep work, not something this paper does.

Nothing here addresses whether an organization **should** build this, only what it would take to build it responsibly if the answer is yes. The safe-output test in §2 and the shadow-mode requirement in §6.2 are the beginnings of that answer, not the whole of it.

9. Conclusion

The last piece of software I argued for, in an earlier essay, was a decoder good enough that an organization could ship less and reconstruct more — while trusting what got reconstructed. This paper is the formulation of what that decoder would need to look like for organizational coordination specifically: a model of initiatives, not people; trained in stages so that a representation-learning claim has to earn its place against a same-information baseline rather than being assumed; collecting from a ledger that already exists rather than one invented for the purpose; and made operational through exactly the type discipline and evidence-over-decoding principles this substrate already holds everything else in it to.

Four adversarial reviews found four real reasons not to train yet. I've stated all four plainly, along with what would have to be true for each to stop applying. That's the actual deliverable of this paper: not a trained model, a formulation precise enough that the next attempt — whenever the data exists to support one — can be evaluated against a standard this one couldn't yet meet.

> **Code and correspondence.** The work described here is implemented, not sketched. The source behind every claim in this paper — the engine, the extension, the fixtures and the tests that hold the invariants — can be shared on request, including the parts that did not work. Write to kenneth.pernyer@reflective.se. We are equally interested in being told where the argument is wrong.

References

Assran, M., Bardes, A., Fan, B., Garrido, Q., Howes, R., Komeili, M., Muckley, M., Rizvi, A., Roberts, C., Sinha, K., others, 2025. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. arXiv preprint.

Assran, M., Duval, Q., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M., LeCun, Y., Ballas, N., 2023. Self-supervised learning from images with a joint-embedding predictive architecture. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

Bardes, A., Garrido, Q., Ponce, J., Chen, X., Rabbat, M., LeCun, Y., Assran, M., Ballas, N., 2024. Revisiting feature prediction for learning visual representations from video. arXiv preprint.

Harnad, S., 1990. The symbol grounding problem. Physica D: Nonlinear Phenomena 42, 335–346.

LeCun, Y., 2022. A path towards autonomous machine intelligence. Open Review 62, 1–62.

Malone, T.W., Crowston, K., 1994. The interdisciplinary study of coordination. ACM Computing Surveys 26, 87–119.

Pernyér, K., 2025a. Building the "last piece of software" means building an intent codec. Reflective Labs Signal.

Pernyér, K., 2025b. Proposal, gate, commitment: deterministic convergence as a substrate for organizational decisions. Reflective Labs technical report.

Pernyér, K., 2025c. Training as a Formation: fitted models under governed provenance. Reflective Labs technical report.

Skenderi, G., Li, H., Tang, J., Cristani, M., 2023. Graph-level representation learning with joint-embedding predictive architectures. arXiv preprint.

Thimonier, H., Costa, J.L.D.M., Popineau, F., Rimmel, A., Doan, B.-L., 2024. T-JEPA: Augmentation-free self-supervised learning for tabular data. arXiv preprint.

A. The staged build plan, operationally

J1 Collect Reuse the Converge ledger's own fact stream as the event vocabulary. Promote plans as revisable facts before drift-from-plan detection is attempted. Pin label timing to strictly post-window facts before any labelled evaluation runs.

J2 Stage one A calibrated classifier over explicit ledger features — block count, escalation count, dependency-churn count, decision age, promotion-rate slope. Ships inside the existing fitted-model extension's provenance discipline with minimal new interface.

J3 Stage two A sequence or graph model reading the full tokenized event history, evaluated against stage one under identical information — the control stage one alone cannot provide for itself.

J4 Stage three JEPA-style self-supervised pretraining — context encoder, predictor, EMA-updated target encoder, loss in representation space — gated on stage two showing the additional machinery earns its cost.

J5 Stage four An intervention model, gated on real causal evidence or controlled experiments existing first. Not attempted on correlational grounds alone.

J6 Serving Shadow mode until an uncontaminated evaluation exists. Suggestor placement at simulation and adversarial review, never as a self-sufficient gate. Confidence reported as one of exactly two honest categories from §6.1, never blended with a third.

B. The four review findings, restated as acceptance criteria

Before a revised hypothesis is worth running through adversarial review again, each of the following should have a stated, checkable answer rather than a hedge:

- **Label timing.** The outcome label's computation is specified precisely enough that a third party could implement it without asking a clarifying question, and it provably cannot reference facts from inside or after the masked window.
- **Information parity.** Every baseline in the comparison receives the same input a JEPA-style model would receive, differing only in how that input is processed — no baseline is deliberately under-featured relative to what it could reasonably use.
- **The feedback loop.** A concrete answer to how long shadow-mode serving runs before any prediction reaches a reviewer, and what a clean, uncontaminated evaluation actually requires before that gate opens.
- **Data volume.** A real number from the census in §4, not an estimate — and if that number is too small for stage two or three, an explicit decision to wait rather than to proceed on an underpowered sample and call the result preliminary.