Nick Clark · Published April 25, 2026 · PDF

How to Meet the EU AI Act's Training-Data Transparency Requirements: A Provenance-Bound Architecture

If you train or fine-tune a general-purpose model, you now have to say what went into it, prove restricted content stayed out, and answer content owners who ask whether their work was used. This guide describes an architecture for doing that at training time rather than reconstructing it afterward. It is an architectural approach disclosed in United States Patent Application 19/647,395, not a shipping library, and it centers on the Training Governance inventive step: governing what a model may learn at the same boundary where you already govern what it may say.

What You Are Building

You are training or fine-tuning a model and you now have obligations about the data that went into it. The EU AI Act pushes providers of general-purpose models toward publishing a sufficiently detailed summary of training content, honoring rights reservations and text-and-data-mining opt-outs, and being able to answer downstream regulators and rights holders about what the model learned from. The recurring engineering problem underneath all of that is the same: at the moment a regulator or a content owner asks "was my content in your training set, and how deeply did the model absorb it," most training pipelines cannot answer, because the pipeline never recorded the question's inputs while it had them.

What you are building is a training loop that answers that question by construction. Concretely: a governed boundary between "we computed a gradient from this example" and "we applied that gradient to the model," where every example is admitted, attenuated, or refused against its own provenance and licensing metadata, and every one of those decisions is written to a tamper-resistant log you can hand to an auditor.

This guide describes the architecture disclosed in United States Patent Application 19/647,395. It is a design you implement yourself, not a package you install.

Why the Obvious Approaches Fall Short

The common approaches are legitimate and widely used. They just leave a structural gap for the specific obligation of proving what a model learned.

Dataset-level manifests and datasheets. Teams catalog the corpora they trained on: this crawl, that licensed dataset, this synthetic set. This is useful and often the baseline a transparency summary is built from. The gap is granularity and coupling. A manifest describes the corpus you intended to use; it does not, on its own, record which individual examples actually reached the optimizer, at what magnitude, and under which policy. When a specific rights holder asks about their specific work, a corpus-level manifest cannot confirm presence or absence of that item, and it cannot say how deeply it was integrated.

Post-hoc unlearning. When restricted or revoked content turns out to be in a trained model, one option is to approximate its influence and apply corrective updates to remove it. This is a real and active research area, but it is inherently approximate: a single example's influence on a deep network is diffused across a very large number of parameters through the non-linear dynamics of gradient-based optimization, so the parameter changes attributable to that example cannot be exactly identified and reversed. You are estimating and undoing damage after the fact rather than preventing it.

Filter-then-train. You clean the corpus up front, drop what you cannot use, and train on what remains. This handles clear exclusions but produces a binary in/out decision made once, disconnected from the training loop, and typically without a per-example record of why each item was kept and what happened to it during training. It also treats "we should not deeply memorize this, but it is fine to learn from lightly" as unrepresentable.

The structural gap common to all three: governance and provenance live outside the training loop and are reconstructed after it, when the information you needed has already been averaged into weights.

The Architecture

The disclosed approach reconceives the training loop as a governed execution environment. Each training iteration is treated as a proposed mutation to the model's knowledge state that must be evaluated for admissibility before it is committed, the same way an inference-time system evaluates a candidate output before emitting it. The following mechanisms all trace to the filing.

A governed boundary inside the loop. A semantic substrate is positioned at the boundary between the forward-pass loss computation and the backward-pass gradient application. Gradients are computed exactly as in conventional training; the substrate does not alter the mathematics of gradient computation or the optimizer update. What it governs is which gradient signals reach which layers, and at what magnitude, based on the semantic properties of the content that produced them. Refusing to integrate an example (non-training) is a valid computational result, not an error, and it is recorded as a governed event.

Every example carries governance metadata. An example presented as raw content alone cannot be evaluated and is inadmissible by default. Each example must carry, at minimum: an entropy band classification (a measure of the content's semantic

complexity and information density relative to the model's current state), a slope position in the platform's trust hierarchy, a content provenance record identifying source, acquisition pathway, and chain of custody, and a policy scope naming the licensing terms, usage restrictions, temporal validity bounds, and exclusion mandates that apply. The corpus stops being an undifferentiated mass of data and becomes a collection of governed, annotated objects.

Depth profiles, not just admit/reject. The admissibility decision is graded. Its output is a depth profile: a per-layer (or per-block) contribution weight vector. A weight of one lets the full gradient reach that layer; zero blocks it entirely; a value in between attenuates it. An example can be admitted for shallow integration and excluded from deep integration. Depth profiles are indexed by entropy band, so content well-represented in the model already tends toward shallow profiles while genuinely novel content tends toward deeper ones, and the association adapts as the model's internal representations stratify during training.

Policy-governed retention and suppression. This is where the transparency obligation is met structurally. Content under time-limited licensing is trained with a suppressed depth profile, with deep-layer weights at or near zero, confining its influence to shallow layers so that later de-emphasis is a targeted shallow adjustment rather than model-wide retraining. Content in a governed exclusion corpus gets a zero-weight profile at every layer and never influences any parameter. Crucially, this is structural prevention, not unlearning: there is no need to unlearn what was never deeply learned, and because a zero weight at a block means no gradient reaches that block, the prevention is exact and auditable rather than approximate. When multiple policies apply to one example, resolution is deterministic and applies the most restrictive policy. The result is a model whose knowledge structure reflects its governance constraints: freely licensed content encoded deeply and durably, restrictively licensed content encoded shallowly and separably, excluded content not encoded at all.

A training provenance log. The substrate writes a chronologically ordered, append-only record for each batch or example: entropy band, slope position, the depth-aggregation profile applied, the per-layer contribution weights that actually reached each block, a governance record naming the policy objects that authorized admission and set the depth profile, the content provenance record, and the admissibility determination with the reason for any modification or rejection. Entries are timestamped and sequentially numbered so they cannot be silently reordered or deleted, and the filing describes periodically sealing the log to produce tamper-evident checkpoints for third-party verification. The log supports forward queries (from a content item to the layers and magnitudes it influenced) and reverse queries (from an observed model behavior back to the bounded set of content that was structurally permitted to influence the active layers). The filing is explicit that a reverse query does not definitively attribute behavior to specific content, because gradient-based optimization precludes exact attribution; it narrows the candidate set well below the full corpus.

This log is what you hand to an auditor. When a content owner asks whether their content was used, the log answers definitively: present, with its provenance and depth records, or absent. When a regulator needs evidence that restricted content was not deeply integrated, the log shows the contribution weights that confined it.

How to Approach the Build

You are implementing this yourself against your own training stack. A workable order:

1. **Enrich the corpus into governed objects.** Before training, attach the required metadata to each example: an entropy/complexity band, a provenance record (source, acquisition pathway, chain of custody), and a policy scope (license terms, temporal bounds, exclusion flags). Decide your default: the filing's default is that missing metadata means inadmissible. That default is what makes the transparency claim honest.

2. **Model policy as first-class objects.** Represent licensing, regulatory, and platform rules as policy objects a resolver can consult, and define the resolution rule (most-restrictive-wins is the disclosed choice). Reusing the same policy objects that govern your inference-time behavior is the point: one governance vocabulary, two enforcement sites.
3. **Insert the substrate at the gradient boundary.** In your training step, after loss and gradients are computed and before the optimizer update, call an admissibility evaluator that returns a depth profile. Illustrative interface sketch, faithful to the filing and not a drop-in implementation:

```
# illustrative only
profile = substrate.evaluate(example.metadata, policies)   # per-block we
if profile.rejected:
    log.append(non_training_event(example, profile.reason)) # refusal is a
    continue
grads = scale_per_block(grads, profile.weights)            # 0 blocks a b
optimizer.step(grads)
log.append(training_event(example, profile))
```

   The optimizer receives a normal-looking gradient buffer; only its per-block magnitudes changed.
4. **Define depth profiles per band and policy class.** Map entropy bands to baseline profiles, then let policy override toward suppression: suppressed (deep weights near zero) for time-limited or revocable content, zero-weight for exclusion-corpus content. Plan for the profiles to adapt across training rather than being fixed at the start.
5. **Make the log append-only and sealable.** Enforce sequential numbering and timestamps, forbid mutation, and periodically seal checkpoints so a third party can verify integrity. Without this property the log is not audit evidence.

6. **Build the query surface last.** Implement forward and reverse queries over the log so you can answer "was this used and how deeply" and "which content could have driven this behavior." Present reverse-query results as a bounded candidate set, not a definitive attribution.

What This Does Not Give You

This is an architecture, not a downloadable SDK. There is no package to install and nothing here "just works" out of the box; you build it against your own model, data loader, and optimizer, and the effort is real. The approach has not been benchmarked or productized here, and this guide reports no performance numbers because the filing states none.

It is also not legal advice or a compliance certification. The EU AI Act's obligations, the shape of an acceptable training-content summary, and how a rights reservation must be honored are legal determinations; this architecture gives you the technical substrate to support them, not a guarantee that any particular regulator will deem your program sufficient. Reverse provenance queries narrow attribution but, per the filing itself, cannot exactly attribute a model behavior to a specific example. The suppression mechanism reduces but does not by itself prove elimination of memorization risk for shallow-encoded content. And the whole thing rests on metadata quality: if your provenance and policy annotations are wrong, the log faithfully records wrong governance. The architecture makes your governance auditable; it does not make it correct.

Disclosure Scope

The approach described here is disclosed in United States Patent Application 19/647,395. This guide is educational: it explains an architectural approach so a skilled developer can understand and build it themselves. It is not a warranty, a compliance certification, legal advice, or an offer of software, and it does not describe a shipping

product. Every mechanism attributed to the disclosed approach traces to that filing; where the filing states a limitation, such as the inability of a reverse query to exactly attribute behavior to specific training content, this guide states that limitation too.

Training Governance (/training-governance)

All 40 steps → (/inventive-steps)

Govern what the model learns, at what depth, with what provenance.

Chapter 11 (/patents/19-647395/chapters/training-governance)

PRIMARY TECHNICAL DISCLOSURE

- Depth-Selective Training Governance for Machine Learning Systems (/articles/depth-selective-training-governance-for-machine-learning-systems)

SECONDARY TECHNICAL

- Training Examples as Proposed Semantic Mutations in Governed Training (/articles/training-governance/mutation-proposals)
- Entropy-Band-Indexed Training Depth Profiles (/articles/training-governance/entropy-depth-profiles)
- Depth-Selective Gradient Routing for Governed Training (/articles/training-governance/gradient-routing)
- Training-Level Memorization Detection (/articles/training-governance/memorization-detection)
- Differential Privacy Through Depth-Selective Routing (/articles/training-governance/differential-privacy)
- Governed Fine-Tuning With Verifiable Provenance (/articles/training-governance/fine-tuning-provenance)
- The Training Loop as a Governed Execution Environment (/articles/training-governance/governed-training-loop)
- Policy-Governed Knowledge Retention and Suppression (/articles/training-governance/knowledge-retention)
- Provenance-Traceable Training Dynamics (/articles/training-governance/provenance-tracing)
- Curriculum-Integrated Depth Scheduling (/articles/training-governance/curriculum-depth)
- Affect-Modulated Training Depth (/articles/training-governance/affect-modulated-depth)

- Training-Inference Governance Integration (/articles/training-governance/training-inference-integration)
- Training Governance for Human-Relatable Agents (/articles/training-governance/human-relatable-training)
- Governed Training with Depth-Selective Gradient Aggregation (/articles/training-governance/edge-fleet-training)

APPLICATIONS · GENERAL

- Rights-Compliant Model Training Through Depth-Selective Gradient Routing (/articles/training-governance/rights-compliant-training)
- Regulated Industry Model Governance: Verifiable Training Provenance for AI Compliance (/articles/training-governance/regulated-model-governance)
- Training Governance for Medical AI: Auditable, Depth-Controlled Clinical Model Training (/articles/training-governance/medical-ai-training)
- Training Governance for Legal AI: Encoding Precedent Hierarchy Into the Model (/articles/training-governance/legal-ai-training)
- Training Governance for Financial AI: Auditable Model Risk Management Under SR 11-7 (/articles/training-governance/financial-model-training)
- Training Governance for Defense AI (/articles/training-governance/defense-ai-training)
- How to Train an Educational AI Tutor That Learns Pedagogy Deeply and Misconceptions Shallowly (/articles/training-governance/educational-model-training)
- Copyright-Compliant Training for Creative and Generative AI Models (/articles/training-governance/creative-ai-training)
- Training-Data Provenance for Regulated Autonomy: UNECE, FDA, and EU AI Act Compliance (/articles/training-governance/regulated-autonomy-training)
- Tamper-Evident Fleet Training Records for Maritime and Agricultural Operations Without Cellular Connectivity (/articles/training-governance/maritime-fleet-training)
- 21 CFR Part 11 Compliance for AI Model Training: Audit Trails, Electronic Signatures, and Provenance (/articles/training-governance/cfr-21-part-11)

APPLICATIONS · SPECIFIC

- OpenAI Training vs Governed Depth-Selective Training (/articles/training-governance/openai-training)
- Anthropic Constitutional Training vs Governed Training: What Depth-Selective Provenance Adds (/articles/training-governance/anthropic-training)
- Stable Diffusion Training vs Governed Provenance: The Missing Layer in Stability AI's Pipeline (/articles/training-governance/stability-ai)

- Midjourney vs Governed Training: Depth-Selective Provenance for Generative Art (/articles/training-governance/midjourney)
- Scale AI Alternative: Governed Learning Beyond Data Labeling (/articles/training-governance/scale-ai)
- Labelbox Alternative for Governed Training: Annotation Workflows vs Learning Dynamics (/articles/training-governance/labelbox)
- Snorkel AI Programs Labels but Does Not Govern Gradient Depth (/articles/training-governance/snorkel-ai)
- Weights & Biases Alternative for Governed Training: Tracking vs Depth-Selective Governance (/articles/training-governance/weights-biases)
- Determined AI Alternative: Governed Training Beyond Compute Orchestration (/articles/training-governance/determined-ai)
- MosaicML Optimizes Training Efficiency, Not Learning Governance (/articles/training-governance/mosaic-ml)
- Tesla Shadow Mode vs Governed Fleet Training (/articles/training-governance/tesla-shadow-mode)
- Symbotic Warehouse Automation vs Governed Training Provenance (/articles/training-governance/symbotic-warehouse)
- Governed Alternative to Google Gemini and Vertex AI Tuning (/articles/training-governance/google-gemini-tuning)
- OpenAI Fine-Tuning and RFT vs Governed, Depth-Selective Training (/articles/training-governance/openai-fine-tuning-rft)
- PathAI Alternative: Governed Digital-Pathology Training (/articles/training-governance/pathai-pathology)
- Tempus AI vs Governed Medical-AI Training: Training-Step Provenance (/articles/training-governance/tempus-ai-medical)

Training Governance overview → (/training-governance)