Audio Echo Engine (AEE) — ML Calibration Layer

Status: Draft for methods review (implementation-guiding, not code)
Date: 2025-11-15 (America/Chicago)
Supersedes: None (first ML-specific calibration doc)
Primary Author: **Keith Hetrick** (Architecture, Methods, Governance)
Contributors: Keith Hetrick — Maintainer [future: Data/ML collaborators]
Affiliation: Bellweather Studios
Contact: keith@bellweatherstudios.com (mailto:keith@bellweatherstudios.com)
Doc ID: CIF-ML-AEE-Cal-v0.1

Abstract

This whitepaper defines **v0.1 of the ML Calibration Layer** for the **Audio Echo Engine (AEE)** inside the Creative Intelligence Framework (CIF). It is designed to sit **on top of** the existing deterministic pipeline that computes six equal-weight axes, the **Equal-Axis Composite Mean (EACM)**, and the audio-only **Hit Confidence Index (HCI)** as formalized in CIF v1.2.

The ML layer **does not replace AEE or HCI math**. Instead, it:

- Learns a **calibration map** from AEE outputs (axes, EACM, key features) to **observed outcome signals** (e.g., hit vs baseline performance).
- Identifies and reduces **systematic bias, false precision, and over/under-sensitivity** in the current hand-tuned thresholds—especially on **Energy** and **Danceability**.
- Produces an **AEE-Calibrated Score** (`HCI_cal`) and **diagnostics** that can be:
  - Used internally to refine AEE parameters,
  - Exposed as an **experimental advisory lane**, while `HCI_v1.2` remains the canonical KPI.

The philosophy mirrors **credit scoring (FICO), standardized testing, and recommender calibration**: a stable, well-specified “rules engine” (AEE) is augmented by a small, auditable ML layer that calibrates and sanity-checks scores against real-world outcomes, under explicit governance and fairness constraints.

Document Control

- **Version**: v0.1 (draft)
- **Change Summary**: Introduces ML Calibration Layer on top of AEE; specifies data requirements, model family, fairness and governance guarantees, integration path, and acceptance tests. HCI math remains unchanged in v1.2.
- **Reviewers**: [list]
- **Approvals**: [list]
- **Effective Date**: [date]
- **Superseded Docs**: None; this complements:
  - *CIF Technical Whitepaper v1.2* (CIF-TW-v1.2) — governance and KPI spec

    - *CIF Technical Whitepaper — Under-the-Hood Methodology (v1.1)* — axis math and examples
- **Verification Thread**: CIF-ML-Cal — Appendix K (Calibration Runbook Charter).

Global Acronyms & Terms (ML Calibration)

Engine & Models

- **AEE** — Audio Echo Engine (six-axis audio model; current KPI domain)
- **HEM** — Historical Echo Model (inside AEE)
- **LEE** — Lyric Echo Engine (advisory-only in v1.2)
- **HLM** — Historical Lyric Model (future, in LEE)
- **AEE-ML** — ML Calibration Layer sitting on top of AEE outputs
- **Base HCI** — `HCI_v1.2` as defined in CIF (caps + EACM, audio-only)
- **HCI_cal** — AEE-Calibrated Score produced by the ML layer (internal / experimental)

Data & Labels

- **Outcome Label (Y)** — aggregated performance signal: e.g., hit vs non-hit, decile, or lane-level success (Radio/Streaming)
- **Backtest Set** — historical songs with both:
    - AEE outputs (axes, HCI_v1.2), and
    - Observed outcomes (charts/streams/supervised labels)
- **Calibration Window** — temporal window used to train ML (e.g., last N years)

Methods

- **Mono-Model** — a compact, single model (e.g., logistic regression or monotone GBDT) used for calibration
- **Isotonic / Platt** — probability calibration methods applied to raw model scores
- **Monotonicity Constraint** — constraint that increasing any axis cannot decrease calibrated score

Governance & KPI

- **Canonical KPI Lane** — Radio US (unchanged; HCI_v1.2 remains canonical)
- **Experimental Lane** — ML-calibrated lane (`HCI_cal`) used in shadow mode or advisory only
- **40/40 Balance Rule** — future rule: equal policy weight to audio and lyric once LEE graduates (relevant for future ML in LEE, not v0.1)

1. Purpose & Scope

1.1 Motivation

The current AEE pipeline:

- Extracts audio features via Automator,
- Normalizes via winsorization, z-scores, and logistic maps,
- Constructs six equal-weight axes,
- Aggregates axes into EACM and applies caps to obtain HCI (audio-only KPI).

This pipeline is **deterministic, interpretable, and reproducible**, but:

1. Certain axes (notably **Energy** and **Danceability**) are **hard to calibrate heuristically** and may misalign with lived experience (e.g., “Blinding Lights” scoring too low compared to real-world dominance).
2. **Static thresholds** tuned by hand can drift or embed subtle bias as the market changes.
3. There is currently **no systematic, statistical calibration** tying `HCI_v1.2` to observed outcomes, beyond manual inspection.

The **ML Calibration Layer** addresses this by:

- Learning how combinations of axes and key features relate to **actual hit outcomes**.
- Producing a calibrated score `HCI_cal` that:
  - can **flag** inconsistencies,
  - **suggest** threshold adjustments,
  - and eventually serve as an **advisory lane**.

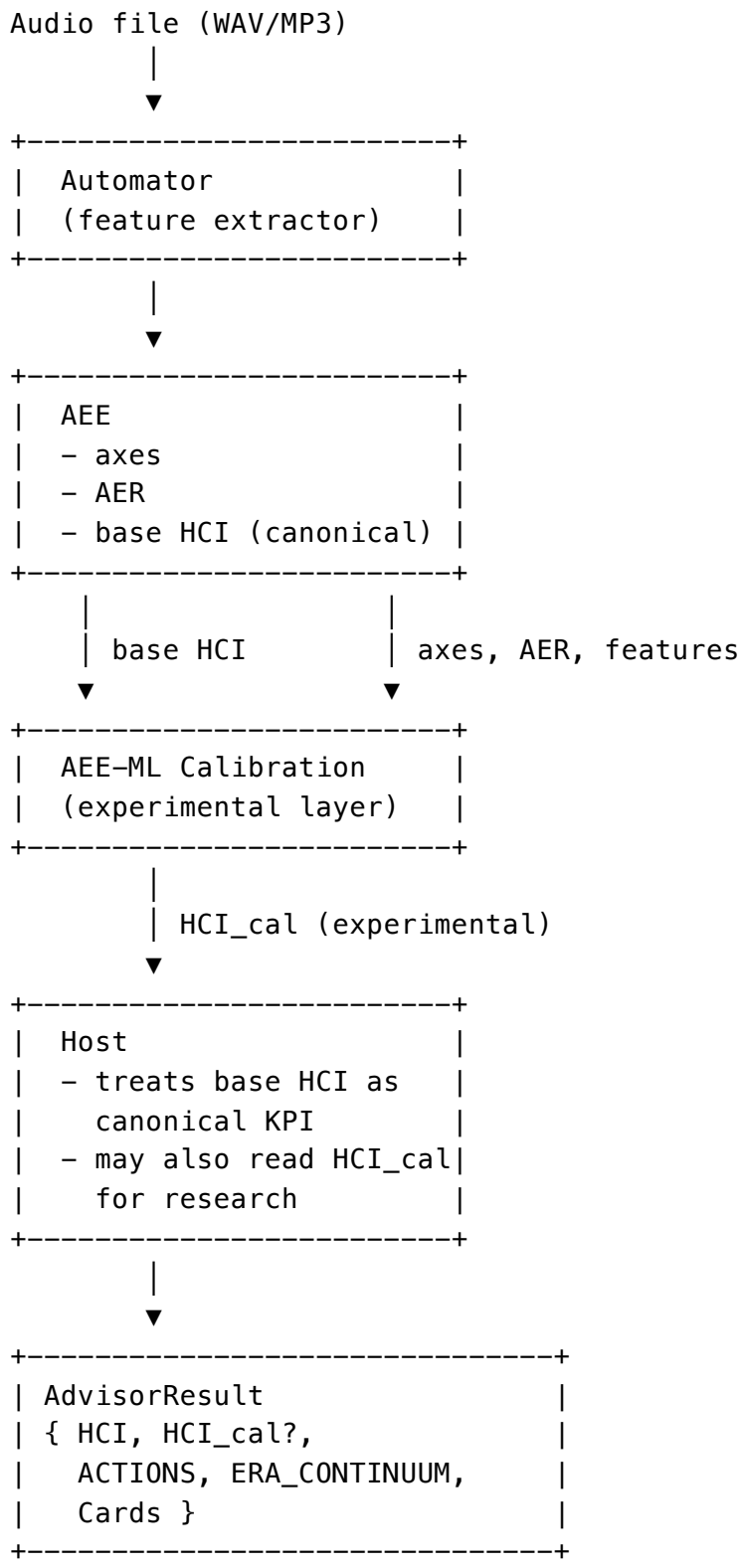
1.2 Non-Goals

- **Not** replacing AEE axes or HCI math defined in v1.2.
- **Not** allowing streaming outcomes or preferences to **directly overwrite** axis definitions or Host governance rules.
- **Not** building a large opaque deep model; v0.1 uses **small, interpretable ML**.

2. Placement in CIF Topology

2.1 High-Level Diagram

This extends the v1.2 interface-only topology by inserting an ML calibrator **after** AEE/AER but **before** Host advisory lanes, and strictly **outside** the canonical KPI path.



In v1.2+, AEE produces canonical base HCI; AEE-ML is a sidecar calibration layer that proposes HCI_cal. The Host always treats base HCI as the KPI, optionally logging or displaying HCI_cal as an experimental comparator.

Invariants:

1. **Base HCI** (v1.2) remains the **canonical KPI** for Radio US.
2. AEE-ML **cannot write back** into features, axes, AER, caps, or Host fusion math.
3. AEE-ML outputs are either:
    - Hidden (shadow mode), or
    - Surfaced as **advisory-only** fields (e.g., `HCI_cal`, calibration residuals, decile flags).

2.2 Data Flow Summary

- **Input**: per-track **Track Analysis Pack (TAP)** including:
    - AEE features, axis scores, EACM, base HCI
    - Profile metadata (`US_Pop_2025` , norms timestamp, seeds, env hash)
- **ML Training**: uses TAP + outcome labels to fit a **calibration model**.
- **ML Inference**: at runtime, TAP → ML model → `HCI_cal` , plus diagnostics.

- **Host**: may expose `HCI_cal` in **Cards** or as an **experimental lane**, but **never replaces** `HCI` without governance approval.

3. Data & Label Definition

3.1 Backtest Dataset

The calibration dataset consists of:

- **Tracks**:
  - Core benchmark set (e.g., your 50-song benchmark used for axis validation).
  - Additional tracks from the same **profile window** (e.g., `US_Pop_2025`, anchored 1986–2025).

- **Features per track**:
  - AEE axes:
    - `A_market`, `A_sonic`, `A_emotional`, `A_historical`, `A_cultural`, `A_creative`
  - EACM and base HCI:
    - `EACM_audio`, `HCI`
  - Key raw-ish features useful for ML:
    - `tempo_bpm`, `runtime_sec`, `loudness_LUFS`
    - `danceability`, `energy`, `valence`
    - segmentation-derived features (TTC, chorus_lift) when available

3.2 Outcome Signals (Labels)

Labels should be **coarse and robust**, not overly granular or overfitted. Examples:

- **Binary**: `hit` vs `non_hit` based on chart peak / sustained performance
- **Ordinal tiers**: S/A/B/C/D/F (e.g., top decile, upper quartile, etc.)
- **Lane success**: boolean for “Radio US lane success” vs “Streaming-only lane success”

For v0.1 we recommend:

> **Binary or 3-tier outcome** (e.g., high/medium/low), with:
>
> - High = tracks clearly and strongly successful in the target lane
> - Medium = modest but meaningful performance
> - Low = no measurable success in that lane

The ML layer learns a mapping:

$$f_{\mathrm{ML}} : \{A_k, EACM, \mathrm{features}\} \rightarrow p(\mathrm{hit} \mid \mathrm{track})$$

where (p) is then calibrated and transformed into **`HCI_cal`** on ([0,1]).

4. Model Family & Constraints

4.1 Design Principles

The model family must:

1. Be **small and interpretable** (no black-box deep nets).
2. Respect **monotonicity**: improving any axis should **never** lower `HCI_cal`.
3. Permit **probability calibration** (isotonic or Platt).
4. Be easy to re-train when norms shift (new calibration windows).

4.2 Recommended v0.1 Model

A two-stage design:

1. **Core scorer**
    - Either **logistic regression** over:
        - AEE axes + EACM + a small set of features (tempo, loudness, TTC, etc.), or
    - A small **monotone gradient-boosted tree** (GBDT) with monotonicity constraints on axes
2. **Probability calibrator**
    - **Isotonic regression** or **Platt scaling** to map raw scores to calibrated hit-probabilities.

Conceptually:

$$z = w_0 + \sum_k w_k A_k + w_{\mathrm{E}} EACM + w_{\mathrm{T}} \phi(\text{tempo}) + w_{\mathrm{L}} \phi(\text{loudness}) + \dots$$

$$p_{\text{raw}} = \sigma(z), \quad p_{\text{cal}} = \mathrm{IsoCal}(p_{\text{raw}})$$

$$\mathrm{HCI\_cal} = \min\left(c_{\text{audio}}, p_{\text{cal}}\right)$$

Where $(c_{\text{audio}} = 0.58\)$ is the existing audio cap from CIF v1.2, keeping the cap semantics consistent.

4.3 Monotonicity Constraints

To preserve interpretability and align with CIF properties:

- Constrain the model such that:

$$\frac{\partial \mathrm{HCI\_cal}}{\partial A_k} \geq 0 \quad \forall k$$

This ensures:

> If a track's Market/Sonic/Emotional/etc axis rises (holding others fixed), `HCI_cal` cannot drop.

In GBDT-style models, this is enforced by **monotone constraints**; in logistic regression, by **sign-constrained weights**.

5. Integration with AEE & HCI

5.1 Modes of Operation

1. **Shadow Mode (default for v0.1)**
    - AEE runs as usual → axes, EACM, HCI.
    - AEE-ML runs in parallel:
        - Produces `HCI_cal` and per-track residuals `Δ = HCI_cal − HCI`.

    - Host logs both in Cards, but **only HCI** is surfaced as KPI.

2. **Advisory Lane**
    - Once validated, `HCI_cal` may be exposed as:
        - A separate lane (e.g., `radio_us_ml_calibrated`), or
        - A field in **AdvisorResult** (e.g., `HCI_calibration` block with confidence metrics).
    - Still **not** a canonical KPI; described clearly as "ML-calibrated advisory signal".

3. **Future Fusion** (beyond v0.1, gated by governance)
    - Possible future: canonical HCI_vX integrates both:
        - AEE structural math and
        - ML-calibrated probability,
    - With strict fairness and monotonicity checks.

5.2 Alignment with CIF v1.2

CIF v1.2 defines (simplified):

$$\mathrm{HCI} = \beta \cdot \min(EACM_{\text{audio}}, c_{\text{audio}}) + (1-\beta) \cdot \min(EACM_{\text{lyric}}, c_{\text{lyric}})$$

with β=1.0 today (audio-only KPI, lyrics advisory).

The ML layer **does not alter this formula** in v0.1. Instead:

- It produces:

$$\mathrm{HCI\_cal} = g_{\mathrm{ML}}(A_k, EACM, \text{features}) \in [0,1]$$

with similar cap semantics, and

- HCI remains **the canonical KPI** printed on Cards and used in evaluation.

6. Fairness, Bias, & Governance

6.1 Bias Sources & Mitigations

Potential bias sources:

- **Selection bias** in which tracks are labeled as "hits" (industry, genre, platform bias).
- **Temporal drift**: calibration trained on one era but applied to another.
- **Proxy leakage**: inadvertently learning from features correlated with non-musical factors.

Mitigations:

1. **Anchored window**: training only within the **profile window** already defined for AEE (e.g., 1986–2025 for `US_Pop_2025`).
2. **Lane-specific labels**: calibrate per-lane (Radio vs streaming) when needed, not mixing them.
3. **Feature discipline**: restrict ML input to:
    - AEE axes and core acoustic features,
    - Avoid direct inclusion of metadata like label, region, or artist identity.

4. **Monotonicity + Caps**: preserve CIF's fairness guardrails; never allow ML to make the score behave non-intuitively when axes improve.

6.2 Governance Hooks

Example **Policy Switch Registry** block:

```
{
  "ml_calibration_enabled": false,
  "ml_calibration_lane": "radio_us_ml_cal",
  "ml_calibration_in_kpi_path": false,
  "ml_calibration_model_id": "AEE-ML-v0.1",
  "ml_calibration_training_window": "2015-2025",
  "ml_calibration_effective_policy_hash": "<hash>"
}
```

Any change to:

- `ml_calibration_enabled` , or
- `ml_calibration_in_kpi_path`

must trigger:

- Change-log entry,
- Golden-run comparison,
- Calibration report attached to Appendix K.

7. Validation, Metrics & Acceptance Tests

7.1 Metrics

For each calibration run, report:

- **Discrimination**
  - AUROC, AUPRC for `hit` vs `non_hit`
- **Calibration**
  - Brier score
  - Calibration plots (predicted vs empirical probability)
- **Stability**
  - Performance across time-slices within training window
  - Performance across sub-cohorts (tempo bands, eras, runtime bins)
- **Consistency with HCI**
  - Distribution of residuals `Δ = HCI_cal − HCI`
  - Fraction of tracks where ML suggests a materially different story (e.g., $|\Delta| > 0.05$)

7.2 Monotonicity & Invariant Checks

- Verify that **monotonicity holds** numerically:
  - For random draws of axis vectors, perturb one axis upward and assert `HCI_cal` does not decrease.
- Confirm **caps** and **bounds**:
  - `HCI_cal ∈ [0, 1]`
  - `HCI_cal ≤ c_audio` (if we keep the same 0.58 cap semantics)

7.3 Acceptance Criteria (v0.1)

Shadow mode can be considered **ready to surface** internally when:

1. Calibration statistics are stable across:
    - 3–5 temporal splits,
    - at least one cross-profile sanity check if applicable.
2. Monotonicity tests pass, with no violations beyond tight numerical epsilons.
3. Golden-run alignment:
    - With ML disabled, HCI results remain unchanged and match v1.2 golden runs (ΔHCI ≤ 0.002 per track).
4. Residuals show:
    - no gross systemic skew (e.g., whole decadal block consistently underpredicted),
    - and ML primarily refines outliers (e.g., “Blinding Lights”-type examples).

8. Example: “Blinding Lights” Calibration Story (Conceptual)

Illustrative numbers only (methodology example; not actual values).

Suppose:

- AEE outputs (normalized):
    - `A_market = 0.52` , `A_sonic = 0.50` , `A_emotional = 0.44` ,
    - `A_historical = 0.68` , `A_cultural = 0.49` , `A_creative = 0.55`
- `EACM_audio = 0.53` , base HCI = `min(EACM_audio, 0.58) = 0.53`
- Outcome label: `hit = 1` (clear massive success)

The ML layer might learn from the backtest that:

- Combinations of:
    - strong historical echo + tempo band + danceability pattern
    - similar to “Blinding Lights” archetypes
- are systematically under-scored by raw axis heuristics.

Then:

- Logistic/monotone GBDT outputs: `p_cal ≈ 0.76`
- After cap semantics: `HCI_cal ≈ 0.58` (top of Audio cap)
- Residual: `Δ ≈ +0.05`

Interpretation:

> AEE thought this was a “good” track (HCI ≈ 0.53).
> AEE-ML, informed by outcomes, says:
> “Within our Audio cap, this is a *top-of-band archetype*; treat as 0.58.”

This does not break EACM math; instead ML informs how to treat that region of axis-space.

9. Implementation Notes & File Layout (Repo-Level)

For the current **MusicAdvisor_AudioTools** repo, a minimal v0.1 ML layout might be:

```
MusicAdvisor_AudioTools/
├── aee_ml/
│   ├── __init__.py
│   ├── ml_config.yaml           # model ID, training window, features used
│   ├── ml_dataset_builder.py    # builds backtest dataset from TAPs + labels
│   ├── ml_model_train.py        # trains + calibrates model, writes model.pkl
│   ├── ml_model_infer.py        # loads model.pkl, computes HCI_cal, Δ
│   └── ml_eval_report.py        # generates calibration/monotonicity reports
├── tools/
│   ├── ma_fit_hci_calibration_from_hitlist.py   # existing; can call into
aee_ml
│   └── ma_apply_ml_calibration.py               # new CLI wrapper (optional)
└── ...
```

CLI examples (conceptual):

```
# Build dataset
python -m aee_ml.ml_dataset_builder \
  --root features_output \
  --labels calibration/hitlist_pop_us_2025_core_v1_2.csv \
  --out calibration/aee_ml_dataset.parquet

# Train model
python -m aee_ml.ml_model_train \
  --dataset calibration/aee_ml_dataset.parquet \
  --config aee_ml/ml_config.yaml \
  --out calibration/aee_ml_model_v0_1.pkl

# Apply in shadow mode
python -m aee_ml.ml_model_infer \
  --root features_output \
  --model calibration/aee_ml_model_v0_1.pkl \
  --out-root features_output_ml_cal
```

10. Assurance Case (Text GSN for ML Layer)

Claim CM0: AEE-ML v0.1 is **stable, monotone, reproducible**, and **does not compromise HCI governance** for `US_Pop_2025`.

- **Strategy SM1**: Show that ML layer preserves CIF invariants.
  - **Evidence EM1**: Monotonicity report; cap-respecting bounds; no changes to base HCI golden runs.
- **Strategy SM2**: Show appropriate **calibration and discrimination** on backtest sets.
  - **Evidence EM2**: AUROC, calibration plots, Brier scores across temporal splits; no catastrophic drift.
- **Strategy SM3**: Demonstrate **governance isolation**.
  - **Evidence EM3**: Policy Switch Registry entries; run cards showing `ml_calibration_in_kpi_path = false`; canonical lane still Radio US.

Assumptions:

- **A1**: Outcome labels used for calibration are **well-defined and auditable**.
- **A2**: The backtest set covers enough variety in tempo, runtime, and era to avoid severe overfitting.

- **A3**: AEE implementation is frozen and stable (versioned) during ML calibration cycles.

11. Roadmap

v0.1 (this doc)

- Shadow-mode ML calibration on AEE outputs
- Internal-only metrics and residuals
- No KPI change

v0.2

- Optional advisory lane surface (`HCI_cal`) in GPT-facing Cards
- More sophisticated residual analysis (per-axis calibration plots, partial dependence)

v1.0

- Governance review: decide whether and how ML calibration informs:
  - Axis-weight manifests
  - Norms tuning
  - or future HCI_vX fusion (while honoring CIF invariants and caps)

Future (LEE/HLM integration)

- Similar ML calibration for lyric-based HLI / LER, respecting **40/40 Balance Rule** and dual-engine governance when LEE graduates.

12. Summary

The ML Calibration Layer for AEE is a **small, disciplined bridge** between:

- The **clean, deterministic, interpretable** structure of CIF v1.2, and
- The **messy, real-world outcome data** (hits, streams, chart performance).

It uses **industry-standard calibration practice**—not as a black box that replaces AEE, but as a **scientific check and refinement tool** that improves trust, reduces bias, and provides a clearer map between HCI and what the market actually did.

It’s intentionally:

- **Narrow** (calibration only),
- **Constrained** (monotone, capped, governed), and
- **Modular** (can live in its own AIE codebase later, with Music Advisor acting as a Host).

That keeps the system **science-first, merit-based, and audit-ready** while still being understandable enough for artists, A&R, and collaborators to trust.