EGWM: Feeling the AGI
A Unified Account of Emotion-Guided World Models, Growth, Compression, Value Scales, and Routing Limits in Toy Continual Learning

Melissa Howard*
melhoward@live.ca

December 7, 2025

Abstract

Most modern large models are trained once in a massive offline phase and then deployed with nearly fixed parameters. They are brittle in non-stationary environments: they suffer catastrophic forgetting, are corrupted by noisy data, and do not manage their own plasticity. Biological agents appear to behave differently. They maintain multiple internal "worlds" or contexts and use fast value-like signals—"feelings" such as confusion, familiarity, and risk—to decide *when* and *how* to change themselves.

This unified paper collects and synthesizes five previous installments on *Emotion-Guided World Models* (EGWM). Across Parts I–V we build a sequence of toy, yet fully implemented, agents in simple 2D classification environments with multiple hidden worlds and noisy, non-stationary streams of experience.

In Part I, we introduce EGWM and show that even a single logistic regression with a simple phase-consistency feeling can refuse self-inconsistent phases and slightly improve robustness to label noise. We also add an Emotion-Guided Experiment Planner (EGEP) that uses uncertainty to choose which unlabeled points to query, achieving better sample efficiency than random querying.

In Part II, we move from one model to a *world bank* of specialist heads, governed by a small set of value channels (self-consistency, familiarity, mismatch, disagreement, and a crude elegance proxy). A hand-designed emotion-gated policy that chooses when to ignore, spawn, or update worlds cleanly recovers multiple regimes in a continual stream where a monolithic learner plateaus around 0.60–0.70 accuracy. An emotion-gated world bank instead reaches per-world accuracies in the 0.94–0.97 range. We show that the key discovery feeling is not inter-model disagreement, but *mismatch* between a fresh scratch model and the best existing world; "spawn-happy" gates that over-segment the world early and compress later perform best. Tiny learned governors trained with REINFORCE can match hand-crafted policies for accuracy and fairness, but naively elegance-weighted rewards often drive the controller toward inaction.

In Part III, we fix this "elegance collapse" by separating *growth* and *compression*. A confusion-driven growth phase creates a rich, redundant world bank; a competence-constrained compression phase merges heads only when it can do so without harming accuracy. In a stationary 3-world environment, staged grow–compress reduces the number of heads from ~ 3–4 down to about 2 while maintaining near-100% per-world accuracy and sharply reducing overlap. In a non-stationary 4-world environment where worlds appear, disappear, and reappear, a

*Independent researcher.

monolithic model exhibits catastrophic interference, while the grow–compress bank maintains high accuracy and a bounded number of heads. Sweeping the compression target reveals a competence–elegance trade-off; an annealed schedule (lenient early, strict late) allows exploration first and compact structure later.

In Part IV, we synthesize the architecture and extend it with *multi-time-scale value signals*. Fast feelings (confusion, uncertainty) operate per batch; medium-scale feelings (competence, elegance) operate over segments; slow feelings (relevance) integrate age and importance. In segmented lives with "alive" and "dead" worlds, age-only pruning deletes both rare-but-important and truly dead worlds; reward-weighted relevance prunes unimportant heads while preserving high-importance worlds and reduces the number of heads by roughly 40% relative to no-prune. Feature growth allows confused heads to add quadratic features, and tiny MLP heads on nonlinear worlds achieve 95–99% per-world accuracy when capacity matches the task. Uncertainty-driven querying then attains essentially the same accuracy as an always-query baseline while using only about 18–25% of the labels.

In Part V, we test a harder question: can the same feelings also solve the *routing problem* when multiple worlds share the same input distribution but differ in their labeling rules? We compare a monolithic model, an oracle multi-head world bank, and several routers that see only the current input and per-head feelings (probabilities and value channels), including a teacher-trained router with perfect experts. In this regime, all such routers remain far below the oracle multi-head performance and behave similarly to the monolithic baseline. Single-step feelings are not sufficient statistics for the latent world identity; this strongly motivates introducing a slower, persistent context code z_t as the next step in EGWM.

Taken together, these five parts describe a small but concrete learning core: a capacity-matched world bank whose growth, compression, forgetting, and information-seeking are governed by a handful of multi-time-scale value signals. Feelings are most powerful when they govern *structure*—when to create, reuse, merge, and retire worlds—rather than being just scalar learning rates. The toy setting is far from AGI, but it illustrates how a system can begin to "feel" when its internal theory of the world is too crude, too fragmented, or beautifully simple.

1 Introduction

Large language models and other modern learners are usually trained in a single pass with a fixed architecture and a scalar loss, then deployed with almost no plasticity. They do not decide when to learn, what to learn from, or which parts of themselves to change. This leads to familiar issues:

- **Catastrophic forgetting** in non-stationary settings.
- **Vulnerability to noise and distribution shift**, since all new data is treated equally unless strong external filters exist.
- **Lack of active learning**: models passively accept whatever data is given, instead of asking targeted questions.

Biological agents behave differently. Humans and animals appear to maintain multiple internal models of the world (contexts, habits, roles) and use fast value-like signals—what we colloquially call emotions—to decide:

- "Does this situation feel like something I've seen before?"
- "Is this new experience trustworthy, or noisy and risky?"
- "Should I form a new mental model, update an existing one, or leave everything unchanged?"

This paper develops and unifies *Emotion-Guided World Models* (EGWM), a family of toy agents that make these ideas explicit and testable in small 2D environments. The core hypothesis is:

A small set of low-dimensional value signals—"feelings" such as confusion, competence, novelty, and relevance—can govern when and how a learner changes itself, not just how strongly it changes.

Across Parts I–V, we progressively build:

1. A single emotion-gated learner that protects itself from noisy phases and uses uncertainty to guide experiment selection.
2. A world bank of specialist models whose spawning and reuse are driven by mismatch, not just disagreement.
3. A staged grow–compress scheme with elegance annealing that resolves degeneracies of naive simplicity objectives.
4. A multi-time-scale value system that grows, compresses, forgets, and actively seeks information while maintaining near-perfect competence.
5. A routing study that exposes a genuine limitation: single-step feelings cannot disambiguate latent worlds with shared inputs.

We focus on clarity rather than scale: worlds are 2D classification problems, models are logistic regressions or tiny MLPs, and all experiments are small enough to understand in detail.

2 Architecture: Emotion-Guided World Models

2.1 World Bank and Heads

EGWM maintains a bank of K heads $\{h_k\}_{k=1}^{K}$, each intended to specialize on a particular "world" or task. In different experiments, each h_k is:

- a logistic regression on raw 2D inputs,
- a logistic regression on quadratic features $\phi(x) = [x_1, x_2, x_1^2, x_2^2, x_1x_2, 1]$, or
- a tiny MLP (e.g. 2–8–1 with a tanh hidden layer).

Each head maintains a small "trusted buffer" of clean examples for replay.

In most experiments, routing at evaluation time is oracle-style: given an input, we pick the head with lowest loss or highest confidence. This isolates the effects of growth, compression, forgetting, and querying from the harder problem of online world identification, which is studied explicitly in Part V.

2.2 Value / Emotion Module

For each phase or batch, an emotion/value module computes scalar feelings from interaction statistics, including:

- **Scratch consistency** α_t: train a scratch model on the current phase D_t, measure its train accuracy.
- **Familiarity** fam_t: best accuracy of any existing head on D_t.
- **Mismatch**: $\text{mismatch}_t = \alpha_t - \text{fam}_t$, a data–model gap.
- **Uncertainty**: e.g. $u = \min(p, 1-p)$ for binary probabilities.
- **Inter-head disagreement**: fraction of points where heads disagree on predicted labels.
- **Competence**: exponential moving average (EMA) of correctness per head.
- **Loss EMA** and **volatility**: moving averages of loss and its changes.
- **Elegance proxies**: penalties based on number of heads and their overlap on a probe set.
- **Relevance**: combining age since last use and cumulative reward/importance for a head.

These values are grouped into a vector v_t and fed to a governor that chooses structural actions.

2.3 Governor and Memory Updater

The governor G_ψ maps value vectors to discrete actions:

- **IGNORE**: skip learning from this phase.
- **UPDATE**: update one chosen head.
- **SPAWN**: create a new head and train it.
- **MERGE**: attempt to merge two heads (compression).
- **PRUNE**: delete low-relevance heads.
- **QUERY**: in active learning, decide to request labels.

In early parts, G_ψ is a hand-designed rule (thresholds on consistency, mismatch, familiarity). Later, it is a tiny learned policy (linear or shallow MLP) trained with REINFORCE under episode-level rewards such as mean or minimum per-world accuracy.

The memory updater U_η applies these actions to the head bank, their buffers, and long-term statistics.

2.4 Time Scales of Feelings

A key unifying theme is that *feelings must have time scales*:

- **Fast** (per step/batch): confusion, uncertainty.
- **Medium** (per segment): competence, elegance-driven compression.
- **Slow** (across segments): relevance for forgetting and retirement decisions.

Fast signals decide whether to trust current data and whether to acquire more information. Medium signals decide when it is safe to merge or distill structure. Slow signals decide which worlds can be forgotten without harming long-term performance.

3 Experiments

All experiments use synthetic classification tasks in $\mathbb{R}^2$ with multiple hidden worlds and non-stationary streams. Unless stated otherwise, results are averaged over 20–30 episodes with different random worlds and phase orderings.

3.1 Part I: Emotion-Gated Learning and Experiment Planning

3.1.1 Experiment 1: Continual Learning with Noisy Phases

Setup. We construct three binary classification worlds in $\mathbb{R}^2$:

- For each world $w \in \{A, B, C\}$, sample a random linear separator (w, b).
- To generate data for world w, sample $x \sim \mathcal{N}(0, I_2)$ and set $y = \mathbf{1}[w^\top x + b > 0]$.

An episode has $T = 12$ phases. For each phase t:

- randomly choose $w_t \in \{A, B, C\}$,
- draw $N_t \in [100, 200]$ samples,
- apply label noise: clean (0% noise) for even t, noisy ($\approx 40\%$ flips) for odd t.

We compare:

- **Baseline**: a single logistic regression updated by SGD on every phase.
- **Emotion-gated**: same model, but for each phase D_t:
  1. train a scratch model on D_t and measure its train accuracy α_t,
  2. if $\alpha_t \geq 0.75$, update the main model on D_t; otherwise, IGNORE.

Metrics. For each episode we measure:

- *Online accuracy*: accuracy on each phase before updating, averaged over phases.

- *Final per-world accuracy*: after the last phase, accuracy on a large clean test set (e.g. 1000 samples) from each of A, B, C, then averaged.

Results. In representative runs with alternating clean/noisy phases and 40% label noise in noisy phases:

- Online accuracy is similar for both methods.
- Final per-world accuracy is modestly higher for the emotion-gated learner.

A schematic result:

Method	Online accuracy	Final per-world accuracy
Baseline (always update)	≈ 0.62	≈ 0.65
Emotion-gated (skip low-acc phases)	≈ 0.63	≈ 0.66

The improvement is small but robust: a tiny phase-consistency feeling that says "this feels like junk, do not learn" can slightly improve long-term stability under noise.

3.1.2 Experiment 2: Emotion-Guided Experiment Planning (EGEP)

Setup. We test an Emotion-Guided Experiment Planner (EGEP) in an active learning setting:

- A single linear world in $\mathbb{R}^2$ as above.
- An unlabeled pool U of 2000 points and a separate clean test set T of 2000 points.
- The learner starts with no labeled data and chooses which point in U to label at each step.

We compare:

- **Random querying**: choose a random point from U at each step.
- **Uncertainty-based querying (EGEP)**: at each step, evaluate the current model on all of U, compute predicted probabilities, and pick the point whose predicted probability is closest to 0.5 (maximum uncertainty).

We remove queried points from U and track test accuracy on T after k labels for $k \in \{2, 5, 10, 20\}$.

Results. In repeated runs we observe the classic active-learning pattern:

Method	2 labels	5 labels	10 labels	20 labels
Random querying	≈ 0.70	≈ 0.79	≈ 0.85	≈ 0.90
EGEP (uncertainty)	≈ 0.76	≈ 0.86	≈ 0.93	≈ 0.94

For small label budgets ($k \leq 10$), uncertainty-based querying attains noticeably higher accuracy; as k grows, the gap shrinks as random querying eventually explores the space. This shows that a simple uncertainty feeling is enough to make experiment planning more sample-efficient.

3.2 Part II: World Banks, Mismatch, and Learned Governors

3.2.1 Experiment 1: Multi-World EGWM vs Single Model

Setup. We consider $K = 3$ worlds, $T = 24$ phases per episode, $N = 200$ samples per phase, and alternating label noise levels $(0.05, 0.40, 0.05, 0.40, \dots)$. We compare:

- a single logistic regression updated on every phase, and
- a world bank governed by a basic EGWM gate (consistency, mismatch, familiarity).

Results. The single model typically reaches only 0.60–0.70 test accuracy per world and shows clear interference. EGWM-basic reliably discovers 2–3 worlds and yields per-world accuracies in the 0.94–0.97 range. A representative run of 20 episodes:

Method	World 1	World 2	World 3
Single model	0.68 ± 0.21	0.74 ± 0.18	0.64 ± 0.18
EGWM-basic	0.96 ± 0.04	0.96 ± 0.04	0.95 ± 0.06

EGWM-basic ends with an average of 2.5 ± 0.6 worlds. Moving from a single model to an emotion-gated world bank makes a qualitative difference: plasticity is routed to specialist models instead of being spread thin over incompatible regimes.

3.2.2 Experiment 2: Mismatch vs Disagreement as Discovery Signals

We ask: which feeling should trigger the creation of new worlds?

Disagreement-based spawning. A governor that uses high inter-world disagreement alone as a spawn signal rarely spawns more than one world and performs similarly to the single-model baseline (per-world accuracies around 0.65–0.70).

Mismatch-based spawning. EGWM gates instead use the mismatch $\text{mismatch}_t = \alpha_t - \text{fam}_t$: if a scratch model explains the phase much better than any existing world and the phase is self-consistent, we spawn a new world. This mismatch-based signal reliably constructs one specialist per hidden world and achieves the high accuracies seen in Experiment 1.

Takeaway. Internal disagreement is a useful curiosity signal, but in this setting it is a weak driver for structural change. The key discovery feeling is *mismatch* between data and the current best world.

3.2.3 Experiment 3: Spawn-Happy vs Conservative Gates

We sweep EGWM thresholds for:

- consistency (for IGNORE),
- familiarity and mismatch (for SPAWN).

With a fixed consistency threshold $\tau_{\text{cons}} = 0.75$ and mismatch threshold $\tau_{\text{mismatch}} = 0.1$:

- EGWM-basic with $\tau_{\text{fam}} = 0.7$ achieves mean per-world accuracy ≈ 0.95 and ends with ≈ 2.2 worlds on average.
- A *spawn-happy* gate with $\tau_{\text{fam}} = 0.8$ achieves mean per-world accuracy ≈ 0.97 and ends with ≈ 2.5 worlds, more frequently exactly three.

Over-segmenting the world initially and compressing later works better than being too conservative.

3.2.4 Experiment 4: Learned Governors for Accuracy and Fairness

We replace hand-coded gates with a learned governor G_ψ that sees v_t and chooses among {IGNORE, SPAWN, UPDA
We train G_ψ with REINFORCE over episodes of $T = 24$ phases, under:

- R_{mean}: mean per-world test accuracy at episode end,
- R_{min}: minimum per-world test accuracy ("no world left behind").

In both reward settings, the learned governor reaches performance comparable to EGWM-basic, with per-world accuracies in the 0.94–0.96 range, far above the single-model baseline. Under $R_{\min}$ it slightly improves the worst-world accuracy and reduces its variance.

A representative comparison (30 evaluation episodes):

Method	World 1	World 2	World 3
Single model	0.64 ± 0.28	0.68 ± 0.19	0.67 ± 0.26
EGWM-basic	0.96 ± 0.05	0.95 ± 0.09	0.95 ± 0.07
Learned gov ($R_{\min}$)	0.95 ± 0.03	0.96 ± 0.04	0.95 ± 0.05

When trained with elegance-weighted rewards (penalizing number of worlds and overlap), the controller often learns to do almost nothing: the easiest way to be "simple" is not to learn. This motivates Part III.

3.3 Part III: Grow–Compress and Elegance Annealing

3.3.1 Experiment 1: Grow–Compress in a 3-World Stationary Environment

Environment. We construct three hidden linear worlds:

- Worlds 0 and 1 share the *same* decision boundary but are centered on different Gaussians.
- World 2 has a different boundary and a different mean.

Each world w generates samples by:

1. drawing $x \sim \mathcal{N}(\mu_w, \sigma^2 I)$,
2. labeling via $y = \mathbf{1}[w_w^\top x + b_w \geq 0]$.

Growth phase. Using a confusion-based growth rule:

- start with one head,
- for each batch, compute the minimum loss over heads,
- if loss is above a threshold (all heads are confused), SPAWN a new head and train it,
- otherwise UPDATE the lowest-loss head.

In a typical run, growth alone yields 3 heads with overall accuracy ≈ 1.0 and overlap ≈ 0.2 (some redundant heads covering similar regions).

Compression phase. We perform constrained compression with a target accuracy A_{target} (e.g. 0.99) and small tolerance δ:

- consider merging pairs (i, j) into a candidate head trained on the union of their buffers,
- accept a merge only if global accuracy remains at least A_{target} and not worse than current accuracy minus δ.

After compression, in a typical run:

- 2 heads, overall accuracy still ≈ 1.0,
- per-world accuracy near 1.0,
- overlap near 0 (almost no redundant correct heads).

Because worlds 0 and 1 share the same boundary, the compressor merges their specialists into one head, while keeping a separate head for world 2.

3.3.2 Experiment 2: Non-Stationary 4-World Environment

We define four worlds:

- Worlds 0 and 1: same boundary, different regions.
- World 2: different boundary and mean.
- World 3: another distinct boundary and mean.

Training is organized in segments:

1. Segment 0: W0 only.
2. Segment 1: W1 only.
3. Segment 2: W2 only.
4. Segment 3: W0 again.
5. Segment 4: W3 new.
6. Segment 5: Mixed all (any world at each step).

At each segment end, we evaluate per-world accuracy.

We compare:

- **Monolith**: single logistic regression trained on the full stream.
- **Growth-only bank**.
- **Grow–compress bank** with fixed target A_{target}.

Results. The monolith exhibits classic catastrophic interference: new worlds overwrite old ones and it struggles to represent incompatible worlds with one boundary. The grow–compress bank:

- grows to 3–4 heads in early segments,
- maintains per-world accuracy typically ≥ 0.9 across segments,
- ends the mixed segment with $\sim$ 3–4 heads and near-1.0 accuracy with moderate overlap.

It remembers old worlds when they return, detects and isolates new ones, and keeps the number of heads bounded.

3.3.3 Experiment 3: Compression Aggressiveness and Trade-Offs

We vary A_{target}:

- **Strict**: $A_{\text{target}} = 0.99$.
- **Medium**: $A_{\text{target}} \approx 0.97$.
- **Loose**: lower targets that allow more aggressive merging.

Stricter compression yields fewer heads but risks under-specialization early; looser compression preserves performance but leaves more redundancy. Together these points trace a competence–elegance frontier.

3.3.4 Experiment 4: Elegance Annealing

To combine robustness and simplicity, we use an annealed schedule over six segments:

$$A_{\text{target}}^{(0..5)} = [0.95, 0.96, 0.97, 0.98, 0.99, 0.99].$$

Compression at the end of each segment uses the segment-specific target.

In a typical run:

- Early segments (W0 only, W1 only): 2–3 heads, overall accuracy $\approx$ 0.92–0.94, moderate overlap.
- Middle segments (W2 only, W0 again): 3 heads, accuracy $\approx$ 0.90–0.91, slightly higher overlap while multiple hypotheses coexist.

- Late segments (W3 new, Mixed all): 3 heads, overall accuracy ≈ 1.0, with a structured overlap pattern (e.g. one head covering both worlds 0 and 1).

The bank ends with three heads and nearly perfect accuracy on all worlds. Elegance annealing lets the system be redundant early (supporting exploration) and strict later (supporting compression).

3.4 Part IV: Multi-Time-Scale Feelings, Relevance, and Active Learning

3.4.1 Experiment 1: Relevance and Forgetting

We study relevance in a segmented life with "alive" and "dead" worlds:

- Heads track the last segment they were updated on (age).
- Some worlds reappear later (alive), others never return (dead).

We compare:

- **No-prune** upper bound.
- **Age-aware pruning**: delete heads not used for a fixed number of segments.
- **Reward-weighted relevance**: relevance combines age and cumulative importance/reward per head; low-relevance heads are pruned.

Age-only pruning tends to delete both alive-but-rare and truly dead worlds, forcing re-learning when important rare worlds return. Reward-weighted relevance:

- preserves high-importance rare worlds close to the no-prune upper bound,
- prunes heads tied to low-importance worlds,
- reduces the total number of heads by roughly 40% compared to no-prune.

3.4.2 Experiment 2: Feature Growth and Non-Linear Heads

Inspired by Oak, confusion can trigger not only new heads but new features:

- when loss remains high, candidate quadratic features are tested,
- if they significantly reduce loss, they are added to the feature set.

With feature growth plus grow–compress, EGWM achieves:

- improved per-world accuracy on under-represented and tricky worlds,
- fewer heads than a purely linear baseline with similar accuracy.

We then move to small neural heads (2–8–1 MLPs) on non-linear worlds generated by similar MLPs. With an always-query baseline and sufficient training, these heads approach 95–99% accuracy per world when capacity matches the task family.

3.4.3 Experiment 3: Information Seeking via Uncertainty

We treat uncertainty as a value channel for information seeking in single- and multi-world settings with unlabeled inputs.

We compare:

- **Always-query**: request every label and update.
- **Random querying**: query with a fixed probability.
- **Uncertainty-based querying**: query only when $u = \min(p, 1 - p)$ exceeds a threshold.

Across linear and non-linear toy worlds:

- Always-query attains the highest accuracy but uses all labels.
- At matched label budgets, uncertainty-based querying performs at least as well as random querying and often slightly better.

- In non-linear MLP experiments, uncertainty-based querying attains essentially the same accuracy as always-query while using roughly 18–25% of the labels.

3.4.4 Experiment 4: Combining Uncertainty and Relevance

In a multi-world setting with different world frequencies and importances:

- Inputs arrive unlabeled from a mixture of worlds.
- The agent uses uncertainty to decide when to query labels.
- It tracks relevance per head and prunes old, low-relevance heads.

Compared to:

- a no-prune, always-query upper bound, and
- an age-only prune with uncertainty,

the combined uncertainty + reward-weighted relevance:

- maintains high accuracy on important worlds,
- prunes unimportant structure aggressively,
- uses significantly fewer labels than always-query.

3.5 Part V: Limits of Single-Step Feelings for Routing

3.5.1 Baselines: Monolith and Oracle Multi-Head Bank

We define a challenging regime:

- three binary tasks (worlds) share the *same* input distribution in $\mathbb{R}^2$,
- each world defines its own labeling rule $y = f_i(x)$,
- worlds appear in alternating segments in a non-stationary schedule,
- the agent never sees world IDs, only (x_t, y_t).

As baselines:

- A **monolithic model** (logistic regression or small MLP) trained on the entire stream shows catastrophic interference: it typically performs well on one world (often the most recent) and poorly on others; overall accuracy is moderate but skewed.
- An **oracle 3-head world bank** with known world IDs during training (each sample updates only its correct head) achieves high per-world and overall accuracy, showing that capacity is sufficient.

3.5.2 Experiment 5: Learned Router with Per-Head Probabilities

We now train a router that must decide, at each step, which head to use and update, without world IDs.

At time t:

1. Each head h produces a probability $p_h(x_t)$.
2. Form a feature vector
$$F_t^{(1)} = [x_t, p_0(x_t), p_1(x_t), p_2(x_t)].$$
3. A small router network R_ψ maps $F_t^{(1)}$ to a distribution $\pi_\psi(h \mid F_t^{(1)})$ over heads.
4. Choose head a_t (sampling or greedy), route (x_t, y_t) to it, compute loss, update that head.
5. Update ψ via policy gradient or supervised feedback based on the correctness of the chosen head.

The router has relatively rich *local* information but only single-step context.

Empirically, this router reduces interference slightly compared to the monolith but remains far below the oracle multi-head performance:

- it tends to allocate most samples to one or two heads that do reasonably well on the majority worlds,
- it fails to carve out clean specialization where each head focuses on a distinct world.

3.5.3 Experiment 6: Adding Value Channels

We augment R_ψ with value channels. For each head h, we maintain:

- competence (EMA of recent correctness),
- loss EMA,
- volatility (EMA of change in loss EMA).

At time t, for each head we compute a tuple of:

$$(x_t, p_h(x_t), \text{competence}_h, \text{lossEMA}_h, \text{volatility}_h),$$

concatenate across heads, and feed this into the router.

Despite seeing richer statistics reminiscent of the feelings used for growth and compression, the router still fails to approach oracle performance and behaves similarly to Experiment 5.

3.5.4 Experiment 7: Teacher-Trained Router with Perfect Experts

Finally, we construct *perfect experts*:

- Train three heads, each on one world with world IDs, until they achieve near-perfect accuracy.
- Freeze these heads.
- Train a router to predict, given $(x_t, \{p_h(x_t)\})$, which head would minimize loss under the true world.

Even with these conditions, a router that only sees single-step input features and per-head probabilities does not approach the oracle performance and behaves much like the monolithic baseline.

Conclusion of Part V. In a multi-world environment with shared input distribution and different labeling rules, a router that sees only single-step feelings and input features cannot reliably recover the latent world identity, even when:

- a bank of experts could, in principle, represent all worlds perfectly, and
- the router is trained with strong teacher signals.

This is a genuine negative result for EGWM as currently instantiated: the feelings that suffice for growth, compression, forgetting, and querying are not sufficient for world identification in this regime. It strongly suggests the need for a slower, persistent context code z_t that tracks which world the agent currently believes it inhabits, with feelings acting over longer temporal horizons.

4 Discussion

Across these experiments, several themes emerge:

Capacity first, then feelings. No amount of clever value signals can compensate for insufficient representational capacity. Linear heads on curved worlds plateau around 60–70% accuracy regardless of how well we manage spawning, merging, and forgetting. Once we move to heads that match the task class (quadratic features or small MLPs), the same value signals suddenly become powerful: they can gate learning, compress structure, schedule queries, and still converge.

Feelings as structural control. Feelings are most powerful when they govern *structure*:

- when to grow new worlds,
- when to merge or compress existing ones,
- when to retire obsolete structure,
- when to ask for more information.

Using them purely as scalar learning rates or penalties (e.g. elegance in the loss) often leads to collapse or inaction.

Time-structured value signals. Fast confusion and uncertainty support exploration, spawning, and data selection. Slow elegance and relevance support consolidation and forgetting. Elegance annealing—lenient early, strict late—is one concrete way to stage these signals over "developmental time."

Routing is harder than growth. The same feelings that successfully manage growth, compression, and forgetting do not automatically solve routing when worlds are latent and share inputs. This suggests that:

- growth/compression/forgetting can be governed by relatively local statistics;
- routing requires additional temporal and contextual machinery (e.g. latent state z_t).

5 Limitations and Future Work

This work has clear limitations:

- Experiments are restricted to low-dimensional synthetic worlds.
- Routing is studied in a simplified setting with small models and short horizons.
- Value channels are hand-designed; we have not shown how to discover them automatically.
- The proposed context code z_t is not implemented; Part V stops at the negative result that motivates it.

Future directions include:

- Implementing an explicit context latent z_t and testing whether longer-horizon feelings can solve the routing problem.
- Scaling EGWM ideas to larger models by realizing heads as adapters or experts in a shared backbone.
- Learning value channels end-to-end, instead of hand-crafting them.
- Applying EGWM as a meta-control loop around pretrained models, deciding when to adapt, which experts to update, and how to curate "trusted" buffers over a long lifetime.

Acknowledgements

I would like to thank Deb and Emo.