Behavioral Competence Without Conceptual Structure: Probing Type Knowledge in a Neural Pokémon World Model

Anonymous Author(s)
Affiliation withheld for double-blind review

Abstract

World models, neural networks trained to predict environment dynamics from observation alone, are used as substrates for planning and interpretation that assume they represent the concepts governing the dynamics they predict. We test this assumption in a domain with a small, fully known concept set: Pokémon battles, governed by an 18-by-18 type-effectiveness chart. From 50,002 public replays we compile 1,709,908 transitions and train world models to predict hit-point change, knockout, and status from a state and a chosen move. They generalize well, surpassing a memorization baseline most clearly on knockout ranking. Probing reveals a dissociation: predicted damage tracks the type chart in aggregate, yet primary type is barely linearly decodable from the species embeddings (13 percent, versus 5.6 percent chance), and same-type species are scarcely closer than random pairs. A bilinear architecture matched to the concept does not fix this. A lightweight auxiliary type-classification loss, on labels for a quarter of the species, raises embedding type decoding to 96 percent and sharpens the recovered chart while slightly improving prediction. The gap is one of training incentive, not capacity. Predictive accuracy is not a proxy for representational fidelity, and sparse auxiliary supervision cheaply surfaces latent concepts the primary objective leaves unorganized.

1 Introduction

World models are neural networks trained to predict the dynamics of an environment from observations alone. They have become central to reinforcement learning, model-based planning, and the study of how learned systems represent the worlds they operate in [Ha and Schmidhuber, 2018; Hafner *et al.*, 2024], and are increasingly used as substrates for planning, counterfactual reasoning, and mechanistic interpretability research [Schrittwieser and others, 2020; Chawla *et al.*, 2026]. Each use rests on an implicit assumption: that a model trained to predict an environment's dynamics also encodes the latent concepts that govern them. Whether predictive competence implies representational competence within the model remains an open empirical question.

This paper poses a specific version of this question. Consider a domain whose dynamics are governed by a small, discrete, and fully known set of abstract concepts. If a neural world model is trained on observed transitions from that domain alone, with no privileged access to the concept set, does it learn those concepts as identifiable features of its internal representation? Or does it instead learn distributed, behavior-equivalent shortcuts such as pair-specific interactions that approximate concepts in aggregate without ever surfacing them as a clean representational axis?

Pokémon battles offer a near-ideal testbed for this question. The dynamics of battles depend heavily on an 18-by-18 type-effectiveness chart with entries fully documented and discrete. Large-scale, natural-play battle data is available through the Pokémon Showdown simulator [Pokémon Showdown, nd], from which we collect 50,002 Generation 9 Random Battle replays comprising 1,709,908 (state, action, outcome) transitions. The action space is small enough to permit compact models on a single machine, yet predicting hit-point change, knockout, and status from a state and a chosen move remains highly nontrivial. Critically, the type chart is a near-perfect natural target for a probing study.

We train a family of world models, comparing them against memorization and constant-prediction baselines, and probe their behavior and their embeddings for evidence of type-effectiveness structure. Our probing protocol uses three converging diagnostics commonly used in the literature: an ensemble type-chart reconstruction that aggregates predictions over multiple representative species per type, a linear probe on the learned species embeddings recovering primary type, and a cosine-similarity gap between same-type and cross-type species pairs [Alain and Bengio, 2018; Hewitt and Manning, 2019; Belinkov and Glass, 2019]. We then conduct two tests. The first is architectural, applying a bilinear interaction model that gives the network an inductive bias matched to the bilinear (attacker-type $\times$ defender-type) structure of type effectiveness. The second is supervisory, a lightweight auxiliary type-classification loss trained on labels for only 25% of the Pokémon species, with the unsupervised damage objective and main heads unchanged [Caruana, 1997; Suddarth and Kergosien, 2005; Liebel and Körner, 2018].

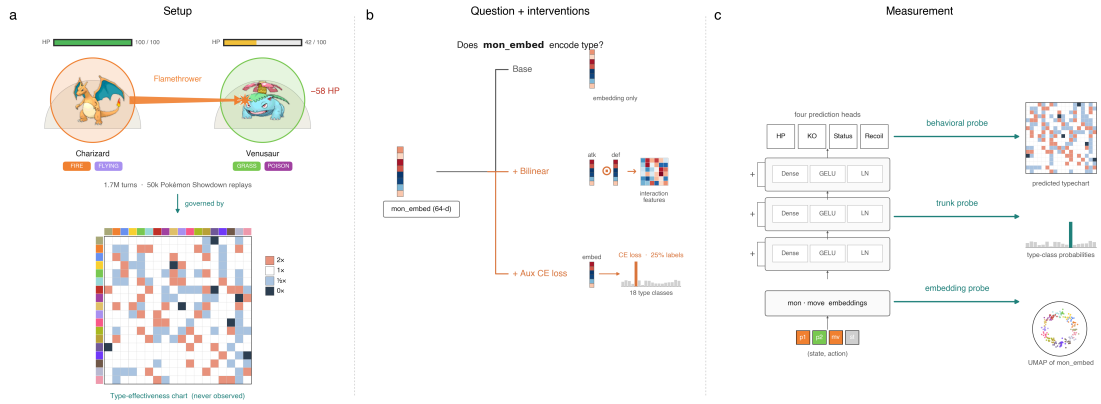

Figure 1: Experimental framework for evaluating whether predictive world models internalize abstract game structure. (Left) A neutral transition model is trained on 1.7M Pokémon battle transitions from 50k public Showdown replays to predict turn-level outcomes. (Middle) We ask whether type knowledge emerges implicitly in the learned representation, and whether it can be encouraged through architectural inductive bias or partial auxiliary supervision. (Right) We evaluate learned structure using three complementary diagnostics: behavioral probes based on predicted damage patterns, linear probes on learned Pokémon embeddings, and probes on intermediate trunk activations.

The contributions of this work are:

- We compile a 1.7-million transition Pokémon battle dataset, derived from 50,002 Generation 9 Random Battle replays, with a documented parsing pipeline, vocabularies, and reproducible splits suitable for both world-modeling and probing research.
- We introduce a probing protocol for measuring conceptual structure in world-model representations, combining behavioral type-chart reconstruction with embedding probes, trunk-activation probes, and cosine-gap analysis.
- We present an intervention study establishing that representational structure in a world model is shaped more by training signal than by inductive bias matched to the concept's ground-truth structure.
- We observe that lightweight auxiliary supervision can surface latent concepts in a world model's representation while simultaneously improving its main predictive objective, evidence that representational fidelity and predictive performance are not in tension, and that cheap supervisory signals may be a practical mechanism for making world models more interpretable and accurate.

2 Related Work

Our paper sits at the intersection of four lines of work: learned world models, probing the contents of neural representations, auxiliary loss and multi-task training, and game environments as testbeds for machine learning.

2.1 World Models and Learned Dynamics

The phrase "world model" traces back to [Ha and Schmidhuber, 2018], which showed that an unsupervised forward-prediction model of a visual environment is rich enough that a policy trained inside its hallucinations transfers back to the real environment. The Dreamer line extended this to standard RL benchmarks. DreamerV2 was the first agent to reach human-level Atari performance while learning entirely from rollouts in a discrete latent world model [Hafner *et al.*, 2022]. MuZero further showed that a learned model can plan even when it never reconstructs observations, predicting only the quantities planning consumes (reward, value, policy) [Schrittwieser and others, 2020].

These works, and those that follow, evaluate world models by what an agent can do with them, such as rollout fidelity, downstream return, and sample efficiency. They do not ask whether the model has internalized the abstract structure of the environment it predicts.

2.2 Probing Learned Representations

The methodological backbone of our analysis comes from the probing literature. Alain and Bengio [Alain and Bengio, 2018] introduced the linear classifier probe as a tool for tracking how separable a target concept is in intermediate layers of a trained model. Hewitt and Manning [Hewitt and Manning, 2019] further sharpen this idea into the structural probe, showing that an entire parse tree is recoverable via a linear transformation of BERT's hidden states, proving that syntactic structure is present within the model's vector geometry. The survey in [Belinkov and Glass, 2019] covers this whole family of methods and warns that recovered concepts from probes only inform you that the concept is decodable, not that the model uses it.

Recent works in mechanistic interpretability have also asked whether models trained on synthetic worlds build geometric representations of those worlds. The Othello-GPT work trained a transformer on Othello move sequences and found that nonlinear probes recovered the board state [Li *et al.*, 2023]. A follow-up study revisited the same model and showed that the representation is actually linear once you probe the correct quantity, in their case "my colour" versus "opponent's colour" [Nanda *et al.*, 2023]. Park, Choe, and

Veitch [Park *et al.*, 2023] formalize this linear-representation view for language models and link it to probing and steering. Our work sits on the same axis. The concept that we probe for is not recoverable from the base model's embeddings, even though the model predicts the dynamics that depend on it.

2.3 Auxiliary Losses and Multitask Learning

Suddarth and Kergosien [Suddarth and Kergosien, 2005] first showed that adding hint outputs that bias a network toward a desired internal representation can flatten the loss landscape and remove pathological minima. Caruana [Caruana, 1997] also states that training a shared representation against multiple related targets acts as an inductive bias, often improving generalization on each task individually. The literature shows how auxiliary signals can influence what intermediate layers encode, not just what the output layer predicts. Our work builds on this by using the auxiliary loss as a diagnostic tool to test whether the representational gap we observe is one of capacity or one of incentive.

2.4 Game Environments as ML Testbeds

Games have long served as ML benchmarks: Atari with DQN [Mnih and others, 2015], StarCraft II with AlphaStar [Vinyals and others, 2019], Dota 2 with OpenAI Five [Berner and others, 2019], and Minecraft with VPT [Baker and others, 2022], all hard control problems with large state spaces, partial observability, and long horizons.

Prior machine learning works specifically on Pokémon Showdown, our testbed, have similar motivations. Huang and Lee [Huang and Lee, 2019] trained a self-play policy-optimization agent on Generation 7 Random Battles, reaching competitive ladder play with reward shaping but no search. More recently, Grigsby et al. [Grigsby *et al.*, 2025] showed that an offline transformer trained on a decade of Showdown battle logs is able to rank in the top 10% among active players from human gameplay alone. Both approaches treat Pokémon as a hard RL problem.

Our motivation differs in that we are interested in the dynamics that govern the game, generated by a small set of fully documented abstract rules: the 18×18 type chart that every player has memorized. Most RL benchmarks have either unknown ground-truth (Atari, Minecraft) or a structure that is too large to enumerate (Go, StarCraft). Pokémon gives us the unusual combination of a large-scale natural-play dataset and a ground-truth concept set small enough to validate whether the model is able to learn it.

3 Methodology

3.1 Dataset

We construct our dataset from public replays on the Pokémon Showdown battle simulator [Pokémon Showdown, nd], using the `gen9randombattle` format. Random Battle assigns each player a randomized team of six species drawn from a curated pool, which removes team-building as a confounding variable while preserving the full turn-level dynamics of battle. We collect 50,002 replay logs from the public replay archive.

Each log is a sequence of pipe-delimited protocol messages. We parse every log into a set of turn-level transitions, where one transition is a (state, action, outcome) triple. A state records the two active species, their normalized hit points in $[0, 1]$, their volatile status (one of none, burn, paralysis, poison, toxic, freeze, sleep), the five stat-stage modifiers per side in $[-6, 6]$, the active weather (one of none, sun, rain, sand, snow), and an indicator of which player is acting. The action is the move chosen by the acting player. We parse only the acting player's chosen move and not the opponent's simultaneous choice, keeping the prediction target unambiguous. The outcome consists of four quantities computed from the state difference across the turn: the change in the defender's hit points (negative for damage dealt), a binary knockout indicator, the status condition inflicted this turn if any, and the recoil or drain to the acting species' own hit points. Any incomplete or incompatible turns are dropped.

The resulting dataset consists of 1,709,908 transitions across 49,623 battles. Knockouts occurred on 19.9% of transitions, a status condition is applied on 4.0%, and 35.2% of transitions deal no damage. We build vocabularies of 626 species, 362 moves, seven status classes, and five weather states.

Each transition is encoded into two embedding indices, where two species and the moves are indexed separately, and a 32-dimensional scalar feature vector. The scalar vector concatenates the acting-player indicator, both normalized hit-point values, two seven-way status one-hot encodings, the ten stat-stage modifiers divided by six to lie in $[-1, 1]$, and a five-way weather one-hot encoding. The hit-point change and recoil target are clipped to $[-1, 1]$.

The dataset is split 80/10/10 by battle identifier, shuffling battle identifiers with a fixed seed of 42 before partitioning. Splitting by battle rather than by transition prevents states from the same battle leaking across the train, validation, and test sets. The held-out test set contains 170,016 transitions.

3.2 Models

All four models share the same input interface and output structure, which isolates the effect of each architectural change. Two embedding tables are shared across the family: a species embedding of dimension 64 and a move embedding of dimension 32. For a given transition, the two species embeddings, the move embedding, and the 32-dimensional scalar vector are concatenated into a 192-dimensional input. Every model terminates in four prediction heads applied to a 256-dimensional trunk output: a hit-point head (linear, regression target), a knockout head (linear followed by a sigmoid), a status head (linear, seven-way logits), and a recoil head (linear, regression target). The models differ only in the trunk that maps the 192-dimensional input to the 256-dimensional representation, and in the case of the type-aware model, in an auxiliary head described in Section 3.7. Table 1 summarizes the family.

3.3 Training

All models are implemented in PyTorch [Paszke and others, 2019] and trained for 10 epochs with a batch size of 1024. We optimize with AdamW [Loshchilov and Hutter, 2017] at

Table 1: Model Family

Model	Parameters	Trunk
MLP	549k	Three dense layers
ResidualMLP	2.39M	Four residual blocks at width 512
Bilinear	2.47M	ResidualMLP plus attacker/defender and move/defender features
TypeAware	2.40M	ResidualMLP plus an 18-way type head trained with an auxiliary loss

a learning rate of 3×10^{-4} and a weight decay of 10^{-5}, and anneal the learning rate to zero over all training steps with a cosine schedule [Loshchilov and Hutter, 2016]. The training objective is a weighted sum of the four task losses, where mean squared error is used on the hit-point-change and recoil targets, binary cross-entropy is used for the knockout probability, and cross-entropy for the seven-way status target. We retain the checkpoint with the lowest validation hit-point mean absolute error. To support reproducibility, we seed the Python, NumPy, and PyTorch random number generators and the data-loader shuffle with a fixed seed of 42.

3.4 Baselines

We compare against two non-neural baselines that share the models' input and output interface. The MeanBaseline returns the training-set marginal statistics, namely the mean hit-point change, the knockout rate, the status-class distribution, and the mean recoil, for every input. It represents the no-learning floor. The TypeLookupBaseline stores, for each move/defending species pair seen in training, the mean of each outcome over that pair, and falls back to the MeanBaseline statistics for pairs absent from training. This baseline is the pure-memorization ceiling and a meaningful upper bar for type-aware behavior without abstraction: a move and a defending species jointly determine the type-effectiveness multiplier, so the lookup table implicitly encodes the type chart for every matchup it observed. On the test set, the lookup table covers 96.36% of transitions.

3.5 Metrics

For the hit-point target, we report mean absolute error over all turns and over damage-only turns, the latter restricted to transitions with a non-zero hit-point change; the damage-only figure is the more informative one, because the 35.2% of zero-damage turns deflate the overall error. For knockouts, we report the area under the ROC curve and the average precision. For status, we report the macro-averaged F1 over the seven classes and a binary "any-status" F1 that collapses the target to status-applied versus none. We retain status accuracy for continuity, but do not rely on it, since the roughly 96% "none" majority makes it uninformative. All metrics are computed with scikit-learn.

3.6 Probing Protocol

To measure whether a trained model represents type effectiveness, we use a protocol of converging diagnostics that probe both behavior and representation. Type labels for the probes come from a hand-curated set of 158 species (about 25% of the vocabulary) annotated with their primary and, when applicable, secondary type.

Single-Probe Type Chart

For each of the 18×18 ordered type pairs, we construct one synthetic state in which the attacker is a single-type representative species of the attacking type using a same-type-attack-bonus move of that type, the defender is a single-type representative of the defending type, and the context is neutral (full hit points, no status, no stat changes, no weather). We read the predicted hit-point change for all 324 cells, convert it to relative damage by negating the prediction and dividing each attacker's row by its median, and assign each cell to a $2\times$, $1\times$, $\frac{1}{2}\times$, or $0\times$ bucket. We compare the recovered chart to the canonical Generation VI type chart using Spearman rank correlation, computed both per attacker row and over the flattened chart, and using bucket accuracy.

Ensemble Type Chart

The single-probe chart is sensitive to per-species confounds such as base power and bulk. We therefore repeat the procedure over a pool of three to four single-type representative species and two same-type-attack-bonus moves per type, averaging the predicted hit-point change over every (attacker species, attacker move, defender species) combination in each cell. This ensemble diagnostic is our recommended behavioral probe.

Embedding Linear Probe

To test whether type is linearly decodable from the representation, we fit a logistic-regression probe [Alain and Bengio, 2018] from the 64-dimensional species embedding to the primary type over the 158 labeled species. We evaluate with stratified k-fold cross-validation, where k is the smaller of five and the least-populated class count, and standardize features within each fold. We report cross-validated accuracy against the chance rate of $1/18 \approx 5.6\%$. As a decodability ceiling, we additionally fit a non-linear probe, a two-layer perceptron with hidden widths 128 and 64, under the same cross-validation.

Trunk-Activation Probe

The same linear and non-linear probes are applied to the 256-dimensional trunk output rather than the embedding. We obtain the trunk activation for each labeled species under a fixed reference context, placing the species as the attacker using Body Slam against a Snorlax defender at full hit points with no status or stat changes. This probe asks whether the trunk constructs type-like features on the fly, even when the raw embedding does not expose them.

Cosine-Gap Analysis

As a geometry-level check, we compute the mean pairwise cosine similarity between same-type and different-type species pairs over the normalized embeddings, and report the gap between the two means. We report two variants: pairs

that share at least one type, using both primary and secondary types, and the stricter case of pairs that are both single-typed and of the same type.

For qualitative analysis, we project the species embeddings to two dimensions with UMAP [McInnes *et al.*, 2018] (15 neighbors, minimum distance 0.15) and color points by primary type.

3.7 Intervention

We test two mechanisms for surfacing type structure in the representation, one architectural and one supervisory. The architectural intervention is the Bilinear model of Section 3.2. Type effectiveness is a bilinear function of the attacker and defender types, so we supply the network with a matching inductive bias in the form of explicit attacker$\odot$defender and move$\odot$defender interaction features. This intervention changes the trunk's input but leaves the training signal unchanged. The supervisory intervention is an auxiliary type-classification loss applied to the TypeAware model. The auxiliary head maps each active species' embedding to 18 type logits, and we supervise it with cross-entropy against the primary type of both active species. Labels are available only for the 158 annotated species; unlabeled species are assigned an ignore index so that they contribute no auxiliary gradient. The auxiliary term, $0.5 \cdot (\mathrm{CE}_{p1} + \mathrm{CE}_{p2})$, is added to the combined objective of Section 3.3, where CE_{p1} and CE_{p2} are the cross-entropies for the two active species. Because the head attaches directly to the species embedding and leaves the trunk and the four prediction heads untouched, this intervention alters only the training signal reaching the embedding, which lets us separate a question of representational capacity from one of training incentive [Caruana, 1997].

4 Results

This section presents four findings: the world models generalize beyond memorization (Section 4.1); the strongest baseline behaves consistently with type effectiveness while encoding no clean representation of it (Section 4.2); a matched architectural prior fails to induce such a representation (Section 4.3); and a lightweight supervisory signal induces it almost completely, without degrading the main task (Section 4.4).

4.1 Generalization

Table 2 reports test-set performance over 170,016 transitions. All four neural models surpass both baselines on every informative metric. The largest margin occurs on knockout average precision, which increases from 0.409 for the TypeLookupBaseline to 0.830 for the residual model, which is evidence of generalization. Because the lookup table stores the complete training-set statistics for each (move, defender) pair, it cannot be outperformed on a previously observed matchup through recall alone; the neural models' superior ranking therefore indicates that they recover structure the table does not represent.

The damage-only hit-point error is consistent with this interpretation. It stabilizes near 0.12 across all model variants, and increasing capacity from the 549k-parameter MLP to the 2.4M-parameter residual family yields only a 7 to 9 percent relative reduction. We attribute this plateau to an irreducible noise floor: Random Battle does not expose individual values, effort values, abilities, or held items, none of which are recorded in the replay logs, so the precise damage roll is underdetermined by the observable state. Status macro-F1 remains near 0.58 for all models, marginally above the lookup baseline, a consequence of the 4.0 percent base rate of status application.

4.2 The Behavior-Representation Dissociation

We then assess whether the ResidualMLP internalizes type effectiveness with three diagnostics that yield consistent conclusions. The behavioral probe recovers a weak but promising signal. The ensemble type chart attains a Spearman correlation of 0.43 with the canonical chart and 64.8 percent bucket accuracy. The single-representative variant is substantially weaker, at 0.24 and 59.6 percent, respectively, indicating that per-species confounds such as base power and bulk dominate any individual reading and that aggregation over multiple representatives is required to expose the type signal.

The representational probes recover little. A linear probe mapping the 64-dimensional species embedding to primary type attains only 13.3 percent accuracy against a chance rate of 5.6 percent, demonstrating that type is barely linearly decodable from the embedding. The equivalent probe applied to the 256-dimensional trunk activations attains 23.5 percent, approximately twice the embedding result, which indicates that the trunk constructs type-like features under a fixed context but does not propagate them back into the embedding.

The embedding geometry further supports these findings. The mean cosine-similarity gap between same-type and different-type species pairs is only $+0.010$, so same-type species are scarcely closer than randomly selected pairs. The embedding is nonetheless richly structured: its pairwise cosine similarities exhibit a standard deviation of approximately 0.12 over the range $[-0.53, +0.52]$. The representation is therefore organized around factors other than type, most plausibly competitive role and base-stat total.

The model thus does not encode type as an explicit feature. It instead learns distributed, pair-specific damage interactions that approximate type effectiveness only in aggregate. Predictive behavior and internal representations are dissociated.

4.3 The Architectural Null

The first intervention supplies the network with a prior matched to the concept. Because type effectiveness is bilinear in attacker and defender type, the Bilinear model augments the trunk input with explicit attacker-defender and move-defender interaction features while leaving the training signal unchanged. Were a matched prior sufficient to surface type, the representational probes would be expected to improve.

However, no such improvement is observed. The ensemble correlation is unchanged within noise, declining from 0.43 to 0.41, and the embedding linear probe decreases from 13.3 to approximately 7 percent. The cosine gap is essentially unchanged, and main-task hit-point error increases slightly from 0.0994 to 0.1025 (Table 2). The most plausible mechanism is that the bilinear pathway performs the type computation

Table 2: Results (Test Split, $n = 170{,}016$)

Model	Params	HP MAE	HP MAE (dmg)	KO AUROC	KO AP	Status Macro-F1	Status Any-F1
MeanBaseline	—	0.2296	0.2138	0.5000	0.1993	0.1399	0.0000
TypeLookupBaseline	—	0.1379	0.1771	0.7723	0.4086	0.5500	0.6645
MLP	549k	0.1097	0.1274	0.9446	0.8097	0.5775	0.6956
ResidualMLP	2.39M	0.0994	0.1175	0.9496	0.8304	0.5757	0.7089
Bilinear	2.47M	0.1025	0.1207	0.9461	0.8204	0.5733	0.7018
TypeAware	2.40M	0.0987	0.1171	0.9498	0.8299	0.5790	0.7146

downstream of the embedding, thereby removing the optimization pressure that would otherwise drive type information into the embedding. A prior matched to the ground-truth structure of the concept is thus insufficient, and in this case harmful. We report this outcome as an informative null result that isolates the training signal as the operative variable in the subsequent experiment.

4.4 The Supervisory Intervention

The second intervention modifies only the signal reaching the embedding. The TypeAware model introduces an auxiliary cross-entropy loss, $0.5 \cdot (\mathrm{CE}_{p1} + \mathrm{CE}_{p2})$, mapping each active species' embedding to its primary type, supervised on labels for 158 species, approximately 25 percent of the vocabulary, with unlabeled species assigned an ignore index. The trunk and the four prediction heads remain unchanged. The effect is substantial (Table 3). The embedding linear probe increases from 13.3 to 95.6 percent, approximately 17 times the chance rate. The same-type cosine gap increases tenfold to $+0.109$, and the stricter pure-single-type gap increases fifteenfold to $+0.204$. The behavioral probe also improves, with the ensemble correlation rising from 0.43 to 0.51 and bucket accuracy from 64.8 to 66.0 percent. These gains incur no cost to the primary objective; test hit-point error improves marginally, from 0.0994 to 0.0987 (Table 2).

Two observations clarify the result. First, the auxiliary gradient reaches only the embedding, yet the trunk-activation probe doubles from 23.5 to 50.0 percent. Because the trunk receives no auxiliary signal, this gain is inherited from the reorganized embedding rather than learned independently. Second, a loss of weight 0.5 applied to a quarter of the vocabulary suffices to move the embedding probe from near-chance to near-perfect accuracy, which establishes that the embedding possessed the capacity to represent type throughout training. The unsupervised objective alone provided no incentive to express it. The representational gap documented in Section 4.2 is therefore one of incentive rather than capacity. Figure 2 illustrates this transition. A UMAP projection of the species embeddings exhibits no type organization for the ResidualMLP and clean, well-separated type clusters for the TypeAware model. Figure 3 presents the corresponding behavioral view, with the recovered ensemble type charts for both models positioned alongside the ground-truth chart.

Table 3: Intervention Study

Diagnostic	ResidualMLP	Bilinear	TypeAware
Embedding linear probe	13.3%	≈7%	95.6%
Trunk-activation linear probe	23.5%	—	50.0%
Cosine gap (share-a-type)	$+0.010$	≈unch.	$+0.109$
Cosine gap (pure single-type)	$+0.014$	—	$+0.204$
Typechart, ensemble ρ	0.43	0.41	0.51
Typechart, ensemble bucket acc.	64.8%	—	66.0%
Test HP MAE (main task)	0.0994	0.1025	0.0987

5 Discussion

5.1 Competence Without Representation

The results separate two properties usually assumed to coincide: predictive competence on a task and representation of the concept that governs it. The base model predicts damage well and tracks the type chart in aggregate, yet type is barely decodable from its embeddings (Section 4.2). The aggregate behavioral signal is produced by distributed, pair-specific interactions rather than an abstracted type feature. This inverts the standard caution from the probing literature, under which a successful probe shows that a concept is decodable but not that the model uses it [Alain and Bengio, 2018; Belinkov and Glass, 2019]. Here a model demonstrably uses a concept it does not cleanly represent.

The two interventions identify what controls whether the concept is represented. A bilinear prior matched to the concept's structure fails to surface it (Section 4.3), whereas a sparse auxiliary loss surfaces it almost completely with the trunk and heads untouched (Section 4.4). Architecture is therefore not the operative variable; the training signal is. That so small a supervisory signal moves the probe from near-chance to near-perfect shows the embedding had the capacity to represent type throughout training (Table 3) and that the unsupervised objective simply gave it no incentive to. A negative probing result on a world model should be read as a statement about training incentive, not representational capacity. Used this way, an auxiliary loss is a diagnostic that

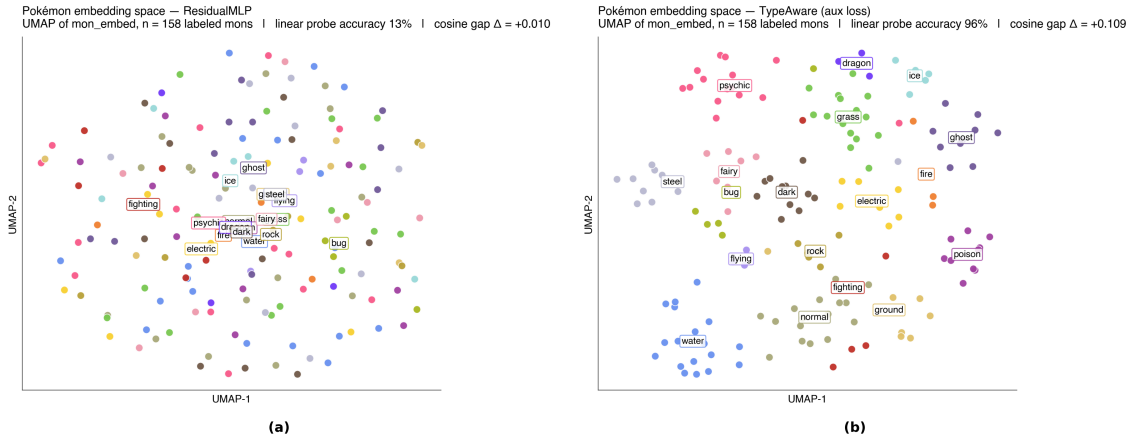

Figure 2: UMAP projection of the species embeddings, colored by primary type. (a) ResidualMLP: no apparent type structure (type-probe accuracy 13%). (b) TypeAware: the auxiliary loss yields clean type clusters (type-probe accuracy 96%).

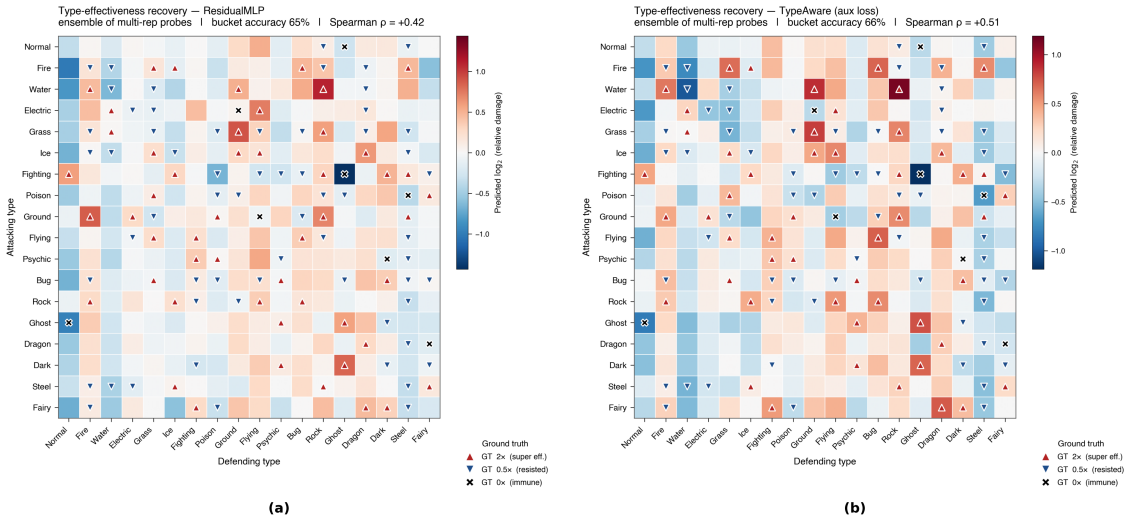

Figure 3: Recovered type charts: predicted relative damage for all 18×18 type pairs (red = super-effective, blue = resisted), ground truth overlaid as markers (▲ 2×, ▼ 0.5×, × 0×). (a) ResidualMLP (ensemble $\rho = 0.43$). (b) TypeAware ($\rho = 0.51$). The auxiliary loss sharpens agreement with the true chart.

separates the two, consistent with earlier evidence that auxiliary signals shape what intermediate layers encode [Caruana, 1997; Suddarth and Kergosien, 2005].

5.2 Implications

This pattern refines the linear representation hypothesis, under which concepts are encoded as linear directions [Park *et al.*, 2023]. In the Othello studies a world model's board state was first recovered only nonlinearly [Li *et al.*, 2023] and later shown to be linear in the correct frame [Nanda *et al.*, 2023]. Our case differs in mechanism rather than frame: type is not linearly decodable from the base embedding but becomes so after supervision, with architecture and data unchanged. Whether a concept comes to occupy a linear direction is thus dependent on the training signal, and such a direction can be installed cheaply and after the fact.

More broadly, these findings caution against treating prediction error as evidence that a world model has internalized the structure of its environment. World models serve as substrates for planning and interpretation, uses that assume the governing concepts are present [Ha and Schmidhuber, 2018; Schrittwieser and others, 2020; Hafner *et al.*, 2022]; yet a model can predict dynamics that depend on a concept while barely representing it. For applications that consume the representation rather than the outputs, predictive benchmarks will not reveal this gap, and a targeted auxiliary objective is a cheap way to close it.

5.3 Limitations

Several limitations bound these claims. The findings come from a single domain, chosen for its discrete, fully documented concept; generality requires cross-domain evidence. Random Battle hides individual values, effort values, abilities, and items, imposing a noise floor (about 0.12 damage-

only MAE) that limits how firmly the marginal main-task gain can be attributed to the reorganized embedding. The auxiliary weight (0.5) and label budget (about 25 percent) were fixed rather than swept, so robustness and the minimum effective supervision are unknown. Finally, we argue but do not demonstrate downstream transfer of the more legible embedding. A weight sweep, a label-budget ablation, and a transfer evaluation would each tighten the result.

6 Conclusion

A self-supervised world model trained on 1.7 million Pokémon battle transitions predicts damage well but does not represent type effectiveness in its embeddings. A bilinear prior matched to the concept's structure fails to surface it; a sparse auxiliary classification loss surfaces it almost completely while slightly improving prediction. The gap is therefore one of incentive, not capacity. Predictive accuracy is not a proxy for representational fidelity, and cheap auxiliary supervision is a practical way to make a world model's representations both accurate and legible.

References

[Alain and Bengio, 2018] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, November 2018. arXiv:1610.01644 [stat.ML].

[Baker and others, 2022] Bowen Baker et al. Video Pre-Training (VPT): Learning to Act by Watching Unlabeled Online Videos, 2022.

[Belinkov and Glass, 2019] Yonatan Belinkov and James Glass. Analysis Methods in Neural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 7:49–72, 2019.

[Berner and others, 2019] Christopher Berner et al. Dota 2 with Large Scale Deep Reinforcement Learning, 2019.

[Caruana, 1997] Rich Caruana. Multitask Learning. *Machine Learning*, 28(1):41–75, July 1997.

[Chawla *et al.*, 2026] Aviral Chawla, Galen Hall, and Juniper Lovato. MetaOthello: A Controlled Study of Multiple World Models in Transformers, February 2026. arXiv:2602.23164 [cs.LG].

[Grigsby *et al.*, 2025] Jake Grigsby, Yuqi Xie, Justin Sasek, Steven Zheng, and Yuke Zhu. Human-Level Competitive Pokémon via Scalable Offline Reinforcement Learning with Transformers, July 2025. arXiv:2504.04395 [cs.LG].

[Ha and Schmidhuber, 2018] David Ha and Jürgen Schmidhuber. World Models. March 2018. arXiv:1803.10122 [cs.LG].

[Hafner *et al.*, 2022] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with Discrete World Models, February 2022. arXiv:2010.02193 [cs.LG].

[Hafner *et al.*, 2024] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Domains through World Models, April 2024. arXiv:2301.04104 [cs.AI].

[Hewitt and Manning, 2019] John Hewitt and Christopher D. Manning. A Structural Probe for Finding Syntax in Word Representations. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[Huang and Lee, 2019] Dan Huang and Scott Lee. A Self-Play Policy Optimization Approach to Battling Pokémon. In *2019 IEEE Conference on Games (CoG)*, pages 1–4, London, United Kingdom, August 2019. IEEE.

[Li *et al.*, 2023] Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task, 2023.

[Liebel and Körner, 2018] Lukas Liebel and Marco Körner. Auxiliary Tasks in Multi-task Learning, 2018.

[Loshchilov and Hutter, 2016] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts, 2016.

[Loshchilov and Hutter, 2017] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, 2017.

[McInnes *et al.*, 2018] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, 2018.

[Mnih and others, 2015] Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[Nanda *et al.*, 2023] Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent Linear Representations in World Models of Self-Supervised Sequence Models, 2023.

[Park *et al.*, 2023] Kiho Park, Yo Joong Choe, and Victor Veitch. The Linear Representation Hypothesis and the Geometry of Large Language Models, 2023.

[Paszke and others, 2019] Adam Paszke et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library, 2019.

[Pokémon Showdown, nd] Pokémon Showdown. Pokémon Showdown! battle simulator, n.d.

[Schrittwieser and others, 2020] Julian Schrittwieser et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, December 2020.

[Suddarth and Kergosien, 2005] S. C. Suddarth and Y. L. Kergosien. Rule-injection hints as a means of improving network performance and learning time. January 2005.

[Vinyals and others, 2019] Oriol Vinyals et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, November 2019.