REASONING OR PATTERN EXPLOITATION? MECHANISTIC INSIGHTS FROM RL-TRAINED LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are increasingly described as acquiring "reasoning" skills after reinforcement learning from human feedback (RLHF) or related alignment methods. Benchmark improvements are widely celebrated as progress toward higher-order reasoning. However, whether these gains reflect genuine structural reasoning or more superficial adaptations remains underexplored. In this work, we probe LLMs trained in a finite and exhaustively analyzable logical domain, namely **Tic-Tac-Toe**, and trace how internal representations evolve across reinforcement learning with Group Relative Policy Optimization (GRPO). Quantitatively, reinforcement learning improves models far more than supervised fine-tuning (SFT), yielding higher accuracy and robustness across prompt variations. Mechanistic interpretability, however, paints a different picture: features extracted with sparse autoencoders (SAEs) reveal that models primarily adapt to better extract and exploit information already explicit in the prompt, such as whose turn it is, game progression and board occupancy. By contrast, high-level concepts like board symmetries, strategic forks and guaranteed wins remain weakly represented, echoing concerns that reasoning benchmarks risk overstating abstraction. This tension between surface-level performance and deeper representational change suggests that RLHF-driven "reasoning" may be conflating task-specific updates with structural reasoning ability. Our contribution is three-fold: (i) a systematic interpretability pipeline **tracing representation dynamics for the first time** across RL training in LLMs; (ii) an extension of SAE-based feature discovery to hypothesis-driven testing in a finite logical domain; and (iii) **the first interpretability based demonstration** that reinforcement learning amplifies prompt-level feature use rather than developing higher-order (game) reasoning. These findings argue for interpretability-first evaluation of reasoning claims, aligning with broader calls to ground reasoning in mechanistic analysis.

1 INTRODUCTION

Large language models (LLMs) are frequently described as acquiring reasoning abilities once trained with reinforcement learning from human or AI feedback (RLHF, GRPO) (Guo et al., 2025; Tang et al., 2024b; Liao et al., 2025; Wang et al., 2025a). Gains on reasoning benchmarks are widely celebrated as evidence of higher-order cognitive skills (Li et al., 2025; Liu et al., 2025; Xu et al., 2025; Sun et al., 2025; Topsakal et al., 2024; Xie et al., 2025). However, whether these improvements correspond to genuine reasoning or instead reflect more superficial adaptations remains an open question. A growing body of work warns that LLM reasoning may be overstated: models can exploit shallow patterns, perform reward hacking, overfit to training data, or depend on distributional cues (Hua et al., 2024; Xie et al., 2024; Zhao et al., 2025; Wu et al., 2025). Paradoxes expose inconsistent behavior (Tang et al., 2024a), and empirical probes suggest that abstraction is fragile (Hazra et al., 2025; Toh et al., 2025; Cosentino & Shekkizhar, 2024).

For systematically evaluating reasoning capabilities, games can provide a structured domain. From Othello and Checkers to Go, grid-worlds and adversarial arenas, board games have long served as testbeds for reasoning (Nanda, 2022; He et al., 2024; Joshi et al., 2024; Todd et al., 2024; Gallotta et al., 2024; Spies et al., 2024; Dao & Vu, 2025; Chen et al., 2024). Studies of LLMs in these domains reveal a tension: models can track state and generate legal moves or actions, but often fail to capture higher-order concepts such as symmetries, forks or long-horizon threats (Yang et al., 2024; Wu et al., 2024; Zhang et al., 2024). This raises the question of whether reinforcement learning agents in games are really learning to reason, or simply extracting and recombining surface-level features.

Mechanistic interpretability offers a way to answer this question. Sparse autoencoders (SAEs) and related methods make it possible to identify feature circuits, track representation dynamics and gauge abstraction in models (Cunningham et al., 2023; Templeton et al., 2024; Marks et al., 2024; Galichin et al., 2025; Demircan et al., 2024; Guan et al., 2025; Paulo et al., 2024; Molinari et al., 2024; Muhamed et al., 2024; Karvonen et al., 2024). Prior work has shown both the promise of monosemantic feature identification and the challenges of superposition, suppression, and compositionality (Elhage et al., 2023; Nanda et al., 2022; Foote & Bricken, 2024; Foote, 2023; Bricken, 2023; Marks, 2024). Empirical studies of model world representations suggest that LLMs do encode

structured features (Li et al., 2023; Hendel et al., 2023; Todd et al., 2023; Gurnee & Tegmark, 2023; Belrose, 2023; Engels et al., 2024; Burns et al., 2022; Kadavath et al., 2022), but whether these support genuine reasoning remains debated (Venhoff et al., 2025; Wang et al., 2025b; Ma et al., 2025).

In this work, we combine reinforcement learning with interpretability to investigate reasoning in small LLMs on the closed logical domain of Tic-Tac-Toe, also known as Noughts and Crosses. This simplicity of the game allows us to fully enumerate game states, rigorously control for symmetry, and distinguish between shallow features (turn order, board occupancy) and higher-order abstractions (strategic threats, forks, symmetries). We train models using supervised fine-tuning and GRPO, and probe their internal representations with SAEs trained on generic corpora. By analyzing activations across training checkpoints, we trace how reinforcement learning changes the internal representations.

Our findings are that reinforcement learning dramatically improves quantitative performance compared to supervised fine-tuning, consistent with prior reports (Dang & Ngo, 2025; Srivastava et al., 2025). However, mechanistic analysis shows that the improvements arise from stronger encoding of prompt-level features, not the development of strategic reasoning. In other words, models become better at exploiting what is already present in the input, rather than reasoning beyond it. This echoes broader critiques of benchmark-driven reasoning evaluation (Shipps, 2024; Anthropic Interpretability Team, 2024; Language Model Interpretability team, 2024; Hubinger, 2024; Reuel & Ma, 2024). Our study highlights the need for interpretability-first approaches to evaluating reasoning in language models.

2 Related Work

Reasoning in language models. Surveys consolidate the growing literature on reasoning in LLMs, ranging from symbolic logic and mathematics to multi-agent interaction and games (Li et al., 2025; Liu et al., 2025; Xu et al., 2025; Patil & Jadon, 2025; Sun et al., 2025; Hu et al., 2024; Gallotta et al., 2024; Zhang et al., 2024). Reinforcement learning, particularly GRPO and RLHF, is frequently credited for unlocking reasoning behaviors beyond supervised fine-tuning (Guo et al., 2025; Tang et al., 2024b; Dang & Ngo, 2025; Srivastava et al., 2025; Liao et al., 2025; Wang et al., 2025a). Empirical studies report improved benchmark scores on structured tasks such as grid worlds (Topsakal et al., 2024), mathematical reasoning (Shin, 2025), and game play (Xie et al., 2025; Wu et al., 2024; Yang et al., 2024). Yet critics emphasize that these benchmarks often reward task-specific shortcuts rather than structural reasoning (Hua et al., 2024; Xie et al., 2024; Zhao et al., 2025; Wu et al., 2025; Stechly et al., 2024). Failures in self-verification (Stechly et al., 2024), paradoxical responses (Tang et al., 2024a), and inconsistent abstraction (Hazra et al., 2025; Toh et al., 2025; Cosentino & Shekkizhar, 2024) highlight the fragility of reasoning claims.

Games as reasoning benchmarks. Games provide structured and interpretable testbeds where reasoning can be precisely defined. Early work on OthelloGPT demonstrated that transformer models can track state and legal moves while failing to generalize abstractions like board symmetries (Nanda, 2022; He et al., 2024). Similar tensions are observed in checkers (Joshi et al., 2024), Gomoku (Todd et al., 2024), and maze-solving tasks (Spies et al., 2024; Dao & Vu, 2025). Larger multi-agent environments, such as LLM-Arena (Chen et al., 2024), show that models adapt quickly to surface cues but lack deeper planning. Benchmark-driven studies suggest that LLMs are effective at tracking local features (turns, legalities, short-horizon tactics) but remain brittle when faced with higher-order logic such as forks, forced wins, or symmetry invariance (Zhang et al., 2024; Liao et al., 2025). This motivates mechanistic approaches that go beyond surface evaluation.

Mechanistic interpretability. Sparse autoencoders (SAEs) have become a central method for opening the black box of LLMs. They allow the discovery of monosemantic features (Cunningham et al., 2023; Templeton et al., 2024; Marks et al., 2024; Galichin et al., 2025; Demircan et al., 2024; Guan et al., 2025; Paulo et al., 2024; Molinari et al., 2024; Muhamed et al., 2024), provide insight into compositionality and superposition (Nanda et al., 2022; Elhage et al., 2023; Foote & Bricken, 2024; Foote, 2023; Bricken, 2023; Marks, 2024), and make it possible to trace how circuits evolve with training. These tools build on broader interpretability frameworks (Hubinger, 2024; Anthropic Interpretability Team, 2024; Language Model Interpretability team, 2024) and dictionary-learning approaches (Zhang et al., 2019; Faruqui et al., 2015; Karvonen et al., 2024). Applied to reasoning domains, SAEs reveal that LLMs often encode shallow patterns more readily than abstract structures (He et al., 2024; Spies et al., 2024). This raises the possibility that reinforcement learning amplifies prompt-level features without inducing genuine reasoning.

Model world representations. Beyond games, studies show that LLMs can learn structured internal models of text and environment. Context vectors, task vectors, and emergent representations highlight the ability of models to organize knowledge in ways resembling world models (Li et al., 2023; Hendel et al., 2023; Todd et al., 2023; Gurnee & Tegmark, 2023; Belrose, 2023; Engels et al., 2024; Burns et al., 2022; Kadavath et al., 2022; Liu et al., 2022; The AI Guide, 2023). Yet whether these representations enable reasoning or simply encode correlations is

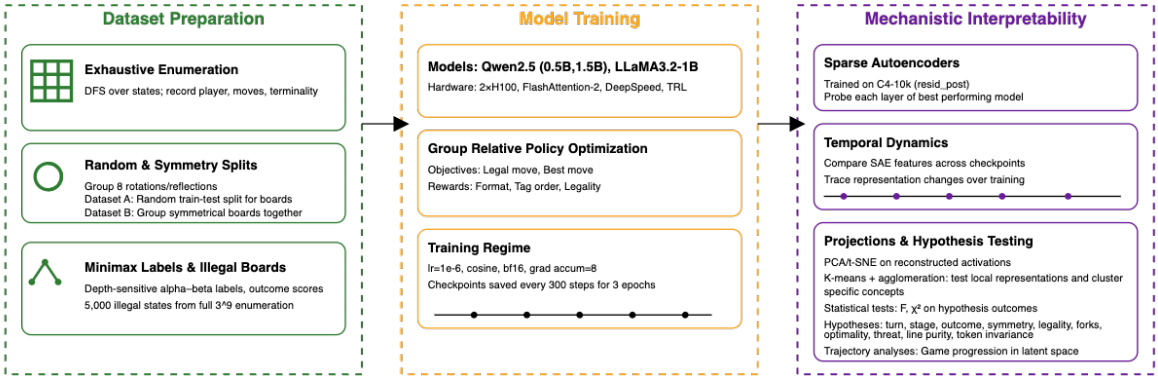


Figure 1: Overview of our experimental setup

actively debated (Venhoff et al., 2025; Wang et al., 2025b; Ma et al., 2025; Wu et al., 2025). Our work contributes to this debate by tracing how reinforcement learning reshapes world representations in a controlled logical setting.

Efficiency, ethics, and broader context. Scaling and efficiency advances (Wolf et al., 2019; Rasley et al., 2020; Dao, 2023; Hillier et al., 2024; Eldan & Li, 2023; Wang et al., 2025c) have made large-scale interpretability and RLHF experiments possible, while fairness and safety considerations (Reuel & Ma, 2024; Shipps, 2024) underscore the need for transparent reasoning evaluation. Our study builds on these insights by situating benchmark gains within mechanistic explanations, aligning with recent calls to ground reasoning claims in interpretable features rather than surface metrics (Anthropic Interpretability Team, 2024; Language Model Interpretability team, 2024).

3 Methodology

We study reasoning development under reinforcement learning in small LLMs using a controlled, fully interpretable environment: Tic-Tac-Toe. The methodology is designed to capture both surface-level task performance and the internal feature representations that emerge across training checkpoints.

3.1 Models and Training

Experiments used three models: Qwen2.5 0.5B Instruct, Qwen2.5 1.5B Instruct, and Llama3.2 1B Instruct. Models were trained with Group Relative Policy Optimization (GRPO) (Tang et al., 2024b; Guo et al., 2025). Supervised fine-tuning (SFT) was attempted as a baseline but failed, with models repeating the input prompt rather than learning valid continuations (Dang & Ngo, 2025; Srivastava et al., 2025).

Training was performed on 2×H100 GPUs with `Transformers reinforcement learning` library (von Werra et al., 2020), `flashattention-2` (Dao, 2023) and DeepSpeed (Rasley et al., 2020). Hyperparameters: learning rate 1×10^{-6}, cosine scheduler, bf16 precision, gradient accumulation 8. Checkpoints were saved every 300 steps up to 2240 steps (3 epochs).

3.2 Datasets

Two datasets were constructed:

- **Random split:** 80-10-10 train/validation/test partition over all legal Tic-Tac-Toe states.
- **Canonical symmetry split:** all eight symmetry variants of a board (rotations and reflections) were grouped by canonical ID. Each symmetry class was placed entirely in train or test, preventing symmetry leakage.

Generation. States were produced via exhaustive depth-first traversal from the empty board. Each state records: current player, legal moves, terminality, minimax-computed best moves, and canonical symmetry ID. Player moves were encoded as integers: 1–9 for X, 10–18 for O. Illegal boards (5,000) for hypothesis testing were added by enumerating all 3^9 possible states and discarding unreachable ones.

Dataset fields. Each entry includes: board (numeric and ASCII), natural language descriptions, move sequences, outcome labels, legal moves, best moves, and symmetry identifiers.

3.3 Objectives and Rewards

We trained models under two objectives:

1. Predict a legal move.
2. Predict a minimax-optimal move.

GRPO rewards combined: (i) legality, (ii) format compliance (`<think>...</think><answer>...</answer>`), (iii) single-tag correctness. Rewards were equally weighted. Best-move training was initialized either from scratch or from the best legal-move checkpoint.

3.4 Evaluation

At each checkpoint, we evaluated on both natural language and ASCII board formats, with robustness tested by substituting X/O with random characters. Metrics included:

- Accuracy.
- Outcome score (minimax evaluation from current perspective).
- Game phase accuracy (early, mid, late).
- Branching factor (legal move count).
- Best-move multiplicity.

Baseline models used structured generation: a chain-of-thought reasoning string followed by a move prediction.

3.5 Mechanistic Interpretability

To analyze internal representations, we used sparse autoencoders (SAEs) (Cunningham et al., 2023; Templeton et al., 2024; Marks et al., 2024; Galichin et al., 2025; Demircan et al., 2024; Guan et al., 2025). We adopt a similar approach to (Engels et al., 2024) for automated feature discovery. We train SAEs on a large-scale generic dataset (`NeelNanda/c4-10k`) to extract general-purpose features, and then applying hypothesis-driven testing in a controlled logical domain. The algorithm used can be found in the Appendix (Algo. 1). This design leverages the interpretability advantages of Tic-Tac-Toe, where hypotheses about symmetry, strategy, and game dynamics can be rigorously defined and tested.

For each layer of each model, we trained SAEs on activations (`resid_post`) using the configuration in Engels et al. (2024). Reconstructed activations were projected using PCA and t-SNE for visualization (Algo. 2). Board states in the projections were labeled along multiple axes: player turn, game stage, outcome, strategic situation (Algo. 3), symmetry group, legality, and correctness of model predictions. To test the models' capabilities on general board and game understanding, the representations of the illegal boards were compared with those of the legal boards used for training and testing. Similarly to quantitative evaluation, model representations were also tested for robustness using token invariance. The player tokens (X,Y) were replaced with another set of tokens (P and Q) to check if the models can generalize their developed representations to variances in prompt input without changes in the task setting. The complete set of algorithms for hypothesis testing can be found in the Appendix section A.

Then, hypothesis-based evaluations were conducted. This involved comparing the same projections across multiple hypotheses to identify potential concepts emerging in the model which were consistent with the updates of board representations across training. Clustering with k-means and hierarchical agglomeration was performed to automatically identify the feature groups, and statistical tests (F-tests, χ^2) evaluated the dependence between clusters and the previously defined game hypotheses. Manual analysis of the concepts and boards was done to identify higher level patterns.

4 Results

This study was designed to be broad in scope. Rather than relying on a handful of checkpoints or a single training split, we ran a wide set of experiments: three models (Qwen2.5 0.5B Instruct, Qwen2.5 1.5B Instruct, Llama3.2 1B Instruct), two training objectives (legal vs. best move), multiple input modalities (natural language, ASCII, random XY swaps, and their combinations), and dense checkpointing across full GRPO training runs. Each setting was evaluated quantitatively—tracking accuracy, robustness, and outcome-aware metrics—and mechanistically, through **sparse autoencoder (SAE)** reconstructions trained across all layers and checkpoints. On top of this, we ran a set of **hypothesis-driven probes** that are only possible in a controlled logical domain: turn identity, game progression, strategic situations, symmetry classes, legality, correctness, and automatically mined line-purity templates.

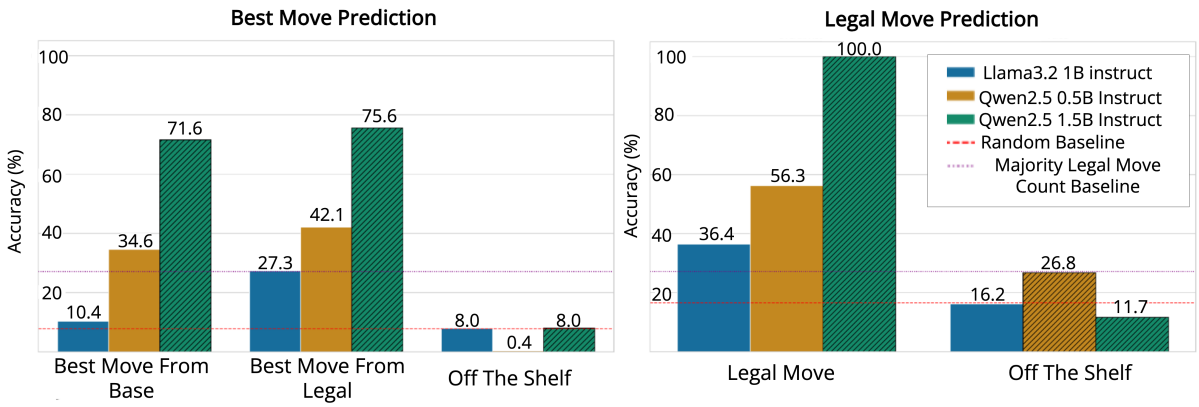


Figure 2: Best performance of each model and training setting for the legal move and best move prediction tasks on the random-split dataset. Complete performance plots for all settings (different board and move token representations) and datasets can be found in Appendix Figure 7.

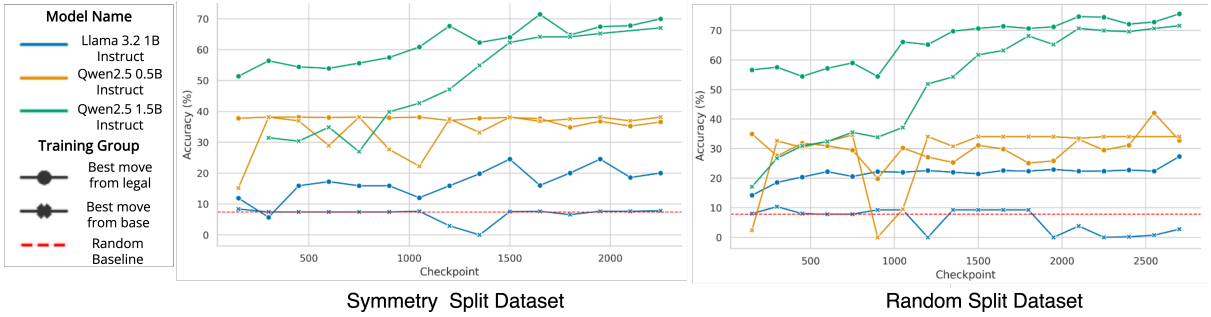


Figure 3: Progression of model performance across model checkpoints for the two datasets. Complete progression plots for all settings (different board and move token representations) can be found in Appendix Figure 6.

The result is a dataset of unusual granularity: millions of activations mapped, thousands of cluster-level tests, and dozens of full training-dynamics curves. What we report here are not isolated observations but patterns that hold consistently across models, scales, input formats, and interpretability probes.

Because of the sheer amount of material, we highlight here the clearest macroscopic findings: (i) scaling helps within limits, (ii) legal move prediction is far easier than best-move play, (iii) input modality strongly shapes generalization, (iv) symmetry remains unused, (v) checkpoint dynamics show preference for shallow cues, (vi) impact of rewards, and (vii) illegal boards expose pattern-matching over reasoning. The appendix contains the full training-dynamics maps, per-layer representational probes, and complete evaluation reports. We encourage readers to explore those figures: the richness of the dynamics is one of the main contributions of this work.

We begin with the quantitative results: performance curves across objectives, scales, and modalities, which sets the stage for how much reinforcement learning improves over supervised fine-tuning, and where it falls short. We then dive deeper with mechanistic analysis, showing how internal representations change under GRPO, which features are strengthened, and which never emerge. Together, these perspectives form a coherent picture: reinforcement learning amplifies prompt-level cues but leaves higher-order abstractions largely untouched.

4.1 Quantitative Results

Summary. Across three models, two objectives, two dataset splits, and four input modalities, GRPO post-training improves performance relative to off-the-shelf and SFT baselines. Gains are largest when inputs remain natural-language prompts. Performance drops with ASCII boards and degrades further when ASCII is combined with random XY remapping (analysis provided in Appendix section B). Symmetry-controlled curves closely track random-split curves (Fig. 3), indicating limited use of symmetry-aware structure.

Scaling helps within limits. Qwen2.5 1.5B achieves the strongest results, surpassing Llama3.2 1B and Qwen2.5 0.5B on both objectives at their best checkpoints (Fig. 2). The weaker models hover near smart baselines on best-move prediction, consistent with observations that small LMs often require targeted signals to form task circuits (Hillier et al., 2024; Eldan & Li, 2023; Wang et al., 2025c). Even the strongest model plateaus short of perfect best-

move play (legal: up to 100%; best move: 71% under symmetry split and 74% under random split), foreshadowing the representational limits seen mechanistically. Differences between the 0.5B Qwen and 1B Llama also suggest pretraining quality matters for downstream "reasoning" (Wang et al., 2025d).

Objective difficulty: accuracy on legal $\gg$ best. Legal-move prediction is substantially easier than best-move selection (Fig. 2), matching reports that models master surface constraints before long-horizon structure (Topsakal et al., 2024; Chen et al., 2024; Wu et al., 2024; Zhang et al., 2024). Outcome-aware analysis (full reports provided in appendix Fig. 8) show peaks on tactically imminent positions (large $|\text{score}|$) and dips on low-magnitude, ambiguous states where lookahead or symmetry reasoning should matter (Spies et al., 2024; Dao & Vu, 2025).

Input condition ablations reveal modality dependence. Overall results show that performance is *not* consistent across input modalities (See Appendix Fig. 7 for detailed results):

- **ASCII board representations** induce the **largest accuracy drop** for both legal and best-move objectives across all models, with the best performing Qwen2.5 1.5B model experiencing a relative 40% reduction in performance compared to natural language representations from 73 % to 44%. (We provide the full reports for all settings in appendix Fig. 7).
- **Random XY remapping** (symbol swap in NL prompts) causes a **smaller, consistent** decrease relative to the standard NL condition.
- The **combined** (ASCII + Random XY) setting yields the **sharpest degradation**, often approaching baseline.

These asymmetries indicate that GRPO training improved robustness to superficial *token* perturbations (e.g., X$\leftrightarrow$Y labels) but did *not* produce modality-invariant board understanding. The models rely on the natural-language scaffold to parse state; when forced to construct an internal spatial map from ASCII, accuracy collapses. Mechanistic findings in later sections support our findings. This pattern provides further evidence for reports that LLM "reasoning" is frequently entangled with input format and context cues rather than abstract structure (Hua et al., 2024; Wu et al., 2025; Stechly et al., 2024; Tang et al., 2024a; Cosentino & Shekkizhar, 2024).

Symmetry split has minimal impact on learning curves. Learning curves under symmetry-controlled splitting remain close to random-split curves (Fig. 3). If models learned symmetry-aware abstractions, canonicalization should change performance. Instead, results align with prior evidence that transformers often rely on surface regularities rather than equivariant structure in games (Nanda, 2022; He et al., 2024) and with reports that world-model features can remain local without explicit biases for invariances (Li et al., 2023; Gurnee & Tegmark, 2023; Liu et al., 2022).

Checkpoint dynamics. Over training, accuracy rises first on states with (i) few legal moves (low branching factor) and (ii) extreme outcome scores (forced wins/blocks). Gains arrive later and remain smaller on mid-game, high-branching states with multiple optimal continuations. (Detailed results can be found in Appendix Fig. 10). This is consistent with RL credit assignment favoring salient, short-horizon signals and with reports that many RLHF/GRPO improvements reflect stronger exploitation of prompt-level regularities rather than discovery of deep structure (Tang et al., 2024b; Guo et al., 2025; Xu et al., 2025; Li et al., 2025).

Format and control rewards work well. The format and tag-count rewards reliably enforce structured outputs (reasoning and answer blocks), but we see instances where a model produces fluent *rationales* while choosing suboptimal moves—another case of decoupled "explanation" from decision quality (Stechly et al., 2024; Shipps, 2024). This resonates with alignment results showing that preference optimization can shape surface behavior (style, format) more readily than internal competence (Tang et al., 2024b; Guo et al., 2025).

Legal-vs-illegal generalization and shallow cues. When evaluated on unreachable (illegal) boards, models frequently propose legal continuations that are locally sensible but globally incoherent, placing these states near legal clusters matched by simple line patterns rather than legality constraints (details can be found in Appendix Figs. 8, 15). This failure mode supports the thesis that the learned policies privilege *pattern exploitation* over *game-theoretic consistency*, a distinction also emphasized in recent small-model reasoning studies (Dang & Ngo, 2025; Srivastava et al., 2025; Shin, 2025).

4.2 Mechanistic Analysis

We group hypotheses into two classes: *Prompt-level* features explicitly available from the input (**H1** turn identity; **H2** game progression), and *high-level* game abstractions (**H3** strategic situation: must-block, guaranteed win, etc.; **H4** symmetry class; **H5** legality; **H6** best-move correctness; **H7** line-purity templates). Game progression bins

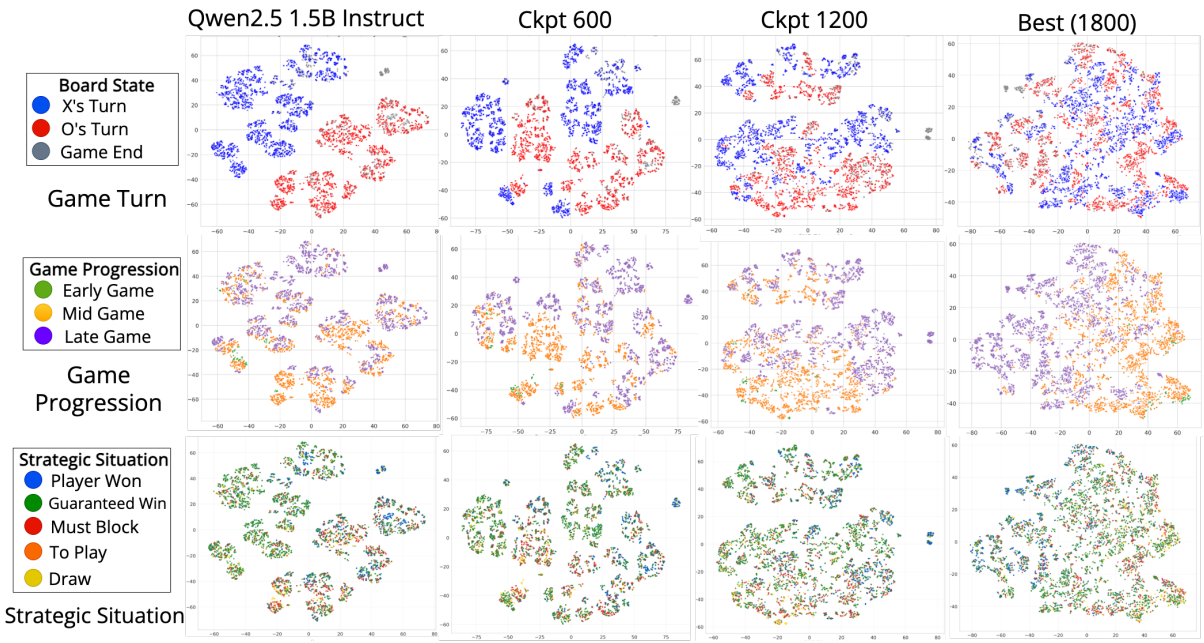


Figure 4: Model representations dynamics of Qwen2.5 1.5B (layer 12) through reasoning (GRPO) training. The figure represents the same projections colored based on three hypotheses: **(H1)** whose turn it is (Game Turn), **(H2)** which stage of the game it is (Game Progression) and **(H3)** what is the logical play for the given board (Strategic Situation). It is evident that while the base model only clearly represents the game turn, during the course of training, the model evolves to represents the boards based on game progression in addition to game turn, leading to improvement in performance. However, logic for strategic situations does not play a big role in model representations and no clear patterns emerge over the course of training. Full training dynamics across all layers and over all hypotheses tested are provided in Appendix Section C.

use move counts: early $= 0-3$, mid $= 4-6$, late $= 7-9$. The controlled domain allows hypothesis-driven tests that are hard to do in open-ended corpora (Nanda, 2022; He et al., 2024).

GRPO based "reasoning" strengthens separability for *H1/H2* (turn identity and progression) but yields weak, unstable representations for *H3/H4/H6* (strategic situation, symmetry, optimality) (Fig. 4). This matches our quantitative findings: large performance gains with GRPO arise predominantly from improved extraction and reuse of prompt-level information rather than from the formation of higher-order abstractions.

4.3 Macroscopic Representation Changes under GRPO

Fig. 4 shows how the internal representation changes over the course of training the best-performing model (Qwen2.5 1.5B), specifically for layer 12. The top, middle and bottom rows show different colorings of all the possible legal board states, with each coloring representing a hypothesis. For the top row, the boards are colored by **H1**: red and blue depending on whose turn it is, and grey when the game has ended. The middle rows colors boards by the **H2** (game progression: current turn number, out of the max possible 9 turns in a game). The bottom row colors boards by **H3** (strategic situation, e.g. whether a player has won already, is guaranteed a win no matter what, must block in order not to lose, etc.)

Turn and progression are the first-class axes. At initialization, clusters align most with **H1** (whose turn), which is a strong axis already present in base models. With GRPO, a second macroscopic axis for **H2** (game progression) emerges and stretches across the manifold (Fig. 4, middle row). This is consistent with reports that small LLMs first acquire representations tied to immediately accessible, local features (Spies et al., 2024; Li et al., 2023; Gurnee & Tegmark, 2023).

Strategic abstractions do not emerge. Coloring by **H3** (strategic situation, e.g. must block, guaranteed win, etc.) yields scattered speckles with no clean-colored clusters even late in training (Fig. 4, bottom row) when the model reaches peak performance. We see occasional micro-islands for *immediate* tactics (e.g., one-move wins) but these may correlate with structural organizations of the boards rather than true logical abstractions. There are no larger coherent sheets that would indicate a compact basis for long-horizon strategy or symmetry invariance. This mirrors Othello-style observations that legal tracking emerges before abstract invariances (Nanda, 2022; He et al.,

2024) and relates to geometric accounts of reasoning that distinguish shallow separators from global structure (Cosentino & Shekkizhar, 2024).

Input Structure matters: ASCII vs. natural language. The quantitative drop under ASCII boards is reflected mechanistically: SAE features for *tokenized NL boards* contain many crisp detectors aligned to lexical markers (*"Row 1", "empty"*) and ordinal positions, cleanly supporting **H1/H2** as seen in Figures 5 and 4 (middle row). When switched to ASCII, activations rotate; turn/progression axes remain but are noisier, and tactical micro-islands thin out. Interestingly, the macro patterns present even in the base model (such as game turn) for natural language representation of the boards are absent when the model is presented with the same boards but in ASCII representation (details are shown in Appendix Fig. 13. This suggests that reasoning training does not make the models robust to true "reasoning". The model does not develop true input-agnostic general reasoning for the task of playing Tic-Tac-Toe, but rather updates its internal representations to adapt to the input patterns which can help it achieve the highest reward for the training setting. This asymmetry aligns with context-dependence findings (Hua et al., 2024) and data-distribution sensitivity (Zhao et al., 2025), and helps explain why random XY remappings hurt slightly while ASCII hurts substantially in the overall plots.

4.4 Local Clusters in Model Representations

Using k-means followed by agglomerative refinement over SAE-PCA space, we automatically mine subclusters with high line-purity (dominant row/column/diagonal templates). We find sharp local pockets capturing structural patterns in the board such as "late-game, exactly one empty in R0" or "mid-game R0: `X . .`, center taken". Such board structure representations are inherent to the base model, and are retained in the trained models while the macroscopic arrangement of the boards evolves to include other prompt level features such as game progression.

Cluster-level statistical tests (ANOVA and χ^2) for each hypothesis on each cluster of each layer of each model checkpoint indicate a consistent pattern: strong dependence on piece count / openness / simple control features (center, corners), weak or no dependence on strategic classes and symmetry.

Mapping the boards by their symmetry group reveals that the model does not account or represent board symmetries in its latent representations (detailed results in Appendix Fig. 17). The ratio of boards belonging to unique symmetries to the total number of boards in each cluster (canonical symmetry ratio) drops below 0.7 in only 4 clusters (out of a total of 792 clusters) across both text and ASCII representations, all of which belong to end game clusters, which correlates more with the game progression based clustering of the boards rather than based on symmetries.

4.5 Layerwise locus of representational change

An important dimension of our findings is that the most substantial representational reorganization occurs in the *middle layers*, particularly around layer 12 in Qwen2.5-1.5B. Early layers (closer to embeddings) and later layers (closer to the output head) remain relatively stable across GRPO training: their representation geometries for hypotheses such as turn identity or game progression do not change appreciably compared to the base model. This is evident in the layerwise map of training dynamics provided in Appendix C. By contrast, middle layers show clear sharpening of prompt-level axes (turn, progression) and the emergence of local structural subclusters (line purity templates as seen in Fig. 5).

This aligns with prior work that early layers often specialize in lexical or surface encoding, while middle layers form reusable abstractions, and later layers map abstractions to task-specific outputs (Nanda, 2022; **?**; Elhage et al., 2023). In reinforcement learning fine-tuning, middle layers are also where preference-aligned features tend to emerge, with output layers primarily adjusting stylistic or formatting control (Tang et al., 2024b; Stechly et al., 2024). Sparse autoencoder studies likewise find that mid-layer dictionaries yield the most interpretable, monosemantic features (Cunningham et al., 2023; Templeton et al., 2024; Marks et al., 2024), whereas later layers contain highly entangled, task-specific mixtures that are harder to disentangle (Marks, 2024; Foote & Bricken, 2024).

Our results thus fit neatly into this emerging picture: GRPO updates primarily reshape mid-layer manifolds to better capture prompt-level structure (whose turn, how far along), while leaving early encoding and late decision mapping comparatively untouched. This helps explain why models gain robustness to prompt variation without forming new higher-order abstractions: the learning signal sharpens already-accessible mid-layer features rather than rewriting the global representational pipeline.

5 Conclusion

Methodologically, the contribution of this work is an interpretability-first pipeline that combines dense RL checkpointing with SAE feature discovery and hypothesis-driven testing. This work offers a tightly controlled look at

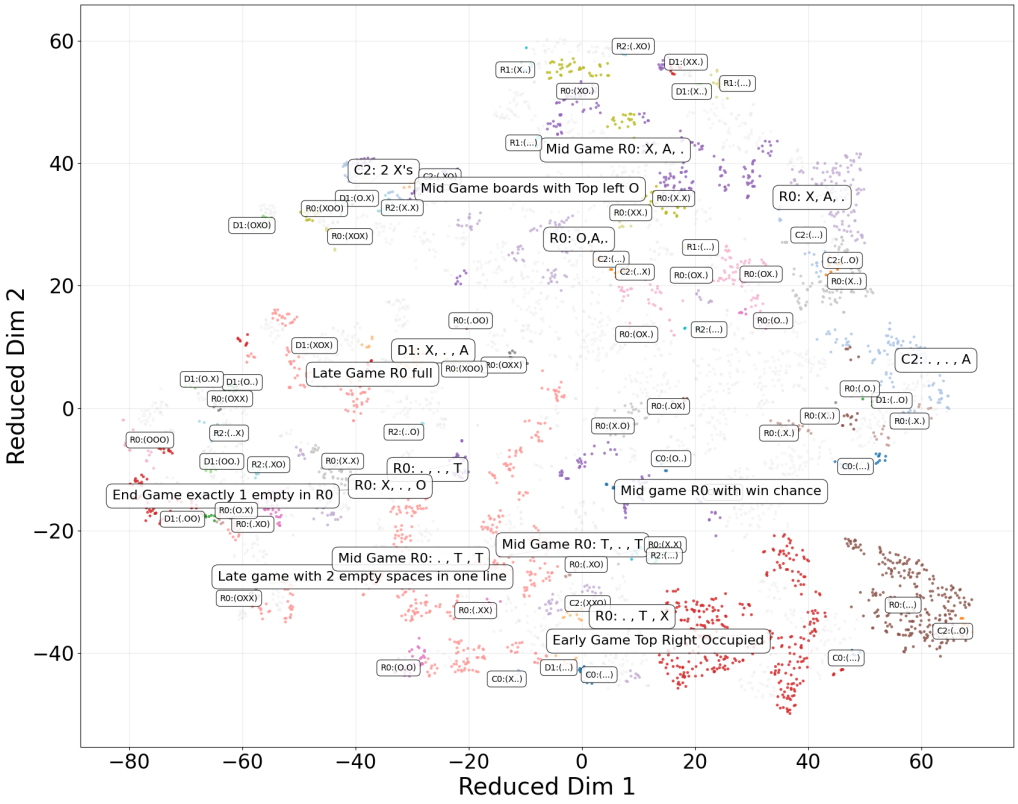


Figure 5: Clustering of board state representations based on structural line purity for Qwen2.5 1.5B (layer 12) on text representations of the boards. The labels represent the dominant line purity sub-clusters or the common line pattern across the board states. The smaller labels represent the automatically mined line subclusters within k-means clusters found through agglomerative clustering. For clarity, we include manually annotated clusters which combine automatically mined subclusters across different k-means clusters. This shows how common structural patterns are represented in local subclusters while the general representation of the boards follows the game progression as found in Figure 4. For example, late game board states which contain exactly one empty cell in the top row are all clustered together. In the labels, R, C and D represent rows, columns and diagonals respectively. O and X represent player tokens, "." represents empty, T represents taken (by either X or O) and A represents any of X, O or empty. The complete map of automatically mined line purity clusters for all layers and models across all representation modes and training checkpoints can be found in Appendix Figure 22.

what current "reasoning" focused reinforcement learning actually changes inside small LLMs. In Tic-Tac-Toe, GRPO training raises legal- and best-move accuracy relative to SFT and off-the-shelf baselines (Figs. 2 & 3). The largest gains arise when the input is represented in natural language; accuracy drops sharply with ASCII boards (Fig. 2). The symmetry-controlled dataset gave rise to similar results as the random split dataset, indicating that the models do not learn equivariant structure.

Mechanistic analysis explains these outcomes. SAE-based probes (Cunningham et al., 2023; Templeton et al., 2024; Marks et al., 2024; Galichin et al., 2025; Demircan et al., 2024; Guan et al., 2025) trained following automated discovery principles show that GRPO sharpens *prompt-level* representations (e.g. whose turn it is and game progression), but not high-level abstractions such as strategic situation (Fig. 4). The most substantial representational reorganization occurs in middle layers (e.g., layer 12), with early and late layers comparatively unchanged, consistent with reports that the most interpretable, reusable features often reside in mid-network dictionaries (Cunningham et al., 2023; Templeton et al., 2024). Local clusters reliably capture line-purity templates and other shallow geometric regularities (Fig. 5). These findings are consistent with prior work arguing that benchmark gains can overstate abstraction (Li et al., 2025; Liu et al., 2025; Xu et al., 2025; Shipps, 2024; Hua et al., 2024; Xie et al., 2024; Zhao et al., 2025; Stechly et al., 2024; Tang et al., 2024a; Hazra et al., 2025; Toh et al., 2025; Cosentino & Shekkizhar, 2024).

6 REPRODUCIBILITY STATEMENT.

We take reproducibility seriously and provide all ingredients to replicate our results. **Models, training, and hyperparameters** are specified in Sec. §3 (Models and Training), including the full GRPO configuration and compute setup (2×H100), with checkpointing frequency and evaluation protocol; per-setting learning curves and best-checkpoint summaries are reported in Figs. 3 and 2, with full curves in Appendix Fig. 6 and comprehensive modality results in Appendix Fig. 7. **Datasets and preprocessing** are described in Sec. §3 (Datasets), covering state-space enumeration, terminal detection, symmetry canonicalization, random vs. symmetry splits, and illegal-board generation; dataset fields and token mappings are enumerated in the same section. **Evaluation metrics and robustness settings** (NL/ASCII, random XY) are defined in Sec. §3 (Evaluation) with outcome-aware analyses summarized in Appendix Figs. 8–10. **Mechanistic interpretability** methodology—SAE training setup, layer hooks, and projection/cluster pipelines—is detailed in Sec. §3 (Mechanistic Interpretability) and the Appendix §C, with pseudocode-style algorithms in §A (Algorithms 1–7) and full hypothesis panels in Appendix Figs. 11–24. Supplemental submission contains: (i) scripts to regenerate datasets and splits, (ii) GRPO training/evaluation code and exact configs, (iii) SAE training configs and visualization scripts. The repository includes fixed random seeds, environment files, and instructions to reproduce the entire set of experiments. A polished repository with the complete code will be released upon acceptance for open source usage.

REFERENCES

Anthropic Interpretability Team. Engineering challenges in interpretability. https://www.anthropic.com/research/engineering-challenges-interpretability, June 2024. Accessed: 2025-09-13.

Nick Belrose. World models. https://lingo.csail.mit.edu/blog/world_models/, July 2023. Accessed: 2025-09-13.

Trenton Bricken. Do sparse autoencoders find "true" features? https://www.lesswrong.com/posts/QoR8noAB3Mp2KBA4B/do-sparse-autoencoders-find-true-features, September 2023. Accessed: 2025-09-13.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*, 2022.

Junzhe Chen, Xuming Hu, Shuodi Liu, Shiyu Huang, Wei-Wei Tu, Zhaofeng He, and Lijie Wen. Llmarena: Assessing capabilities of large language models in dynamic multi-agent environments. *arXiv preprint arXiv:2402.16499*, 2024.

Romain Cosentino and Sarath Shekkizhar. Reasoning in large language models: A geometric perspective. *arXiv preprint arXiv:2407.02678*, 2024.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

Quy-Anh Dang and Chris Ngo. Reinforcement learning for reasoning in small llms: What works and what doesn't. *arXiv preprint arXiv:2503.16219*, 2025.

Alan Dao and Dinh Bach Vu. Alphamaze: Enhancing large language models' spatial intelligence via grpo. *arXiv preprint arXiv:2502.14669*, 2025.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

Can Demircan, Tankred Saanum, Akshay K Jagadish, Marcel Binz, and Eric Schulz. Sparse autoencoders reveal temporal difference learning in large language models. *arXiv preprint arXiv:2410.01280*, 2024.

Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

Nelson Elhage, Neel Nanda, Catherine Joseph, Nicholas DasSarma, Tom Henighan, Tristan Hatfield-Dodds, Matthew Hume, Tamera Olsson, Thomas Rzad, Zico loving, Tom Johnson, Darcy Mckay, Saul Johnston, Chris Clark, Amanda Askell, Evan Hubinger, Andy Chen, Jamie Kernion, David Drain, Tom Mann, Yuntao Bai, Joseph Bou-Saba, Dawn Chan, Michael Jones, Ben Kerr, Kamal Ndousse, Andy Jones, Samuel Bowman, Anna Johnston, Zac Hatfield-Dodds, John Miller, Sam McCandlish, Chris Olah, and Jared Kaplan. Superposition, composition, and automatic circuits. https://transformer-circuits.pub/2023/superposition-composition/index.html, November 2023. Accessed: 2025-09-13.

Joshua Engels, Eric J Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language model features are one-dimensionally linear. *arXiv preprint arXiv:2405.14860*, 2024.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah Smith. Sparse overcomplete word vector representations. *arXiv preprint arXiv:1506.02004*, 2015.

Charlie Foote. Sparse autoencoders find composed features in small toy models. https://www.lesswrong.com/posts/a5wwqza2cY3W7L9cj/sparse-autoencoders-find-composed-features-in-small-toy, November 2023. Accessed: 2025-09-13.

Charlie Foote and Trenton Bricken. Addressing feature suppression in saes. https://www.lesswrong.com/posts/3JuSjTZyMzaSeTxKk/addressing-feature-suppression-in-saes, February 2024. Accessed: 2025-09-13.

Andrey Galichin, Alexey Dontsov, Polina Druzhinina, Anton Razzhigaev, Oleg Y Rogov, Elena Tutubalina, and Ivan Oseledets. I have covered all the bases here: Interpreting reasoning features in large language models via sparse autoencoders. *arXiv preprint arXiv:2503.18878*, 2025.

Roberto Gallotta, Graham Todd, Marvin Zammit, Sam Earle, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. Large language models and games: A survey and roadmap. *IEEE Transactions on Games*, 2024.

Haoxiang Guan, Jiyan He, and Jie Zhang. Sparse autoencoders reveal interpretable structure in small gene language models. *arXiv preprint arXiv:2507.07486*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Wes Gurnee and Max Tegmark. Language models represent space and time. *arXiv preprint arXiv:2310.02207*, 2023.

Rishi Hazra, Gabriele Venturato, Pedro Zuidberg Dos Martires, and Luc De Raedt. Have large language models learned to reason? a characterization via 3-sat phase transition. *ArXiv*, abs/2504.03930, 2025. URL https://api.semanticscholar.org/CorpusID:277621858.

Zhengfu He, Xuyang Ge, Qiong Tang, Tianxiang Sun, Qinyuan Cheng, and Xipeng Qiu. Dictionary learning improves patch-free circuit discovery in mechanistic interpretability: A case study on othello-gpt. *arXiv preprint arXiv:2402.12201*, 2024.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*, 2023.

Dylan Hillier, Leon Guertler, Cheston Tan, Palaash Agrawal, Chen Ruirui, and Bobby Cheng. Super tiny language models. *arXiv preprint arXiv:2405.14159*, 2024.

Sihao Hu, Tiansheng Huang, Gaowen Liu, Ramana Rao Kompella, Fatih Ilhan, Selim Furkan Tekin, Yichang Xu, Zachary Yahn, and Ling Liu. A survey on large language model-based game agents. *arXiv preprint arXiv:2404.02039*, 2024.

Wenyue Hua, Kaijie Zhu, Lingyao Li, Lizhou Fan, Shuhang Lin, Mingyu Jin, Haochen Xue, Zelong Li, JinDong Wang, and Yongfeng Zhang. Disentangling logic: The role of context in large language model reasoning capabilities. *arXiv preprint arXiv:2406.02787*, 2024.

Evan Hubinger. Toward a mathematical framework for computation in mechanistic interpretability. https://www.lesswrong.com/posts/2roZtSr5TGmLjXMnT/toward-a-mathematical-framework-for-computation-in, March 2024. Accessed: 2025-09-13.

Abhinav Joshi, Vaibhav Sharma, and Ashutosh Modi. Checkersgpt: Learning world models through language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pp. 576–588, 2024.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. *Advances in Neural Information Processing Systems*, 37: 83091–83118, 2024.

Language Model Interpretability team. Gemma scope: helping the safety community shed light on the inner workings of language models. https://deepmind.google/discover/blog/gemma-scope-helping-the-safety-community-shed-light-on-the-inner-workings-of-language July 2024. Accessed: 2024-08-01.

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. *ICLR*, 2023.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.

Yi Liao, Yu Gu, Yuan Sui, Zining Zhu, Yifan Lu, Guohua Tang, Zhongqian Sun, and Wei Yang. Think in games: Learning to reason in games via reinforcement learning with large language models. *arXiv preprint arXiv:2508.21365*, 2025.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Hanmeng Liu, Zhizhang Fu, Mengru Ding, Ruoxi Ning, Chaoli Zhang, Xiaozhang Liu, and Yue Zhang. Logical reasoning in large language models: A survey. *arXiv preprint arXiv:2502.09100*, 2025.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-reasoner: Advancing llm reasoning across all domains. *arXiv preprint arXiv:2505.14652*, 2025.

Ryan Marks. SAE reconstruction errors are empirically pathological. https://www.lesswrong.com/posts/rZPiuFxESMxCDHe4B/sae-reconstruction-errors-are-empirically-pathological, April 2024. Accessed: 2025-09-13.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*, 2024.

Marco Molinari, Victor Shao, Luca Imeneo, Mateusz Mikolajczak, Vladimir Tregubiak, Abhimanyu Pandey, and Sebastian Kuznetsov Ryder Torres Pereira. Interpretable company similarity with sparse autoencoders. *arXiv preprint arXiv:2412.02605*, 2024.

Aashiq Muhamed, Mona Diab, and Virginia Smith. Decoding dark matter: Specialized sparse autoencoders for interpreting rare concepts in foundation models. *arXiv preprint arXiv:2411.00743*, 2024.

Neel Nanda. Othello-gpt: A case study in mechanistic interpretability. https://www.neelnanda.io/mechanistic-interpretability/othello, October 2022. Accessed: 2025-09-13.

Neel Nanda, Joseph Cammarata, Leo Vaintrob, Tim Toubiana, Nelson Elhage, Zico McLeavey, Thomas andloving, Tamera Rzad, Morgan Pigott, and Nick McDougall. A toy model of superposition. https://transformer-circuits.pub/2022/toy_model/index.html, December 2022. Accessed: 2025-09-13.

Avinash Patil and Aryan Jadon. Advancing reasoning in large language models: Promising methods and approaches. *arXiv preprint arXiv:2502.03671*, 2025.

Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models. *arXiv preprint arXiv:2410.13928*, 2024.

Jeff Rasley, Samyam Rajbhandari, Rohan Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pp. 254–268, 2020.

Anka Reuel and Devin Ma. Fairness in reinforcement learning: A survey. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, volume 7, pp. 1218–1230, 2024.

Andrew Shin. Can a gamer train a mathematical reasoning model? *arXiv preprint arXiv:2506.08935*, 2025.

Alex Shipps. Reasoning skills of large language models are often overestimated. https://news.mit.edu/2024/reasoning-skills-large-language-models-often-overestimated-0711, July 2024. Accessed: 2024-08-01.

Alex F Spies, William Edwards, Michael I Ivanitskiy, Adrians Skapars, Tilman Räuker, Katsumi Inoue, Alessandra Russo, and Murray Shanahan. Transformers use causal world models in maze-solving tasks. *arXiv preprint arXiv:2412.11867*, 2024.

Gaurav Srivastava, Shuxiang Cao, and Xuan Wang. Towards reasoning ability of small language models. *arXiv preprint arXiv:2502.11569*, 2025.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*, 2024.

Haoran Sun, Yusen Wu, Yukun Cheng, and Xu Chu. Game theory meets large language models: A systematic survey. *arXiv preprint arXiv:2502.09053*, 2025.

Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. On the paradox of generalizable logical reasoning in large language models. *OpenReview*, 2024a.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*, 2024b.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

The AI Guide. LLMs and World Models, Part 2. https://aiguide.substack.com/p/llms-and-world-models-part-2, October 2023. Accessed: 2025-09-13.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*, 2023.

Graham Todd, Alexander G Padula, Matthew Stephenson, Éric Piette, Dennis J Soemers, and Julian Togelius. Gavel: Generating games via evolution and language models. *Advances in Neural Information Processing Systems*, 37:110723–110745, 2024.

Vernon YH Toh, Yew Ken Chia, Deepanway Ghosal, and Soujanya Poria. The jumping reasoning curve? tracking the evolution of reasoning performance in gpt-[n] and o-[n] models on multimodal puzzles. *arXiv preprint arXiv:2502.01081*, 2025.

Oguzhan Topsakal, Colby Jacob Edell, and Jackson Bailey Harper. Evaluating large language models with grid-based game competitions: an extensible llm benchmark and leaderboard. *arXiv preprint arXiv:2407.07796*, 2024.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*, 2025.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Haozhe Wang, Qixin Xu, Che Liu, Junhong Wu, Fangzhen Lin, and Wenhu Chen. Emergent hierarchical reasoning in llms through reinforcement learning. *arXiv preprint arXiv:2509.03646*, 2025a.

Shangshang Wang, Julian Asilis, Ömer Faruk Akgül, Enes Burak Bilgin, Ollie Liu, Deqing Fu, and Willie Neiswanger. Resa: Transparent reasoning models via saes. *arXiv preprint arXiv:2506.09967*, 2025b.

Shangshang Wang, Julian Asilis, Ömer Faruk Akgül, Enes Burak Bilgin, Ollie Liu, and Willie Neiswanger. Tina: Tiny reasoning models via lora. *arXiv preprint arXiv:2504.15777*, 2025c.

Xinyi Wang, Shawn Tan, Mingyu Jin, William Yang Wang, Rameswar Panda, and Yikang Shen. Do larger language models imply better reasoning? a pretraining scaling law for reasoning. *arXiv preprint arXiv:2504.03635*, 2025d.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Timothée Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Juncheng Wu, Sheng Liu, Haoqin Tu, Hang Yu, Xiaoke Huang, James Zou, Cihang Xie, and Yuyin Zhou. Knowledge or reasoning? a close look at how llms think across domains. *arXiv preprint arXiv:2506.02126*, 2025.

Shuang Wu, Liwen Zhu, Tao Yang, Shiwei Xu, Qiang Fu, Yang Wei, and Haobo Fu. Enhance reasoning for large language models in the game werewolf. *arXiv preprint arXiv:2402.02330*, 2024.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*, 2024.

Yunfei Xie, Yinsong Ma, Shiyi Lan, Alan Yuille, Junfei Xiao, and Chen Wei. Play to generalize: Learning to reason through game play. *arXiv preprint arXiv:2506.08011*, 2025.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025.

Yunhao Yang, Leonard Berthellemy, and Ufuk Topcu. Reasoning, memorization, and fine-tuning language models for non-cooperative games. *arXiv preprint arXiv:2410.14890*, 2024.

Juexiao Zhang, Yubei Chen, Brian Cheung, and Bruno A Olshausen. Word embedding visualization via dictionary learning. *arXiv preprint arXiv:1910.03833*, 2019.

Yadong Zhang, Shaoguang Mao, Tao Ge, Xun Wang, Adrian de Wynter, Yan Xia, Wenshan Wu, Ting Song, Man Lan, and Furu Wei. Llm as a mastermind: A survey of strategic reasoning with large language models. *arXiv preprint arXiv:2404.01230*, 2024.

Chengshuai Zhao, Zhen Tan, Pingchuan Ma, Dawei Li, Bohan Jiang, Yancheng Wang, Yingzhen Yang, and Huan Liu. Is chain-of-thought reasoning of llms a mirage? a data distribution lens. *arXiv preprint arXiv:2508.01191*, 2025.

A Algorithms for Hypothesis Testing

This section details the algorithms used for Sparse Autoencoder (SAE) feature discovery, board state analysis, and hypothesis testing.

Algorithm 1 SAE Feature Discovery and Clustering

Require: Language Model $\mathcal{M}$, Target Layer L
Ensure: Trained SAE $\mathcal{S}$, Feature Clusters $\mathcal{C}$

$\mathcal{S}_{path} \leftarrow$ Path to cached SAE for $\mathcal{M}, L$
if $\mathcal{S}_{path}$ exists **then**
    $\mathcal{S} \leftarrow \text{LoadSAE}(\mathcal{S}_{path})$
else
    $\mathcal{D}_{train} \leftarrow$ Load text corpus (e.g., C4-10k)
    $\mathcal{A}_{train} \leftarrow$ Extract activations from $\mathcal{M}$ at layer L on $\mathcal{D}_{train}$
    $\mathcal{S} \leftarrow \text{TrainSAE}(\mathcal{A}_{train})$
    $\text{SaveSAE}(\mathcal{S}, \mathcal{S}_{path})$
$W_{dec} \leftarrow \mathcal{S}$.decoder_weights
$W_{norm} \leftarrow W_{dec} / \|W_{dec}\|_2$ ▷ Normalize feature vectors
$Labels \leftarrow \text{SpectralClustering}(W_{norm})$
$\mathcal{C} \leftarrow$ Group feature indices by $Labels$
return $\mathcal{S}, \mathcal{C}$

Algorithm 2 t-SNE Dimensionality Reduction and Caching

Require: Model $\mathcal{M}$, SAE $\mathcal{S}$, Tic-Tac-Toe Dataset $\mathcal{D}$, Target Cluster Indices $\mathcal{I}_{cluster}$
Ensure: 2D Reduced Activations R, Associated Board Metadata B_{meta}

$Cache_{path} \leftarrow$ Path to cached t-SNE results for $\mathcal{M}$, layer, style
if $Cache_{path}$ exists **then**
    $R, B_{meta} \leftarrow$ LoadFromCache($Cache_{path}$)
    return R, B_{meta}
$B_{meta} \leftarrow$ Get unique boards from $\mathcal{D}$
$Prompts \leftarrow \{$GeneratePrompt(b) for $b \in B_{meta}\}$
$\mathcal{A}_{orig} \leftarrow$ Get activations from $\mathcal{M}$ for $Prompts$
$f_{acts} \leftarrow \mathcal{S}$.encode($\mathcal{A}_{orig}$)
mask $\leftarrow$ Zeros like f_{acts}
mask$[:, \mathcal{I}_{cluster}] \leftarrow 1$
$\mathcal{A}_{recon} \leftarrow \mathcal{S}$.decode($f_{acts} \odot$ mask) ▷ Filter with cluster features
$R \leftarrow$ t-SNE($\mathcal{A}_{recon}$, n_components $= 2$)
SaveToCache($Cache_{path}, R, B_{meta}$)
return R, B_{meta}

Algorithm 3 Game-Theoretic Strategic Situation Analysis

Require: Reduced Activations R, Board Metadata B_{meta}
Ensure: Saved plot colored by game-theoretic state

function EVALUATE($board, player$) ▷ Memoized function
    $winner, terminal \leftarrow$ CheckTerminal($board$)
    if $terminal$ **then**
        return 1 if $winner = player$, 0 if $draw$, -1 if $loss$
    $best\_outcome \leftarrow -2$ ▷ Losing is the default
    for move in LegalMoves($board$) **do**
        $next\_board \leftarrow$ ApplyMove($board, move, player$)
        $outcome \leftarrow$ Evaluate($next\_board$, opponent($player$))
        if $outcome = -1$ **then return** 1
            ▷ Opponent loss is a win for me
        $best\_outcome \leftarrow \max(best\_outcome, -outcome)$
    return $best\_outcome$
$Categories \leftarrow []$
for board b in B_{meta} **do**
    $winner, terminal \leftarrow$ CheckTerminal(b)
    if $terminal$ **then**
        Append "Player Won" or "Draw" to $Categories$
    else
        $gt\_eval \leftarrow$ Evaluate(b, CurrentPlayer(b))
        $has\_threat \leftarrow$ OpponentHasImmediateWin(b, CurrentPlayer(b))
        if $gt\_eval = 1$ **then**
            Append "Guaranteed Win"
        else if has_threat **then**
            Append "Must Block"
        else if $gt\_eval = 0$ **then**
            Append "Draw"
        else
            Append "To Play" ▷ Forced loss, no immediate threat
Plot R, coloring points by $Categories$. Save figure.

Algorithm 4 Cluster-Based Statistical Analysis

Require: Clustered Boards $\mathcal{C}_{boards}$ (map from cluster ID to board lists)
Ensure: Printed statistical test results
$ClusterFeatures \leftarrow \{\}$
for cluster cid in $\mathcal{C}_{boards}$ **do**
  $Features_{cid} \leftarrow []$
  for board b in $\mathcal{C}_{boards}[cid]$ **do**
    Append {'piece_count' : CountPieces(b), 'center' : GetCenter(b), ... } to $Features_{cid}$
  $ClusterFeatures[cid] \leftarrow Features_{cid}$
▷ Example for a continuous feature
$Data_{anova} \leftarrow$ [[f['piece_count'] for f in $ClusterFeatures[cid]$] for cid in $\mathcal{C}_{boards}$]
$F, p \leftarrow$ ANOVA($Data_{anova}$)
Print("Piece Count", F, p)
▷ Example for a categorical feature
$Table_{chi2} \leftarrow$ BuildContingencyTable('center', $ClusterFeatures$)
$\chi^2, p \leftarrow$ ChiSquaredTest($Table_{chi2}$)
Print("Center Control", χ^2, p)

Algorithm 5 Hybrid Hierarchical-Agglomerative Analysis

Require: Reduced Activations R, Board Metadata B_{meta}
Ensure: Saved hybrid plot visualization
$L0_{labels} \leftarrow$ KMeans($R, n\_clusters = 18$)
$L0_{analysis} \leftarrow$ AnalyzeClusters($L0_{labels}$) ▷ For L0 properties
$Map_{L0\_to\_Sub} \leftarrow \{\}$
for cluster cid from 0 to 17 **do**
  $L0_{indices} \leftarrow$ Indices where $L0_{labels} = cid$
  $R_{sub} \leftarrow R[L0_{indices}]$
  $k_{micro} \leftarrow \max(5, \lfloor |L0_{indices}|/10 \rfloor)$
  $Micro_{labels} \leftarrow$ KMeans($R_{sub}, n\_clusters = k_{micro}$)
  $PatternMap \leftarrow \{\}$ ▷ Map pattern key to list of global indices
  for micro-cluster $mcid$ in k_{micro} **do**
    $MC_{indices} \leftarrow$ Global indices for micro-cluster $mcid$
    purity, pattern_key $\leftarrow$ CalculateDominantLinePattern($B_{meta}[MC_{indices}]$)
    if purity $\geq$ THRESHOLD **then**
      Append $MC_{indices}$ to $PatternMap$[pattern_key]
  $Map_{L0\_to\_Sub}[cid] \leftarrow$ {MergeIndicesByPattern($PatternMap$)}
Visualize: For each L0 cluster, draw its convex hull. Inside, color and annotate each discovered pure sub-cluster from $Map_{L0\_to\_Sub}$ based on its defining line pattern.

Algorithm 6 Illegal vs. Legal Board Contrastive Analysis

Require: Model $\mathcal{M}$, SAE $\mathcal{S}$, Legal Dataset $\mathcal{D}_L$, Illegal Dataset $\mathcal{D}_I$
Ensure: Saved contrastive visualizations
$B_L \leftarrow$ Sample($\mathcal{D}_L, N_{legal}$)
$B_I \leftarrow$ Sample($\mathcal{D}_I, N_{illegal}$)
$B_{combined} \leftarrow []$
for b in B_L **do**
  Append $\{'board' : b, 'type' : 'legal'\}$ to $B_{combined}$
for b in B_I **do**
  Append $\{'board' : b, 'type' : 'illegal', 'reasons' : b.reasons\}$ to $B_{combined}$
$R \leftarrow$ Run t-SNE on SAE-reconstructed activations for $B_{combined}$ ▷ As in Alg. 2
Plot 1: Legality View
Plot R, coloring points blue if 'legal' and red if 'illegal'. Save figure.
Plot 2: Pattern Agglomeration View
$Labels_{kmeans} \leftarrow$ KMeans($R, n\_clusters = k$)
for cluster cid from 0 to $k-1$ **do**
  $Cluster_{indices} \leftarrow$ Indices where $Labels_{kmeans} = cid$
  $PatternStats \leftarrow$ FindDominantLinePatterns($B_{combined}[Cluster_{indices}]$)
  Annotate cluster centroid with top patterns, showing their legal/illegal counts.
Plot R colored by legality. Overlay cluster annotations and convex hulls. Save figure.

Algorithm 7 Causal Intervention via Activation Patching

Require: Model $\mathcal{M}$, SAE $\mathcal{S}$, Dataset $\mathcal{D}$, Target Cluster Indices $\mathcal{I}_{cluster}$
Ensure: Printed intervention success/failure reports

for square i from 0 to 8 **do**
  $b_{dirty}, b_{clean} \leftarrow$ FindBoardPair$(i, \mathcal{D})$ ▷ Boards differ only at square i
  if no pair found **then continue**
  $p_{dirty}, p_{clean} \leftarrow$ Prompt(b_{dirty}), Prompt(b_{clean})
  $a_{clean} \leftarrow$ GetActivation$(\mathcal{M}, p_{clean})$
  $f_{clean} \leftarrow \mathcal{S}$.encode$(a_{clean})$
  function PATCHHOOK($a_{dirty\_runtime}$)
    $f_{dirty} \leftarrow \mathcal{S}$.encode$(a_{dirty_runtime})$
    $f_{dirty}[\mathcal{I}_{cluster}] \leftarrow f_{clean}[\mathcal{I}_{cluster}]$ ▷ The patch
    return $\mathcal{S}$.decode(f_{dirty})
  $logits_{orig} \leftarrow \mathcal{M}(p_{dirty})$
  $logits_{patched} \leftarrow \mathcal{M}$.run_with_hooks$(p_{dirty}, \text{hook} = \text{PatchHook})$
  $move_{orig} \leftarrow$ GetMove$(logits_{orig})$
  $move_{patched} \leftarrow$ GetMove$(logits_{patched})$
  $move_{expected} \leftarrow$ GetBestMove(b_{clean})
  Print results comparing $move_{orig}, move_{patched}, move_{expected}$.

Algorithm 8 Depth-Sensitive Minimax Evaluation

Require: Board state B, current player P, current depth d
Ensure: Game-theoretic score s, best move index m

function MINIMAXGETSCORE(B, P, d)
  $winner, is\_terminal \leftarrow$ CheckWinner(B)
  if $is\_terminal$ **then**
    if $winner = 1$ **then return** $10 - d$, None ▷ Faster wins are better
    else if $winner = 2$ **then return** $-10 + d$, None ▷ Slower losses are better
    elsereturn 0, None ▷ Draw
  $EmptyCells \leftarrow$ FindEmptyCells(B)
  if $P = 1$ (Maximizing) **then**
    $max\_eval \leftarrow -\infty$, $best\_move \leftarrow$ None
    for move in $EmptyCells$ **do**
      $B_{new} \leftarrow$ ApplyMove$(B, move, P)$
      $evaluation, \_ \leftarrow$ MinimaxGetScore$(B_{new}$, Player 2, $d + 1)$
      if $evaluation > max\_eval$ **then**
        $max\_eval \leftarrow evaluation$, $best\_move \leftarrow move$
    return $max\_eval, best\_move$
  else($P = 2$, Minimizing)
    $min\_eval \leftarrow \infty$, $best\_move \leftarrow$ None
    for move in $EmptyCells$ **do**
      $B_{new} \leftarrow$ ApplyMove$(B, move, P)$
      $evaluation, \_ \leftarrow$ MinimaxGetScore$(B_{new}$, Player 1, $d + 1)$
      if $evaluation < min\_eval$ **then**
        $min\_eval \leftarrow evaluation$, $best\_move \leftarrow move$
    return $min\_eval, best\_move$

Algorithm 9 Strategic Threat Detection and Fork Analysis

Require: Board state B, player P
Ensure: Count of immediate threats for player P
function COUNTLINETHREATS(B, P)
    $threats \leftarrow 0$
    $\mathcal{L} \leftarrow$ All 8 winning lines of the board
    for line in $\mathcal{L}$ **do**
        $pieces \leftarrow$ GetPiecesOnLine(B, line)
        if count($pieces, P$) $= 2$ and count($pieces$, empty) $= 1$ **then**
            $threats \leftarrow threats + 1$
    return $threats$

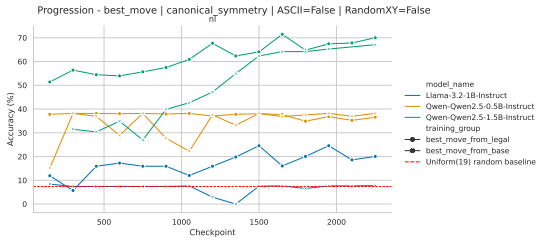
Require: Board state B, player P
Ensure: Boolean indicating if a fork opportunity exists for player P
function HASFORK(B, P)
    $OpenSquares \leftarrow$ FindEmptyCells(B)
    for square in $OpenSquares$ **do**
        $B_{temp} \leftarrow$ ApplyMove(B, square, P)
        if CountLineThreats(B_{temp}, P) ≥ 2 **then**
            return true
    return false

Algorithm 10 Board Canonical Form Generation

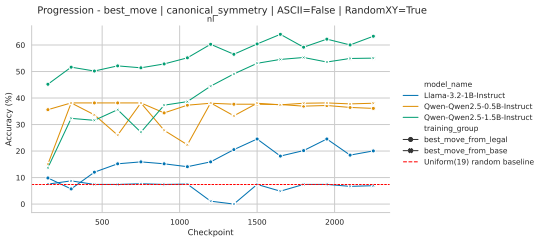
Require: A 3×3 board matrix B
Ensure: The lexicographically smallest (canonical) representation of the board
function GETCANONICALFORM(B)
    $Symmetries \leftarrow []$
    $B_{current} \leftarrow B$
    for $i = 1$ to 4 **do**
        Append Flatten($B_{current}$) to $Symmetries$
        $B_{flipped} \leftarrow$ FlipLeftRight($B_{current}$)
        Append Flatten($B_{flipped}$) to $Symmetries$
        $B_{current} \leftarrow$ Rotate90Degrees($B_{current}$)
    return $\min(Symmetries)$

B QUANTITATIVE ANALYSIS

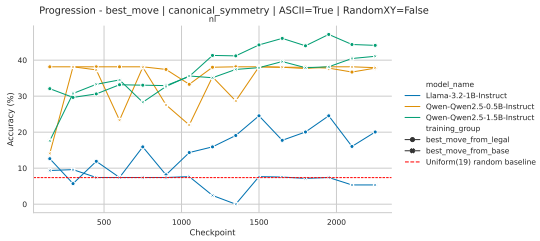
This section provides the complete set of quantitative results for all settings. For both random and symmetry datasets, we conducted evaluations for both text based natural language board representations as well as ascii based board representations. These were done to evaluate both legal move and best move obectives across all trained model checkpoints. Model's robustness to the prompt variations was tested by chosing random alphanumeric characters to replace the player tokens (X, Y) for the same board.



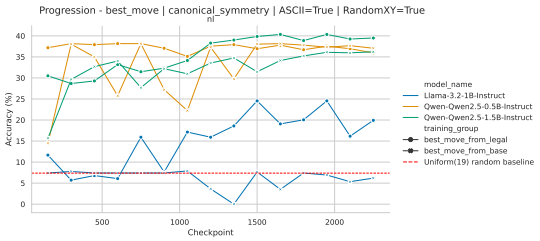
(a) Progression, best move, Natural language board representation, random-False



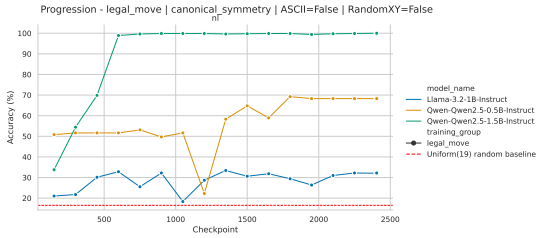
(b) Progression, best move, Natural language board representation, random-True



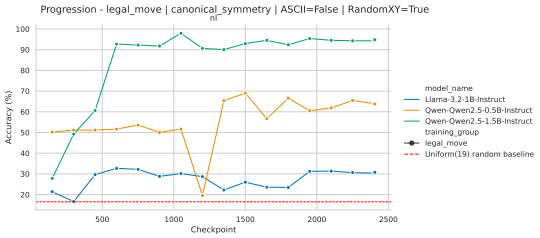
(c) Progression, best move, ASCII board representation, random-False



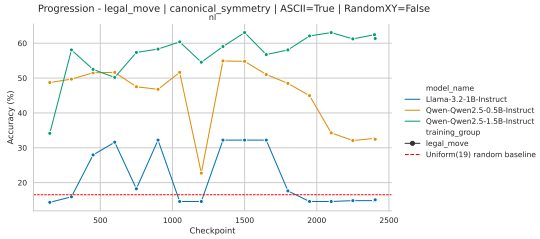
(d) Progression, best move, ASCII board representation, random-True



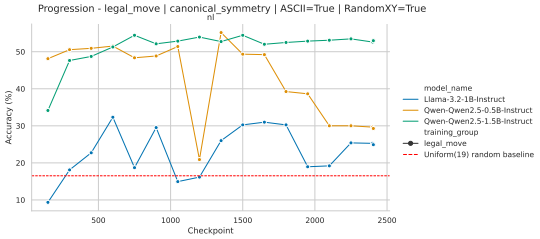
(e) Progression, legal move, Natural language board representation, random-False



(f) Progression, legal move, Natural language board representation, random-True

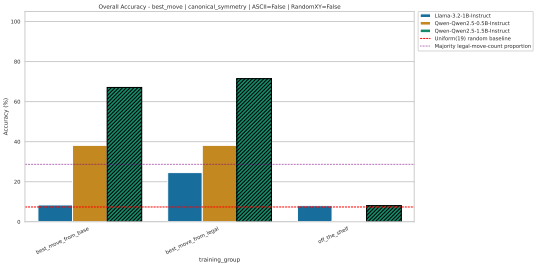


(g) Progression, legal move, ASCII board representation, random-False

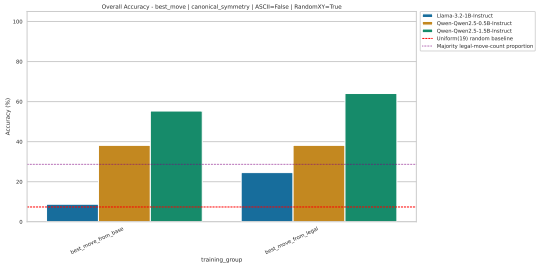


(h) Progression, legal move, ASCII board representation, random-True

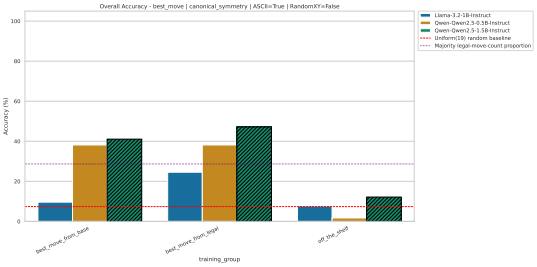
Figure 6: Progression analysis across board representations and randomization.



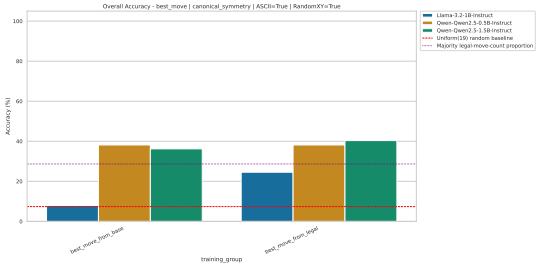
(a) Overall, best move, Natural language board representation, random-False



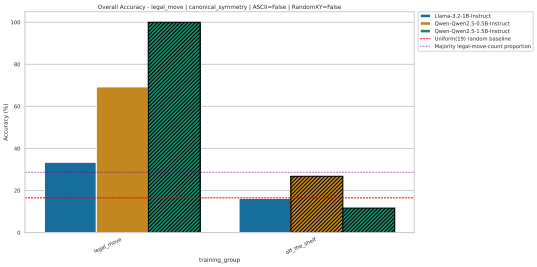
(b) Overall, best move, Natural language board representation, random-True



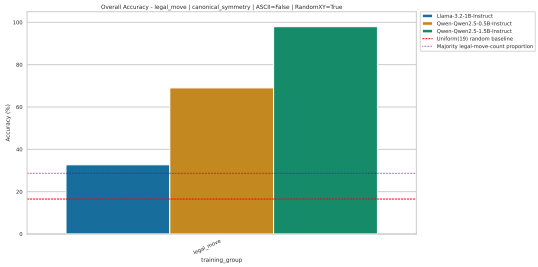
(c) Overall, best move, ASCII board representation, random-False



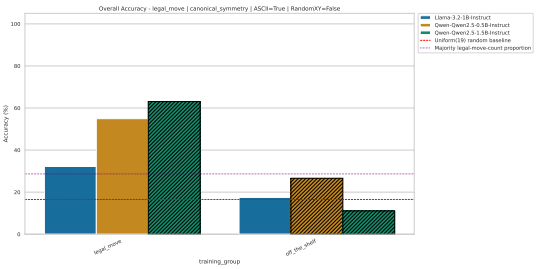
(d) Overall, best move, ASCII board representation, random-True



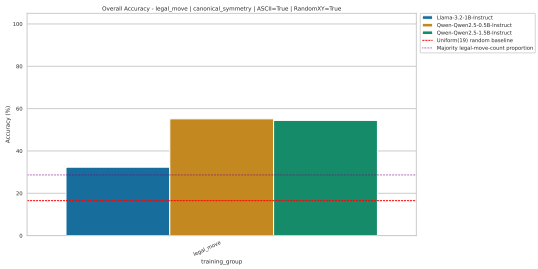
(e) Overall, legal move, Natural language board representation, random-False



(f) Overall, legal move, Natural language board representation, random-True

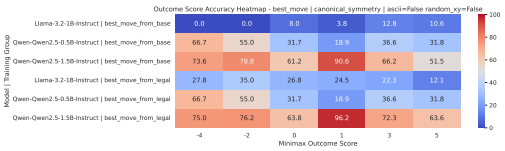


(g) Overall, legal move, ASCII board representation, random-False

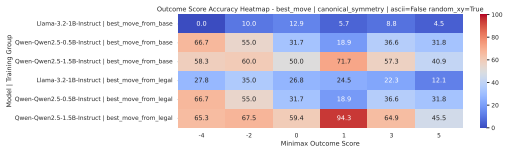


(h) Overall, legal move, ASCII board representation, random-True

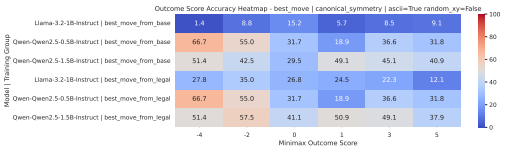
Figure 7: Overall performance across board representations and randomization.



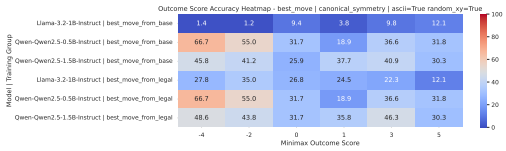
(a) Heatmap, best move, Natural language board representation, random-False



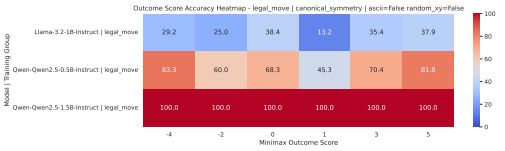
(b) Heatmap, best move, Natural language board representation, random-True



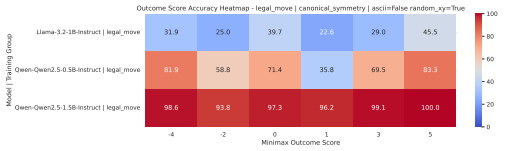
(c) Heatmap, best move, ASCII board representation, random-False



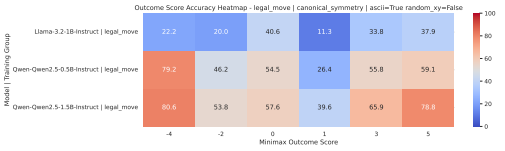
(d) Heatmap, best move, ASCII board representation, random-True



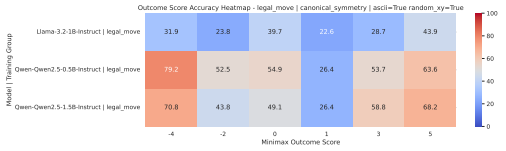
(e) Heatmap, legal move, Natural language board representation, random-False



(f) Heatmap, legal move, Natural language board representation, random-True

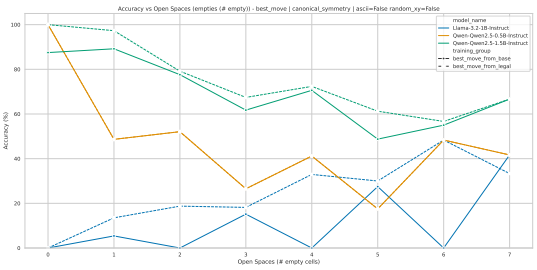


(g) Heatmap, legal move, ASCII board representation, random-False

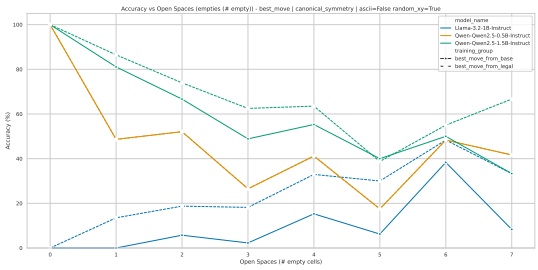


(h) Heatmap, legal move, ASCII board representation, random-True

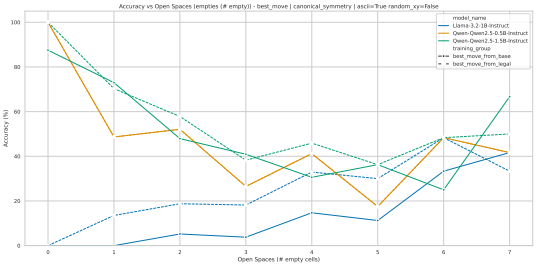
Figure 8: Outcome score heatmaps across board representations and randomization.



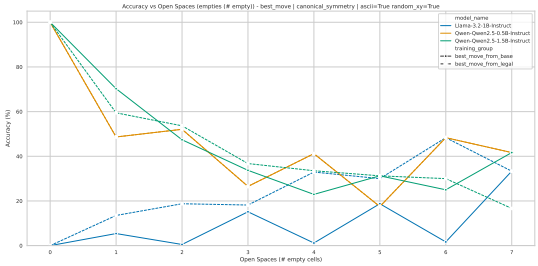
(a) Open spaces, best move, Natural language board representation, random-False



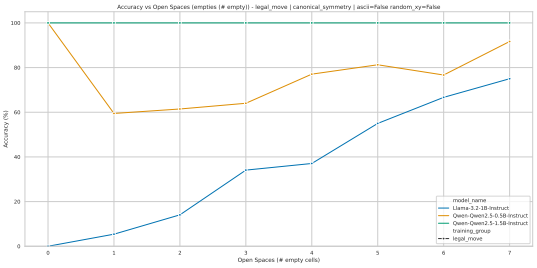
(b) Open spaces, best move, Natural language board representation, random-True



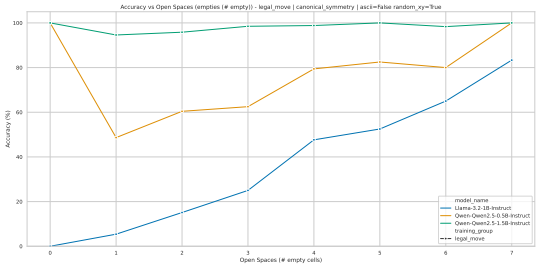
(c) Open spaces, best move, ASCII board representation, random-False



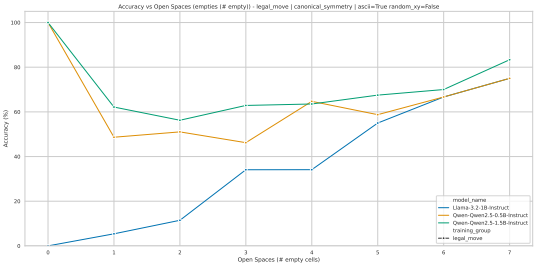
(d) Open spaces, best move, ASCII board representation, random-True



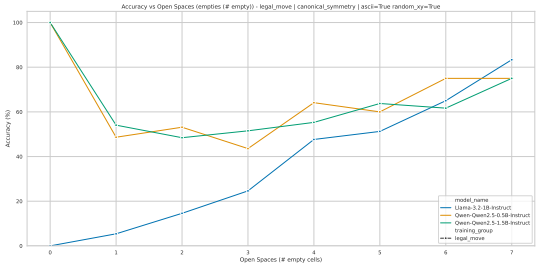
(e) Open spaces, legal move, Natural language board representation, random-False



(f) Open spaces, legal move, Natural language board representation, random-True

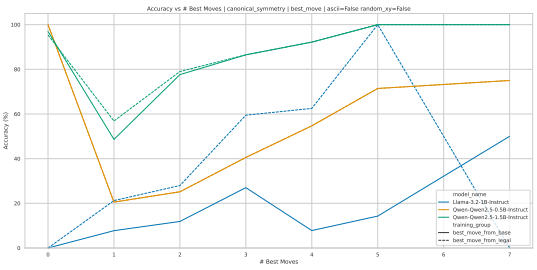


(g) Open spaces, legal move, ASCII board representation, random-False

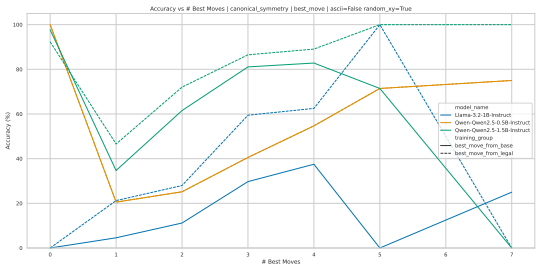


(h) Open spaces, legal move, ASCII board representation, random-True

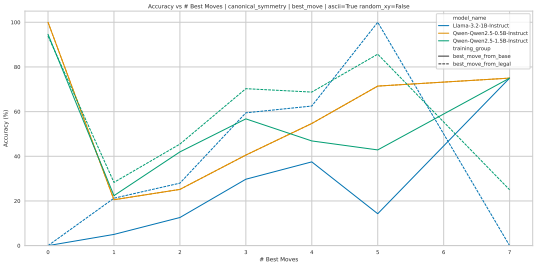
Figure 9: Open spaces analysis across board representations and randomization.



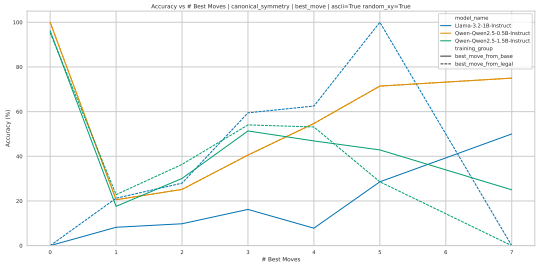
(a) Complexity, best move, Natural language board representation, random-False



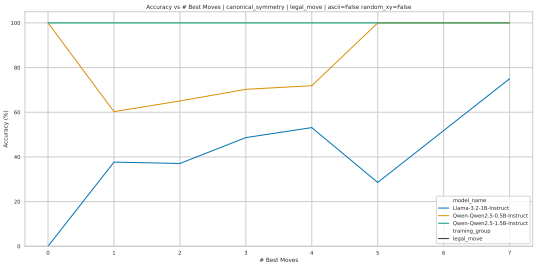
(b) Complexity, best move, Natural language board representation, random-True



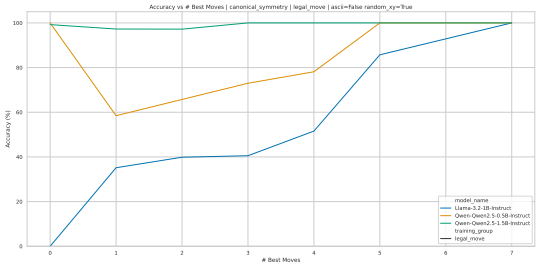
(c) Complexity, best move, ASCII board representation, random-False



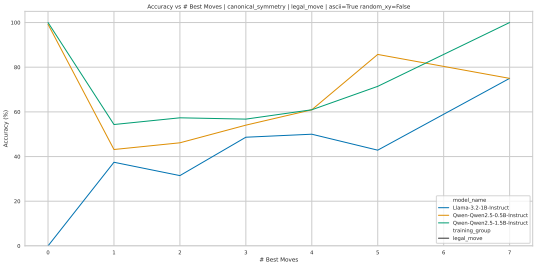
(d) Complexity, best move, ASCII board representation, random-True



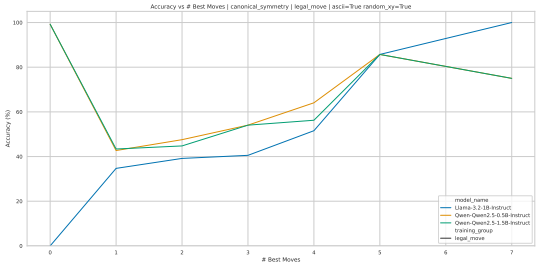
(e) Complexity, legal move, Natural language board representation, random-False



(f) Complexity, legal move, Natural language board representation, random-True



(g) Complexity, legal move, ASCII board representation, random-False

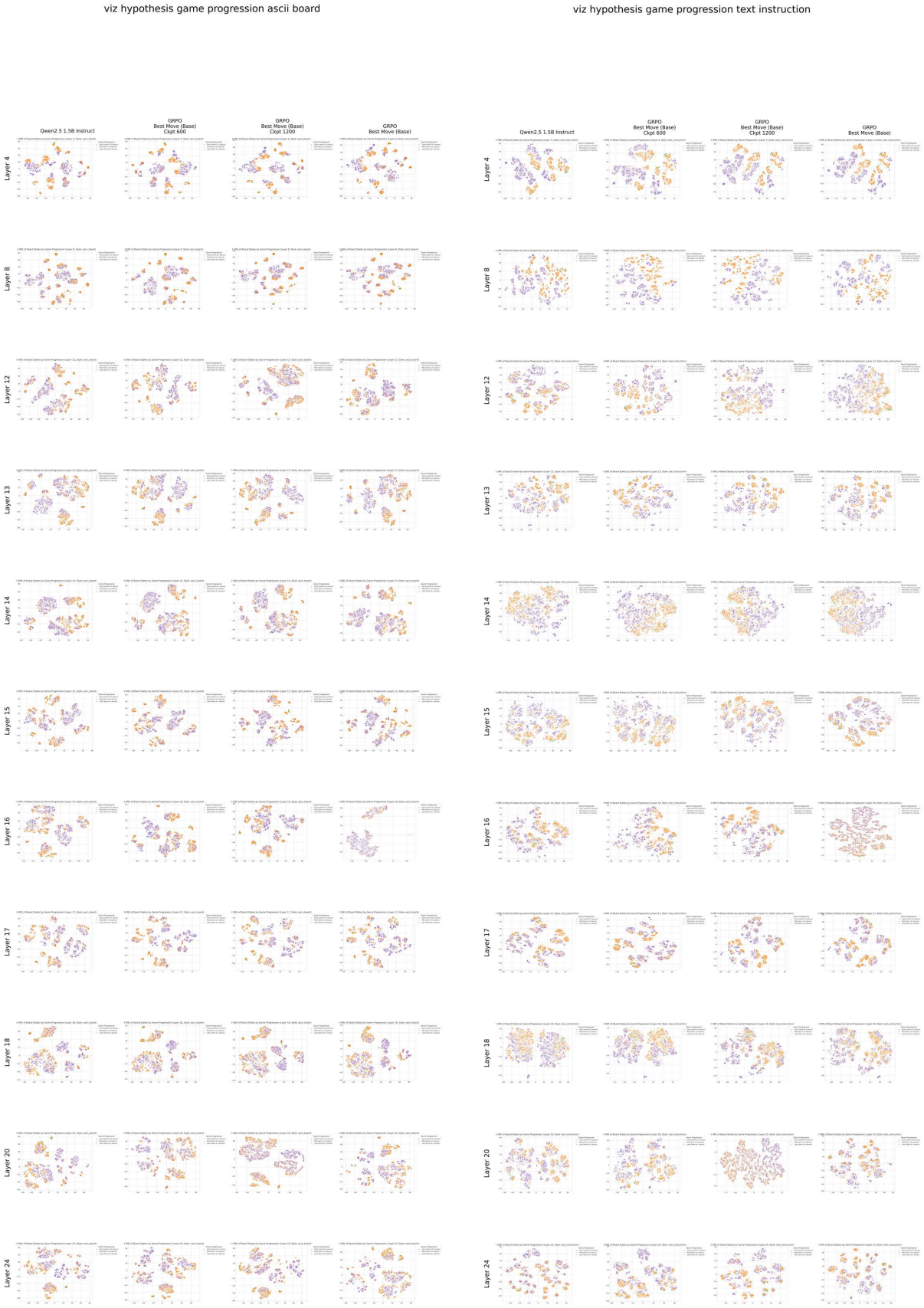


(h) Complexity, legal move, ASCII board representation, random-True

Figure 10: Complexity analysis across board representations and randomization.

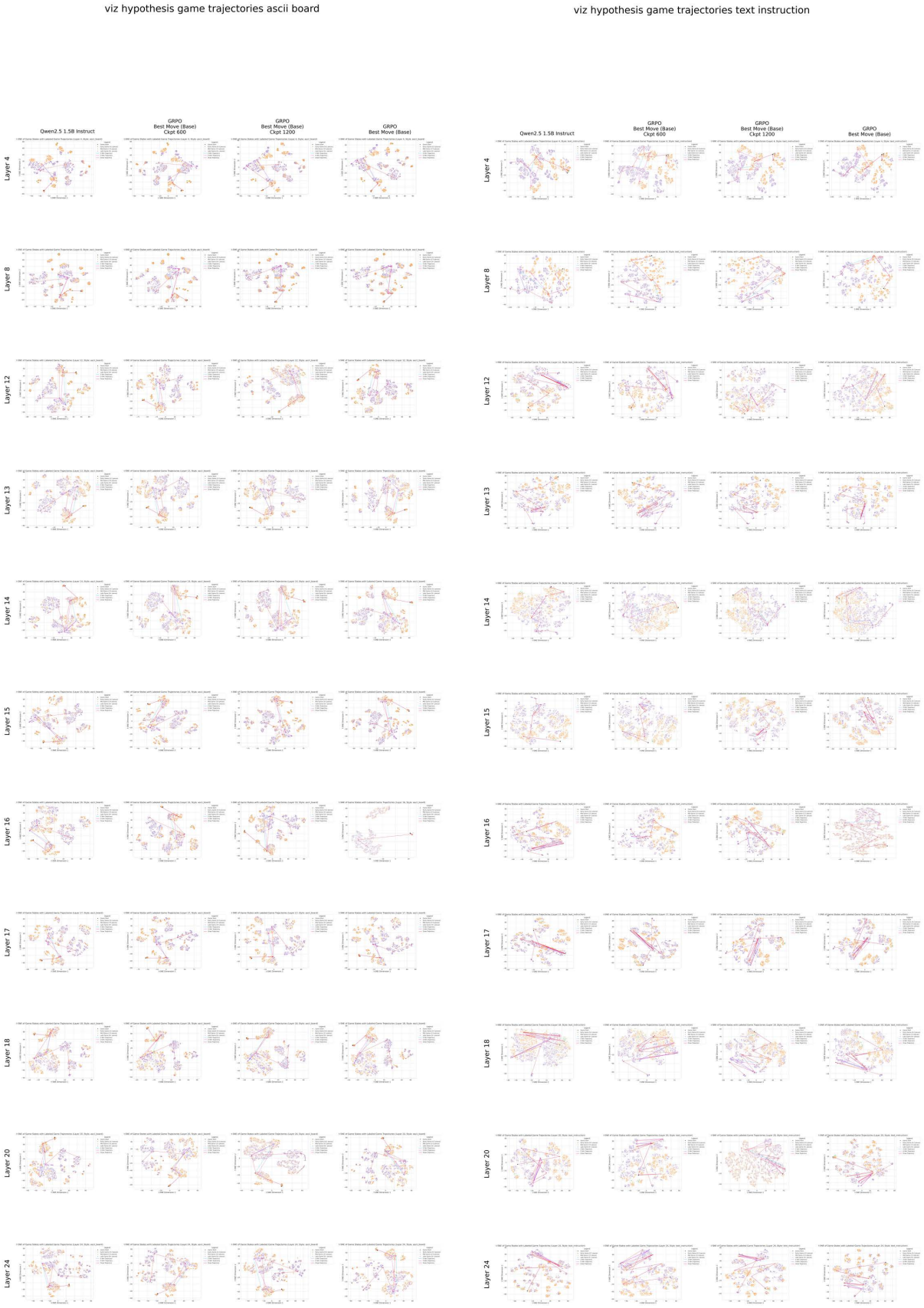
C Additional Mechanistic Interpretability Results

The plots below map the hypotheses from SAE probing across all layers and along training checkpoints for both natural language and ascii representations. Due to size constraints on the main paper, the plots have been compressed to allow the PDF to stay within 50MB. Complete plots have been provided with the supplemental submission for further investigation.



(a) ASCII board representation (b) Natural language instruction representation

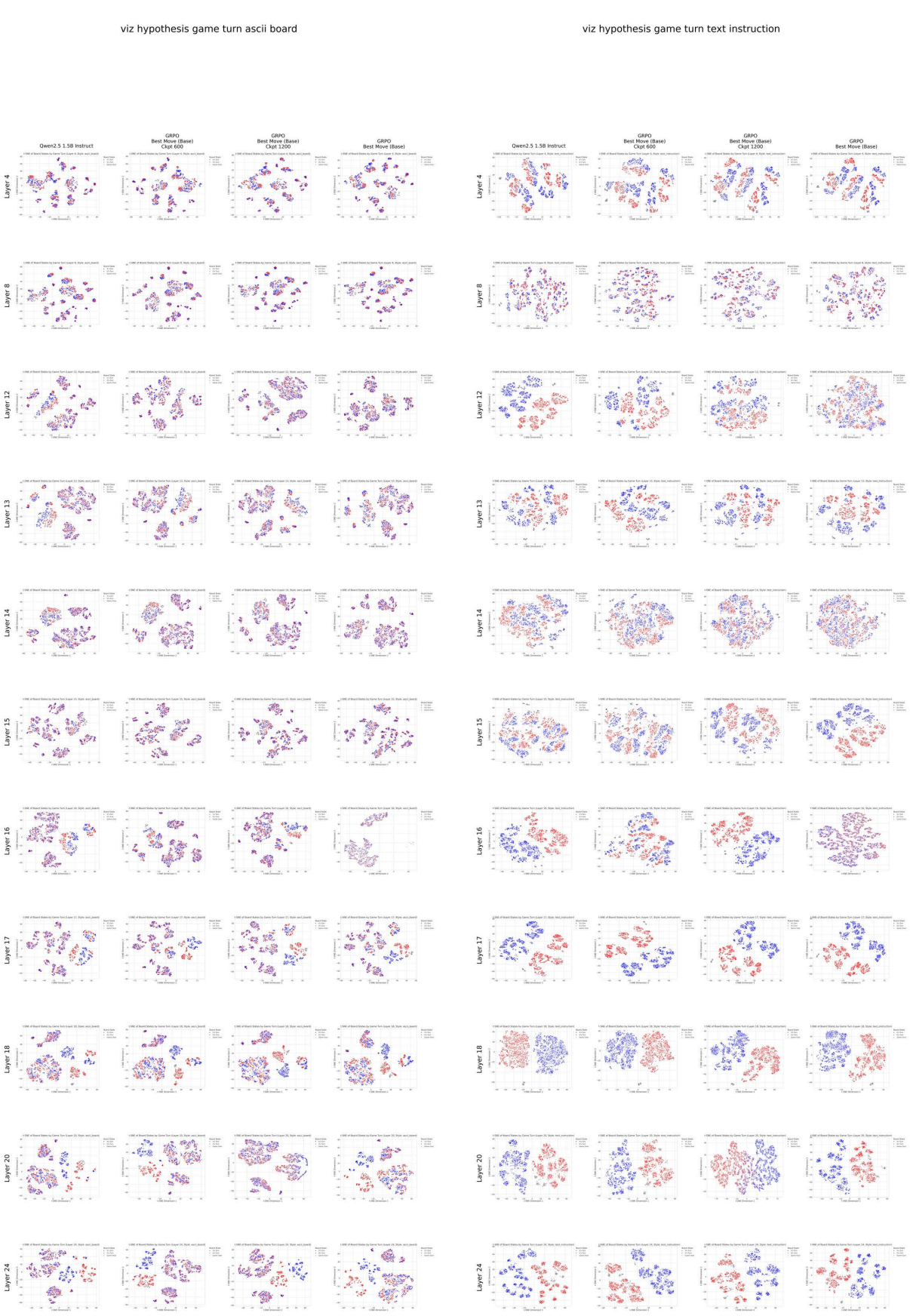
Figure 11: Hypothesis testing: Game progression.



(a) ASCII board representation

(b) Natural language instruction representation

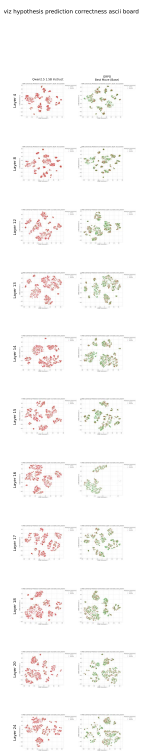
Figure 12: Hypothesis testing: Game trajectories.



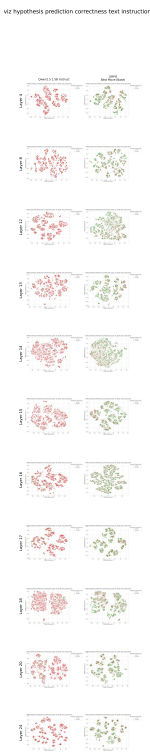
(a) ASCII board representation

(b) Natural language instruction representation

Figure 13: Hypothesis testing: Game turn.



(a) ASCII board representation

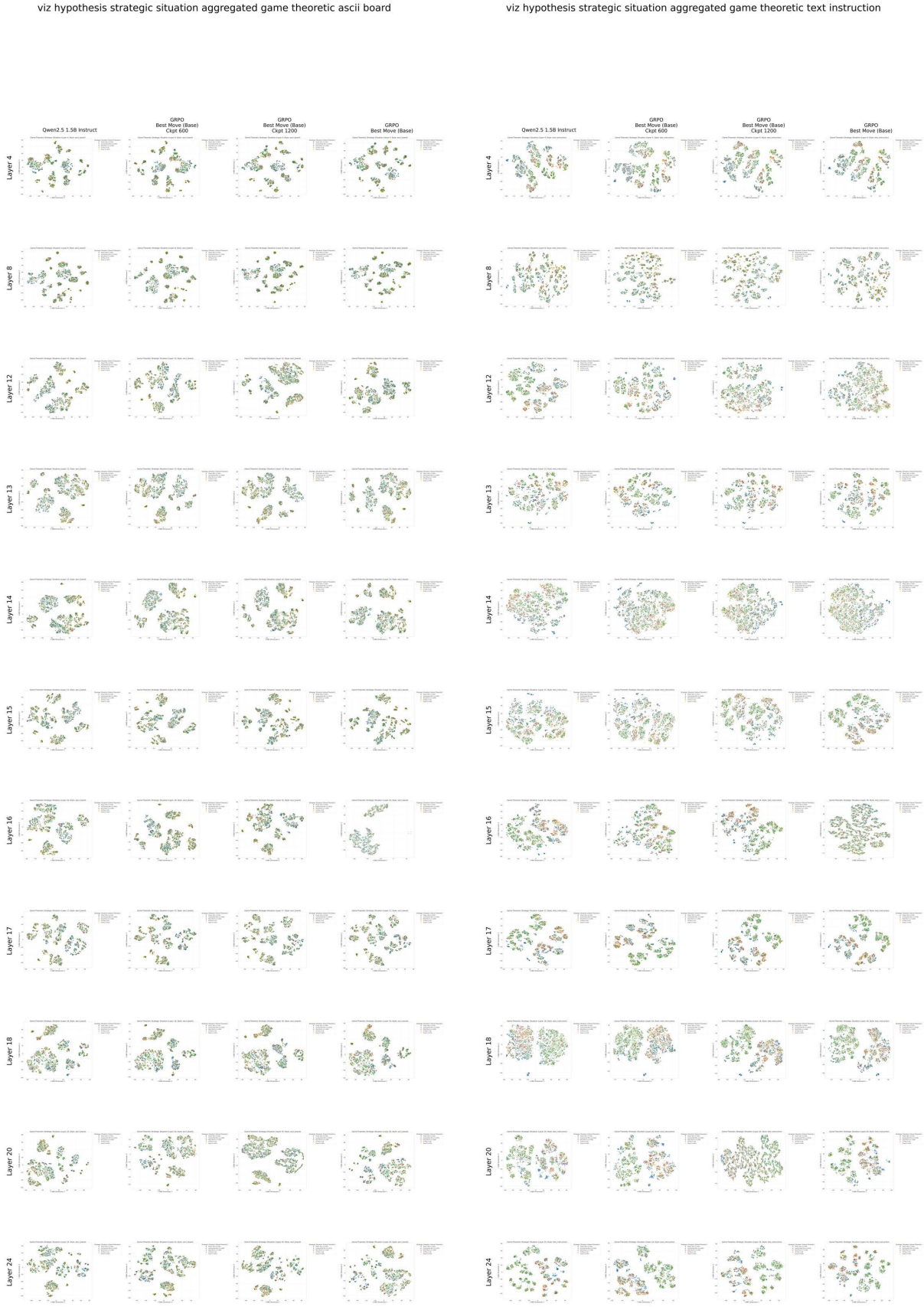


(b) Natural language instruction representation

Figure 14: Hypothesis testing: Prediction correctness.

viz illegal vs legal ascii board

viz illegal vs legal text instruction



(a) ASCII board representation

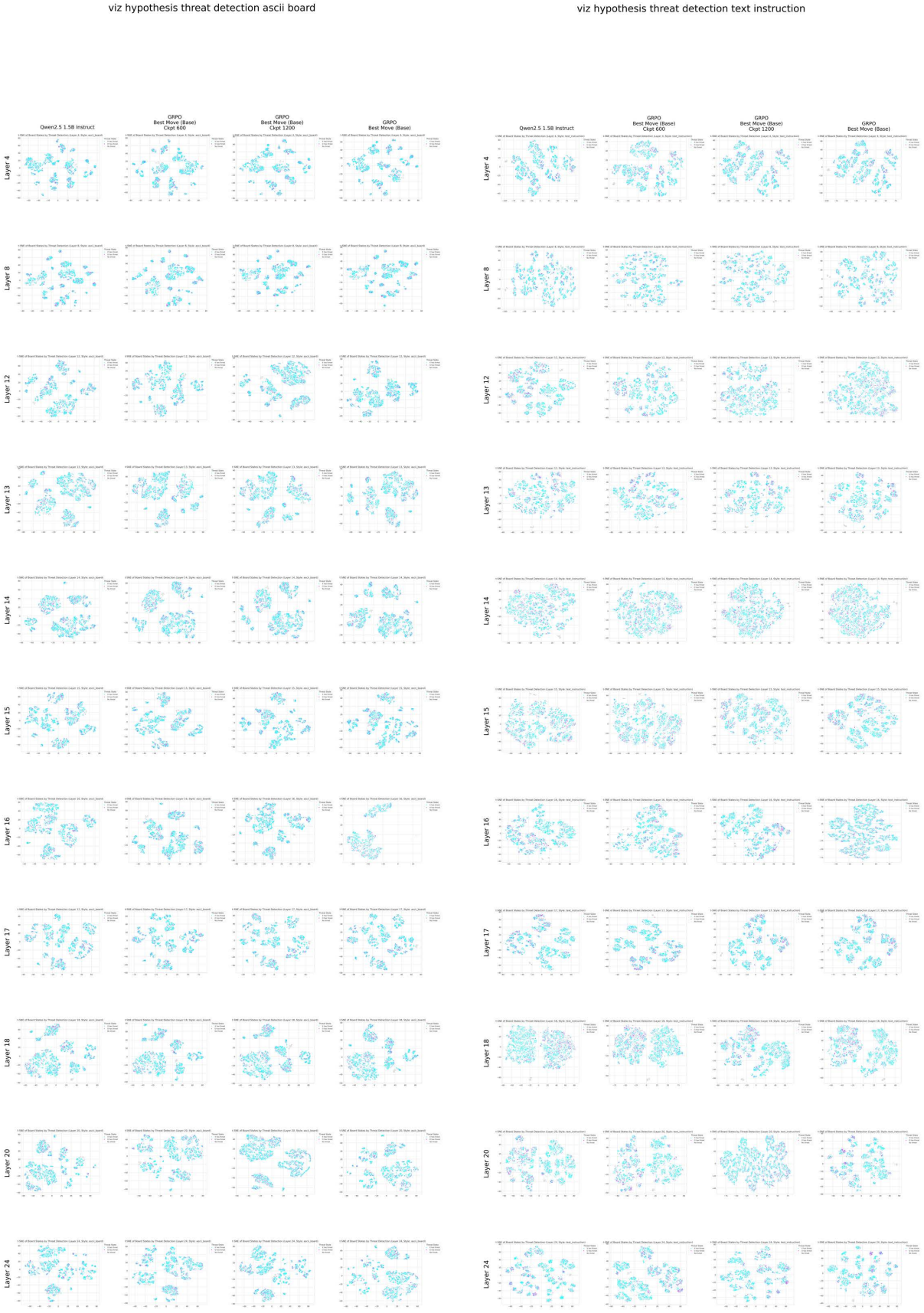
(b) Natural language instruction representation

Figure 16: Hypothesis testing: Strategic situations.

(a) ASCII board representation

(b) Natural language instruction representation

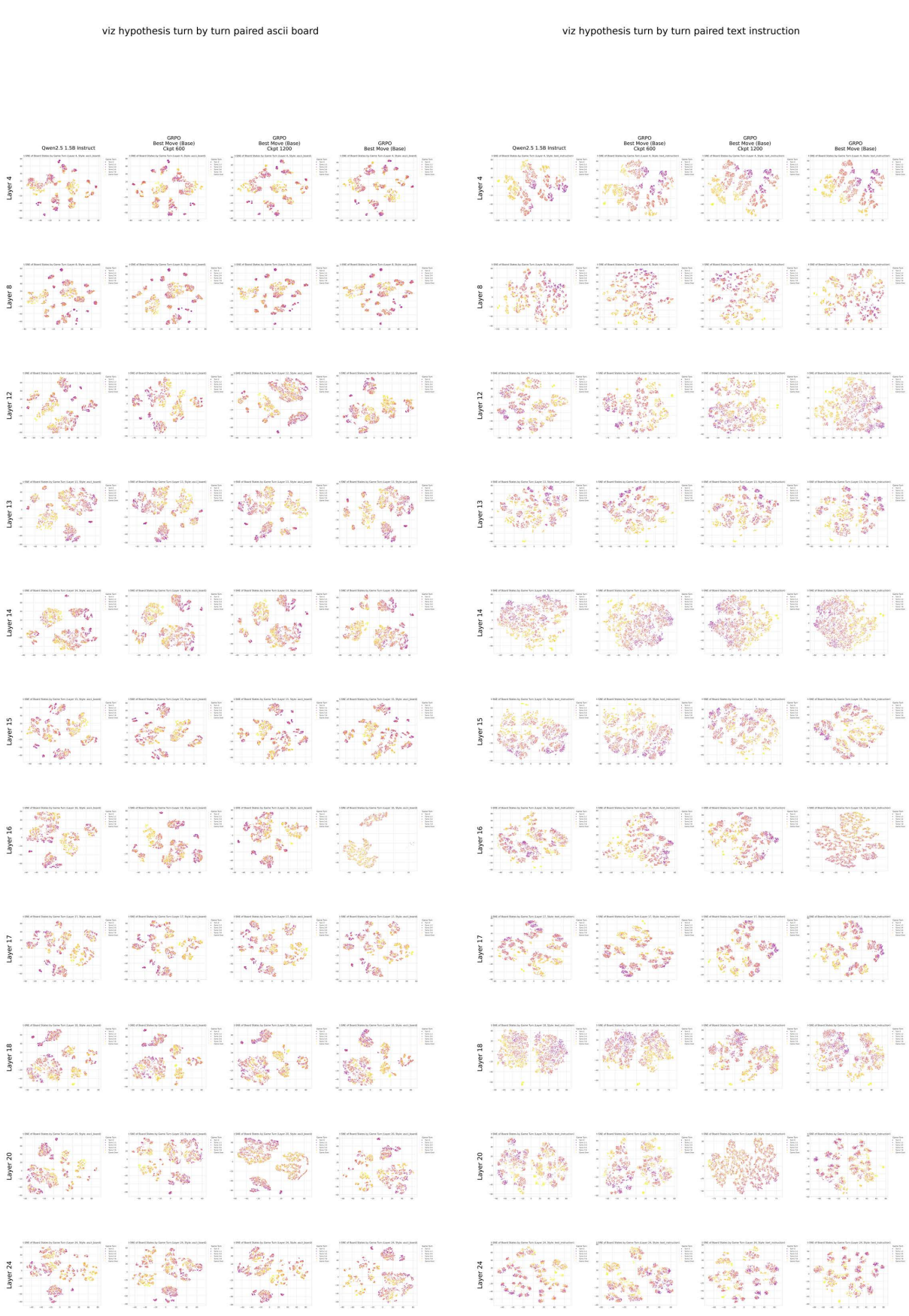
Figure 17: Hypothesis testing: Symmetry.



(a) ASCII board representation

(b) Natural language instruction representation

Figure 18: Hypothesis testing: Threat detection.



(a) ASCII board representation

(b) Natural language instruction representation

Figure 19: Hypothesis testing: Turn-by-turn paired analysis.

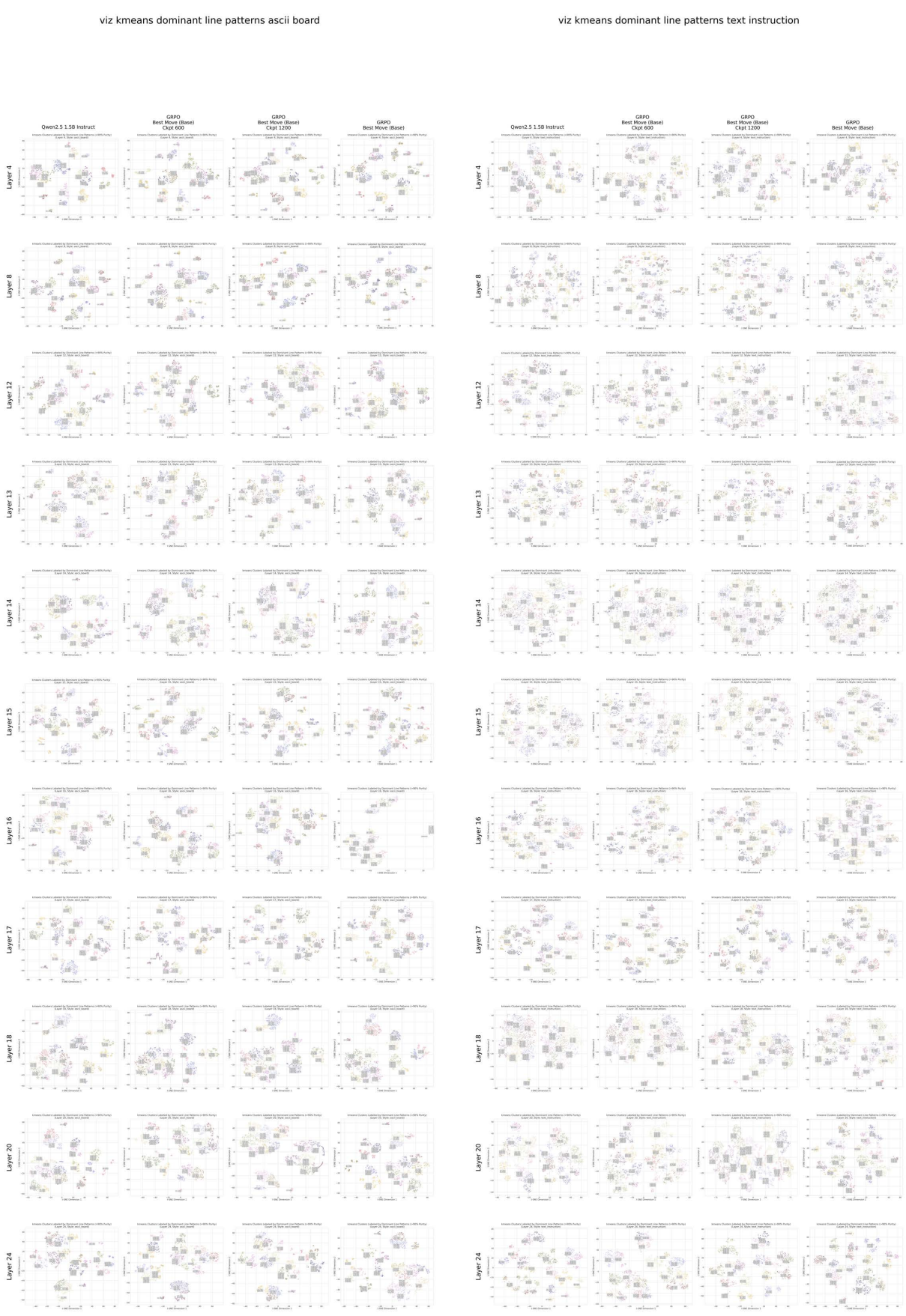
(a) ASCII board representation (b) Natural language instruction representation

Figure 20: Hypothesis testing: Winner identification.

(a) ASCII board representation

(b) Natural language instruction representation

Figure 21: Hypothesis testing: Winner identification with best labels.



(a) ASCII board representation

(b) Natural language instruction representation

Figure 22: KMeans clustering: Dominant line patterns.

(a) ASCII board representation

(b) Natural language instruction representation

Figure 23: KMeans clustering: Global extremes (normalized).

(a) ASCII board representation

(b) Natural language instruction representation

Figure 24: KMeans clustering: Hypothesis summary (normalized).