Evaluation Beyond Task Performance: Analyzing Concepts in AlphaZero in Hex

Anonymous Author(s)
Affiliation
Address
email

Abstract

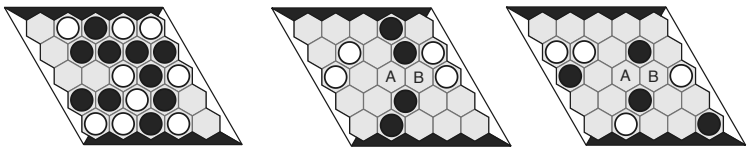
AlphaZero, an approach to reinforcement learning that couples neural networks and Monte Carlo tree search (MCTS), has produced state-of-the-art strategies for traditional board games like chess, Go, shogi, and Hex. While researchers and game commentators have suggested that AlphaZero uses concepts that humans consider important, it is unclear how these concepts are captured in the network. We investigate AlphaZero's internal representations in the game of Hex using two evaluation techniques from natural language processing (NLP): model probing and behavioral tests. In doing so, we introduce several new evaluation tools to the RL community, and illustrate how evaluations other than task performance can be used to provide a more complete picture of a model's strengths and weaknesses. Our analyses in the game of Hex reveal interesting patterns and generate some testable hypotheses about how such models learn in general. For example, we find that the MCTS discovers concepts before the neural network learns to encode them. We also find that concepts related to short-term end-game planning are best encoded in the final layers of the model, whereas concepts related to long-term planning are encoded in the middle layers of the model.

1 Introduction

AlphaZero [49], a reinforcement learning agent that combines Monte Carlo tree search [7] with deep reinforcement learning, has achieved impressive performance at games like Go, chess, shogi, and Hex. Domain experts have observed that AlphaZero uses, but does not master, identifiable game concepts. For example, despite being exceptionally strong overall, AlphaZero appeared unable fully to project the implications of a *ladder*—an important concept in the game of Go [54].

Good performance can mask flaws in deep learning systems generally [18, 43, 2], and deep reinforcement learning systems in particular [60, 61]. Evaluating systems in terms of task performance alone makes it impossible to know if systems are "right for the right reasons" and difficult to predict how they will generalize to new situations. Recently, the field of natural language processing (NLP) has begun to develop evaluation techniques that go beyond "just" task performance [3, 4]. For example, some techniques, *probing classifiers*, inspect the form of models' internal representations to test whether they are consistent with linguistic theory [1, 8, 52]; other techniques, *behavioral tests* or *challenge sets*, evaluate specific types of out-of-distribution generalization to assess whether models encode human-like inductive biases [15, 28, 30, 32, 36, 57]. The insights generated by these techniques are not only scientifically interesting, but have begun to yield actionable insights on how to employ and improve models. For example, Geva et al. [16] found that MLPs in transformer layers act as key-value memories, and Meng et al. [38] leveraged this understanding to manipulate model behavior in a controlled manner.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) **An example winning board for black**, which connects the black edges together.

(b) **Short- vs Long-term concepts.** Left: If black plays A or B, black immediately wins; the board concept containing A and B, *bridge* (See 2a), is relevant in the *short-term*. Right: A and B can help black win only in the *long-term*.

Figure 1: To win in Hex, a player must use their pieces to form a connecting chain between the edges matching their color (a). Hex boards can be of varying size; we evaluate AlphaZero on a 9x9 board.

In this paper, we demonstrate how these ideas from NLP can be applied to reinforcement learning and begin to illuminate the capabilities and limitations of our models in a new way. Specifically, we leverage the above-described analysis techniques—probing classifiers and behavioral tests—to interpret AlphaZero's behavior at a conceptual level. We use probing classifiers to determine if reinforcement-learning agents encode tactical and strategic conceptual information. However, probing performance alone is insufficient: information may be encoded but not used [33]. To address this issue, we also use behavioral tests, which evaluate an agent's decisions in a situation tailored to require the understanding of a specific concept.

Given these concept-level evaluation methods, we do an in-depth study of AlphaZero (AZ) trained to play Hex. Hex is a board game similar to Go (§2), but provides an opportune test bed for analysis: the game is complex, yet has perfect-play baselines for smaller board sizes. Moreover, concepts and strategies within Hex have been studied by the Hex-playing community, making Hex a rich vector for study. Furthermore, the clear behavioral expectations in Hex make it easier to interpret and guide future work.[1] In our work, we probe for concepts that are taught to new players of Hex, like the *bridge* (Fig. 2a), and test that the model is able to use them to win games. We investigate how concepts are represented within AZ, when concepts are learned during training, and where concepts are represented within AZ's neural network.

Overall, our findings suggest that there are some regular dynamics to how AlphaZero learns concepts with the game of Hex, and generate interesting predictions about how AlphaZero behaves in general, which could be tested in other games. (In fact, concurrent work on chess [37] and Go [55] already begin to provide convergent evidence that some trends we see here generalize elsewhere.) We introduce a novel way of analyzing deep RL models on which such subsequent work can easily build.

In summary, our main contributions are:

1. We introduce several evaluation techniques from NLP to the RL community, and illustrate how evaluations other than task performance can be used to provide a more complete picture of a model's strengths and weaknesses.
2. We analyze a top-performing model from Jones [26], and find that (1) short-term end-game concepts are best represented in the final layers of the network, whereas long-term concepts are best represented in the middle layers of the network; (2) concepts appear to originate with MCTS—with MCTS overriding the deep learning policy prediction early in training—but later in training these concepts are incorporated directly into the model's network.

2 Concepts in Hex

We study AlphaZero (AZ) agents trained to play Hex [13], a game where two players take turns filling cells until one player builds a chain across the board. Hex has well-studied rules, reasonable computational costs, and it can be evaluated against perfect play, making it an ideal experimental

[1] Again drawing inspiration from work in NLP, Linzen et al. [31] focused on one syntactic phenomenon in simple RNNs, and the basic insight and methods enabled later studies to make broader claims about learning linguistic structure in many types of models [22, 57, 58].

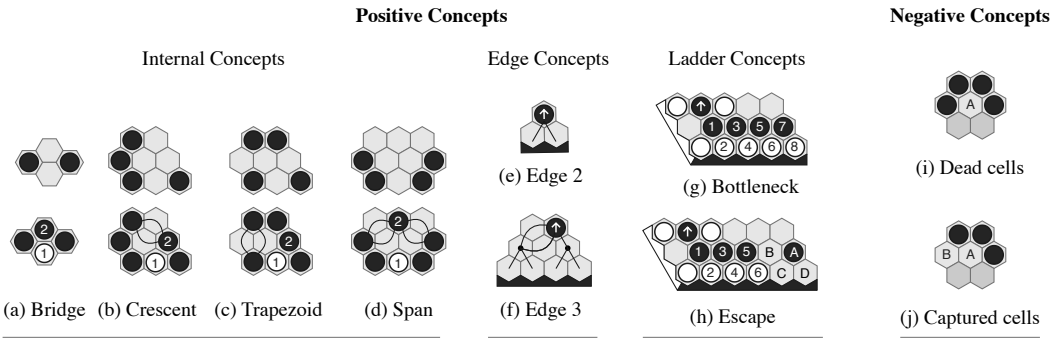


Figure 2: **Hex templates exemplifying game concepts.** Concepts within the game of Hex are templates on the board formed by a player's pieces with known strategic and tactical implications. Positive concepts provide the player with the concept with multiple ways to connect the pieces within the concept together, despite possible attacks from the opponent. An example of these properties is the bridge (a). If white plays move 1, black can connect the two pieces of the bridge by playing move 2. Negative concepts change the strategic value of specific open spots of the board, such that the opponent is disincentived to play those open spots, such as move A in (i). Each concept is further described in our Supplementary Material. Arrows indicate that the piece is connected to the opposite side of the board; the lines show the bridge concept within the other concepts.

vehicle for model probing. Unlike Go, there are no captures; once a cell is filled with a piece, the pieces stays there for the remainder of the game. For example, in Fig. 1a, white must connect pieces from left to right (marked on the edges), and black top to bottom. Hex cannot end in a tie [11], and given perfect play, black, the first player, will win [13]. [2]

In Hex, certain templates – patterns of cells – have been recognized as useful. Because building up a chain from one side of the board to the other is easily thwarted, a key part of learning how to play Hex is recognizing when it is possible to connect groups of pieces together. We consider these templates to be *concepts* within the game. Concepts in Hex have different move implications depending on the condition of the board. We discuss two conditions, long- and short-term, in detail and provide additional discussion of conditions in the Supplementary Material.

While the properties of concepts are debated [34], here, in a board game setting, we consider a concept to be a useful template that generalizes across board configurations. For our purposes, "understanding" a concept amounts to recognizing it and leveraging its implications during gameplay (§2.2).

2.1 Long-term vs short-term Concepts

We define a concept in Hex to be *short-term* if its use is sufficient to win the game. Concepts are typically short-term when there exists a connection between the concept and the player's board edges. For example, in Fig. 1b, moves A and B belong to the *bridge* concept (see Fig. 2a). In the left board, the board edges are connected, and all that is required for black to win is to play moves A or B. Conversely, *long-term* concepts are insufficient to winning the game if immediately used. In the right board of Fig. 1b, playing the *bridge* concept is useful but insufficient for winning immediately; the concept is not connected to the edges of the board. Empirically, we find that whether the concept is short- or long-term has a significant impact on AlphaZero's representations (Fig. 4).

2.2 Concept Taxonomy

From Seymour [48] and King [29], we identify the nine concepts that we use in our analysis, summarized in Fig. 2. These concepts fall into four categories, as described below.

[2] Hex is often played with a "swap rule" that makes the game more even between black and white. See [26], whose implementation we use, for further discussion on the swap rule. Jones did not include it to simplify the game implementation.

Positive Concepts With the goal of Hex being to build a chain across the board, it is helpful to recognize when cells are virtually connected, that is, even in response to perfect adversarial play, the cells are guaranteed to connect [19, 41]. There are a few different types of positive concepts. All the positive concepts favor the player that owns the concept on the board.

Internal concepts are templates that appear within the interior of the board. The *bridge* (Fig. 2(a)) is the simplest such concept. The larger internal templates – *crescent*, *trapezoid*, *span* (Fig. 2(b,c,d)) – provide several possibilities to connect a player's pieces. *Edge* concepts concern connecting a single cell to a given edge. Ladders in Hex are similar to ladders in Go. There are two different ladder concepts, *bottleneck* (g) and *escapes* (h). A bottleneck favors the defender, because the attacking player cannot break through. An escape, however, allows the attacker to break through.

Negative Concepts Negative concepts do not inform one on which actions to play, but rather, which actions not to play. *Dead cells* (Fig. 2(i)) cannot impact the outcome of the game regardless of the color with which they are filled. While it is in general difficult to compute if a cell is dead [6], there are some known templates where it is easier to deduce. If a player can make a cell dead, that cell is *captured*. Both captured and dead cells should not be filled.

3 Designing Probing Tasks and Behavioral Tests for Hex

To understand the concepts encoded within AlphaZero, we probe its internal representations; to evaluate if these concepts are used by AZ, we test its behavior on tailored board configurations. By evaluating AZ across training checkpoints, and across neural network layers, we can build up an understanding of how and where the model recognizes these concepts. Specifically, we evaluate the top-performing agent trained by Jones [26] across 21 training checkpoints. We additionally evaluate other publicly available agents [26] from that differ by width and depth and we report on those results in the Supplementary Material. Code from Jones is released under the MIT License.

3.1 Reproducibility

Our code and results are publicly available. Furthermore, we release example images of boards created for our probing classifiers and videos of the behavioral tests. The code, results and examples can be found at `https://bit.ly/alphatology`. We report hyperparameters in the Supplementary Material. All error bars presented in plots are one standard deviation above and below the mean. We used NVIDIA GeForce RTX 3090. The total compute across all experiments was about 24 GPU hours. We present a breakdown of the compute in the Supplementary Materials.

3.2 Representational Probing

Model probing measures how well a model's learned representations encode a concept [1]. Probing entails training a linear classifier (the *probe*) over model activations to predict the presence of the desired concept. Thus, concepts are defined by a set of examples. The linear classifier's test performance is used to interpret how well the original model encoded the concept. We follow a procedure similar to Tenney et al. [52]: For a given board $H^{(0)}$, we extract activations $H^{(l)}, l \in 1..L$ for each neural network layer. We then train linear classifiers $\mathcal{P}^{(l)}, l \in 0..L$ per layer to predict the presence vs. absence of a concept in the board; these classifiers are our concept probes.

It is important to compare probing results against a baseline. We follow Hewitt and Liang [20]'s procedure to measure *concept selectivity*, the delta between probing accuracy over a concept and random control. To form the random control, for each board $(H^{(0)}, y)$ in the probing dataset, we consistently map each cell in that board to a random cell, forming $H_s^{(0)}$. (Here, y is whether or not the board contains the concept.) In this way, the same information is encoded in the original boards, but we expect the shuffled boards to be irrelevant to Hex. Next, we train a set of linear probes $\mathcal{P}_s^{(l)}, l \in 0..L$ over the control boards $H_s^{(0)}$ to predict y . Now, finally, we can compute the concept selectivity by finding the delta in accuracy between $\mathcal{P}_s^{(l)}$ and $\mathcal{P}^{(l)}$.

Implementation Details. Each concept is defined by a set of boards with vs. without that concept. We train and evaluate probing classifiers over AlphaZero's encoding of these boards. To generate

a set of ($N = 2500$) boards for each concept, we translate the minimal templates across an empty board. Then, we add random enemy pieces to the board. Negative instances of a given concept match the statistics of the positive examples, except that the pieces pertaining to the concept template are randomly moved across the board. This constitutes the long-term version of a concept. To form the short-term version of a concept, we connect the template to the edges of the board.

3.3 Behavioral Tests

Where model probing asks if concepts are represented within the model, behavioral tests asks if the model knows how to use the concept in gameplay. To interpret the behavioral tests, they must have clear behavioral expectations. We construct our behavioral tests for positive concepts (§2) to be forced: If AZ understands the concept and plays the expected moves, AZ will win and pass the test; otherwise, AZ will lose the game and fail the test. For negative concepts (§2), we have clear behavioral expectations. Dead and captured cells should never be filled. Thus, the behavioral test for negative cells checks that during a selfplay continuation of a board containing a dead (or captured) cell, the agent does not fill that cell.

Success on these behavioral tests are necessary but not sufficient to establish that the model has the concept. A negative result means that AZ is unable to use the concept, whereas a positive result means that AZ can use the concept to win games in forced situations.

Implementation Details. For each concept, we create behavioral tests that comprise a board, forcing moves, and expected moves. (For the dead and captured cells there are no forcing or expected moves, only the moves to avoid). We discuss the motivation for our behavioral test setup and its connection to AlphaZero's style of gameplay in the Supplementary Material. By way of example, see Fig. 3. To generate a set ($N = 100$ samples) of boards, we translate the templates to a sampled valid board position. We then connect the concept to the edges of the board. Finally, we add connections for the defending player up to the region of the concept *such that if the attacking player fails to complete the behavioral test, the other play would win*. Finally, we add the appropriate number of random pieces such that the board position is valid. The behavioral tests determine when/if AZ learns to navigate these situations correctly.

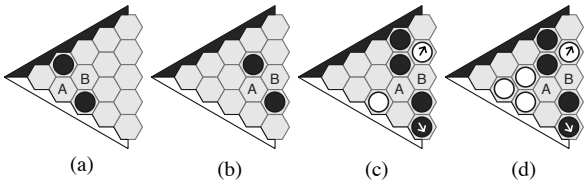
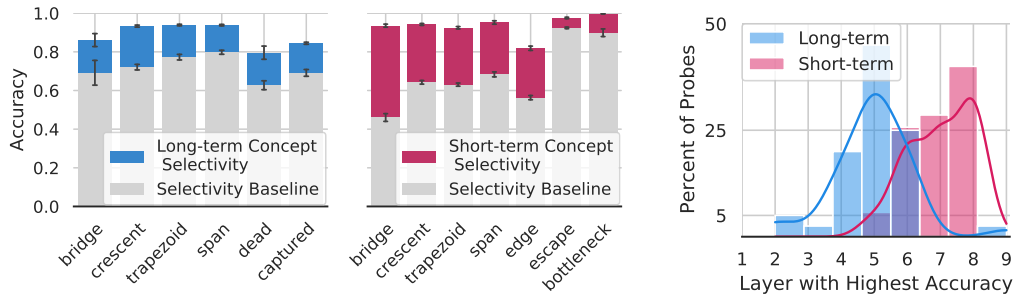


Figure 3: **Creating behavioral tests from concept templates.** To evaluate AlphaZero's ability to utilize concepts during game play, we build synthetic boards where utilizing the strategic advantages of the given concepts allows the player to win the game. In this example, we demonstrate the building of a behavioral test for the bridge concept (Figure 2a). The minimal template (a) is translated to a random position on the board (b). Then both players' pieces are connected to their respective edges they need to utilize to win the game (c). Finally, noise pieces are added to form a valid board (d). Cells A, B are used to define the behavioral test. If white plays A, black must play B to win the game. (Only half a 5x5 board is shown for space.)

3.4 Considerations

Limitations. While we find a consistent relationship between the probing and behavioral tests in Section 4.3, we do not run a counterfactual study. For example, we do not show that mistakes in recognizing a concept on a given board, lead to mistakes in using that concept. Studying the causal mechanisms of how the concept representations detected by model probing impact downstream model behavior is a rich direction for future work.

Societal Impact. Deep reinforcement learning models such as AZ are not interpretable [10], and yet are being applied to impactful, real world domains [17, 44, 14, 5, 27, 39]. In this work, we start to analyze how AZ comes to its decisions. Understanding a model's decisions is critical for accountability, but it does open the door to some avenues for exploitation. If it is known that a given



(a) **AlphaZero successfully encodes long-term and short-term concepts.** The selectivity [20], indicated by the colored bars, are the accuracy of a probe trained to identify a concept based on network activations, minus the accuracy of a selectivity baseline. The selectivity baseline randomly maps board pieces so that the new boards do not contain structures known to be relevant to Hex. We report selectivity based on the layer with the highest test accuracy.

(b) **Long-term concepts are best represented in the middle layers of the network whereas short-term concepts are best represented in the final layers of the network.** Each distribution shows where in the network layers probes had the highest accuracies. The counts are across all concepts and random seeds.

Figure 4: **Probing performance on long- and short-term concepts.**

algorithm does not reason about a concept well, this could be leveraged for ill. However, this risk only strengthens the argument for uncovering such problems and fixing them to prevent such risks.

4 Results

To understand which concepts AlphaZero (AZ) learns, we examine if its neural network activations encode the concepts (probing tests) and if AZ can use the concepts to win games (behavioral tests).

4.1 AlphaZero Recognizes and Uses Concepts

AZ successfully encodes short-term concepts, with high selectivity scores (Fig. 4a). The long-term concept scores are also learned, but with slightly lower scores. By the end of training AZ is able to use all the positive concepts to win games (Fig. 5a).

AZ improves on the behavioral tests 50% of the way through training. Unsurprisingly, the MCTS passing rates increase before the policy network passing rates (Fig. 5). Interestingly, the relative magnitude of the correct action logits initially increase much earlier. (We measure this by tracking the rate of the Z-score of the correct logits being greater than 1, meaning that the correct logit is at least one standard deviation above the mean logit score.) This suggests "pre-conceptual" information is learned, and coalesces (for bridge) 60% of the way through training into an actionable understanding of the concept. (In Section 4.4, we investigate further and find that this "pre-conceptual" information is not board structure.)

AZ also improves behaviorally upon negative concepts. However, it does not reach a perfect passing rate (Fig. 5b). The probing performance for the negative concepts, shown in Fig. 4a, is also lower than for other concepts. This aligns with evidence that AZ wastes moves at the end of the game. This highlights a weakness in AlphaZero and a risk: Some concepts may be "provable" and useful to people, but "deemed" less important by AZ – an agent that plays remarkably well. We will have to consider this dynamic as people begin to try to learn concepts from machines.

4.2 Long-term and Short-term Concepts are Best Represented in Different Layers

Fig. 4b highlight that short and long-term concepts are best represented at different layers. Long-term concepts, by the end of training, are best represented in the middle layers of the network. Short-term concepts, throughout training, are best represented in the upper layers of the network. We have two conjectures, which are not mutually exclusive, for why this may be the case. We leave the verification/refutation to future work. (1) Short-term concepts generally require more global

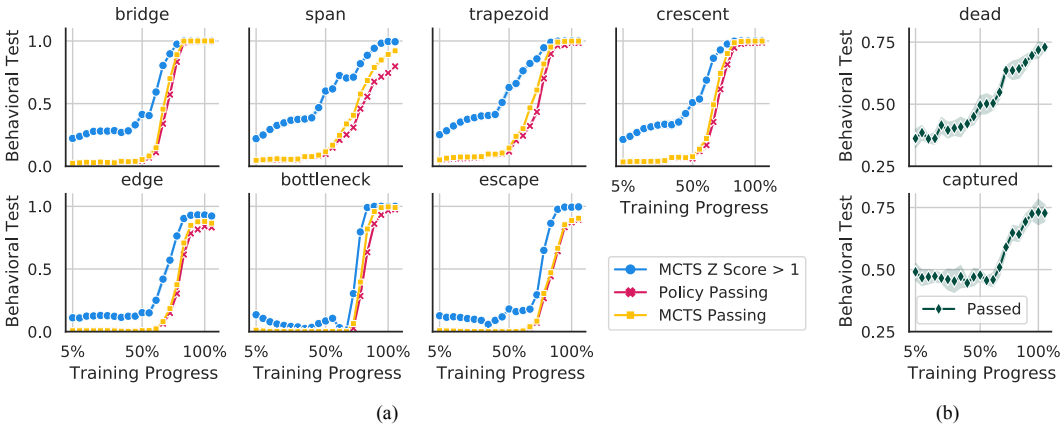


Figure 5: **(a): AlphaZero learns to use positive concepts.** At each checkpoint, we present AlphaZero (AZ) with a set of example boards that test its ability to utilize each concept. MCTS and the policy network both select actions that pass our behavioral tests with increasing frequency throughout training. We additionally report the rate at which the logit score of the action that passes our behavioral test is one standard deviation above the mean logit score (z score > 1).
(b): AZ does not fully use the negative concepts. Passed denotes the rate at which AZ avoids the negative concept throughout selfplay rollouts. So, at the end of training, AZ plays moves in 25% of our behavioral tests that will not impact the game [6].

information and so require more depth; (2) Short-term concepts factor directly into action selection and so are more proximal to the final task-specific layer of the network.

We find that two and four layer networks demonstrate the same pattern; see our Supplementary Material. Lastly, McGrath et al. [37]'s results in chess are similar, although they did not make particular note of it. In chess, they found "in check" (short-term) was best represented higher in the network and "material imbalance" (long-term) was largely represented lower in the network. [3]

4.3 Improvements in Behavioral Tests Occur Before Improvements in Probing Accuracy

Our goal is to determine if AlphaZero (AZ) understands game concepts in Hex. We find that the first improvements in behavioral tests precede the first improvement in probing accuracy (Fig. 6). MCTS seems to discover concepts, especially the internal concepts (Fig. 6). Then, as the policy network is trained to match the MCTS logits, the concept representation is absorbed into the network.

We find evidence that the structure of the board is first learned at about the same time other concepts are learned. In Section 4.4, we discuss how we probe for this structure, determining if AZ encodes the relative distance between all cells on the board. The last row of Fig. 6, labeled *neighbors*, shows that improvements on this concept occur at a similar time as to other concepts. This suggests that AlphaZero does not learn concepts according to an obvious order or curriculum, but rather, concepts of differing levels of complexity develop in parallel.

The results shown in Fig. 6 outline the relationship between the concepts as they are learned during training, but does not establish a causal relationship between how the concept representation (as detected by the probing models) impacts how the model uses that concept.

[3] Figures from [37] also suggest this finding: Informally categorized, the short-term concepts (McGrath et al.'s Fig. 2)[c,e,f] are better represented in higher layers than are the long-term concepts (Fig. 2)[a,g,h,i].

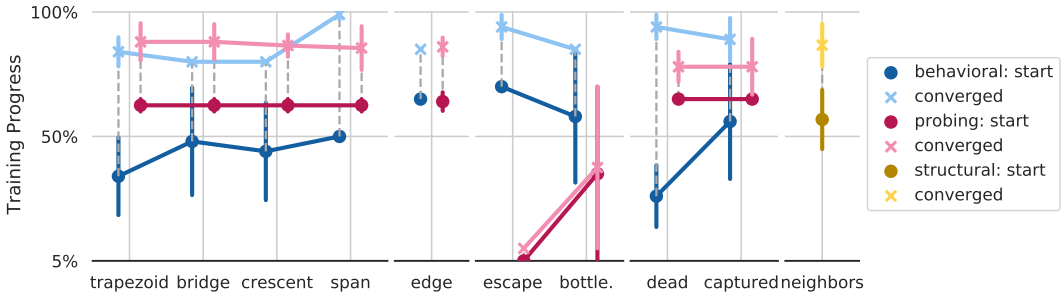


Figure 6: **Improvements in behavioral tests occur before improvements in probing accuracy.** Each point marks the mean checkpoint in training that AZ started to learn (or converge upon) the behavioral (or probing) test. Error bars are standard deviation across random seeds and short- vs long-term. We group the concepts as in Fig. 2. While behavioral tests start to improve before the probing, they both converge near the end of training. Exception: the ladder escape and bottleneck concepts (Fig. 2h) are easy for the probes to detect, converging early in training, with the behavior only improving late in training. The concept neighbors, (§4.4), which evaluates how well AlphaZero's cell embeddings capture Hex's neighborhood structure, is learned and converges in a similar time frame as the probing task.

4.4 AlphaZero's Cell Embeddings Capture the Structure of the Board

Understanding Hex's concepts requires understanding the board's structure, i.e., which cells connect to which other cells. AlphaZero (AZ), with its feed forward network architecture, does not *a priori* represent this structure. Fig. 5 shows evidence that some information is learned before the model is able to use the concepts. A possibility is that AZ spends the initial portions of training building up a representation of the board. However, we find no evidence that the neighborhood structure is learned in the initial stages of training, rather, it appears to be learned only after the first improvements in behavioral tests (Fig. 6).

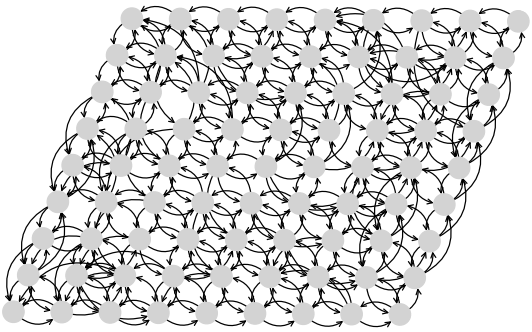


Figure 7: **Implicit board structure.** Arrows mark the learned nearest neighbors of each cell.

To study if Hex's board is represented in AZ, we hypothesized that the structure of Hex's board is implicitly learned by AZ's first layer. For each Hex cell, we extract a cell embedding from the first layer of AZ.[4] Next, we compute the dot-products between each cell embedding. The dot-product score between ground-truth neighbors increases throughout AZ's training.

The nearest-neighbors (according to these dot-product scores) nearly match the ground-truth by the 15th of 20 checkpoints, and eventually match the ground-truth before deviating slightly. We evaluate how well the dot product scores align with the ground truth cell distances via a ranking metric, Normalized Discounted Cumulative Gain (NDCG). Interestingly, the improvements first occur about 50% of the way through training as highlighted in Fig. 6. Quantitative results and the progressions of the learned structure are presented in our Supplementary Material.

5 Related Work

Applying MCTS and Deep Learning to the Game of Hex. [26], whose trained agents we use, studied scaling laws between various parameters, finding a relationship between compute, board size, and desired agent performance. This type of result can help others make informed tradeoffs when

[4] AlphaZero's first layer takes a flattened 1-hot encoding of the board and matrix multiplies this with a weight matrix. The embedding of each cell can be spliced out of that weight matrix. We only extract the cell embeddings for the current player. This constitutes a structural probe [21]. A structural probe tests the relationship between neural network activations (or weights).

training board game playing agents. Where [12] and [51] both use value functions in concert with MCTS to play Hex, [23] (MoHex) combine programmatic connection detection, pattern matching, and MCTS, solving Hex for the 9x9 grid.

AlphaZero, anecdotally, uses concepts, but only achieves a partial mastery. Domain experts have observed that AlphaZero [49] and related agents that combine Monte Carlo tree search [7] with deep reinforcement learning use identifiable concepts within board games. Michael Redmond identifies several common gameplay concepts demonstrated by AlphaZero [9] in AlphaZero's Go matches against Lee Sedol. Silver et al. [50] noted that common human-played corner moves in Go—*joseki*—were used by AlphaZero during self-play training before being abandoned, presumably as other more effective moves were found. Additional analysis provided by Tian et al. [54] note that Elf OpenGo only partially mastered ladder sequences within the game. Chess commentator Antonio Radić [45] detailed how AlphaZero used *zugzwang* [59] in the course of defeating Stockfish [46]. Experts have already incorporated some of AlphaZero's innovations into their play [40, 47].

Probing neural networks for human concepts. A range of linguistic concepts have been detected in NLP models using probing models [8, 42, 35, 53, 21]. For example, [53], with probing models, found that BERT appears to best encode linguistic concepts in the same order those concepts might be processed by the typical NLP pipeline. There has been some discussion on what good probing accuracy signifies. [20] calls for baseline controls, and to measure only the gain in accuracy compared to these baselines. [56] found that measures beyond accuracy, namely Minimum Description Length (MDL) – which is used to measure how easy it to detect a given concept, provided more stable results.

Understanding reinforcement learning agents trained to play board games. Sadler and Regan [47] covers how some particulars from AlphaZero's gameplay has impacted high-level chess. Concurrent to our work, McGrath et al. [37] also looks at how AlphaZero acquires game knowledge (which we term concepts) in chess. Using similar probing techniques, and the original (and larger) AlphaZero, they find that AlphaZero learns to encode many of the prototypical chess concepts. Related, but not focused on model understanding persay, Jhamtani et al. [24] collect an annotated set of chess games. This type of resource is similar to what is used by McGrath et al. [37] for their probing task. In our work, we focus on controlled programmatic generation of boards with (and without) concepts present. While these board instances may not occur during selfplay, the human annotated games are also off-distribution with respect to AlphaZero. Beyond finding similar results in a different game (Hex vs chess), we also leverage behavioral tests, tying the concept representation to agent behavior.

6 Discussion

Our analyses suggest that AlphaZero (AZ) learned to both represent and use concepts that humans consider important when playing Hex. However, we found that AZ is sometimes either blind or agnostic to a winning move. This had interesting ramifications. For instance, the negative concepts, especially dead cells, were not as well-encoded or used compared to other concepts. This may be because AZ has no "qualms" with playing wasted moves, like dead cells, at the end of the game.

A layerwise analysis showed that the same concept is represented most strongly in different layers, depending on its context: short-term concepts that inform actions at the end of the game are encoded in the upper layers of the model, whereas long-term concepts are absorbed deeper into the network. This absorption mirrors how AZ's policy head was trained to predict the policy outputs of MCTS.

Combining both representational and behavioral approaches to analyze reinforcement learning agents allows for a fuller understanding of how agents learn. Studying the representations of concepts allows us to ask (and answer) a rich set of questions about where those concepts reside. Studying the behavior of the model with respect to a given concept tests that this representation can be translated into action. Behavioral tests can also expose heuristics the model may be using. They are complementary approaches. In future work, applying a causal approach to the study of agent's policies will further illustrate how well these agents understand these concepts.

References

[1] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. *ICLR Workshops*, 2017.

[2] M. A. Badgeley, J. R. Zech, L. Oakden-Rayner, B. S. Glicksberg, M. Liu, W. Gale, M. V. McConnell, B. Percha, T. M. Snyder, and J. T. Dudley. Deep learning predicts hip fracture using confounding patient and healthcare variables. *NPJ Digit Med*, 2:31, Apr. 2019.

[3] Y. Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[4] Y. Belinkov, S. Gehrmann, and E. Pavlick. Interpretability and analysis in neural nlp. In *Proceedings of the 58th annual meeting of the association for computational linguistics: tutorial abstracts*, pages 1–5, 2020.

[5] M. G. Bellemare, S. Candido, P. S. Castro, J. Gong, M. C. Machado, S. Moitra, S. S. Ponda, and Z. Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, Dec. 2020.

[6] Y. Björnsson, R. Hayward, M. Johanson, and J. van Rijswijck. Dead cell analysis in Hex and the Shannon game. In *Graph Theory in Paris*, pages 45–59. Springer, 2006.

[7] B. Brügmann. Monte Carlo go. Technical report, Max Planck Institute of Physics, 1993.

[8] A. Conneau, G. Kruszewski, G. Lample, L. Barrault, and M. Baroni. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1198. URL `https://www.aclweb.org/anthology/P18-1198`.

[9] DeepMind. Match 1—Google DeepMind challenge match: Lee Sedol vs AlphaGo, Mar. 2016.

[10] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv*, Feb. 2017.

[11] D. Gale. The game of hex and the brouwer Fixed-Point theorem. *Am. Math. Mon.*, 86(10): 818–827, Dec. 1979.

[12] C. Gao, R. Hayward, and M. Müller. Move prediction using deep convolutional neural networks in hex. *IEEE Transactions on Games*, 10(4):336–343, 2017.

[13] M. Gardner. *The 2nd Scientific American Book of Mathematical Puzzles and Diversions*. Simon & Schuster, 1961.

[14] J. Gauci, E. Conti, Y. Liang, K. Virochsiri, Y. He, Z. Kaden, V. Narayanan, X. Ye, Z. Chen, and S. Fujimoto. Horizon: Facebook's open source applied reinforcement learning platform. Nov. 2018.

[15] J. Gauthier, J. Hu, E. Wilcox, P. Qian, and R. P. Levy. Syntaxgym: An online platform for targeted evaluation of language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, page 70–76, 2020.

[16] M. Geva, R. Schuster, J. Berant, and O. Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.

[17] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3389–3396. ieeexplore.ieee.org, May 2017.

[18] S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. R. Bowman, and N. A. Smith. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.

[19] R. Hayward, Y. Björnsson, M. Johanson, M. Kan, N. Po, and J. Van Rijswijck. Solving 7× 7 hex with domination, fill-in, and virtual connections. *Theoretical Computer Science*, 349(2): 123–139, 2005.

[20] J. Hewitt and P. Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1275. URL `https://www.aclweb.org/anthology/D19-1275`.

[21] J. Hewitt and C. D. Manning. A structural probe for finding syntax in word representations. In *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. doi: 10.18653/v1/N19-1419. URL `https://www.aclweb.org/anthology/N19-1419`.

[22] J. Hu, J. Gauthier, P. Qian, E. Wilcox, and R. Levy. A systematic assessment of syntactic generalization in neural language models. In *Proc. of NAACL*. URL `https://www.aclweb.org/anthology/2020.acl-main.158`.

[23] S.-C. Huang, B. Arneson, R. B. Hayward, M. Müller, and J. Pawlewicz. MoHex 2.0: A Pattern-Based MCTS hex player. In *Computers and Games*, pages 60–71. Springer International Publishing, 2014.

[24] H. Jhamtani, V. Gangal, E. Hovy, G. Neubig, and T. Berg-Kirkpatrick. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1671, 2018.

[25] A. Jones. LICENSE at master · andyljones/boardlaw, 2021.

[26] A. L. Jones. Scaling scaling laws with board games. *arXiv preprint arXiv:2104.03113*, 2021.

[27] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[28] N. Kim and T. Linzen. Compositionality as directional consistency in sequential neural networks. In *Workshop on Context and Compositionality in Biological and Artificial Neural Systems*, 2019.

[29] D. King. Hall of hexagons. `https://www.drking.org.uk/hexagons/index.html`, 2004. Accessed: 2021-11-12.

[30] B. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proc. of ICML*, 2018. URL `https://arxiv.org/abs/1711.00350`.

[31] T. Linzen, E. Dupoux, and Y. Goldberg. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016. doi: 10.1162/tacl_a_00115. URL `https://www.aclweb.org/anthology/Q16-1037`.

[32] T. Linzen, E. Dupoux, and Y. Goldberg. Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *Transactions of the Association for Computational Linguistics*, 4: 521–535, 2016. URL `http://aclweb.org/anthology/Q16-1037`.

[33] C. Lovering, R. Jha, T. Linzen, and E. Pavlick. Predicting inductive biases of pre-trained models, 2021.

[34] E. Margolis, S. Laurence, et al. *Concepts: core readings*. Mit Press, 1999.

[35] R. Marvin and T. Linzen. Targeted syntactic evaluation of language models. *arXiv preprint arXiv:1808.09031*, 2018.

[36] R. T. McCoy, E. Pavlick, and T. Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, pages 3428–3448. Association for Computational Linguistics (ACL), 2020.

[37] T. McGrath, A. Kapishnikov, N. Tomašev, A. Pearce, D. Hassabis, B. Kim, U. Paquet, and V. Kramnik. Acquisition of chess knowledge in alphazero. *arXiv preprint arXiv:2111.09259*, 2021.

[38] K. Meng, D. Bau, A. Andonian, and Y. Belinkov. Locating and editing factual knowledge in gpt. *arXiv preprint arXiv:2202.05262*, 2022.

[39] A. Mirhoseini, A. Goldie, M. Yazgan, J. W. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, A. Nazi, J. Pak, A. Tong, K. Srinivasa, W. Hang, E. Tuncer, Q. V. Le, J. Laudon, R. Ho, R. Carpenter, and J. Dean. A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212, June 2021.

[40] P. H. Nielsen. The exciting impact of a game changer: When Magnus met AlphaZero. *New In Chess*, 2019.

[41] J. Pawlewicz, R. Hayward, P. Henderson, and B. Arneson. Stronger virtual connections in hex. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(2):156–166, 2014.

[42] A. Poliak, A. Haldar, R. Rudinger, J. E. Hu, E. Pavlick, A. S. White, and B. Van Durme. Collecting diverse natural language inference problems for sentence representation evaluation. *arXiv preprint arXiv:1804.08207*, 2018.

[43] A. Poliak, J. Naradowsky, A. Haldar, R. Rudinger, and B. Van Durme. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*, 2018.

[44] M. Popova, O. Isayev, and A. Tropsha. Deep reinforcement learning for de novo drug design. *Science Advances*, 4(7):eaap7885, July 2018.

[45] A. Radić. AlphaZero's "immortal zugzwang game" against Stockfish, 2017. URL `https://www.youtube.com/watch?v=lFXJWPhDsSY`.

[46] T. Romstad, M. Costalba, J. Kiiski, and others. Stockfish, 2008.

[47] M. Sadler and N. Regan. Game changer. *AlphaZero's Groundbreaking Chess Strategies and the Promise of AI. Alkmaar. The Netherlands. New in Chess*, 2019.

[48] M. Seymour. *Hex: A Strategy Guide*. 2019. URL `http://www.mseymour.ca/hex_book/hexstrat.html`.

[49] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587): 484–489, Jan. 2016.

[50] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419): 1140–1144, Dec. 2018.

[51] K. Takada, H. Iizuka, and M. Yamamoto. Reinforcement learning for creating evaluation function using convolutional neural network in hex. In *2017 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pages 196–201. IEEE, 2017.

[52] I. Tenney, D. Das, and E. Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL `https://aclanthology.org/P19-1452`.

[53] I. Tenney, P. Xia, B. Chen, A. Wang, A. Poliak, R. T. McCoy, N. Kim, B. V. Durme, S. Bowman, D. Das, and E. Pavlick. What do you learn from context? Probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=SJzSgnRcKX`.

[54] Y. Tian, J. Ma, Q. Gong, S. Sengupta, Z. Chen, J. Pinkerton, and L. Zitnick. Elf opengo: An analysis and open reimplementation of alphazero. In *International Conference on Machine Learning*, pages 6244–6253, 2019.

[55] N. Tomlin et al. Understanding game-playing agents with natural language annotations. *To Appear: ACL*, 2022.

[56] E. Voita and I. Titov. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.14. URL `https://www.aclweb.org/anthology/2020.emnlp-main.14`.

[57] A. Warstadt, A. Parrish, H. Liu, A. Mohananey, W. Peng, S.-F. Wang, and S. R. Bowman. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

[58] A. Warstadt, Y. Zhang, X. Li, H. Liu, and S. R. Bowman. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proc. of EMNLP*, 2020. URL `https://www.aclweb.org/anthology/2020.emnlp-main.16`.

[59] E. Winter. Zugzwang. `https://www.chesshistory.com/winter/extra/zugzwang.html`, 1997. Accessed: 2021-11-12.

[60] S. Witty, J. K. Lee, E. Tosch, A. Atrey, M. Littman, and D. Jensen. Measuring and characterizing generalization in deep reinforcement learning. *arXiv preprint arXiv:1812.02868*, 2018.

[61] A. Zhang, Y. Wu, and J. Pineau. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 3.4
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 3.4
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See 3.1
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 3.2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report one standard deviation above and below the mean in Fig. 4a and Fig. 6.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Our hardware used and compute estimates is described in Section 3.1. We will include a breakdown of compute costs in the Supplementary Materials.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We acknowledge the work of Jones [26] in Section 3
   (b) Did you mention the license of the assets? [Yes] See Section 3.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See Section 3.1
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] See Section 3.4
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]