Entity Multiplexing Through Activation Strength: Understanding goals in A Maze Solving Agent

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this work we provide an extensive analysis into the operations of a maze solving reinforcement learning agent trained in the Procgen Heist environment. We target this model because it presented a high degree of polysemanticity due to the fact that it has to target multiple different entities to succeed. By focusing on an agent that has to target multiple similar entities we hope to answer questions about how each of these entities might be processed by the network. Our main finding is that the signals related to the targeting of different entities are encoded at different activation strengths within a single channel in the network. These "steering channels" are often highly redundant, with large numbers of channels enabling precise agent steering, but often only within narrow ranges of activation values. We also discover a paradoxical ablation effect in which removing both steering channels and navigation circuits improves entity collection rates compared to partial ablation, suggesting unexpected interference between these systems. These findings demonstrate that amplitude-based multiplexing is a fundamental strategy for encoding multiple goals in RL agents, while our counterintuitive ablation studies suggest surprising specialization and informational dependencies within the network.

1 Introduction

Understanding how deep learning models perform their tasks is currently an unsolved problem in the field of AI. This state of affairs ensures that highly reliable, controllable, and understandable models remain out of reach. This is particularly the case in the field of reinforcement learning (RL) that has been somewhat neglected by the techniques of mechanistic interpretability (MI) whose focus has largely been on Large Language Models (LLMs). With techniques from RL being applied to frontier models to a greater extent, it is our belief that lessons learned from applying MI to RL based agents can yield generalizable insights that can improve our understanding of AI agents and neural networks in general.

This research builds on the work of Mini et al. [2023] in which precise control of a maze solving agent was achieved by intervening directly on activations within the network. The current contribution begins by extending their work to an environment that involves multiple competing entities that an agent needs to reach in sequence, using the Procgen Heist environment as the target of our research.

The Procgen Heist environment requires the agent to collect up to 3 keys and 3 locks before reaching the final goal, a gem. The order in which the keys need to be collected is always the same (blue, green, red). The environment is procedurally generated and might generate with any combination of no keys, or 1-3 keys. The size of the maze will also change depending on how many objects are included.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

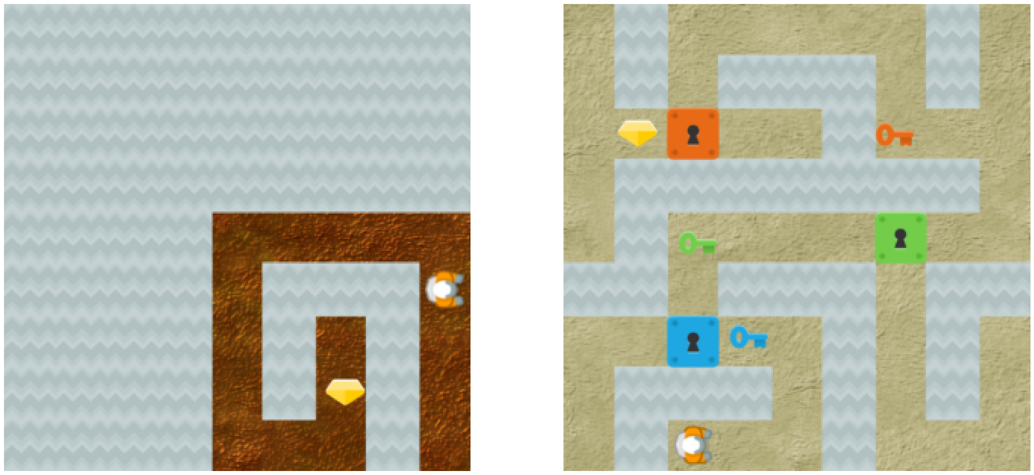

Figure 1: The Procgen Heist environment presents a procedurally generated maze where agents must collect keys and unlock corresponding doors in a specific sequence (blue, green, red) before reaching the final goal (gem). The environment varies in complexity, sometimes only having the gem present, and other times including multiple keys and locks.

The environment provides sparse rewards, only giving out a score of 10 if the agent makes it all the way to the final goal and 0 otherwise. The fact that the environment provides a variety of difficulties with some environments only having the gem and others including all 3 keys and locks establishes a natural curriculum for the agent. It also gives the agent a reasonable chance of reaching the end with a random walk before it starts to develop an actual policy.

Part of the reason we selected an environment from the Procgen suite for this task was that the Procgen environments naturally force the agent to learn sufficiently general representations of different entities to allow us to potentially extract them from the weights. If the model could simply encode easily memorized heuristics then this might lead to uninterpretable and highly specific rules rather than general representations of entities.

2 Related Work

2.1 Mechanistic Interpretability in RL

Previous work by Mini et al. [2023] demonstrated the ability to manipulate an agent's navigation in a simple maze environment with an intervention to a single channel in the network. This serves as a foundation for our work, though we extend it to a more complex multi-objective setting. The idea of activation steering originated in RL and was later successfully applied to LLMs Turner et al. [2024], demonstrating the potential for the transfer of techniques between very different neural network architectures and domains.

Work exploring the Procgen Coinrun environment provides a rich set of ideas for how to attribute attention from objects in the input to specific weights within the model, as well as visualizing the model weights Hilton et al. [2020]. They showed clear positive and negative attributions to specific entities according to how it might affect its ability to complete the task within the environment.

Other interpretability approaches for RL include saliency-based methods Greydanus et al. [2018], Atrey et al. [2020] which identify important input pixels but do not enable behavioral manipulation.

2.2 Ablation Studies and Network Interpretability

Systematic ablation studies have been fundamental to understanding CNNs since Zeiler and Fergus [2013] pioneered their use to identify important model components. However, recent work has revealed that networks exhibit surprising robustness to ablations McGrath et al. [2023]. While early interpretability work searched for individual neurons encoding specific concepts, Morcos et al. [2018]

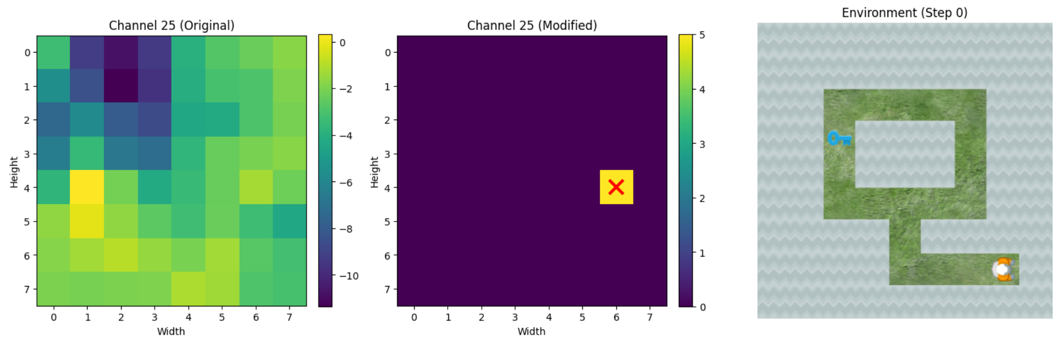

Figure 2: Example of the intervention experiment. We test channels to see the extent to which a single channel is capable of steering the agent to another location given the presence of a single entity in the environment. We consider an intervention a success if the agent enters the modified region.

demonstrated that networks achieving good generalization distribute information redundantly, with no single neuron being critical for performance.

Network Dissection Bau et al. [2017] assumes one-to-one neuron-concept mappings, but more recent work has demonstrated that neural networks are often polysemantic Elhage et al. [2022] and show that networks store multiple semantic concepts in superposition. Sparse Autoencoder research Bricken et al. [2023] successfully disentangles these polysemantic representations by training an autoencoder to separate semantic representations into a sparse overcomplete basis.

3 Methods

3.1 Model Architecture

We train a compressed version of the Impala model that uses 5 convolutional layers instead of 15 as used in the original Impala paper. This architecture comes from Hilton et al. [2020] where they find that this model was more interpretable with no discernible loss in performance. We include the model architecture in full in Appendix A.

When training our model we use a simple PPO implementation from an open source implementation designed to train Procgen models called Procgen-pfrl. We used the easy distribution of the environment due to compute limitations. We tested model training over a number of different batch sizes and distributions. The model checkpoints used to derive our main findings were trained with standard hyperparameters: learning rate 5e-4, 64 parallel environments collecting 256 steps each (16,384 total steps per update), processed in batches of 8 over 3 epochs for 800 million steps. The training dynamics were significantly impacted by changes to batch size often completely collapsing training performance, but over the course of 5 runs with separate seeds with the parameters above we were able to replicate similar model dynamics in each case.

3.1.1 Intervention experiments

When doing our experiments, we would place a specific entity into the environment. We then intervene on a single channel in a single layer by applying a zero mask to the channel setting it to zero and setting a specific region of the channel to a chosen value. In this experiment, we would do a sweep over a range of intervention values between 0 and the maximum values that the channel would reach during typical functioning as determined by sampling from the environment during operation. By running this sweep, we can get a sense of what the different channels do at different intervention strengths. We kept the length of the episode to 20 steps to ensure that the agent would either be able to reach the target zone, or the actual entity, but not both.

For each entity-channel combination, we ran 500 trials with the activation value incrementing linearly: $v_i = i \cdot \frac{v_{\max}}{N}$, where i is the trial index (1 to N), $N = 500$ is the total number of trials, and $v_{\max}$ is the maximum activation range.

We use a "Q" shaped maze with the design visible in Figure 2 because it was challenging enough that the agent would need to retain its navigational abilities to reach either target, while providing a clear decision point where the agent would need to choose between pursuing the original entity or our artificial target zone.

We test a single entity at a time in this way meaning that steering that is effective with a particular value on a given channel may not be effective for redirecting the agent in the presence of another entity.

To confirm that our results are robust, we also train Sparse Autoencoders (SAEs) Bricken et al. [2023] on the convolutional layers using techniques from Gorton [2024] to confirm that the results are not merely the result of polysemanticity.

3.2 Channel Ablation Study

To systematically assess the role of individual channels, we developed a comprehensive ablation methodology. For each channel in the network, we ran episodes with only that channel active while masking all other channels to zero. This allowed us to measure the isolated capability of each channel to support navigation to different entities in the environment. We recorded successful entity collection counts for each channel across 400 rollouts, providing a quantitative measure of each channel's specialization and capability. This methodology allows us identify which channels are sufficient for basic navigation. An example of the maze used is given in 3.

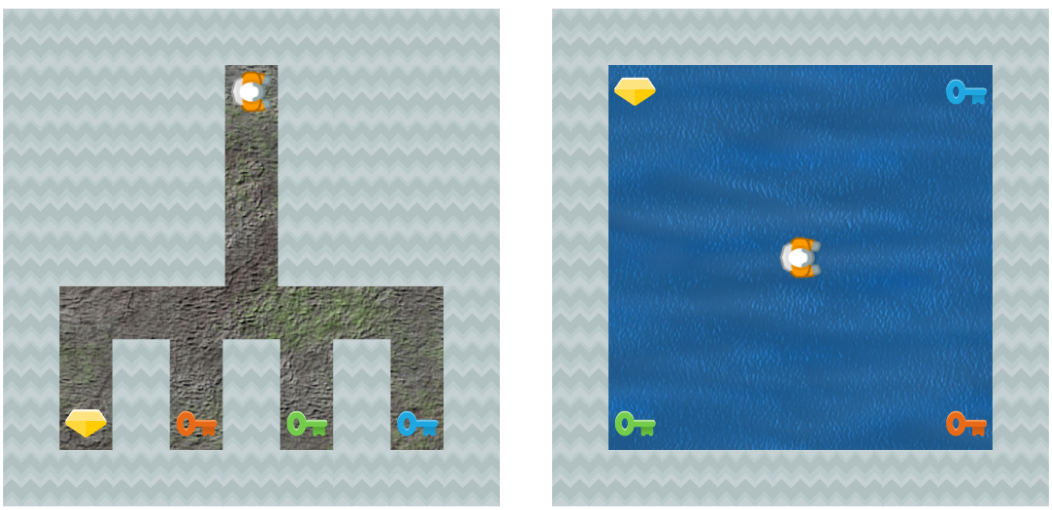

Figure 3: Maze configurations for ablation studies. (a) Fork maze tests individual channel navigation capability by requiring explicit path choices towards different entities and providing sufficient complexity that navigational abilities must be preserved. We orient the stem toward the model's inherent bias direction to minimize false positives. (b) Open maze with entities in corners tests the effect of ablating intervention spans without navigational constraints, ensuring only preferences regarding entity collection are observed.

3.3 Expanded ablation experiments

To derive further insights from our earlier results, we ablate regions of the network based on their success in modifying model behavior in the previous experiments. Our experiment uses a modified maze which removes all walls, and has the keys and gem randomly distributed in the corners of the maze.

We perform four targeted ablations based on our steering and navigation findings:

Ablation 1 (Intervention spans): We zero out any activations within the value ranges that successfully steered the agent. Specifically, for entity e and channel c, we set $h_{c,i,j} = 0$ whenever the activation falls within the successful steering range $\mathcal{S}_{e,c}$ identified in our intervention experiments.

Ablation 2 (Navigation channels): We completely ablate the top 10 channels identified as navigation-critical in our channel isolation study, setting these channels to zero while preserving all others.

Ablation 3 (Combined): We apply both ablations simultaneously, zeroing both the navigation channels and the intervention spans.

Ablation 4 (Inverse): We preserve only the intervention spans that successfully steered the agent in the presence of entity e, ablating everything else. This tests whether these spans alone are sufficient for navigation.

4 Results

All results presented are from checkpoint 35001 (35k update steps), well past convergence which occurred around 20k, unless otherwise noted.

4.1 Quantitative Steering Results

We apply our incremental steering experiments across all channels in conv3a and conv4a and find a large number of channels which successfully steer the agent away from a given entity. We present our results from conv4a here as its effective value ranges were more sparse and provided clearer results.

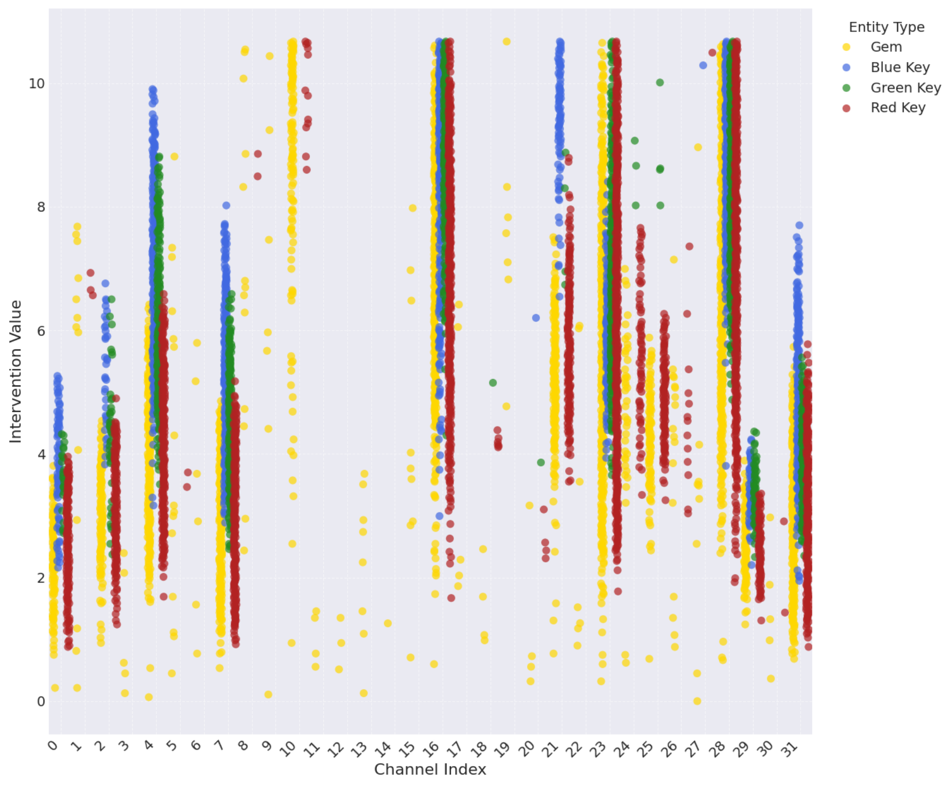

Figure 4: To determine the frequency with which a given channel can be used to steer the model away from a given entity, we create an artificial maze and place an artificial activation at a given point of the maze. We determine a successful intervention based on whether the agent makes contact with the region we specify. Each colored dot above represents a successful intervention with the specified entity as the target for the agent to move towards. Intervention success is partially localized to specific activation value ranges based on entity. This sample is from channel conv4a.

The most striking aspect of our results is the fact that particular value ranges worked for specific entities, illustrating how the channel seemed to signal the presence of a particular entity through the amplitude of the activation within a channel.

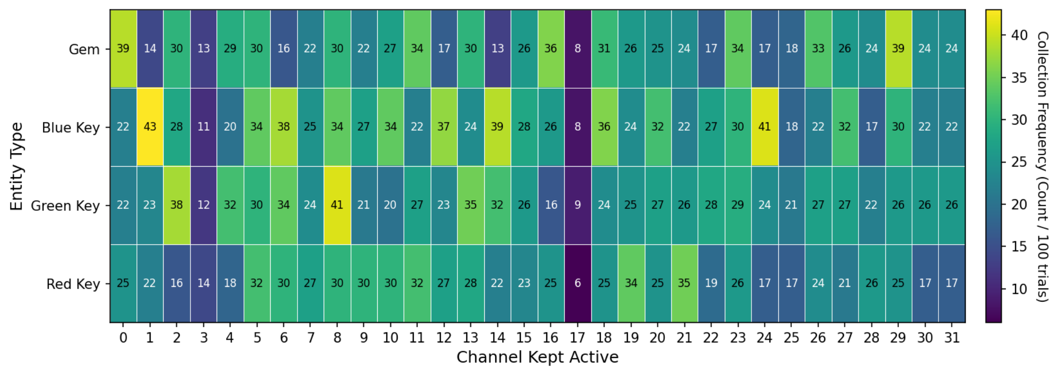

Figure 5: Heatmap showing entity collection counts when only a single channel is active (all others ablated). The vertical axis shows different entity types (gem, blue_key, green_key, red_key), while the horizontal axis shows channel indices. Brighter colors (yellow) indicate higher collection success rates.

We trained SAEs on these layers and found they exhibit identical multiplexing patterns, confirming this is a robust organizational strategy rather than the result of polysemanticity.

4.2 Channel Ablation Studies

Our ablation analysis reveals that the ability to navigate is surprisingly redundantly encoded throughout the network. The heatmap in Figure 5 shows the results in full. We observe that all channels have some success with reaching the entities some of the time with the best performing channels achieving success rates greater than 40% for certain entities, while others have lower than 10% success rates between entities. We also see that some channels show similar performance across entities, while others are significantly better at particular entities.

Table 1: Functional specialization between steering and navigation channels

Entity	Steering-only	Navigation-only	Overlap
Gem	8 channels	8 channels	2 (20%)
Blue Key	7 channels	7 channels	3 (30%)
Green Key	8 channels	8 channels	2 (20%)
Red Key	9 channels	9 channels	1 (10%)
Total unique	15 channels	19 channels	6 channels

4.3 Expanded ablation study

Our ablation studies reveal two counterintuitive findings.

First, ablating the gem's identified steering channels improves collection rates from 96.8% to 98.0%, while the same ablation reduces key collection by 43% on average. This suggests the gem may be encoded through the absence of key and lock signals, rather than through dedicated positive features. This broader intervention space suggests the network affords greater flexibility to gem-pursuit circuitry, as it can safely activate without competing with other navigation objectives once all preceding entities have been collected.

Second, we discover that partial ablation can be more harmful than complete ablation. When we ablate only navigation channels, gem collection drops to 34.0%. However, when we additionally ablate the intervention spans (removing more of the network) performance improves to 47.8%. This suggests an interference mechanism between the navigation and steering systems.

To further investigate the distributed nature of these representations, we performed inverse ablations where we preserved only the intervention spans for each entity. This revealed a clear preference of pursuit: blue key maintained 44% performance (well above the 26% random baseline), while

Table 2: Effect of targeted ablations on entity collection rates. Bold values indicate performance improvements or minimal degradation compared to baseline.

Entity	Baseline	Intervention Spans Only	Navigation Channels Only	Both	Preserve Only Intervention Spans	Random (Control)
Gem	96.8%	**98.0%**	34.0%	47.8%	20.0%	24.0%
Blue Key	98.0%	57.0%	**98.3%**	66.3%	40.0%	26.0%
Green Key	94.8%	52.0%	**96.3%**	57.0%	32.0%	22.0%
Red Key	80.0%	34.8%	**89.0%**	39.3%	28.0%	31.5%

gem performance dropped to 16% (below the 24% random baseline). This confirms that early-game entities have more localized representations while the final goal requires whole-network context.

5 Discussion

5.1 Solution Multiplicity and Representational Drift

A striking observation from our experiments was that intervention positions changed completely across different checkpoints even after convergence. Despite maintaining similar performance, the network continued exploring different encoding schemes within the solution space. This suggests that amplitude-based multiplexing is not a unique solution but rather one of many equivalent representations the network can adopt. The continuous drift between encoding schemes post-convergence indicates the network exists on a "valley floor" of equally viable solutions, constantly reorganizing its internal representations while preserving behavioral performance.

We find a notable checkpoint at update step 30001 where no steerable channels at all are found for any entity, though we only found 1 example of this in the 60 checkpoints we tested, suggesting it might be less optimal than those with the stronger intervention configuration.

This has meaningful implications for approaching understanding the inner workings of models. The specific channels and activation ranges we identify for steering are not fixed properties of the task but rather snapshots of a dynamic system exploring equivalent representations as possible solutions. This representational drift may explain some of the difficulty in creating robust interpretability tools and suggests that interventions may need to be continuously recalibrated as networks evolve, though the extent to which this phenomena occurs beyond RL is unknown.

5.2 Activation-Level Entity Encoding

Our findings regarding how the intervention spans encode multiple entities at different activation strengths reveal a sophisticated information compression mechanism. Rather than requiring separate channels for each entity type, the model has learned to use activation magnitude as an additional dimension for encoding information. This suggests that interventions on neural networks may need to consider not just which channels to modify, but also the precise activation values to use.

This activation-level encoding also helps explain why the model can maintain full functionality with so few active channels - the network has effectively learned to multiplex information within individual channels.

An explanation for why this happens might be that due to the similarity of the task of tracking each key, it proves the most efficient solution to have re-purpose channels already capable of entity detection in general, and to have them simply target each one sequentially, while indicating the nature of the given entity by varying the activation strength.

5.3 Role specialization

A surprising finding was that some channels seem able to unilaterally alter where the agent will navigate to, many other channels were more reliably able to navigate the agent when the other channels were ablated, but were completely unable to individually steer the agent, as evidenced by examining 4 and 5. This indicates that the specific function of indicating that the agent "must go to this spot" were exclusive to the steering channels, and the channels that were more successful in navigating had a role specifically in identifying how to get to a position, that in the absence of any other channels providing contrasting signals would lead the agent to go to that position.

This idea of specialization and reliance between channels is reinforced by the fact that ablating both the navigation channels and the intervention spans produced better collection rates than ablating the best navigation channels while leaving the intervention spans untouched. The fact that this occurred 52% of cases with n=400 trials is evidence that this effect is robust and not just noise.

Unfortunately we do not have a clear understanding of the precise nature of the intervention spans at this point, or what leads to the interference effect specifically, and hope to advance this question in future work.

6 Conclusion

Our work demonstrates that entities can be multiplexed within single channels in a neural network, and exploring these channels leads to surprising specialization within the network.

Our key findings include:

- Steering using a single channel worked in multiple layers (conv3a, conv4a) across checkpoints during training.
- While some channels seemed to both track certain entities and were able to affect steering, some channels were highly efficient at tracking, but couldn't modify the location the agent would go to, indicating specialized roles in the model.
- We found that the model was still able to navigate to different keys despite only having a single channel active.
- We found that there exist many regions within channels in the model that were capable of successful agent re-targeting in the presence of different entities at different activation strengths, revealing a mechanism for multiplexing information within a single channel.
- We uncovered surprising results showing that in some cases when ablating the best navigation channels it in fact improved performance to also remove the steering channels.
- We discover the surprising phenomenon of models slowly exploring a variety of global minima of possible solutions to the Procgen Heist environment without degrading performance as their changes occur.

These results show that there is specialization that occurs within the network both at the level of activation amplitude to signal specific entities, and also between channels where certain channels seem to play a role of directing the heading of the network while other channels provide navigational functionality to the agent.

7 Future Work

Promising directions for future work include:

- Better understand the precise mechanism of navigation within the network, and the mechanism of steering, and how these impact the next layer.
- Train a decision transformer or RNN style network that we can train probes on, and do targeted interventions in the residual stream.
- Better understanding the role of negative activations in agent navigation.
- Do further analysis on the difference in the treatment of keys and locks within the network.
- Techniques such as attribution-based parameter decomposition Braun et al. [2025] are also likely to yield valuable and potentially generalizable insights if applied to networks such as this one.

As RL techniques increasingly influence frontier models, understanding these fundamental organizational principles will likely provide valuable insights into building interpretable and controllable AI systems.

References

A. Atrey, K. Clary, and D. Jensen. Exploratory Not Explanatory: Counterfactual Analysis of Saliency Maps for Deep Reinforcement Learning, Feb. 2020. URL http://arxiv.org/abs/1912.05743. arXiv:1912.05743 [cs].

D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network Dissection: Quantifying Interpretability of Deep Visual Representations, Apr. 2017. URL http://arxiv.org/abs/1704.05796. arXiv:1704.05796 [cs].

D. Braun, L. Bushnaq, S. Heimersheim, J. Mendel, and L. Sharkey. Interpretability in Parameter Space: Minimizing Mechanistic Description Length with Attribution-based Parameter Decomposition, Feb. 2025. URL http://arxiv.org/abs/2501.14926. arXiv:2501.14926 [cs].

T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, R. Lasenby, Y. Wu, S. Kravec, N. Schiefer, T. Maxwell, N. Joseph, Z. Hatfield-Dodds, A. Tamkin, K. Nguyen, B. McLean, J. E. Burke, T. Hume, S. Carter, T. Henighan, and C. Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, S. Kravec, Z. Hatfield-Dodds, R. Lasenby, D. Drain, C. Chen, R. Grosse, S. McCandlish, J. Kaplan, D. Amodei, M. Wattenberg, and C. Olah. Toy models of superposition. *Transformer Circuits Thread*, Sept. 2022. URL https://transformer-circuits.pub/2022/toy_model/index.html. Publisher: Anthropic.

L. Gorton. The Missing Curve Detectors of InceptionV1: Applying Sparse Autoencoders to InceptionV1 Early Vision, June 2024. URL http://arxiv.org/abs/2406.03662. arXiv:2406.03662 [cs] version: 1.

S. Greydanus, A. Koul, J. Dodge, and A. Fern. Visualizing and Understanding Atari Agents. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1792–1801. PMLR, July 2018. URL https://proceedings.mlr.press/v80/greydanus18a.html. ISSN: 2640-3498.

J. Hilton, N. Cammarata, S. Carter, G. Goh, and C. Olah. Understanding RL Vision. *Distill*, 5(11): e29, Nov. 2020. ISSN 2476-0757. doi: 10.23915/distill.00029. URL https://distill.pub/2020/understanding-rl-vision.

T. McGrath, M. Rahtz, J. Kramar, V. Mikulik, and S. Legg. The Hydra Effect: Emergent Self-repair in Language Model Computations, July 2023. URL http://arxiv.org/abs/2307.15771. arXiv:2307.15771 [cs].

U. Mini, P. Grietzer, M. Sharma, A. Meek, M. MacDiarmid, and A. M. Turner. Understanding and Controlling a Maze-Solving Policy Network, Oct. 2023. URL http://arxiv.org/abs/2310.08043. arXiv:2310.08043 [cs].

A. S. Morcos, D. G. T. Barrett, N. C. Rabinowitz, and M. Botvinick. On the importance of single directions for generalization, May 2018. URL http://arxiv.org/abs/1803.06959. arXiv:1803.06959 [stat].

A. M. Turner, L. Thiergart, G. Leech, D. Udell, J. J. Vazquez, U. Mini, and M. MacDiarmid. Steering Language Models With Activation Engineering, Oct. 2024. URL http://arxiv.org/abs/2308.10248. arXiv:2308.10248 [cs].

M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks, Nov. 2013. URL http://arxiv.org/abs/1311.2901. arXiv:1311.2901 [cs].

A Technical Appendices

A Model Architecture Details

ImpalaCNN architecture:

- Input: Image (H,W,C) → Normalization [0,1] → Format adaptation
- Conv Block 1: Conv(C→16, 7×7) → ReLU → LPPool(2×2, s=2)
- Conv Block 2: Conv(16→32, 5×5) → ReLU → Conv(32→32, 5×5) → ReLU → LPPool(2×2, s=2)
- Conv Block 3: Conv(32→32, 5×5) → ReLU → LPPool(2×2, s=2)
- Conv Block 4: Conv(32→32, 5×5) → ReLU → LPPool(2×2, s=2)
- Flatten → Linear(flattened→256) → ReLU
- Linear(256→512) → ReLU
- Dual heads: Policy(512→num_outputs) & Value(512→1)

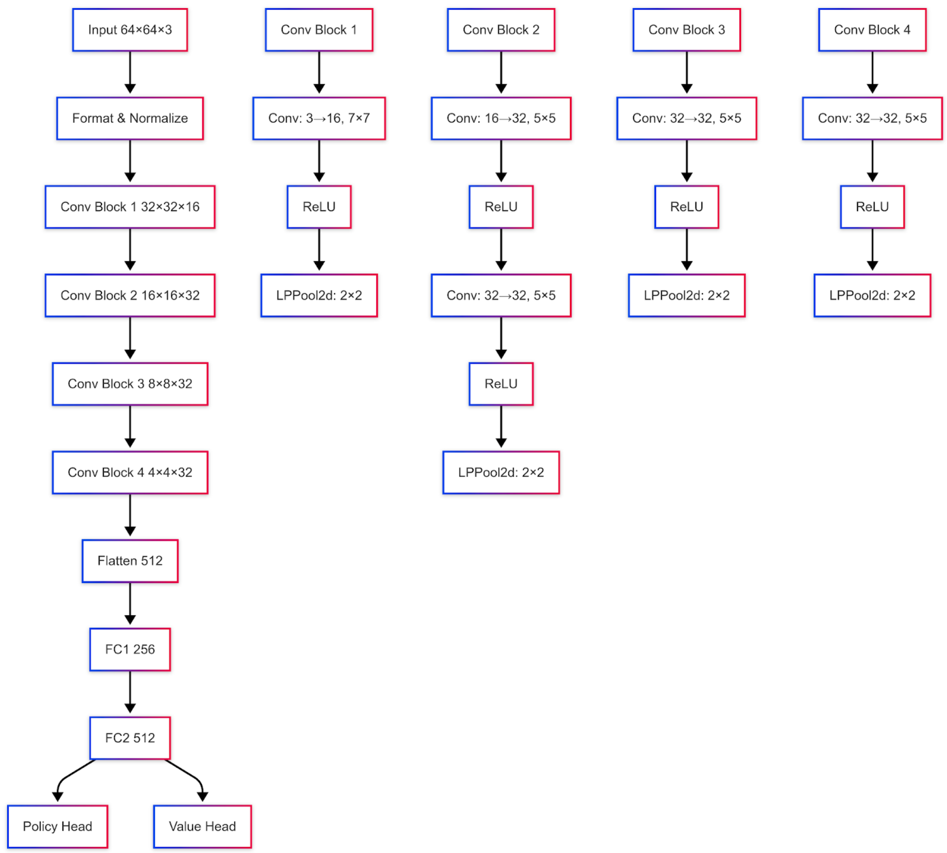

Figure 6: The convolutional neural network architecture used in our experiments. We use a modified version of the Impala CNN with 5 convolutional layers rather than the original 15 layers, which provides better interpretability without compromising performance.