Mining the Diamond Miner: Mechanistic Interpretability on the Video PreTraining Agent

Sonia Joseph[*]
Mila and McGill University
sonia.joseph@mila.quebec

Artem Zholus[*]
Mila and École Polytechnique de Montréal
artem.zholus@mila.quebec

Mohammad Reza Samsami[*]
Mila and Université de Montréal
mohammad-reza.samsami@mila.quebec

Blake A. Richards
Mila and McGill University
Learning in Machines and Brains Program, CIFAR
blake.richards@mila.quebec

Abstract

Although decision-making systems based on reinforcement learning (RL) can be widely used in a variety of applications, their lack of interpretability raises concerns, especially in high-stakes scenarios. In contrast, Mechanistic Interpretability (MI) has shown potential in breaking down complex deep neural networks into understandable components in language and vision tasks. Accordingly, in this study, we apply MI to understand the behavior of a Video PreTraining (VPT) agent, exhibiting human-level proficiency in numerous Minecraft tasks. Our exploration is centered on the task of diamond mining and its associated subtasks, such as crafting wooden logs and iron pickaxes. By employing circuit analysis, we aim to decode the network's representation of these tasks and subtasks. We find a significant head in the VPT model encoding for an attacking action, although its ablation doesn't markedly affect the agent's performance. Our findings indicate that this approach can provide useful insights into the agent's behavior.

1 Introduction

Deep RL has achieved remarkable results in a wide range of applications, from game playing to robotics and autonomous driving. However, these systems often lack interpretability and learn opaque representations, making it challenging to understand how they reason about their actions. This lack of transparency can be particularly concerning in high-impact applications where deep RL systems make decisions that influence human lives. As deep RL becomes increasingly deployed in decision-making systems, ensuring their safety and reliability has become a pressing issue (Amodei et al., 2016). Therefore, it is crucial to develop interpretability techniques that can help us understand how these systems make decisions and anticipate their behavior.

Mechanistic interpretability (MI) is a field that seeks to reverse-engineer artificial neural networks' computation into human-understandable components in a systematic way. Understanding the underlying mechanisms of networks is useful for gaining insight into how they behave, predicting their

[*] Equal Contribution

ATTRIB: Workshop on Attributing Model Behavior at Scale at NeurIPS 2023.

misalignments and errors, and addressing their out-of-distribution behavior. Circuit analysis (Olah et al. (2020)) is a common practice in MI that involves discovering a minimal induced subgraph in the computational graph responsible for accomplishing a task. Recently, Wang et al. (2022) attempted to search for circuits in a commonly used language model, the Generative Pretrained Transformer (GPT; Radford et al. (2017)). They proposed an end-to-end approach that can find a circuit capable of performing a natural language task called indirect object identification (IOI) in GPT-2 small (Radford et al., 2019).

In this work, we aim to understand a transformer-based RL agent inspired by the circuit discovery method in Wang et al. (2022). To do so, we conducted a study on the Video Pretraining (VPT; Baker et al. (2022)) agent, which is a semi-supervised imitation learning agent that uses a vast quantity of unlabeled video data of individuals playing Minecraft and is fine-tuned using RL. This agent can exhibit human-level performance for many tasks, including crafting diamond tools. Since its policy is a transformer, it is a good candidate to be studied by the framework introduced in Wang et al. (2022).

Our focus in this study is the task of mining diamonds, as well as the subtasks such as crafting wooden logs and iron pickaxes. Collecting diamonds is a very challenging task which takes about 24000 actions on average. The goal is to use a circuit analysis to uncover how the network is representing its tasks and subtasks.

2 Background

2.1 Video PreTraining Model

In a standard reinforcement learning (RL) setting, an agent interacts with an environment by taking actions based on the current observation from the environment. The environment responds to the agent's actions with a new observation and a reward signal, which indicates the desirability of the action taken by the agent in that state. The goal of the agent is to learn a policy, which is essentially its behavior function, that maximizes the expected cumulative reward over time.

Imitation learning offers an alternative approach, where the goal is to construct a policy that mimics a demonstrator's behavior based on a dataset of action-observation pairs. This method can be particularly effective in scenarios where we have access to a large dataset, allowing us to reduce the number of expensive interactions between the agent and the environment needed for RL.

To make use of the vast amount of video data available on the internet, a new semi-supervised imitation learning method called Video PreTraining (VPT) has been introduced. Unlike language modeling, where the next words in a sentence serve as action labels, videos do not come with precise action labels indicating how a particular task was accomplished. VPT aims to overcome this challenge by learning from large amounts of unlabeled video data, allowing us to build large-scale foundation models. To exploit the benefits of RL, VPT is also fine-tuned by specifying a reward function to solve challenging tasks, such as crafting a diamond pickaxe, which takes humans more than 20 minutes on average.

The architecture of VPT begins with a vision module consisting of a convolutional layer and ResNet blocks. The resulting output is then directed to a policy that comprises four subsequent residual transformer blocks; each block consists of an attention layer followed by two dense layers that operate frame-wise. The last component of the architecture is a final embedding layer. We opted to analyze the VPT 2x model in this work, which it is the second largest model and includes 16 attention heads in each block. The aim of this research is to employ the framework introduced by Wang et al. (2022) to examine the attention heads of every block.

2.2 Circuit Discovery

Our goal, in the context of mechanistic interpretability, is to establish a connection between the internal components of a model and its resulting behavior. We drew inspiration from the reverse-engineering circuit approach outlined in Wang et al. (2022). In this approach, a "circuit" C is defined as a subgraph of a computational graph M that is responsible for executing a specific task. Since we can express the overall functionality of the model on a given input x as $M(x)$, Wang et al. (2022) proposed a method to associate a function with C by "knocking out" the nodes outside of the circuit.

To accomplish this knockout, we must disable certain nodes without affecting the functionality of those within C. Therefore, the authors introduced a technique known as "mean ablation". This method involves replacing each node with the average activation value across a predetermined reference distribution.

To measure VPT's performance, we look at the *logit difference* between "Attack"[2] and "Non-Attack" actions (see Appendix, Figure 4). To quantify the importance of a particular attention head to the task, we look at the new logit difference between attack and non-attack actions after ablating that head.

3 Methodology

3.1 Tasks and Dataset

Mining diamonds in Minecraft presents a significant challenge for an autonomous agent and involves obtaining a sequence of specific items, each requiring a distinct set of actions. To accomplish this task within a typical RL framework, the agent must tackle the hard problem of long-term credit assignment (Harutyunyan et al., 2019). Therefore, the objective of this research is to interpret how VPT's inner dynamics lead to the completion of this complex task.

In this study, we collected the dataset by deploying the VPT-RL agent (i.e., fine-tuned via RL) in Minecraft and collecting environment samples until it gets a diamond. In total, we collected 389 episodes of approx. 7800 environment steps, resulting in nearly 3 million environment samples or 41 hours of gameplay. The dataset consisted of image observation for the agent, rewards, and actions taken in the environment. An example of milestones of each episode is shown in Figure 1.

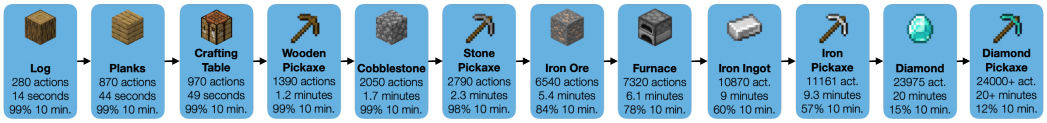


Figure 1: Task sequence to get a diamond in Minecraft. Figure taken from Baker et al. (2022)

3.2 Attacking Head

We computed the attacking action probability in the following way. In VPT, the keypress probability distribution is implemented as a joint head outputting the probability in the product space of several categorical keypress actions. This means that several logit bins can correspond to doing attack action in the game. Therefore, we simply compute the *logsumexp* aggregate of all logit bins that correspond to doing an attack and also we do so separately for not doing an attack action. Then we subtract the *not attack* logit from the *attack* logit forming the *attack logit difference*.

Using the dataset we collected, we conducted mean ablation to assess the impact of each head on the attack action; for each head, we calculated its direct effect on the attack action's logit.

4 Results and Discussion

4.1 Attack Probability Changer Head

We found that ablating the attention head 9 of layer 0 of the transformer consistently decreases the logit difference for all attack tasks (Figure 2a). Further, we found that ablating the same head increases the logit difference for all non-attack tasks (Figure 2b). We call this head the *Attack Probability Changer Head.*[3]

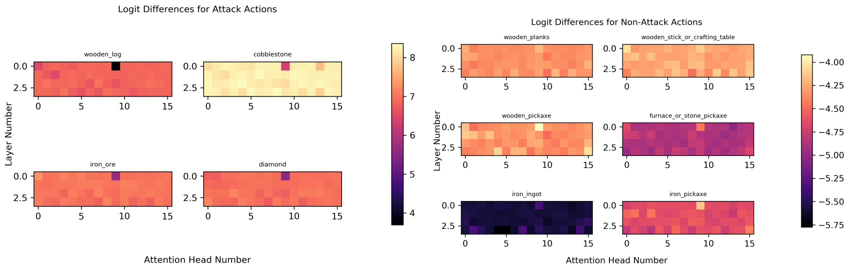
[2] The agent/user either grabs a set of items or puts them down in a pile. Additionally, if the agent/user double-clicks on the items, the system will perform a series of actions (attack, no attack, attack) and merge all of the items of the same type that are currently in the user's inventory into a single stack.

[3] For a timeseries of the Attack Probability Changer Head's logit difference as the agent interacts with the environment, go to: https://drive.google.com/file/d/15UUwwgjV20ulaa0O0ptZX$_{y}IOwLF8-wO/view?usp=sharing$

We find that the logit difference of the ablated Attack Probability Changer Head is statistically significant compared to the logit difference of the non-ablated vanilla model, and compared to the average logit difference of all the other attention heads (see Appendix, Figure 5).

4.2 Effect of Head Ablation on Performance

Interestingly, when we ablate the Attack Probability Changer Head and re-run the agent on the same environment, the Attack Probability Changer Head does not impact *performance* in a statistically significant way (Figure 3). Performance is measured as steps until the task is completed. It is possible that this head impacts behavior in other ways, such as making the agent less likely to choose "attack" when there are multiple options presented. Further, it is possible that the Attack Probability Changer Head encodes for attack information in a circuit of other heads, which we could find by ablating the remaining attention heads in conjunction with this one.



(a) (b)

Figure 2: Logit difference (attack - non-attack) for the transformer throughout the trajectory after running mean ablations for each attention head. The y-axis is the layer number of the transformer (for a total of four layers), while the x-axis is the total number of attention heads (for a total of 16 attention heads). (a) Mean-ablating Head 9 of Layer 0 decreases the attack probability significantly for all four attack tasks. (b) Logit difference for non-attack-related tasks. Here, the mean ablation on Layer 0, Head 9 significantly *increases* the logit probability for attack.

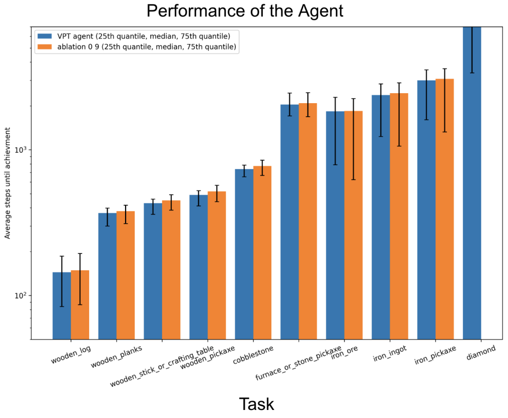


Figure 3: The performance of the vanilla and ablated agent when run on the environment. Performance is measured as the number of steps until the task is completed. The ablated agent's performance is not statistically significant compared to the vanilla agent's.

Future work includes running the same study on many actions beyond attack (e.g. perhaps there is a different attention head for each action). There is also the possibility that the Attack Probability Changer Head is operating in conjunction with other heads to encode for the attack action, and we can continue ablating heads to see if they comprise a circuit.

While our analysis illuminated some information processing of this particular VPT transformer agent, our method of circuit tracing by ablating head-by-head is unlikely to scale to very large neural nets.

Future steps should explore methods that scale to large neural networks, perhaps drawing upon practices from neuroscience and causal representation learning, and moving toward a more general theory of when and why neural networks store a single concept in a single neuron.

5 Conclusion

We found a statistically significant head on the VPT model that encodes for an attacking action. However, while ablating the head changes the logit probability, it does not impact the performance of the agent. More work remains to be done to understand the encoding of information in the VPT transformer.

Acknowledgements

This research was generously supported by the Bank of Montreal (S.J.); NSERC (Discovery Grant: RGPIN-2020-05105; Discovery Accelerator Supplement: RGPAS-2020-00031; Arthur B. McDonald Fellowship: 566355-2022) and CIFAR (Canada AI Chair; Learning in Machine and Brains Fellowship). The authors acknowledge the material support of NVIDIA in the form of computational resources.

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. (2016). Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

Baker, B., Akkaya, I., Zhokov, P., Huizinga, J., Tang, J., Ecoffet, A., Houghton, B., Sampedro, R., and Clune, J. (2022). Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35:24639–24654.

Harutyunyan, A., Dabney, W., Mesnard, T., Gheshlaghi Azar, M., Piot, B., Heess, N., van Hasselt, H. P., Wayne, G., Singh, S., Precup, D., et al. (2019). Hindsight credit assignment. *Advances in neural information processing systems*, 32.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. (2020). Zoom in: An introduction to circuits. *Distill*. https://distill.pub/2020/circuits/zoom-in.

Radford, A., Jozefowicz, R., and Sutskever, I. (2017). Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. (2022). Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.

A Appendix

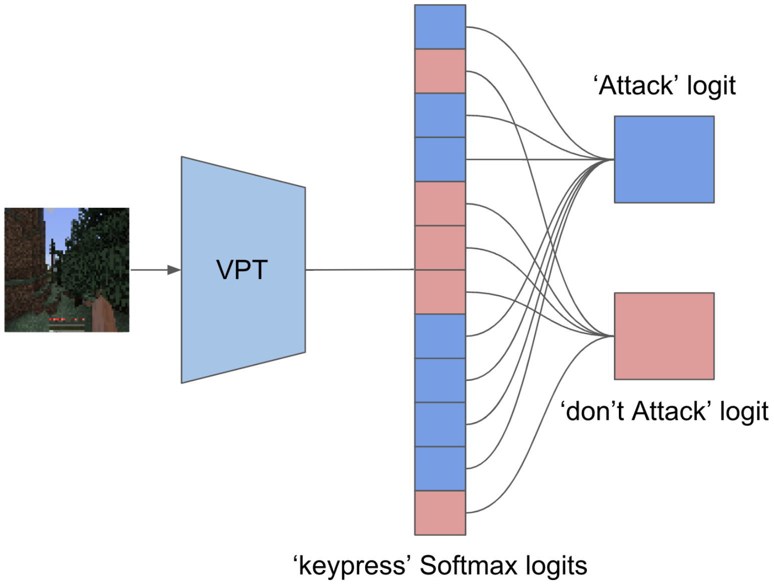


Figure 4: The logit difference between the attack and non-attack actions. A more positive logit-difference means that an attack is more likely than a non-attack. For more details, see Section 2.2

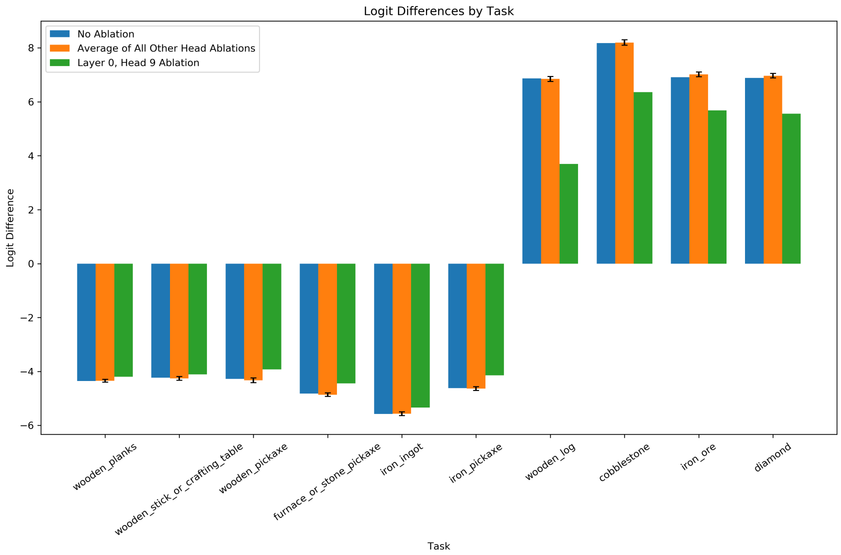


Figure 5: The logit difference by task for non-attack and attack actions for the zero-ablation vanilla agent (blue), the average of each other ablated attention head (orange), and the ablated Attack Probability Changer Head (green). The ablated Attack Probability Changer Head's logit difference is statistically significant compared to the logit difference of both the no-ablation model and the average of the other attention heads. For more details, see Section 4.2.