Explainable Concept Generation through Vision-Language Preference Learning

Aditya Taparia
Arizona State University
ataparia@asu.edu

Som Sagar
Arizona State University
ssagar6@asu.edu

Ransalu Senanayake
Arizona State University
ransalu@asu.edu

Abstract

Concept-based explanations have become a popular choice for explaining deep neural networks post-hoc because, unlike most other explainable AI techniques, they can be used to test high-level visual "concepts" that are not directly related to feature attributes. For instance, the concept of "stripes" is important to classify an image as a zebra. Concept-based explanation methods, however, require practitioners to guess multiple candidate concept image sets, which can often be imprecise. Addressing this limitation, in this paper, we frame concept image set creation as an image generation problem. However, since naively using a generative model does not result in meaningful concepts, we devise a reinforcement learning-based preference optimization algorithm that fine-tunes the vision-language generative model from approximate textual descriptions of concepts. Through a series of experiments, we demonstrate the capability of our method to articulate complex, abstract concepts that are otherwise challenging to craft manually.

1 Introduction

In an era where black box deep neural networks (DNNs) are becoming seemingly capable of performing general enough tasks, our ability to explain their decisions post-hoc has become even more important before deploying them in the real world. Humans utilize high-level concepts as a medium for providing and perceiving explanations. In this light, post-hoc concept-based explanation techniques, such as Testing with Concept Activation Vectors (TCAV) [1], have gained great popularity. Their ability to use abstractions that are not necessarily feature attributes or some pixels in test images helps with communicating these high-level concepts with humans. For instance, as demonstrated in TCAV, the concept of stripes is important to explain why an image is classified as a zebra.

Although concept-based XAI methods are a good representation, their requirement to create collections of candidate concept sets necessitate the human to know which concepts to test for. This is typically done by guessing what concepts might matter and manually extracting such candidate concept tests from existing datasets. While the stripe-zebra analogy is attractive as an example, where it is obvious that stripes is important to predict zebras, in most applications, we cannot guess what concepts to test for, limiting the usefulness of concept-based methods in testing real-world systems. Additionally, even if a human can guess a few concepts, it does not encompass most concepts a DNN has learned because the DNN was trained without any human intervention. Therefore, it is important to automatically find human-centric concepts that matter to the DNN's decision-making process.

As attempts to automatically discover and create such concept sets, several work has focused on segmenting the image and using these segments as potential concepts, either directly [2] or through factor analysis [3, 4]. In such methods, which we call as retrieval methods, because the extracted concept set is already part of the test images, it is difficult for them to imagine new concepts that do not have a direct pixel-level resemblance to the original image class.

Interpretable AI: Past, Present and Future Workshop at NeurIPS 2024

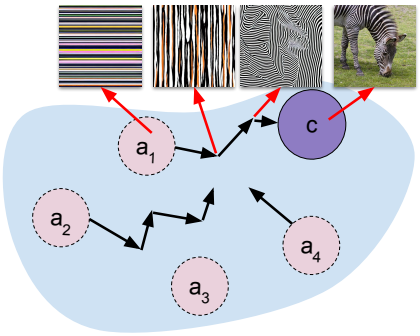

Figure 1: Our algorithm, RLPO, gradually updates stable diffusion weights (black arrows) while selecting the potential text prompts (a). Each step in this update process provides an explanation at a different level of abstraction.

For instance, it is arguable if some patches of zebra—instead of stripes—qualify as high-level concepts to explain the zebra class.

By departing from existing concept set creation practices of human handcrafting and retrieval, we redefine concept set creation as a concept generation problem. Modern generative models such as stable diffusion (SD) can produce noise-free, realistic images. Nevertheless, since a generative model can generate arbitrary images, we need to guide it to produce what we desire by using text prompts. One obvious approach is to engineer long, descriptive text prompts to generate concepts. However, engineering such prompts is not realistic. Therefore, to automate prompting, we extract keywords related to the image using an image-to-text model (we call them seed prompts). We propose a reinforcement learning-based preference optimization (RLPO) algorithm that guides the generative model to automatically generate meaningful concepts based on these seed prompts. Preferences are solely decided by the explanation score—not by a human—that the deep RL (DRL) algorithm is trying to optimize.

2 Preliminaries and Related Work

Testing with Concept Activation Vectors (TCAV): The TCAV score quantifies the importance of a "concept" for a specific class in a DNN classifier [1]. Here, a concept is defined broadly as a high-level, human-interpretable idea such as stripes, sad faces, etc. A concept (e.g., stripes), c, is represented by sample images, X_c (e.g., images of stripes). For a given set of test images, X_m (e.g., zebra images), that belongs to the same decision class (e.g., zebra), m, TCAV scores (TS) is defined as the fraction of test images for which the model's prediction increases in the direction of the concept. By decomposing the DNN under test as $f(x) = f_2(f_1(x))$, where $f_1(x)$ is the activation at layer l, TCAV score is computed as,

$$\begin{aligned} TS_{c,m} &= \frac{1}{|X_m|} \sum_{X_m} \mathbb{I}\left(\frac{\partial \text{output}}{\partial \text{activations}} \cdot (c \text{ direction}) > 0 \right) \\ &= \frac{1}{|X_m|} \sum_{x_i \in X_m} \mathbb{I}\left(\frac{\partial f(x_i)}{\partial f_1(x_i)} \cdot v > 0 \right) \end{aligned} \tag{1}$$

Here, $\mathbb{I}$ is the indicator function that counts how often the directional derivative is positive. Concept activations vector (CAV), v, is the normal vector to the hyperplane that separates activations of concept images, $\{f_1(x); x \in X_c\}$, from activations of random images, $\{f_1(x); x \in X_r\}$.

ACE [2] introduced a way to automatically find relevant concepts by extracting them from the input class. It uses image segmentation of multiple sizes to get a pool of segments and then grouped them based on similarity to compute TCAV scores. Though the ACE concepts are human understandable, they are very noisy. EAC [5] extracts concepts through segmentation. CRAFT [3] introduced a recursive strategy to detect and decompose concepts across layers. Lens [4] elegantly unified concept extraction and importance estimation as a dictionary learning problem. However, since all these methods obtain concepts from test images, the concepts they generate tend to be very similar to the actual class. In contrast, we generate concepts from a generative model. Under generative models, LCDA [6] simply queries an LLM to get attributes but does not generate concepts.

Deep Q Networks (DQN): DQN [7] is a DRL algorithm that combines Q-learning with deep neural networks. It is designed to learn optimal policies in environments with large state and action spaces by approximating the Q-value function using a neural network. A separate target network, $Q_{\text{target}}(s, a', \theta')$, Here a' is $argmax Q(s_{next}, a)$ which is a copy of the Q-network with parameters θ', is updated less frequently to provide stable targets for Q-value updates,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(r(s_t, a_t) + \gamma \max_{a'} Q_{\text{target}}(s_{t+1}, a') - Q(s_t, a_t) \right). \tag{2}$$

Here, s_t is the state at step t, a_t is the action taken in state s_t, and r_t is the reward received after taking action a_t. The parameters α and γ are learning rate and discount factor, respectively. DQNs are used for controlling robots [8, 9, 10], detecting failures [11], etc.

Direct Preference Optimization (DPO): DPO [12] is a technique used to ensure models, such as large language models, learn to align its outputs with human preference by asking which of its generated output is preferred. In this study, preference is provided using TCAV score and it is used to align a text-to-image model to generates concepts with respect to DNN under test.

3 Methodology: Explainable Concept Generation

Algorithm 1 The RLPO algorithm.

```
Input: Set of test images, f(·)
Run pre-processing and get the seed prompts (action space)
for each episode do
    for each time step t do
        Execute a_t by picking a keyword
        Generate image groups G_1 & G_2
        Evaluate TCAV scores TS_1 & TS_2
        Update SD based on better score
        Compute reward
    end for
end for
Output: Set of concept images
```

Our objective is generating concept images, $\mathcal{C}$, that provide a higher TCAV score, $TS_{c,m}$. To this end, we leverage the state-of-the-art text-to-image generative models to generate high quality explainable concepts. However, because the search space of potential text prompts is too large, we use DRL to search for text that are relevant. Our algorithm, named reinforcement learning-based preference optimization (RLPO), picks a seed prompt using RL and optimizes stable diffusion weights to generate images that have a preference for higher TCAV scores. As summarized in Fig. 1 and Algorithm 1, we describe this process in the rest of this section. This work help engineers and other users of DNNs to obtain high-level visual explanations of under which conditions the neurons are activated in a DNN, making it a useful diagnostic tool.

Notation: Our framework contains multiple machine learning models. First, we have a pre-trained neural network classifier that we want to explain, which is indicated by $f(\cdot)$. We then have a generative neural network, $g(\cdot)$, whose purpose is generating concept image sets, given some text prompts. In this paper, we use Stable Diffusion (SD) v1-5 as the generator as it is a state-of-the-art generative model that can generate realistic images. If the weights of the SD model are w, for a small constant λ, we augment it as $w + \lambda ab$, where A and B are low-rank matrices that we fine-tune using DPO [12]. The core search algorithm that we train contains a DQN which is denoted by $h(\cdot)$. We use off-the-shelf visual question answering (VQA) to generate the action space (seed prompts) for the DQN. There is also a linear classification model that we train to extract explainable concept directions using TCAV.

3.1 The Deep Reinforcement Learning Formulation

Our objective of using DRL is automatically controlling text prompts used in the text-to-image generative model. As text prompts, we start with a list of keywords, $\mathcal{K}$, that have the potential to generate meaningful concept images after many DRL episodes. How we extract these keywords automatically will be discussed in detail in Section 3.2. We would like to highlight that these keywords act as seeds and they do not directly impact the accuracy of the DRL algorithm although good seeds will make the policy learning faster. We setup our RL state-action at iteration t as follows:

- **Action** a_t: Selecting a keyword, $k_t \in \mathcal{K}$, that best influences concept image generation
- **State** s_t: Concept images generated from the keyword prompt, k_{t-1}.

Our objective in DRL is to learn a policy, $\pi : s \rightarrow a$, that takes actions (i.e., picking seed prompt) leading to explainable states (i.e., correct concept images) from proxy states (i.e., somewhat correct concept images). This traversal is illustrated in Fig. 1 and formally defined as *explainable states* definition 1 and *proxy state* defination 2 in Appendix 1. In practice, we set the threshold η (used in defination 1 and 2) to a relatively large number, such as 0.7, to ensure that we look at highly meaningful concepts.

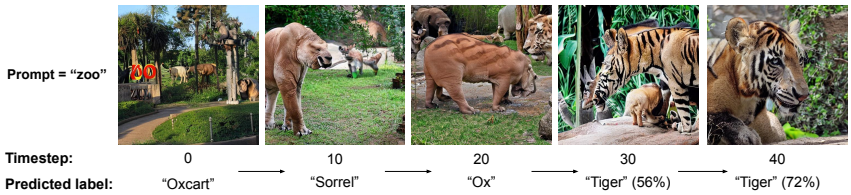

Figure 2: Different levels of abstractions for the class "Tiger." The generated image begins as a random image and gradually transition to images with tiger features.

In DQN, in relation to Eq. 2, we learn a policy that iteratively maximizes the $Q(s, a)$ value by using the update rule,

$$Q^*(s, a) = \mathbb{E}_{s' \sim P(\cdot|s,a)}[\xi_t r(s, a) + \gamma \max_{a' \in A} Q_{\text{target}}(s', a')], \tag{3}$$

where,

- **Reward** r_t: Reward r_t is proportional to the TCAV score computed at state s_t on action a_t, adjusted by a monotonically increasing scaling factor ξ_t. This factor is used to scale the reward over time t Since the $g(.)$ is getting optimized at each timestep t. The scaling factor is updated as $\xi_{t+1} \leftarrow \min(1, \xi_t + k)$. Therefore, the expected cumulative adjusted reward is $R(\pi) = \mathbb{E}[\sum_{t=0}^{T} \xi_t \cdot r_t(s_t, a_t)]$.

3.2 Extracting the Action Space (Seed Prompts)

Since a generative model can generate arbitrary images, if we provide good starting point for optimization then the convergence to explainable states would be faster. In this paper, to extract seed prompts for a particular class we make use of the off-the-shelf VQA model followed by several preprocessing steps.

We start by splitting each test image into 9 patches. These patch from the test images are then passed to the VQA model to extract relevant and useful information about the corresponding class. In this study, we choose BLIP [13] as our VAQ model. We posed a set of targeted questions to the VQA model, aiming to gain insights into the class-specific features represented in the patches. The questions are designed to probe various aspects of the image patches, helping the model focus on class-defining attributes. We then remove stop words and duplicates from the generated responses using lemmantizing and perform a cross-similarity check using CLIP between all the unique words and further filtered words which are more than 95% similar. To further select most relevant keywords to the class images, we perform a VLM check using class images and the extracted keyword to get the softmax score of how much the keyword and image are related. This score is then averaged over all the class images and this average is use to sort the keywords. Now, from the sorted keywords, we select top 20 keywords as our RL action space.

3.3 Optimizing the States

At time t, the policy picks the seed prompt k_t, which is then used by the generative model, $g(k_t; w_t)$, with model weights w, to generate $2Z$ number of images. We randomly divide the generated images into two groups: $X_{c_1,t} = \{x_{c_1,t,i}\}_{i=1}^{Z}$ and $X_{c_2,t} = \{x_{c_2,t,i}\}_{i=1}^{Z}$. Let the TCAV scores of each group be $TS_{c_1,m,t}$ and $TS_{c_2,m,t}$. Since our objective is to find concepts that generate a higher TCAV score, concept images that have a higher score is preferred. Unlike in the classical preference optimization setting our optimization comes from TCAV scores (e.g., $X_{c_1,t} \succ X_{c_2,t}$). If the generative model at time t is not capable of generating concepts that are in an explainable state (measured based on threshold η), $\max(TS_{c_1,m,t}, TS_{c_2,m,t}) \leq \eta$, we then perform preference update on SD's weights. Following Low-Rank Adaptation (LoRA) [14], we only learn auxiliary weights a and b at each time step, and update the weights as $w_{t+1} \leftarrow w_t + \lambda ab$.

As the DRL agent progresses over time, the states become more relevant as it approaches explainable states (Fig. 2), thus the same action yields increasing rewards over time. To accommodate this, with reference to the rewards defined in Section 3.1, we introduce a parameter ξ which starts at 0.1 and incrementally rises up to 1 as the preference threshold, η, is approached. Different actions

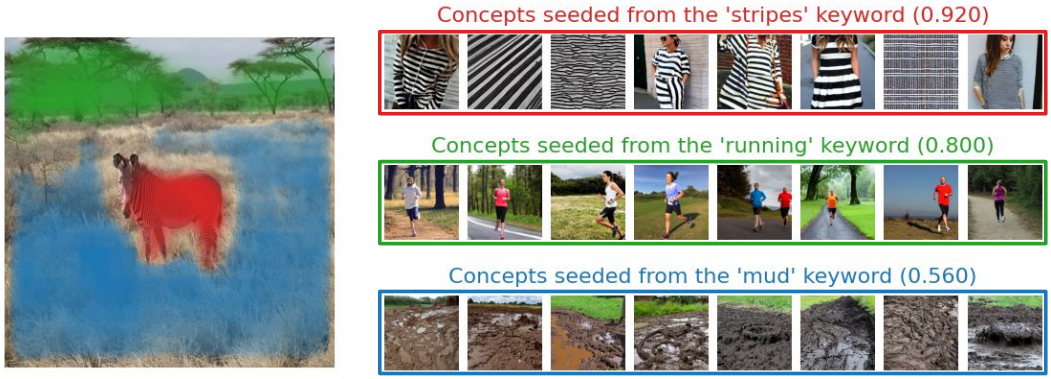

Figure 3: The figure shows the concepts identified RLPO and where they are located in the class image ("zebra") for GoogleNet. As highlighted the "stripes" concept are located near zebra, the "running" concept, showing trees are highlighting in background of the input image, and the "mud" concept highlighting the grass and soil in the input image. The concepts are ordered in their importance (TCAV score) with "stripes" being the highest and "mud" being the lowest for the selected class.

may result in different explainable states, reflecting multiple concepts inherent to the $f(\cdot)$. Some actions might take longer to reach an explainable state. Also, it is possible for different actions to lead to the same explainable state. As the goal is to optimize all states to achieve a common target, DQN progressively improves action selection to expedite reaching these states. Thus, DRL becomes relevant as it optimizes over time to choose the actions that are most likely to reach an explainable state efficiently.

4 Experiments

We performed our concept-based XAI method on GoogleNet [15] trained on ImageNet classes. We conducted experiments on an NVIDIA GeForce RTX 4090 GPU.

4.1 What kind of concepts can RLPO generate?

Novel concepts. As illustrated in observed that the Fig. 3, as hypothesized, generative model can generate new concepts that a human would not typically think of but leads activations of the DNN to trigger. To validate this hypothesis, we conducted a survey to see if humans can think of these concepts as important for the neural network to understand a certain class (Table 1). We presented a random class image followed by two concepts, one generated by our method and another from a previous retrieval based method [4, 3]. We also provide options from the top priority concepts from both method and we discovered that while most participants could recognize retrieval-based concepts, only those with domain-specific knowledge could identify generated concepts. This indicates that most people can only identify retrieval concepts a small subset of what $f(.)$ learns during training.

Table 1: Accuracy and Odds[1] calculated based on the responses for ours and retrieval based method, respectively, from the human survey.

	Laymen (n=26)	Expert (n=24)
Accuracy (Retrieval)	93.46%	89.55%
Accuracy (Ours)	8.46%	34.55%
Odds (Retrieval)	14.29	8.57
Odds (Ours)	0.09	0.53

Multiple concepts. Because the RL algorithm explores various various explainable states, we can obtain multiples concepts with varying TCAV scores. Fig. 3 shows the top three class-level concepts identified by our method for the "zebra" class on GoogleNet. Further, the image on the left highlights where these concepts are located in the class image. This is obtained by using ClipSeg [16], a transformer-based segmentation models which can take image prompts, X_c, and highlights in

[1] **Odds**: Odds describe how many times an event is expected to happen compared to how many times it is not. They are often used in gambling, sports betting, and statistics. The odds of an event with probability p (where p is the probability of the event happening) are calculated as: $\frac{p}{1-p}$.

Table 2: Novel concepts: $TS_{c,m}$ (TCAV score), CS (Cosine similarity), ED (Euclidean distance), RCS, and RED (CS and ED with ResNet50 embedding)

Methods	Concepts	$TS_{c,m}(\uparrow)$	CS ($\downarrow$)	ED ($\uparrow$)	RCS ($\downarrow$)	RED ($\uparrow$)
EAC [5]	C	1.0	0.76 ± 0.03	7.21 ± 0.63	0.67 ± 0.14	6.34 ± 2.16
Lens [4]	C1	1.0	0.77 ± 0.02	7.17 ± 0.34	0.50 ± 0.18	9.70 ± 3.20
	C2	1.0	0.72 ± 0.04	8.02 ± 0.87	0.42 ± 0.10	10.90 ± 2.80
	C3	1.0	0.69 ± 0.05	8.45 ± 0.96	0.45 ± 0.05	11.03 ± 2.17
CRAFT [3]	C1	1.0	0.76 ± 0.04	7.37 ± 0.62	0.57 ± 0.16	8.80 ± 3.20
	C2	1.0	0.72 ± 0.02	8.25 ± 0.39	0.50 ± 1.90	9.90 ± 3.40
	C3	1.0	0.73 ± 0.04	7.98 ± 0.79	0.44 ± 0.07	10.80 ± 1.90
RLPO (Ours)	C1	1.0	0.52 ± 0.04	10.48 ± 0.50	0.04 ± 0.01	16.80 ± 1.40
	C2	1.0	0.49 ± 0.02	10.65 ± 0.20	0.02 ± 0.02	17.20 ± 0.80
	C3	1.0	0.49 ± 0.02	10.74 ± 0.30	0.03 ± 0.01	17.60 ± 4.40

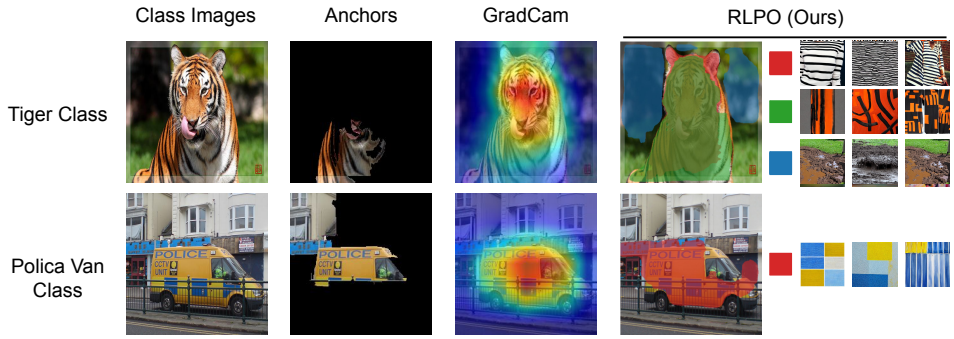

Figure 4: Comparison of concepts identified by different methods and our method.

a test image, $x \in X_m$, which part resembles the input prompt. Fig. 4 shows how our method identifies multiple concepts with importance, when compared to other methods like Anchors [17] and GradCam [18]. Table 2 highlights how concepts generated by our method are different from one extracted from the dataset.

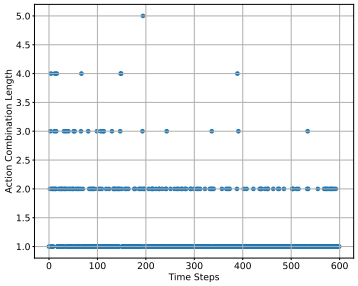

Figure 5: Combined actions (multiple keywords) count over training time.

Abstract concepts. In Fig. 2, we observe the progression of output images generated by the SD when preference optimization is applied for the initial prompt "zoo," which was decided by the DQN, on the tiger class. These abstractions gives us hint about what the model prefers when it is looking for tiger, starting from a four-legged orange furred animal, to black and white stripes with orange furred animal, to black and white stripes with orange furred and whiskers. Though we obtain concepts with various abstractions by changing threshold η, currently it is not possible for our method to decide on a threshold η to get a particular level of abstraction.

Complex concepts. The RL algorithm is capable of integrating multiple concepts to identify complex concept combinations. In this scenario, RL episodes conclude either when a concept, formed through sequential actions, achieves a TCAV score of 1, or when a maximum of four actions have been taken. This upper limit prevents the agent from lingering indefinitely within a single episode, ensuring there is no stagnation in the optimization of the generative model. As shown in Fig. 5, we observe that during training with multiple combination, the RL agent starts taking multiple actions initially, but as the training progresses it limits to taking one or two action combination because it starts getting good scores as over time the less number action generates better results.

4.2 Are the generated concepts reliable?

After finding the relationship between generated concepts and input images, it is important to validate the importance of the identified concepts. To do that, we applied c-deletion to the class images for each identified concept. We tested it with, 1) traditional c-deletion, where we gradually delete concept segments based on decreasing order of TCAV importance and 2) on a modified c-deletion, which stops deleting after removing the concept we test. The results for the c-deletion, original and clipped are shown in the Fig. 6.

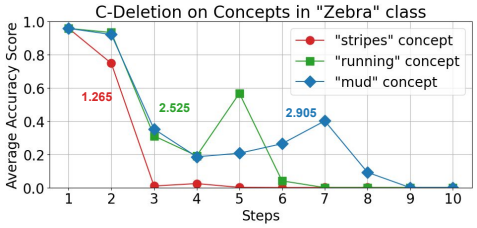

Figure 6: C-deletion. Removing concepts over time to measure the reliability. Colored values indicate the cumulative avg. accuracy over time (the lower the better as it indicates important concepts are removed sooner.)

5 Limitations and Conclusions

The process of navigating an infinitely large concept space and generating explainable concepts from textual inputs presents several challenges, particularly when dealing with complex, high-level concepts. We showed how DRL and preference optimization can be combined to efficiently navigate this space. However, RLPO also suffers from several limitations. First, this analysis cannot be performed real-time since generating images from SD, learning the DQN, and fine-tuning the SD with preferences takes some time. Also, the concepts that our algorithm generates can be diverse as it tries to reveal the concepts inherent to the $f(\cdot)$, making it less domain-specific. Despite the challenges, our results show how to leverage the strengths of visual representations and adaptive learning to provide intuitive and effective solutions for understanding complex, high-level concepts in black-box neural networks.

References

[1] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[2] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in neural information processing systems*, 32, 2019.

[3] Thomas Fel, Agustin Picard, Louis Bethune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2711–2721, 2023.

[4] Thomas Fel, Victor Boutin, Louis Béthune, Rémi Cadène, Mazda Moayeri, Léo Andéol, Mathieu Chalvidal, and Thomas Serre. A holistic approach to unifying automatic concept extraction and concept importance estimation. *Advances in Neural Information Processing Systems*, 36, 2024.

[5] Ao Sun, Pingchuan Ma, Yuanyuan Yuan, and Shuai Wang. Explain any concept: Segment anything meets concept-based explanation. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] An Yan, Yu Wang, Yiwu Zhong, Chengyu Dong, Zexue He, Yujie Lu, William Yang Wang, Jingbo Shang, and Julian McAuley. Learning concise and descriptive attributes for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3090–3100, 2023.

[7] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[8] Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *arXiv preprint arXiv:2408.03539*, 2024.

[9] Ransalu Senanayake. The role of predictive uncertainty and diversity in embodied ai and robot learning. In *arXiv preprint arXiv:2405.03164*, 2024.

[10] Harrison Charpentier, Ransalu Senanayake, Mykel Kochenderfer, and Stephan Günnemann. Disentangling epistemic and aleatoric uncertainty in reinforcement learning. In *ICML 2022 Workshop on Distribution-Free Uncertainty Quantification*, 2022.

[11] Som Sagar, Aditya Taparia, and Ransalu Senanayake. Failures are fated, but can be faded: Characterizing and mitigating unwanted behaviors in large-scale vision and language models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.

[12] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[16] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7086–7096, June 2022.

[17] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[18] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: visual explanations from deep networks via gradient-based localization. *International journal of computer vision*, 128:336–359, 2020.

[19] Lisa Schut, Nenad Tomasev, Tom McGrath, Demis Hassabis, Ulrich Paquet, and Been Kim. Bridging the human-ai knowledge gap: Concept discovery and transfer in alphazero. *arXiv preprint arXiv:2310.16410*, 2023.

Appendix

1 Definitions

Definition 1. *Explainable states: States that have a concept score $TS_{c,m} \geq \eta$ for a user-defined threshold $\eta \in [0, 1]$ for concept c and class m is defined as an explainable state.*

Definition 2. *Proxy states: States that have a concept score $TS_{c,m} < \eta$ for the threshold $\eta \in [0, 1]$ for concept c and class m is defined as a proxy state.*

2 The Rationales Behind Design Choices

Before presenting the algorithm in detail, we provide the rationale for design choices, which are validated through ablation studies and comparisons in the Experiments section.

Rationale 1: Why concept generation is a better idea. Let us denote the set of human-interpretable concepts that the NUT has learned be $\mathcal{C}_N$. If we use concept-based explanation the traditional way [1, 19], then the end users need to manually guess what concepts to test for. Automatically retrieving the concept set by segmenting test images [5] also results in a limited concept set. In contrast, a SOTA generative model can generate high quality images.

Rationale 2: Why a deep RL-controlled VLM fine-tuning for generating concepts is a better idea. *"A picture is worth a thousand words but words flow easier than paint."*

As the saying goes, "a picture is worth a thousand words," it is much easier for people to explain and understand high-level concepts when images are used instead of language. For instance, we need a long textual description such as "The circles are centered around a common point, with alternating red and white colors creating a pattern" to describe a simple image of a dart board (i.e., Target Co. logo). Therefore, we keep our ultimate concept representation as images. However, controlling a generative model from visual inputs is much harder. However, since human language can be used as a directed and easier way to seed our thought process, as the saying goes, "words flow easier than paint," we control the use of text prompts. Since the vastness of the search space cannot be handled by most traditional search strategies, we resort to a DQN for controlling text. Since simple text alone cannot generate complex, high-level visual concepts, in each DQN update step, we use preference optimization to further guide the search process to guide towards more preferred outcome, allowing the DQN to focus on states similar to the target. This approach improves our starting points for each DQN episode, enabling more efficient search and incremental progress towards the desired target.

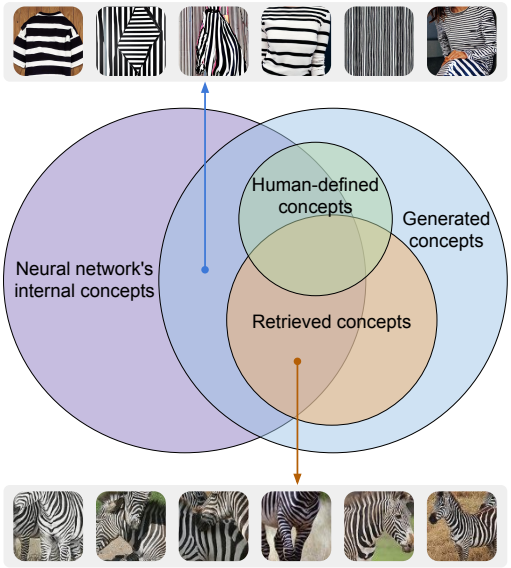

Figure 7: We can generate concepts that encompass both human-defined and retrieved concepts. Note that retrieved concepts are very similar to class images, making them less useful as concepts.

3 The Effects of RLPO-based Concept Space Traversal

As depicted in Fig. 7, we can formalize the notion that a generative model can encompass concepts pre-defined by a human or retrieved through segmentation as follows.

Theorem 1. *Let the set of human-interpretable concepts that the NUT has learned be $\mathcal{C}_N$, and the concept sets human collected, retrieved though segmentation, and generated using a generative model be $\mathcal{C}_H$, $\mathcal{C}_R$, and $\mathcal{C}_G$, respectively. Then, $|\mathcal{C}_G \cap \mathcal{C}_N| \geq |\mathcal{C}_H \cap \mathcal{C}_N| \geq 0$ and $|\mathcal{C}_G \cap \mathcal{C}_N| \geq |\mathcal{C}_R \cap \mathcal{C}_N| \geq 0$.*

Proof sketch. $\mathcal{C}_H \subseteq \mathcal{C}_G$ and $\mathcal{C}_R \subseteq \mathcal{C}_G \implies |\mathcal{C}_H \cap \mathcal{C}_N| \geq 0$ and $|\mathcal{C}_R \cap \mathcal{C}_N| \geq 0$ □

We now formalize what concepts the DQN has learned, with reference to Fig. 1.

Theorem 2. *When traversing in the concept space, with each reinforcement learning step,*

1. *Case 1: Moving from a proxy state towards an explainable state monotonically increases the reward.*
2. *Case 2: Moving from an explainable state towards the target class does not increase the reward.*

Proof sketch. Obtain the rewards before and after η and compute the difference in reward for each segment. □

Property 1. *As ξ escalates, the reward function proportionally amplifies, particularly enhancing the significance of outcomes near t_η.*

Theorem 2 characterizes the how TCAV scores (i.e., proportional to rewards) are increased up to η. As a result, as shown in Theorem 3, if the generator moves close to the image class, then the explainer generates images similar to the class. Therefore, by varying η we can generate concepts with different levels of abstractions.

Theorem 3. *As we go closer to the concept class, $|\mathcal{C}_G \cap \mathcal{C}_N|$ becomes larger for generated concepts $\mathcal{C}_G$ and NUT's internal concepts, $\mathcal{C}_N$.*

Proof sketch. Measure the sensitivity difference between $S_{c_1,m,t}$ and $S_{c_2,m,t}$ as $t \to \infty$. □

4 Additional Experiments

Few more examples on concepts identified by our method on goldfish and zebra class is shown in Fig. 8.

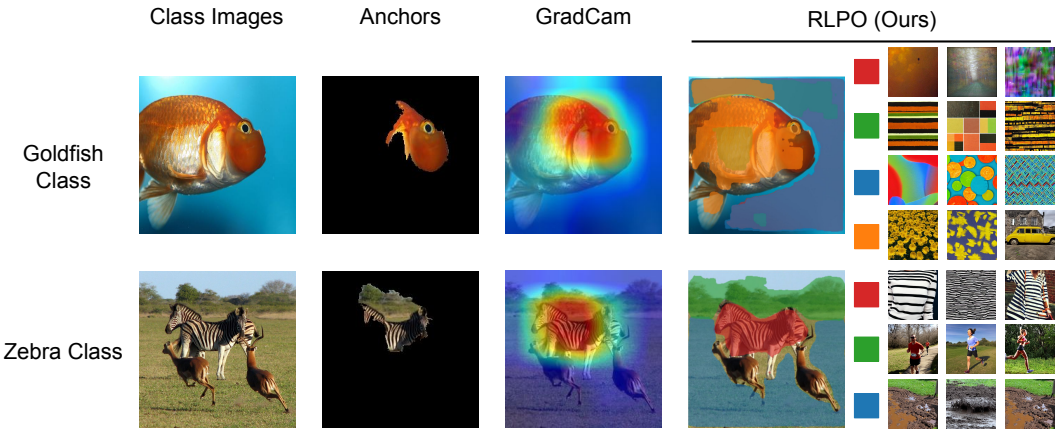

Figure 8: Comparison of concepts identified by different methods and our method.

4.1 Ablation study

An RL algorithm was used because of the large state-action space. For this study, we chose DQN as our RL algorithm because of its ability to effectively traverse through small (20 actions) and discrete action space (unique keywords). We assess the effectiveness of RL by disabling preference optimization step. As shown in Table 3, on GoogleNet, compared to ϵ-greedy methods, our RL setup exhibits a higher entropy, higher average normal count (ANC) and lower coefficient of variance (CV), indicating that it is able to explore the space more broadly by picking diverse keywords.

Table 3: Comparison between search strategies

Method	Entropy (↑)	ANC (↑)	CV (↓)
RL (Ours)	**2.80**	**0.43**	**0.46**
0.25 Greedy	2.40	0.21	0.96
0.5 Greedy	1.95	0.15	1.67
0.75 Greedy	1.85	0.15	1.77

5 Application

We discussed how RLPO can be used as a diagnostic tool for the engineers. As a specific application, we see what concepts are removed and added as well as how the concept importance changes when we fine-tune ResNet50 on ImageNet to improve accuracy (Fig. 9).

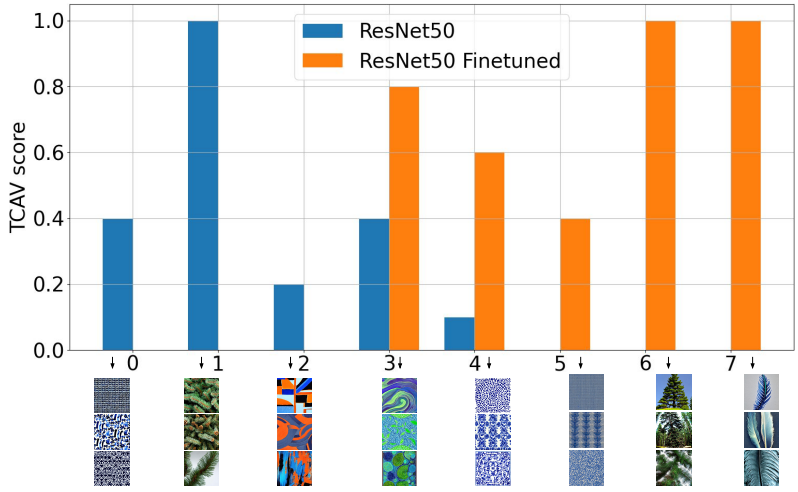

Figure 9: Concept-shift analysis pre-post fine-tuning plot