INSIGHTS ON VISUAL REPRESENTATIONS FOR EMBODIED NAVIGATION TASKS

Erik Wijmans[1*], **Julian Straub**[2], **Dhruv Batra**[1,3], **Judy Hoffman**[3], **Ari Morcos**[3]
[1]Georgia Institute of Technology, [2]Facebook Reality Labs (FRL), [3]Facebook AI Research (FAIR)
[1]{etw}@gatech.edu, [2]{jstraub}@oculus.com, [3]{dbatra, judyhoffman, arimorcos}@fb.com

ABSTRACT

Recent advances in deep reinforcement learning require a large amount of data and result in representations that are often over specialized to the target task. In this work, we study the underlying potential causes for this specialization by measuring the similarity between representations trained on related, but distinct tasks. We use the recently proposed projection weighted Canonical Correlation Analysis (PWCCA) to examine the task dependence of visual representations learned across different embodied navigation tasks. Surprisingly, we find that slight differences in task have no measurable effect on the visual representation. We then empirically demonstrate that visual representations learned on one task can be effectively transferred to a different task. Finally, we show that if the tasks constrain the agent to spatially disjoint parts of the environment, differences in representation emerge, providing insight on how to design tasks that induce general, task-agnostic representations.

1 INTRODUCTION

Recent advancements in Deep Reinforcement Learning (Deep RL) have allowed for the creation of systems that are able to out-perform human experts on various different games such as Chess, Go, Dota2, and Starcraft2. However, these advances have largely relied on using significant amounts of computation to account for lack of sample efficiency. Deep RL has also been shown to be able to greatly over-fit on even complex problems Zhang et al. (2018), giving concern that the representations learned via Deep RL will be specific to their task and won't be reusable for new tasks.

In this paper we seek to answer the following question: Do different embodied navigation tasks induce different visual representations? There have been a number of recent simultaneous works proposing to train robots as Embodied Agents in simulated environments with the ultimate goal of transferring agents learned in simulation to reality. Embodied navigation tasks decouple the task from the environment, in contrast to video game objectives in which the task and environment are fundamentally coupled. Furthermore, the ability to reuse representations for new tasks and in new environments is of particular concern to the goal of transferring embodied agents from simulation to reality. Once in the real world, the agent should be capable of learning new tasks – such as finding new objects or handling new questions – and be able to cope with the non-stationarity of a changing world.

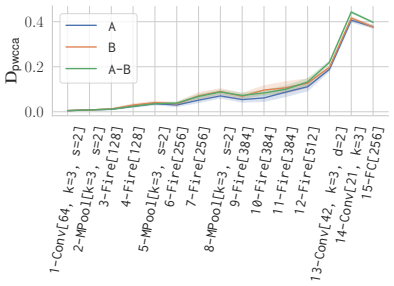
To study our primary question, we first adapt the methodologies proposed in Raghu et al. (2017); Morcos et al. (2018) and find that, surprisingly, differences in task do not lead to differences in visual representation. We leverage this knowledge to show that visual representations trained for embodied tasks are useful for learning new tasks. Finally, we design a special case where the different tasks constrain the agent to spatially disjoint locations in the environment, resulting in different representations and providing insight on how task independent visual representations emerge.

2 HOW DEPENDENT ON THE TASK ARE REPRESENTATIONS?

In this experiment, we analyze visual representations learned for different embodied tasks.

[*]Work done while an intern at FRL and FAIR

(a) Top-down view in the environment. Circles denote the location of all target objects. Coloring denotes which target set objects are in for the multi-target disjoint split: blue for $\mathcal{A}$, red for $\mathcal{B}$, and green is unused.



(b) PWCCA results of comparing networks trained on different embodied tasks. On the x-axis is a description of the layer. All layers are marked with their number of output channels, along with any parameters. `k` is kernel size, `s` is stride, `d` is dilation. `Fire` are SqueezeNet Fire modules.

Figure 1

Core Hypothesis. Training for different embodied tasks induces different visual representations. Due to Deep RL's ability to over-fit on even complicated tasks, it is reasonable to expect that the representations learned will be highly tuned to their specific task.

2.1 Experimental setup

Task. We instantiate the Object Goal Navigation Task (ObjectNav) due to its reliance on both semantic and spatial understanding. In ObjectNav, an agent is given a token describing an object in the environment, such as *fridge*, and then must navigate through the environment until it finds a good view of the fridge and calls the `stop` action. The reward given to the agent when it calls stop is proportional to how much of the target object is in the agent's field of view. At every time-step, a shaped reward of $-\Delta_{\text{geo\_dist}}$ is also provided.

To gain insight into the impact of task differences on visual representations, we must first understand the differences between the tasks themselves. An ideal task set should contain tasks for which the learning and reward dynamics are very similar, but which differ in simple and easily understandable ways. To accomplish this, we randomly divide the set of target objects, $\mathcal{X}$, into two equally sized and disjoint subsets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \cap \mathcal{B} = \varnothing$, $\mathcal{A} \cup \mathcal{B} = \mathcal{X}$, and $|\mathcal{A}| = |\mathcal{B}|$ (assuming $|\mathcal{X}|$ is even). We average our results over five different choices of $\mathcal{A}$ and $\mathcal{B}$.

Environment. We use the state-of-the-art reconstruction method proposed in Whelan et al. (2018) to create an extreme high-fidelity reconstruction. We utilize a perceptual and semantically realistic environment so that our analysis will be more applicable to the ultimate goal of agents operating in reality. See Fig. 1a for a top-down view of the environment.

Agent. The agent has 4 primitive actions, `move_forward`, which moves 0.1 meters forward; `turn_left` and `turn_right` (which turn 9 degrees left and right, respectively), and `stop` which signals that the agent believes it has completed its task. At every time-step, the agent receives an egocentric `RGB` image and the token specifying the target object.

Policy. We parameterize our agent with 3 components. A visual encoder, a target encoder, and a recurrent policy. The visual encoder utilizes SqueezeNet1.2 as the backbone architecture due to its parameter efficiency and representational power. Given an `RGB` image, the visual encoder produces a 256 dimensional embedding. See Fig. 1b for the full architecture of the visual encoder.

Note that the vast majority ($\sim 80\%$) of the learnable parameters are in the visual encoder. This is key to our analysis as we find that a policy with significantly more parameters is able to do very well on the task with a frozen randomly initialized visual encoder.

Training. We use Proximal Policy Optimization (PPO) to train our agent.

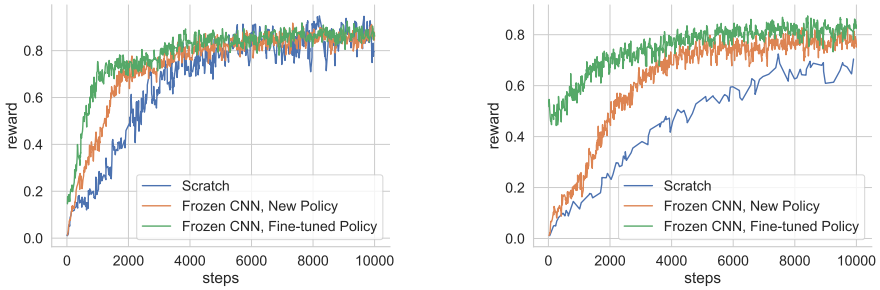


Figure 2: Results of transferring policies learned on one task to the other task. Train reward while learning target set $\mathcal{B}$ (left) and target set $\mathcal{A} \cup \mathcal{B}$ (right) under three different regimes.

2.2 Measuring similarity of representations

In order to compare the representations of two deep neural networks, we follow the approach of Raghu et al. (2017); Morcos et al. (2018).

Given two neural networks, A and B, and a set of N inputs, Raghu et al. (2017); Morcos et al. (2018) compare the representations at layer L of both networks by 1) extracting the neuron activation matrix, X, of both networks – where $X_{i,j}$ is the activation of the i^{th} neuron on the j^{th} input; and 2) compute the distance between the neuron activation matrices using Canonical Correlation Analysis (CCA). CCA finds a basis which maximizes the correlation between two matrices and then computes the correlation in that basis to account for any rotational differences in the activation matrices.

We follow the technique proposed by Morcos et al. (2018) to account for differing numbers of noise dimensions between representations. Given each of the CCA directions h_i and correlation coefficients ρ_i, Morcos et al. (2018) first computes the projection coefficients $\alpha_i = \sum_k |\langle d_i, X_k \rangle|$ and then computes 1 minus the weighted average of the correlation coefficients, $\mathbf{D}_{\text{pwcca}} = 1.0 - \frac{1}{\sum_k \alpha_k} \sum_k \alpha_k \rho_k$, as the distance between representations.

A naive approach for using PWCCA to measuring the effect of different target sets on the representation learned would be to train a policy for $\mathcal{A}$ and a policy for $\mathcal{B}$ and then measure the dissimilarity. This approach 1) doesn't control for the effect of different random initialization, and, more importantly, 2) doesn't ground the values reported by PWCCA (which is a unit-less metric). To control for these issues, we compare the distance between models trained on *different* tasks to the distance between models trained on the *same* task. To compare representations across different tasks, we train N networks for $\mathcal{A}$ and N networks for $\mathcal{B}$, compute the PWCCA distance for each pair of networks, and then average over the N^2 pairwise comparisons to control for the random seed. To compare representations learned for the same task, we take the N networks trained on $\mathcal{A}$ (or $\mathcal{B}$), and compute the PWCCA distance for the $\binom{N}{2}$ network pairs.

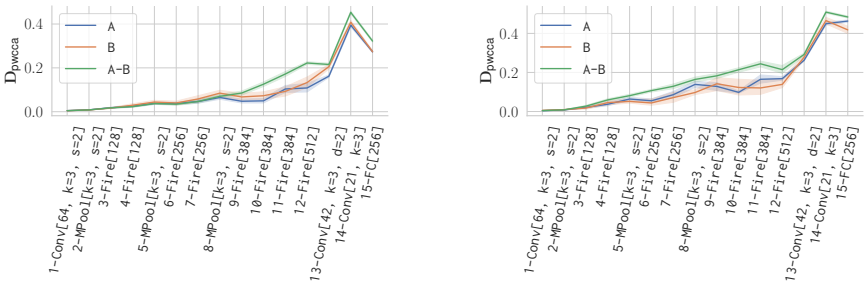
2.3 Results

We use the following notation to denote our comparison: comparisons across networks trained on the same task are denoted without a dash, *e.g.* `A` is the comparison of networks trained on $\mathcal{A}$ among themselves. Comparisons across networks trained on different tasks are denoted with a dash, *e.g.* `A-B` is the comparison *between* networks trained on $\mathcal{A}$ and networks trained on $\mathcal{B}$. These comparisons are repeated over five different choices of $\mathcal{A}$ and $\mathcal{B}$.

If networks trained on different tasks learn different representations, we would expect that the `A-B` distance should be higher than that for `A` or `B` alone. In contrast, we found that distances were similar regardless of task trained, suggesting that networks learn task-agnostic visual representations (Fig. 1b). This result implies that the representation learned for one task should transfer to another.

3 Transferring between $\mathcal{A}$ and $\mathcal{B}$

Setup. We examine two types of transfer experiments: 1) transferring the policy learned on $\mathcal{A}$ to $\mathcal{B}$ (or from $\mathcal{B}$ to $\mathcal{A}$); 2) transferring the policy learned on $\mathcal{A}$ to $\mathcal{A} \cup \mathcal{B}$, the full set of targets. In *all*



(a) PWCCA analysis on the single target disjoint target sets

(b) PWCCA analysis on the multiple target disjoint target sets

Figure 3

transfer experiments, every layer of the visual encoder is frozen. We also compare fine-tuning the policy learned on $\mathcal{A}$ to learning a new policy from scratch.

Results. Consistent with the PWCCA experiments, we found that visual representations learned on $\mathcal{A}$ are effective for learning both $\mathcal{B}$ and $\mathcal{A} \cup \mathcal{B}$ (Fig. 2). We also found fine-tuning to be more effective than learning a new policy from scratch, suggesting that general navigation skills can transfer. The most striking result is how quickly $\mathcal{A} \cup \mathcal{B}$ can be learned using representation learned on $\mathcal{A}$.

4 A SPATIALLY DISJOINT SPLIT

In the previous sections, we demonstrated that the visual representations learned across tasks are highly similar and can be transferred across tasks, but the aspects of these tasks which enable task-agnostic learning remain unclear. One possibility is that both target sets cover the entire visual manifold, leading agents to explore the same portions of the environment across tasks. To test this hypothesis, we created hand-designed target sets which contain little to no spatial overlap.

Single target. For the single target case, we find the two targets which are farthest apart and assign them to the two target sets. This construction results in agents with minimal spatial overlap in their trajectories. Consistent with our hypothesis, we found that the PWCCA distance across tasks is now substantially higher than that within the same tasks (Fig. 3a).

Multiple targets. We also examine spatially disjoint sets with multiple target objects. See Fig. 1a for a visualization of the multi-target disjoint split. Again, we found that spatially disjoint target sets result in greater dissimilarity of the visual representations across tasks (Fig. 3b), consistent with spatial overlap leading to similar visual representations.

5 FUTURE WORK

In the future, we plan to extend this analysis along other axes of variation, including different architectures, different environments, and tasks with different reward structures.

REFERENCES

Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *NeurIPS*, 2018. 1, 3

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *NeurIPS*, 2017. 1, 3

Thomas Whelan, Michael Goesele, Steven J Lovegrove, Julian Straub, Simon Green, Richard Szeliski, Steven Butterfield, Shobhit Verma, and Richard Newcombe. Reconstructing scenes with mirror and glass surfaces. *ACM Transactions on Graphics*, 37(4):102, 2018. 2

Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint*, 2018. 1