Efficient Morphology-Aware Policy Transfer to New Embodiments

Michael Przystupa[1,3,4], Hongyao Tang[3,4], Martin Jagersand[1], Santiago Miret[5], Mariano Phielipp[5], Matthew E. Taylor[1,2], Glen Berseth[3,4]

Keywords: Transfer Learning, Morphology-Aware Learning, Online Learning

Summary

In this work, we investigate means of reducing the computation costs to finetune pre-trained morphology-aware policies to target morphologies with on-policy learning. Morphology-aware learning is a paradigm which attempts to learn several optimal policies across agent embodiments in a *single* neural network. A limitation of prior works have been focusing on end-to-end finetuning to adapt these policies to a target morphology. We address this gap by exploring parameter efficient techniques used successfully in other domains such as computer vision or natural language processing to specialize a policy. Our results suggest that using as few as 1% of total learnable parameters as the pre-trained model, we can achieve statistically significant performance improvements.

Contribution(s)

1. We conduct an extensive series of experiments to compare the effects of parameter-efficient finetuning methods in the morphology-aware policy learning setting.
   Context: Prior works which include transfer learning experiments have generally focused on end-to-end finetuning or else at most consider low-rank adapter layers (LoRA), a form of delta weight learning, as part of their experiments (Octo Model Team, 2024). When LoRA has been used, experiments have only been conducted only in the *behavioral cloning* setting. This is a limitation in the literature because a wide variety of parameter-efficient techniques have been investigated in other fields such as prefix tuning in large language models (Li & Liang, 2021) and direct-finetuning in computer vision (Lee et al., 2023).
2. We are the first work to successfully learn policies using prefix tuning methods in the reinforcement learning settings.
   Context: Prefix tuning has been almost exclusively investigate in supervised learning settings such as natural language processing (Li & Liang, 2021), computer vision (Nie et al., 2023), or continual learning (Wang et al., 2022). The closest related to our work is Liu et al. (2024) who investigate prefix tuning techniques in the imitation learning setting and across *tasks* as opposed to agent morphology.
3. Our experiments reveal a number of trends in the morphology-aware policy setting. Generally we find that both input-adapter and prefix tuning methods converge to behaving similar to tuning the decoder head of the base model. Prefix tuning is particularly sensitive to hyperparameter choices where some configurations notably affect performance at the beginning of training and never recover. Generally, more parameters are always beneficial to improving policy performance in the tasks we considered.
   Context: Other such prescriptive research has been done in computer vision or language when investigating different PEFT techniques. The work of Lester et al. (2021) demonstrated the potential of prefix tuning over a number of factors including prompt initialization and number of prompt tokens. The work of Liu et al. (2022) highlights the benefits of injecting prompts in multiple layers in transformers. The work of Lee et al. (2023) suggests that intelligent layer different types of domain shifts in computer vision.

Efficient Morphology-Aware Policy Transfer to New Embodiments

Michael Przystupa[1,3,4], **Hongyao Tang**[3,4], **Martin Jagersand**[1], **Santiago Miret**[5], **Mariano Phielipp**[5], **Matthew E. Taylor**[1,2], **Glen Berseth**[3,4]
przystup@ualberta.ca

[1] **University of Alberta, Canada**, [2]**Alberta Machine Intelligence Institute (Amii)**, [3]**Université de Montréal**, [4] **Mila Institute**, [5] **Intel Labs**

Abstract

Morphology-aware policy learning is a means of enhancing policy sample efficiency by aggregating data from multiple agents. These types of policies have previously been shown to help generalize over dynamic, kinematic, and limb configuration variations between agent morphologies. Unfortunately, these policies still have sub-optimal zero-shot performance compared to end-to-end finetuning on morphologies at deployment. This limitation has ramifications in practical applications such as robotics because further data collection to perform end-to-end finetuning can be computationally expensive. In this work, we investigate combining morphology-aware pretraining with *parameter efficient finetuning* (PEFT) techniques to help reduce the learnable parameters necessary to specialize a morphology-aware policy to a target embodiment. We compare directly tuning sub-sets of model weights, input learnable adapters, and prefix tuning techniques for online finetuning. Our analysis reveals that PEFT techniques in conjunction with policy pre-training generally help reduce the number of samples to necessary to improve a policy compared to training models end-to-end from scratch. We further find that tuning as few as less than 1% of total parameters will improve policy performance compared the zero-shot performance of the base pretrained a policy.

1 Introduction

Learning agents that can reuse knowledge across tasks demonstrate improved sample efficiency and better learning capabilities (Reed et al., 2022; Driess et al., 2023; Deng et al., 2023). Deep reinforcement learning (RL), despite its potential, faces significant challenges when applied to multiple tasks due to its sensitivity to even minor environmental variations and sample inefficiency (Henderson et al., 2018; Du et al., 2020). Prior research suggests that even subtle dynamic or kinematic differences can notably affect policy performance (Chen et al., 2018; Schaff et al., 2019). This brittleness and inefficiency create substantial barriers when developing versatile agents that can adapt to new scenarios. Morphology-aware learning is one means of enabling knowledge transfer across different physical agent configurations. Morphology adaptation techniques can improve policy robustness and sample efficiency by explicitly accounting for agent embodiments.

Morphology-aware policy learning incorporates agent morphology knowledge by representing embodiments as graphs processed through GNNs (Scarselli et al., 2009) or transformers (Vaswani et al., 2017). Representing agents as graphs is valuable because it enables policies to represent agents with changing limb configurations, and thus varying action spaces (Wang et al., 2018; Huang et al., 2020; Kurin et al., 2021). Research has focused on effective graph structure utilization through adjacency matrices (Hong et al., 2022; Li et al., 2024), feature grouping (Trabucco et al., 2022; Xiong et al., 2023; Sferrazza et al., 2024), and geometric symmetries (Chen et al., 2023). Morphology-aware

learning can improve sample efficiency as supported by theoretical sample bounds in multi-task learning (Brunskill & Li, 2013; Maurer et al., 2016; D'Eramo et al., 2020; Bohlinger et al., 2025), with empirical results suggesting policies optimized over morphology distributions outperform specialized ones (Gupta et al., 2022; Xiong et al., 2023). Applications include autonomous robot design (Pathak et al., 2019; Luck et al., 2020; Yuan et al., 2022) and large-scale control models (Bousmalis et al., 2024; Open X-Embodiment Team, 2024; Octo Model Team, 2024).

Unfortunately, deploying morphology-aware policies on new embodiments continues to be challenging because of the employment of computationally inefficient transfer learning techniques. Prior works suggest that pre-training morphology-aware policies provide better policy initialization when transferring, but additional finetuning is necessary to elicit optimal performance on new morphologies (Gupta et al., 2022; Xiong et al., 2023; Furuta et al., 2023). These works have focused mainly on end-to-end finetuning algorithms, which can be computationally intensive for larger monolithic policies. In resource-constrained settings like robotics (Huai et al., 2019; Neuman et al., 2022), reducing further computation for learning is referable for transferring policies.

In this work, we investigate parameter-efficient finetuning (PEFT) algorithms as a solution to improve policy transfer performance with reduced computational resources. PEFT algorithms use subsets of a model's parameters to finetune a pre-trained neural network or otherwise introduce a small set of new learning parameters that specialize to a target task (Dong et al., 2023; Kirk et al., 2023). The latter approach is more flexible because these new parameters can be introduced in ways that do not directly change the pre-trained model (Tsai et al., 2020). Researchers have shown that PEFT methods work well on large networks in natural language tasks (Li & Liang, 2021) and in computer vision problems (Lee et al., 2023) while reducing additional computation costs to perform gradient updates on a small set of PEFT parameters compared to the entire model. Closely related to our work is the work of Liu et al. (2024), who investigate PEFT methods in continual imitation learning. Our research is different as we deal with *morphology transfer* and evaluate PEFT methods with deep RL, which presents other challenges from supervised learning.

In summary, the primary contribution of our work is the analysis of several PEFT techniques for morphology-aware policy transfer. Our results demonstrate that it is generally achievable to substantially reduce the total parameters used and achieve statistically measurable improvement over zero-shot performance, even with strong initial zero-shot performance. Using even 1% total learnable parameters relative to the base model's total parameter count leads to measurable performance improvement while significantly reducing learning computation costs compare to end-to-end finetuning. As part of our work, we show how input-learnable PEFT algorithms preserve strong zero-shot capabilities as a performance floor and consistently outperform these initial capabilities as training progresses, making them particularly suitable for online reinforcement learning scenarios with limited data collection opportunities. This research has potential in real-world applications like robotic learning. Our results provide practical guidelines for researchers to determine which PEFT techniques best balance sample efficiency, computational requirements, and performance gains for their specific deployment settings.

2 Background

2.1 Contextual Markov Decision Process

Morphology-aware policy learning can be understood as a form of contextual Markov decision process (CMDP) (Hallak et al., 2015). A CMDP is characterized by a distribution $\mathcal{C}$, where for $c \sim p(C)$ we have an induced tuple $M(c) = (\mathcal{S}^c, \mathcal{A}^c, p^c(s'|s,a), r, p^c(s_0))$. For each c, $\mathcal{S}^c$ is a finite set of states, $p^c(s_0)$ represents the initial state distribution, and $\mathcal{A}^c$ is a finite set of actions. The state transition probability function, $p^c(s'|s,a) = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a; c)$, defines the probability of transitioning from state s to state s' when action a occurs. The reward function, $r^c(s,a,s')$, represents the immediate value of transitioning from s to s' due to a. A policy $\pi : \mathcal{S} \times \mathcal{C} \to \mathcal{P}(\mathcal{A})$ is a mapping from states and contexts to a probability distribution over actions, where π samples actions

$a \sim \pi(s, c)$ to transition following $p^c(s'|s, a)$. For a given CMDP, the objective is to maximize the expected sum of rewards over the distribution of contexts,

$$\pi^\star(s, c) = \arg\max_{\pi \in \Pi} \mathbb{E}_{p(c)}[G_c],$$

where $G_c = \mathbb{E}_{p^c(\tau)}[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)]$ is the expected cumulative reward for a given context with discount factor $\gamma \in [0, 1)$. We only consider the finite horizon case where the tasks will terminate after $T \in \mathbb{N}^+$ steps, and $p^c(\tau) = p^c(s_0) \prod_{t=0}^{T} \pi(s_t, c)\mathcal{P}^c(s_{t+1}|s_t, a_t)$ is the distribution over trajectories in the environment.

In our work, the context c is the morphology information, which we represent as a sequence $\mathbf{c} \in \mathbb{R}^{l(c) \times d^c}$ which has $l(c) \in \mathbb{N}^+$ limbs and $d^c \in \mathbb{N}^+$ features. Each token $\mathbf{c}_i$ contains information such as the limb adjacency matrix, link dynamic values (mass, friction, etc.), and link kinematic information (e.g. joint limits and values). We optimize MDPs with continuous action and state spaces, $\mathbf{a} \in \mathbb{R}^{l(c)}$ and $\mathbf{s} \in \mathbb{R}^{l(c) \times d^s}$ with $d^s \in \mathbb{N}^+$ are state features. This differs from typical CMDPs which usually assume a fixed dimensionality of states and actions.

2.2 Transformers

An essential component of the morphology-aware policies in previous works are transformer models (Gupta et al., 2022; Vaswani et al., 2017; Xiong et al., 2023). We treat our data as an observation sequence $\mathbf{o} \in \mathbb{R}^{l(c) \times d}$ with $l(c)$ limb embeddings with $d \in \mathbb{N}^+$ features. Each token $\mathbf{o}_i = [\mathbf{s}_i; \mathbf{c}_i]$ contains limb-level state and context variables of a morphology for $i \in [1, 2, ..., l(c)]$. A morphology-independent linear transformation projects the limb-specific features to a shared embedding space $\bar{\mathbf{o}} = \text{LN}(\mathbf{o}W^{embed} + W^{position}[1 : l])$, where $W^{embed} \in \mathbb{R}^{d \times h}$ is a linear projection operation that transforms the input features to the hidden dimension $h \in \mathbb{N}^+$. $W^{position} \in \mathbb{R}^{L \times h}$ represents the positional embeddings up to some assumed max sequence length $L \in \mathbb{N}^+$, where only the first l columns of $W^{position}$ are used. LN refers to the LayerNorm function (Ba et al., 2016).

The major component of transformers are the *self-attention* mechanism, which generates weighted combinations of the sequence $\bar{\mathbf{o}}$, $f(\bar{\mathbf{o}}) = \text{softmax}(\epsilon Q K^T)V$. We call $Q = \bar{\mathbf{o}}W^Q$, $V = \bar{\mathbf{o}}W^V$, and $K = \bar{\mathbf{o}}W^K$ the query, key, and value, respectively, and $\epsilon = 1/\sqrt{h}$ is a constant chosen to prevent the dot products from causing extremely peaked softmax distributions. The softmax operator, which converts vectors of real numbers to vectors of probabilities, $\text{softmax}(\mathbf{o})_i = \exp(\mathbf{o}_i)/\sum_{j=1}^{l} \exp(o_j)$, defines the weight each vector $\mathbf{o}_i$ contributes. The parameter set $W^{attn} = \{W^Q, W^V, W^K\} \in \{\mathbb{R}^{h \times h}, \mathbb{R}^{h \times h}, \mathbb{R}^{h \times h}\}$ are linear projections. We learn model parameters with gradient descent. Self-attention is followed by a nonlinear transformation function $f(\bar{\mathbf{o}})$ and residual connection to form transformer layer $T_i(\bar{\mathbf{o}}) = W^{out}\sigma(W^{in}(\text{LN}(\bar{\mathbf{o}} + f(\bar{\mathbf{o}}))) + \text{LN}(f(\bar{\mathbf{o}})) + \bar{\mathbf{o}}$, where $W^{out}, W^{in} \in \mathbb{R}^{h \times h}$, and σ are ReLU functions.

3 Efficient Morphology Transfer Learning

This section discusses our work investigating the efficacy of PEFT algorithms for morphology-aware online RL. Control policies can require immense computation and physical resources to learn for real world systems (e.g. robotics). If policies can explicitly account for agent morphology, this can reduce computation costs by aggregating knowledge between morphologies and improve the policy's generalization capabilities to new embodiments.

Unfortunately, a morphology-aware policy may not elicit the optimal performance of a target morphology due to these generalization capabilities. For real-world applications, it is likely necessary that pretrained model components continue to learn to maximize task performance. Reducing the total necessary learnable parameters is thus significant to achieving this result because, at deployment, it may not be feasible to access sufficient computation resources to perform learning updates. These limitations motivate the potential of PEFT solutions, which are applicable in varying resource limitations when deploying these policies.

We formalize the parameter efficient finetuning problem by first assuming access to a trained policy $\pi(\mathbf{s}, \mathbf{c}\,; \theta^\star)$ with optimized parameters $\theta^\star$. For a new morphology $\bar{c} \sim p(C)$, we expect this policy to have some initial performance $\mathbb{E}_{\pi(s,\bar{c};\theta^\star)}[G_{\bar{c}}]$ which we call the *zero-shot* policy performance. The goal of our work is to optimize new parameters ϕ to maximize the cumulative reward objective,

$$\phi^\star = \arg\max_{\phi} \mathbb{E}_{\pi(s,\bar{c}\,;\theta^\star \cup \phi)}[G_{\bar{c}}(s)],$$

where the new parameters are optimized only for the specific morphology $\bar{c}$. We hypothesize that learning a small set ϕ will perform measurably better than the base policy's zero-shot performance, $\mathbb{E}_{\pi(s,\bar{c}\,;\theta^\star \cup \phi^\star)}[G_{\bar{c}}(s)] > \mathbb{E}_{\pi(s,\bar{c}\,;\theta^\star)}[G_{\bar{c}}(s)]$ where $|\phi| \ll |\theta|$. We investigate this problem learning policies using the *Metamorph* framework and then finetuning this pre-trained policy with one of several PEFT techniques, which describe next.

3.1 Metamorph Framework

Metamorph is morphology-aware learning framework that is an instantiation of the CMDP formulation we described in Section 2. In Metamorph, a policy is trained over a set of 100 training morphologies.[1] Each morphology c induces an observation sequence $\mathbf{o} = [\mathbf{o}_1, \mathbf{o}_2, \mathbf{o}_3, ..., \mathbf{o}_{l(c)}]$ *for each* time step. To account for varying $l(c) \in \mathbb{N}^+$ between morphologies the policy is a transformer (Section 2). The transformer encoders hidden representations $\mathbf{h} \in \mathbb{R}^{l(c)\times h}$ with $h \in \mathbb{N}^+$ hidden features per limb. Actions are predicted with a multi-layer perceptron *per limb* as $\mathbf{a}_i = g_\theta(\mathbf{h}_i)$, where $g : \mathcal{H} \to \mathcal{A}$ is a mapping from hidden representations to actions. Here, $\mathbf{a}_i \in \mathbb{R}$ while $\mathbf{h}_i \in \mathbb{R}^h$. Having a token per limb enables a metamorph policy to adapt to varying limb configurations in practice. The policy $\pi_\theta(\mathbf{o})$ is optimized using Proximal Policy Optimization (Schulman et al., 2017).

We chose to use this framework because it uses transformer-based policies as the morphology-aware policy. Several PEFT techniques we consider in this paper are designed specifically for use with transformer models. The framework code is open sourced, making it accessible to researchers to reproduce our results and compare other PEFT techniques in potential future work. Several works have also built off this repository to improve the base-architecture design (Xiong et al., 2023; 2024).

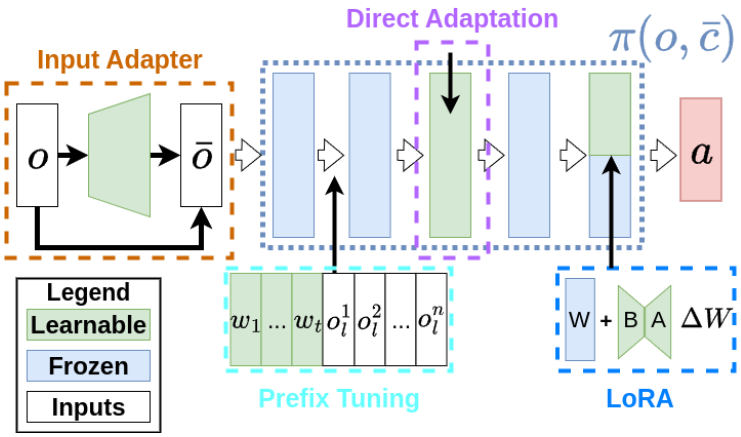


Figure 1: A visualization of the various PEFT techniques considered in this paper. We investigate applying PEFT techniques *independently* from each other.

3.2 Parameter Efficient Finetuning Across Morphologies

We group PEFT approaches as either direct, input, or prefix adaptation techniques. *Direct adaptive* PEFT approaches modify some subset of the weights $\phi \subseteq \theta^\star$ or else add learnable delta weights

[1] We explicitly mention training on 100 morphologies because that is done in the original paper. Any number of training morphologies can be used in practice.

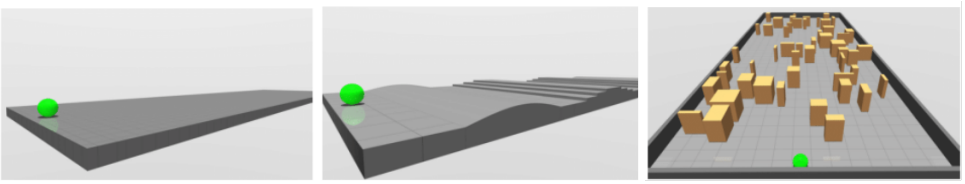
$\hat{W} = W + \Delta W$. *Input-adaptive* PEFT approaches perform some transformation of the inputs to elicit the optimal performance in the model. Prefix tuning prepends a learnable sequenes of tokens to each input sequence. We visualize the various types of PEFT algorithms evaluated in Figure 1.

We consider tuning subsets of $\theta^\star$ for direct adaptive PEFT learning. We list the combinations studied and their experiment result identifiers in Appendix A, for example *Layer 5* represents directly tuning the final transformer layer. For attention and nonlinear transformer layers, we used low-lank adapters (LoRA) (Hu et al., 2022), to learn $\Delta W \in \mathbb{R}^{h^1 \times h^2} = AB$, where $A \in \mathbb{R}^{h^1 \times r}$ and $B \in \mathbb{R}^{r \times h^2}$ are low-rank matrices of rank $r \in \mathbb{N}^+$ to reduce learnable weights for the weight dimensions $h_1 \in \mathbb{N}^+, h_2 \in \mathbb{N}^+$. We describe LoRA initialization details in the Appendix B.

For input-adaptive PEFT approaches, we consider learning an extra input adapter layer. We consider an input adapter layer that modifies the policy observation as $h : \mathbb{R}^{d^c} \to \mathbb{R}^{d^c}$, so that policy uses modified inputs $a \sim \pi_{\theta^\star}(h(\mathbf{o}))$. We consider two variations of the function h where one is a direct nonlinear transform $h(o) = H^{out}\sigma(H^{in}\mathbf{o})$ or else a nonlinear transformation with a residual connection $h(\mathbf{o}) = \mathbf{o} + H^{out}\sigma(H^{in}\mathbf{o})$, with learnable weights $\phi = \{H^{in}, H^{out}\}$. We use a hidden layer size of 256 units. The input adapter transforms observations to elicit better performance from a frozen pre-trained model.

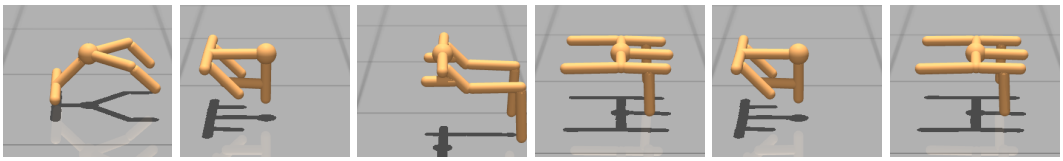
Prefix-tuning is a PEFT approach where a set of learnable tokens are pre-pended to the input sequence to elicit desired outputs from the model (Li & Liang, 2021). These prefixes are a sequence $\phi = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_m]$ of $m \in \mathbb{N}^+$ tokens, where $\mathbf{w}_i \in \mathbb{R}^h$ is a vector. These tokens are then pre-pended to the observations $o^{prefix} = [\phi; \mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_{l(c)}]$ and otherwise processed normally by the transformer layers. Tokens optionally can be pre-pended deeper in the model (e.g., $\mathbf{o}_l^{prefix} = [\phi; T^l(\mathbf{o}^{l-1})]$ for layer $l > 1$) or multiples prefix sets can be used (e.g., $\phi = \{\phi_1, \phi_2, ..., \phi_l\}$ would be learnable prefixes for each layer).

4 Experiments



(a) Flat Terrain (b) Variable Terrain (c) Obstacle Avoidance

Figure 2: Locomotion environments. Diagrams are reproduced from Gupta et al. (2022).



Morphology 1 Morphology 2* Morphology 3 Morphology 4** Morphology 5* Morphology 6**

Figure 3: The six testing morphologies used in our evaluation. Morphology numbers correspond to those shown in relevant results. Morphologies {2, 4}* and {4, 6}** have the same limb configurations but different kinematic and dynamic parameters.

This research aims to evaluate the efficacy of PEFT approaches for online learning on target morphologies. These experiments strive to address the following research questions: *(1) How effectively does each PEFT learning approach compare between each other and end-to-end finetuning?*

(2) What is the relationship between the total number of learnable parameters and the performance when adapting to target morphologies? (3) What are the relevant factors for using prefix tuning and LoRA in online reinforcement learning? Our results contribute to understanding the efficacy of these approaches in online learning, and can help guide future research developing PEFT algorithms for morphology-aware policy transfer. As part of our experiments, we also compare to learning a policy from scratch to determine whether or not if pretraining does help policy transfer.

We report experimental findings on the efficacy of different forms of parameter-efficient finetuning in morphological transfer. We use three locomotion tasks that differ in the terrain types shown in Figure 2; these include a flat surface, randomized variable terrain, and rectangular obstacle avoidance. Each task's reward function emphasizes running as fast as possible to the right. To evaluate the PEFT techniques, we randomly sampled six morphologies from the Metamorph test dataset (Gupta et al., 2022). We visualize the testing morphologies in Figure 3 which include four unique limb configurations and two sets of varying kinematic and dynamic differences. We evaluate PEFT techniques on eighteen environment-morphology combinations.

As mentioned in Section 3, we generate our pre-trained models using the Metamorph framework with default hyperparameters (Gupta et al., 2022). We train five base models using one hundred training morphologies for ten million time steps for each environment. We then apply each PEFT approach with the pre-trained models on the six test morphologies for five million timesteps each. We repeat experiments for five random seeds for every set of PEFT hyperparameters we report. For each seed, we use one of the pre-trained models without replacement. We use the same learning hyperparameters for the pre-training phase, except we *do not use Dropout* in the transformer embedding. Previous research shows that Dropout is helpful for Metamorph pre-training (Xiong et al., 2023), but in preliminary evaluations, we found Dropout was not helpful for finetuning models.

4.1 Best Performances Across Methods

In this section, we report results towards answering our first two research questions on the efficacy of different PEFT techniques. We report results in Figure 4 which shows the performance of different PEFT techniques. We normalize cumulative rewards after performing parameter-efficient finetuning by the pre-trained policy's zero-shot performance averaging these scores across the six testing morphologies. The x-axis shows percentage of learnable parameters to the base-models original parameter counts. We include the original cumulative reward scores by best PEFT hyperparameter configuration in Appendix C.

Our results reveal a number of notable trends across PEFT approaches. An interesting finding suggests that morphology-pretraining utility is dependent on task complexity. On the flat terrain tasks, learning from scratch is comparable to end-to-end finetuning but between variable terrain or obstacle avoidance learning-from-scratch performs substantially worse. Across morphologies, results suggest that the best input-learnable configurations behave similarly to directly tuning the input Embedding and Decoder, suggesting some equivalence between the two approaches for the model sizes used in our experiments. We observed substantial performance improvements tuning just the fifth transformer block, suggesting that if direct model access is possible and a more generous computation budget is available, this layer substantially influences the policy performance. When possible, results suggest more learning parameters are generally favorable given end-to-end finetuning results.

4.2 Ablation of LoRA and Prefix Tuning

In this section, we report results comparing different hyperparameter choices for LoRA and Prefix approaches to address our third research question. We include additional results in Appendix D. The reported results represent the consistent behaviours observed between the evaluations in each environment. Figure 5b shows the results of using LoRA in either the nonlinear transformations (MLP) or attention layer (Attn.) of the fifth transformer layer. The results show that across morphologies for a single layer's full rank matrices are necessary. Applying LoRA to the nonlinear transformation

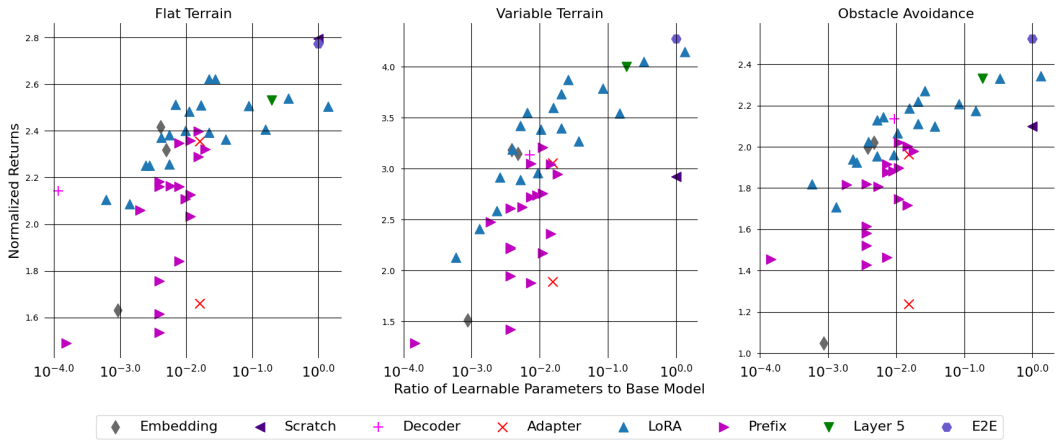


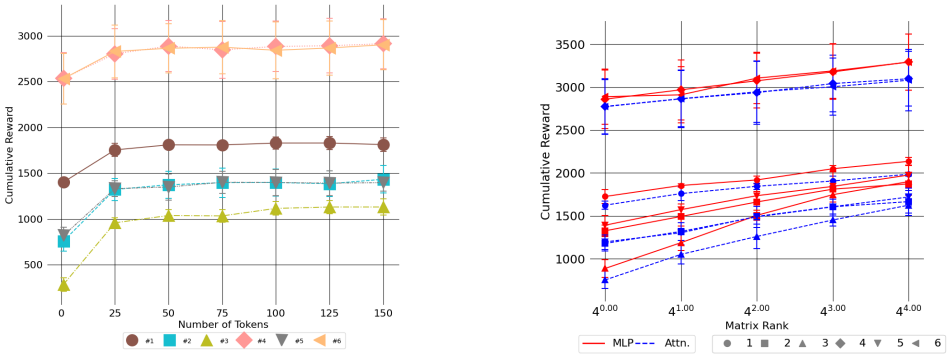
Figure 4: Percentage of trainable ratios to total base model parameters vs achieved normalized results. Results suggest total learnable parameters are a contributing factor in final policy performance.

is preferable for adaption to elicit optimal performance, but results suggest that directly tuning a single layer can be better to avoid introducing more learning parameters.

Prefixing tuning results have more nuanced conclusions. We consider three major factors for effective prefix usage: (1) the number of tokens, (2) the injection layer, and (3) comparing token initialization approaches. Each factor has been shown to substantially impact performance (Ding et al., 2023; Li & Liang, 2021). For (3), we propose a second pretraining stage to learn morphology-aware tokens. This second stage repeats the Metamorph training but keeps the base model frozen while learning the tokens.

We generally observe that more learnable parameters are beneficial, such as by increasing the number of tokens used (see Figure 5a), which agrees with our other findings previously discussed. In our experiments, a complication with prefix tuning is that introducing un-trained tokens can negatively impact policy zero-shot performance. This performance drop can occur because the base model is not trained jointly with the prefix which initially are noisy observations. This problem is largely missed in supervised learning applications because performance is evaluated *after training*. In contrast, we care for performance *during training* especially because it's preferable policies have strong initial performance for real-world systems to avoid consequences of poor-performing policies (e.g., damage to the hardware). We conducted experiments adding 50 prefix tokens as input before different transformer blocks to investigate their impact on learning performance. We compared different token initializations, including zero vectors, small Gaussian noise ($N(0; 10^{-4})$), or the pretrained tokens as previously described in this section. We include results when learning from scratch to understand the value of pretraining for sample efficiency. Figure 6 show's learning curves.

Generally, we observed that the initial zero-shot performance is often negatively affected by zero or random initialization approaches, especially when introducing prefix tokens to the earlier transformer layers. This result suggests that deep layers are less sensitive to the base models' perturbations and better steer feature representations for target morphologies. Interestingly, pre-trained prompting embeddings significantly improved policy performance during learning compared to other initialization approaches, especially on Morphology #3, which we found most PEFT approaches struggled to learn. This demonstrates that prefix initialization can mitigate loss in zero-shot performance during finetuning in online learning.



(a) Number of randomly initialized prefix tokens

(b) Lora in different layers of fifth transformer block.

Figure 5: Ablation studies on prefix tokens and LoRA in variable terrain.

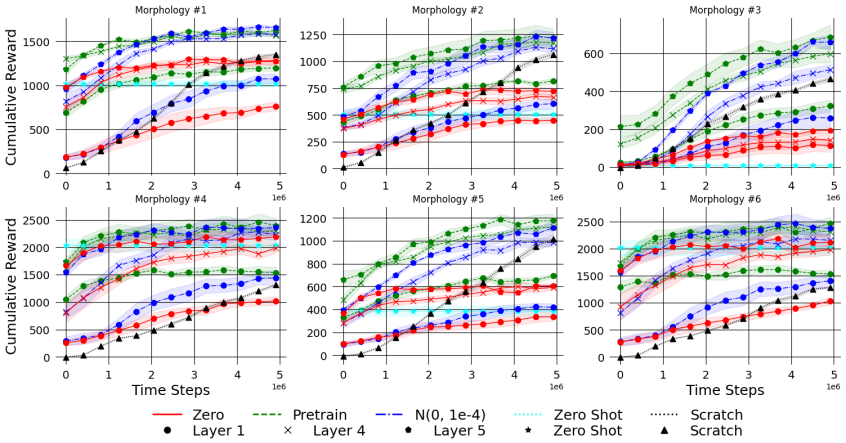


Figure 6: Choice of initialization and injection layers of prefix tuning in variable terrain. Initial zero-shot results of E2E learning are plotted to compare affect of prefixes.

5 Conclusions and Future Work

In this paper, we have investigated the impact of PEFT approaches for finetuning morphology-aware policies. We demonstrate that in most cases, one should train as many parameters online as possible to elicit the best performances of a pre-trained policy. Our analysis reveals that many PEFT approaches provide substantial benefits in deeper layers, so tuning the final transformer block is likely effective for policy finetuning. In scenarios where directly finetuning the base model is difficult, learnable inputs perform similarly to tuning either the input embeddings or decoder layers of the transformer-based policy.

There are several promising future research directions to extend our findings. One crucial factor, particularly for prefix tuning, is the scale of the model. Many reported successes of PEFT approaches are on models with tens of millions to billions of parameters (Li & Liang, 2021). In this work, we used relatively small models ($\sim$3.5 million parameters at most between policy and value function in PPO). We also focused on vanilla transformer architectures used in Metamorph, but researchers have proposed variations for morphology-aware policies (Trabucco et al., 2022; Xiong et al., 2023). Given the promise of PEFT techniques in RL, we see much potential for future development in PEFT development for online learning.

Broader Impact Statement

Although our work focuses on the positives of input adapter and prefix tuning techniques, an undesirable consequences of this work is revealing a means of adversarially attacking deep learning-based control policies. In AI security research this is called *adversarial reprogramming* in which models are repurposed for nefarious uses (Elsayed et al., 2019; Zheng et al., 2023; Englert & Lazic, 2022). Our results show that input adapter finetuning approaches can measurably affect policy performance without changing the base policy weights. We hypothesize that these techniques could be used to re-purpose pre-trained policies for other tasks, potentially in adversarial ways. Some scenarios we imagine this could arise is through benign adversarial action decisions by the pre-trained policy (delaying purchase in investment agent systems, adding extra torque during control, etc.). Given these implications, we caution that research in PEFT techniques should also consider the negative consequence of using input adapter or prefix tuning approaches for control tasks.

Acknowledgments

This work has taken place in part in the Intelligent Robot Learning (IRL) Lab and Vision & Robotics Lab at the University of Alberta, which are supported in part by research grants from Alberta Innovates; Alberta Machine Intelligence Institute (Amii); a Canada CIFAR AI Chair, Amii; Digital Research Alliance of Canada; Mitacs; and the National Science and Engineering Research Council (NSERC). We further acknowledge working with members of the Robotics and Embodied AI Lab (REAL) at the Université de Montréal and Mila who received funding support from the Natural Sciences and Engineering Research Council (NSERC) of Canada, Samsung AI Lab, Google Research, the Canadian Institute for Advanced Research (CIFAR) and compute support from Digital Research Alliance of Canada, Mila IDT, and NVidia. We thank Intel Labs for funding this research.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

Nico Bohlinger, Grzegorz Czechmanowski, Maciej Piotr Krupka, Piotr Kicki, Krzysztof Walas, Jan Peters, and Davide Tateo. One policy to run them all: an end-to-end learning approach to multi-embodiment locomotion. In *Conference on Robot Learning*, 2025.

Konstantinos Bousmalis et al. RoboCat: A self-improving generalist agent for robotic manipulation. *Transactions on Machine Learning Research*, 2024.

Emma Brunskill and Lihong Li. Sample complexity of multi-task reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, 2013.

Runfa Chen, Jiaqi Han, Fuchun Sun, and Wenbing Huang. Subequivariant graph reinforcement learning in 3D environments. In *International Conference on Machine Learning*, 2023.

Tao Chen, Adithyavairavan Murali, and Abhinav Gupta. Hardware conditioned policies for multi-robot transfer learning. In *Advances in Neural Information Processing Systems*, 2018.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. MIND2WEB: towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, 2023.

C D'Eramo, D Tateo, A Bonarini, M Restelli, and J Peters. Sharing knowledge in multi-task deep reinforcement learning. In *International Conference on Learning Representations*, 2020.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao

Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey for in-context learning. *CoRR*, abs/2301.00234, 2023.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. In *International Conference on Machine Learning*, 2023.

Simon S. Du, Sham M. Kakade, Ruosong Wang, and Lin F. Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2020.

Gamaleldin F. Elsayed, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial reprogramming of neural networks. In *International Conference on Learning Representations*, 2019.

Matthias Englert and Ranko Lazic. Adversarial reprogramming revisited. In *Advances in Neural Information Processing Systems*, 2022.

Hiroki Furuta, Yusuke Iwasawa, Yutaka Matsuo, and Shixiang Shane Gu. A system for morphology-task generalization via unified representation and behavior distillation. In *International Conference on Learning Representations*, 2023.

Agrim Gupta, Linxi Fan, Surya Ganguli, and Li Fei-Fei. MetaMorph: Learning universal controllers with transformers. In *International Conference on Learning Representations*, 2022.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual Markov decision processes. *CoRR*, abs/1502.02259, 2015.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Sunghoon Hong, Deunsol Yoon, and Kee-Eung Kim. Structure-aware transformer policy for inhomogeneous multi-task reinforcement learning. In *International Conference on Learning Representations*, 2022.

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. In *International Conference on Learning Representations*, 2022.

Zhibo Huai, Bo Ding, Huaimin Wang, Mingyang Geng, and Lei Zhang. Towards deep learning on resource-constrained robots: A crowdsourcing approach with model partition. In *IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation*, 2019.

Wenlong Huang, Igor Mordatch, and Deepak Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*, 2020.

Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of zero-shot generalisation in deep reinforcement learning. *Journal of Artificial Intelligence Research*, 76, 2023.

Vitaly Kurin, Maximilian Igl, Tim Rocktäschel, Wendelin Boehmer, and Shimon Whiteson. My body is a cage: the role of morphology in graph-based incompatible control. In *International Conference on Learning Representations*, 2021.

Yoonho Lee, Annie S Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *International Conference on Learning Representations*, 2023.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Conference on Empirical Methods in Natural Language Processing*, 2021.

Boyu Li, Haoran Li, Yuanheng Zhu, and Dongbin Zhao. MAT: Morphological adaptive transformer for universal morphology policy learning. *IEEE Transactions on Cognitive and Developmental Systems*, 16(4):1611–1621, 2024.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics*, 2021.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. TAIL: Task-specific adapters for imitation learning with large pretrained models. In *International Conference on Learning Representations*, 2024.

Kevin Sebastian Luck, Heni Ben Amor, and Roberto Calandra. Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. In *Conference on Robot Learning*, 2020.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Sabrina M. Neuman, Brian Plancher, Bardienus P. Duisterhof, Srivatsan Krishnan, Colby Banbury, Mark Mazumder, Shvetank Prakash, Jason Jabbour, Aleksandra Faust, Guido C.H.E. de Croon, and Vijay Janapa Reddi. Tiny robot learning: Challenges and directions for machine learning in resource-constrained robots. In *IEEE International Conference on Artificial Intelligence Circuits and Systems*, 2022.

Xing Nie, Bolin Ni, Jianlong Chang, Gaofeng Meng, Chunlei Huo, Shiming Xiang, and Qi Tian. Pro-tuning: Unified prompt tuning for vision tasks. *IEEE Transactions on Circuits and Systems for Video Technology*, 34(6):4653–4667, 2023.

Octo Model Team. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems*, 2024.

Open X-Embodiment Team. Open X-embodiment: Robotic learning datasets and RT-X models : Open X-embodiment collaboration. In *IEEE International Conference on Robotics and Automation*, 2024.

Deepak Pathak, Christopher Lu, Trevor Darrell, Phillip Isola, and Alexei A. Efros. Learning to control self-assembling morphologies: A study of generalization via modularity. In *Advances in Neural Information Processing Systems*, 2019.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *CoRR*, abs/2205.06175, 2022.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Charles B. Schaff, David Yunis, Ayan Chakrabarti, and Matthew R. Walter. Jointly learning to construct and control agents using deep reinforcement learning. In *International Conference on Robotics and Automation*, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. Body transformer: Leveraging robot embodiment for policy learning. In *Workshop on Embodiment-Aware Robot Learning*, 2024.

Brandon Trabucco, Mariano Phielipp, and Glen Berseth. AnyMorph: Learning transferable polices by inferring agent morphology. In *International Conference on Machine Learning*, 2022.

Yun-Yun Tsai, Pin-Yu Chen, and Tsung-Yi Ho. Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources. In *International Conference on Machine Learning*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. NerveNet: Learning structured policy with graph neural networks. In *International Conference on Learning Representations*, 2018.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Zheng Xiong, Jacob Beck, and Shimon Whiteson. Universal morphology control via contextual modulation. In *International Conference on Machine Learning*, 2023.

Zheng Xiong, Risto Vuorio, Jacob Beck, Matthieu Zimmer, Kun Shao, and Shimon Whiteson. Distilling morphology-conditioned hypernetworks for efficient universal morphology control. *CoRR*, abs/2402.06570, 2024.

Ye Yuan, Yuda Song, Zhengyi Luo, Wen Sun, and Kris M. Kitani. Transform2act: Learning a transform-and-control policy for efficient agent design. In *International Conference on Learning Representations*, 2022.

Yang Zheng, Xiaoyi Feng, Zhaoqiang Xia, Xiaoyue Jiang, Ambra Demontis, Maura Pintor, Battista Biggio, and Fabio Roli. Why adversarial reprogramming works, when it fails, and how to tell the difference. *Information Sciences*, 632:130–143, 2023.

Table 1: Layer tuning parameters and experiment identifiers

Layer Tuned	**Parameters** ϕ	**Exp. Identifier**
End-to-end	θ^*	E2E
Transformer layers	$\{T_i; i \in [1, L]\}$	Layer 5
Attention layers	$\{W_i^{\text{attn}}; i \in [1, L]\}$	Lora
Nonlinear transformation	$\{W_i^{\text{in}}, W_i^{\text{out}}; i \in [1, L]\}$	Lora
Input Embedding	$\{W^{\text{embed}}, W^{\text{position}}\}$	Embedding
Decoder	$\{W_i^{\text{decoder}}; i \in [1, L^{\text{dec}}]\}$	Decoder

Supplementary Materials

The following content was not necessarily subject to peer review.

A Direct Finetuning Configurations

In our experiments, we consider various finetuning scenarios in our evaluations. For direct finetuning methods, we include combinations of subsets we finetune online in Table 1. Our evaluations included subsets of the direct tuning configurations of weight combinations. For example, *Input Embedding* includes combinations in which just W^{embed}, W^{position} and both $\{W^{\text{emebed}}, W^{\text{position}}\}$ are tuned online during training.

B LoRA Initialization Details

When using LoRA in our experiments, we initialize B to small Gaussian noise $b_{ij} \sim N(0; 10^{-4})$ and A to a zero matrix which eliminates LoRA adapters affect on the zeros-hot performance at the beginning of training. LoRA was included as a finetuning method because we want to reduce the total number of parameters used which LoRA can explicitly do via the rank.

C Morphology-Aware Policy Performance

This section reports results for the best-performing PEFT algorithms for each significant grouping of methods we consider. Table 2 show flat terrain results, Table 3 shows variable terrain results, and Table 4 shows results for obstacle avoidance. These results report statistical significance when comparing results to zero-shot pretraining performance and training policies from scratch. Surprisingly, training from scratch worked surprisingly well in flat terrain. Still, most PEFT techniques perform better after five million samples than training from scratch on more complex tasks.

D Prefix Tuning Additional Results

In this section, we include plots similar to those in the main paper for our prefix-tuning ablation experiments. Flat terrain results are shown in Figure 7 and obstacle avoidance in Figure 8. We also show similar ablation results for LoRA and prefix tuning for flat terrain in Figure 9 and obstacle avoidance in Figure 10.

Table 2: Flat Terrain Cumulative Rewards for each testing morphology. Values show mean (top) and standard deviation (bottom). † statistical significance compared to Zero Shot and ‡ statistical significance to Scratch (p < 0.01). P-values and the hypothesis test run (T: t-test, M: Mann-Whitney) comparing against Zero shot and Scratch results.

Morphology	1	2	3	4	5	6
Full Model	$4281.46^{\dagger\ddagger}$	$4552.77^{\dagger\ddagger}$	$1635.82^{\dagger}$	$5545.44^{\dagger\ddagger}$	$5019.71^{\dagger\ddagger}$	$5558.61^{\dagger\ddagger}$
	± 181.77	± 239.11	± 405.64	± 280.16	± 143.13	± 286.80
Layer 4	$3761.11^{\dagger}$	$4121.84^{\dagger}$	$1491.06^{\dagger}$	$5183.88^{\dagger\ddagger}$	$4666.95^{\dagger\ddagger}$	$5192.58^{\dagger\ddagger}$
	± 102.11	± 120.48	± 230.10	± 167.91	± 255.22	± 152.33
Lora	$3798.90^{\dagger}$	$4208.41^{\dagger\ddagger}$	$1639.69^{\dagger}$	$5223.47^{\dagger\ddagger}$	$4761.58^{\dagger\ddagger}$	$5223.47^{\dagger\ddagger}$
	± 138.55	± 77.12	± 41.31	± 174.25	± 210.57	± 174.25
Decoder Only	$2732.26^{\dagger\ddagger}$	$3112.46^{\dagger\ddagger}$	$1398.42^{\dagger}$	$4868.54^{\ddagger}$	$3404.67^{\dagger\ddagger}$	$4858.76^{\ddagger}$
	± 71.35	± 221.65	± 210.79	± 263.81	± 92.07	± 248.01
Embeding	$3308.43^{\dagger\ddagger}$	$3684.66^{\dagger}$	$1554.28^{\dagger}$	$4986.16^{\ddagger}$	$4062.05^{\dagger}$	$4997.82^{\ddagger}$
	± 115.83	± 104.99	± 190.82	± 183.78	± 248.20	± 191.07
Input Adapt	$3231.84^{\dagger\ddagger}$	$3529.41^{\dagger}$	$1510.46^{\dagger}$	$4927.72^{\ddagger}$	$3946.59^{\dagger}$	$4963.53^{\ddagger}$
	± 100.03	± 104.58	± 242.36	± 225.75	± 220.78	± 222.35
Prefix	$3332.33^{\dagger\ddagger}$	$3750.54^{\dagger}$	$1604.92^{\dagger}$	$5064.15^{\ddagger}$	$4199.89^{\dagger}$	$5066.47^{\ddagger}$
	± 126.28	± 201.62	± 336.88	± 133.91	± 276.27	± 137.63
Scratch	$3754.15^{\dagger}$	$3840.33^{\dagger}$	$2191.50^{\dagger}$	3727.29	$4085.82^{\dagger}$	3608.55
	± 210.65	± 211.59	± 624.72	± 733.60	± 217.00	± 777.33
Zero Shot	1867.58	1703.19	253.70	4392.08	1849.41	4431.93
	± 82.55	± 447.69	± 188.25	± 434.01	± 338.10	± 405.78
P-Values comparing against Zeroshot Performance†						
Full Model	$9.1 \times 10^{-9}(T)$	$3.6 \times 10^{-6}(T)$	$2.6 \times 10^{-4}(T)$	$2.1 \times 10^{-3}(T)$	$1.3 \times 10^{-7}(T)$	$1.9 \times 10^{-3}(T)$
Layer 4	$2.3 \times 10^{-9}(T)$	$6.2 \times 10^{-6}(T)$	$3.3 \times 10^{-5}(T)$	$9.3 \times 10^{-3}(T)$	$9.7 \times 10^{-7}(T)$	$8.0 \times 10^{-3}(T)$
Lora	$9.8 \times 10^{-9}(T)$	$4.1 \times 10^{-6}(T)$	$5.3 \times 10^{-7}(T)$	$7.5 \times 10^{-3}(T)$	$4.7 \times 10^{-7}(T)$	$7.1 \times 10^{-3}(T)$
Decoder Only	$2.5 \times 10^{-7}(T)$	$4.9 \times 10^{-4}(T)$	$4.0 \times 10^{-5}(T)$	$9.7 \times 10^{-2}(T)$	$2.1 \times 10^{-5}(T)$	$1.1 \times 10^{-1}(T)$
Embeding	$3.7 \times 10^{-8}(T)$	$2.5 \times 10^{-5}(T)$	$7.9 \times 10^{-3}(M)$	$3.6 \times 10^{-2}(T)$	$5.7 \times 10^{-6}(T)$	$3.6 \times 10^{-2}(T)$
Input Adapt	$2.7 \times 10^{-8}(T)$	$4.6 \times 10^{-5}(T)$	$7.9 \times 10^{-3}(M)$	$6.0 \times 10^{-2}(T)$	$6.4 \times 10^{-6}(T)$	$5.1 \times 10^{-2}(T)$
Prefix	$5.1 \times 10^{-8}(T)$	$3.2 \times 10^{-5}(T)$	$1.1 \times 10^{-4}(T)$	$1.8 \times 10^{-2}(T)$	$4.9 \times 10^{-6}(T)$	$1.8 \times 10^{-2}(T)$
Scratch	$0.0(T)$	$2.9 \times 10^{-8}(T)$	$2.7 \times 10^{-5}(T)$	$1.1 \times 10^{-1}(T)$	$2.2 \times 10^{-9}(T)$	$5.9 \times 10^{-2}(T)$
Zero Shot	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
P-Values comparing against Scratch Performance‡						
Full Model	$6.6 \times 10^{-4}(T)$	$1.1 \times 10^{-4}(T)$	$1.2 \times 10^{-1}(T)$	$2.5 \times 10^{-4}(T)$	$1.9 \times 10^{-6}(T)$	$2.2 \times 10^{-4}(T)$
Layer 4	$9.5 \times 10^{-1}(T)$	$2.3 \times 10^{-2}(T)$	$4.2 \times 10^{-2}(T)$	$1.3 \times 10^{-3}(T)$	$8.9 \times 10^{-4}(T)$	$1.0 \times 10^{-3}(T)$
Lora	$6.9 \times 10^{-1}(T)$	$3.9 \times 10^{-3}(T)$	$8.9 \times 10^{-2}(T)$	$1.1 \times 10^{-3}(T)$	$1.3 \times 10^{-4}(T)$	$9.1 \times 10^{-4}(T)$
Decoder Only	$2.2 \times 10^{-7}(T)$	$6.7 \times 10^{-5}(T)$	$2.3 \times 10^{-2}(T)$	$7.8 \times 10^{-3}(T)$	$2.9 \times 10^{-5}(T)$	$6.1 \times 10^{-3}(T)$
Embeding	$1.2 \times 10^{-3}(T)$	$1.7 \times 10^{-1}(T)$	$1.6 \times 10^{-1}(M)$	$3.8 \times 10^{-3}(T)$	$8.6 \times 10^{-1}(T)$	$2.9 \times 10^{-3}(T)$
Input Adapt	$2.9 \times 10^{-4}(T)$	$1.3 \times 10^{-2}(T)$	$9.9 \times 10^{-2}(M)$	$5.4 \times 10^{-3}(T)$	$3.0 \times 10^{-1}(T)$	$3.5 \times 10^{-3}(T)$
Prefix	$2.1 \times 10^{-3}(T)$	$4.8 \times 10^{-1}(T)$	$9.1 \times 10^{-2}(T)$	$2.4 \times 10^{-3}(T)$	$4.3 \times 10^{-1}(T)$	$1.9 \times 10^{-3}(T)$
Scratch	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
Zero Shot	$0.0(T)$	$2.9 \times 10^{-8}(T)$	$2.7 \times 10^{-5}(T)$	$1.1 \times 10^{-1}(T)$	$2.2 \times 10^{-9}(T)$	$5.9 \times 10^{-2}(T)$

Table 3: Variable Terrain Cumulative Rewards for each testing morphology. Values show mean (top) and standard deviation (bottom). † statistical significance compared to Zero Shot and ‡ statistical significance to Scratch (p < 0.01). P-values and the hypothesis test run (T: t-test, M: Mann-Whitney) comparing against Zero shot and Scratch results.

Morphology	1	2	3	4	5	6
Full Model	$2253.96^{\dagger\ddagger}$	$1983.81^{\dagger\ddagger}$	$2001.18^{\dagger\ddagger}$	$3560.43^{\dagger\ddagger}$	$2047.49^{\dagger\ddagger}$	$3595.38^{\dagger\ddagger}$
	± 41.47	± 154.82	± 42.14	± 317.89	± 117.06	± 368.99
Layer 4	$2093.75^{\dagger\ddagger}$	$1871.09^{\dagger\ddagger}$	$1879.22^{\dagger\ddagger}$	$3254.06^{\dagger\ddagger}$	$1912.17^{\dagger\ddagger}$	$3279.03^{\ddagger}$
	± 34.23	± 79.86	± 33.91	± 353.70	± 135.29	± 379.46
Lora	$2141.39^{\dagger\ddagger}$	$1848.53^{\dagger\ddagger}$	$1786.88^{\dagger\ddagger}$	$3230.13^{\dagger\ddagger}$	$1878.93^{\dagger\ddagger}$	$3234.25^{\ddagger}$
	± 53.29	± 113.44	± 72.97	± 327.39	± 107.89	± 329.42
Decoder Only	$1969.63^{\dagger\ddagger}$	$1623.70^{\dagger\ddagger}$	$1299.89^{\dagger}$	$3164.72^{\dagger\ddagger}$	$1672.47^{\dagger\ddagger}$	$3180.90^{\ddagger}$
	± 28.01	± 126.14	± 70.71	± 307.28	± 112.14	± 316.43
Embeding	$1836.54^{\dagger\ddagger}$	$1529.38^{\dagger}$	$1441.65^{\dagger}$	$2872.51^{\ddagger}$	$1549.29^{\dagger}$	$2887.67^{\ddagger}$
	± 25.22	± 84.38	± 41.51	± 307.30	± 106.71	± 311.40
Input Adapt	$1820.01^{\dagger\ddagger}$	$1521.18^{\dagger}$	$1338.57^{\dagger}$	$2869.53^{\ddagger}$	$1512.25^{\dagger}$	$2895.01^{\ddagger}$
	± 48.63	± 106.76	± 61.81	± 293.57	± 109.46	± 299.56
Prefix	$1902.95^{\dagger\ddagger}$	$1643.33^{\dagger\ddagger}$	$1406.55^{\dagger}$	$2930.13^{\ddagger}$	$1601.95^{\dagger}$	$2918.47^{\ddagger}$
	± 43.36	± 165.26	± 83.55	± 261.58	± 134.90	± 300.10
Scratch	$1679.33^{\dagger}$	$1406.59^{\dagger}$	$1406.58^{\dagger}$	$1735.22^{\dagger}$	$1449.59^{\dagger}$	$1758.99^{\dagger}$
	± 82.91	± 101.71	± 164.55	± 166.72	± 69.66	± 168.21
Zero Shot	1259.92	591.83	136.82	2452.59	685.54	2476.77
	± 61.93	± 67.70	± 103.66	± 291.96	± 71.67	± 349.00
P-Values comparing against Zeroshot Performance†						
Full Model	$4.6 \times 10^{-9}(T)$	$2.1 \times 10^{-6}(T)$	$5.2 \times 10^{-6}(T)$	$4.7 \times 10^{-9}(T)$	$3.5 \times 10^{-8}(T)$	$1.5 \times 10^{-8}(T)$
Layer 4	$1.8 \times 10^{-7}(T)$	$1.5 \times 10^{-6}(T)$	$5.1 \times 10^{-5}(T)$	$6.7 \times 10^{-4}(M)$	$1.9 \times 10^{-6}(T)$	$6.7 \times 10^{-4}(M)$
Lora	$9.5 \times 10^{-8}(T)$	$8.0 \times 10^{-6}(T)$	$5.1 \times 10^{-4}(T)$	$6.7 \times 10^{-4}(M)$	$9.4 \times 10^{-7}(T)$	$6.7 \times 10^{-4}(M)$
Decoder Only	$8.2 \times 10^{-6}(T)$	$5.3 \times 10^{-3}(T)$	$2.2 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$	$7.2 \times 10^{-4}(T)$	$6.7 \times 10^{-4}(M)$
Embeding	$2.0 \times 10^{-3}(T)$	$4.9 \times 10^{-2}(T)$	$6.7 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$	$6.4 \times 10^{-2}(T)$	$6.7 \times 10^{-4}(M)$
Input Adapt	$6.2 \times 10^{-3}(T)$	$8.2 \times 10^{-2}(T)$	$4.2 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$	$2.3 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$
Prefix	$4.8 \times 10^{-4}(T)$	$7.2 \times 10^{-3}(T)$	$1.0 \times 10^{0}(T)$	$6.7 \times 10^{-4}(M)$	$1.9 \times 10^{-2}(T)$	$6.7 \times 10^{-4}(M)$
Scratch	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
Zero Shot	$4.1 \times 10^{-7}(T)$	$1.3 \times 10^{-9}(T)$	$1.8 \times 10^{-9}(T)$	$2.7 \times 10^{-3}(M)$	$0.0(T)$	$2.4 \times 10^{-4}(T)$
P-Values comparing against Scratch Performance‡						
Full Model	$1.5 \times 10^{-3}(T)$	$3.2 \times 10^{-8}(T)$	$1.3 \times 10^{-2}(T)$	$3.7 \times 10^{-7}(T)$	$9.3 \times 10^{-8}(T)$	$1.3 \times 10^{-7}(T)$
Layer 4	$2.7 \times 10^{-1}(T)$	$1.1 \times 10^{-6}(T)$	$2.6 \times 10^{-3}(T)$	$2.9 \times 10^{-6}(T)$	$2.3 \times 10^{-7}(T)$	$2.7 \times 10^{-6}(T)$
Lora	$5.6 \times 10^{-3}(T)$	$1.5 \times 10^{-4}(T)$	$3.6 \times 10^{-3}(T)$	$7.3 \times 10^{-6}(T)$	$1.3 \times 10^{-6}(T)$	$1.2 \times 10^{-6}(T)$
Decoder Only	$3.8 \times 10^{-1}(T)$	$2.1 \times 10^{-4}(T)$	$7.5 \times 10^{-4}(T)$	$5.2 \times 10^{-5}(T)$	$2.3 \times 10^{-4}(T)$	$1.5 \times 10^{-6}(T)$
Embeding	$9.8 \times 10^{-7}(T)$	$4.1 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$	$3.6 \times 10^{-4}(T)$	$1.0 \times 10^{-1}(T)$	$3.7 \times 10^{-4}(T)$
Input Adapt	$1.5 \times 10^{-6}(T)$	$5.9 \times 10^{-1}(T)$	$5.7 \times 10^{-4}(T)$	$2.0 \times 10^{-4}(T)$	$2.1 \times 10^{-1}(T)$	$7.7 \times 10^{-5}(T)$
Prefix	$1.3 \times 10^{-5}(T)$	$1.3 \times 10^{-2}(T)$	$1.2 \times 10^{-3}(T)$	$2.2 \times 10^{-4}(T)$	$2.3 \times 10^{-3}(T)$	$1.2 \times 10^{-5}(T)$
Scratch	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
Zero Shot	$0.0(T)$	$5.6 \times 10^{-7}(T)$	$0.0(T)$	$2.0 \times 10^{-2}(T)$	$8.8 \times 10^{-8}(T)$	$8.3 \times 10^{-3}(T)$

Table 4: Obstacle Avoidance Cumulative Rewards for each testing morphology. Values show mean (top) and standard deviation (bottom). † statistical significance compared to Zero Shot and ‡ statistical significance to Scratch (p < 0.01). P-values and the hypothesis test run (T: t-test, M: Mann-Whitney) comparing against Zero shot and Scratch results.

Morphology	1	2	3	4	5	6
Full Model	$2652.41^{\dagger\ddagger}$	$3101.42^{\dagger\ddagger}$	$1705.64^{\dagger}$	$3577.09^{\dagger\ddagger}$	$3219.75^{\dagger\ddagger}$	$3558.26^{\dagger\ddagger}$
	± 193.57	± 177.17	± 4.16	± 341.74	± 199.13	± 365.21
Layer 4	$2246.88^{\dagger}$	$2684.70^{\dagger\ddagger}$	$1592.29^{\dagger\ddagger}$	$3276.76^{\dagger\ddagger}$	$2888.34^{\dagger\ddagger}$	$3194.19^{\ddagger}$
	± 184.03	± 85.63	± 140.05	± 314.17	± 64.04	± 351.87
Lora	$2137.75^{\dagger\ddagger}$	$2585.71^{\dagger\ddagger}$	$1672.75^{\dagger\ddagger}$	$3191.40^{\dagger\ddagger}$	$2851.48^{\dagger\ddagger}$	$3189.01^{\ddagger}$
	± 116.21	± 191.00	± 12.63	± 320.51	± 107.16	± 319.76
Decoder Only	$2263.74^{\dagger}$	$2531.13^{\dagger\ddagger}$	$1456.02^{\dagger\ddagger}$	$3061.26^{\ddagger}$	$2672.06^{\dagger\ddagger}$	$3132.18^{\ddagger}$
	± 186.99	± 161.72	± 218.88	± 360.37	± 160.55	± 302.67
Embeding	$1863.25^{\dagger\ddagger}$	$2189.46^{\dagger}$	$1556.72^{\dagger\ddagger}$	$2882.24^{\ddagger}$	$2398.29^{\dagger}$	$2877.35^{\ddagger}$
	± 94.00	± 167.27	± 151.65	± 361.86	± 139.50	± 417.09
Input Adapt	$1839.40^{\dagger\ddagger}$	$2159.73^{\dagger}$	$1458.49^{\dagger\ddagger}$	$2929.91^{\ddagger}$	$2367.39^{\dagger}$	$2833.04^{\ddagger}$
	± 117.50	± 125.84	± 206.98	± 356.45	± 142.90	± 312.07
Prefix	$1841.45^{\dagger\ddagger}$	$2334.01^{\dagger}$	$1514.42^{\dagger\ddagger}$	$2877.43^{\ddagger}$	$2538.31^{\dagger\ddagger}$	$2935.63^{\ddagger}$
	± 142.33	± 133.00	± 186.34	± 324.42	± 119.05	± 293.07
Scratch	$2334.75^{\dagger}$	$2119.16^{\dagger}$	$1843.23^{\dagger}$	2112.34	$2265.47^{\dagger}$	$2144.60^{\dagger}$
	± 92.45	± 124.11	± 99.74	± 216.38	± 124.23	± 123.50
Zero Shot	1300.21	1184.87	332.64	2467.45	1295.92	2488.64
	± 117.01	± 248.07	± 114.42	± 246.89	± 202.99	± 274.84
P-Values comparing against Zeroshot Performance†						
Full Model	$2.2 \times 10^{-6}(T)$	$1.5 \times 10^{-6}(T)$	$9.7 \times 10^{-9}(T)$	$7.6 \times 10^{-4}(T)$	$8.5 \times 10^{-7}(T)$	$1.6 \times 10^{-3}(T)$
Layer 4	$2.4 \times 10^{-5}(T)$	$3.1 \times 10^{-6}(T)$	$6.8 \times 10^{-7}(T)$	$3.7 \times 10^{-3}(T)$	$3.9 \times 10^{-7}(T)$	$1.3 \times 10^{-2}(T)$
Lora	$7.6 \times 10^{-6}(T)$	$1.9 \times 10^{-5}(T)$	$1.2 \times 10^{-8}(T)$	$7.2 \times 10^{-3}(T)$	$8.4 \times 10^{-7}(T)$	$1.1 \times 10^{-2}(T)$
Decoder Only	$2.3 \times 10^{-5}(T)$	$1.7 \times 10^{-5}(T)$	$1.7 \times 10^{-5}(T)$	$2.6 \times 10^{-2}(T)$	$5.4 \times 10^{-6}(T)$	$1.4 \times 10^{-2}(T)$
Embeding	$6.9 \times 10^{-5}(T)$	$1.5 \times 10^{-4}(T)$	$7.9 \times 10^{-3}(M)$	$9.5 \times 10^{-2}(T)$	$1.9 \times 10^{-5}(T)$	$1.6 \times 10^{-1}(T)$
Input Adapt	$1.9 \times 10^{-4}(T)$	$1.1 \times 10^{-4}(T)$	$1.2 \times 10^{-5}(T)$	$6.5 \times 10^{-2}(T)$	$2.5 \times 10^{-5}(T)$	$1.4 \times 10^{-1}(T)$
Prefix	$8.9 \times 10^{-4}(T)$	$3.8 \times 10^{-5}(T)$	$4.7 \times 10^{-6}(T)$	$7.9 \times 10^{-2}(T)$	$5.6 \times 10^{-6}(T)$	$5.7 \times 10^{-2}(T)$
Scratch	$0.0(T)$	$5.6 \times 10^{-7}(T)$	$0.0(T)$	$2.0 \times 10^{-2}(T)$	$8.8 \times 10^{-8}(T)$	$8.3 \times 10^{-3}(T)$
Zero Shot	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
P-Values comparing against Scratch Performance‡						
Full Model	$1.5 \times 10^{-3}(T)$	$3.2 \times 10^{-8}(T)$	$1.3 \times 10^{-2}(T)$	$3.7 \times 10^{-7}(T)$	$9.3 \times 10^{-8}(T)$	$1.3 \times 10^{-7}(T)$
Layer 4	$2.7 \times 10^{-1}(T)$	$1.1 \times 10^{-6}(T)$	$2.6 \times 10^{-3}(T)$	$2.9 \times 10^{-6}(T)$	$2.3 \times 10^{-7}(T)$	$2.7 \times 10^{-6}(T)$
Lora	$5.6 \times 10^{-3}(T)$	$1.5 \times 10^{-4}(T)$	$3.6 \times 10^{-3}(T)$	$7.3 \times 10^{-6}(T)$	$1.3 \times 10^{-6}(T)$	$1.2 \times 10^{-6}(T)$
Decoder Only	$3.8 \times 10^{-1}(T)$	$2.1 \times 10^{-4}(T)$	$7.5 \times 10^{-4}(T)$	$5.2 \times 10^{-5}(T)$	$2.3 \times 10^{-4}(T)$	$1.5 \times 10^{-6}(T)$
Embeding	$9.8 \times 10^{-7}(T)$	$4.1 \times 10^{-1}(T)$	$6.7 \times 10^{-4}(M)$	$3.6 \times 10^{-4}(T)$	$1.0 \times 10^{-1}(T)$	$3.7 \times 10^{-4}(T)$
Input Adapt	$1.5 \times 10^{-6}(T)$	$5.9 \times 10^{-1}(T)$	$5.7 \times 10^{-4}(T)$	$2.0 \times 10^{-4}(T)$	$2.1 \times 10^{-1}(T)$	$7.7 \times 10^{-5}(T)$
Prefix	$1.3 \times 10^{-5}(T)$	$1.3 \times 10^{-2}(T)$	$1.2 \times 10^{-3}(T)$	$2.2 \times 10^{-4}(T)$	$2.3 \times 10^{-3}(T)$	$1.2 \times 10^{-5}(T)$
Scratch	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$	$1.0(T)$
Zero Shot	$0.0(T)$	$5.6 \times 10^{-7}(T)$	$0.0(T)$	$2.0 \times 10^{-2}(T)$	$8.8 \times 10^{-8}(T)$	$8.3 \times 10^{-3}(T)$

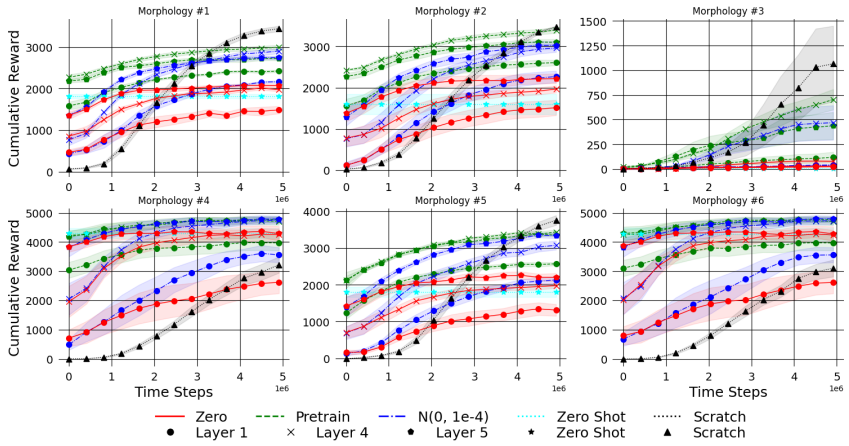


Figure 7: Choice of initialization and injection layers of prefix tuning in flat terrain. Initial zero-shot results of E2E learning are plotted to compare affect of prefixes.

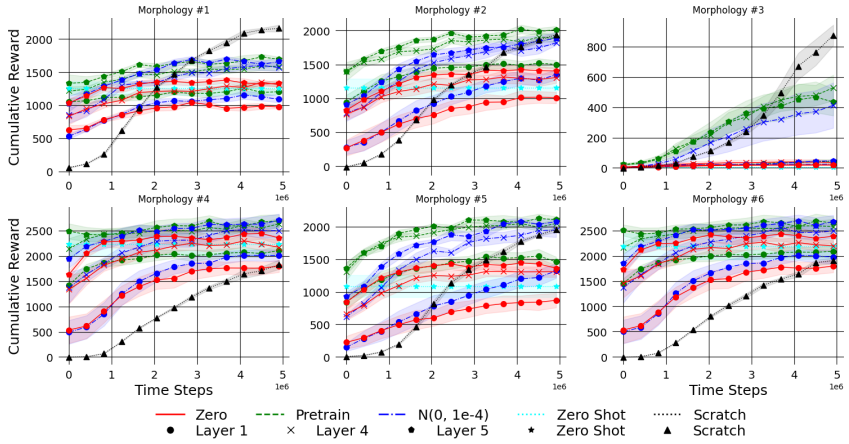
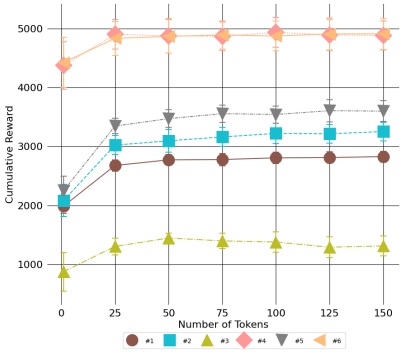
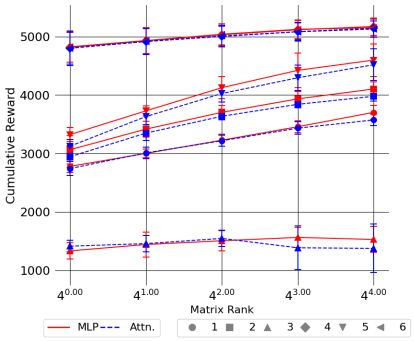


Figure 8: Choice of initialization and injection layers of prefix tuning in obstacle avoidance. Initial zero-shot results of E2E learning are plotted to compare affect of prefixes.

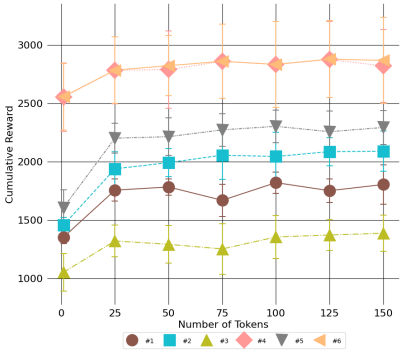


(a) Number of randomly initialized prefix tokens

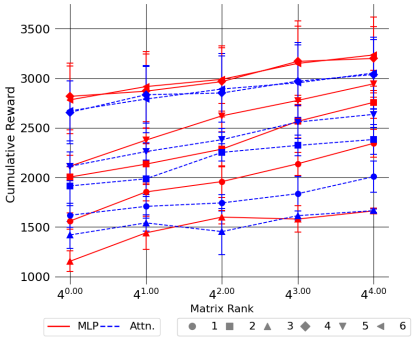


(b) Lora in different layers of fifth transformer block.

Figure 9: Ablation studies on number of prefix tokens and LoRA in flat terrain task.



(a) Number of randomly initialized prefix tokens



(b) Lora in different layers of fifth transformer block.

Figure 10: Ablation studies on number of prefix tokens and LoRA in obstacle avoidance task.