Multi-Task Reinforcement Learning Enables Parameter Scaling

Reginald McLean, Evangelos Chatzaroulas, J.K. Terry, Isaac Woungang, Nariman Farsad, Pablo Samuel Castro

Keywords: Reinforcement learning, multi-task reinforcement learning, parameter scaling

Summary

Multi-task reinforcement learning (MTRL) aims to endow a single agent with the ability to perform well on multiple tasks. Recent works have focused on developing novel sophisticated architectures to improve performance, often resulting in larger models; it is unclear, however, whether the performance gains are a consequence of the architecture design itself or the extra parameters. We argue that gains are mostly due to scale by demonstrating that naïvely scaling up a simple MTRL baseline to match parameter counts outperforms the more sophisticated architectures, and these gains benefit most from scaling the critic over the actor. Additionally, we explore the training stability advantages that come with task diversity, demonstrating that increasing the number of tasks can help mitigate plasticity loss. Our findings suggest that MTRL's simultaneous training across multiple tasks provides a natural framework for beneficial parameter scaling in reinforcement learning, challenging the need for complex architectural innovations.

Contribution(s)

1. We perform a thorough empirical evaluation that shows that parameter scaling with simple architectures exceeds performance of complex MTRL-specific architectures on Meta-World benchmarks.
   Context: Increasing the scale of reinforcement learning models has been of particular interest of late (Obando Ceron et al., 2024; Nauman et al., 2024b; Schwarzer et al., 2023). This interest is likely due to the advances that other areas of research, such as computer vision (Zhai et al., 2022) and natural language processing (Kaplan et al., 2020), have made thanks to increasing the amount of data, compute, and model parameters during training. While some works have noted that naïvely scaling can be detrimental to performance (Obando Ceron et al., 2024), MTRL models have been quietly increasing in size. We compare results across recent multi-task reinforcement learning (MTRL) specific architectures, finding that increasing the parameter count on a simple feed forward baseline architecture outperforms the MTRL specific architectures.
2. We uncover a significant inverse relationship between task diversity and plasticity loss, showing that models trained on more tasks maintain neuron activation even as parameter counts increase.
   Context: When increasing the parameter scale alone, recent works have found that networks exhibit a loss of network plasticity (Lyle et al., 2023), or even a significant drop in performance when naïvely scaling (Obando Ceron et al., 2024). Across three different sizes of benchmark tasks, ten, twenty-five, and fifty tasks, we uncover a previously unknown relationship between the number of tasks and the number of parameters. We find that by increasing both model scale and the number of tasks being trained on can be an effective method to mitigate plasticity loss.
3. We empirically identify that critic scaling provides greater benefits than actor scaling in MTRL settings.
   Context: Recently, Nauman et al. (2024b) uncovered that the performance of the critic of an actor-critic algorithm was more affected by scale in single task settings. We extend this question to MTRL and have similar findings: that the scaling of the critic is more tightly coupled with the performance of the MTRL agent as a whole than the scale of the actor.

Multi-Task Reinforcement Learning Enables Parameter Scaling

Reginald McLean[1,2,*], **Evangelos Chatzaroulas**[3,*], **J.K. Terry**[2], **Isaac Woungang**[1,†], **Nariman Farsad**[1,†], **Pablo Samuel Castro**[4,5,†]
reginald.mclean@torontomu.ca, e.chatzaroulas@surrey.ac.uk

[1]**Department of Computer Science, Toronto Metropolitan University, Canada**
[2]**Farama Foundation**
[3]**Department of Computer Science, University of Surrey, United Kingdom**
[4]**Mila - Québec AI Institute**
[5]**Department of Computer Science and Operations Research, Université de Montréal, Canada**

[*] indicates equal contribution, † indicates equal advising

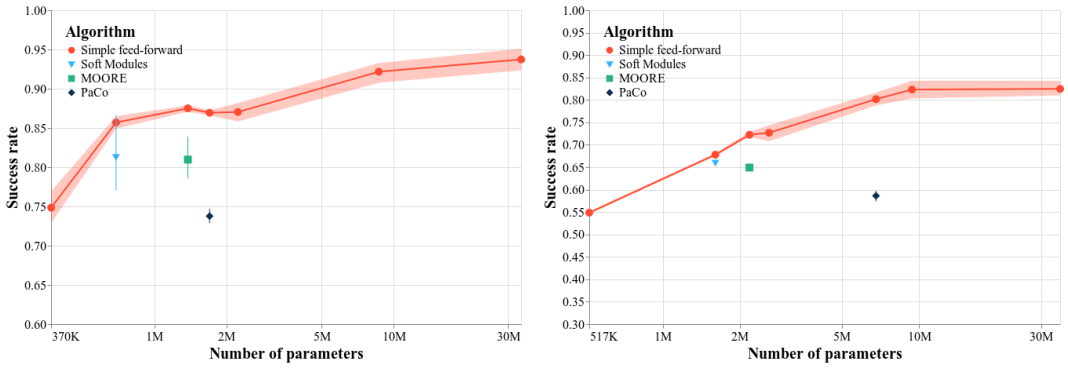
Abstract

Multi-task reinforcement learning (MTRL) aims to endow a single agent with the ability to perform well on multiple tasks. Recent works have focused on developing novel sophisticated architectures to improve performance, often resulting in larger models; it is unclear, however, whether the performance gains are a consequence of the architecture design itself or the extra parameters. We argue that gains are mostly due to scale by demonstrating that naïvely scaling up a simple MTRL baseline to match parameter counts outperforms the more sophisticated architectures, and these gains benefit most from scaling the critic over the actor. Additionally, we explore the training stability advantages that come with task diversity, demonstrating that increasing the number of tasks can help mitigate plasticity loss. Our findings suggest that MTRL's simultaneous training across multiple tasks provides a natural framework for beneficial parameter scaling in reinforcement learning, challenging the need for complex architectural innovations.

1 Introduction

Reinforcement learning (RL) has emerged as the go-to method for super-human decision making in the past few years, achieving incredible results in Atari games (Schwarzer et al., 2023), the control of stratospheric balloons (Bellemare et al., 2020), and the magnetic control of tokomak plasma for nuclear fusion (Degrave et al., 2022). The rise of RL has been in large part thanks to the introduction of neural networks for modeling actions and/or values from learning on many iterations of trial and error. During this process, it is possible for the RL agent to learn to utilize and exploit specific features to be successful. One of the drawbacks of this exploitation is that the RL agent can be limited in the number of tasks that it can accomplish, as the learned features may not generalize to other tasks.

To overcome this, RL researchers have developed methods of training multi-task reinforcement learning (MTRL) agents. These agents learn to accomplish multiple tasks using only a single policy and value function, with the aim of being able to bootstrap learning in one task by simultaneously learning on related tasks. In order to evaluate MTRL agents, researchers often utilize the Meta-World suite (Yu et al., 2020b), which provides two sets of benchmarks with different difficulty levels based



(a) (b)

Figure 1: Effects of scaling the number of parameters of actor and critic in multi-task, multi-head, soft actor-critic (Yu et al., 2020b), in the Meta-World (a) ten task and (b) fifty task benchmarks over ten random seeds. Y-axes report the inter-quartile mean. Shaded regions, for plots, and bars, for singular points, denote standard errors. MTRL specific methods are plotted here as singular points, while the Simple FF method is the result scaling a simple baseline to a similar number of parameters.

on the number of tasks. Meta-World's unified observation and action space, along with its consistent robot workspace, makes it particularly suitable for MTRL research.

Researchers have approached MTRL through two main directions: gradient projection methods to limit interference between tasks (Yu et al., 2020a), and MTRL-specific architectures designed with human intuition about how agents should solve multiple tasks. While recent work (Kurin et al., 2022) has shown that gradient projection methods do not significantly outperform well-regularized baselines, MTRL-specific architectures continue to push performance boundaries. However, these architectural innovations often lack comparison to simple baselines with similar parameter counts.

Recent works in computer vision (Zhai et al., 2022), and language modeling (Kaplan et al., 2020) have found that increasing the amount of compute to train the model, whether it is more parameters or more gradient steps, leads to models that perform better at their respective tasks. There may even be some signs of emergent capabilities at these scales (Wei et al., 2022). However, scaling in RL has proven more challenging, with research showing that naïvely increasing model parameters can actually harm performance (Obando Ceron et al., 2024). Successful scaling in RL requires specific adaptations, such as modified network architectures (Obando Ceron et al., 2024), policy and value function regularization (Nauman et al., 2024b), or limiting parameter increases to world models only (Hansen et al., 2024; Hafner et al., 2023).

Similar to recent attempts to scale model size in single-task RL that utilize architectural changes such as Soft Mixture of Experts (Obando Ceron et al., 2024), or skip connections with regularization (Nauman et al., 2024b), recent MTRL works have been able to leverage architectures to scale their models. However, there is not much insight into whether the MTRL architecture or the number of parameters provides the increase in performance. In order to investigate the correlation between model performance and scale, we first implement and benchmark four recent MTRL architectures on two Meta-World benchmarks. We demonstrate that a simple MTRL baseline, scaled to match these architectures' parameter counts, surpasses their performance. Further scaling of this basic architecture yields additional performance gains, as shown in Figure 1.

Beyond raw performance improvements, we investigate whether both policy and value functions benefit equally from increased scale. Our analysis reveals that the scaling of the critic is directly related to MTRL performance. We also examine how scaling affects model plasticity - the ability to continually learn. Interestingly, we find that large models trained on few tasks show signs of reduced plasticity, while similarly sized models trained on more tasks maintain their adaptability. These findings suggest an important relationship between model capacity and task diversity in MTRL.

Based on these findings, the contributions of our work include:

- Empirical evaluation that shows that parameter scaling with simple architectures **exceeds** performance of complex MTRL-specific architectures on Meta-World benchmarks.
- We empirically identify that critic scaling provides greater benefits than actor scaling in MTRL settings.
- We discover a previously unknown relationship between model size and task count: plasticity loss in large models can be mitigated by increasing the number of training tasks.
- An open source code base that can be used to further advance MTRL research.[1]

2 Related Work

Before the introduction of a unified MTRL benchmark, single task RL benchmarks such as the Arcade Learning Environment (ALE) (Bellemare et al., 2013) were leveraged as a platform for MTRL research. However, using the ALE can be difficult due to large visual differences between games which can lead to negative interference (Yu et al., 2020b). Thus the introduction of a unified benchmark specifically for MTRL research was needed, such as Meta-World. Meta-World utilizes a unified observation and action space, and a single robot & workspace, which allows for effective multi-task and meta reinforcement learning research.

Upon publication, Yu et al. (2020b) produced an extension to Soft Actor-Critic (SAC) (Haarnoja et al., 2018), specifically for MTRL. This extension, multi-task multi-head SAC (MTMHSAC), maintains N independent entropy heads & utilizes independent action heads for each task. The next MTRL architecture to be introduced was the Soft-Modularization (SM) architecture (Yang et al., 2020). This SM architecture posited that it is unclear whether gradients from all tasks should affect all parameters and which parameters should be shared. Thus to overcome this issue, SM proposed a soft-routing method. The **Pa**rameter-**Co**mpositional (PaCo) MTRL architecture creates a shared base representation and task-specific compositional vectors (Sun et al., 2022). For each task, parameters are generated by combining the shared base representation with the corresponding task-specific vector. This design ensures that the shared components learn from all tasks while maintaining task-specific adaptability through the compositional vectors. The architecture explicitly separates gradient flow between shared and task-specific components, with shared parameters updated using all task losses while task-specific components only receive gradients from their respective tasks. Lastly, a **M**ixture **O**f **OR**thogonal **E**xperts (MOORE) was proposed in Hendawy et al. (2024). In this architecture a mixture of experts processes the incoming state into a shared representation space, which is then used in combination with task specific information to generate task relevant representations.

Recent works in RL have attempted to scale the number of parameters to improve performance of the RL agent. One of the first attempts to do this was a model-based approach that showed it was possible to learn a policy entirely in a world model (Łukasz Kaiser et al., 2020). Building upon that work, there have been multiple attempts to scale models when using Łukasz Kaiser et al. (2020)'s Atari 100k dataset, including representation learning approaches (Schwarzer et al., 2021; Farebrother et al., 2023; Schwarzer et al., 2023), soft mixture-of-experts (Obando Ceron et al., 2024), and regularization based (Nauman et al., 2024b)[2]. Recent work in model-based RL have shown that scaling the world model achieves significant improvements (Hafner et al., 2023; Hansen et al., 2024; Wang et al., 2024) with the caveat that most of the scaled parameters are in the world model itself, not the RL agent. Recent works have also shown that naïvely scaling parameters can be detrimental to overall performance of the RL agent (Obando Ceron et al., 2024).

Recently, the single task reinforcement learning has seen an influx of retrospective works that have shaped our approach to this work. This line of work began with a study into the implementation details of Proximal Policy Optimization and Trust-Region Policy optimization algorithms (Engstrom

[1] https://github.com/rainx0r/mtrl

[2] Appendix E of Nauman et al. (2024b) shows some signs that scaling also aids in MTRL

et al., 2020), revealing how subtle implementation choices can significantly impact performance. These findings were further extended into a large scale study into what matters in on-policy RL (Andrychowicz et al., 2021), while Taiga et al. (2023) demonstrated how pre-training on simple game variants affects the generalization abilities of RL agents of various sizes. Recent works have also investigated why RL agents lose their network plasticity and have proposed several mitigation techniques including: loss landscape stabilization (Lyle et al., 2023), plasticity injection (Nikishin et al., 2023), neuron recycling (Sokar et al., 2023), and appropriate regularization (Nauman et al., 2024a). These plasticity findings are particularly relevant to MTRL, where agents must maintain flexibility to learn multiple tasks simultaneously. Other works have investigated the effects and consistency of hyper-parameter selections (Ceron et al., 2024), which informed our systematic approach to evaluating MTRL architectures across different configurations and settings.

While these architectural innovations and optimization techniques have advanced MTRL, our work suggests that simpler approaches focusing on parameter scaling and critic accuracy may be equally or more effective. This aligns with recent trends in other areas of deep learning where scale has proven to be a crucial factor in performance improvements. Our findings indicate that the field's focus on complex architectural solutions may be better directed toward ensuring robust critic training and adequate model capacity.

3 Background

Reinforcement learning attempts to provide solutions to sequential decision making problems where an agent interacts with an environment during a set of discrete time steps $t = 0, 1, 2, 3, ...$ During each time step t, the agent receives a representation of an environment's state s_t, selects an action a_t, and applies that action to the environment. As a consequence of applying this action, the agent then receives a new state s_{t+1}, and a scalar reward r_{t+1} for transitioning from state s_t to s_{t+1}. This framework is formalized as a Markov Decision Process (MDP), represented as $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho)$ with the set of all possible states $\mathcal{S}$, the set of available actions $\mathcal{A}$, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ defines the transition function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbf{R}$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and ρ is the initial state distribution. In order to select actions, an agent's behavior is formalized using a policy $\pi : \mathcal{S} \to \mathcal{A}$. The objective of the agent then is to maximize the expected sum of returns through interactions with the environment according to policy π as $\mathbf{E}_{(s_t,a_t)\sim\mathcal{P}_\pi}[R(\tau)]$ where $\mathcal{P}_\pi$ is the transition distribution induced by policy π.

The MTRL problem extends the usage of an MDP across all N tasks the agent must interact with. We assume that task i is drawn from a distribution of tasks $p(\mathcal{T})$. Instead of a single MDP, the MTRL agent must interact with MDPs $(\mathcal{S}, \mathcal{A}, \mathcal{P}_i, \mathcal{R}_i, \gamma, \rho_i)$ where the transition function $\mathcal{P}_i$, and reward function $\mathcal{R}_i$ are specific to sampled task i. The goal of the agent is to learn a single, task conditioned, policy $\pi(a_t|s_t, z)$ where z is a representation of which task i the agent is interacting with. The objective of the agent is now to maximize expected returns across all N tasks as $\mathbf{E}_{i\sim p(\mathcal{T})}[\mathbf{E}_{(s_t,a_t)\sim\mathcal{P}_\pi}[R_i(\tau)]]$. In this work we leverage Soft Actor-Critic (Haarnoja et al., 2018), as the underlying RL algorithm to maximize the expected returns.

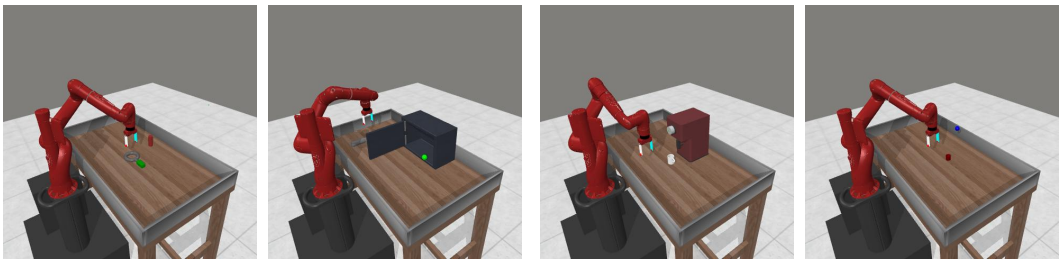
4 Experimental Methodology

This section outlines the experimental setup of our study. We aim to answer the following questions:

1. Does increasing the number of parameters account for the recent advances in MTRL?
2. Does increasing the number of parameters predictably increase MTRL performance?
3. Where does scaling aid in improving performance the most?
4. How does scaling the number of parameters affect plasticity loss? Does this effect change with different number of tasks?

4.1 Benchmark environments

We focus on the two available benchmarks from Meta-World (Yu et al., 2020b). The first benchmark contains 10 tasks, we will refer to it as MT10. The second benchmark contains fifty tasks, which we will refer to as MT50. In Figure 2 we show some of the tasks from Meta-World. We refer the interested reader to Yu et al. (2020b) for the remaining tasks. Each state vector has a one-hot task ID vector appended to it, indicating which task the state vector belongs to. We train our MTRL agent in MT10 for twenty million timesteps and one-hundred million timesteps for MT50, with two million timesteps per task. We utilize per-task replay buffers, with an equal number of samples per task for each actor & critic update. We evaluate our policies for fifty episodes, every forty episodes of training. For MT10 this equates to every 200,000 timesteps while MT50 is evaluated every 1,000,000 timesteps. For each experiment we report results across 10 random seeds. Unless otherwise noted, we report the inter-quartile means (IQM) following recommendations from Agarwal et al. (2021).



(a) (b) (c) (d)

Figure 2: Sample tasks from Meta-World. (a) assembly, where the MTRL agent must grasp a wrench and place it on a peg. (b) door close, where the agent must swing the door from the current position to the green goal to close it. (c) coffee button, where the agent must press a button on the front of the coffee machine. (d) pick place, where the agent must grasp the red object and move it to the blue goal.

4.2 Architectures

We implement four MTRL architectures from the MTRL literature: multi-task, multi-head, Soft Actor-Critic (MTMHSAC) (Yu et al., 2020b), MTRL with Soft Modularization (SM) (Yang et al., 2020), MTRL with Mixture of Orthogonal Experts (MOORE) (Hendawy et al., 2024), and Parameter Compositional MTRL (PaCo) (Sun et al., 2022). These architectures were selected to represent the evolution and diversity of MTRL approaches, ranging from fundamental baseline methods, MTMHSAC, to sophisticated state-of-the-art architectures like MOORE. Each architecture embodies different design philosophies in handling multi-task learning, from MTMHSAC's simple feed forward design with separate action heads to MOORE's expert-based parameter sharing strategy. We include a diagram of the respective architectures in Figure 3. In Supplementary Materials D we report the baseline results of our implemented architectures in comparison to the published results.

In addition to their architectural contributions, SM and PaCo incorporate additional optimization tweaks beyond their architectural innovations. To isolate and evaluate the impact of architectural choices alone, we implement these methods without their supplementary losses. This focused comparison necessarily means our results may differ from those reported in the original papers. Furthermore, SM's published results used an earlier version of Meta-World's reward functions, making direct numerical comparisons to their reported results inappropriate.

4.3 Hyperparameters

To ensure fair comparisons across architectures, we standardize a set of common MTRL hyperparameters including batch size, log standard deviation bounds, and replay buffer capacity. While some

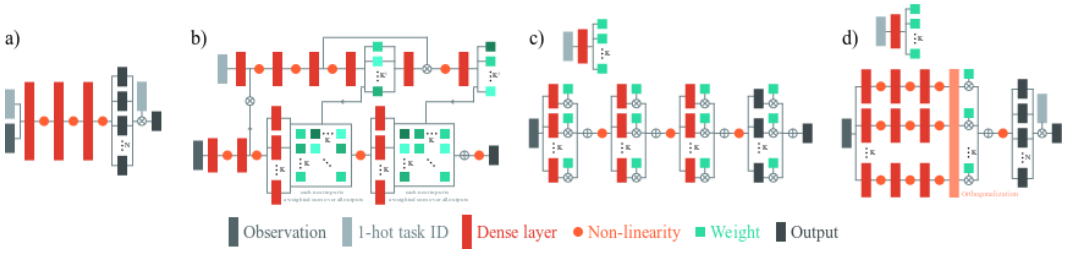


Figure 3: Visualization of architectures used in this work. (a) MTMHSAC from Yu et al. (2020b), (b) Soft-Modularization from Yang et al. (2020), (c) PaCo from Sun et al. (2022), and (d) MOORE from Hendawy et al. (2024).

of our chosen values differ from those in the original publications, this standardization is essential for meaningful comparative analysis. For instance, we adopt a minimum log standard deviation of e^{-20} across all methods, in contrast to MOORE's original e^{-10}, and implement a uniform replay buffer size of $100k \times N$ (where N is the number of tasks) rather than SM's larger $200k \times N$ buffer. These modifications enable direct architectural comparisons by eliminating performance variations that might stem from differing hyperparameter choices.

5 Empirical Results

5.1 Comparison with MTRL Architectures

Figure 1 reports results on the MT10 and MT50 benchmarks across various parameter scales. In order to investigate scaling the number of parameters in our Actor Critic algorithm, we first choose parameter scales that match the number of parameters in each of our selected baselines, SM, MOORE, and PaCo. In Figure 1 we show the IQM plots of using a simple feed-forward neural network architecture combined with increasing the number of parameters to match our selected baselines. To do so we use 3 hidden layers[3] for our actor and critic, and increase the width of each neural network layer until the number of parameters in the actor and critic is, at most, equal to the parameter count of the MTRL specific architecture. As Figure 1 shows, we find that simply scaling the number of parameters to match the number of parameters in each baseline matches or, exceeds baseline architecture performance. Additionally, in Appendix E, we show the results from increasing the scale of the MTRL architectures where we find that performance at similar parameter counts is equivalent to the baseline. These results confirm our initial hypothesis that scaling accounts for recent MTRL advances.

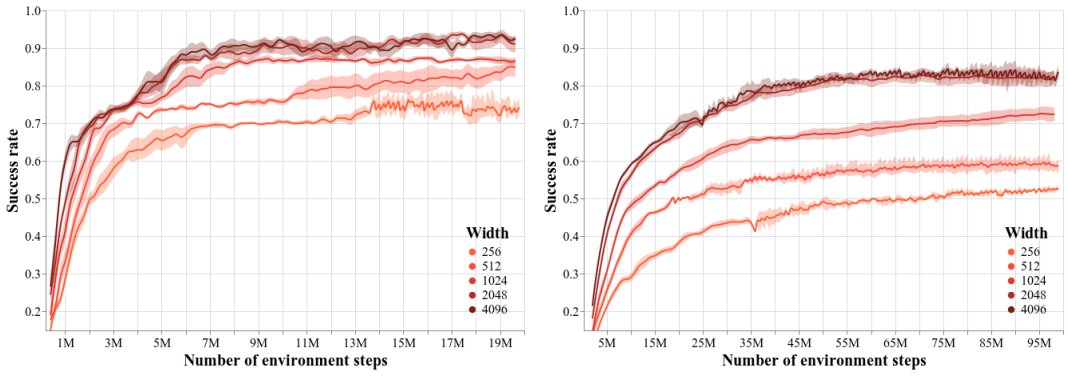
5.1.1 Increasing Scale

To further explore the effect that scaling has on MTRL performance in Meta-World, we continue experimenting with increasing the parameter counts of our models. In Figure 4a and 4b we report the IQM on various scales of models. We find that as we continue to scale, there are some benefits of making the models increasingly larger. However, there does seem to be diminishing returns from scaling for performance.

5.2 Who benefits the most from scale, the actor or critic?

In our previous experiments we show that the effect of scaling the number of parameters can have great effects in both the MT10 and MT50 benchmarks. Now we seek to answer whether the actor

[3]We also explored using 4 hidden layers and found that performance increased in comparison to the MTRL architecture baselines, but provided no gain compared to the 3 layer depth networks. We include those results in Supplementary Material B. We hypothesize that going to four or more layers may require architecture changes, such as skip connections He et al. (2016), in order to exceed three layer performance due to gradient vanishing.



(a) (b)

Figure 4: Extended Scaling Results. Each line is the average IQM produced by a feed-forward network scaled to a certain parameter count, with shaded regions indicating 95% CIs. Colours indicate which parameter count the model used. (a) MT10 scaling results, (b) MT50 scaling results.

or the critic benefits from the effects of scale more. To perform this experiment, we use the MT10 benchmark and select the network from the extended scaling experiments with the layer width of 1024 as our baseline. This model size provides a decent trade-off between performance and speed for these experiments. In order to determine whether the actor or critic benefits from scaling more, we increase the actor's layer width to 1024 while maintaining a critic width of 400, and vice versa for the actor. In addition to this, we further reduce the size of our models from a width of 400 to 200 to further limit the ability for the shrunk model to learn.

Figure 5 reveals a clear asymmetry in how scaling affects different components of MTRL architectures. When using our actor and critic both with width 1024 as baseline, we observe that experiments maintaining a critic of size 1024 consistently achieve comparable performance to this baseline, regardless of actor size reduction. Conversely, configurations with a full-sized actor (width 1024) but reduced critic (width 400 or 200) show substantial performance degradation. This asymmetric response to parameter reduction demonstrates that critics benefit more significantly from increased model capacity in multi-task settings. This finding aligns with our intuitions, as critics must approximate multiple state-action value functions across diverse tasks, thereby requiring greater representational capacity. Our results extend and confirm similar observations from single-task RL settings reported by Nauman et al. (2024b), suggesting this asymmetry may be a fundamental property of actor-critic architectures rather than a task-specific phenomenon.

5.3 Impact of Scale on Plasticity

Next we investigate the effects that scaling has on plasticity loss in our policy and value functions. We implement several metrics for approximating the loss of plasticity (Lyle et al., 2023), similar to the prior work of Nauman et al. (2024a). We find that the percentage of dormant neurons (Sokar et al., 2023) to be the most representative metric, but we include plots for each of the remaining metrics in Supplementary Materials C. In Figure 6, we report the percentage of dormant neurons when training on MT10 and MT50. When examining the MT10 dormant neurons, we find that as we scale the number of parameters we see an increase in the number of dormant neurons, with our highest parameter count model Width 4096 exhibiting the highest percentage of dormant neurons. This somewhat contrasts with our earlier results from scaling. While the performance in terms of success rate improves, there are signs of losing the ability to learn new tasks when training on MT10. When examining the MT50 dormant neuron percentages we find that at these parameter scales with fifty tasks, there is almost no indication of plasticity loss. This indicates that there is a relationship between the number of parameters in the actor & critic, the number of tasks that the agent is interacting with, and the plasticity of the agent's networks.

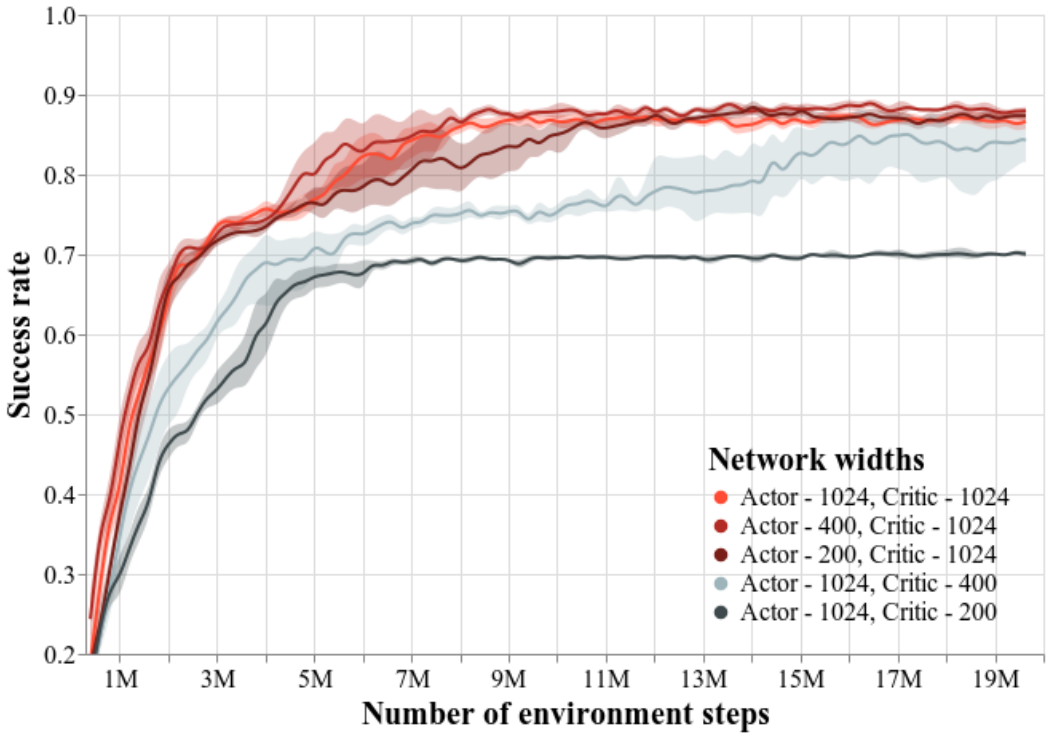


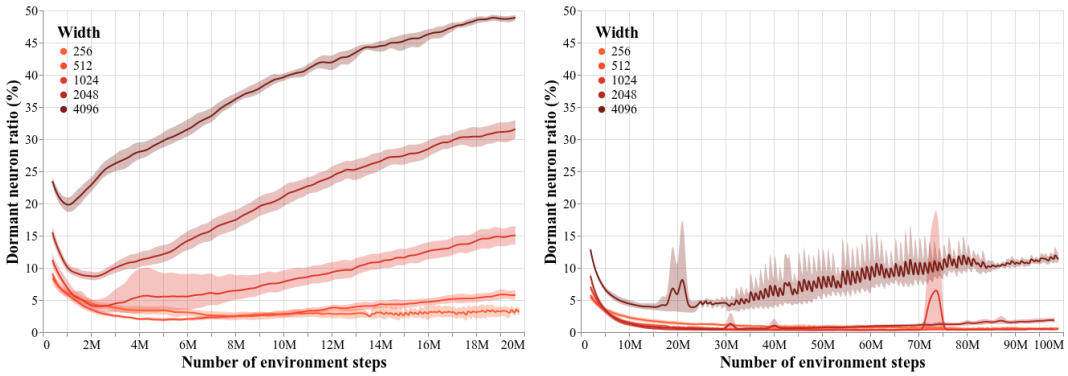
Figure 5: Where does scale matter the most? Here we report the IQM through training across various actor & critic size configurations. Our baseline model uses equally sized actor & critics using layer width 1024. We then iterate over various actor & critic widths to determine which component benefits more from scale. These results show that the ability of the critic is more affected by the scaling of parameters, in line with results from Nauman et al. (2024b) in the single task setting.

This observation reveals a critical insight into the dynamics of plasticity in MTRL systems: the ratio between model capacity and task diversity appears to govern plasticity loss. In MT10, the limited task diversity relative to model size allows networks to specialize neurons for specific tasks, leaving portions of the network underutilized and potentially dormant. However, in MT50, the increased task diversity creates sufficient learning pressure to activate neurons throughout the network, even as model size increases. This finding suggests an important design principle for MTRL architectures—model scaling should be balanced with commensurate increases in task diversity to maintain neuronal activity and preserve plasticity. The absence of dormant neurons in large models trained on MT50 indicates that task diversity may serve as a natural regularizer against plasticity loss, enabling efficient utilization of increased parameter capacity without sacrificing the ability to learn new tasks.

5.4 Task & Parameter Scaling

To further examine the relation between the number of parameters, the number of tasks, and an agent's network plasticity, we perform the following experiments. First we create a new set of tasks called MT25 that contains the ten tasks from MT10, and the remaining tasks are a mix of solved and unsolved tasks from MT50. Next, we train MTRL agents across various scales of model sizes. Figure 7 shows the dormant neuron percentages plotted against the number of model parameters for the ten, twenty, and fifty tasks benchmark sets. From this experiment we find that large models trained on a large number of tasks exhibit much lower rates of dormant neurons, and less plasticity loss.

This finding has significant implications for scaling in RL. The inverse relationship between task diversity and plasticity loss suggests that scaling both model size and task count simultaneously



(a) (b)

Figure 6: Impact of Scale on Plasticity. (a) MT10 (b) MT50. In MT10, we find that a bigger model has significantly more dormant neurons through training than smaller models. However, in MT50 we find that increasing the number of parameters and the number of tasks seems to limit plasticity loss.

may be a more effective strategy than focusing on either dimension alone. Rather than viewing large models as susceptible to plasticity loss, our results indicate that increasing task diversity creates sufficient learning pressure to maintain neuron activation across the network. This challenges the conventional wisdom that larger models necessarily suffer from greater plasticity issues and points toward a potential method for scaling RL models.

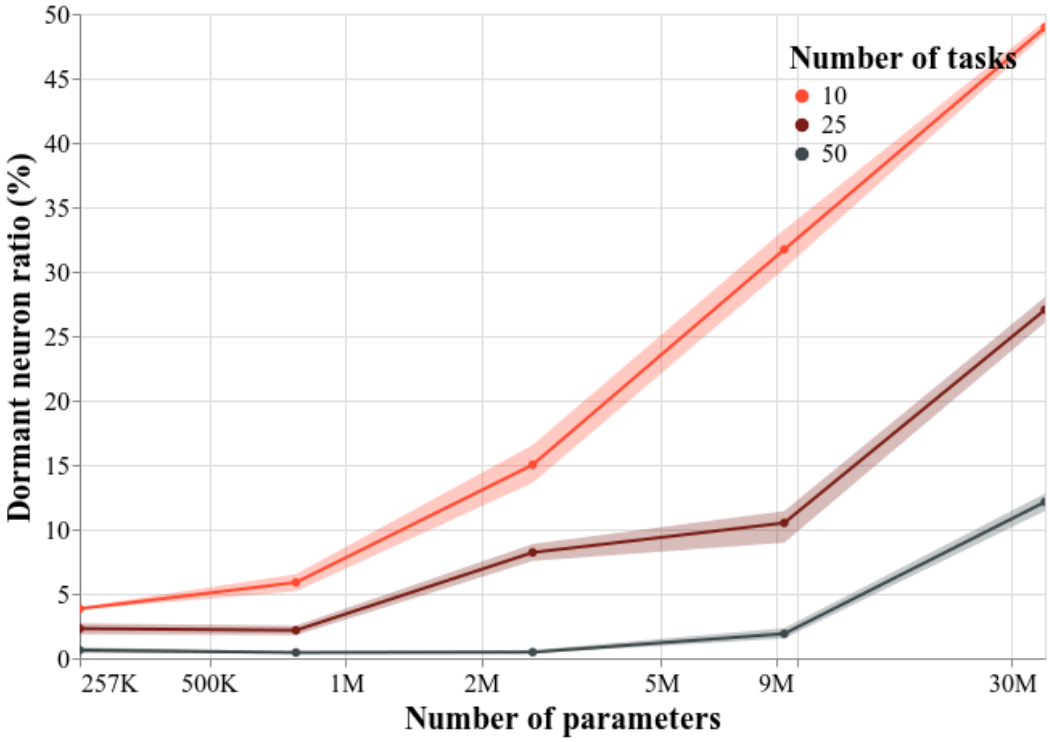


Figure 7: The effect of increasing the number of model parameters, when training on various numbers of tasks, on the percentage of dormant neurons in the MTRL agent. There seems to be some relationship between the number of tasks and the number of parameters, where scaling both leads to an effective usage of model parameters.

6 Limitations

While our analysis demonstrates that MTRL enables effective scaling of reinforcement learning models through multi-task training, several limitations warrant discussion. Our study primarily focuses on Meta-World, which, despite being a valuable MTRL benchmark, has inherent constraints (a) tasks are limited to short time horizons, (b) all tasks provide dense reward signals, (c) tasks operate in isolated environments with no occlusions, (d) MTRL agents receive privileged state information rather than pixel observations, (e) Meta-World tasks are much more similar than other RL tasks (i.e. tasks in the ALE (Bellemare et al., 2013)). Despite these limitations, we contend that our findings provide valuable insights applicable across different task domains and MTRL methods. The fundamental relationship between parameter scaling, task diversity, and performance likely extends beyond Meta-World's specific characteristics.

7 Conclusion

This work proposes that recent advances in multi-task reinforcement learning (MTRL) can be attributed to the number of parameters that recent works have used in their models and not the specialized MTRL architectures that each work proposes. In order to test this hypothesis, we implemented four MTRL architectures and re-benchmarked them across two Meta-World benchmarks. We then took a simple feed-forward baseline architecture and increased the number of parameters to match the number of parameters in the MTRL specific architectures. In doing so, we found that simply scaling the feed-forward neural network architecture outperformed the MTRL specific architectures. Next, we explored the effects of scaling the actor or the critic. We found that scaling the critic is more beneficial than scaling the actor, likely due to the Q-function having to fit multiple state-action functions. We also explored how far we can scale feed-forward neural networks to continue improving performance. While performing these experiments, we also discovered a previously unknown relationship between the number of tasks being trained on and the number of parameters in the model and plasticity: increasing both the number of parameters and the number of tasks being used can be an effective way to limit plasticity loss in RL. Finally, we acknowledged that Meta-World can be a simpler benchmark in comparison to other RL problems available in the literature but we hope that this work can provide a simple framework for future work exploring the scaling of reinforcement learning methods.

Our results likely have implications for single task reinforcement learning. We have found that a simple plasticity loss mitigation strategy may be to increase the number of tasks that the reinforcement learning agent is currently being trained on. This may be because the additional tasks act as auxiliary tasks (Jaderberg et al., 2017). However, in some scenarios it may not be possible to add additional tasks for the agent to train on. In these cases, it may be possible to mitigate plasticity loss by increasing the number of parametric variations that the agent interacts with. In Meta-World, each task has up to fifty variations of object and/or goal location. It may in fact be these variations, in combination with a large number of tasks, that allows for better plasticity loss mitigation. Thus, it may be useful to investigate the introduction of smaller sub-tasks or parametric variations when attempting to scale single task reinforcement learning. Additionally, as we continued to scale our networks, we found that experiments would become more computationally expensive. It is likely a useful line of future work to explore how to leverage a relatively small number of parameters but have the performance of a much larger model when training online.

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=uqv8-U4lKBe.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. What matters for on-policy deep actor-critic methods? a large-scale study. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=nIAxjsniDzg.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: an evaluation platform for general agents. *J. Artif. Int. Res.*, 47(1):253–279, May 2013. ISSN 1076-9757.

Marc G. Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C. Machado, Subhodeep Moitra, Sameera S. Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, December 2020. ISSN 1476-4687. DOI: 10.1038/s41586-020-2939-8. URL https://doi.org/10.1038/s41586-020-2939-8.

Johan Samir Obando Ceron, João Guilherme Madeira Araújo, Aaron Courville, and Pablo Samuel Castro. On the consistency of hyper-parameter selection in value-based deep reinforcement learning. In *Reinforcement Learning Conference*, 2024. URL https://openreview.net/forum?id=szUyvvwoZB.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de las Casas, Craig Donner, Leslie Fritz, Cristian Galperti, Andrea Huber, James Keeling, Maria Tsimpoukelli, Jackie Kay, Antoine Merle, Jean-Marc Moret, Seb Noury, Federico Pesamosca, David Pfau, Olivier Sauter, Cristian Sommariva, Stefano Coda, Basil Duval, Ambrogio Fasoli, Pushmeet Kohli, Koray Kavukcuoglu, Demis Hassabis, and Martin Riedmiller. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, February 2022. ISSN 1476-4687. DOI: 10.1038/s41586-021-04301-9. URL https://www.nature.com/articles/s41586-021-04301-9. Number: 7897 Publisher: Nature Publishing Group.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1etN1rtPB.

Jesse Farebrother, Joshua Greaves, Rishabh Agarwal, Charline Le Lan, Ross Goroshin, Pablo Samuel Castro, and Marc G Bellemare. Proto-value networks: Scaling representation learning with auxiliary tasks. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=oGDKSt9JrZi.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://openreview.net/forum?id=HJjvxl-Cb.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023. URL https://doi.org/10.48550/arXiv.2301.04104.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition . In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Los Alamitos, CA, USA, June 2016. IEEE Computer Society. DOI: 10.1109/CVPR.2016.90. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.90.

Ahmed Hendawy, Jan Peters, and Carlo D'Eramo. Multi-task reinforcement learning with mixture of orthogonal experts. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=aZH1dM3GOX.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJ6yPD5xg.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020. URL https://api.semanticscholar.org/CorpusID:210861095.

Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and M. Pawan Kumar. In defense of the unitary scalarization for deep multi-task learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=wmwgLEPjL9.

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23190–23211. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/lyle23b.html.

Michal Nauman, Michał Bortkiewicz, Piotr Miłoś, Tomasz Trzciński, Mateusz Ostaszewski, and Marek Cygan. Overestimation, overfitting, and plasticity in actor-critic: the bitter lesson of reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=fu0xdh4aEJ.

Evgenii Nikishin, Junhyuk Oh, Georg Ostrovski, Clare Lyle, Razvan Pascanu, Will Dabney, and Andre Barreto. Deep reinforcement learning with plasticity injection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=jucDLW6G9l.

Johan Samir Obando Ceron, Ghada Sokar, Timon Willi, Clare Lyle, Jesse Farebrother, Jakob Nicolaus Foerster, Gintare Karolina Dziugaite, Doina Precup, and Pablo Samuel Castro. Mixtures of experts unlock parameter scaling for deep RL. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38520–38540. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/obando-ceron24b.html.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Max Schwarzer, Johan Obando-Ceron, Aaron Courville, Marc G. Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: human-level atari with human-level efficiency. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/sokar23a.html.

Lingfeng Sun, Haichao Zhang, Wei Xu, and Masayoshi Tomizuka. Paco: Parameter-compositional multi-task reinforcement learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=LYXTPNWJLr.

Adrien Ali Taiga, Rishabh Agarwal, Jesse Farebrother, Aaron Courville, and Marc G Bellemare. Investigating multi-task pretraining and generalization in reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sSt9fROSZRO.

Shengjie Wang, Shaohuai Liu, Weirui Ye, Jiacheng You, and Yang Gao. Efficientzero v2: Mastering discrete and continuous control with limited data. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=LHGMXcr6zx.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-task reinforcement learning with soft modularization. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/32cfdce9631d8c7906e8e9d6e68b514b-Abstract.html.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020a. Curran Associates Inc. ISBN 9781713829546.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 30 Oct–01 Nov 2020b. URL https://proceedings.mlr.press/v100/yu20a.html.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, pp. 1204–1213, 2022. URL https://doi.org/10.1109/CVPR52688.2022.01179.

Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for atari. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1xCPJHtDB.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Additional MTRL Architectures

Soft Modules:

1. Smaller replay buffers, we use $100k * N$ where N is the number of tasks
2. They apply an additional dynamic weighting scheme for summing the per-task losses, we utilize a uniform weighting of task losses

MOORE:

1. We use a minimum logarithm standard deviation output of e^{-20} instead of e^{-10}
2. The main text of Hendawy et al. (2024) suggests that they use 20 million timesteps for MT50, however the appendix suggests that they report results over 100 million timesteps. We explicitly use 100 million timesteps.
3. We evaluate over fifty episodes, while Hendawy et al. (2024) reports performance over 10 evaluation episodes.

PaCo:

1. Their results leverage a weight reset and a loss maskout in order to stabilize training. We do not leverage those tricks

B Depth 4

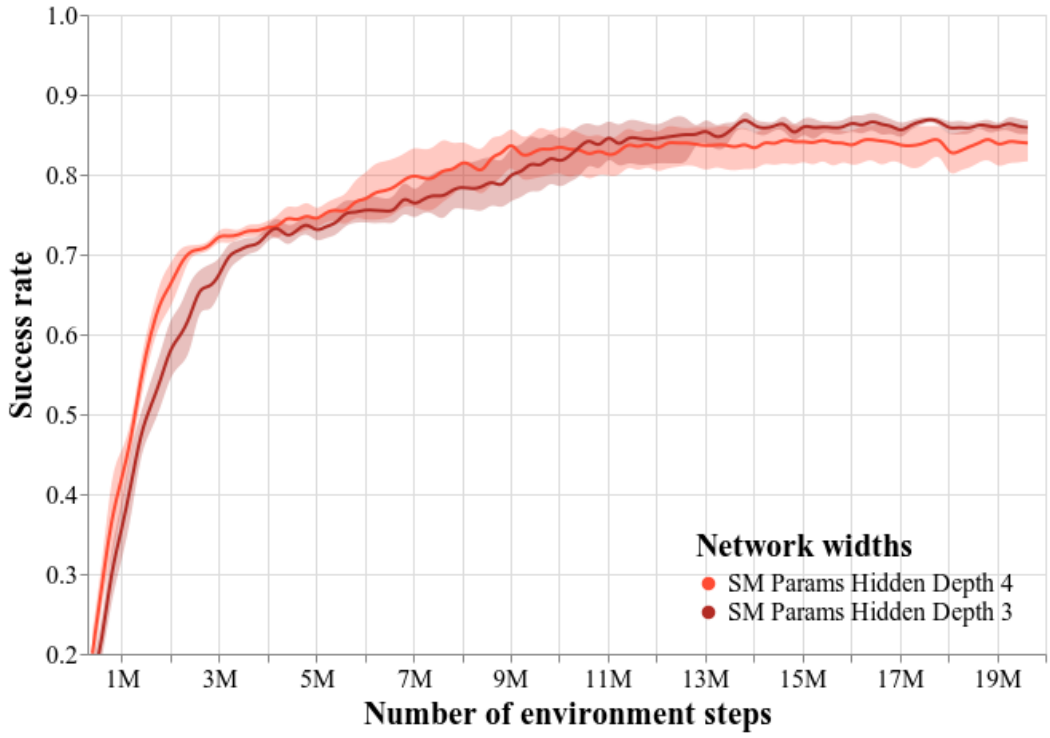


Figure 8: Effect of depth with SM number of parameters on MT10

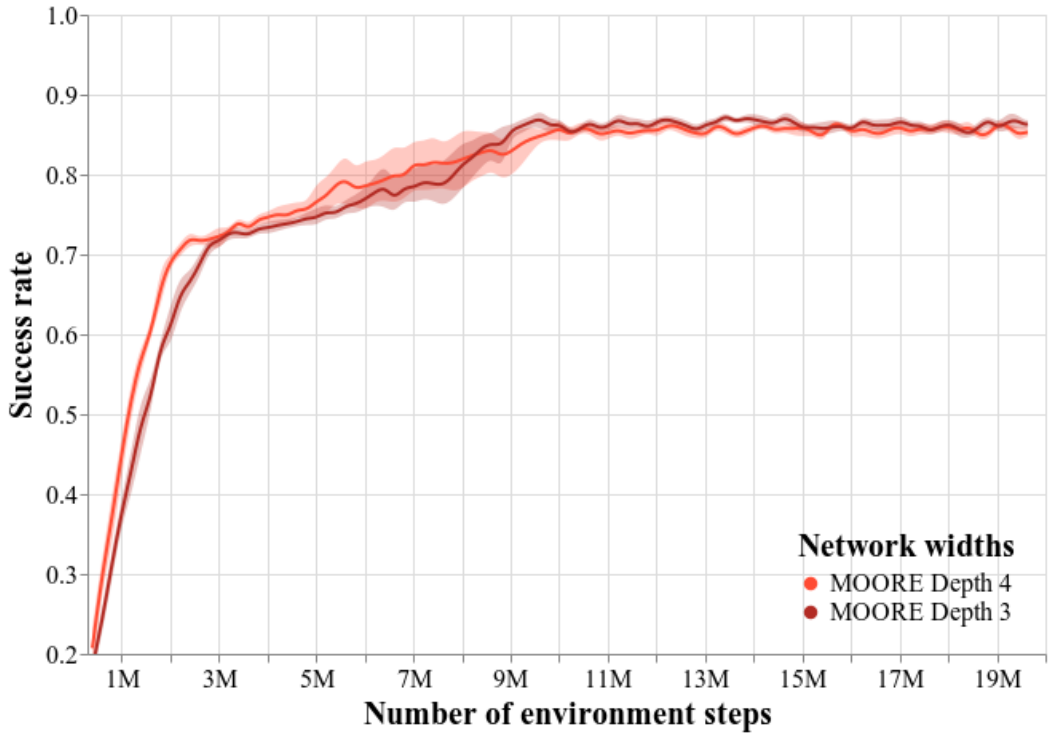


Figure 9: Effect of depth with MOORE number of parameters on MT10

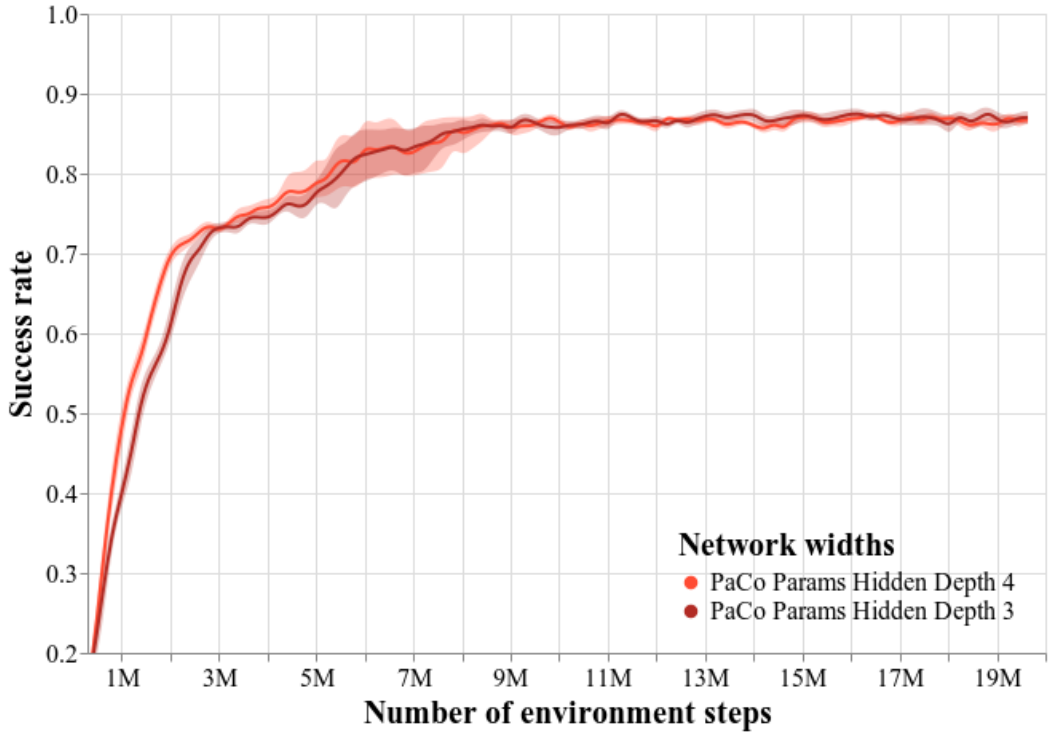


Figure 10: Effect of depth with PaCo number of parameters on MT10

C Plasticity Metrics

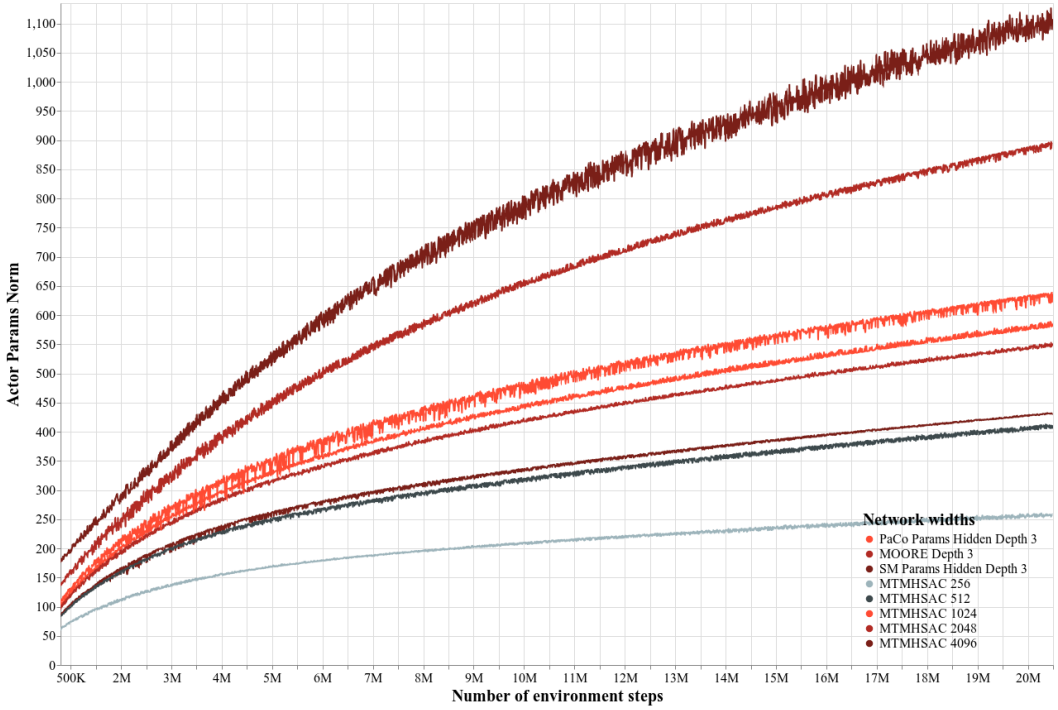


Figure 11: Actor Parameter Norms on MT10

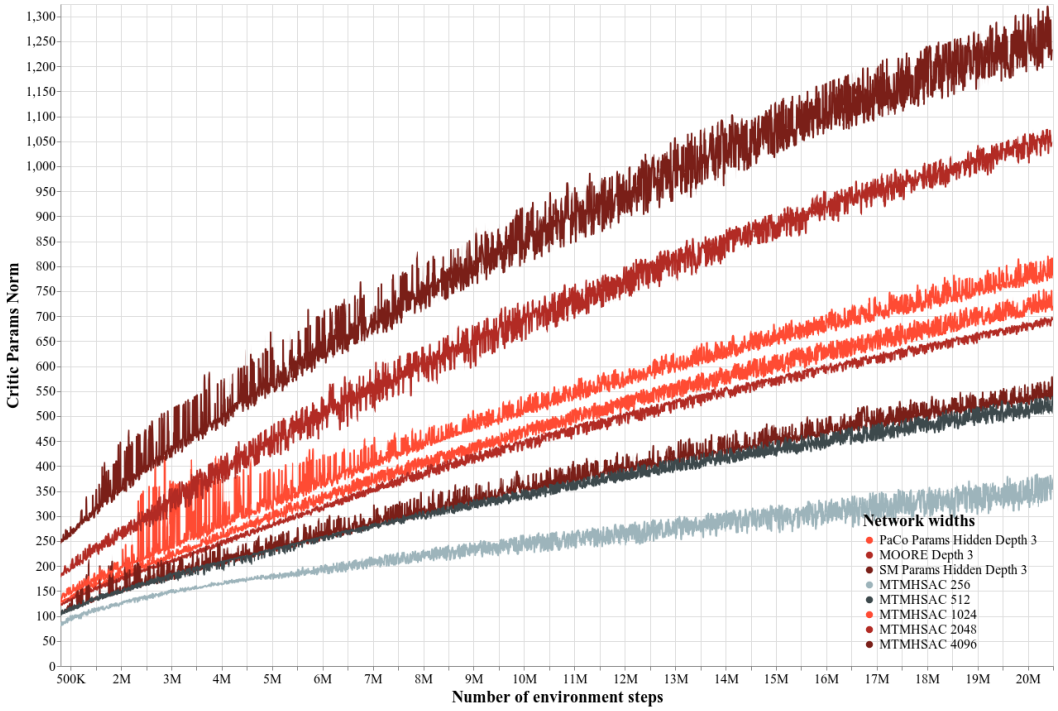


Figure 12: Critic Parameter Norms on MT10

D Baseline Architecture Results

In this section we report the results of our implemented MTRL architectures in relation to their publications. Results indicate that, even across various evaluation methods, that our implementations are correct.

Architecture	MT10 (Ours)	MT10 (Pub)	MT50 (Ours)	MT50 (Pub)
Soft Modularization (Yang et al., 2020)	71.4	71.8	NA	NA
PaCo (Sun et al., 2022)	73.6	71.6	58.9	57.3
MOORE (Hendawy et al., 2024)	83.2	88.7	72.0	72.9

Table 1: Replication of selected MTRL results. For the Soft Modularization (SM) results, we note that SM reports results on an earlier version of Meta-World not currently available. For our reproduction we added the V1 rewards to the MT10 environments. For our PaCo MT10 results we compare to their "PaCo-Vanilla" results from Table 2 in Sun et al. (2022). This is the version of their architecture that does not include any loss masking or resets.

E Scaled MTRL Architecture Results

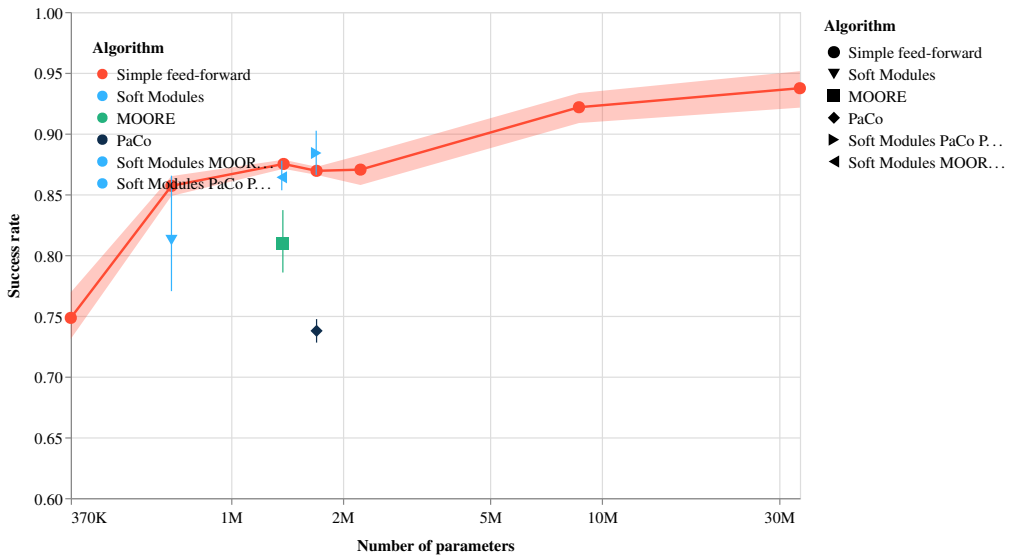


Figure 13: Additional Scaling of MTRL Architectures, MT10

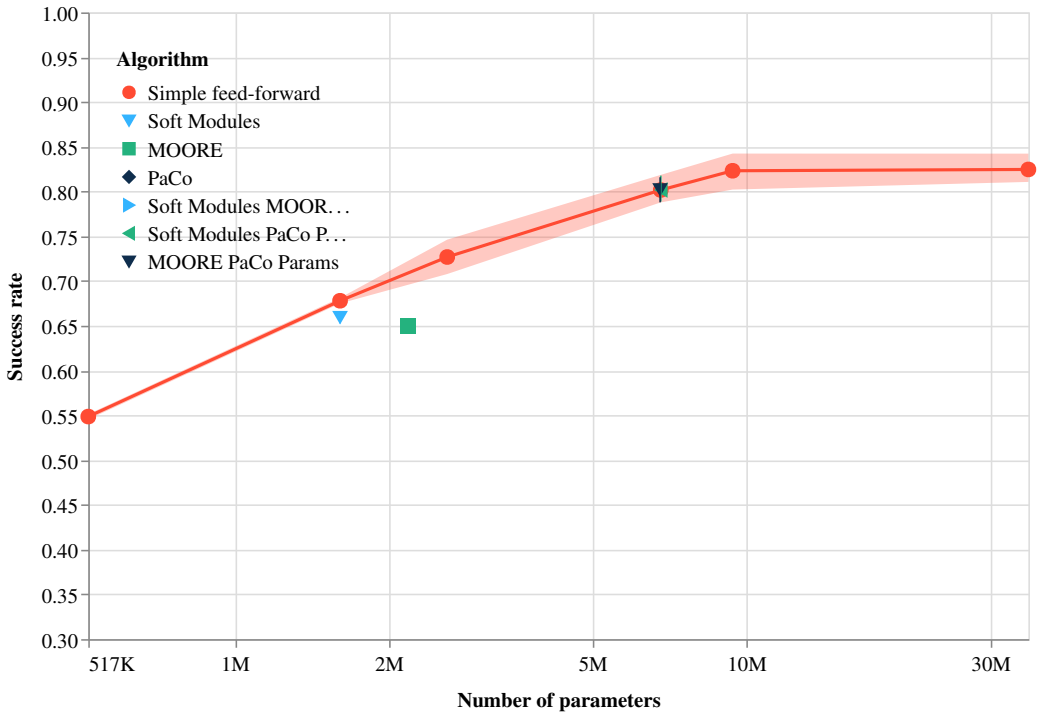


Figure 14: Additional Scaling of MTRL Architectures, MT50