In Support of Over-Parametrization in Deep Reinforcement Learning: an Empirical Study

Brady Neal [1] **Ioannis Mitliagkas** [1 2]

Abstract

There is significant recent evidence in supervised learning that, in the over-parametrized setting, wider networks achieve better test error. In other words, the bias-variance tradeoff is not directly observable when increasing network width arbitrarily. We investigate whether a corresponding phenomenon is present in reinforcement learning. We experiment on four OpenAI Gym environments, increasing the width of the policy and value networks beyond their prescribed values. Our empirical results lend support to this hypothesis. However, tuning the hyperparameters of each network width separately remains as important future work in environments/algorithms where the optimal hyperparameters vary noticably across widths, confounding the results when the same hyperparameters are used for all widths.

1. Introduction

A longstanding notion in supervised learning is that, as model complexity increases, test error decreases initially and, eventually, increases again. Intuitively, the idea is that as the size of your hypothesis class grows, the closer you can approximate the ground-truth function with some function in your hypothesis class. At the same time, the larger amount of functions to choose from in your hypothesis class leads to higher estimation error (overfitting) from fitting the finite data sample too closely. This is the essential bias-variance tradeoff in supervised learning. We discuss these tradeoffs in more depth in Section 2.2.

However, Neyshabur et al. (2015) found that increasing the width of a single hidden layer neural network leads to decreasing test error on MNIST and CIFAR-10. Since then, there has been a large amount of evidence that wider networks generalize better in a variety of different architectures and hyperparameter settings (Zagoruyko & Komodakis, 2016; Novak et al., 2018; Lee et al., 2018; Neal et al., 2018; Belkin et al., 2018; Spigler et al., 2018; Liang et al., 2017), once in the over-parametrized setting (Spigler et al., 2018; Belkin et al., 2018). In other words, the bias-variance tradeoff is not observed in this over-parametrized setting, as network width grows (Neal et al., 2018).

How far can we inductively infer from this? Is this phenomenon also present in deep reinforcement learning or do we eventually see a degradation in performance as we increase network width? In this paper, we present preliminary evidence that this phenomenon is also present in reinforcement learning. For example, using default hyperparameters, we can already see performance increase well past the default width that is commonly used (64) in Fig. 1. We test the hypothesis that wider networks (both policy and value) perform monotonically better than their smaller counterparts in policy gradients methods. Of course, we will hit diminishing returns as the network width gets very large, but this is very different from the competing hypothesis that larger networks will overfit more.

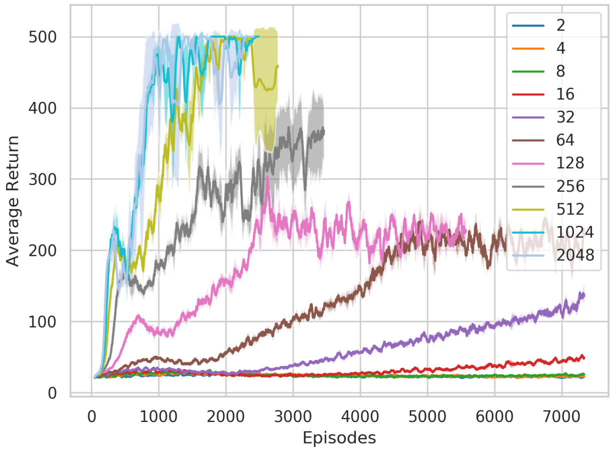

Figure 1: PPO with network widths up to 2048 on the Cart-Pole task, using the default Stable Baselines hyperparameters. Wider networks perform better.

[1] Mila, Université de Montréal [2] Canada CIFAR AI Chair. Correspondence to: Brady Neal <bradyneal11@gmail.com>.

ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena, Long Beach, California. Copyright 2019 by the author(s).

2. Preliminaries

2.1. Supervised Learning Setting

We are given a training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ of m training examples, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. Furthermore, $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, so $S \in \mathcal{Z}^m$. $\mathcal{D}$ denotes a distribution over $\mathcal{Z}$, so we have $(x_i, y_i) \sim \mathcal{D}$ and $S \sim \mathcal{D}^m$. We use lowercase x and y to denote random variables because of convention in this field.

We learn a hypothesis $h \in \mathcal{H}$ via a learning algorithm $\mathcal{A} : \mathcal{Z}^m \to \mathcal{H}$. We denote a hypothesis learned from training set S as $h_S = \mathcal{A}(S)$. Given a loss function, $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, the goal is to minimize the *expected risk*:

$$R(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\, \ell(h(x), y) \tag{1}$$

2.2. Tradeoffs in Model Complexity

We present a discussion on tradeoffs in model complexity because it does not appear to be much of a focus in the reinforcement learning community. A common way of thinking about the generalization performance of a learner is through the lens of a tradeoff. For example, when h_S is chosen from a hypothesis class $\mathcal{H}$, $R(h_S)$ can be decomposed into approximation error and estimation error

$$R(h_S) = \mathcal{E}_{\text{app}} + \mathcal{E}_{\text{est}}$$

where $\mathcal{E}_{\text{app}} = \min_{h\in\mathcal{H}} R(h)$ and $\mathcal{E}_{\text{est}} = R(h_S) - \mathcal{E}_{\text{app}}$. Shalev-Shwartz & Ben-David (2014, Section 5.2) present this decomposition and frame it as a tradeoff. Bottou & Bousquet (2008) describe this as the "well known tradeoff between approximation error and estimation error" and present it in a slightly more lucid way as a decomposition of the *excess risk*:

$$\begin{aligned}\mathbb{E}[R(h_S) - &R(h^*)] \\ &= \mathbb{E}[R(h^*_{\mathcal{H}}) - R(h^*)] + \mathbb{E}[R(h_S) - R(h^*_{\mathcal{H}})]\end{aligned}$$

where $R(h^*)$ is the Bayes error and $h^*_{\mathcal{H}} = \arg\min_{h\in\mathcal{H}} R(h)$ is the best hypothesis in $\mathcal{H}$. The approximation error can then be interpreted as the distance of the best hypothesis in $\mathcal{H}$ from the Bayes classifier, and the estimation error can be interpreted as the average distance of the learned hypothesis from the best hypothesis in $\mathcal{H}$. It is common to associate larger $\mathcal{H}$ with smaller approximation error and larger estimation error.

The commonly cited universal approximation property of neural networks (Cybenko, 1989; Hornik, 1991; Leshno & Schocken, 1993) means that the approximation error goes to 0 as the network width increases; these results do not say anything about estimation error.

A similar tradeoff in model complexity is known as the bias-variance tradeoff (Geman et al., 1992). Bias is analogous to the approximation error while variance is analogous to the estimation error. This tradeoff is probably even more pervasive (Bishop (2006, Chapter 3.2), Geman et al. (1992), Hastie et al. (2001, Chapter 2.9), Goodfellow et al. (2016, Chapter 5.4.4)). It is common to view the problem of designing a good learning algorithm as choosing a good $\mathcal{H}$ that optimizes this tradeoff.

2.3. Comparison of Supervised Learning with Reinforcement Learning

Statistical learning theory for supervised learning is given in the i.i.d. setting. That is, examples are independent and identically distributed. This also means the training distribution is the same as the test distribution. In reinforcement learning, training examples are not independent because examples within the same episode depend on each other through the current behavior policy and through the environment's transition dynamics. Training examples are not identically distributed because the policy produces training examples, and the policy changes over time. For the same reason, the training distribution and the test distribution are not completely the same. These differences make it non-obvious that the phenomenon seen in supervised learning would extend to reinforcement learning.

3. Experiments

3.1. Experimental Details

We run experiments, with a variety of combinations of environments and learning algorithms, where we vary the width of the shared policy and value network. We use four different environments from OpenAI Gym (Brockman et al., 2016): CartPole, Acrobot, MountainCar, and Pendulum. We use four different learning algorithms: PPO (Schulman et al., 2017), A2C (Mnih et al., 2016), ACER (Wang et al., 2017), and ACKTR (Wu et al., 2017). We make use of the existing implementations of these algorithms in the Stable Baselines library (Hill et al., 2018), an improved fork of OpenAI Baselines (Dhariwal et al., 2017). We were only able to train ACKTR up to width 512 because it is an approximate second-order method. Experiments with ACKTR are in Appendix B.

We get hyperparameters that were tuned on networks of width 64 from the RL Baselines Zoo that was built on top of Stable Baselines. One hyperparameter is how many time steps the learners are trained for. It is different for different environment/learner pairs, but always on the order of 1 million. It is always the same across widths within an environment/learner pair. In some of the plots, learners see fewer episodes because their episodes are, on average, longer.

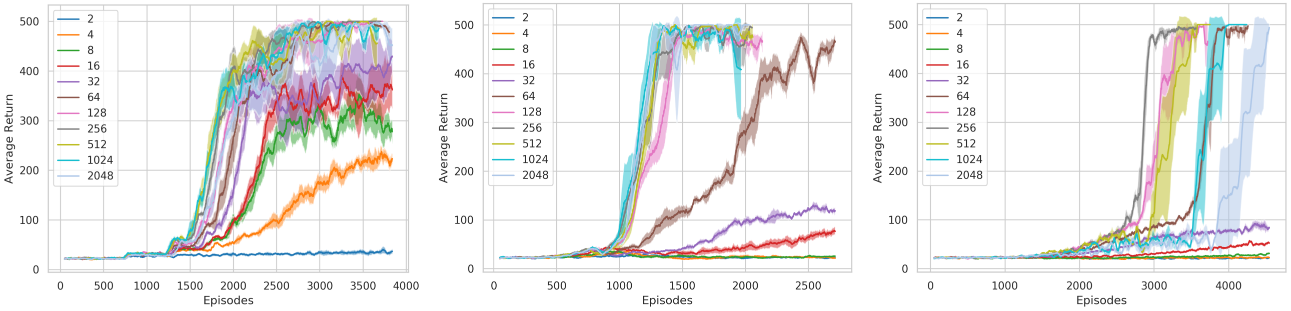

Figure 2: PPO (left), ACER (center), and A2C (right) experiments on CartPole

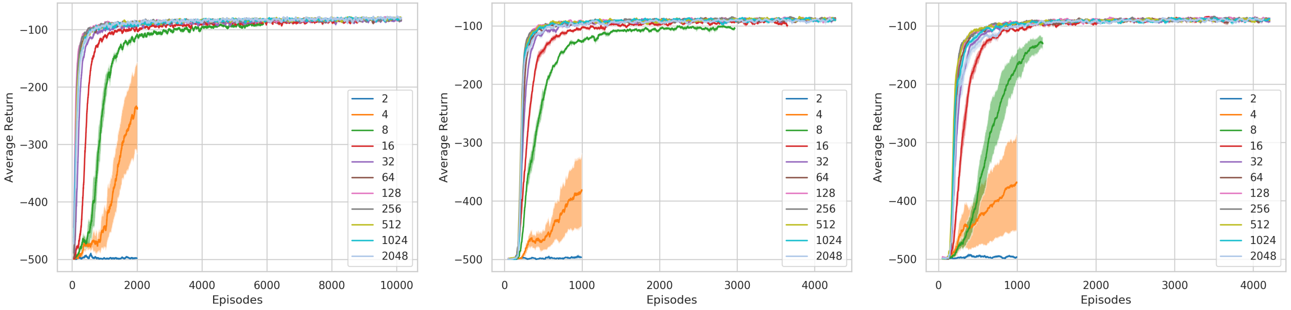

Figure 3: PPO (left), ACER (center), and A2C (right) experiments on Acrobot

We choose relatively simple tasks for these experiments partially because they are faster to train on, but more importantly, we choose them because their simplicity lends itself to more ease of overfitting. In other words, on these tasks, we will see diminishing returns with much smaller networks, so we can test the "very wide networks will not see degraded performance" hypothesis with a much smaller range of networks.

We run each experiment with 5 different random seeds. The policy and value networks are shared. The architecture consists of 2 hidden fully connected layers followed by separate linear transformations: one to yield the policy and one to yield the value. We use 2 hidden layers, rather than just 1, because 2 hidden layers are more common in reinforcement learning.

3.2. CartPole, Acrobot, and Pendulum Environments

In CartPole (Fig. 2), we see a lot of evidence for the hypothesis. In the both the PPO and A2C experiments, peak performance is reached by width 64, and that level of performance is maintained through width 2048. In the ACER experiment, near peak performance is reached by width 128, and through width 2048, we see peak performance.

Similarly, in Acrobot (Fig. 3), we see even more evidence for the hypothesis. We see peak performance as early as width 16 in PPO, ACER, and A2C. This means that Acrobot is simple enough to only require a network of width 16 (compared to 64 for CartPole). Still, we see peak performance through width 2048 in all 3 learners.

In Pendulum (Appendix A), we see more evidence for the hypothesis. The default width (64) network, performs distinctly worse than the wider networks. We do not see any degradation of performance through width 2048. We only run PPO with the Pendulum environment because RL Baselines Zoo did not have tuned hyperparameters for the other algorithms.

3.3. MountainCar Environment

In the MountainCar environment, we see the first hint of what looks like could be evidence against the hypothesis (Fig. 4). PPO (left) performance begins to degrade at width 2048, ACER (center) performance begins to degrade at width 512, and we see a sharp drop in performance from width 1024 to width 2048 in A2C (right).

RL algorithms are known to be highly sensitive to hyperparameter settings (Henderson et al., 2018a; Islam et al., 2017), especially learning rate (Henderson et al., 2018b). We believe this performance degradation is due to more variability across widths of the optimal hyperparameters on MountainCar (compared CartPole, Acrobot, and Pendulum).

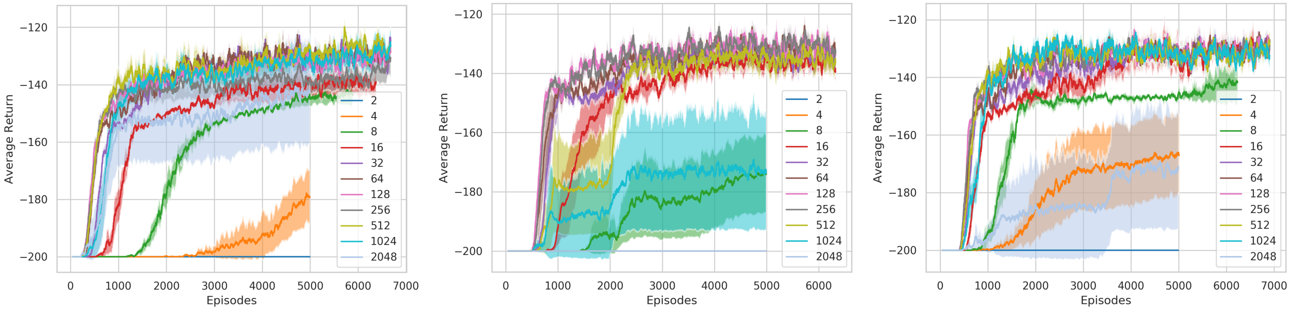

Figure 4: PPO (left), ACER (center), and A2C (right) experiments on MountainCar

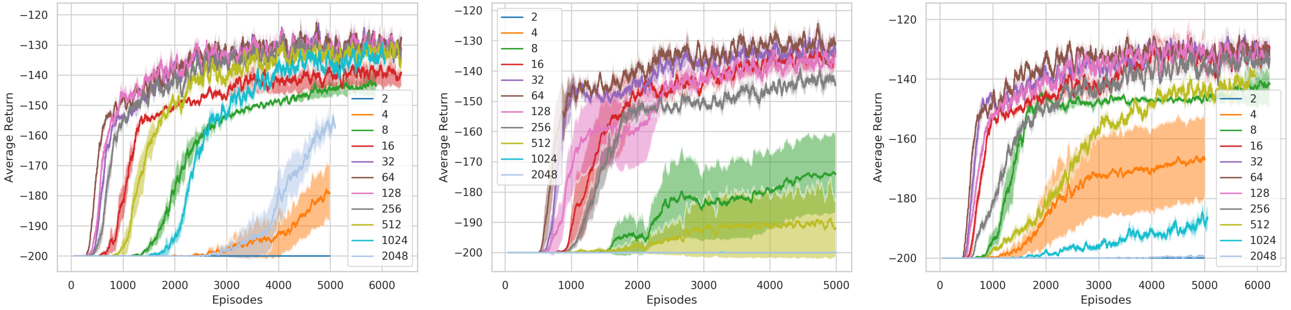

Figure 5: PPO (left), ACER (center), and A2C (right) experiments on MountainCar with learning rate scaling of h^{-1}

3.4. Automatic Learning Rate Scaling

In order to fairly compare all the widths, we would like the hyperparameters for each of them to be optimal. (Geiger et al., 2019) study test error when scaling network width in supervised learning, and they scale the learning rate as $h^{-1.5}$, where h is the network width. This scaling is motivated by making the number of steps to convergence independent of width, but it does not necessarily make the learning rate for each network optimal. Because learning rate is such an important and sensitive hyperparameter in reinforcement learning (Henderson et al., 2018b), we try scaling the learning rate α with both of the following schemes: $\alpha \leftarrow \min(\alpha^*_{64}, (h/64)^{-1})$ and $\alpha \leftarrow \min(\alpha^*_{64}, (h/64)^{-1.5})$, where α^*_{64} is the learning rate that was tuned to network width 64 (pulled from RL Baselines Zoo). We see that scaling the learning rate as h^{-1} (Fig. 5) and $h^{-1.5}$ (Appendix C, Fig. 8) actually make the largest networks perform worse, indicating that this scaling is not useful for comparing networks with optimal hyperparameters. We present these scalings on MountainCar because it was the environment that did not look like the others, but the scalings on CartPole and Acrobot are in Appendix D.

4. Conclusion and Future Work

The phenomenon in supervised learning that motivated this work is that, in the over-parametrized setting, increasing network width leads to monotonically lower test error (no U curve). We find a fair amount of evidence of this phenomenon extending to reinforcement learning in our preliminary experiments (namely CartPole, Acrobot, and Pendulum).

However, we also saw that performance did consistently degrade in the MountainCar experiments. We believe this to be because that environment is more sensitive to hyperparameters; since the hyperparameters were chosen using width 64 and then used for all of the other widths, the hyperparameters are likely not optimal for the other widths like they are for width 64. The MountainCar environment exaggerates this lack suboptimality more than the other 3 environments.

The main next experiments we plan to run will use an automated tuning procedure that chooses the hyperparameters for each width individually. We believe this protocol will yield MountainCar results that look much more like the CartPole and Acrobot results. We then plan to replicate these findings across more learning algorithms and more environments.

References

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine learning and the bias-variance trade-off. *arXiv e-prints*, art. arXiv:1812.11118, December 2018.

Bishop, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

Bottou, L. and Bousquet, O. The tradeoffs of large scale learning. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *Advances in Neural Information Processing Systems 20*, pp. 161–168. Curran Associates, Inc., 2008.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec 1989.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

Geiger, M., Jacot, A., Spigler, S., Gabriel, F., Sagun, L., d'Ascoli, S., Biroli, G., Hongler, C., and Wyart, M. Scaling description of generalization with number of parameters in deep learning. *CoRR*, abs/1901.01608, 2019.

Geman, S., Bienenstock, E., and Doursat, R. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. Deep reinforcement learning that matters. In *AAAI*, 2018a.

Henderson, P., Romoff, J., and Pineau, J. Where did my optimum go?: An empirical analysis of gradient descent optimization in policy gradient methods. *CoRR*, abs/1810.02525, 2018b.

Hill, A., Raffin, A., Ernestus, M., Gleave, A., Traore, R., Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., and Wu, Y. Stable baselines. https://github.com/hill-a/stable-baselines, 2018.

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251 – 257, 1991.

Islam, R., Henderson, P., Gomrokchi, M., and Precup, D. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *CoRR*, abs/1708.04133, 2017.

Lee, J., Sohl-dickstein, J., Pennington, J., Novak, R., Schoenholz, S., and Bahri, Y. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Leshno, M. and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6:861–867, 1993.

Liang, T., Poggio, T. A., Rakhlin, A., and Stokes, J. Fisher-rao metric, geometry, and complexity of neural networks. *CoRR*, abs/1711.01530, 2017.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Balcan, M. F. and Weinberger, K. Q. (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR.

Neal, B., Mittal, S., Baratin, A., Tantia, V., Scicluna, M., Lacoste-Julien, S., and Mitliagkas, I. A modern take on the bias-variance tradeoff in neural networks, 2018.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *International Conference on Learning Representations workshop track*, 2015.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Spigler, S., Geiger, M., d'Ascoli, S., Sagun, L., Biroli, G., and Wyart, M. A jamming transition from under- to over-parametrization affects loss landscape and generalization. *CoRR*, abs/1810.09665, 2018.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. Sample efficient actor-critic with experience replay. *ICLR 2017*, 2017.

Wu, Y., Mansimov, E., Liao, S., Grosse, R. B., and Ba, J. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *CoRR*, abs/1708.05144, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In Richard C. Wilson, E. R. H. and Smith, W. A. P. (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 87.1–87.12. BMVA Press, September 2016.

Appendices

A. Pendulum

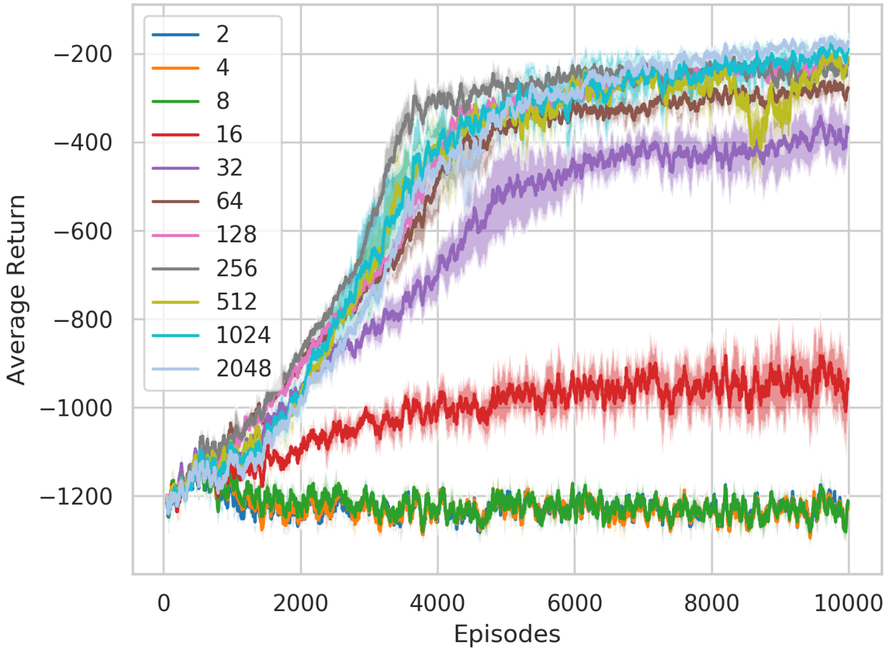

Figure 6: PPO experiment on Pendulum

B. Experiments with ACKTR

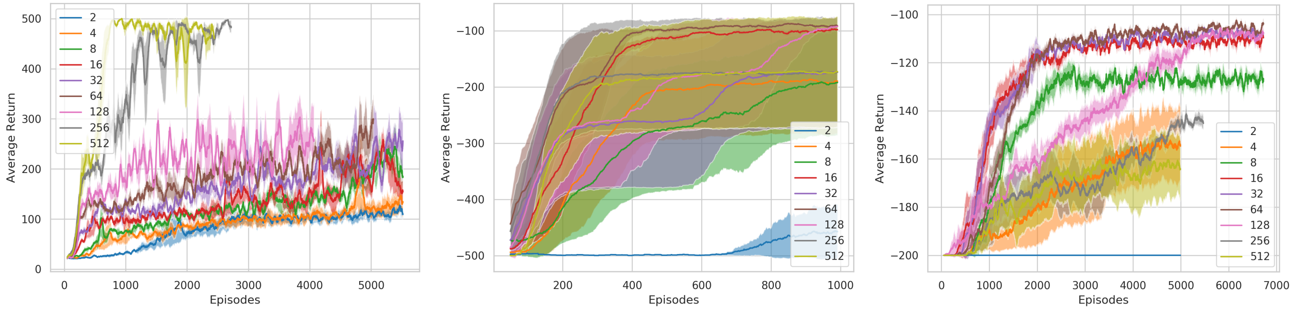

Figure 7: ACKTR experiments on CartPole (left), Acrobot (center), and MountainCar (right)

C. Learning Rate Scaling on MountainCar

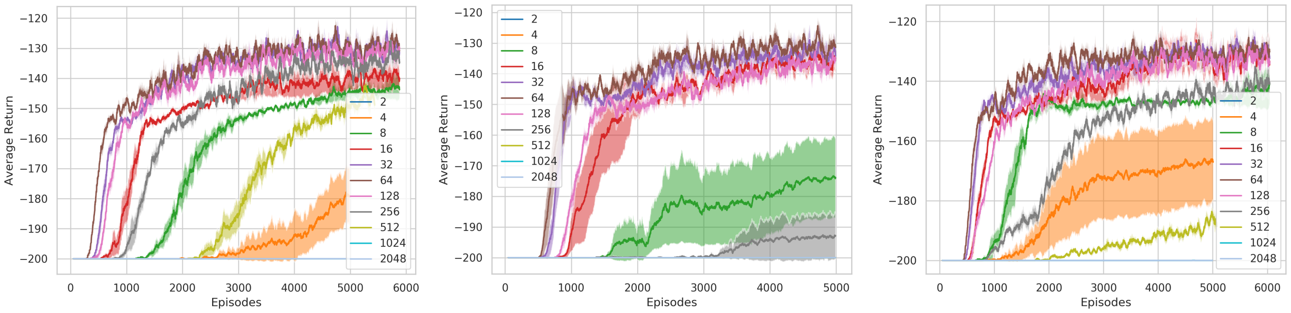

Figure 8: PPO (left), ACER (center), and A2C (right) experiments on MountainCar with learning rate scaling of $h^{-1.5}$

D. Learning Rate Scaling Experiments on CartPole and Acrobot

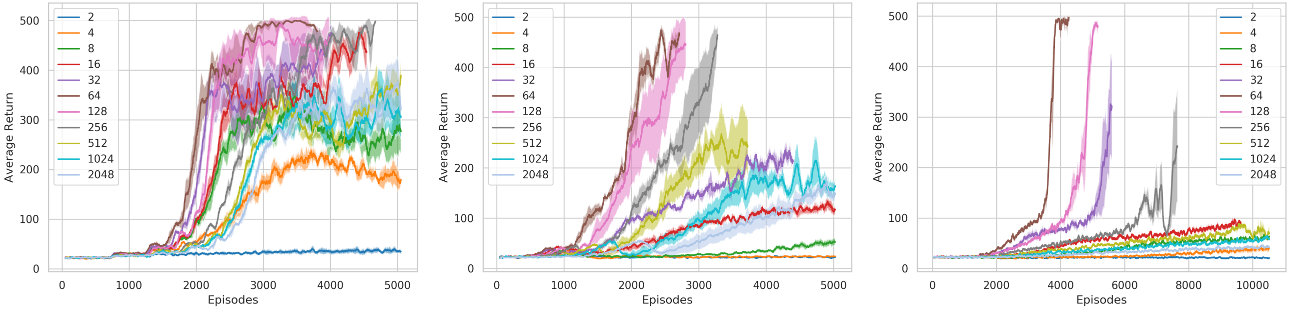

Figure 9: PPO (left), ACER (center), and A2C (right) experiments on CartPole with learning rate scaling of h^{-1}

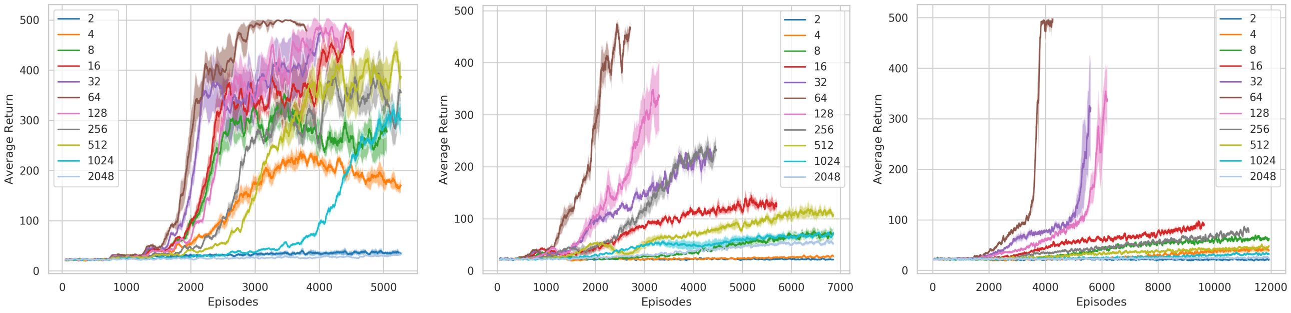

Figure 10: PPO (left), ACER (center), and A2C (right) experiments on CartPole with learning rate scaling of $h^{-1.5}$

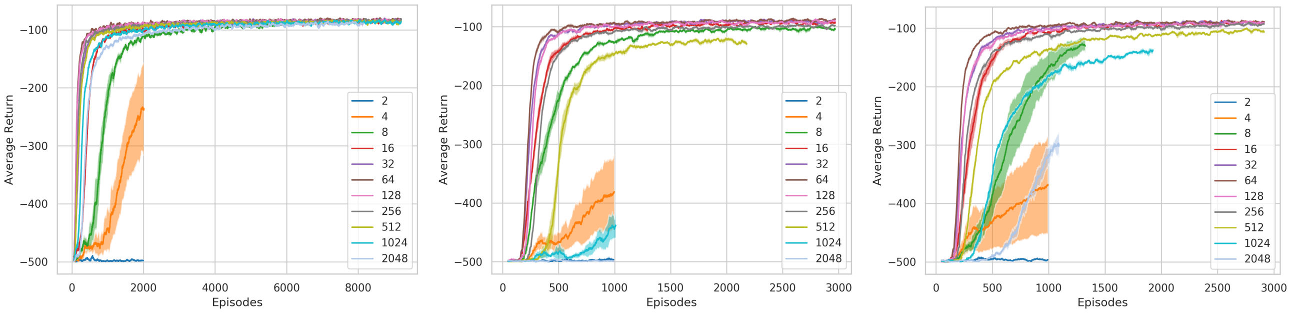

Figure 11: PPO (left), ACER (center), and A2C (right) experiments on Acrobot with learning rate scaling of h^{-1}

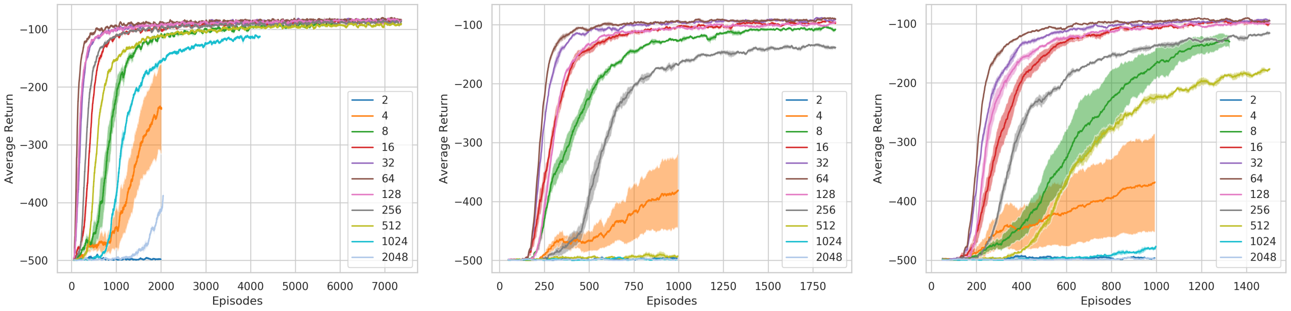

Figure 12: PPO (left), ACER (center), and A2C (right) experiments on Acrobot with learning rate scaling of $h^{-1.5}$