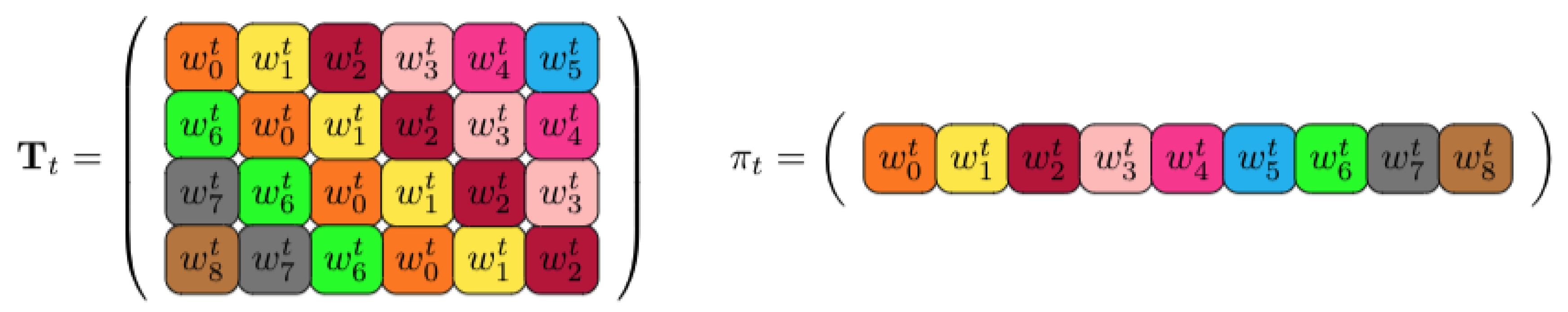
Reinforcement Learning with Chromatic Networks for Compact Architecture Search

Xingyou Song†, Krzysztof Choromanski†, Jack Parker-Holder‡, Yunhao Tang‡
Wenbo Gao‡, Aldo Pacchiano*, Tamas Sarlos†, Deepali Jain†§, Yuxiang Yang†§
Google Research†, Columbia University‡, UC Berkeley*

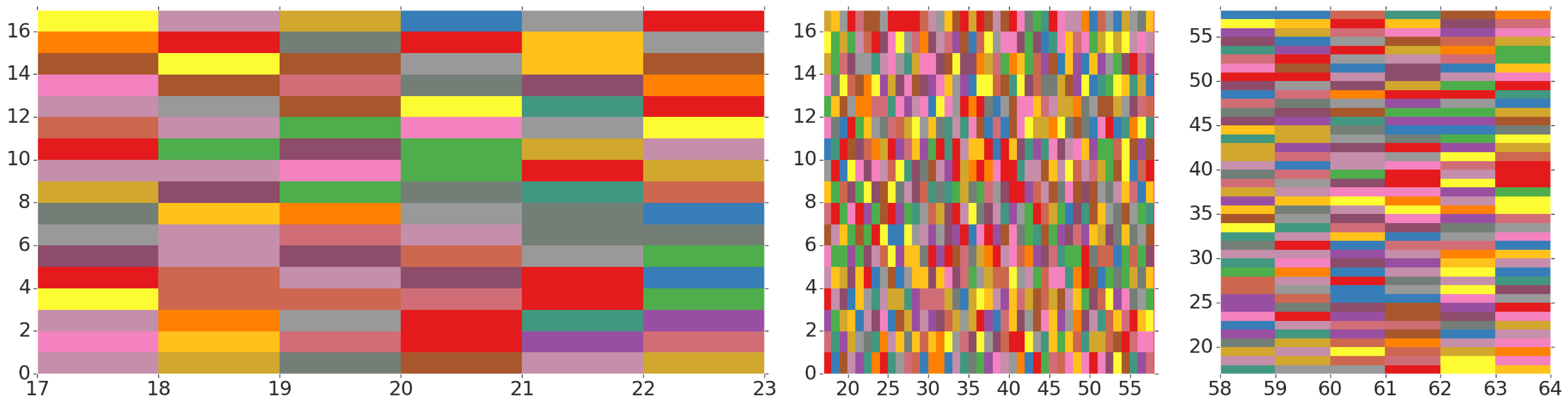
Chromatic Networks for Reinforcement Learning

Compact, efficient architectures are very important for practical use on real robots. [1] has shown that **Toeplitz** Policies, which use diagonally similar weights, are just as effective as standard policies. However, can we do even better through weight sharing? Yes!



We optimize over the space of **weight sharings** (partitions) over neural network weights. Our emphasis is on **fast inference time.** For a matrix of shape (m,n), our method allows matrix-vector multiplication in $\mathcal{O}\left(\frac{mn}{\log(\max(m,n))}\right)$ time.

(Above): Example of Toeplitz policy and diagonal weight sharing pattern.



(Left): A partitioning for a Linear Policy found using our method. **(Right):** A partitioning for a 1-Layer MLP Policy found using our method.

ENAS with ES

Our key observation is that Efficient Neural Architecture Search (ENAS) [2] combines with Evolutionary Strategies (ES) [3, 4] very naturally. We can use the standard Pointer Network Architecture to define a distribution π_θ on the set of all partitionings (colorings) of weights. If $f(P, \mathcal{W})$ defines the rollout reward using neural network policy with weights $\mathcal{W}$ and partitioned with P, then ENAS performs alternating optimization on $F(\theta, \mathcal{W}_{shared}) = \mathbb{E}_{P\sim\pi_\theta}[f(P, \mathcal{W}_{shared})]$

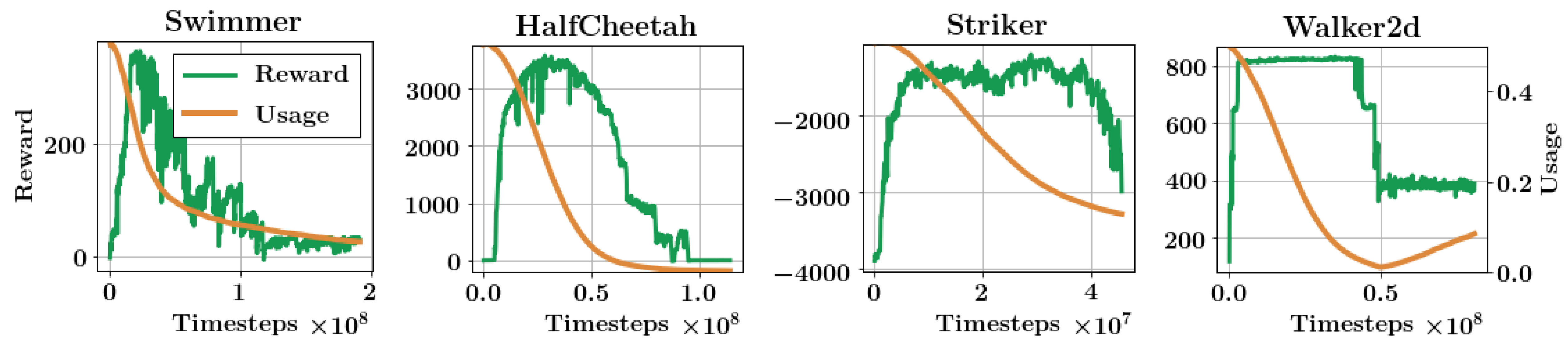
Instead of using backpropagation (which is not possible in Reinforcement Learning) when optimizing over $\mathcal{W}_{shared}$, we use **Evolutionary Strategies (ES)** which estimates the gradient of the Gaussian smoothed objective $F^\sigma(\theta, \mathcal{W}_{\text{shared}}) = \mathbb{E}_{\mathbf{g}\in\mathcal{N}(0,\mathbf{I}_M)}[F(\theta, \mathcal{W}_{\text{shared}} + \sigma\mathbf{g})]$ for a fixed smoothing parameter $\sigma > 0$. We approximate its gradient given by: $\nabla_{\mathcal{W}_{\text{shared}}} F^\sigma(\theta, \mathcal{W}_{\text{shared}}) = \frac{1}{\sigma}\mathbb{E}_{\mathbf{g}\in\mathcal{N}(0,\mathbf{I}_M)}[F(\theta, \mathcal{W}_{\text{shared}} + \sigma\mathbf{g})\mathbf{g}]$ with the following *forward finite difference* unbiased estimator:

$$\widehat{\nabla}_{\mathcal{W}_{\text{shared}}} F^\sigma(\theta, \mathcal{W}_{\text{shared}}) = \frac{1}{t}\sum_{i=1}^{t} \mathbf{g}_t \left[\frac{f(\theta, \mathcal{W}_{\text{shared}} + \sigma\mathbf{g}_t) - F(\theta, \mathcal{W}_{\text{shared}})}{\sigma}\right] \quad (1)$$

where $\mathbf{g}_1, \ldots, \mathbf{g}_t$ are sampled independently at random from $\mathcal{N}(0, \mathbf{I}_M)$. The number of workers t can be scaled **highly** (1000+), as we **only need to use CPU's.**

Compactification Results

Simply masking neural network weights does not work very well:



However, we find Chromatic networks are able to provide $> 90\%$ compression in some cases:

Environment	Dimensions	Architecture	Partitions	Mean Reward	Max Reward
Swimmer	(8,2)	L	8	97	365
Reacher	(11,2)	L	11	-144	-6
Hopper	(11,3)	L	11	216	999
Hopper	(11,3)	H41	11	247	3408
HalfCheetah	(17,6)	L	17	1812	3653
HalfCheetah	(17,6)	L	50	1383	4318
HalfCheetah	(17,6)	H41	17	2148	3779
HalfCheetah	(17,6)	H41, H41	17	3036	5285
Walker2d	(17,6)	H41	17	1943	3695
Pusher	(23,7)	H41	23	-419	-144
Striker	(23,7)	H41	23	-1926	-248
Thrower	(23,7)	H41	23	-1651	-61
Ant	(111,8)	H41, H41	50	1047	1440
Minitaur	(7, 13)	L	13	4.84	7.2
Minitaur	(7, 13)	L	50	6.08	7.91
Minitaur	(7, 13)	H41	13	7.12	9.34

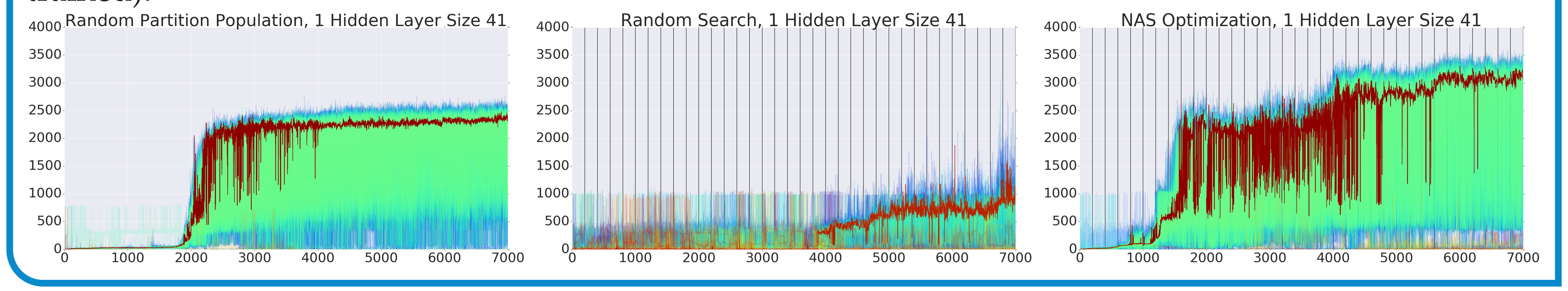
Environment	Architecture	Reward	# weight-params	compression	# bits
Striker	Chromatic	-248	23	95%	8198
	Masked	-967	25	95%	8262
	Toeplitz	-129	110	88%	4832
	Circulant	**-120**	82	90%	3936
	Unstructured	**-117**	1230	0%	40672
HalfCheetah	Chromatic	**3779**	17	94%	6571
	Masked	**4806**	40	92%	8250
	Toeplitz	2525	103	85%	4608
	Circulant	1728	82	88%	3936
	Unstructured	3614	943	0%	31488
Hopper	Chromatic	**3408**	11	92%	3960
	Masked	2196	17	91%	4726
	Toeplitz	**2749**	94	78%	4320
	Circulant	2680	82	80%	3936
	Unstructured	2691	574	0%	19680
Walker2d	Chromatic	**3695**	17	94%	6571
	Masked	1781	19	94%	6635
	Toeplitz	1	103	85%	4608
	Circulant	3	82	88%	3936
	Unstructured	**2230**	943	0%	31488

(Left) Rewards when using Chromatic Networks. **(Right)** Comparisons to other methods.

Note that adding more layers while maintaing the same number of partitions (i.e. true weights) can boost performance, due to increased representation power.

Comparisons to Random Search

NAS Search produces better partitionings than random sampling or random search (i.e. controller is not trained):



References

[1] Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard E. Turner, and Adrian Weller. Structured evolution with compact architectures for scalable policy optimization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 969–977, 2018.

[2] Hieu Pham, Melody Y. Guan, Barret Zoph, Quoc V. Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 4092–4101, 2018.

[3] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search of static linear policies is competitive for reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1800–1809, 2018.

[4] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv*, abs/1703.03864, 2017.