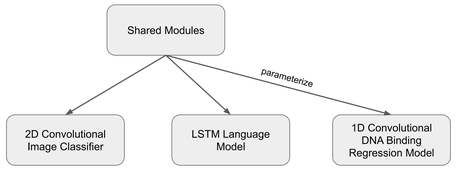
Modular Universal Reparameterization: Deep Multi-task Learning Across Diverse Domains

Elliot Meyerson [1] Risto Miikkulainen [1,2]

[1]Cognizant [2]The University of Texas at Austin

Overview

- Deep learning applications continue to become more diverse; they benefit from *methodological sharing*.
- Beyond shared methods, can general problem solving arise from *learned sharing* across such diverse tasks?
- Consider the setting where there is no obvious overlap between task architectures. A first approach (MUiR):
  1. Decompose any set of (architecture,task) pairs into potentially related subproblems;
  2. Optimize their sharing with an efficient stochastic algorithm.
- Results confirm that sharing learned functionality across diverse domains and architectures is beneficial.



Problem Statement

Given an arbitrary set of (architecture,task) pairs, can learned functionality be shared across architectures to improve performance in each task?

Solution Requirements and Related Work

1. A solution must support any given set of architectures;
2. A solution must automatically *align* parameters across the architectures.

Existing deep multi-task learning (DMTL) methods do not satisfy both conditions:

- Classical [5, 6, 11, 12, 36] and factorization methods [20, 27, 32] do not meet condition (2);
- Existing module assembly approaches [19, 22, 25, 28] do not satisfy (1);
- Designing a new customized architecture [4, 10, 14] satisfies neither (1) nor (2).

Reparameterizing Architectures with Hypermodules

Given T tasks $\{\{\boldsymbol{x}_{ti}, \boldsymbol{y}_{ti}\}_{i=1}^{N_t}\}_{t=1}^{T}$, with architectures $\{\mathcal{M}_t\}_{t=1}^{T}$, with parameters $\theta_{\mathcal{M}_t}$. The goal of MTL is to find

$$\bigcup_{t=1}^{T} \theta^*_{\mathcal{M}_t} = \underset{\bigcup_{t=1}^{T} \theta_{\mathcal{M}_t}}{\operatorname{argmin}} \frac{1}{T} \sum_{t=1}^{T} \frac{1}{N_t} \sum_{i=1}^{N_t} \mathcal{L}_t(\boldsymbol{y}_{ti}, \hat{\boldsymbol{y}}_{ti}), \tag{1}$$

If each tensor in each $\theta_{\mathcal{M}_t}$ can be decomposed into equally-sized parameter blocks $\boldsymbol{B}_\ell \in \mathbb{R}^{m \times n}$, we can write

$$\bigcup_{t=1}^{T} \theta_{\mathcal{M}_t} = (\boldsymbol{B}_1, \ldots, \boldsymbol{B}_L). \tag{2}$$

Parameter tensors in practice can be decomposed in this way so that each $\boldsymbol{B}_\ell$ induces a linear pseudo-task [23].

To share structure, the $\boldsymbol{B}_\ell$ are parameterized by $K < L$ hypernetworks $\{\boldsymbol{H}_k\}_{k=1}^{K}$ [9, 29] and contexts $\{\boldsymbol{z}_\ell\}_{\ell=1}^{L}$:

$$\bigcup_{t=1}^{T} \theta_{\mathcal{M}_t} = [(\boldsymbol{H}_1, \ldots, \boldsymbol{H}_K), (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_L)]. \tag{3}$$

In this paper, $\boldsymbol{B}_\ell = \psi(\ell) \bar{\times}_1 \boldsymbol{z}_\ell$, where $\psi : \{1, \ldots, L\} \to \{\boldsymbol{H}_k\}_{k=1}^{K}$ is the alignment function.

Efficiently Optimizing the Mapping of Modules to Blocks

Subalignments $\{\psi_d\}_{d=1}^{D}$ of ψ can be optimized in parallel, if each has a distinct evaluation function h_d.

For simplicity, let each be optimized by a (1+λ)-EA [7, 24, 30], a component of previous DMTL methods [19, 23].

Theorem The expected time of the decomposed K-valued (1+1)-EA is $O(\frac{KL(\log L - \log D)\log D}{D})$, when all h_d are linear.

Decomposition Level	None (Multi-task)	Per-task	Per-block
Expected Convergence Time	$O(KL \log L)$	$O\left(\frac{KL(\log L - \log T)\log T}{T}\right)$	$O(K \log L)$

Biasing optimization towards more frequently-used hypermodules addresses the bottleneck of large K:

Theorem The expected time of the decomposed K-valued (1+1)-EA with pessimistic initialization and proportional sampling is $O(\log L)$, when $D = L$, and all h_d are linear.

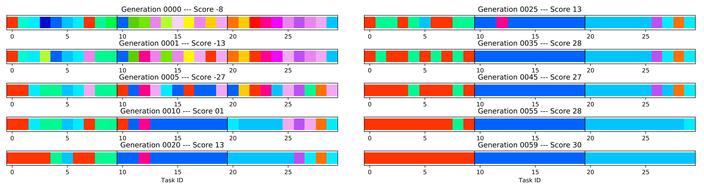
To achieve per-block decomposition, MUiR interleaves iterations of (1+λ)-EA with backpropagation via

$$\boldsymbol{B}_\ell = \sum_{i=0}^{\lambda} \psi^i(\ell) \bar{\times}_1 \boldsymbol{z}_\ell \cdot \text{softmax}(\boldsymbol{s}_\ell)_i, \tag{4}$$

where the learned $\text{softmax}(\boldsymbol{s}_\ell)_i$ indicates the model's belief that $\psi^i(\ell) \approx h_d$ is the best choice for the ℓth block.

Validation on a Classic Synthetic Dataset

The data set has 30 tasks with well-defined task-grouping [15]. MUiR quickly converges to the optimal mapping:



- h_d is compared against random and oracle evaluation functions.
- MUiR achieves optimal RMSE in the clean setting.
- MUiR on par with best reported results in noisy setting, despite increased problem difficulty due to
  1. data withheld for validation,
  2. absence of additional regularization,
  3. having to learn the number of groups automatically.

Method	Clean	Noisy
STL [15]	-	0.97
MTL-FEAT [3]	-	0.48
DG-MTL [15]	-	0.42
GO-MTL [17]	-	**0.35**
STL (ours)	1.35 ± 0.01	1.49 ± 0.01
MUiR + Random	1.26 ± 0.04	4.67 ± 1.48
MUiR + Oracle	0.77 ± 0.77	$\mathbf{0.37} \pm 0.00$
MUiR + Optimization	$\mathbf{0.00} \pm 0.00$	$\mathbf{0.38} \pm 0.00$

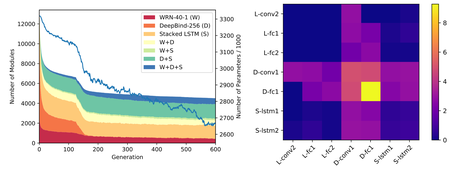
Experiments with Diverse Architectures and Modalities

Cross-modal experiments were run with CIFAR-10 [16], WikiText-2 [21], and CRISPR DNA Binding Prediction [13]:

Modality	Architecture	Metric	Baseline	Intratask	W+S	W+D	S+D	W+S+D	L+S	L+D	L+S+D
Vision	WRN-40-1 (W) [33]	Classificatin Error	**8.48**	**8.50**	8.69	9.20	-	9.02	-	-	-
Text	Stacked LSTM (S) [34]	Perplexity	134.41	132.06	130.63	-	132.62	**128.10**	**129.73**	-	130.77
DNA	DeepBind-256 (D) [2, 35]	Mean Squared Error	0.1540	0.1466	-	**0.1461**	0.1469	**0.1464**	-	0.1469	**0.1464**
Vision	LeNet (L) [18]	Classification Error	21.08	20.67	-	-	-	-	21.02	**19.59**	**20.23**

- MUiR always improves the text and genomics models, especially when they share with WRN.
- For L+S+D, the improvement due to MUiR is significant for all tasks.
- WRN behavior suggests the power of MUiR comes from diversity and identifies a key optimization challenge.

Analysis of Cross-modal Module Sharing Dynamics



Parameter Group	Stdev	Mean	Norm	Max
Hypermodules	7e-4	3e-1	8e-4	6e-3
Contexts	1e-43	1e-143	4e-138	5e-126
Linear Maps	3e-153	5e-2	5e-153	4e-146

- *Left.* The proportion of modules shared exclusively by each subset of tasks is architecture-dependent.
- *Mid.* W.r.t. sharing rate, the 1D-Conv model plays a central role between the 2D-Conv and 1D-LSTM models.
- *Right.* p-values for differences in Generic vs. Specific tensor statistics; The Generic show better generalizability.

Ablation and Comparison to other DMTL Methods

Method	LeNet	Stacked LSTM	DeepBind
Single Task Learning	21.46	135.03	0.1543
Classical DMTL (e.g., [8, 12, 36])	21.09	145.88	0.1519
Parallel Adapters [27]	21.05	132.02	0.1600
MUiR + Hierarchical Init.	**20.72**	**128.94**	**0.1465**
MUiR	**20.51**	**130.70**	**0.1464**

$\dim(\boldsymbol{z}_\ell)$	LeNet	Stacked LSTM	DeepBind
0	21.89	144.52	0.1508
1	21.80	140.94	0.1477
2	**20.40**	133.94	0.1504
4	**20.51**	**130.70**	**0.1464**
8	20.62	**130.80**	**0.1468**

- *Left.* Cross-modal comparisons were run with existing DMTL methods using Eq. 2.
  - The required hierarchical alignment was implemented by a topological sort of blocks in each architecture.
  - Even this rough alignment leads to improvement over STL, showing the value of Eq. 2.
  - Initializing ψ with this alignment leads to similar results for MUiR, showing robustness to design choices.
- *Right.* Experiments with different context sizes $\dim(\boldsymbol{z}_\ell)$ show the value of hypermodules over hard sharing.

Discussion and Future Work

- MUiR shows that functionality can be effectively shared in diverse settings. Many improvements are possible:
  - more sophisticated factorization than hypermodules [20, 32]; more sophisticated optmization than (1+λ)-EA.
- For lifelong learning [1, 26, 31], modules can be collected, refined, and assembled to solve tasks as they appear.

References

[1] D. Abel, D. Arumugam, L. Lehnert, and M. Littman. State abstractions for lifelong reinforcement learning. In *Proc. of ICML*, 2018.

[2] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 2015.

[3] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 2008.

[4] H. Bilen and A. Vedaldi. Integrated perception with recurrent multi-task neural networks. In *NIPS*. 2016.

[5] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer US, 1998.

[6] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *Proc. of ICRA*. 2017.

[7] B. Doerr, T. Jansen, and C. Klein. Comparing global and local mutations on bit strings. In *Proc. of GECCO*. 2008.

[8] D. Dong, H. Wu, W. He, D. Yu, and H. Wang. Multi-task learning for multiple language translation. In *Proc. of ACL*, pages 1723–1732. 2015.

[9] D. Ha, A. M. Dai, and Q. V. Le. Hypernetworks. In *Proc. of ICLR*. 2017.

[10] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proc. of EMNLP*. 2017.

[11] J. T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *Proc. of ICASSP*. 2013.

[12] Z. Huang, J. Li, S. M. Siniscalchi, et al. Rapid adaptation for deep neural networks through multi-task learning. In *Proc. of Interspeech*. 2015.

[13] C. Jung, J. A. Hawkins, S. K. Jones, et al. Massively parallel biophysical analysis of crispr-cas complexes on next generation sequencing chips. *Cell*, 2017.

[14] L. Kaiser, A. N. Gomez, N. Shazeer, A. Vaswani, N. Parmar, L. Jones, and J. Uszkoreit. One model to learn them all. *arXiv*, 2017.

[15] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *Proc. of ICML*, 2011.

[16] A. Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. 2009.

[17] A. Kumar and H. Daumé, III. Learning task grouping and overlap in multi-task learning. In *Proc. of ICML*, 2012.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 1998.

[19] J. Liang, E. Meyerson, and R. Miikkulainen. Evolutionary architecture search for deep multitask networks. In *Proc. of GECCO*, 2018.

[20] M. Long, Z. Cao, J. Wang, and P. S. Yu. Learning multiple tasks with multilinear relationship networks. In *NIPS*. 2017.

[21] S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models. *arXiv*, 2016.

[22] E. Meyerson and R. Miikkulainen. Beyond shared hierarchies: Deep multitask learning through soft layer ordering. In *Proc. of ICLR*. 2018.

[23] E. Meyerson and R. Miikkulainen. Pseudo-task augmentation: From deep multitask learning to intratask sharing—and back. In *Proc. of ICML*. 2018.

[24] F. Neumann and C. Witt. On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling. In *Proc. of IJCAI*. 2015.

[25] P. Ramachandran and Q. V. Le. Diversity and depth in per-example routing models. In *Proc. of ICLR*. 2019.

[26] S.-A. Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *NIPS*, pages 506–516. 2017.

[27] S.-A. Rebuffi, H. Bilen, and A. Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proc. of CVPR*. 2018.

[28] C. Rosenbaum, T. Klinger, and M. Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *Proc. of ICLR*. 2018.

[29] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *ALife*, 2009.

[30] D. Sudholt. On the robustness of evolutionary algorithms to noise. Refined results and an example where noise helps. In *Proc. of GECCO*, 2018.

[31] S. Thrun and L. Pratt. *Learning to Learn*. 2012.

[32] Y. Yang and T. Hospedales. Deep multi-task representation learning: A tensor factorisation approach. In *Proc. of ICLR*. 2017.

[33] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv*, 2016.

[34] W. Zaremba, I. Sutskever, and O. Vinyals. Recurrent neural network regularization. *arXiv*, 2014.

[35] H. Zeng, M. D. Edwards, G. Liu, and D. K. Gifford. Convolutional neural network architectures for predicting dna-protein binding. *Bioinformatics*, 2016.

[36] Z. Zhang, L. Ping, L. C. Chen, and T. Xiaoou. Facial landmark detection by deep multi-task learning. In *Proc. of ECCV*, 2014.