Efficient Task-Relationship Estimation With Dominant Subnet Structures

Onkar Deshpande, Prateek Varshney, Nikil Ravi

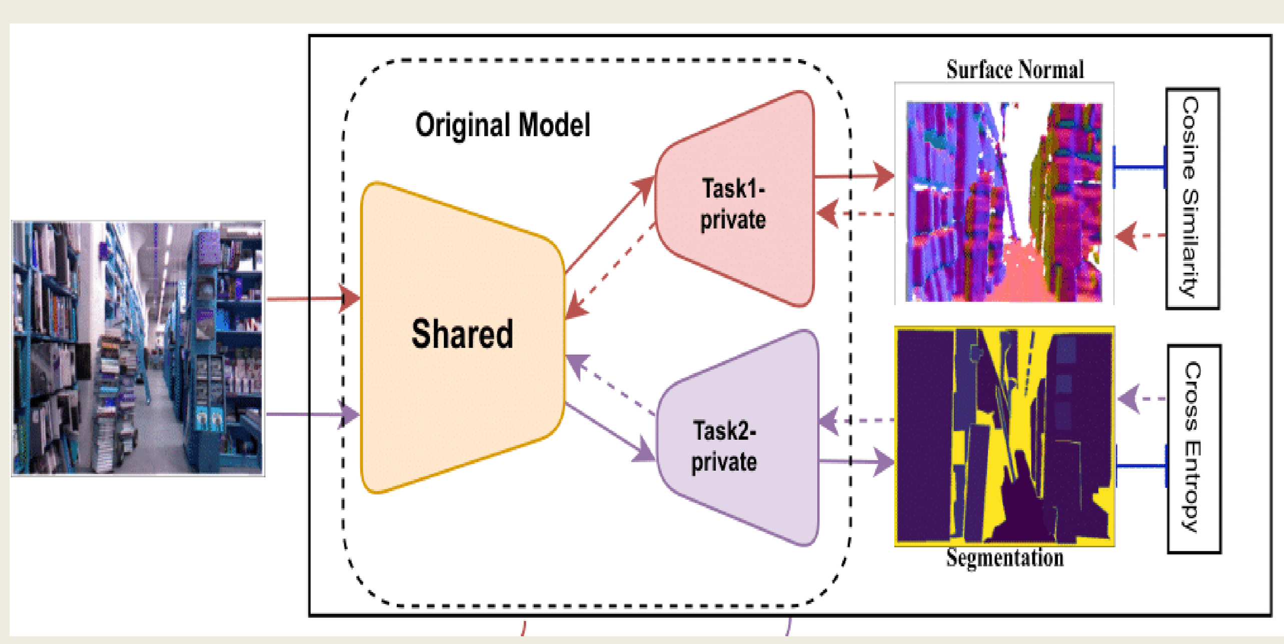
CS330 Project | Mentor: Alex Sun

Introduction

- **Objective**: To measure task-similarity efficiently with no end-to-end training.
- **Motivation:** Task similarity helps us select the best tasks groupings for joint multi-task training. Current methods have high computational requirements.
- **Hypothesis**: Tasks having strong relationships exhibit similar dominant sub-network structures
- **Solution**: Devise metrics to measure subnetwork similarity. If hypothesis is validated, we can use these metrics to efficiently discover task relationships

Background

- **Dataset**: NYUv2 dataset
- **Tasks**: Semantic Segmentation (seg), Surface Normalization (sn), Depth Prediction (depth)
- **Architecture**: DeepLab-ResNet for shared backbone + ASPP for task-specific heads. The model was trained using a multi-task loss; training was done for 5000 epochs.



Technical Methods

- Compute parameter-level importance scores using gradient over multiple batches of each task to determine the "dominant" subnetwork in the shared backbone
- Compute similarity scores for each C(k, 2) pair of tasks using saliency scores
- Correlate with accuracy measures; that is, for tasks A, B & C, does Sim(A, B) < Sim(A, C) => Acc(A | B) < Acc(A | C) ?

Sub-network computation: We use a SNIP-like pruning criterion with a **freshly-initialized NN** (no training required!) to get importance scores for each parameter; these scores gives us dominant subnetworks.

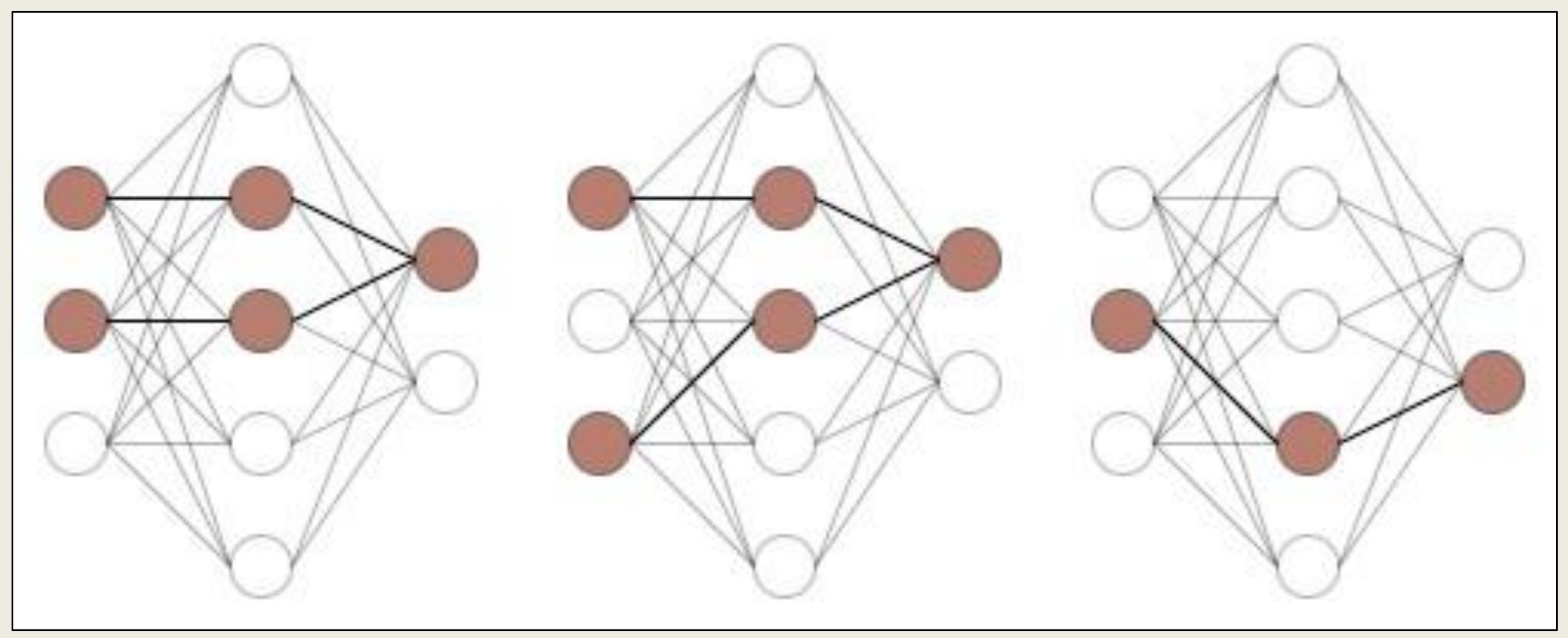


Figure 1. We obtain a mask/importance scores for task in the shared backbone. In the toy figure above, subnets 1 and 2 are more similar than subnets 1 and 3.

We compute subnetwork similarity using four different metrics. These are computed layer-wise and are averaged across layers.

$$\text{EPI-Sim}(T_1, T_2) = 1 - \frac{1}{L}\sum_{l=1}^{L} w_l \frac{|n_{(1,l)} - n_{(2,l)}|}{n_{(1,l)} + n_{(2,l)}}$$

$$\text{CSS}(T_1, T_2) = \frac{1}{L}\sum_{l=1}^{L} w_l \text{cosine}\Big(\text{Sal}(\mathbf{W}_{(1,l)}), \text{Sal}(\mathbf{W}_{(2,l)})\Big)$$

$$\text{IOU}(T_1, T_2) = \frac{1}{L}\sum_{l=1}^{L} w_l \frac{|\text{Mask}(\mathbf{W}_{(1,l)}) \cap \text{Mask}(\mathbf{W}_{(2,l)})|}{|\text{Mask}(\mathbf{W}_{(1,l)}) \cup \text{Mask}(\mathbf{W}_{(2,l)})|}$$

$$\text{MCSS}(T_1, T_2) = \frac{1}{L}\sum_{l=1}^{L} w_l \text{cosine}\Big(\text{Mask}(\mathbf{W}_{(1,l)}) * \text{Sal}(\mathbf{W}_{(1,l)}), \text{Mask}(\mathbf{W}_{(2,l)}) * \text{Sal}(\mathbf{W}_{(2,l)})\Big)$$

We experiment with 3 weighted-different averaging schemes:
Equal weights, linearly increasing weights, sparsity-level based weights

Results

	Metric	seg/sn	seg/d	sn/d
Equal weights	EPI based	0.876	0.785	0.855
	IOU	0.782	0.711	0.750
	Saliency (CSS)	0.969	0.975	0.964
	Masked saliency (MCSS)	**0.865**	**0.837**	**0.835**
Sparsity based weights	EPI based	0.799	0.668	0.782
	IOU	0.631	0.545	0.596
	Saliency (CSS)	0.959	0.960	0.938
	Masked Saliency (MCSS)	**0.768**	**0.731**	**0.726**

- MCSS was the most aligned with our task-specific (co-trained) accuracies.
- Metrics used for accuracy include (depending on task): pixel accuracy, IoU, absolute and relative error. For co-trained models, we have:

  Acc(*seg* | *sn)* > Acc(*seg* | *depth)*
  Acc(*sn* | *seg)* > Acc(*sn* | *depth)*
  Acc(*depth* | *seg)* > Acc(*depth* | *sn)*

References

[1] Silberman, Hoiem and Fergus. Indoor segmentation and support inference from rgbd images. In ECCV, 2012.

[2] Sun, Ali, Wang, Huang, and Shi. Disparse: Disentangled sparsification for multitask model compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12382–12392, 2022.

[3] Maying Shen, Pavlo Molchanov, Hongxu Yin, and Jose M Alvarez. When to prune? a policy towards early structural pruning. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.