A Closer Look at the Multi-Task Learning Neural Networks

Raphael Liu
Department of Applied Mathematics
University of Washington
Seattle, WA 98195
`raph651@uw.edu`

Abstract

Multi-Task Learning (MTL) aims to train Neural Networks on different tasks simultaneously, commonly with decoders handling specific tasks separately. The simultaneity makes it already practical and computation efficient in current applications like autonomous driving, natural language processing, robotic control, etc. Many researchers and industries have been including MTL-like models in their implementations, although the underlying MTL structures are not apparent. This project dedicates to examine state-of-the-art MTL methodologies and will show direct applications on two datasets: Fingers and NYUv2.

1 Introduction

Deep neural networks (DNNs) have been successful in different areas like computer vision, natural language processing, generative tasks, and classifications. Naturally, real-world applications require neural networks to predict more than one outputs at the same time. These outputs can be different in types. The MTL seeks to learn the relation between tasks, so related tasks can learn each others' features during one inference. For example, it is unnecessary to use one DNN for segmentation task, use another DNN for object detection task, and use one more DNN for object classification task, given the same image input. The MTL ideology is that a combined DNN extracts features from the input at once, and handles both tasks in parallel 1. Recent studies show that multi-task networks

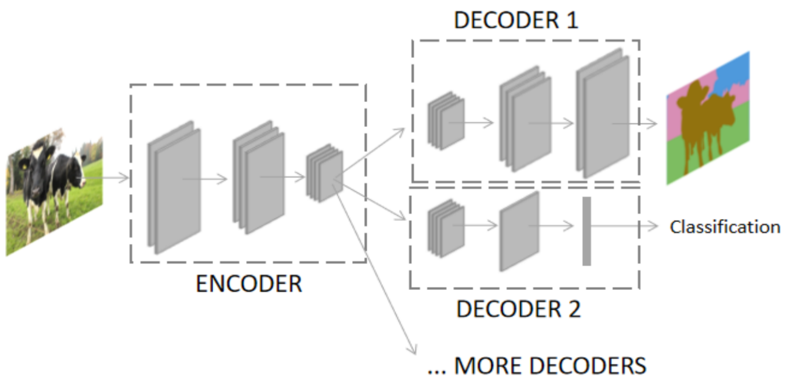


Figure 1: Simultaneous outputs in MTL

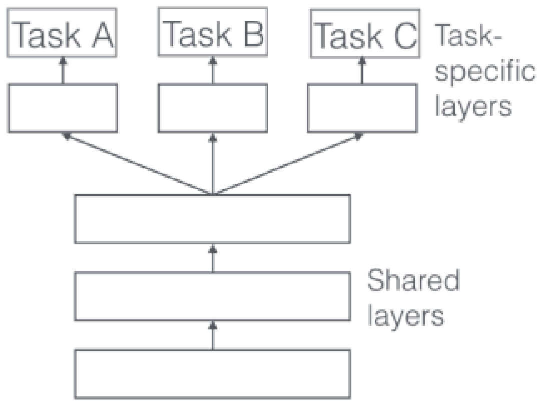


Figure 2: A typical hard-parameter sharing structure

can achieve equal, and even exceeding, performance value of its corresponding single-task networks. Heuer et al. [3] augmented the anchor-free CenterNet [16] approach to a multitask network called Multitask CenterNet (MCN) for object detection, semantic segmentation, and human-pose estimation on the MS COCO dataset. The inference time of MCN was less than half the inference time of the composite single-task network, which stacked up three models each solving one task independently. The overall performance was maintained, whereas MCN achieved higher accuracy on human-pose estimation than the single-task network.

We are interested in the current diverse methodologies in MTL, including weighting schemes, parameter sharing architectures, and attention modules. To understand how the MTL works, we implement a naive multi-task network that solves a toy classification task from Kaggle: Fingers [5]. The task is to count fingers and distinguish between left and right hands. Moving forwards, we apply a more complex multi-task network on NYUv2 dataset [12], which contains three image-to-image tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction.

2 Methods and Results

2.1 A toy example: Fingers

The Fingers dataset contains 21k 128x128 images of hand gesture, each labeled with the number of fingers in that image as well as left or right hand. The number of fingers has 6 classes, ranging from 0 to 5. Along with 2 classes in left or right, one way we approach this problem is jointing it to one hot encoded vector of 12 classes. The MTL aims to use a shared encoder to extract common features and use 2 decoders for separate classifications. The total loss is the sum of two classification loss (cross-entropy loss). The architecture is modeled as hard-parameter sharing (HPS) [10], where each decoder is isolated with other decoders as shown in Fig.2.

The encoder for both cases has the same structure of 4 convolutional layers. Three fully connected (FC) layers function as decoder in both cases, whereas in multi-task case each decoder is smaller in hidden size.

The loss curve and accuracy curve for both cases are shown in Fig.3. In multi-task case, both loss and accuracy converge faster. Only little effort in setting up hyperparameters will lead to a 100% accuracy. However, in single-task case it remains difficult to fine tune the hyperparameters such as batch size, learning rate, and optimizer to achieve a 99% or 100% accuracy.

2.2 3 image-to-image tasks on NYUv2

The NYUv2 dataset is comprised of 1449 images of indoor scenes as recorded by both RGB and Depth cameras. We use the proprocessed dataset which contains 795 images for training and 694 for testing [9]. Each image has been resized to 3x284x384 and three labeled images correspond to the three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The

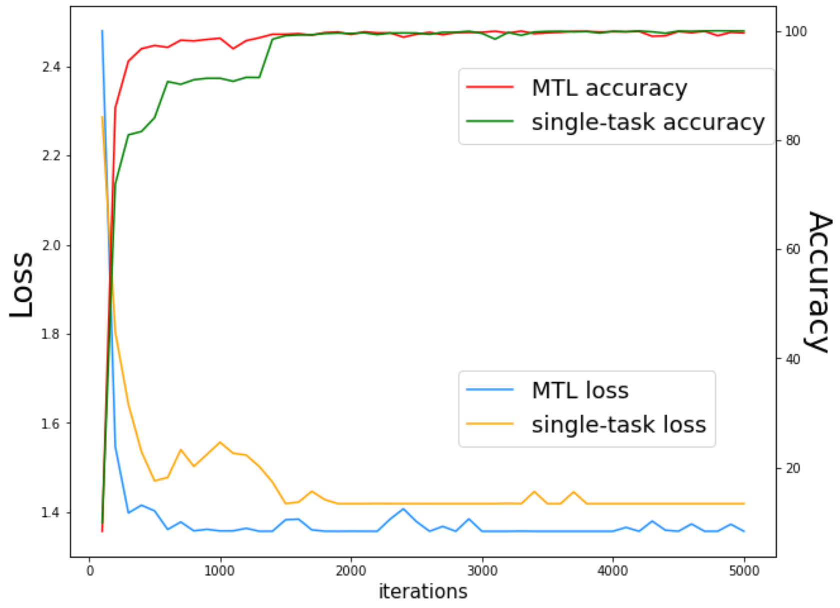


Figure 3: Loss and accuracy curve for both single and multi-task cases

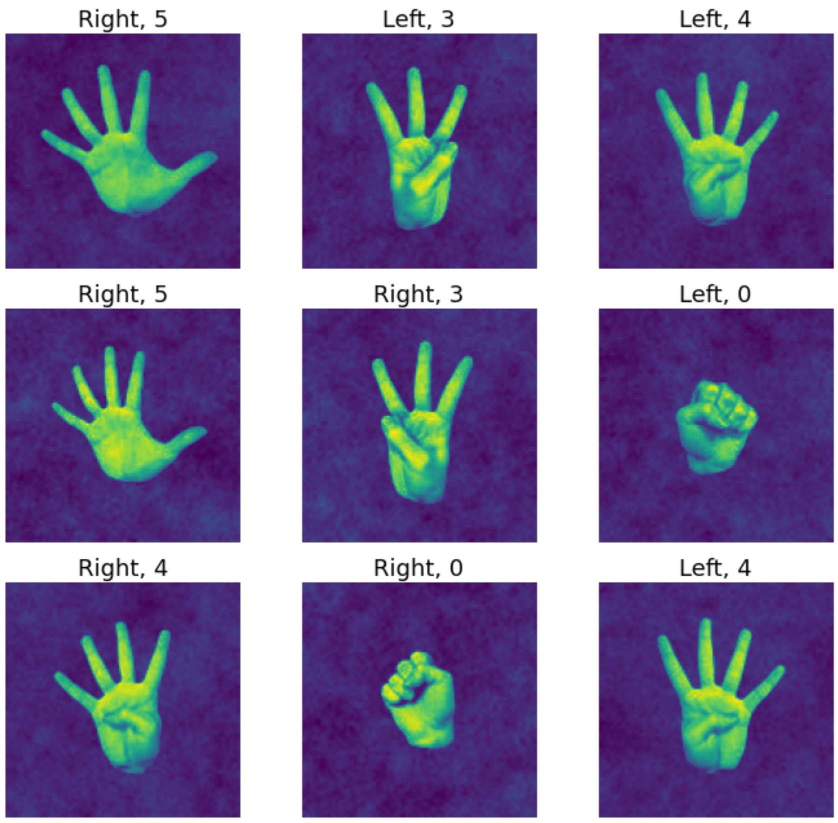


Figure 4: Prediction on Fingers test set

pixel-wise cross-entropy loss is applied to the semantic segmentation task. The L1 loss is applied to the depth estimation task. The cross-similarity loss is applied to the surface normal prediction task.

It it noted that even though the three tasks are related intuitively, each task loss can differ in magnitude. Therefore, the model can suffer from the unbalance: one task converges quickly while the others converge too slowly. To address the issue, multiple loss weighting strategies have been proposed to find the appropriate loss for each task $\{\lambda_t\}_{t=1}^{T}$ in Eq.1, where $\mathcal{D}_t$ denotes the dataset for each task, θ denotes the shared parameters, and ψ_t denotes the task-specific parameters.

$$\mathcal{L}_{MTL} = \sum_{t=1}^{T} \lambda_t l_t(\mathcal{D}_t; \theta; \psi_t) \tag{1}$$

State-Of-The-Art (SOTA) loss weighting strategies include Gradient Normalization (GradNorm)[1], Uncertainty Weights (UW) [4], MGDA [11], Dynamic Weight Average (DWA) [8], Projecting Conflicting-Gradient (PCGrad) [15], Gradient Sign Dropout (GradDrop) [2], Impartial Multi-Task Learning (IMTL) [7], Gradient Vaccine (GradVac) [13], and Random Loss Weighting (RLW) [6]. These weighting strategies can be categorized into three types: learning approach, solving approach, and the calculating approach. Both GradNorm and UW make the loss weights to be learnable parameters and update them during optimizer step. MGDA solves for the optimal loss weights as a quadratic programming problem. DWA, PCGrad, GradDrop, and GradVac, on the other hand computes $\{\lambda_t\}_{t=1}^{T}$ by combining gradients and losses of all the tasks.

Lin et al. (2021) proposed the RLW scheme which generates loss weights from a distribution in a sampling way. The convergence for RLW is almost always guaranteed. RLW also has higher probability to escape local minima when comparing to fixed loss weights, resulting in better performance. Besides the convergence analysis, Lin's experiments showed RLW can achieve comparable performance with other SOTA weighting methods without introducing additional computational costs. Therefore, we adopt the RLW scheme to assign weights for the three losses on NYUv2.

We utilize the dilated ResNet-50 pretrained on the ImageNet dataset as the shared encoder [14]. We adopt the Multi-Task Attention Network (MTAN) [9] approach to implement an attention module that shares features between encoder and decoders.

The network is ran on a GPU provided by Google Colab. The validation performance and comparison with SOTAs on the NYUv2 dataset is depicted in Table 1. Fig 5 shows three outputs for a test image.

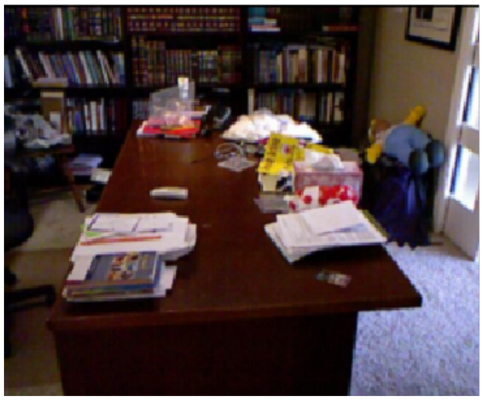
Table 1: Performance and comparison with SOTAs on NYUv2 dataset for segmentation, depth estimation, and surface normal prediction

	Segmentation		**Depth**		**Surface Normal**				
					Angle Distance		**Within t^o**		
Approach	**mIoU** (%)	**Pix Acc** (%)	**Abs Err**	**Rel Err**	Mean	Median	11.25	22.5	30
PSPNet (only-seg)	52.5	77.7							
CMX (only-seg)	56.9	80.1							
TokenFusion (only-seg)	54.2	79.0							
MTAN (seg+dep+norm)	53.3	75.4	0.3834	0.1556	23.58	17.12	35.18	60.88	71.91
Ours	54.7	75.6	0.3800	0.1574	22.47	15.66	37.86	63.95	74.01

3 Limitations

There are still possible limitations to MTL. Specifically, one should consider problems such as unbalanced tasks, less related tasks, high memory for heavy decoders, difficulty relating tasks, etc.

Unbalanced tasks can happen when one task is way easier while another is more subtle and difficult. The easier task requires a light decoder, and it can be fast to converge. But the more challenging tasks require a heavier decoder, which may coverge slowly. The weighting scheme plays a vital role in balancing the tasks by applying corresponding weighting factors.


(a) Input image

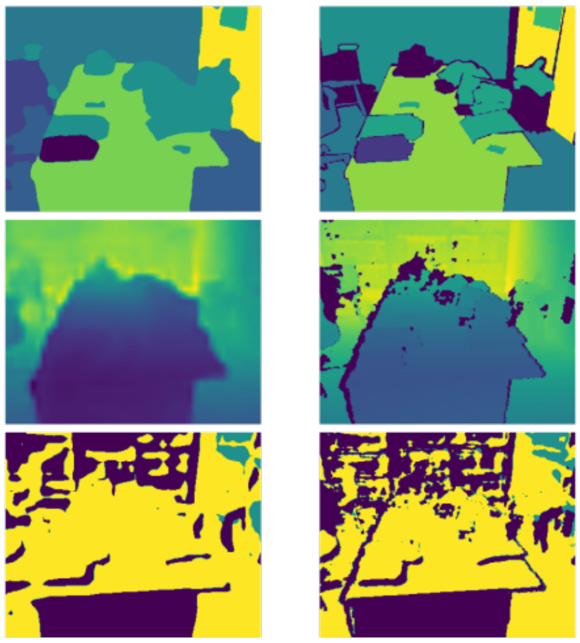

(b) Prediction vs Ground-Truth

Figure 5: Image-to-image outputs on NYUv2 test

Less related tasks can have a significant impact on MTL's success. When tasks are no longer related, their shared encoder will not provide enough information for some tasks. For those tasks, the performance will drop substantially.

High memory for heavy decoders can happen when most tasks require a heavy decoder, for instance, segmentation and generating tasks.

Researches are still ongoing to explore ways to connect tasks. In the future, there will be more various tasks, and it is vital to train the MTL model to learn their relations

References

[1] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *In Proceedings of the International Conference on Machine Learning. PMLR*, 2018.

[2] Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *In Proceedings of the 33rd Advances in Neural Information Processing Systems*, 2020.

[3] Falk Heuer, Sven Mantowsky, Syed Saqib Bukhari, and Georg Schneider. Multitask-centernet (mcn): Efficient and diverse multitask learning using an anchor free approach. *in ICCV 2021, arXiv preprint arXiv:2108.05060v2*, 2021.

[4] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[5] Pavel Koryakin. Fingers dataset. 2019. URL `https://www.kaggle.com/datasets/koryakinp/fingers`.

[6] Baijiong Lin, Feiyang Ye, and Yu Zhang. A closer look at loss weighting in multi-task learning. *arXiv preprint arXiv:2111.10603*, 2021.

[7] Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. *In Proceedings of the 9th International Conference on Learning Representations*, 2021.

[8] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[9] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. *in CVPR 2019, arXiv:1803.10704v2*, 2019.

[10] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098v1*, 2017.

[11] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *In Proceedings of the 31st Advances in Neural Information Processing Systems*, 2018.

[12] Nathan Silberman, Pushmeet Kohli, Derek Hoiem, and Rob Fergus. Indoor segmentation and support inference from rgbd images. 2012. URL `https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html`.

[13] Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *In Proceedings of the 9th International Conference on Learning Representations*, 2021.

[14] Fisher Yu, Vladlen Koltun, and Thomas A. Funkhouser. Dilated residual networks. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[15] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *In Proceedings of the 33rd Advances in Neural Information Processing Systems*, 2020.

[16] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850v2*, 2019.