Can Small Heads Help? Understanding and Improving Multi-Task Generalization

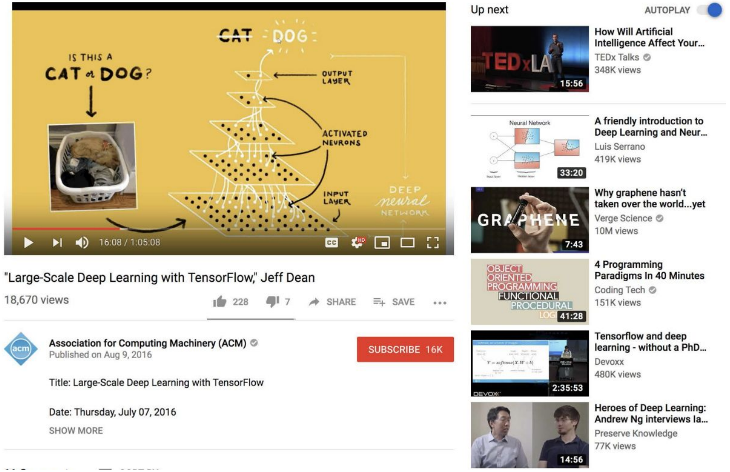
Yuyan Wang, Zhe Zhao, Bo Dai, Christopher Fifty, Dong Lin, Lichan Hong, Li Wei, Ed H. Chi

Google Research, Brain Team

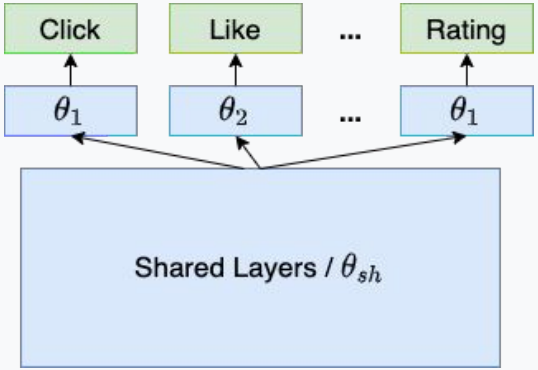
Google Research

Multi-Task Learning

A content recommendation platform may care about multiple types of user responses (click, like, rating etc.) [1]...



... How do we model them accurately and efficiently?



Google Research

[1] Zhao et al. "Recommending what video to watch next: a multitask ranking system." Recsys 2019.

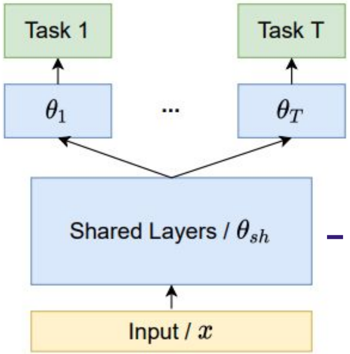
Multi-Task Learning (MTL)

Setup: T tasks sharing an input space X

Observations: $\{(x_i, y_i^1, \ldots, y_i^T)\}_{i=1}^{n}$

Optimization problem: Joint optimization of a vector-valued loss function $\min_{\theta}(\hat{\mathcal{L}}_1(\theta), \ldots, \hat{\mathcal{L}}_T(\theta))^\top$,

where $\hat{\mathcal{L}}_t(\theta) := \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_t(f_t(x_i; \theta_{sh}, \theta_t), y_i^t)$ is the empirical loss for task *t*.



Task-specific heads: Learning task specificities

Shared representation: Learning task commonalities

Figure 1: Shared-bottom architecture for a multi-task model.

[1] Zhao et al. "Recommending what video to watch next: a multitask ranking system." Recsys 2019.

MTL comes with trade-offs

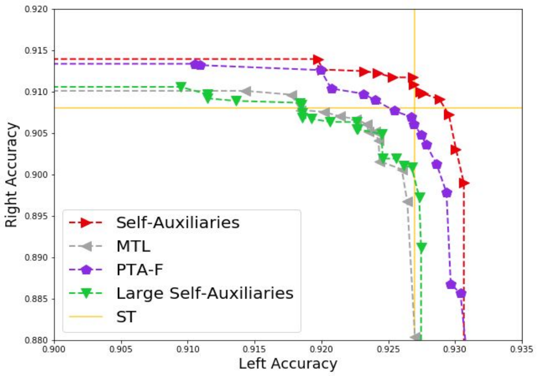
There exists a **Pareto frontier** across the performance of different tasks.

$$\hat{\mathcal{L}}(\theta) := \sum_{t=1}^{T} w_t \hat{\mathcal{L}}_t(\theta)$$

weight for task t

loss for task t

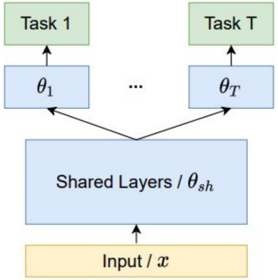
Multi-MNIST Dataset [1].



Multi-task trade-off between predicting **left** and **right** digits.

[1] Sener & Koltun. Multi-task learning as multi-objective optimization. NeurIPS 2018.

Multi-Task Learning

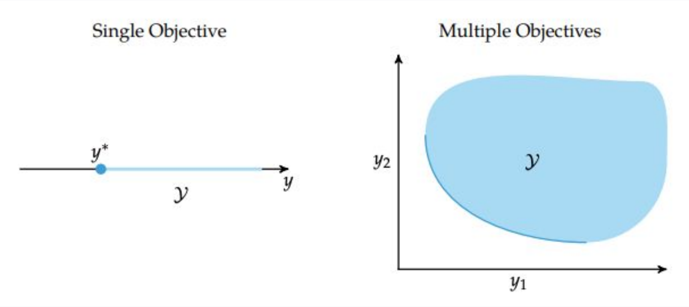


Goal:
Jointly optimize the performance of multiple tasks (a.k.a. **multi-task generalization**).

Why: Transfer learning / regularization / model efficiency/...

Challenge: Mitigate task training conflicts

Multi-objective optimization



Goal:
Obtain the Pareto frontier (set of Pareto optimal solutions)

Why: More than one objective **needs** to be optimized simultaneously

Challenge: Mitigate task training conflicts

MOO theory suggests that sufficient parameterization is needed for properly handling task conflicts -> Are larger models necessarily better for MTL?

Google Research

Figure source

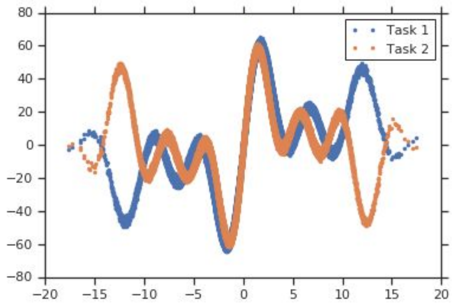
Insights

Are larger models necessarily better for MTL?

Are larger models always better for MTL?

Experiments on synthetic data

Data

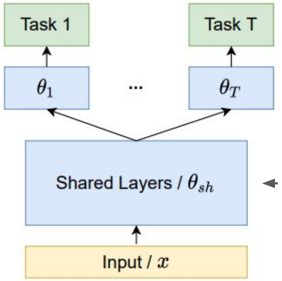


Task 1: $$y_1 = \sum_{w_1 \in W_1} \sin(w_1 x + 0.2 e_1)$$

Task 2: $$y_2 = \sum_{w_2 \in W_2} \sin(w_2 x + 0.2 e_2),$$

Small overlap to introduce both task **relatedness** and task **conflicts**.

Model Architecture



Task-specific heads

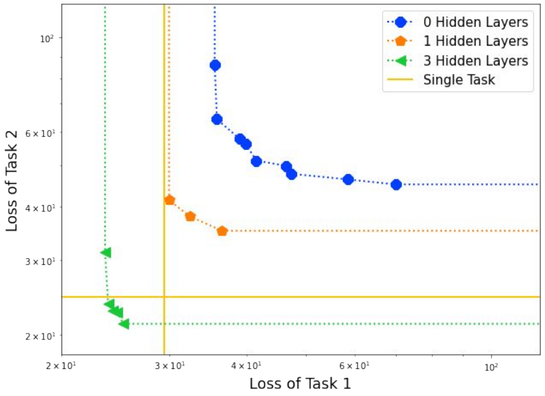
Varying sizes to observe effect on the Pareto frontier

Shared representation

Google Research

Are larger models always better for MTL?

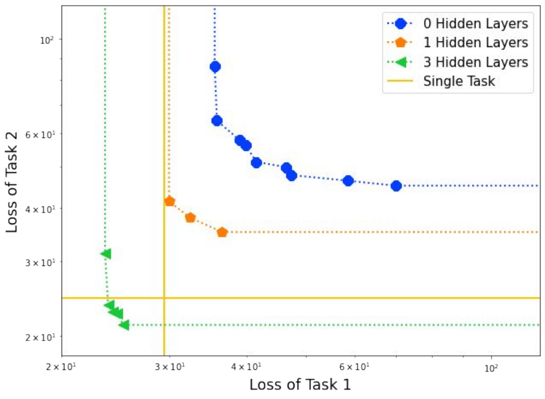
Benefits from larger MTL models...



By increasing the # hidden layers for the **task-specific heads** from 0 to 3, the Pareto frontier **improves**.

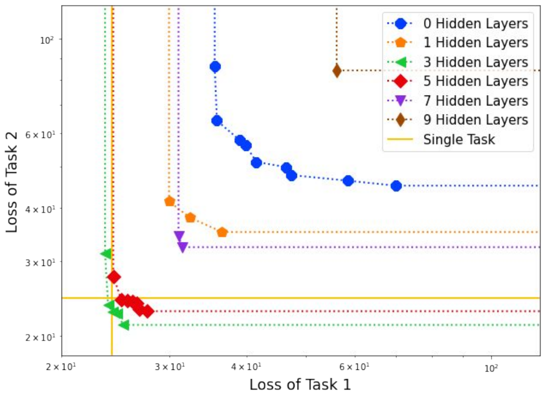
Are larger models always better for MTL?

Benefits from larger MTL models...



By increasing the # hidden layers for the **task-specific heads** from 0 to 3, the Pareto frontier **improves**.

Challenges from larger MTL models...



By further increasing the # hidden layers for the **task-specific heads** from 5 to 9, the Pareto frontier **deteriorates**.

Are larger models always better for MTL?

Insights: Largely ignored trade-off between **mitigating task training conflicts** and **multi-task generalization**:

MTL as a MOO problem:	MOO theory suggests that sufficient model capacity is needed for minimizing task training conflicts.	⇒ "Larger models are always better for MTL" (**?**)
However, treating MTL as MOO is limiting:	MTL leverages parameter sharing and inductive transfer which benefit the generalizability of the learned shared representations.	⇒ Larger models have **less** such benefits
Trade-off exists:	Large models mitigate task training conflicts better, but also undermines the benefit of sharing and hurt multi-task generalization.	⇒ Larger models are **NOT** always better for MTL!

Can we improve the trade-off?

Method

Improving multi-task generalization with small heads

Motivation: Achieving best of both worlds

***Small MTL models** generalize well to multiple tasks but suffer from task training conflicts.*

***Large MTL models** are able to better mitigate task training conflicts, but suffer from loss of multi-task generalization.*

Can we design an adaptive treatment that achieves the best of **both** worlds?

Method: Under-parameterized self-auxiliaries

Original task head

Self-auxiliary task head (small)

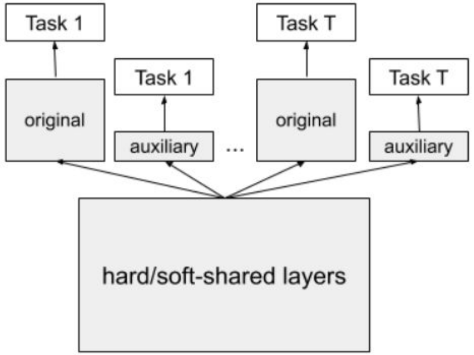


Figure 2: An illustration of under-parameterized self-auxiliaries for multi-task learning.

Original training loss for task t

$$\hat{\mathcal{L}}(\theta) = \sum_{t=1}^{T} w_t\left(\hat{\mathcal{L}}_t(\theta_{sh}, \theta_t) + \gamma \hat{\mathcal{L}}_t(\theta_{sh}, \theta_t^a)\right)$$

Self-auxiliary loss for task t

- At training time, each task is trained with an additional under-parameterized head (**little** extra training cost);
- At serving time, discard the self-auxiliaries and use the original head's output as prediction (**no** extra serving cost);

Method: Under-parameterized self-auxiliaries

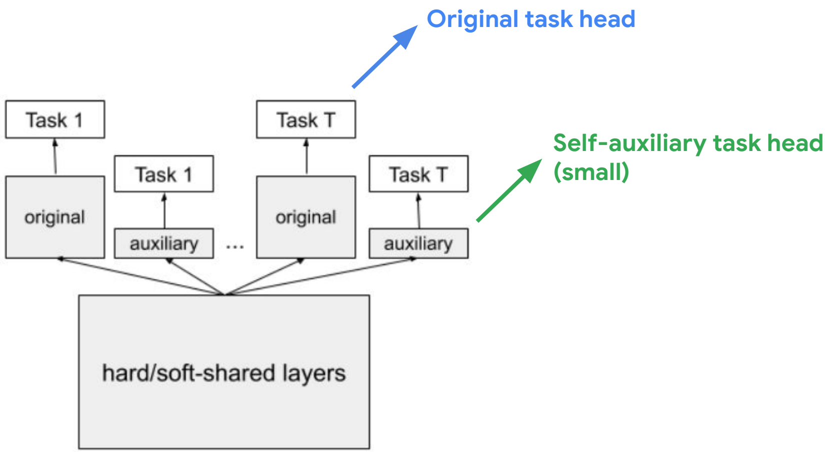


Figure 2: An illustration of under-parameterized self-auxiliaries for multi-task learning.

Simultaneously training the same task with two towers – once with full parameterization and the other with under-parameterization

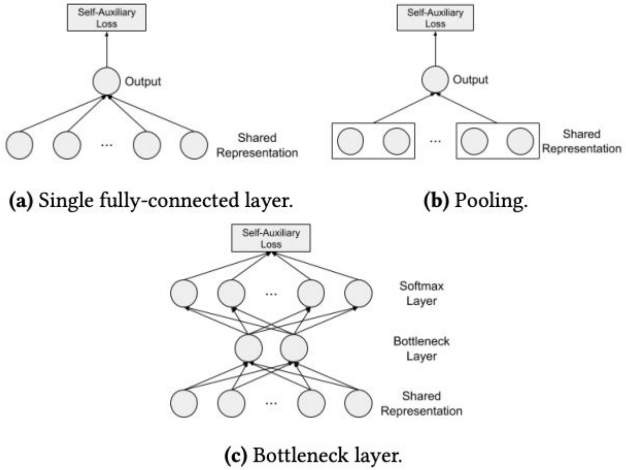
The shared representation is "forced" toward learning a representation which suits both task-specific parameterizations.

Therefore, sharing happens in the shared layers as much as possible - the proposed self-auxiliaries act as ***implicit regularization***!

Improved multi-task generalization performance

Google Research

Architectures for self-auxiliaries



(a) Single fully-connected layer.

(b) Pooling.

(c) Bottleneck layer.

Figure 3: Example architectures for under-parameterized self-auxiliaries. (a): Single fully-connected layer. (b): Single layer with average pooling. (c): Two-layer tower with bottleneck layer.

Experiments

Experiments: Multi-MNIST & Multi-FashionMNIST

- Datasets:
  - **Multi-MNIST & Multi-FashionMNIST**
    - overlaying two images on top of each other with a small offset.
- Tasks:
  - Predicting left (Task 1) and right (Task 2) digit / item jointly.
- Baselines:
  - **ST** (single task learning)
  - **MTL** (vanilla linear weighting)
  - **Uncertainty** reweighting [2]
  - **MGDA** [1]
  - **PCGrad** [3]
  - **PTA-F** [4]

Multi-MNIST Dataset [2].

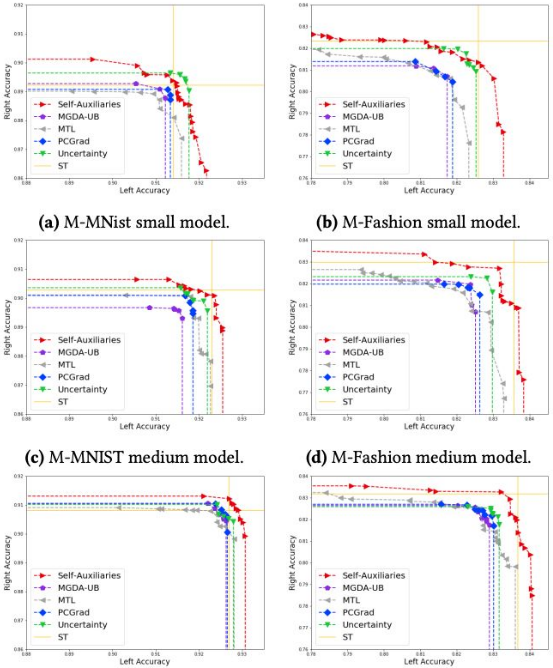
[1] Sener & Koltun. Multi-task learning as multi-objective optimization. NeurIPS 2018.
[2] Kendall et al. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. CVPR 2018.
[3] Yu et al. Gradient surgery for multi-task learning. NeurIPS 2020.
[4] Meyerson & Miikkulainen. Pseudo-task Augmentation: From Deep Multitask Learning to Intratask Sharing and Back. ICML 2018.

Experiments: Multi-MNIST & Multi-FashionMNIST

Small-sized model

Medium-sized model

Large-sized model



(a) M-MNist small model. **(b)** M-Fashion small model.

(c) M-MNIST medium model. **(d)** M-Fashion medium model.

(e) M-MNIST large model. **(f)** M-Fashion large model.

Larger models introduce more (multi-task) generalization challenges

⇩

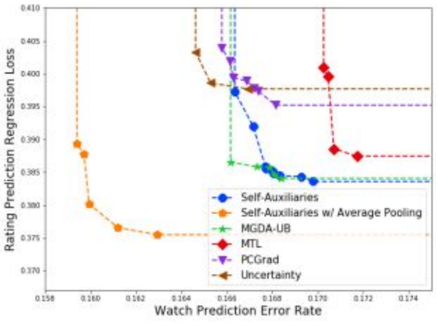
The larger the model, the greater the improvement our method ("Self-Auxiliaries") exhibits over the baselines.

Google Research

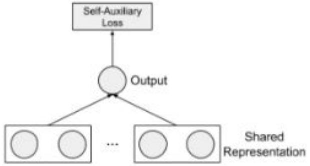
Experiments: MovieLens

	Watch Error	Rating MSE
MTL	0.172	0.387
Uncertainty	0.165	0.399
MGDA-UB	0.168	0.385
PCGrad	0.167	0.397
Self-Auxiliaries	0.168	0.385
Self-Auxiliaries-pooling	**0.161**	**0.377**

Table 1: Numerical results on MovieLens dataset.



(a) Self-auxiliaries vs. baselines.



(b) Pooling.

By further reducing the parameterization of self-auxiliary heads with average pooling, we can further improve its performance.

Experiments: An industrial content recommendation platform

- Setup:
    - An **industrial content recommendation platform** serving billions of users everyday.
- **8 Tasks** in total:
    - 4 tasks predicting user **satisfaction**-related scores
    - 4 tasks predicting user short-term and long-term **engagement** behaviors.

Experiments: An industrial content recommendation platform

Offline results

Task	Metric Name	PTA-F	Self-Auxiliaries (Ours)
S1	AUC	+0.22%	**+0.55%**
S2	AUC	+0.33%	+0.33%
S3	AUC	+1.13%	**+1.27%**
S4	AUC	+0.12%	+0.12%
E1	AUC	**+0.27%**	+0.14%
E2	AUC	+0.12%	+0.12%
E3	RMSE	-0.00%	**-0.08%**
E4	RMSE	-0.09%	**-0.18%**

(a) Per-task performance.

Metric Name	PTA-F	Self-Auxiliaries (Ours)
Average AUC	+0.351%	**+0.416%**
Average RMSE	-0.072%	**-0.153%**

(b) Average performance for classification (AUC) and regression (RMSE) tasks. Average AUC computes the average AUC over $S_1, S_2, S_3, S_4, E_1, E_2$, and Average RMSE computes the average RMSE over E_3, E_4.

Live experiment results

Metric Name	PTA-F	Self-Auxiliaries (Ours)
Page-specific Satisfaction	+0.15%***	**+0.17%*****
Site-wide Satisfaction	+0.01%	**+0.06%****
Page-specific Engagement	+0.13%***	**+0.15%*****
Site-wide Engagement	0.00%	**+0.05%****

** p-value < 0.05; *** p-value < 0.01.

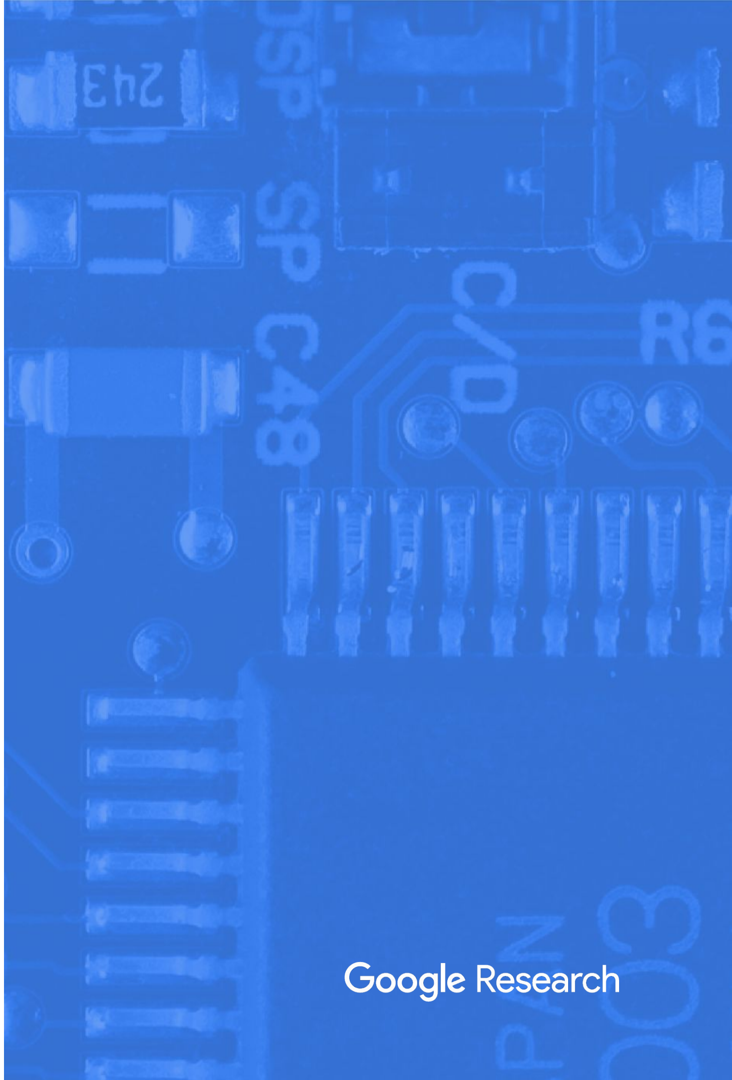
Table 3: Live experiment results. Metrics are shown in percentage improvement compared with current production model (control).

Our method achieves better Pareto efficiency from offline and online experiments.

Google Research

Key Takeaways

- Reveal the largely ignored **trade-off** between **minimizing task training conflicts** and **improving multi-task generalization** in multi-task deep learning.
- Larger models are **not** necessarily better than smaller ones in terms of multi-task generalization.
- Propose the use of **under-parameterized self-auxiliaries** to **automatically** balance multi-task generalization with mitigating task conflicts, via implicit regularization.



Thank You

Yuyan Wang
Google Research, Brain Team
yuyanw@google.com

Google Research