Harvard John A. Paulson School of Engineering and Applied Sciences

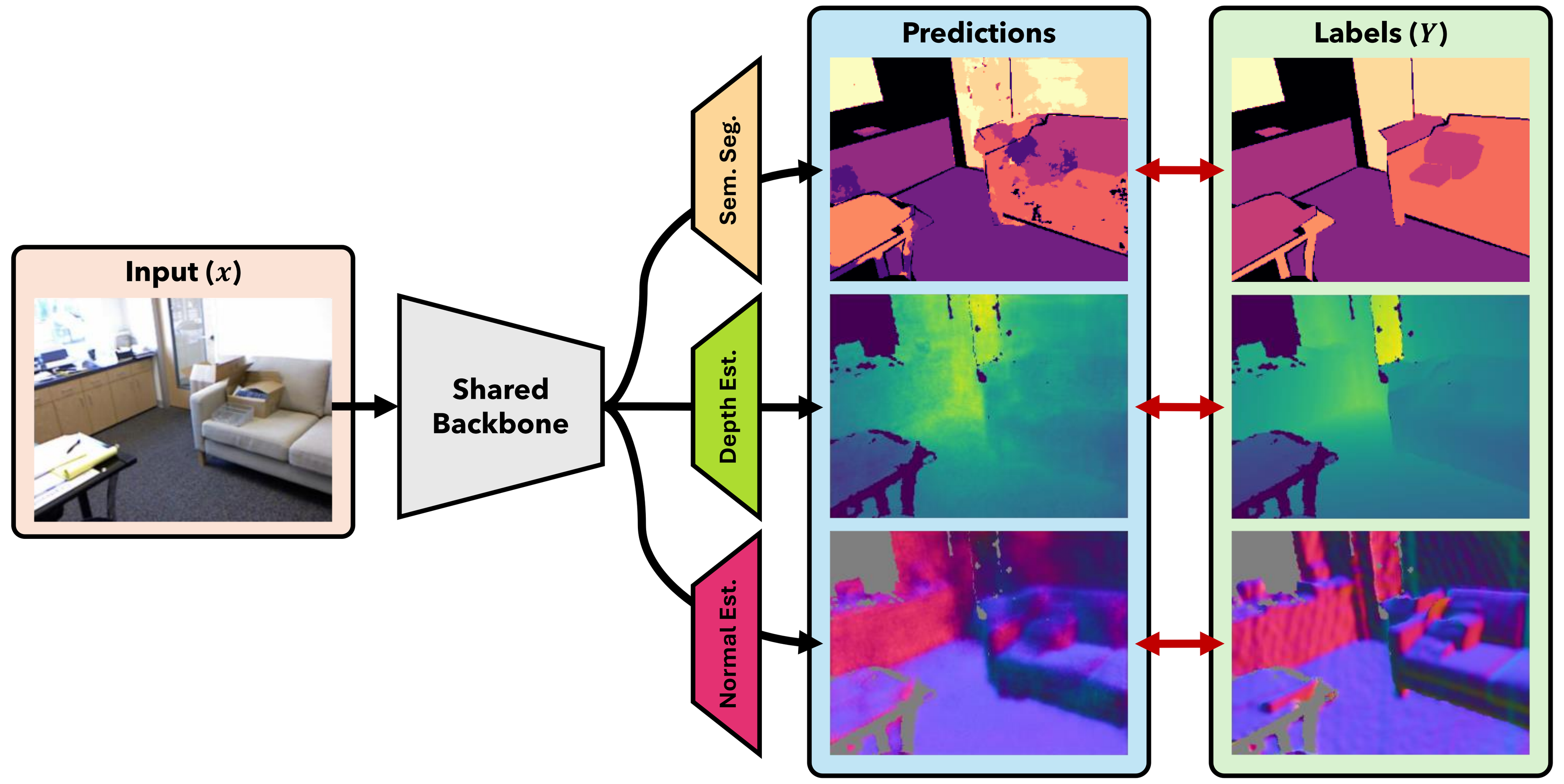
Joint-Task Regularization for Partially Labeled Multi-Task Learning

Kento Nishi*, Junsik Kim*, Wanhua Li, Hanspeter Pfister
kentonishi.com/JTR-CVPR-2024

CVPR JUNE 17-21, 2024 SEATTLE, WA

Problem

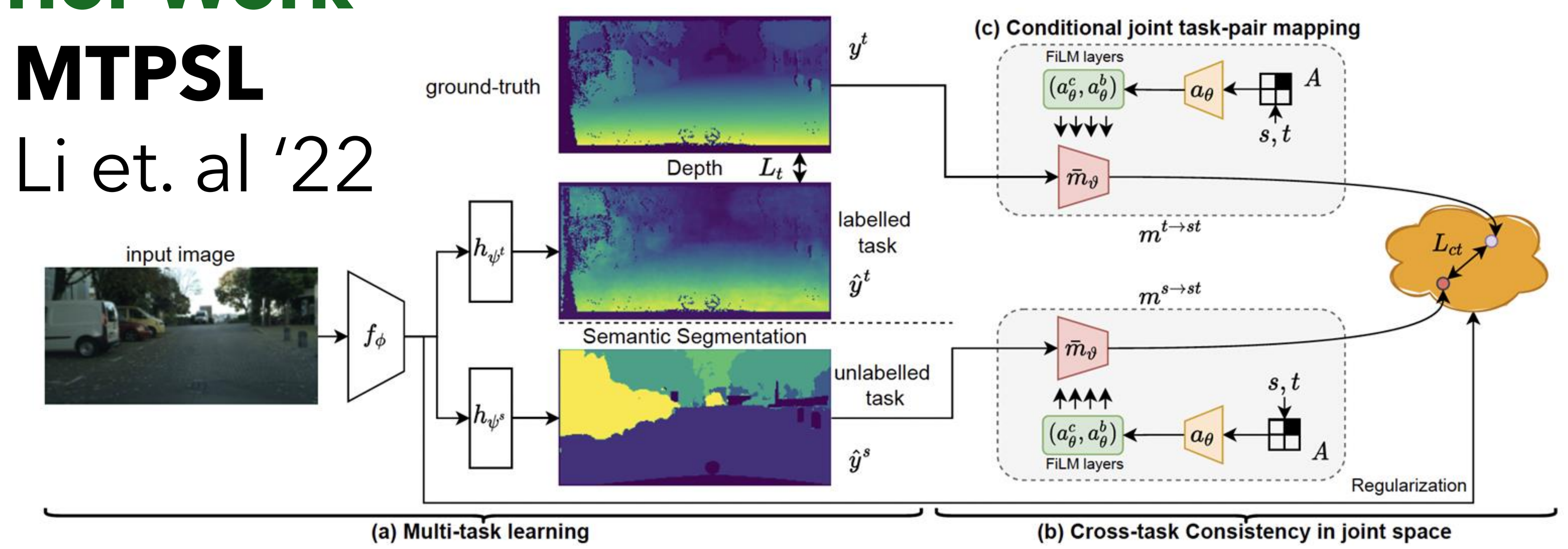
- **Multi-Task Learning** (MTL) aims to train **one model** to perform **multiple tasks jointly**.
- MTL in vision typically requires **each input** image (or pixel) to be **labeled** for **all tasks**.



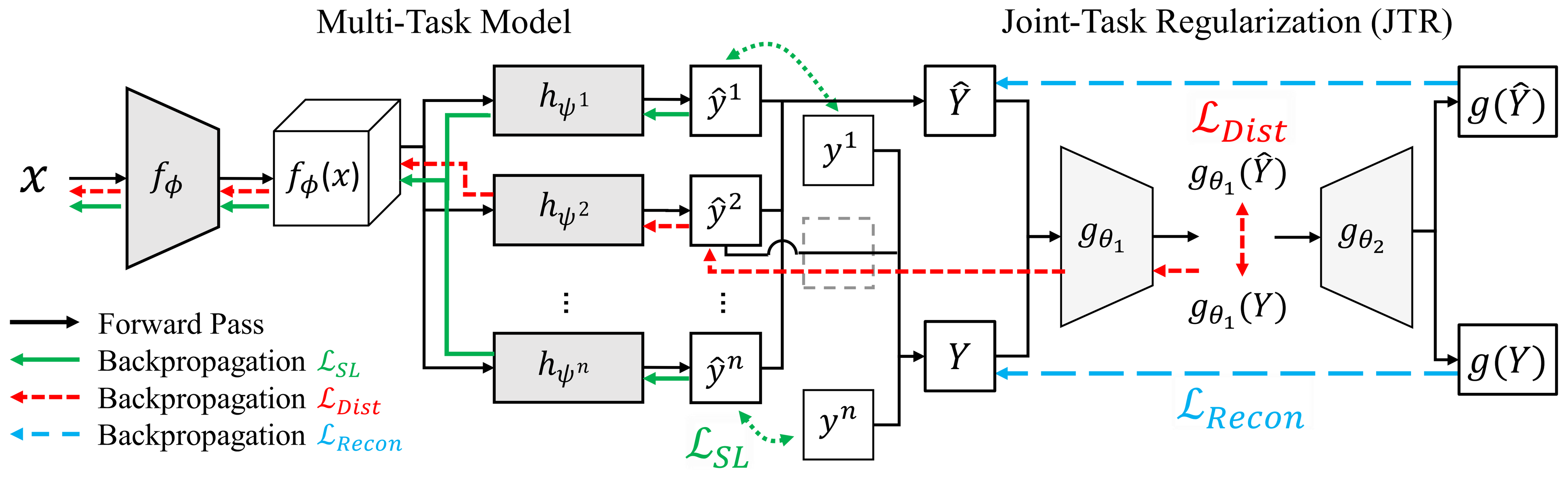
- Collecting multi-task data is **expensive**.
- Solution: Do MTL with **partially labeled data** (inputs may not have labels for all tasks).

Prior Work

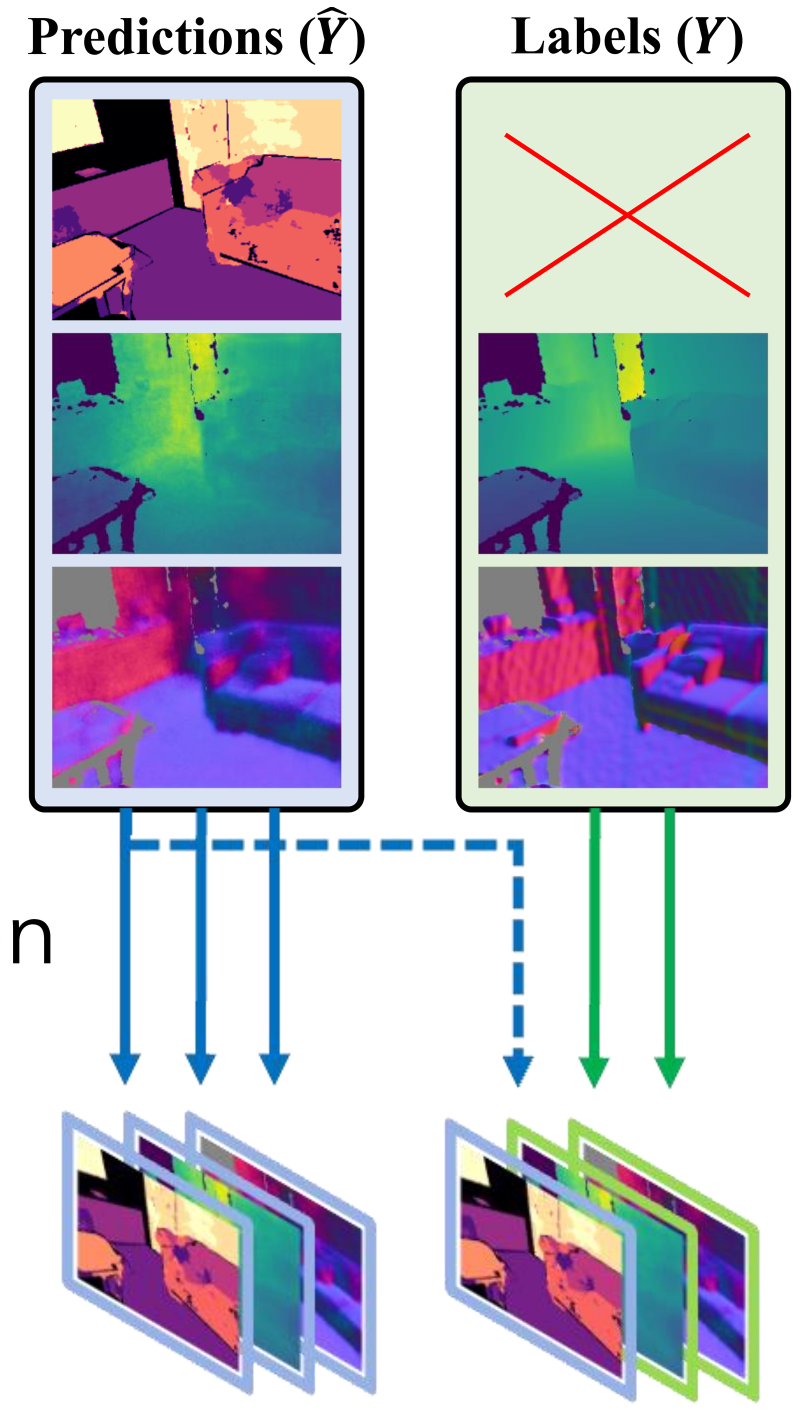
- **MTPSL**
  Li et. al '22



Our Method



- For each input, **stack all tasks** across channel dim.
  - Predictions → $\hat{Y}$
  - Labels, or predictions if task has no label → Y
- **Encode $\hat{Y}$ and Y** into a learned **Joint-Task Space** (with dims < $\sum$ pred. dims).
- **Minimize distance** between embeddings of $\hat{Y}$ and Y in the joint-task space.
- Enforce an **auto-encoder reconstruction loss** to prevent trivial JT spaces.



$[\hat{y}1; \hat{y}2; \hat{y}3]$ $[\hat{y}1; y2; y3]$

- **Benefits: simpler**, **linear complexity** wrt # of tasks, **better learning** of **cross-task relations!**

Experiments

- **Benchmarks** with **synthetic label scenarios**
  - ***onelabel***: labeled for only one of K tasks
  - ***randomlabels***: labeled for 1 to K − 1 tasks
  - ***halflabels***: labeled for ~50% of tasks
- **Select results** (see paper for more):

Method	Seg. ↑	Depth ↓	Norm. ↓	Δ% ↑
Supervised MTL*	27.05	0.6624	33.58	+0.000
Consistency Reg.*	29.50	0.6224	33.31	+5.300
Direct Map* [74]	29.17	0.6128	33.63	+5.060
Perceptual Map* [74]	32.20	0.6037	32.07	+10.80
MTPSL [33]	35.60	0.5576	29.70	+19.66
Ours (JTR)	**37.08**	**0.5541**	**29.44**	**+21.92**

NYU-v2 (3 tasks), *randomlabels*

Method	Seg. ↑	Depth ↓	Δ% ↑
Supervised MTL*	69.50	0.0186	+0.000
Consistency Reg.*	71.67	0.0178	+3.712
MTPSL [33]	72.09	0.0168	+6.702
Ours (JTR)	**72.33**	**0.0163**	**+8.219**

Cityscapes (2 tasks), *onelabel*

Scenario	Method	Δ% ↑
onelabel	Consistency Reg.	+1.567
	MTPSL [33]	+1.973
	Ours (JTR)	**+3.512**
randomlabels	Consistency Reg.	+2.254
	MTPSL [33]	+1.261
	Ours (JTR)	**+4.095**
halflabels	Consistency Reg.	+2.326
	MTPSL [33]	+3.160
	Ours (JTR)	**+5.075**

Taskonomy (7 tasks)

Method	Cityscapes Time ↓	Cityscapes VRAM ↓	Taskonomy Time ↓	Taskonomy VRAM ↓
MTPSL [33]	22h 10m	14.4GiB	223h 10m	34.4GiB
Ours (JTR)	23h 45m	19.2GiB	105h 00m	23.9GiB

Training Time/VRAM, NVIDIA A100

- **Conclusion:** JTR is **effective and efficient**!