Multi-Domain Incremental Learning for Semantic Segmentation

Prachi Garg[1] Rohit Saluja[1] Vineeth N Balasubramanian[2] Chetan Arora[3] Anbumani Subramanian[1] C. V. Jawahar[1]

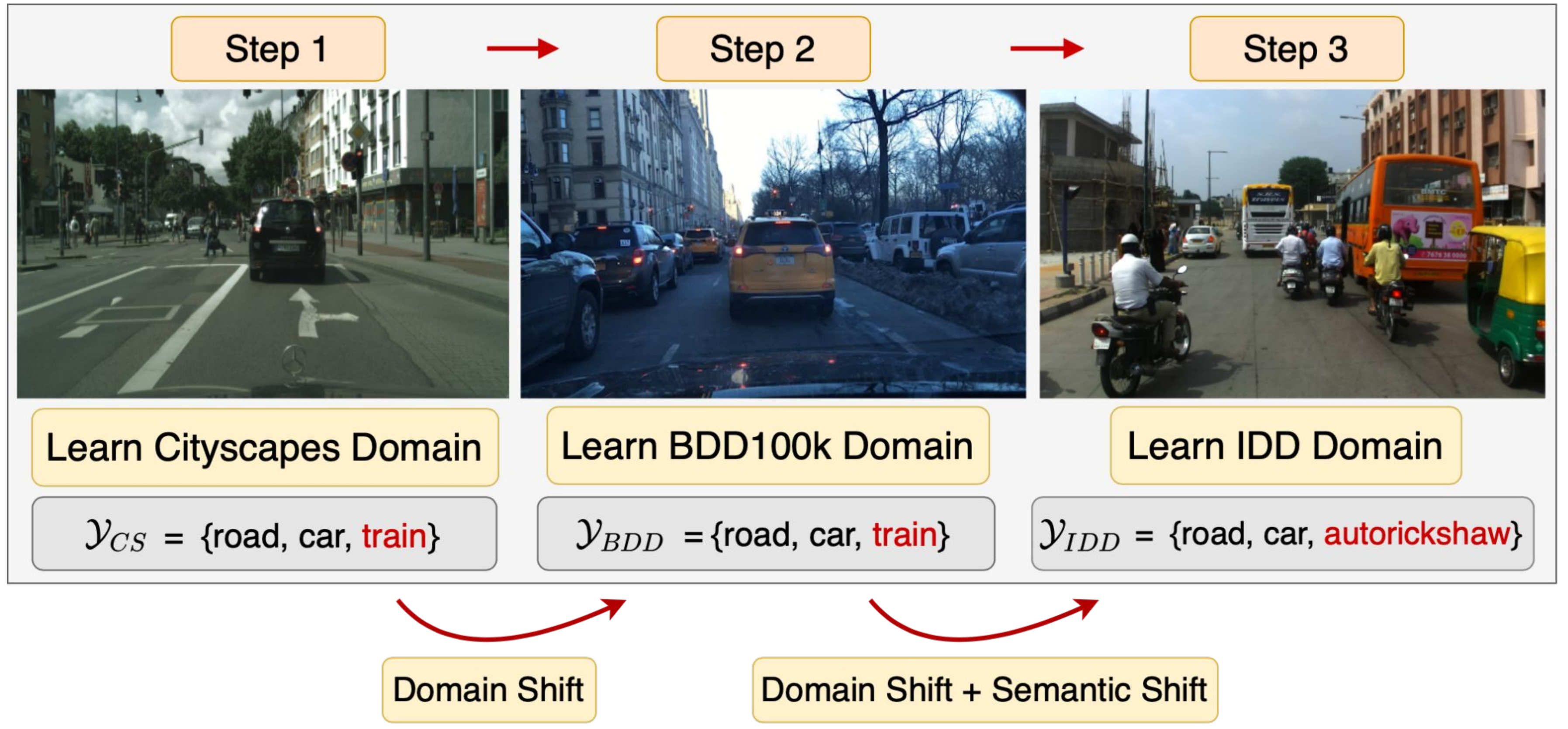
[1]CVIT - IIIT Hyderabad, India [2]IIT Hyderabad, India [3]IIT Delhi, India

Code/Arxiv

WAIKOLOA, HI JAN 4-8

Can a segmentation model trained on the road scenes of a particular city extend to incrementally learn novel geographical domains?

Our Novel Problem Setting



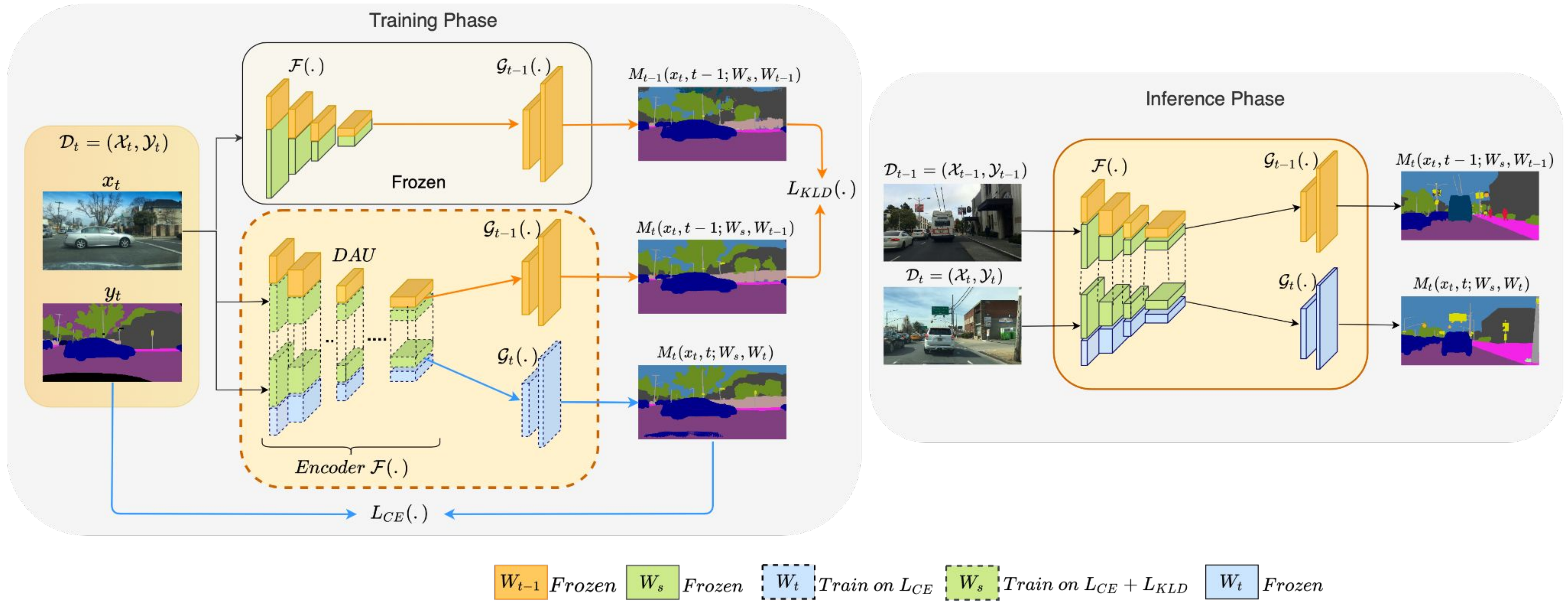
What is Multi-Domain Incremental Learning?

- ➔ Incrementally (*sequentially*) learn a new domain on an existing model.
- ➔ While RETAINING performance on the previously seen domains.
- ➔ Given previous domains' data is not accessible.
- ➔ Label spaces may be non-overlapping.

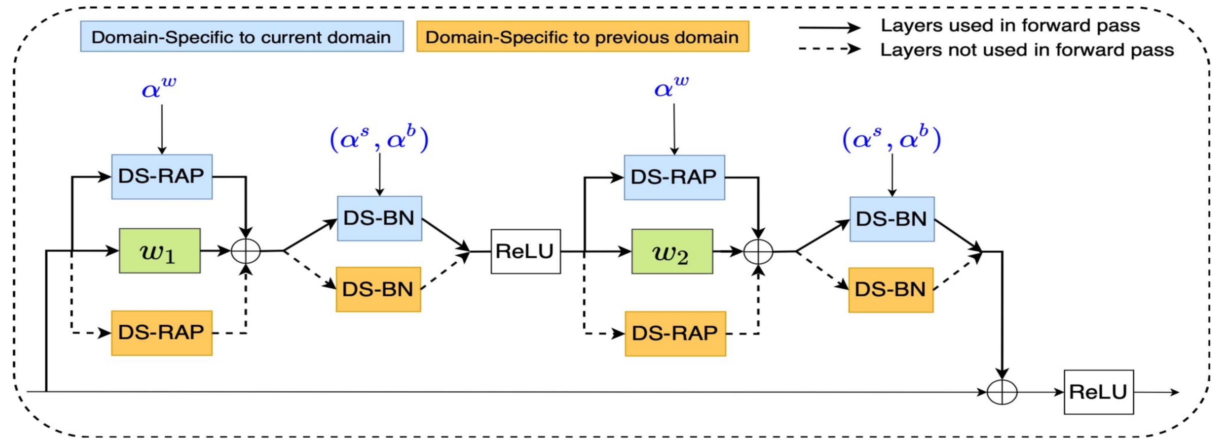
Challenging Objectives Achieved

- Learn new domain, while mitigating forgetting on previous domain:
  - Towards a **stability-plasticity trade-off**
- **DOMAIN SHIFT + SEMANTIC SHIFT** (label spaces can be non-overlapping)
- Learn **without storing** any old samples (without replay based approaches)

Architecture Overview



DAU: Domain-Aware Residual Unit



$$L_{CE_t} = \frac{1}{N} \sum_{x_t \epsilon \mathcal{D}_t} \psi_t(y_t, \mathcal{G}_t(\mathcal{F}(x_t, t; \mathcal{W}_s, \alpha_t))$$

$$L_{KLD} = \lambda_{KLD} \cdot \sum_{i=1}^{t-1} \sum_{x_t \epsilon \mathcal{D}_t} \phi(q_i^s, q_i^t)$$

$$L_{W_s} = L_{CE_t} + L_{KLD}$$

- Network reparameterization into domain-specific (DS) layers and shared (domain-invariant) layers.
- **Shared layers** capture homogeneous semantic features common among domains.
  - Fine-tuned on L_{W_S} using differential learning rates
- **Domain-specific layers** capture heterogeneous dataset statistics.
  - Trained only on L_{CE}

Contributions and Potential Use Cases


Continual, life-long learning

First real-world benchmark for domain incremental semantic segmentation


Autonomous Driving

Extensible model that is capable of learning across new domains, as and when the data is collected

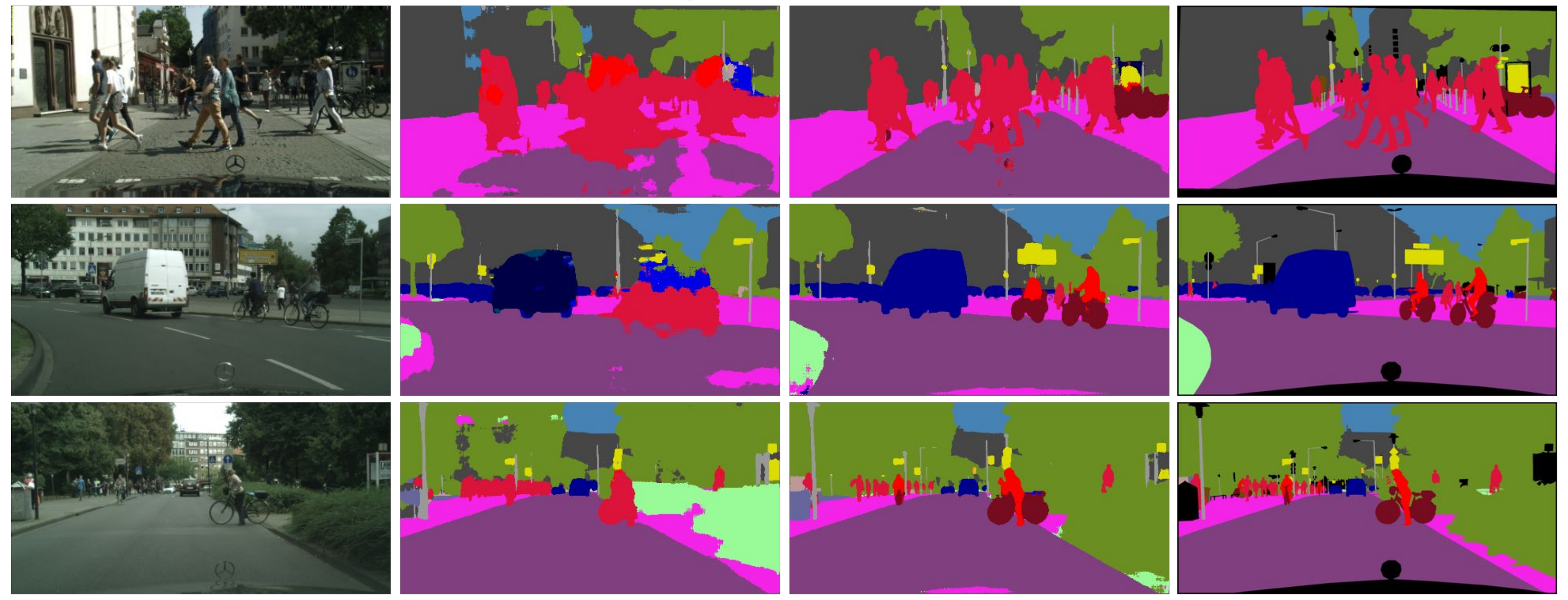

Semantic Segmentation

Studies transfer learning for cross-domain semantic segmentation

How is our setting different from existing semantic segmentation settings?

Problem Setting	Sequential	Differences, Source vs target: Label Space	Differences, Source vs target: Domain Shift	Data (availability, supervision): Source	Data (availability, supervision): Target	Goals	Solution Type: Task-Aware	Solution Type: Multi-Head
UDA	✓	same	✓	✓	✓ (unlabeled)	learn new	×	×
Class-IL	✓	different	×	×	✓	retain old, learn new	×	×
MDL	×	different	✓	✓	✓	retain all	✓	✓
MDIL (*ours*)	✓	different	✓	×	✓	retain old, learn new	✓	✓

Results And Analysis

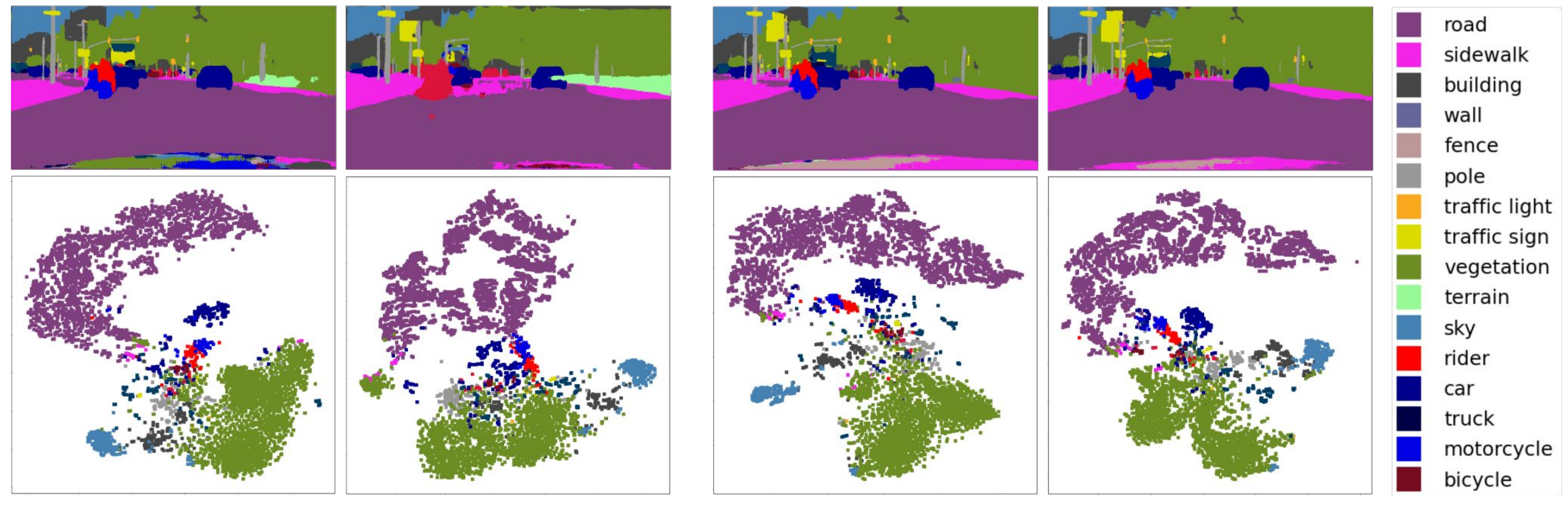


CS (Step 1) → BDD (Step 2): Results on Cityscapes *after* incrementally learning BDD100k: Our model mitigates forgetting by 25.16% as compared to Fine-tuning baseline

IL Step	Step 1 CS	Step 2: $D_A \neq D_B, \mathcal{Y}_A = \mathcal{Y}_B$ $CS \rightarrow BDD$			Step 2: $D_A \neq D_B, \mathcal{Y}_A \neq \mathcal{Y}_B$ $CS \rightarrow IDD$		
Methods	CS ↑	CS ↑	BDD ↑	Δ_m% ↓	CS ↑	IDD ↑	Δ_m% ↓
Single-task	72.55	72.55	54.1		72.55	61.97	
Multi-task	72.55	69.42	57.69	1.16% (↑)	71.11	60.85	1.89%
FT	72.55	40.05 (-32.5)	52.74	23.66%	36.81 (-35.74)	61.56	24.96%
FE	72.55	72.55 (-0.00)	42.93	10.32%	72.55 (-0.00)	45.69	13.14%
Ours	71.82	65.21 (-7.34)	55.73 (+1.63)	**3.55%**	64.58 (-7.97)	59.11 (-2.86)	**7.80%**

IL Step	Step 3: $D_A \neq D_B, \mathcal{Y}_A \neq \mathcal{Y}_B$ $CS \rightarrow BDD \rightarrow IDD$			
Methods	CS ↑	BDD ↑	IDD ↑	Δ_m% ↓
Single-task	72.55	54.1	61.97	
Multi-task	69.37	58.13	59.37	0.38%
FT	30.49 (-42.06)	32.05 (-22.05)	60.65	33.62%
FE	72.55 (-0.00)	42.93 (-11.17)	46.09	15.42%
Ours	59.19 (-13.36)	49.66 (-4.44)	59.16	**10.39%**

- **Forgetting is mitigated** significantly
- When the label spaces are same, **forward transfer** of knowledge achieved.
- When label spaces are different, there can be **domain interference.**



(a) Fine-tuning (b) Ours

CS→BDD: A comparison of CS latent space before and after learning BDD. Our model preserves the latent space of previous domain, which gets distorted during fine-tuning.