Domain Adaptation for Structured Output via Discriminative Patch Representations, ICLR'19

Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, Manmohan Chandraker

presented by Lennart Van der Goten

Domain Adaptation

Supervised Learning:

- Let $\mathcal{X}$ denote some input space and $\mathcal{Y}$ some output space and that we are interested in fitting a classifier $\eta : \mathcal{X} \to \mathcal{Y}$ on some finite-sized sample $S \subset \mathcal{X} \times \mathcal{Y}$ of some underlying (joint) distribution $\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$ defined on $\mathcal{X} \times \mathcal{Y}$.
- Find η that minimizes expected loss w.r.t to some loss function $\ell(\cdot,\cdot)$ and $\mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$
- *Common assumption*: $s_1, \ldots, s_m \overset{\text{i.i.d.}}{\sim} \mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$, to enable learning on finite-sized samples

Domain Adaption

- We assume that we are ultimately interested in applying η on a different (but related) distribution $\mathcal{P}'_{\mathcal{X}\times\mathcal{Y}}$
- Sampling $s'_1, \ldots, s'_m \overset{\text{i.i.d.}}{\sim} \mathcal{P}'_{\mathcal{X}\times\mathcal{Y}}$ and training η directly is presumed to be difficult, due to:
  - Complicated acquisition protocol
  - Scarcity of labeled data
  - Label noise
- **Idea**: Also use samples from $\mathcal{P}'_{\mathcal{X}\times\mathcal{Y}}$ in training process to accomplish objective

Domain Adaption [cont.]

Assume $S = \{s_1, \ldots, s_n\} \sim \mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$ resp. $S' = \{s'_1, \ldots, s'_m\} \sim \mathcal{P}'_{\mathcal{X}\times\mathcal{Y}}$

Disciplines:

- Unsupervised: S is labeled, S' is not
- Semi-supervised: S is labeled, S' contains a few labeled examples
- Supervised: Both S and S' are labeled

Common Representation Space Architectures

- Most common case (for $F^{(n)} = F_n \circ \ldots \circ F_1$):
    1. Fix some $i \in \{1, \ldots, n\}$
    2. Define random variables $Z = F^{(i)}(X)$ and $Z' = F^{(i)}(X')$ for $X \sim \mathcal{P}_{\mathcal{X}}$ resp. $X' \sim \mathcal{P}'_{\mathcal{X}}$
    3. Incentivize Z and Z' to become indistinguishable
- **Adversarial learning**: Let discriminator decide whether output of $F^{(i)}$ comes from source or target distribution

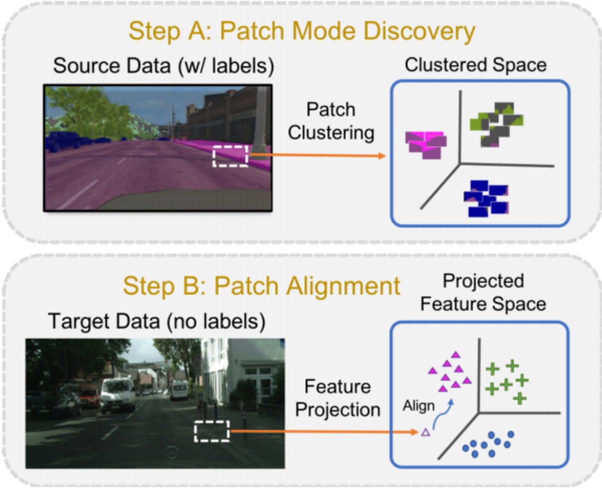
Motivation of Paper

- **Task**: Semantic segmentation
  - Source: `GTA-V`, `SYNTHIA`
  - Target: `CityScapes`, `Oxford-RobotCar`
- **Main challenge** (Tran *et al.*, 2019): It is difficult to capture all modes of the data distribution
- **Authors' hypothesis:** Discriminators that have not learned to capture a majority of the data distribution's modes can only evaluate low-level differences
- **Suggested solution**: Enforce the discriminator to learn *many* modes in an *unsupervised* manner
- **Implication**: Novel discriminator can be plugged into other architectures to enhance their capabilities

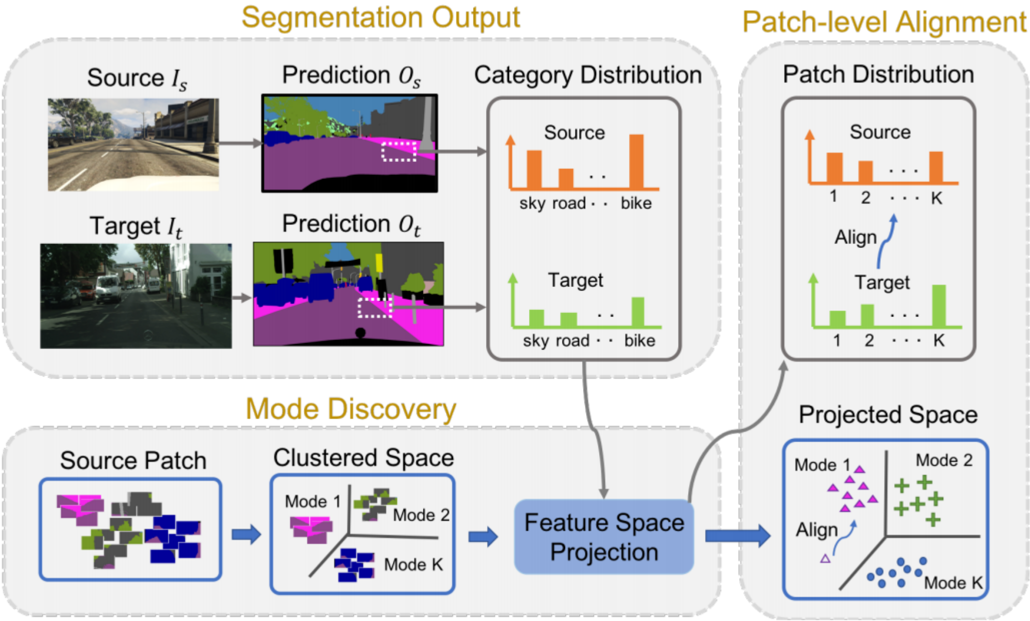
Related Work

- Segmentation:
  - Pixel-Level:
    1. **CyCADA**, *Hoffman et al.*, 217: Leverages *CycleGAN* to align domains on the pixel-level. Trained encoder-decoder pairs can then translate between domains.
  - Feature-Level:
    1. **FCNs in the Wild**, *Hoffman et al.*, 2016: Both feature-level & category-specific alignment
  - Output-Level:
    1. **ROAD**, *Chen et al.*, 2018: Model-destillation driven approach where the segmentation in the target domain should follow the one of a pre-trained network
    2. **OUTPUT SPACE**, *Tsai et al.*, 2018: Aligns probability *output* distributions between source and target domain
  - Pseudo-Label Re-Training:
    1. **CBST**, *Zou et al.*, 2018: Alternate between generating pseudo-labels on target data and re-training the network using these newly-generated labels. No adversarial training. Pseudo-labels are target-domain decisions that the net is confident about.

Idea


Step A: Patch Mode Discovery
Source Data (w/ labels)
Clustered Space
Patch
Clustering
Step B: Patch Alignment
Projected
Feature Space
Target Data (no labels)
Feature
Projection
Align

Overview


Segmentation Output
Source I_s
Prediction O_s
Target I_t
Prediction O_t
Category Distribution
Source
sky road · · bike
Target
sky road · · bike
Patch-level Alignment
Patch Distribution
Source
1 2 · · · K
Align
Target
1 2 · · · K
Mode Discovery
Source Patch
Clustered Space
Mode 1
Mode 2
Mode K
Feature Space Projection
Projected Space
Mode 1
Mode 2
Align
Mode K

Patch Mode Discovery

Suppose that $h \in \mathbb{R}^{(2h)\times(2w)\times C}$ denotes some arbitrary (one-hot encoded) patch extracted from the ground-truth segmentation (covering C classes) of a **training** set image:

- Mode discovery typically requires a supervised setting and thus labels
- How to get **one** label for **each** patch?
  - Unsupervised representation learning: Does not guarantee a semantic separation of patches
  - **Histogram-based**:
    1. Partition h into 2×2 grid of vectors $\begin{bmatrix} \mathbf{h}_{1,1} & \mathbf{h}_{1,2} \\ \mathbf{h}_{2,1} & \mathbf{h}_{2,2} \end{bmatrix}$ along height and width dimension
    2. Compute normalized histogram $\boldsymbol{\xi}_{i,j} \in [0,1]^C$ for each $\mathbf{h}_{i,j} \in \mathbb{R}^{h\times w\times C}$
    3. Gather histograms to get $\xi = \begin{bmatrix} \boldsymbol{\xi}_{1,1} & \boldsymbol{\xi}_{1,2} \\ \boldsymbol{\xi}_{2,1} & \boldsymbol{\xi}_{2,2} \end{bmatrix} \in [0,1]^{2\times 2\times C}$
- Perform `K-Means` on resulting histograms to discover K modes
- Map each patch to index of closest cluster
  - If we have a label map Y_s of $u \times v$ patches we get a new cluster-indexed map $\Gamma(Y_s) \in \{0, \ldots, K-1\}^{u\times v}$
  - We add a classification head that acts on the segmentation output and tries to predict the logits $F_s \in \mathbb{R}^{u\times v\times K}$ of the **patches' cluster indices** (i.e. $\Gamma(Y_s)$)

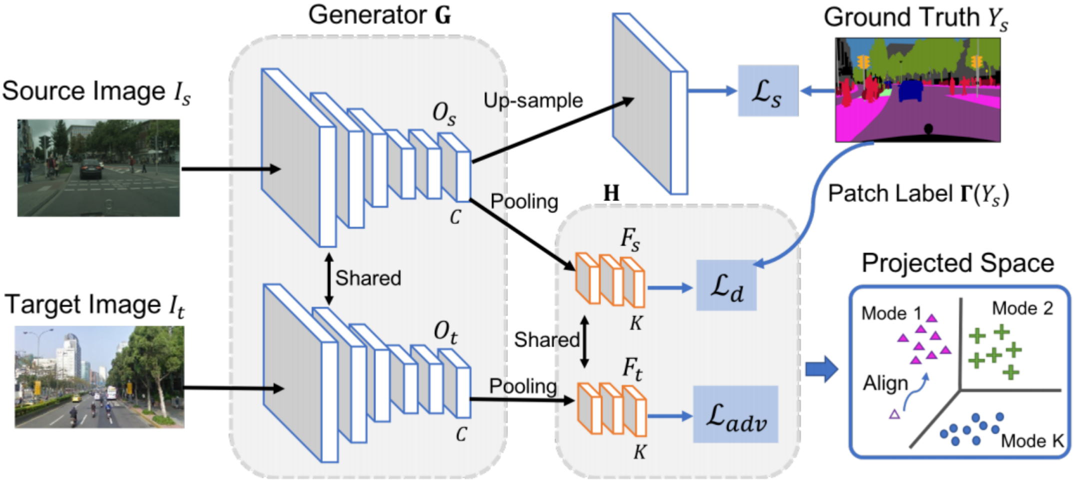
Adversarial Alignment

- **What we have now**:
  - A clustered space (of dim. C) as well as K modes of the patches of the label maps
  - A K-dimensional space where the i-th coordinate is equal to one whenever the i-th cluster is the closest one
- **Idea**: Align the representations of the target patches with the K modes of the clustered space $\implies$ K-ary classification
- Let $F_t \in \mathbb{R}^{u \times v \times K}$ denote the predicted logits from the classification head (as defined on the last slide) for some **target** example:
  - Give F_s and F_t to a discriminator and let it decide from where the examples are coming from (i.e. source vs. target)
- Loss components:
  - Segmentation loss: $\mathcal{L}_s$
  - Patch cluster index loss: $\mathcal{L}_d$ (K-way cross-entropy)
  - Adversarial *source vs. target* loss: $\mathcal{L}_{\mathrm{adv}}$ (binary cross-entropy)
  - Combine linearly w.r.t weights $\lambda_d, \lambda_{\mathrm{adv}}$

Network Architecture

- Segmentation network: `DeepLab-v2` w. `ResNet-101`
- Patch cluster-index network:
  - Gets the output $G(I) \in \mathbb{R}^{H \times W \times C}$ of the segmentation network as input
  - Has to output logits of shape $U \times V \times K$
  - Use *global average pooling* to get intermediate size $U \times V \times C$
  - Apply two conv. layers where the last layer produces K output channels
- Discriminator
  - Input is of shape K
  - `MLP` of three layers (256, 512, 1) w. `Leaky-ReLU` activations

Overview


Generator $\mathbf{G}$
Source Image I_s
Target Image I_t
O_s
C
Shared
O_t
C
Up-sample
Pooling
Pooling
Ground Truth Y_s
$\mathcal{L}_s$
Patch Label $\mathbf{\Gamma}(Y_s)$
$\mathbf{H}$
F_s
K
Shared
F_t
K
$\mathcal{L}_d$
$\mathcal{L}_{adv}$
Projected Space
Mode 1
Mode 2
Align
Mode K

Implementation Details

- **Optimizer**:
  - Discriminator: `Adam`
    - Initial learning rate: 10^{-4}
    - Momentum: 0.99
  - Generator: `SGD`
    - Initial learning rate: $2.5 \cdot 10^{-4}$
    - Momentum: 0.9
    - Weight decay: $5 \cdot 10^{-4}$
  - Learning rate schedule: Polynomial decay ($\alpha = 0.9$)
- **Hyperparameters**:
  - Patch cluster-index loss: $\lambda_d = 10^{-2}$
  - Adversarial loss: $\lambda_{\text{adv}} = 5 \cdot 10^{-4}$
  - **Number of clusters**: $K = 50$

Experiments

Datasets

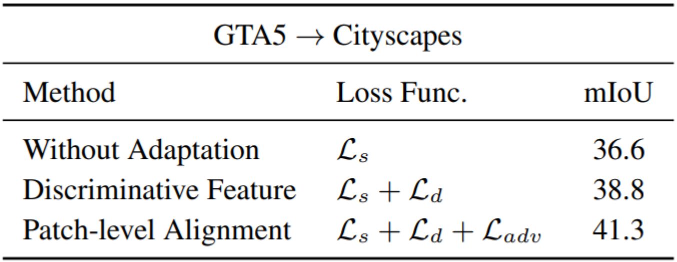
- `GTA-V`: Car rides extracted from computer game *GTA-V* (synthetic)
- `Cityscapes`: Real road-scene images [**labeled**]
- `SYNTHIA`: Frames are rendered given a highly-realistic computer-generated city
- `Oxford RobotCar`: Contains 100 repetitions of a consistent route through Oxford, UK, captured over a period of over a year [**unlabeled**]

Evaluation

- Intersection-over-Union [IoU]

Ablation Study

Loss Functions



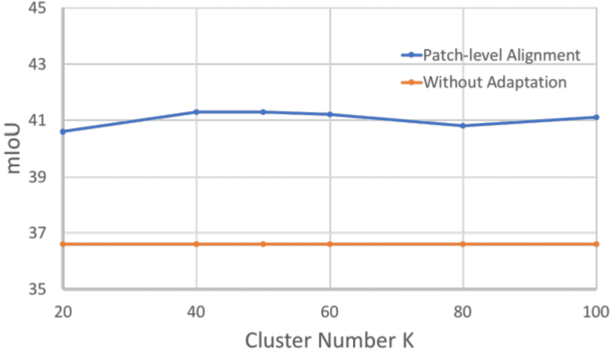
	GTA5 → Cityscapes	
Method	Loss Func.	mIoU
Without Adaptation	$\mathcal{L}_s$	36.6
Discriminative Feature	$\mathcal{L}_s + \mathcal{L}_d$	38.8
Patch-level Alignment	$\mathcal{L}_s + \mathcal{L}_d + \mathcal{L}_{adv}$	41.3

Outcome

Performance increases consistently

Experiments

Impact of Cluster Number *K*

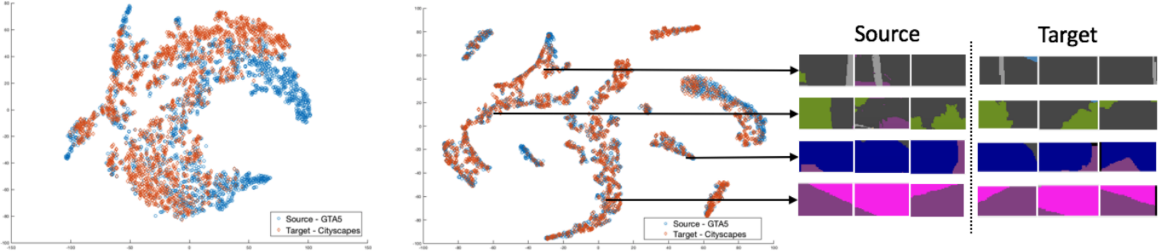


Outcome

No noticeable effect

t-SNE

Impact of Cluster Number *K*



Without Patch-level Alignment

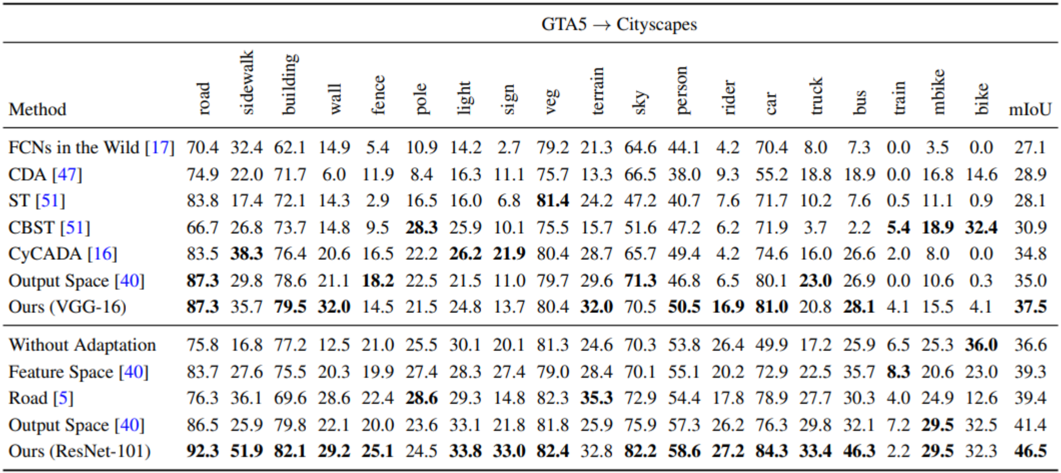
Our Method

Outcome

Suggested method enables a good source/target overlap while reference method does not

SOTA-Comparisons

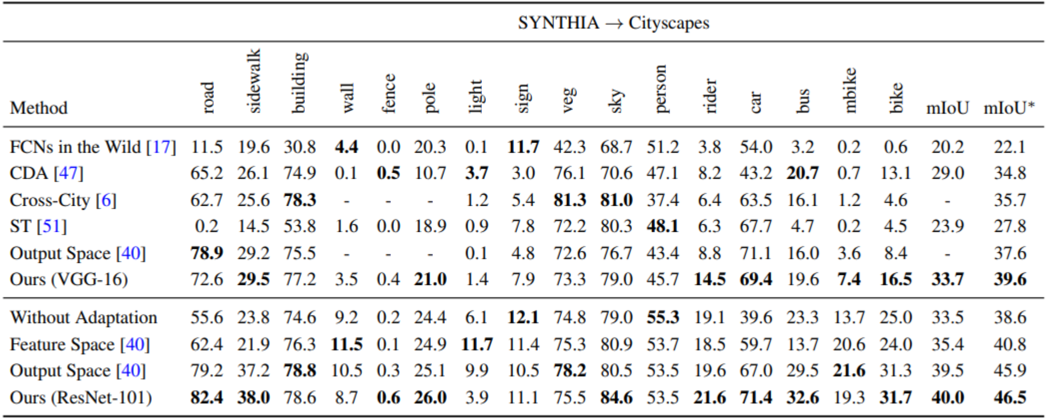
GTA-V:



	GTA5 → Cityscapes																			
Method	road	sidewalk	building	wall	fence	pole	light	sign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
FCNs in the Wild [17]	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0	27.1
CDA [47]	74.9	22.0	71.7	6.0	11.9	8.4	16.3	11.1	75.7	13.3	66.5	38.0	9.3	55.2	18.8	18.9	0.0	16.8	14.6	28.9
ST [51]	83.8	17.4	72.1	14.3	2.9	16.5	16.0	6.8	**81.4**	24.2	47.2	40.7	7.6	71.7	10.2	7.6	0.5	11.1	0.9	28.1
CBST [51]	66.7	26.8	73.7	14.8	9.5	**28.3**	25.9	10.1	75.5	15.7	51.6	47.2	6.2	71.9	3.7	2.2	**5.4**	**18.9**	**32.4**	30.9
CyCADA [16]	83.5	**38.3**	76.4	20.6	16.5	22.2	**26.2**	**21.9**	80.4	28.7	65.7	49.4	4.2	74.6	16.0	26.6	2.0	8.0	0.0	34.8
Output Space [40]	**87.3**	29.8	78.6	21.1	**18.2**	22.5	21.5	11.0	79.7	29.6	**71.3**	46.8	6.5	80.1	**23.0**	26.9	0.0	10.6	0.3	35.0
Ours (VGG-16)	**87.3**	35.7	**79.5**	**32.0**	14.5	21.5	24.8	13.7	80.4	**32.0**	70.5	**50.5**	**16.9**	**81.0**	20.8	**28.1**	4.1	15.5	4.1	**37.5**
Without Adaptation	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	**36.0**	36.6
Feature Space [40]	83.7	27.6	75.5	20.3	19.9	27.4	28.3	27.4	79.0	28.4	70.1	55.1	20.2	72.9	22.5	35.7	**8.3**	20.6	23.0	39.3
Road [5]	76.3	36.1	69.6	28.6	22.4	**28.6**	29.3	14.8	82.3	**35.3**	72.9	54.4	17.8	78.9	27.7	30.3	4.0	24.9	12.6	39.4
Output Space [40]	86.5	25.9	79.8	22.1	20.0	23.6	33.1	21.8	81.8	25.9	75.9	57.3	26.2	76.3	29.8	32.1	7.2	**29.5**	32.5	41.4
Ours (ResNet-101)	**92.3**	**51.9**	**82.1**	**29.2**	**25.1**	24.5	**33.8**	**33.0**	**82.4**	32.8	**82.2**	**58.6**	**27.2**	**84.3**	**33.4**	**46.3**	2.2	**29.5**	32.3	**46.5**

SOTA-Comparisons

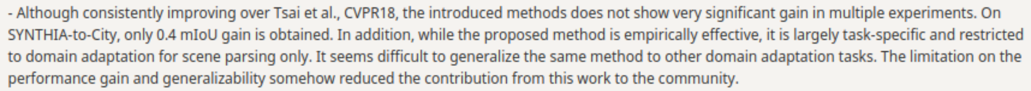
SYNTHIA:



	SYNTHIA → Cityscapes																	
Method	road	sidewalk	building	wall	fence	pole	light	sign	veg	sky	person	rider	car	bus	mbike	bike	mIoU	mIoU*
FCNs in the Wild [17]	11.5	19.6	30.8	**4.4**	0.0	20.3	0.1	**11.7**	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6	20.2	22.1
CDA [47]	65.2	26.1	74.9	0.1	**0.5**	10.7	**3.7**	3.0	76.1	70.6	47.1	8.2	43.2	**20.7**	0.7	13.1	29.0	34.8
Cross-City [6]	62.7	25.6	**78.3**	-	-	-	1.2	5.4	**81.3**	**81.0**	37.4	6.4	63.5	16.1	1.2	4.6	-	35.7
ST [51]	0.2	14.5	53.8	1.6	0.0	18.9	0.9	7.8	72.2	80.3	**48.1**	6.3	67.7	4.7	0.2	4.5	23.9	27.8
Output Space [40]	**78.9**	29.2	75.5	-	-	-	0.1	4.8	72.6	76.7	43.4	8.8	71.1	16.0	3.6	8.4	-	37.6
Ours (VGG-16)	72.6	**29.5**	77.2	3.5	0.4	**21.0**	1.4	7.9	73.3	79.0	45.7	**14.5**	**69.4**	19.6	**7.4**	**16.5**	**33.7**	**39.6**
Without Adaptation	55.6	23.8	74.6	9.2	0.2	24.4	6.1	**12.1**	74.8	79.0	**55.3**	19.1	39.6	23.3	13.7	25.0	33.5	38.6
Feature Space [40]	62.4	21.9	76.3	**11.5**	0.1	24.9	**11.7**	11.4	75.3	80.9	53.7	18.5	59.7	13.7	20.6	24.0	35.4	40.8
Output Space [40]	79.2	37.2	**78.8**	10.5	0.3	25.1	9.9	10.5	**78.2**	80.5	53.5	19.6	67.0	29.5	**21.6**	31.3	39.5	45.9
Ours (ResNet-101)	**82.4**	**38.0**	78.6	8.7	**0.6**	**26.0**	3.9	11.1	75.5	**84.6**	53.5	**21.6**	**71.4**	**32.6**	19.3	**31.7**	**40.0**	**46.5**

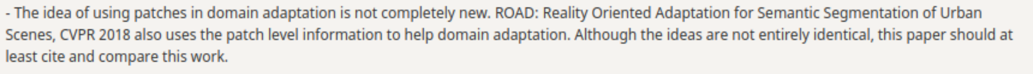
Open Review Opinions

- Most reviewers emphasize that the paper is clearly structured and technically sound
- Some criticize that the related-work section is incomplete
- Others state that the results are not good enough:

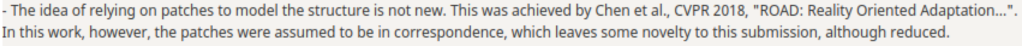


- Although consistently improving over Tsai et al., CVPR18, the introduced methods does not show very significant gain in multiple experiments. On SYNTHIA-to-City, only 0.4 mIoU gain is obtained. In addition, while the proposed method is empirically effective, it is largely task-specific and restricted to domain adaptation for scene parsing only. It seems difficult to generalize the same method to other domain adaptation tasks. The limitation on the performance gain and generalizability somehow reduced the contribution from this work to the community.

- Two reviewers criticize that the patch-level alignment idea is not entirely new:



- The idea of using patches in domain adaptation is not completely new. ROAD: Reality Oriented Adaptation for Semantic Segmentation of Urban Scenes, CVPR 2018 also uses the patch level information to help domain adaptation. Although the ideas are not entirely identical, this paper should at least cite and compare this work.



- The idea of relying on patches to model the structure is not new. This was achieved by Chen et al., CVPR 2018, "ROAD: Reality Oriented Adaptation...". In this work, however, the patches were assumed to be in correspondence, which leaves some novelty to this submission, although reduced.

- Most of the authors did not like the presumptuous use of the word *disentanglement*:
  - Was removed from the final version eventually and renamed to "discriminative" (e.g. in the title)

Conclusion

- Novelty:
  - Patch-level alignment: Ablation study shows that this is an advancement in terms of performance albeit novelty might only be due to using `K-Means`
- Technically sound paper that features state-of-the-art performance on common datasets
- Using `K-Means` to structure the latent space is an interesting idea
- There is no implementation available