A Domain Agnostic Normalization Layer for Unsupervised Adversarial Domain Adaptation

Master's thesis

Rob Romijnders

Eindhoven, University of Technology

July 3, 2018

End goal

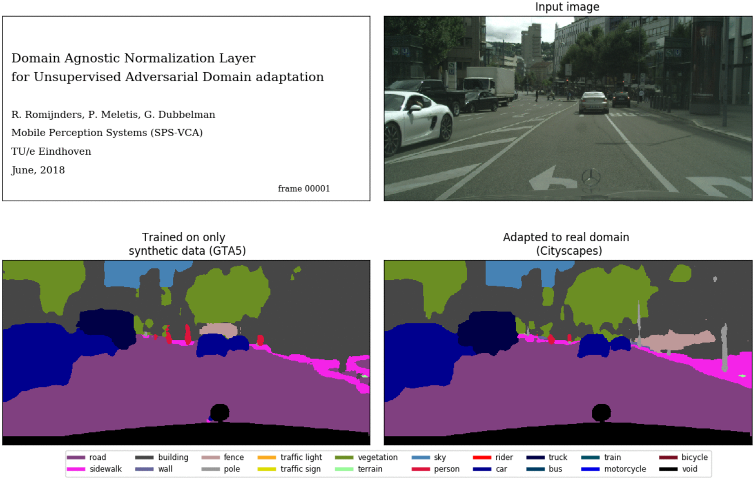
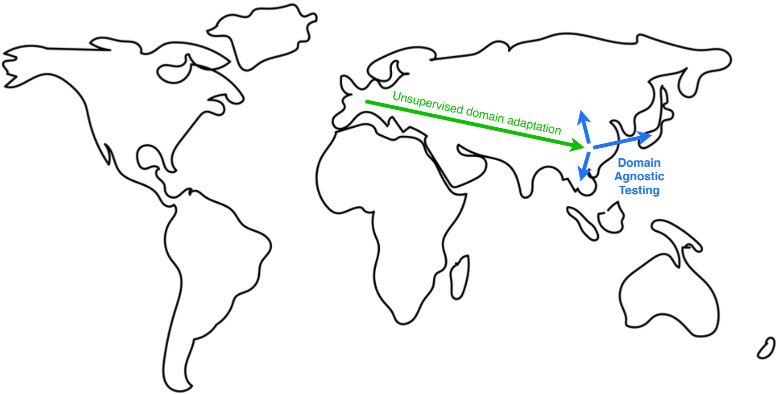


Figure: Animation requires media plugin (Use, for example, Adobe reader)

Outline

Outline

Domain Agnostic testing



Alleviating labeling cost

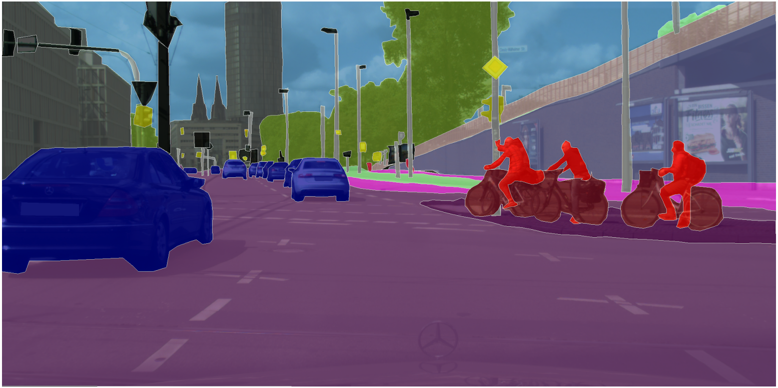


Figure: Labeling an image requires on average 90 minutes in the Cityscapes data set [3]

Leverage synthetic data

Figure: Example of GTA5 [13]

For example, a model trained on synthetic achieves 62.1% mIOU, but achieves 34.8% mIOU on real data.

Summary of motivation

Unsupervised domain adaptation

- Domain agnostic testing
- Reduce labeling burden
- Leverage synthetic data

Outline

Learning representations

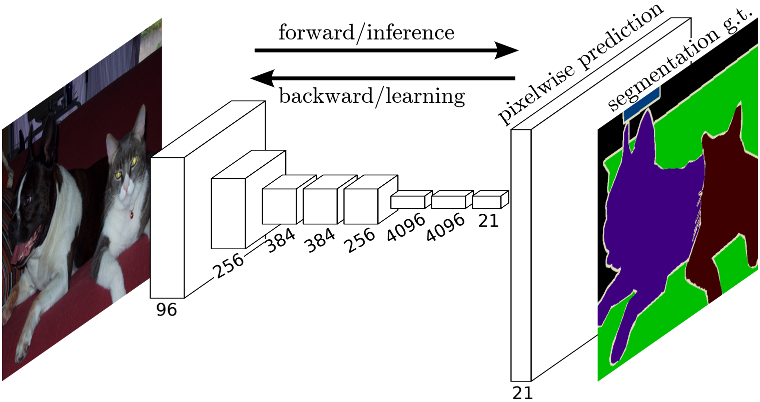


Figure: Adapted from [11]

Stochastic Gradient Descent

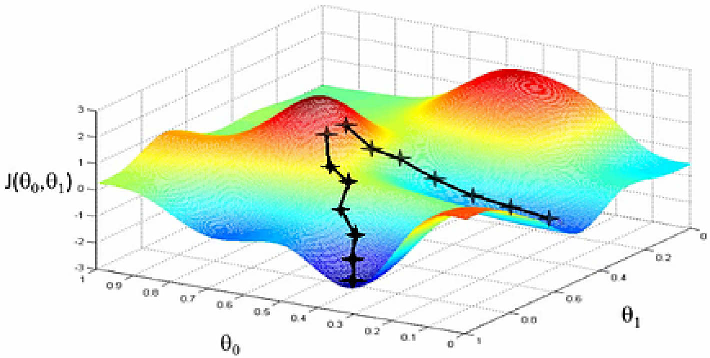
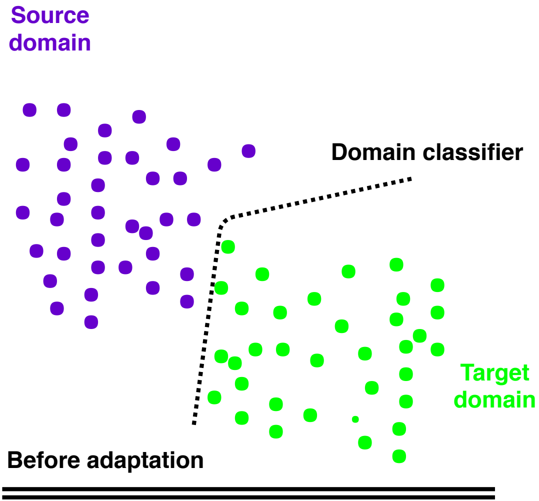
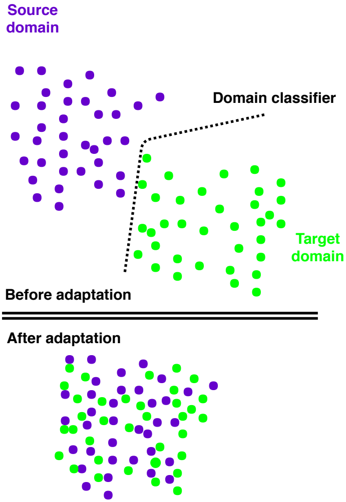


Figure: source-link

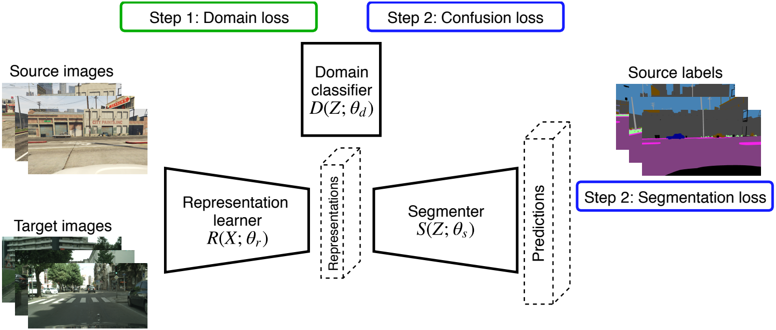
Adversarial training



Align representations



UADA overview



Batch normalization

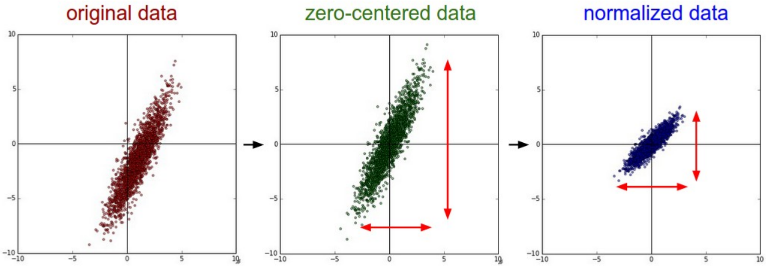
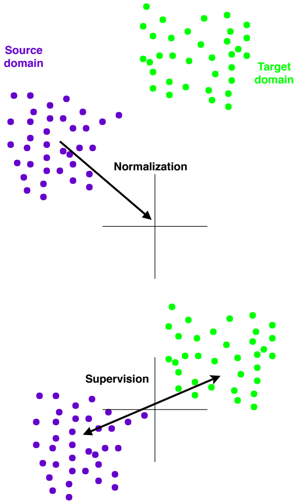


Figure: Explanation of batch normalization [9]

Problem with batch normalization in UADA



Requirements

Goal: Unlock the benefits of batch normalization for UADA

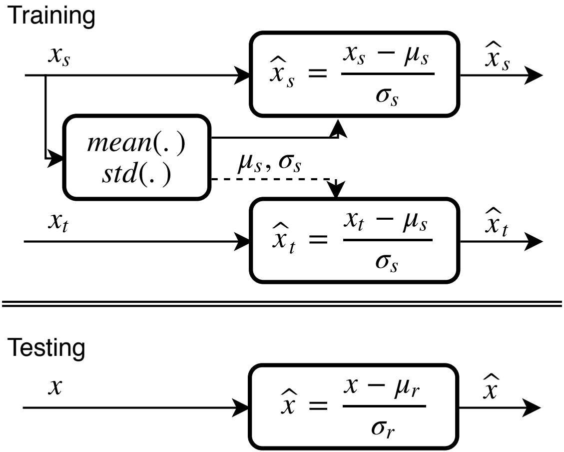
1. No dependency

A normalization introduces no dependency across domains.

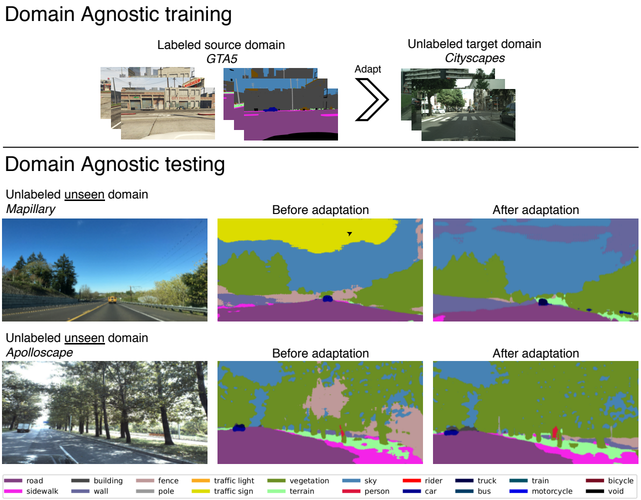
2. Same transformation

A normalization layer applies the same transformation on any domain.

DAN


Training
x_s
$\hat{x}_s = \frac{x_s - \mu_s}{\sigma_s}$
$\hat{x}_s$
$mean(.)$
$std(.)$
μ_s, σ_s
x_t
$\hat{x}_t = \frac{x_t - \mu_s}{\sigma_s}$
$\hat{x}_t$
Testing
x
$\hat{x} = \frac{x - \mu_r}{\sigma_r}$
$\hat{x}$

Overview


Domain Agnostic training
Labeled source domain
GTA5
Adapt
Unlabeled target domain
Cityscapes
Domain Agnostic testing
Unlabeled unseen domain
Mapillary
Before adaptation
After adaptation
Unlabeled unseen domain
Apolloscape
Before adaptation
After adaptation
road
sidewalk
building
wall
fence
pole
traffic light
traffic sign
vegetation
terrain
sky
person
rider
car
truck
bus
train
motorcycle
bicycle
void

Outline

Experiments

Normalization layers

- Batch normalization in multi-domain training
- Compare alternatives for DAN

UADA

- *GTA5* → *Cityscapes*
  - Compare to SOTA
  - Evaluate on unseen domain
- *Cityscapes* → *Mapillary*
  - Evaluate on unseen domain

Data sets

(a) GTA5 [13]

(b) Cityscapes [3]

(c) Mapillary [12]

(d) Apolloscape [8]

Figure: Data sets

Figure of merit

Mean intersection over union (% mIOU)

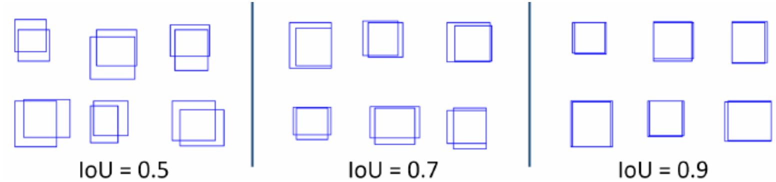
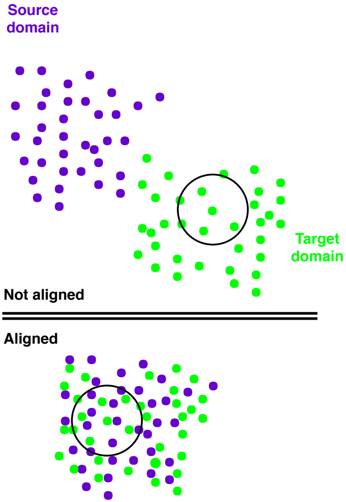
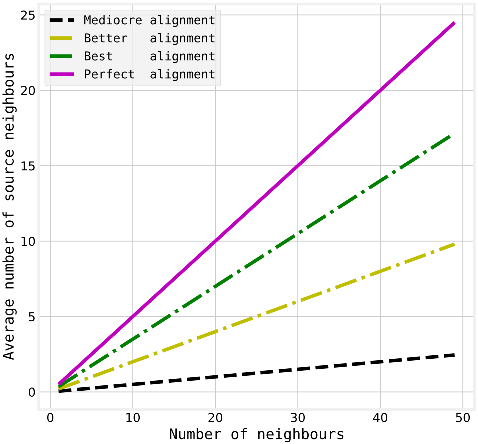


Figure: source

Explaining retrieval



Retrieval curve [14]



Outline

Batch normalization in multi-domain training

Table: **Analyzing batch normalization on multi-domain training.** The units are % mIOU. The *batch constituents* column indicates the domains of images in the batch, G → GTA5 and C → Cityscapes. In this experiment, we train models using only the GTA5 labels.

Normalization layer	Batch constituents	Tested on GTA5	Tested on Cityscapes
Batch norm	(G, G)		
Batch norm	(G, G, C, C)		
DAN	(G, G, C, C)		

Batch normalization in multi-domain training

Table: **Analyzing batch normalization on multi-domain training.** The units are % mIOU. The *batch constituents* column indicates the domains of images in the batch, G → GTA5 and C → Cityscapes. In this experiment, we train models using only the GTA5 labels.

Normalization layer	Batch constituents	Tested on GTA5	Tested on Cityscapes
Batch norm	(G, G)	62.1	34.8
Batch norm	(G, G, C, C)		
DAN	(G, G, C, C)		

Batch normalization in multi-domain training

Table: **Analyzing batch normalization on multi-domain training.** The units are % mIOU. The *batch constituents* column indicates the domains of images in the batch, G → GTA5 and C → Cityscapes. In this experiment, we train models using only the GTA5 labels.

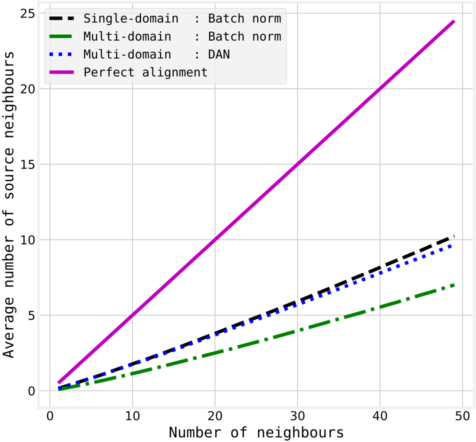
Normalization layer	Batch constituents	Tested on GTA5	Tested on Cityscapes
Batch norm	(G, G)	62.1	34.8
Batch norm	(G, G, C, C)	61.2	22.1
DAN	(G, G, C, C)		

Batch normalization in multi-domain training

Table: **Analyzing batch normalization on multi-domain training.** The units are % mIOU. The *batch constituents* column indicates the domains of images in the batch, G → GTA5 and C → Cityscapes. In this experiment, we train models using only the GTA5 labels.

Normalization layer	Batch constituents	Tested on GTA5	Tested on Cityscapes
Batch norm	(G, G)	62.1	34.8
Batch norm	(G, G, C, C)	61.2	22.1
DAN	(G, G, C, C)	61.9	35.0

Retrieval curve [14]



Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.			
Batch norm. [9]			
Split batch norm. [2]			
Instance norm. [16]			
DAN [ours]			

Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.	26.5	28.4	1.9
Batch norm. [9]			
Split batch norm. [2]			
Instance norm. [16]			
DAN [ours]			

Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.	26.5	28.4	1.9
Batch norm. [9]	**34.8**	29.6	-5.2
Split batch norm. [2]			
Instance norm. [16]			
DAN [ours]			

Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.	26.5	28.4	1.9
Batch norm. [9]	**34.8**	29.6	-5.2
Split batch norm. [2]	**34.8**	35.4	0.6
Instance norm. [16]			
DAN [ours]			

Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.	26.5	28.4	1.9
Batch norm. [9]	**34.8**	29.6	-5.2
Split batch norm. [2]	**34.8**	35.4	0.6
Instance norm. [16]	30.3	31.4	1.1
DAN [ours]			

Comparison of normalization layers

Table: **Comparing normalization layers on adaptation GTA5 → Cityscapes.** Numbers represent performance on the target domain. Gap reduction indicates the difference between source-only and UADA. Norm. abbreviation normalization. All units are % mIOU.

Normalization layer	Source only	UADA	Gap reduction
No norm.	26.5	28.4	1.9
Batch norm. [9]	**34.8**	29.6	-5.2
Split batch norm. [2]	**34.8**	35.4	0.6
Instance norm. [16]	30.3	31.4	1.1
DAN [ours]	**34.8**	**38.2**	**3.4**

Task 1: *GTA5* → *Cityscapes* (*synthetic* → *real*)

Task 2: *Cityscapes* → *Mapillary* (*real* → *real*)

UADA - source/target

Table: **GTA5 → Cityscapes**. First row, source, indicates a model trained on GTA5 only. We also report performance on two unseen domains, Mapillary and Apolloscape. All units are % mIOU.

GTA5 → Cityscapes				
	Tested on			
Method	GTA5 (*Source*)	Cityscapes (*Target*)	Mapillary (*Unseen*)	Apolloscape (*Unseen*)
Source	62.2	34.8		
UADA	**62.7**	**38.2**		

UADA - source/target

Table: **GTA5 → Cityscapes**. First row, source, indicates a model trained on GTA5 only. We also report performance on two unseen domains, Mapillary and Apolloscape. All units are % mIOU.

GTA5 → Cityscapes				
	Tested on			
Method	GTA5 (*Source*)	Cityscapes (*Target*)	Mapillary (*Unseen*)	Apolloscape (*Unseen*)
Source	62.2	34.8	37.1	25.3
UADA	**62.7**	**38.2**		

UADA - source/target

Table: **GTA5 → Cityscapes**. First row, source, indicates a model trained on GTA5 only. We also report performance on two unseen domains, Mapillary and Apolloscape. All units are % mIOU.

GTA5 → Cityscapes				
	Tested on			
Method	GTA5 (*Source*)	Cityscapes (*Target*)	Mapillary (*Unseen*)	Apolloscape (*Unseen*)
Source	62.2	34.8	37.1	25.3
UADA	**62.7**	**38.2**	**38.5**	**27.4**

UADA - SOTA

Table: **Comparison with three recent works in UADA.** The first column indicates at what level the adversary operates. Image size indicates number of pixels in height $\times$ width. Performance units are % mIOU.

GTA5 → Cityscapes			
Adversarial adaptation level	Norm. layer	Image Size	Target perf.
Representation [7]			
Representation and logit [17]			
Representation and output [14]			
Representation [ours]			

UADA - SOTA

Table: **Comparison with three recent works in UADA.** The first column indicates at what level the adversary operates. Image size indicates number of pixels in height × width. Performance units are % mIOU.

GTA5 → Cityscapes			
Adversarial adaptation level	Norm. layer	Image Size	Target perf.
Representation [7]	No		
Representation and logit [17]	No		
Representation and output [14]	No		
Representation [ours]	Yes		

UADA - SOTA

Table: **Comparison with three recent works in UADA.** The first column indicates at what level the adversary operates. Image size indicates number of pixels in height × width. Performance units are % mIOU.

GTA5 → Cityscapes			
Adversarial adaptation level	Norm. layer	Image Size	Target perf.
Representation [7]	No	x	
Representation and logit [17]	No	x	
Representation and output [14]	No	512 × 1024	
Representation [ours]	Yes	384 × 768	

UADA - SOTA

Table: **Comparison with three recent works in UADA.** The first column indicates at what level the adversary operates. Image size indicates number of pixels in height × width. Performance units are % mIOU.

GTA5 → Cityscapes			
Adversarial adaptation level	Norm. layer	Image Size	Target perf.
Representation [7]	No	x	25.5
Representation and logit [17]	No	x	28.9
Representation and output [14]	No	512×1024	37.1
Representation [ours]	Yes	384×768	**38.2**

UADA - source/target

Table: **Cityscapes** → **Mapillary**. The first row, source, indicates a model trained on Cityscapes only. We also report performance on an unseen domain, Apolloscape. All units are % mIOU.

Cityscapes → *Mapillary*			
	Tested on		
Method	Cityscapes (*Source*)	Mapillary (*Target*)	Apolloscape (*Unseen*)
Source	63.6	43.2	
UADA	**64.0**	**45.0**	

UADA - source/target

Table: **Cityscapes → Mapillary**. The first row, source, indicates a model trained on Cityscapes only. We also report performance on an unseen domain, Apolloscape. All units are % mIOU.

Cityscapes → Mapillary			
	Tested on		
Method	Cityscapes (*Source*)	Mapillary (*Target*)	Apolloscape (*Unseen*)
Source	63.6	43.2	25.8
UADA	**64.0**	**45.0**	

UADA - source/target

Table: **Cityscapes → Mapillary**. The first row, source, indicates a model trained on Cityscapes only. We also report performance on an unseen domain, Apolloscape. All units are % mIOU.

Cityscapes → Mapillary			
	Tested on		
Method	Cityscapes (*Source*)	Mapillary (*Target*)	Apolloscape (*Unseen*)
Source	63.6	43.2	25.8
UADA	**64.0**	**45.0**	**27.1**

Video of Stuttgart

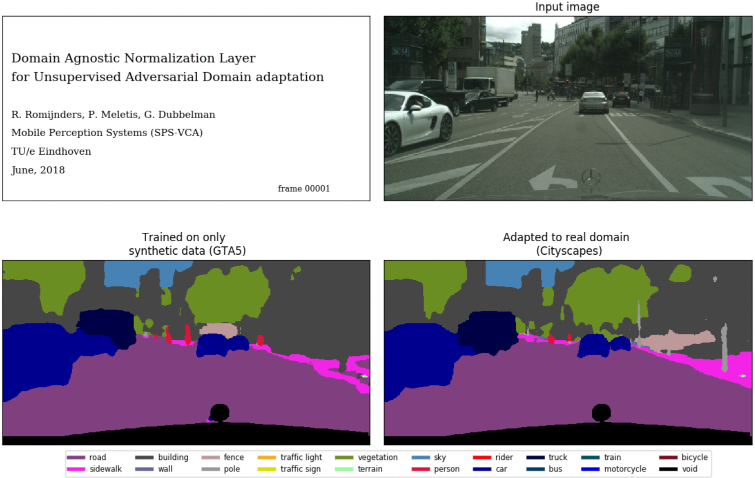


Figure: Animation requires media plugin (Use, for example, Adobe reader)

Video

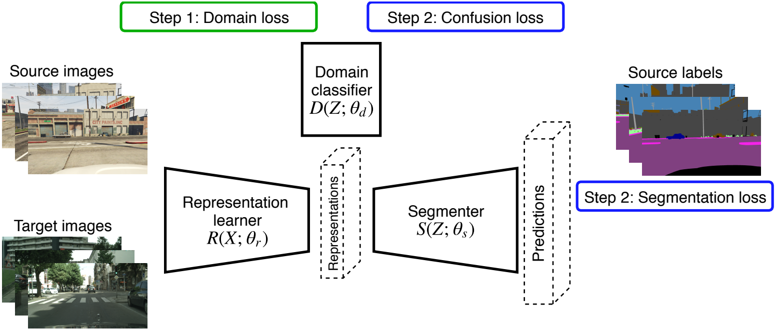
Outline

Hyperparameter

Table: **Hyperparameter analysis** Comparing target performance for ranging values of λ. Numbers represent target performance in units % mIOU

value of λ	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
GTA5 → *Cityscapes*		33.6	34.6	**38.2**	34.6
Cityscapes → *Mapillary*	42.6	43.0	**45.0**	37.6	

Level of adversary



Outline

Conclusion

1. UADA improves segmentation performance on a target domain, without requiring labels.
2. Batch normalization has benefits, but is incompatible with UADA
3. We propose DAN, compare favourably against alternatives and show compatibility with UADA
4. With UADA using DAN, we surpass state of the art and we show improvements on unseen domains.

Questions

References

[1] Johan Bjorck, Carla Gomes, and Bart Selman. Understanding Batch Normalization. *arXiv*, 2018.

[2] Soumith Chintala. How to train a GAN, Workshop on Adversarial Training at NIPS 2016, 2016.

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3213—-3223, apr 2016.

[4] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research (JMLR)*, 17(1):189–209, 2017.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014.

[6] E Hoffer, R Banner, I Golan, and D Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. *ArXiv*, 2018.

[7] J Hoffman, D Wang, F Yu, and T Darrell. FCNs in the Wild: Pixel-level Adversarial and Constraint-based Adaptation. *ArXiv*, dec 2016.

References (cont.)

[8] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The ApolloScape Dataset for Autonomous Driving. *arXiv*, 2018.

[9] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[10] J Kohler, H Daneshmand, A Lucchi, M Zhou, K Neymeyr, and T Hofmann. Towards a Theoretical Understanding of Batch Normalization. *ArXiv*, 2018.

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[12] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision (ICCV)*, pages 22–29, 2017.

[13] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In European Conference on Computer vision (ECCV), editor, *Lecture Notes in Computer Science*, volume 9906, pages 102–118. Springer, Cham, oct 2016.

References (cont.)

[14] S Sankaranarayanan, Y Balaji, A Jain, S.˜N. Lim, and R Chellappa. Unsupervised Domain Adaptation for Semantic Segmentation with GANs. *ArXiv*, nov 2017.

[15] S Santurkar, D Tsipras, A Ilyas, and A Madry. How Does Batch Normalization Help Optimization? (No, It Is Not About Internal Covariate Shift). *ArXiv*, 2018.

[16] D Ulyanov, A Vedaldi, and V Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization. *ArXiv*, 2016.

[17] Yang Zhang, Philip David, and Boqing Gong. Curriculum Domain Adaptation for Semantic Segmentation of Urban Scenes. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

Active research on batch normalization

- (03-2015) Batch normalization,
  Ioffe, Sergey; Szegedy, Christian [9]
- (03-2018) Norm matters,
  Hoffer, E; Banner, R; Golan, I; Soudry, D [6]
- (05-2018) How Does Batch Normalization Help Optimization?
  Santurkar, S; Tsipras, D; Ilyas, A; Madry, A [15]
- (05-2018) Towards a Theoretical Understanding of Batch Normalization,
  Kohler, J; ... Hofmann, T [10]
- (06-2018) Understanding Batch Normalization,
  Bjorck, Johan; Gomes, Carla; Selman, Bart [1]

Batch norm in 2D example

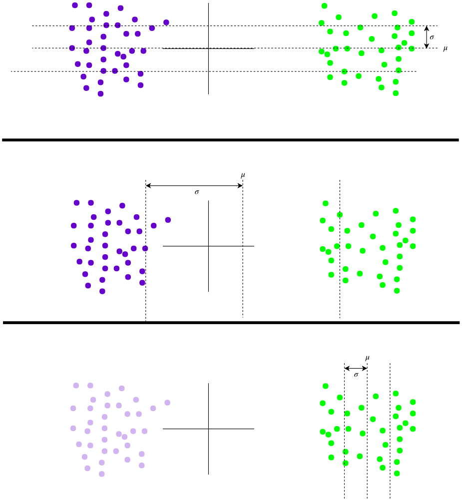
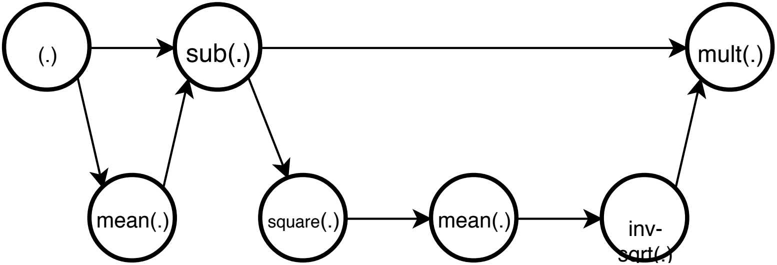
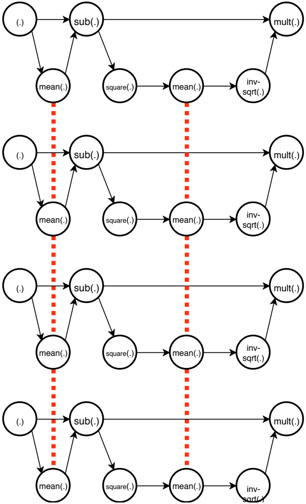


Figure: Considering batch norm in the multi-domain setting in 2D example

Comp. graph for batch normalization

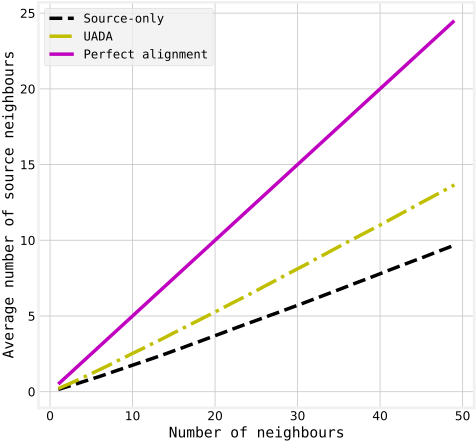


Ext. comp. graph for batch normalization



]

Retrieval curve [14]



Sizes

Effect of sizes

This slide displays **old results**.
First row reports on size 384×768
Second row reports on size 512×1024

Tested on Cityscapes			Tested on GTA5		
CS only	GTA5 only	GTA5 bn-none	CS only	GTA5 only	GTA5 bn-none
63.6	34.2	25.4	40.8	62.2	43.4
65.4	35.4	26.2	41.9	63.5	44.3
1.8	1.2	0.8	1.1	1.3	1.0

Difference between GAN [5] and Adversarial domain adaptation[4]

Distributions

GAN:

Data samples fix	Model samples change

Adversarial adaptation:

Source representations change	Target representations change

Intuition

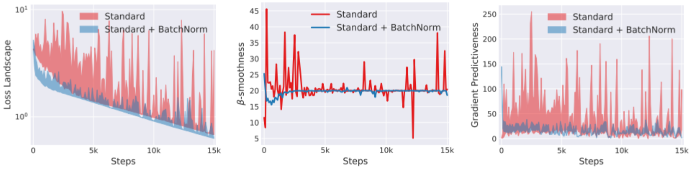
GAN	is like hide and seek
Adversarial adaptation	is like cat and mouse

Domain Classifier

Think of domain classifier like:

- Learning an adversary to the representation learning
- Parametric estimate of $\log \frac{p_z^{source}}{p_z^{target}}$

Batch normalization and the loss landscape



(a) loss landscape (b) "effective" β-smoothness (c) gradient predictiveness

Figure 4: Analysis of the optimization landscape of VGG networks. At a particular training step, we measure the variation (shaded region) in loss (a) and ℓ_2 changes in the gradient (c) as we move in the gradient direction. The "effective" β-smoothness (b) refers to the maximum β value observed while moving in this direction. There is a clear improvement in each of these measures of smoothness of the optimization landscape in networks with BatchNorm layers. (Here, we cap the maximum distance to be $\eta = 0.4\times$ the gradient since for larger steps the standard network just performs worse (see Figure 1). BatchNorm however continues to provide smoothing for even larger distances.)

Figure: Figure from [15]

Modeling assumptions

Order of assumptions we make:

1. We make an assumption on the segmentation model: CNN
2. We make an assumption that the target domain has no labels
3. We make an assumption on the influence of the target images: influence on the representations
4. We make an assumption on the representations: they should be confused between source and target
5. We make an assumption on the confusion: it follows from a domain classifier
6. We make an assumption on the domain classifier: it should be a neural network

DAN with learnable parameters