Published in Image Processing On Line on 2022–10–31.
Submitted on 2022–09–30, accepted on 2022–09–30.
ISSN 2105–1232 © 2022 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2022.433

A Study of RobustNet, a Domain Generalization Method for Semantic Segmentation

Xavier Bou

Centre Borelli, ENS Paris-Saclay, France
xavier.bou_hernandez@ens-paris-saclay.fr

Communicated by Jean-Michel Morel *Demo edited by* Xavier Bou

Abstract

Domain Generalization alleviates the domain gap between training set and test set, improving the performance of deep neural networks on out-of-dataset data. This opens the possibility of deploying models on unlabelled data that were previously pretrained on other datasets. In this article, we study the ideas and performance of RobustNet [Choi et al. CVPR 2021], a recent method for Domain Generalization in Urban-Scene Semantic Segmentation. Instead of exposing the network to a wide range of domains, RobustNet tries to separate domain-variant from domain-invariant features via a whitening transformation. Then, only the domain invariant features are used for training, which allows to reduce training time since no combination of datasets is needed to achieve domain invariance. In addition, we provide an easy-to-use demo where users can quickly test their own data and compare the results of RobustNet against the state of the art for semantic segmentation.

Source Code

The source code and documentation for this algorithm are available from the web page of this article[1]. Usage instructions are included in the README.md file of the archive. The original implementation of the method is available here[2].
This is an MLBriefs article, the source code has not been reviewed!

Keywords: computer vision; domain generalization; RobustNet; semantic segmentation; convolutional neural networks

[1] https://doi.org/10.5201/ipol.2022.433
[2] https://github.com/shachoi/RobustNet

1 Introduction

The cost and effort associated with building a large annotated dataset makes developers and organizations commonly deploy Deep Neural Networks (DNN) in real data, while training them on a specific dataset. Nevertheless, datasets will have a set of embedded characteristics given by the particularities of the acquisition process, such as the location where samples were taken, the weather conditions, the season of the year, etc. Additionally, the real-world test data can present domain-specific differences that might not be well represented in the dataset, such as illumination changes. This challenge, known as the domain shift problem, causes DNNs to decrease their performance when tested in out-of-dataset data, since the network will learn the domain particularities from the training dataset that will not apply in real-world data.

In the context of safety-critical applications, such as autonomous driving, solving the domain shift problem is vital. A common approach to overcome the domain gap between training data and real-world data is Domain Adaptation (DA). Given a source domain and a target domain, DA focuses on mapping the source distribution into the target distribution so that the same model trained on the source domain can be used on the target domain [5]. While multiple DA approaches have been proposed to decrease the impact of the domain shift problem [20, 5, 14, 16, 12], samples from the target domain are typically needed. This enforces the requirement of knowing in which domain will the network be deployed, something unpredictable in fields like autonomous driving. In cases where the target domain is the entire world, fully covering the target domain span with limited data is practically impossible.

Domain Generalization (DG) attempts to solve this without the DA limitations by improving the capability of neural networks to perform well in domains that have never been seen during training. The most common approach in DG is to leverage multiple source domains and learn from the features that are consistent across them. Li et al. [10] proposes a *learning to learn* approach for heterogeneous DG where an auxiliary loss term used for generalization is itself learned. Li et al. [7] designed an episodic training strategy for DG, where a DNN is decomposed into two partner components: feature extractor and classifier. Then, the components are paired with a domain-specific partner that is mismatched with the current data being input, leading to a domain-agnostic model. MMD-AAE [8], proposed by Haoliang et al, is an adversarial autoencoder-based method that aligns distributions from different domains using the Maximum Mean Discrepancy (MMD) measure, and then matches them to prior distributions via adversarial feature learning. A domain randomization strategy is used in [18], where, for every input, a number of domain shifts are simulated using CycleGAN [19] and a randomly selected reference image. Then, a pyramid consistency loss term is used to enforce these images with the same content but different domain to have similar representations at several layers of the network. While learning by exposing the network to a number of domains has proven to be effective and improving state-of-the art DG capabilities, these approaches require a large number of different, realistic domains. In addition, the performance depends on the amount of domains used at training.

This article reviews a recent DG method for semantic segmentation known as RobustNet [3], which proposes a different approach from the ones mentioned earlier. Instead of achieving domain-invariance by exposing the model to numerous domains, RobustNet tries to suppress the features that encode style information so that only content information is used for learning. Specifically, a training strategy with an instance selective whitening loss is proposed to discriminate domain-specific from domain-invariant properties of the feature representations, thus suppressing the domain-specific ones. The following sections discuss the method in detail, present some results and cover its limitations, and explain how to use our demo to test custom data and compare it to the baseline model for semantic segmentation.

2 Method

RobustNet separates the style information from the content information using an Instance Whitening Loss (IWL), and then suppresses the domain-variant terms so that only domain-invariant features are used for learning. Finally, this approach is embedded into a DNN architecture for semantic segmentation. Technical details of these three steps are covered in the following subsections.

2.1 Whitening Transformation and Standardized Covariance Matrix

The Whitening Transformation (WT) is a linear transformation that, given the covariance matrix of a feature map, makes the diagonal terms (variance term of each feature) equal to one and the off-diagonal terms (covariance between pairs of features) equal to zero [3]. The WT has been associated with the capability to remove style information from images in style transfer [9]. However, the WT is typically computed via eigenvalue decomposition, which is computationally expensive. For this reason, RobustNet proposes an Instance Whitening Loss inspired in GDWCT [2], which approximates the covariance matrix to the identity matrix. To do so, we need to minimize the off-diagonal elements of the covariance matrix of an input feature space, while making the diagonal terms equal to one. Let $x_i \in \mathbb{R}^{HW}$ be the i^{th} channel of the intermediate feature map $X \in \mathbb{R}^{C \times HW}$, where C, H and W are the number of channels, height and width, respectively. An instance normalization layer is first applied to standardize the intermediate feature map into X_s

$$X_s = diag(\Sigma_\mu))^{-0.5} \odot (X - \mu \cdot 1^T), \tag{1}$$

where $\odot$ is the element-wise multiplication, μ is the mean vector and $diag(\Sigma_\mu) \in \mathbb{R}^{C \times 1}$ corresponds to the diagonal elements of the covariance matrix. Then, the covariance matrix is computed as follows

$$\Sigma_s = \frac{1}{HW}(X_s)(X_s)^T \in \mathbb{R}^{C \times C}. \tag{2}$$

Due to the standardization step, the resulting covariance matrix Σ_s will have ones across its diagonal. During the first n epochs, no additional loss term is applied and this covariance matrix is used to capture the statistics of the variance of Σ_s. After n, which is a parameter set to 5, the following epochs add an Instance Selective Whitening (ISW) Loss, explained in more detail in the following section.

2.2 Instance Selective Whitening Loss

As mentioned earlier, RobustNet's goal is to separate the covariance matrix terms that encode domain information from the ones that encode content information, and suppress the first ones from the learning process. The approach proposed for this is to simulate an image domain shift by applying a photometric transformation, which consists in color jittering and Gaussian blurring. Then, the covariance matrix Σ_s of each of the two images is computed according to Equation (2). After that, the variance between the covariance matrices V for all image samples is computed as

$$V = \frac{1}{N}\sum_{i=1}^{N} \sigma_i^2, \tag{3}$$

from mean μ_{Σ_i} and variance σ_i^2, for each element from two different covariance matrices of the i^{th} image, i.e.

$$\mu_{\Sigma_i} = \frac{1}{2}\Big(\Sigma_s(x_i) + \Sigma_s(\tau(x_i))\Big), \tag{4}$$

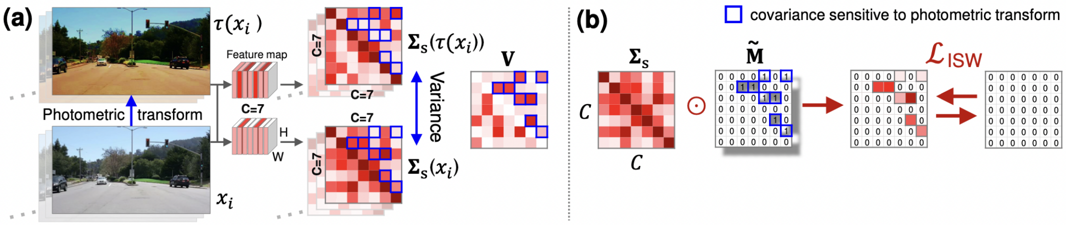


Figure 1: Image extracted from the original paper [3], which illustrates the computation of the ISW loss. (a) shows how the variance of the covariances matrix V is computed. (b) Shows how the covariance matrix σ_i^2 is masked to only suppress the domain-variant terms.

$$\sigma_i^2 = \frac{1}{2}\Big((\Sigma_s(x_i) - \mu_{\Sigma_i})^2 + (\Sigma_s(\tau(x_i)) - \mu_{\Sigma_i})^2\Big), \tag{5}$$

where N is the number of image samples, x_i is the i^{th} image sample, τ is a photometric transformation, and $\Sigma_s(\cdot)$ extracts the covariance matrix from an input image. The obtained V will therefore consist of elements of the variance of each covariance element with multiple photometric transformations.

We can consider V to express the sensitivity of the covariance to the photometric transformation. Thus, the covariance terms with high variance will encode domain-variant information, such as color or blurriness, while the ones with low variance will be attributed to domain-invariant information. To separate these terms, a k-means algorithm is applied to the strict upper triangular elements of $V_{i,j}$ $(i < j)$ to cluster the terms into k clusters $C = \{c_1, c_2, \dots, c_k\}$. Then, these clusters will be divided into two groups, $G_{low} = \{c_1, \dots, c_m\}$, containing the low variance terms, and $G_{high} = \{c_{m+1}, \dots, c_k\}$, which will include the high variance terms. The hyper-parameters k and m are empirically set to 3 and 1 in the original paper, respectively. Finally, an Instance Selective Whitening (ISW) loss is introduced to suppress only the style-dependent covariance terms. Given a mask $\hat{M}$ that keeps only the strict upper triangular terms that are classified within G_{high}, i.e.

$$\hat{M}_{i,j} = \begin{cases} 1 & \text{if } V_{i,j} \in G_{high} \text{ and i} < \text{j}, \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

the ISW loss is defined as

$$\mathcal{L}_{ISW} = \mathbb{E}[\|\Sigma_s \odot \hat{\mathrm{M}}\|_1]. \tag{7}$$

This loss is added to the regular segmentation loss after the initial n epochs. Figure 1 illustrates all the steps to compute the ISW loss.

2.3 Network Architecture with ISW

The official implementation of RobustNet is based on IBN-b, which adds an instance normalization layer after the addition operation of a ResNet residual block [13]. After all, three instance normalization layers are added after the first three convolution groups (i.e., conv1, conv2_x, and conv3_x). Then, the ISW loss is added to the regular task loss as

$$\mathcal{L}_{total} = \mathcal{L}_{task} + \lambda(\frac{1}{L}\sum_{i=1}^{L}\mathcal{L}_{ISW}^{i}), \tag{8}$$

where λ is the weight for the ISW loss, which is set to 0.6, L is the number of layers to which the ISW loss is applied, i indicates the i^{th} layer, and $\mathcal{L}_{task}$ is the task loss (e.g. cross-entropy loss for semantic segmentation).

3 Experiments

RobustNet is implemented in the official source code for semantic segmentation with DeepLabV3+ architecture using the SGD optimizer. The initial learning rate is set to 10^{-2} with a momentum of 0.9 and a polynomial learning rate scheduling with the power of 0.9. Models are trained for 40k iterations and the multi-source models are trained for 110k iterations. As mentioned earlier, the first n iterations are used to gather statistics of the variance of covariances without using the ISW loss, where n is set to 5. After that, the ISW loss is introduced with the parameters k and m set to 3 and 1, respectively. The photometric transformation to simulate the domain shift is applied using color jitter and Gaussian blur. In addition, data augmentation to the input images is performed with color jitter, Gaussian blur, random crops, random horizontal flipping and random scaling in the range of $[0.5, 2.0]$.

In the original article RobustNet shows considerable quantitative improvements in several datasets. Nonetheless, qualitative results in uncommon events should be further analyzed to better comprehend the domain generalization capabilities of the method. Thus, a collection of out-of-dataset images were handpicked to test the performance of RobustNet in complex scenarios, which can be categorized as night, rain, snow, sun reflection, and a combination of two or more of them.

3.1 Adverse Weather Conditions

We evaluate the results of the method for adverse weather conditions, which include the *rain* and *snow* categories. Overall, RobusNet shows an improvement in the details of some challenging classes with respect to the baseline model, such as terrain, sidewalk or vegetation. Nevertheless, some confusing labelling can be produced by the model. Figure 2 displays results for some adverse weather examples. On the one in the second row, RobustNet improves the baseline output for the left side of the road, but the detection of the car on the right is confusingly mixed with *road* and *sidewalk* labels. Furthermore, we can observe on the third row how RobustNet produces some ghost detections of the *car* label as well.

3.2 Adverse Lightning Conditions

We additionally examine the performance of RobustNet for adverse lighting conditions, including *night* and *sun reflection*. As it can be observed in Figure 3, scenes at night are generally more challenging than sun reflection scenes. While the baseline model does not deal effectively with night scenes, RobustNet slightly improves the results. Despite this, we can observe how the model tends to label the sky as buildings, or how the shape of the cars is not consistent with human reasoning (the first row of Figure 3 shows how the method assigns strange parts of the road to cars). Reflections from the sun can affect differently to the model. In the third example, the sun does not impact much the baseline model nor RobustNet. On the fourth one, the scene is very challenging for both. RobustNet labels some buildings as vegetation, while the sky still is labelled as *building*.

3.3 Impact of Training Data

Change of location does not seem to affect the performance of RobustNet. However, the training dataset plays an important role. While RobustNet reduces the domain gap between train and test data, the larger the gap the more challenging is for the model to perform well. This can be observed in Figure 4, which shows the results of RobustNet when trained on real data (Cityscapes dataset [4]) and on synthetic data (GTAV dataset [15]). The domain gap between the training synthetic data

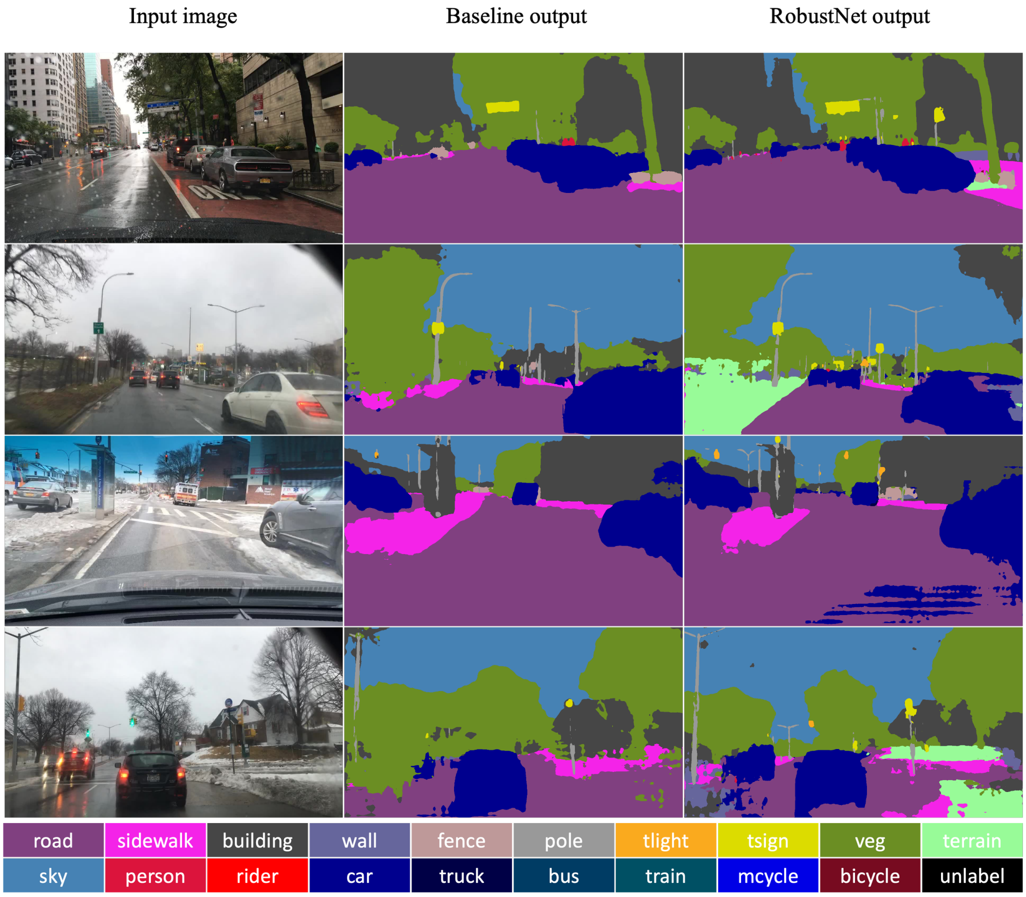


Figure 2: Adverse weather conditions results for RobustNet. The two upper rows correspond to *rain*, while the two on the bottom correspond to *snow*. Input images are shown in the left column. In the middle, baseline model outputs are displayed, while RobustNet outputs are shown on the right column. All images were produced with a DeepLabV3+ architecture with ResNet-50 encoder, trained on the Cityscapes dataset [4].

and the real testing data is greater than when both sets consist of real data. In consequence, the results for RobustNet trained on GTAV show many more inconsistencies.

4 Demo

This article includes an online demo which allows to quickly test custom data on the RobustNet algorithm, and to compare it to the state of the art for semantic segmentation. The demo refers to the official source code from the original paper [3] and uses the provided pretrained network weights. The following subsections discuss how to use the demo and explain how to test the users' own data.

4.1 Input

The demo only requires one input image to be provided. While uploading custom data for particular use cases is encouraged, six scenes are provided to quickly visualize the domain generalization capabilities of RobustNet. The provided data consist of three images that present challenging weather and environment conditions, extracted from the BDD100K dataset [17], and three images from different locations around the world, extracted from the Mapillary Vistas dataset [11]. These are shown in Figure 5.

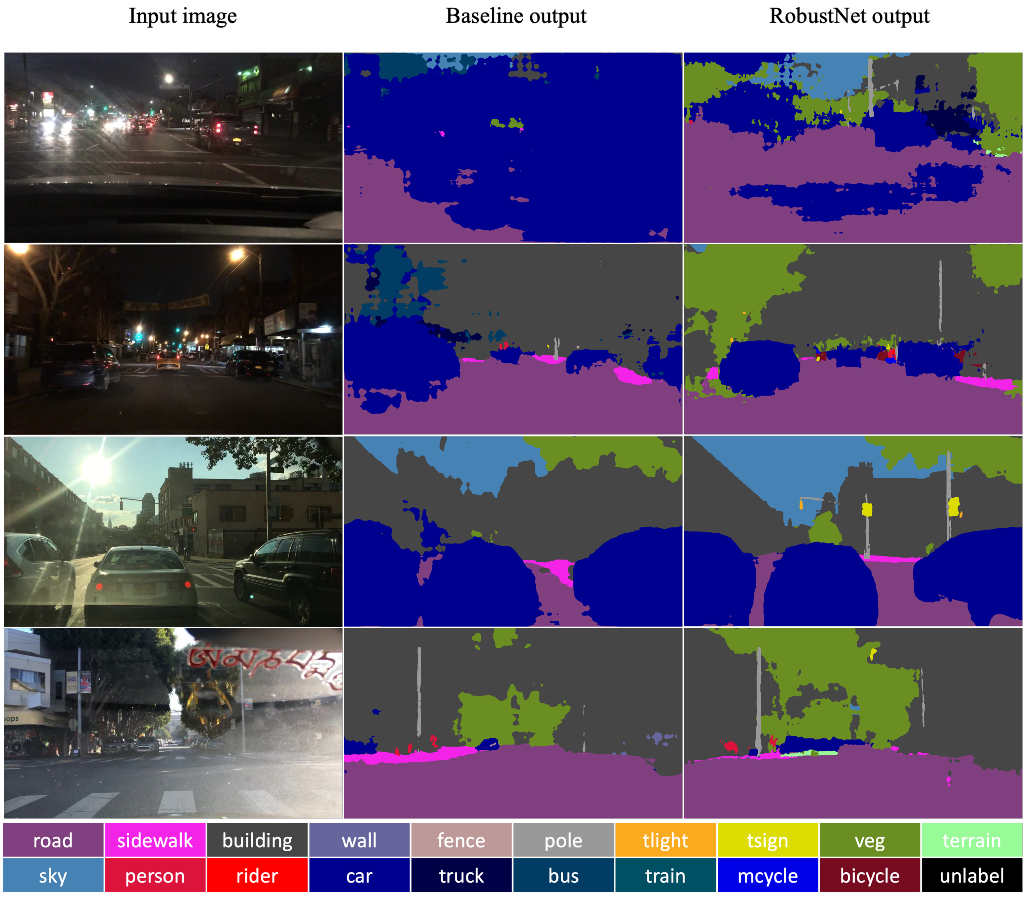


Figure 3: Adverse lightning conditions results for RobustNet. The two upper rows correspond to *night*, while the two on the bottom correspond to *sun reflection*. Input images are shown in the left column. In the middle, baseline model outputs are displayed, while RobustNet outputs are shown on the right column. All images were produced with a DeepLabV3+ architecture with ResNet-50 encoder, trained on Cityscapes dataset [4].

4.2 Parameters

The demo loads the pretrained weights of the baseline and the RobustNet models to produce the results. It includes only one parameter that specifies the network architecture and the training dataset. There are three possible options:

- DeepLabV3+ [1] with a ResNet-50 [6] encoder, trained on the Cityscapes [4] dataset.
- DeepLabV3+ with a ResNet-101 [6] encoder, trained on the Cityscapes [4] dataset.
- DeepLabV3+ with a ResNet-50 encoder, trained on the GTAV [15] dataset.

4.3 Output

When one of the options and an input image have been selected, the user can click on the button *Run* to execute the demo. Execution time of the pre-selected images ranges from 20 to 30 seconds. Note that if the input images are considerably larger, execution time can increase. The output of the demo will show the input image, followed by the segmentation output maps from the baseline model and RobustNet. To clarify, if the architecture chosen is DeepLabV3+ with ResNet-50, the baseline output will show the predictions of this architecture trained on the selected dataset, while the RobustNet prediction will show the same architecture with the Instance Whitening Loss approach

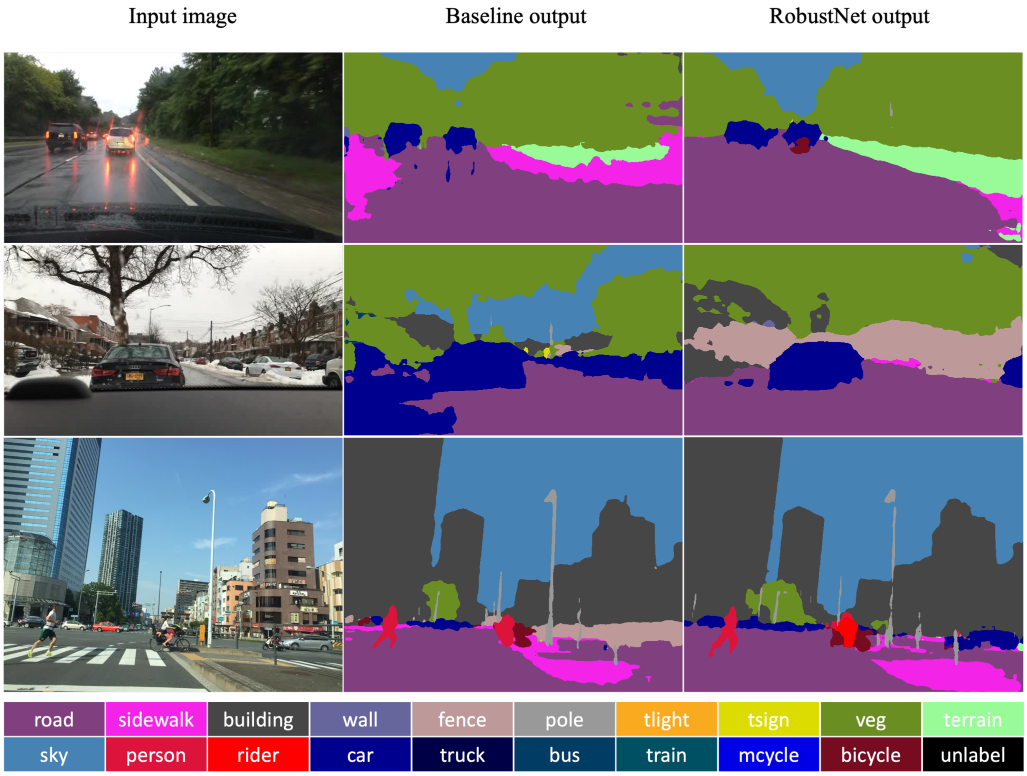


Figure 4: Evaluation of performance of RobustNet for different training datasets. Input images are shown in the left column. In the middle, the results for RobustNet trained on the Cityscapes dataset are displayed, while on the right column the results for RobustNet trained on the GTAV dataset are shown. All images were produced with a DeepLabV3+ architecture with ResNet-50 encoder.

for Domain Generalization. Figure 6 shows an output example for the input image *Rain*, one of the suggested input images.

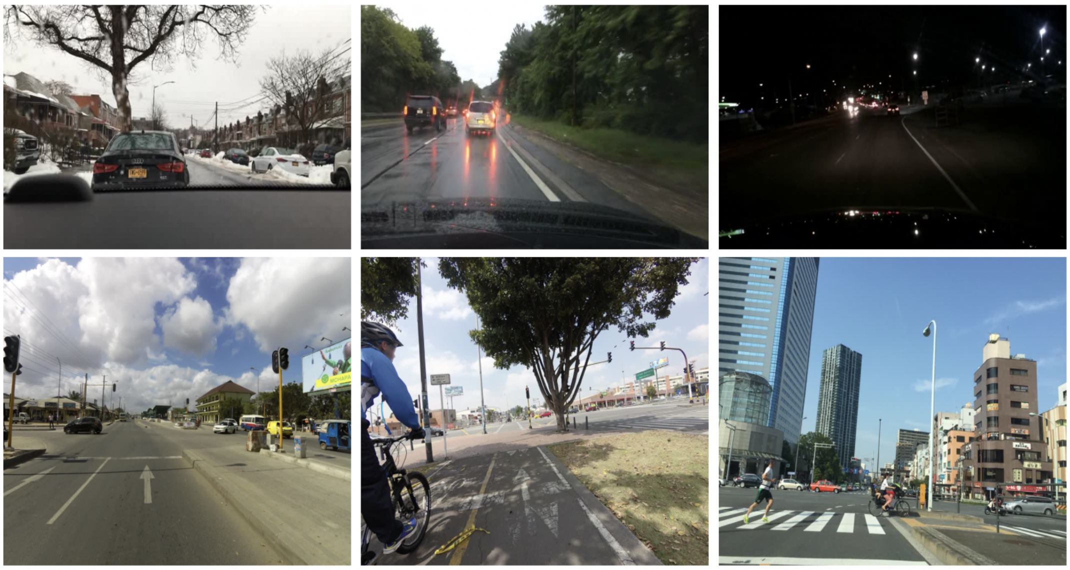
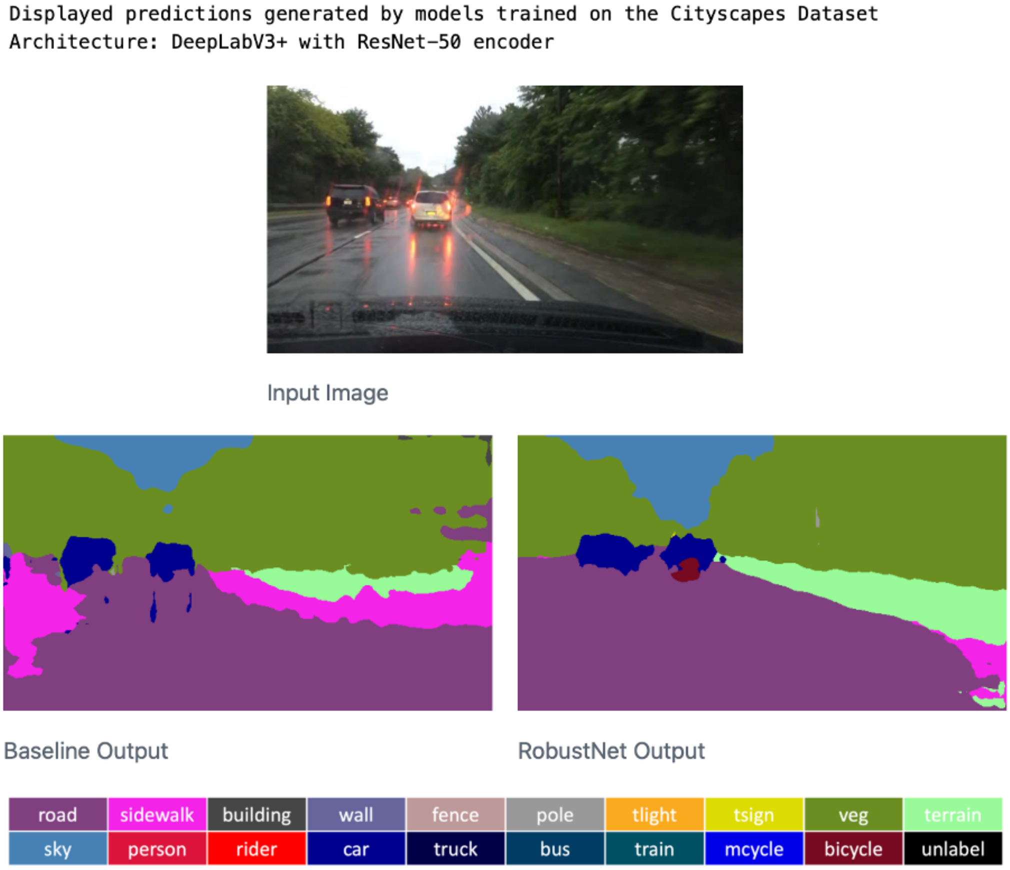


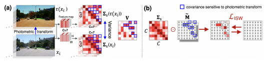
Figure 5: Sample images suggested in the demo. The images on the top row were extracted from the BDD100K dataset [17] and include challenging weather and environment conditions (from left to right: snow, rain and night). The images on the bottom were extracted from the Mapillary Vistas dataset [11], and correspond to a set of diverse locations (from left to right: Africa, South America and Asia).

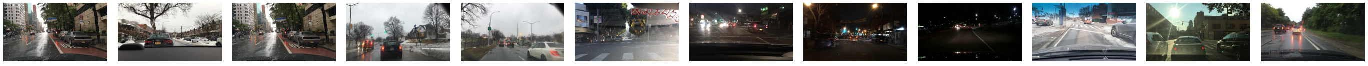


```
Displayed predictions generated by models trained on the Cityscapes Dataset
Architecture: DeepLabV3+ with ResNet-50 encoder
```

Figure 6: Example of the output of the demo for one of the suggested input images (Rain).

Image Credits

 from the RobustNet original article [3].



from the BDD100K dataset [17].

from the Mapillary Vistas dataset [11].

References

[1] L-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, *Encoder-decoder with atrous separable convolution for semantic image segmentation*, in European Conference on Computer Vision (ECCV), 2018. https://doi.org/10.1007/978-3-030-01234-2_49.

[2] W. Cho, S. Choi, D. Park, I. Shin, and J. Choo, *Image-to-image translation via group-wise deep whitening-and-coloring transformation*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 10631–10639. https://doi.org/10.1109/CVPR.2019.01089.

[3] S. Choi, S. Jung, H. Yun, J. Kim, S. Kim, and J. Choo, *RobustNet: Improving Domain Generalization in Urban-Scene Segmentation via Instance Selective Whitening*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 11575–11585. http://dx.doi.org/10.1109/CVPR46437.2021.01141.

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, *The Cityscapes Dataset for Semantic Urban Scene Understanding*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016. https://doi.org/10.1109/CVPR.2016.350.

[5] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V.S. Lempitsky, *Domain-adversarial training of neural networks*, CoRR, abs/1505.07818 (2015). http://arxiv.org/abs/1505.07818.

[6] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

[7] D. Li, J. Zhang, Y. Yang, C. Liu, Y-Z. Song, and T. Hospedales, *Episodic training for domain generalization*, in IEEE International Conference on Computer Vision (ICCV), 2019, pp. 1446–1455. https://doi.org/10.1109/ICCV.2019.00153.

[8] H. Li, S.J. Pan, S. Wang, and A.C. Kot, *Domain generalization with adversarial feature learning*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018. https://doi.org/10.1109/CVPR.2018.00566.

[9] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M-H. Yang, *Universal style transfer via feature transforms*, in International Conference on Neural Information Processing Systems (NIPS), 2017. https://dl.acm.org/doi/pdf/10.5555/3294771.3294808.

[10] Y. Li, Y. Yang, W. Zhou, and T.M. Hospedales, *Feature-critic networks for heterogeneous domain generalization*, in International Conference on Machine Learning (ICML), 2019. https://doi.org/10.48550/arXiv.1901.11448.

[11] G. Neuhold, T. Ollmann, S. Rota Bulò, and P. Kontschieder, *The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes*, in IEEE International Conference on Computer Vision (ICCV), 2017, pp. 5000–5009. https://doi.org/10.1109/ICCV.2017.534,.

[12] F. Pan, I. Shin, F. Rameau, S. Lee, and I. Kweon, *Unsupervised intra-domain adaptation for semantic segmentation through self-supervision*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. https://doi.org/10.1109/CVPR42600.2020.00382.

[13] X. Pan, P. Luo, J. Shi, and X. Tang, *Two at Once: Enhancing Learning and Generalization Capacities via IBN-Net*, in European Conference on Computer Vision (ECCV), 2018. https://doi.org/10.1007/978-3-030-01225-0_29.

[14] F. Pizzati, R. de Charette, M. Zaccaria, and P. Cerri, *Domain bridge for unpaired image-to-image translation and unsupervised domain adaptation*, in IEEE Winter Conference on Applications of Computer Vision (WACV), 2020, pp. 2979–2987. https://doi.org/10.1109/WACV45572.2020.9093540,.

[15] S. Richter, V. Vineet, S. Roth, and V. Koltun, *Playing for data: Ground truth from computer games*, in European Conference on Computer Vision (ECCV), vol. 9906, 2016. https://doi.org/10.1007/978-3-319-46475-6_7.

[16] T-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Perez, *ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019. https://doi.org/10.1109/CVPR.2019.00262.

[17] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2633–2642. https://doi.org/10.1109/CVPR42600.2020.00271.

[18] X. Yue, Y. Zhang, S. Zhao, A. Vincentelli, K. Keutzer, and B. Gong, *Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data*, in IEEE International Conference on Computer Vision (ICCV), 2019, pp. 2100–2110. https://doi.org/10.1109/ICCV.2019.00219.

[19] J-Y. Zhu, T. Park, P. Isola, and A. Efros, *Unpaired image-to-image translation using cycle-consistent adversarial networks*, in IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2242–2251. https://doi.org/10.1109/ICCV.2017.244.

[20] Y. Zou, Z. Yu, B.V.K. Vijaya Kumar, and J. Wang, *Unsupervised domain adaptation for semantic segmentation via class-balanced self-training*, in European Conference on Computer Vision (ECCV), 2018. https://doi.org/10.1007/978-3-030-01219-9_18.