Feature Generator Layer for Semantic Segmentation Under Different Weather Conditions for Autonomous Vehicles

Özgür Erkent[1], Christian Laugier[1]

***Abstract*— Adaptation to new environments such as semantic segmentation in different weather conditions is still a challenging problem. We propose a new approach to adapt the segmentation method to diverse weather conditions without requiring the semantic labels of the known or the new weather conditions. We achieve this by inserting a feature generator layer (FGL) into the deep neural network (DNN) which is previously trained in the known weather conditions. We only update the parameters of FGL. The parameters of the FGL are optimized by using two losses. One of the losses minimizes the difference between the input and output of FGL for the known weather domain to ensure the similarity between generated and non-generated known weather domain features; whereas, the other loss minimizes the difference between the distribution of the known weather condition features and the new weather condition features. We test our method on SYNTHIA dataset which has several different weather conditions with a well-known semantic segmentation network architecture. The results show that adding an FGL improves the accuracy of semantic segmentation for the new weather condition and does not reduce the accuracy of the semantic segmentation of the known weather condition.**

I. Introduction

Semantic information of the environment provide valuable data for the navigation and planning in intelligent vehicles. Recent developments in deep learning methods make them dominant in semantic segmentation such as DeepLabV3 [1] and SegNet [2]. Furthermore, they can be integrated with spatial 2D maps of the environment [3], [4] to produce 2D spatial semantic maps of the surroundings [5]. However, adaptation to new environments such as varying weather conditions is still a challenging problem.

Deep learning methods need a tremendous amount of supervised data to train. Although datasets with labeled images exist for semantic segmentation of RGB images taken from autonomous vehicles at pixel level for classes such as *road, car, pedestrian, etc.* (e.g. [6], [7]), providing a dataset for each separate weather condition is exhaustive. Thus, we focus on the problem of unsupervised adaptation to new weather conditions and use the same deep neural network (DNN) model to semantically segment successfully both known weather condition and the new weather condition images.

In unsupervised domain adaptation, the known weather condition for which we have the labels is called as the *source domain data* and the new weather condition without labels is called as the *target domain data*. We assume that we already have an initial DNN model to estimate the semantic

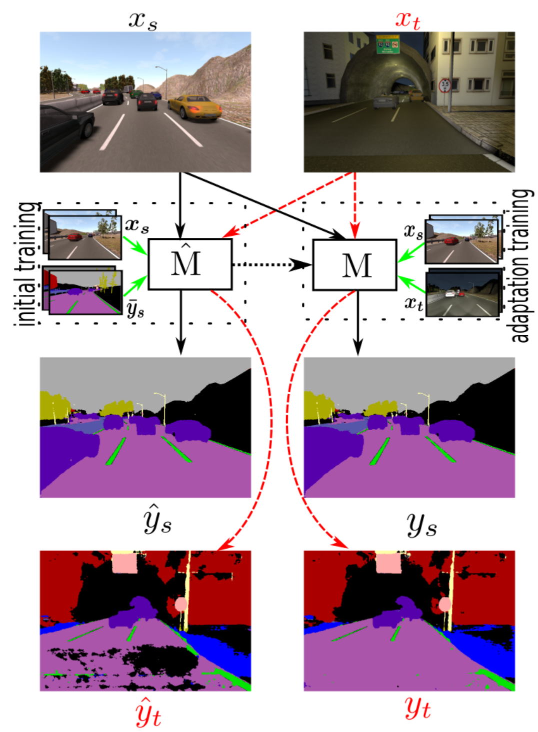


Fig. 1: A general overview. The initial model $\hat{\mathrm{M}}$ is trained with source domain images x_s and labels y_s. Then, it adapts to the new weather condition by only using the source domain images x_s and target domain images x_t. No labels are used during adaptation training. Green lines represent the flow of data used for training, black straight lines represent the flow for the segmentation of source domain and red dashed lines represent the flow for the segmentation of target domain. Note that the holes on the road segmentation at night diminish after adaptation while the segmentation of the daylight condition remains intact.

segmentation of the RGB images trained on *source domain data* as shown in Fig. 1. We propose to obtain a new DNN model by inserting a feature generator layer (FGL) such that the accuracy of the network improves with respect to the original DNN model for target domain segmentation and its accuracy does not degrade for the source domain segmentation. This implies that the same model can be used in different weather conditions for autonomous vehicles, which would be an advantage since the weather conditions can change during driving, e.g. it can start and stop raining during the same ride.

We insert the FGL after the initial feature extraction layers of the DNN, which corresponds to the "encoder" part of

[1] INRIA, Chroma Team, Rhône-Alpes, France. Correspondence: `ozgur.erkent@inria.fr`

the network. We only update the parameters of FGL for adaptation. We define two losses to optimize FGL. One loss is the difference between the distribution of the source and target domain data. We investigate three alternatives for this; Wasserstein distance [8], Jensen-Shannon divergence [9] and maximum mean discrepancy (MMD) [10]. The other loss measures the difference between the output and input of the FGL for source domain. We require the output of FGL to be similar to its input for the source domain.

The contributions of this paper can be listed as follows:

- To use only the source domain data without labels, target domain data without labels and pre-trained DNN model;
- To update only the parameters of the inserted FGL which results in faster adaptation training;
- To obtain an adapted DNN model with an improved accuracy for the target domain and a similar accuracy to the pre-trained model for source domain data;
- To report the accuracy of the adapted model for both domains.

We evaluate our approach by adapting the SegNet [2] for varying weather conditions since it is fast and does not require much memory which are important criteria for autonomous vehicles. We test it in Winter, Spring and Night conditions on SYNTHIA [7].

The rest of the paper is organized as follows. First, in Section. II, we discuss briefly the literature related to the domain adaptation. Then, In Section. III, we explain our solution by inserting a FGL with three alternative distance measures for data distribution. In Section. IV, we evaluate our method with SYNTHIA under two different weather conditions. Finally, we conclude the paper with a brief summary and possible future directions of research.

II. Related Work

One of the approaches to unsupervised domain adaptation is to use images with incremental changes in the target domain. Dai *et al.* [11] train a network with the images obtained at different times of the twilight with incremental darkness to adapt to darkness. The labels for the first twilight time zone are estimated with the original network which was trained with daylight images are accepted as "*target domain ground truth*" labels. The network is re-trained with a mixture of source and "*target domain ground truth*" labels. Wulfmeier *et al.* [12] use an other incremental approach. They train a separate encoder for target domain data with generative adversarial network (GAN) [9] to make the target domain features similar to source domain features. Although the performance is improved for the target domain, one of the concerns is the requirement of incremental data collection which will not be feasible for various weather/illumination conditions for autonomous vehicles.

Another approach is to transform the input images from the target domain to the source domain so that they are similar in appearance. Porav *et al.* [13] obtain a dataset with a special stereo camera to capture the image of the scene both with rain and without rain and train a network to convert the rainy images into de-rainy ones. The rainy condition is maintained with a glass with water droplets. The requirement of the exactly same scenes under different weather conditions is difficult to obtain for each weather condition. Cycada method [14] uses generators at different parts of the network. The usage of multiple generators ensure a cyclic transformation. Requirement of multiple generators makes this approach computationally expensive.

Another group of studies focus on minimizing the distribution in the output of the network. Vu *et al.* [15] implement adversarial entropy minimization for the outputs of the source and target domains. Wu *et al.* [16] find the geodesic loss [17] on the output of the network and back-propagate it for optimization. Zhang *et al.* [18] implement a super-pixel and label generator to transform the target domain output into the source domain output. The output is converted into a higher-dimensional feature space to be used with adversarial training, which may not always guarantee to represent the difference of the distribution between source and target domains. Furthermore, another problem with this group of approaches is that the necessary information to transform target domain into source domain data may have already been lost in the initial layers of the network.

Finally, we consider studies that compare the distributions of the features in both domains. The features generally have a high-dimension; therefore, adversarial training can be directly used. Tzeng *et al.* [19] proposed one of the early unsupervised domain adaptation in DNNs. They adapt a DNN by inserting a generator into the network and train it by using the MMD [20] to measure the discrepancy between the source and target domain features. Long *et al.* [21] also propose a similar architecture, but with multiple MMD kernels instead of single, and without additional adaptation layer. Additionally, both use the labels of the source domain to optimize the parameters of the network. Ganin *et al.* [22] used GANs [9] to adapt. In some other variants, GANs [9] are used for adaptation in semantic segmentation by deploying a discriminator at different layers of the network to measure the difference in distribution between the source and domain data features and the output (e.g. [23], [24], [25]). Peng *et al.* [26] and Hong *et al.* [27] insert a generator into the network to transform the distribution of target feature distribution into source domain feature distribution. Peng *et al.* [26] train based on W-GANs [8] with penalty gradient [28] while Hong *et al.* [27] use a GAN [9] based approach.

We covered the domain adaptation approaches that can be applied to DNNs by considering the computational complexity and the memory requirements for autonomous vehicles. It should be noted that some of the mentioned methods are used for adaption from the simulator domain to the real world domain; however, we will focus only on the new weather condition adaptation. We propose an approach which is different from the mentioned methods in three aspects. Firstly, we develop a method which does not require the source domain labels for adaptation. Secondly, it is usually a common practice to report the accuracy of the target domain with the new trained model [10]; since we propose to use

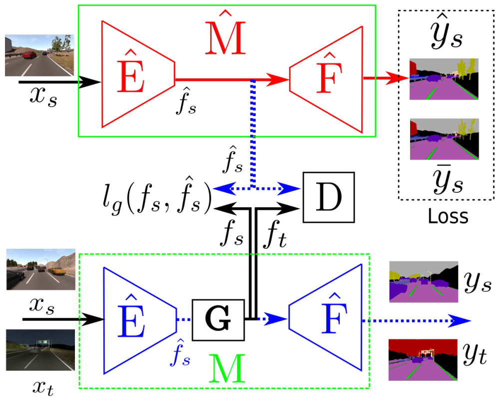


Fig. 2: First, the initial model $\hat{\mathrm{M}}$ is trained with source domain ground truth labels $\bar{y}_s$. The training of the initial model is shown with red straight lines. Then, FGL layer G is inserted into this model and the parameters of G is optimized by using the source domain and target domain features f_s and f_t. The training of adaptation is shown with black straight lines. The blue dotted lines are not used for back-propagation.

the same model in different weather conditions, we provide the performance of the same network in both target and source domains. Our final difference is that we don't update the parameters of the pre-trained model. We only update the parameters of the generator which significantly reduces memory and time requirements for training.

III. Domain Adaptation

In classical unsupervised domain adaptation, source domain data with its labels $\mathcal{S} = \{(\mathbf{x}_s^i, y_s^i)\}_{i=1}^{n_s}$ and target domain data $\mathcal{X}_\mathrm{t} = \{\left(\mathbf{x}_t^j\right)\}_{j=1}^{n_t}$ are provided where n_s and n_t are the number of source and target domain samples respectively. In our problem, we consider that only source data $\mathcal{X}_\mathrm{s} = \{\left(\mathbf{x}_s^i\right)\}_{i=1}^{n_s}$, target data $\mathcal{X}_\mathrm{t}$ and a previously learned deep neural network model $\hat{\mathrm{M}}_{\hat{\gamma}}$ with parameters $\hat{\gamma}$ to predict the labels y_s^i which can minimize the source cost $c_s\left(\hat{\mathrm{M}}_{\hat{\gamma}}\right) = \mathrm{Pr}_{(\mathbf{x},y)\sim p}\left[\hat{\mathrm{M}}_{\hat{\gamma}}(\mathrm{x}) \neq y\right]$ are available. The source and target domains have probability distributions of Pr_p and Pr_q respectively. Our objective is to obtain a model such that it can minimize the target cost $c_t\left(\mathrm{M}_\gamma\right) = \mathrm{Pr}_{(\mathbf{x},y)\sim q}\left[\mathrm{M}_\gamma(\mathrm{x}) \neq y\right]$ and source cost $c_s\left(\mathrm{M}_\gamma\right) = \mathrm{Pr}_{(\mathbf{x},y)\sim p}\left[\mathrm{M}_\gamma(\mathrm{x}) \neq y\right]$ simultaneously using available source data , target data and the previously learned model.

We divide the neural network $\hat{\mathrm{M}}_{\hat{\gamma}}$ into two: encoder $\hat{\mathrm{E}}_{\hat{\theta}}$ and the decoder $\hat{\mathrm{F}}_{\hat{\alpha}}$ such that $\hat{\mathrm{M}}_{\hat{\gamma}}(\mathrm{x}) = \hat{\mathrm{F}}_{\hat{\alpha}}(\hat{\mathrm{E}}_{\hat{\theta}}(\mathrm{x}))$ as shown in Fig. 2. We propose a method where we insert a feature generator layer G_ϑ between $\hat{\mathrm{F}}_{\hat{\alpha}}$ and $\hat{\mathrm{E}}_{\hat{\theta}}$. All the parameters are kept constant except G_ϑ. The output of the new adapted layer becomes $\mathbf{f} = \mathrm{G}_\vartheta(\hat{\mathrm{E}}_{\hat{\theta}}(\mathbf{x}))$. The inputs and outputs of the layer are same in size. We require the generator to produce the target domain features $\mathbf{f}_t$ similar to the source domain features $\mathbf{f}_s$ in distribution and in addition, we need the source domain features to be generated with minimal change, which can be formulated as the following optimization:

$$\inf_{\mathrm{G}(.)\in\mathcal{G}} \mathrm{D}(\mathrm{Pr}_{(\hat{\mathbf{f}})\sim p}, \mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q}) + \lambda l_g(\hat{\mathbf{f}}_s, \mathrm{G}(\hat{\mathbf{f}}_s)) \tag{1}$$

where $\mathcal{G}$ is the set of all possible generators, D measures the distance between the target and source domain feature distributions, $\lambda > 0$ is a regularization hyperparameter and l_g measures the cost between the source domain features $\hat{\mathbf{f}}_s^i = \hat{\mathrm{E}}_{\hat{\theta}}(\mathbf{x}_s^i)$ and generated source domain features $\mathrm{G}(\hat{\mathbf{f}}_s^i)$. We use the mean squared error (MSE) $l_g(\hat{\mathbf{f}}_s, \mathrm{G}(\hat{\mathbf{f}}_s)) = ||\hat{\mathbf{f}}_s - \mathrm{G}(\hat{\mathbf{f}}_s)||_2^2$. We hold on the assumption that features $\{\left(\mathbf{f}_t^j\right)\}_{j=1}^{n_t}$ sampled from the target distribution $\mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q}$ contain necessary and sufficient information to classify the target domain data similar to the source data.

We propose three different distribution distance measures for $\mathrm{D}(\mathrm{Pr}_{(\hat{\mathbf{f}})\sim p}, \mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q})$:

W-GAN proposal: First, we start with Wasserstein GANs [8]. $m-$th order Wasserstein distance between two distributions can be defined as follows:

$$W_m(\mathrm{Pr}_{(\hat{\mathbf{f}})\sim p}, \mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q}) = \inf_{\mathrm{Pr}_{\hat{\mathbf{f}}_s\sim p,\mathrm{G}(\hat{\mathbf{f}}_t)\sim q}} \mathbb{E}\left[||\hat{\mathbf{f}}_s - \mathrm{G}(\hat{\mathbf{f}}_t)||^m\right] \tag{2}$$

where the minimization is over all joint distributions. Then, Eq. 1 can be rewritten as follows:

$$\inf_{\mathrm{G}(.)\in\mathcal{G}} \inf_{\mathrm{Pr}_{\hat{\mathbf{f}}_s\sim p,\mathrm{G}(\hat{\mathbf{f}}_t)\sim q}} \mathbb{E}\left[||\hat{\mathbf{f}}_s - \mathrm{G}(\hat{\mathbf{f}}_t)||^m\right] + \lambda l_g(\hat{\mathbf{f}}_s, \mathrm{G}(\hat{\mathbf{f}}_s)) \tag{3}$$

the second part of the optimization deals with the loss in the source domain. Hence its optimal value is not affected by the target domain, we solve it as a minimization problem in the source domain only. However, the first part needs to consider the joint distribution of both source and target domains for optimization. It is an optimal transport problem and a linear programming problem, therefore it always has a dual formulation (Kantorovich-Rubinstein) [29]. Arjovsky *et al.* [8] proposed an adversarial network for $m = 1$ and its variant with gradient penalty has been introduced by Gulrajani *et al.* [28]. We describe the implementation of the generative method with W-GANs in Algorithm.1.

GAN proposal: Secondly, we use the GANs based on Jensen-Shannon (JS) distance implemented by Goodfellow *et al.* [9] JS distance between $\mathrm{Pr}_{(\hat{\mathbf{f}})\sim p}$ and $\mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q}$ can be defined as follows:

$$JS = KL(\mathrm{Pr}_{\hat{\mathbf{f}}_s\sim p}||\mathrm{Pr}_{\hat{\mathbf{f}}\sim r}) + KL(\mathrm{Pr}_{\mathrm{G}(\hat{\mathbf{f}}_t)\sim q}||\mathrm{Pr}_{\hat{\mathbf{f}}\sim r}) \tag{4}$$

where KL is the Kullback-Leibler divergence and $\mathrm{Pr}_{\hat{\mathbf{f}}\sim r} = (\mathrm{Pr}_{\hat{\mathbf{f}}_s\sim p} + \mathrm{Pr}_{\mathrm{G}(\hat{\mathbf{f}}_t)\sim q})/2$ is the mixture. We describe the implementation of the generative method with GANs in Algorithm.1.

MMD proposal: Finally, we use maximum mean discrepancies which was used in domain adaptation problems with DNNs (e.g. Long *et al.* [21]). It maximizes the two-sample test power and minimizes the Type II error. If $\mathcal{H}_k$ is the reproducing Hilbert space (RKHS) with a characteristic positive-definite reproducing kernel k and the mean embedding of distribution p in $\mathcal{H}_k$ is a unique element $\mu_k(p)$ such that $\mathbb{E}_{\mathbf{x}\sim p} f(\mathbf{x}) = \langle f(\mathbf{x}), \mu_k(p)\rangle_\mathcal{H}$, $\forall f \in\mathcal{H}_k$; then, we define $\mathrm{D}(\mathrm{Pr}_{(\hat{\mathbf{f}})\sim p}, \mathrm{Pr}_{(\mathrm{G}(\hat{\mathbf{f}}))\sim q})$ to be the MMD between distributions p and q which is the RHKS distance between the mean

Algorithm 1: Feature Generator Domain Adaptation

Require: $\mathcal{X}_s$, $\mathcal{X}_t$, G_ϑ,D_β, $\lambda > 0$, $\hat{\mathrm{E}}$, n: Mini-batch size, $A \in \{\text{WGAN,GAN,MMD}\}$, n_{crit}, $\lambda_p > 0$: Penalty gradient
if ($A =$ WGAN) OR ($A =$ GAN) **then**
  initialize the parameters of D_β and G_ϑ
else
  provide a characteristic positive-definite kernel k and initialize the parameters of G_ϑ
while (β,ϑ) not converged **do**
  Sample $\{(\mathbf{x}_s^i)\}_{i=1}^n \in \mathcal{X}_s$, $\{(\mathbf{x}_t^i)\}_{i=1}^n \in \mathcal{X}_t$
  Compute $\hat{\mathbf{f}}_s^i = \hat{\mathrm{E}}(\mathbf{x}_s^i)$, $\mathbf{f}_t^i = \mathrm{G}(\hat{\mathrm{E}}(\mathbf{x}_t^i))$,$\mathbf{f}_s^i = \mathrm{G}(\hat{\mathrm{E}}(\mathbf{x}_s^i))$;
  if $A =$ WGAN **then**
    for $k = 1, \cdots, n_{\text{crit}}$ **do**
      Sample a random number $\epsilon \sim U[0,1]$;
      $\widetilde{\mathbf{f}}^i = \epsilon\hat{\mathbf{f}}_s^i + (1-\epsilon)\mathbf{f}_t^i$;
      Update D by ascending

$$\frac{1}{n}\sum_{i=1}^{n}\mathrm{D}(\hat{\mathbf{f}}_s^i) - \mathrm{D}(\mathbf{f}_t^i) - \lambda_p(||\nabla_{\widetilde{\mathbf{f}}}\mathrm{D}(\widetilde{\mathbf{f}}^i)||_2 - 1)^2$$

    Update G by descending

$$\frac{1}{n}\sum_{i=1}^{n}\mathrm{D}(\mathbf{f}_t^i) + \lambda l_g(\mathbf{f}_s^i, \hat{\mathbf{f}}_s^i)$$

  else if $A =$ GAN **then**
    for $k = 1, \cdots, n_{\text{crit}}$ **do**
      Update D by ascending

$$\frac{1}{n}\sum_{i=1}^{n}\log\mathrm{D}(\hat{\mathbf{f}}_s^i) + \log(1 - \mathrm{D}(\mathbf{f}_t^i))$$

    Update G by descending

$$\frac{1}{n}\sum_{i=1}^{n}\lambda l_g(\mathrm{G}(\mathbf{f}_s^i, \hat{\mathbf{f}}_s^i) - \log(\mathrm{D}(\mathbf{f}_t^i))$$

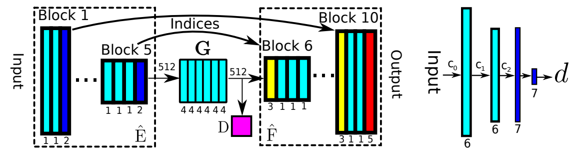
  else if $A =$ MMD **then**
    Update G by descending

$$\frac{1}{n}\sum_{i=1}^{n}\lambda l_g(\mathbf{f}_s^i, \hat{\mathbf{f}}_s^i) + \frac{1}{n(n-1)}\sum_{l\neq j}k(\mathbf{f}_t^l, \mathbf{f}_t^j))$$
$$+\frac{1}{n(n-1)}\sum_{l\neq j}k(\hat{\mathbf{f}}_s^l, \hat{\mathbf{f}}_s^j)) - \frac{2}{n^2}\sum_{l,j}k(\mathbf{f}_t^l, \hat{\mathbf{f}}_s{}^j))$$

embeddings of p and q. Using the Kernel trick, MMD can be computed with the expectation of kernel functions as follows:

$$\begin{aligned}\mathrm{MMD}_k(\mathrm{Pr}_p, \mathrm{Pr}_q) = {}& \mathbb{E}_{\hat{\mathbf{f}}_s, \hat{\mathbf{f}}_{s'}}k(\hat{\mathbf{f}}_s, \hat{\mathbf{f}}_{s'}) + \mathbb{E}_{\mathbf{f}_t, \mathbf{f}_{t'}}k(\mathbf{f}_t, \mathbf{f}_{t'}) \\ & - 2\mathbb{E}_{\hat{\mathbf{f}}_s, \mathbf{f}_t}k(\hat{\mathbf{f}}_s, \mathbf{f}_t)\end{aligned} \tag{5}$$

where $\mathbf{f}_s, \mathbf{f}_{s'} \sim p$, $\mathbf{f}_t, \mathbf{f}_{t'} \sim q$ which means that $\mathbf{f}_{s'}$ and $\mathbf{f}_{t'}$ are sampled from source and target domains respectively and they are not same samples with $\mathbf{f}_s$ and $\mathbf{f}_t$. We explain the implementation of the method in Algorithm. 1. Since MMD has an unbiased U-statistic estimator, it can be used with a stochastic gradient descent (SGD) method to optimize the parameters of a deep neural network [21]. It is important to remind that MMDs require a large number of samples and a high computation time to optimize the parameters of the network.



(a) FGL in SegNet (b) Discriminator

Fig. 3: (a) FGL is inserted after Encoder $\hat{\mathrm{E}}$ and before decoder $\hat{\mathrm{F}}$. D measures the distance between the distributions of generated source and target domain features and used only during adaptation training. The labels for the layers are as follows: 1: 2D Convolution and Batch Normalization (BN) and ReLU, 2: Max-Pooling, 3: Upsampling, 4: 2D Convolution and ReLU, 5: Softmax. The generator has 512 input channels and 512 output channels. (b) Input has c_0 channels. After second convolutional layer, it is reduced to c_2 channels. Layer labels: 6: 2D Convolution and Instance Normalization, 7: Fully Connected Layer. The final output d is a scalar value.

IV. Experiments

The number of datasets for the semantic segmentation with specific weather conditions is limited. Although datasets may contain examples from different weather conditions, the label of the weather condition for each image is not available.

SYNTHIA is a synthetic dataset obtained for semantic and instance segmentation of the surroundings of a vehicle [7]. We use only the frontal left RGB images and its semantically annotated label images for the ground truth. We use three different weather conditions in two different subway scenarios: Spring, Winter and Night from Sequence 01 and 06. We train our initial SegNet model with the labels and RGB images of Spring-01 which contains 1188 images.

The encoder layers of SegNet use convolution layers similar to VGG-16 [30]. The initial parameters of the network are taken from a pre-trained version of VGG-16 network for classification of ILSVRC/Imagenet [31].

We use Stochastic Gradient Descent for optimization of initial SegNet model with Spring as source domain and a learning rate of 1×10^{-4} and a momentum of 0.9. The mini-batch size is 6. Approximately 198 epochs are necessary to cover all the images in the training set. We train until the loss converges or the maximum number of iterations is 2000. Once the network converges, the parameters are kept constant and the FGL is inserted into the network after the encoder layer as shown in Fig. 3a. FGL has six 2D convolution layers with same size of input and outputs and a kernel size of 3×3. Each convolution is followed by a rectified-linear non-linearity (ReLU) unit. We use a network for the discriminator when we use W-GAN [8] or GAN [9] to measure the differences in the distributions of the features. The Discriminator network is shown in Fig. 3b. It has two 2D-convolutional layers followed by the instance normalization layer [32]. Two fully connected (fc) layers follow these convolutional layers where the first fc is followed by a ReLU. The parameters

TABLE I: Adaptation to Winter for SYNTHIA with SegNet

%	Init	FGL+W1	FGL+JS	FGL+MMD
Spring-06	**59.8(0)**	59.4(-0.7)	59.5(-0.4)	59.7(-0.2)
Winter-01	55.9(0)	**61.7(10.4)**	60.6(8.4)	59.6(6.6)
Winter-06	47.8(0)	48.5(1.4)	48.5(1.4)	48.5(1.5)

TABLE II: Adaptation to Night for SYNTHIA with SegNet

%	Init	FGL+W1	FGL+JS	FGL+MMD
Spring-06	59.8(0)	59.7(-0.1)	59.9(0.2)	**60.0(0.4)**
Night-01	53.7(0)	**58.6(9.1)**	58.2(8.4)	57.5(7.0)
Night-06	35.6(0)	**38.8(8.9)**	38.3(7.6)	38.4(7.8)

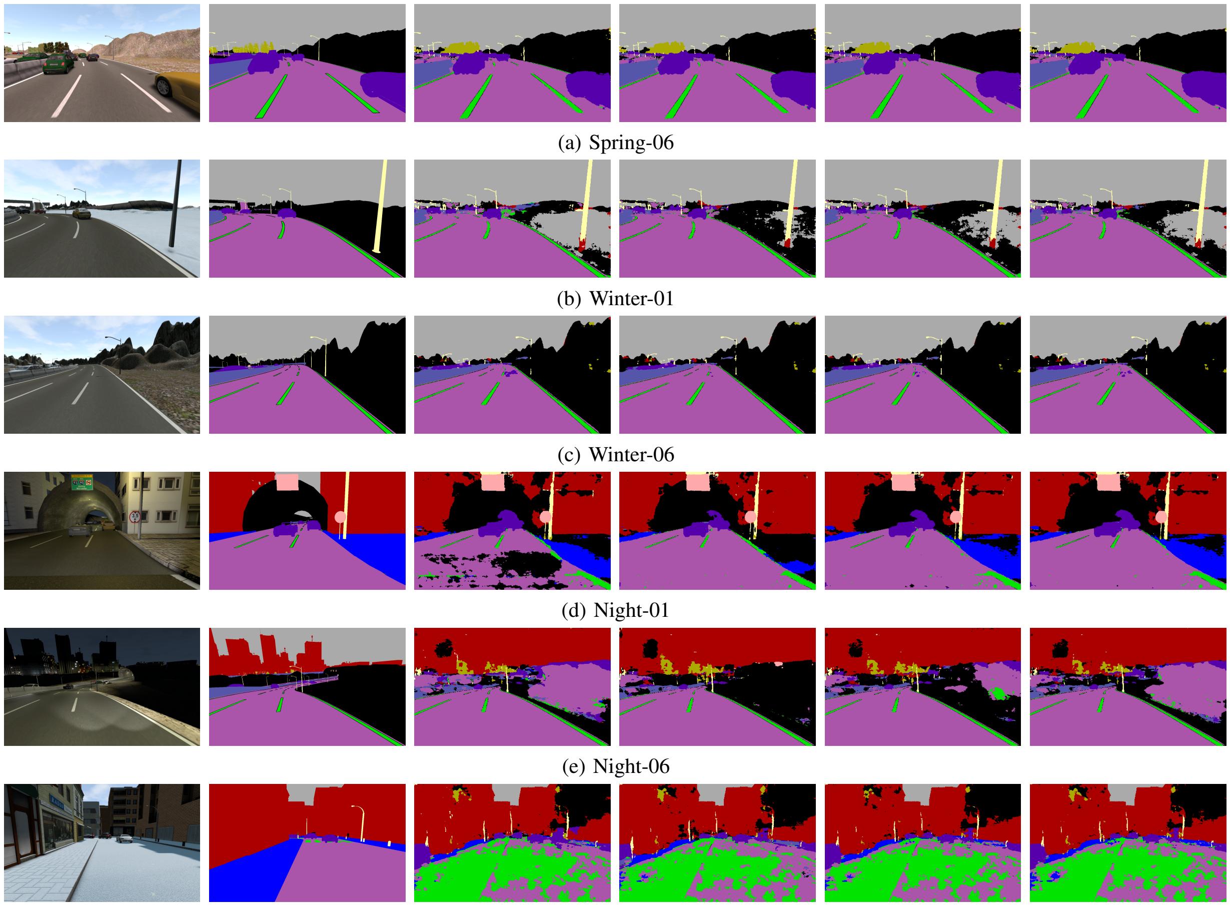
are $c_0 = 512$, $c_1 = c_2 = 64$. For GAN, additional ReLu and softmax layer is added to the end of the discriminator. The initial parameters are chosen randomly from a normal distribution. To optimize the parameters of the Generator and the Discriminator network, we use Adam for optimization with hyperparameters $\alpha = 1 \times 10^{-4}$, $\beta_1 = 0.7$ and $\beta_1 = 0.9$. The hyperparameters for GAN and W-GAN are $\lambda_p = 0.1$, $\lambda = 10$, $n_{\text{crit}} = 1$ and a mini batch size of 128 is used. For the characteristic kernel k in MMD, we used an inverse multiquadratics kernel $k(\mathbf{f}_t, \mathbf{f}_s) = C/(C + ||\mathbf{f}_t - \mathbf{f}_s||_2^2)$ where $C = 2d_f\sigma_f$. d_f is the dimensionality of the features. For MMD, the hyperparameters are $\lambda = 100$, $\sigma_f = 1$ and a mini batch size of 64 is used. The number of maximum iterations is 10000 for GAN and W-GAN and 5000 for MMD. The training takes approximately 4.8 h for W-GAN and GAN, and 13.7 h for MMD. The input size of the images are downsampled to 380×500 pixels. For analysis, the output image is later upsampled to the size of the SYNTHIA dataset ground truth label images. The inference time is around 1.3 ms on the GPU, this doesn't include the transfer time of the image to the GPU. All computation time evaluations are made on an Nvidia GTX 1080Ti.

We adapt the network to two different new weather conditions: winter and night. For adaptation, we use the images from sequence Winter-01 for winter with 1027 images and from sequence Night-01 for night with 935 images. After we obtain the adapted model, we test the source weather condition with images from Spring-06 to test the amount of accuracy degradation in the source domain after adaptation; with the target images used to adapt the network Winter-01 (or Night-01) and with the images that were not used for adaptation training, Winter-06 (or Night-06) to test the performance of the network with the unobserved target domain images. We believe that these cases are in accordance with a real scenario for an autonomous vehicle. The weather may change during driving and the incoming images will be the ones that were not used during adaption.

The results are shown in Table. I and Table. II. We use the mean of intersection over union (mIoU), which is widely used in semantic segmentation analysis. The first value is the mIou, while the value in parenthesis is the amount of change with respect to the initial model after the adaptation method is applied. We compare three measure distances after the insertion of FGL and the initial model without FGL (Init).

It can be seen that the accuracy of the initial model trained with Spring-01 degrades when the weather conditions change. When the road path sequence is different from the initially trained Sequence-01, the degradation is even higher (for the Sequence 06). Furthermore, the accuracy drop in Night-06 condition is higher. After adaptation in the same sequence, the accuracy increases for all the measures under the new weather condition. However, the accuracy increase for Winter-06 is not as high as the original sequence. To understand its reason better, we trained initial SegNet model with Winter-01 and tested Winter-6 with this new model. Even in this case, the mIou is only 53.9%. Therefore, the capacity of the SegNet may not be sufficient to segment the Winter conditions for SYNTHIA. It should be noted that the accuracy still increases slightly for Sequence-06 after adaptation. Interestingly, the accuracy for Winter-01 is better than the source domain which can be due to the similar structures in the same sequence. For the Night condition, even the accuracy of the Spring-06 increases slightly for some distance measures. Probably this is due to the improvement of the segmentation of the dark regions such as inside the tunnel after adaptation to Night. For Night condition, the accuracy increases for both Sequences. For both Winter and Night conditions, Wasserstein critic for the distance measure gives a better result for the adaptation domains. Therefore, it can be concluded that for the network with a small capacity, W-GAN approach can be used with FGL to adapt to a new weather condition.

The qualitative results are shown in Fig. 4. The segmentation of Spring is not affected after adaptation (Fig. 4a) to the initial model for Winter condition. In Fig. 4b, the results are shown after the model is adapted with images from Winter-01 without any labels from either domains. In GT, the right side of the road is covered with undefined region. However, the initial model falsely segments this region as the sky. After adaptation with W-GAN, the region wrongly labeled as sky is reduced significantly. It should be noted that W-GAN is more effective for this case. In Fig. 4c, the Winter-06 is tested on the same model used in Winter-01, model which is adapted to the Winter condition based on the initial model on Spring-01 and adapted with the images from Winter-01 only. After segmentation with the original model, there is a hallucination of a vehicle segmentation in front of our vehicle (on the 3^{rd} column). This diminishes after adaptation. For Night condition, we adapt the initial model by using the images from Night-01 and it is used for segmenting all the Night images. As it can be seen in Fig. 4d, there are holes on the road after segmentation with the initial model; however, they disappear after adaptation for all adaptation approaches. We use the same adapted model to segment the images of Night-06. As it can be seen in Fig. 4e, the right side of the road is undefined in the GT; however, the initial model segments this region as road, which is dangerous if the vehicle decides to drive towards this area. After adaptation with W-GAN, the wrong segmented area disappears. It reduces in size after adaptation with GAN, but MMD approach is not very effective in this case. Finally, we show a failed case in Fig. 4f. The road is covered with



(a) Spring-06

(b) Winter-01

(c) Winter-06

(d) Night-01

(e) Night-06

(f) Winter-06

Fig. 4: Visual result samples for SYNTHIA dataset. From left to right: RGB, Ground Truth (GT), Result with the initial model (initial), Adaptation with W-GAN, Adapted with GAN and Adaptation with MMD.

snow and none of the methods segment the road successfully. This is probably due to the unrealistic visualization of the winter conditions where the snow resemble the lanemarks and most of it is recognized as landmark. Also, probably the target features do not contain sufficient information for the road with snow to be classified as road similar to the source domain data.

V. Conclusion

We showed that an unsupervised adaptation method can be used to improve the accuracy of semantic segmentation of a DNN for different weather conditions without using the labels of the source domain to modify the parameters of a pre-trained DNN model. We used SegNet [2] due to its speed and accuracy and a synthetic dataset with different weather conditions. As part of the future work, we will evaluate the method with real images in different weather conditions and develop the approach to be used with more complicated network architectures.

Acknowledgement

This work was supported by Toyota Motor Europe.

References

[1] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation," 2017. [Online]. Available: http://arxiv.org/abs/1706.05587

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE PAMI*, vol. 39, no. 12, pp. 2481–2495, 2017.

[3] L. Rummelhard, A. Nègre, and C. Laugier, "Conditional monte carlo dense occupancy tracker," in *ITSC*. IEEE, 2015, pp. 2485–2490.

[4] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[5] O. Erkent, C. Wolf, C. Laugier, D. Gonzalez, and V. Cano, "Semantic grid estimation with a hybrid bayesian and deep neural network approach," in *IEEE IROS*, 2018, pp. 888–895.

[6] 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 *CVPR*, 2016.

[7] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez, "The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes," 2016.

[8] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

[10] B. Gong, K. Grauman, and F. Sha, "Connecting the Dots with Landmarks: Discriminatively Learning Domain-Invariant Features for Unsupervised Domain Adaptation," in *ICML*, vol. 28, 2013, pp. 1–9.

[11] D. Dai and L. Van Gool, "Dark Model Adaptation: Semantic Image

Segmentation from Daytime to Nighttime," in *ITSC*, 2018. [Online]. Available: http://arxiv.org/abs/1810.02575

[12] M. Wulfmeier, A. Bewley, and I. Posner, "Incremental Adversarial Domain Adaptation for Continually Changing Environments," in *ICRA*, 2018.

[13] H. Porav, T. Bruls, and P. Newman, "I Can See Clearly Now : Image Restoration via De-Raining," in *ICRA*, 2019. [Online]. Available: http://arxiv.org/abs/1901.00893

[14] J. Hoffman, E. Tzeng, T. Park, J.-y. Zhu, U. C. Berkeley, P. Isola, K. Saenko, A. A. Efros, T. Darrell, and U. C. Berkeley, "CYCADA: Cycle-Consistent Adversarial Domain Adaptation," in *ICML*, 2018, pp. 1–15.

[15] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez, "ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation," in *CVPR*, 2019. [Online]. Available: http://arxiv.org/abs/1811.12833

[16] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud," in *ICRA*, 2019. [Online]. Available: http://arxiv.org/abs/1809.08495

[17] P. Morerio, J. Cavazza, V. Murino, and P. Analysis, "Minimal-Entropy Correlation Alignment For Unsupervised Deep Domain Adaptation," in *ICLR*, no. 2011, 2018, pp. 1–14.

[18] Y. Zhang, P. David, and B. Gong, "Curriculum Domain Adaptation for Semantic Segmentation of Urban Scenes," in *ICCV*, 2018.

[19] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *CoRR*, vol. abs/1412.3474, 2014.

[20] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schoelkopf, and A. Smola, "A Kernel Method for the Two-Sample Problem," *JMLR*, vol. 13, pp. 723–773, 2012.

[21] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning Transferable Features with Deep Adaptation Networks," in *ICML*, vol. 37, 2015. [Online]. Available: http://arxiv.org/abs/1502.02791

[22] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[23] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker, "Learning to Adapt Structured Output Space for Semantic Segmentation," in *CVPR*, 2018, pp. 7472–7481. [Online]. Available: http://arxiv.org/abs/1802.10349

[24] Y. Zhang, Z. Qiu, T. Yao, D. Liu, and T. Mei, "Fully Convolutional Adaptation Networks for Semantic Segmentation," in *CVPR*, 2018, pp. 6810–6818. [Online]. Available: http://arxiv.org/abs/1804.08286

[25] S. Sankaranarayanan, Y. Balaji, A. Jain, S. N. Lim, and R. Chellappa, "Learning from Synthetic Data: Addressing Domain Shift for Semantic Segmentation," in *CVPR*, 2018, pp. 3752–3761. [Online]. Available: http://arxiv.org/abs/1711.06969

[26] X. Peng, Y. Li, Y. L. Murphey, X. Wei, and J. Luo, "Domain Adaptation with One-step Transformation," in *Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018*. IEEE, 2018, pp. 539–546.

[27] W. Hong, Z. Wang, M. Yang, and J. Yuan, "Conditional Generative Adversarial Network for Structured Domain Adaptation," in *CVPR*, 2018, pp. 1–10.

[28] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.

[29] C. Villani, *Optimal transport: old and new*. Springer Science & Business Media, 2008, vol. 338.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[32] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016.