StandardGAN: Multi-source Domain Adaptation for Semantic Segmentation of Very High Resolution Satellite Images by Data Standardization

Onur Tasar[1]   Yuliya Tarabalka[2]   Alain Giros[3]   Pierre Alliez[1]   Sébastien Clerc[4]

[1]Université Côte d'Azur, Inria   [2]LuxCarta   [3]Centre National d'Études Spatiales   [4]ACRI-ST

onur.tasar@inria.fr

Abstract

Domain adaptation for semantic segmentation has recently been actively studied to increase the generalization capabilities of deep learning models. The vast majority of the domain adaptation methods tackle single-source case, where the model trained on a single source domain is adapted to a target domain. However, these methods have limited practical real world applications, since usually one has multiple source domains with different data distributions. In this work, we deal with the multi-source domain adaptation problem. Our method, namely StandardGAN, standardizes each source and target domains so that all the data have similar data distributions. We then use the standardized source domains to train a classifier and segment the standardized target domain. We conduct extensive experiments on two remote sensing data sets, in which the first one consists of multiple cities from a single country, and the other one contains multiple cities from different countries. Our experimental results show that the standardized data generated by StandardGAN allow the classifiers to generate significantly better segmentation.

1. Introduction

Over the years, semantic segmentation of remote sensing data has become an important research topic, due to its wide range of applications such as navigation, autonomous driving, and automatic mapping. In the last decade, a significant progress has been made, especially after *convolutional neural networks (CNNs)* had revolutionized the computer vision community. Among CNNs, *U-net* [26] has gained an increasing attention due to its capability to generate highly precise semantic segmentation from remote sensing data.

Nonetheless, it is a known issue that the performance of U-net or other CNNs immensely depends on the representativeness of the training data [33]. However, in remote sensing, having data that are representative to classify the whole world is challenging, because various atmospheric

(a) City A   (b) City B   (c) City C

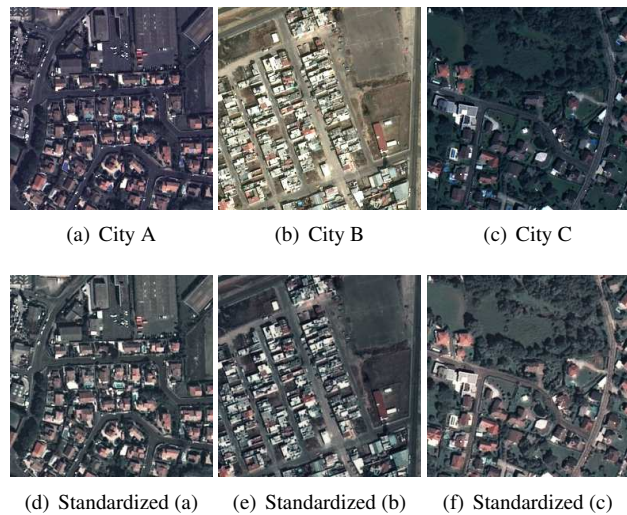(d) Standardized (a)   (e) Standardized (b)   (f) Standardized (c)

Figure 1. Real cities and the standardized data generated by StandardGAN.

effects, intra-class variations, and differences in acquisition usually cause the images collected over different locations to have largely different data distributions. Such differences induce CNNs to generate unsatisfactory segmentation. This problem is referred to as *domain adaptation* in the literature [33]. One way to overcome this issue is to manually annotate a small portion of test data to *fine-tune* the already trained classifier [20]. However, every time when new data are received, annotating even a small portion of them is labor-intensive.

Oftentimes, it is a good practice to perform *data augmentation* [4] to enlarge the training data and to reduce the risk of over-fitting. For example, in remote sensing, color jittering with random gamma correction or random contrast change is commonly used [31]. However, common data augmentation methods are limited to perform complex data transformations, which would greatly help the classifiers to better generalize. A more powerful data augmentation method would be to use *generative adversarial networks (GANs)* [12] to generate fake source domains with the style

of target domain. Here, the main drawback is that the generated samples are representative only for the target domain. However, in multi-source case, we want the generated samples to be representative for all the domains we have at hand. In addition, style transfer needs to be performed between the target and each source domain; therefore, it is inconvenient.

In the field of remote sensing, each satellite image can be regarded as a domain. In our multi-source domain adaptation problem definition, we assume that each source and target domains have significantly different data distributions (see the real data in the first row of Fig. 1). Our method aims at finding a common representation for all the domains by *standardizing* the samples belonging to each domain using GANs. As shown in Fig. 1, in a way, the standardized data could be considered as spectral interpolation across the domains. Adopting such a standardization strategy has two advantages. Firstly, in the training stage, it prevents the classifier from capturing the idiosyncrasies of each source domain. The classifier rather learns from the common representation. Secondly, since in the common representation the samples belonging to source domains and target domain have distributions close to each other, we expect the classifier trained on the standardized source domains to segment well the standardized target domain.

Standardizing multiple domains using GANs raises several challenges. Firstly, when training GANs, one needs real data so that the generator can generate fake data with the distribution that is as close as possible to the distribution of the real data. However, in our case, the standardized data do not exist. In other words, we wish to generate data without showing samples drawn from a similar distribution. Secondly, all the standardized domains need to have similar data distributions. Otherwise, the advantages mentioned above would be lost. Thirdly, the standardized data and the real data themselves must be semantically consistent. For example, when generating the standardized data, the method should not replace some objects by the others, add artificial objects, or remove some objects existing in the real data. Otherwise, the standardized data and the ground-truth for the real data would not match, and we could not train a model. Finally, the method should be efficient. If the number of networks and their structures are not kept as small as possible, depending on the number of domains, we could face with issues in terms of memory occupation and computational time.

In this work, we present novel StandardGAN, which overcomes all the aforementioned challenges. The main contributions are three fold. Firstly, we introduce the use of GANs in the context of data standardization. Secondly, we present a GAN that is able to generate data samples without providing it with data coming from the same or similar distribution. Finally, we propose to apply this multi-source

domain adaptation solution to the semantic segmentation of Pléiades data collected over several geographic locations.

2. Related Work

Adapting the classifier. These methods aim at adapting the classifier to target domain. A common approach is to perform multi-task learning, where one of the tasks is to train a classifier from the source domain via common supervised learning approaches, and the other one is to align the features extracted from both source and target domains by adversarial training [14, 32, 15]. A similar approach [7] has also been applied to remote sensing data (SpaceNet challenge [9]). Other approaches include self learning [35, 40], using task-specific decision boundaries [28], introducing new normalization [25, 22] or regularization methods [27], and adding specific loss functions for domain adaptation [39].

Adapting the inputs. These methods, in general, try to perform image-to-image translation (I2I) or style transfer between domains to generate target stylized fake source data. The fake data are then used to train or to fine-tune the classifier. For example, CyCADA [13] uses CycleGAN [38] to generate target stylized fake source data. CycleGAN has also been applied to aerial images [2]. For the style transfer between satellite images, Tasar *et al*. have recently introduced ColorMapGAN [29] that learns to map each color of the source image to another one, and SemI2I [30] that switches the styles of the source and the target domains. To accomplish the same task, one can also consider using other I2I approaches in the computer vision community such as UNIT [19], MUNIT [17], DRIT [18], or common approaches like histogram matching [11].

Multi-source domain adaptation (MDA). The most straightforward approach would be to perform I2I between each source and target domains to stylize all of the source domains as target domain. However, this method is extremely cumbersome, because the training must be performed for each source domain and the target domain pair. In addition, the data distribution of each source domain is made similar to the distribution of only one domain (i.e., target domain). Instead, finding a common representation that is representative for all the domains is desired. Recently, specifically for MDA, a few methods focusing on image classification have been proposed [36, 34, 23]. However, it may not be possible to extend these works to semantic segmentation, as precisely structured output is required. To address the issue of MDA for semantic segmentation, Zhao *et al*. have proposed MADAN [37], which is an extension of CyCADA, but it is extremely compute-intensive. JCPOT [24] investigates optimal transport for MDA problem. Elshamli *et al*. have recently proposed a method con-
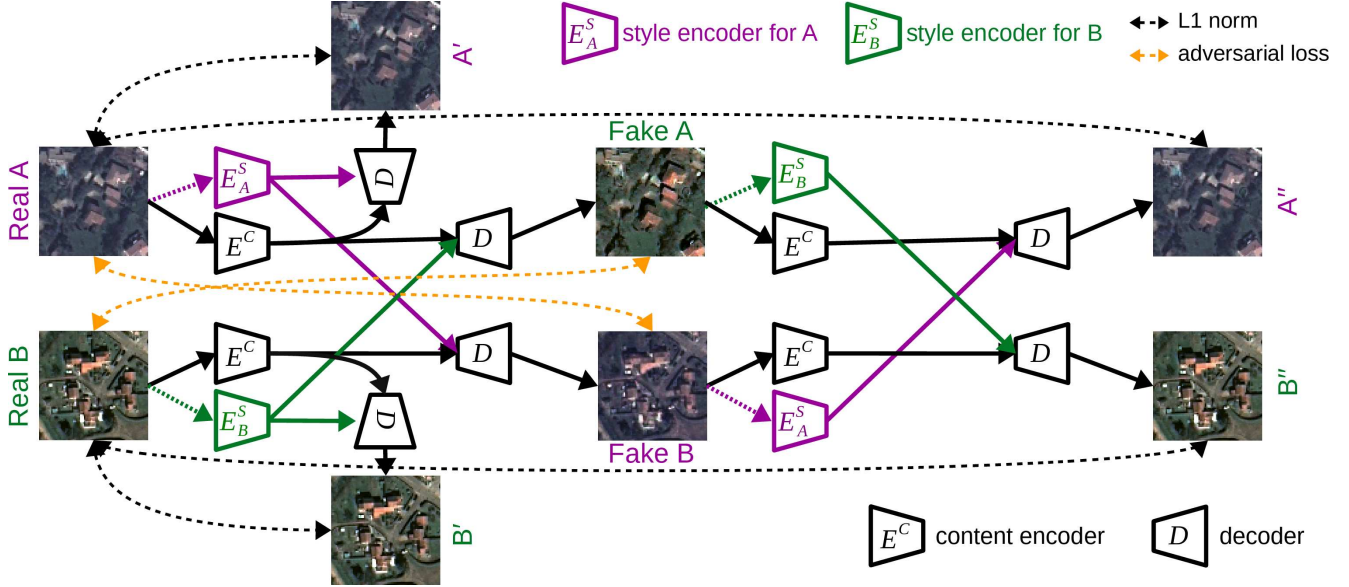
Figure 2. Style transfer between two cities. In this example, there exists 2 style encoders, 1 content encoder, 1 decoder, and 1 discriminator.
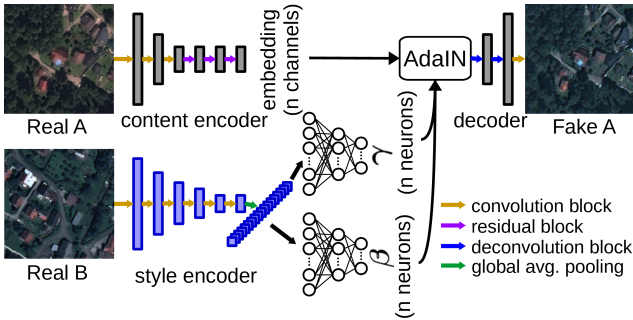

Figure 3. Combining the content of one city with the style of another city.

sisting in patch based networks [8]. However, since the network architectures are not fully convolutional, the method may not be suitable for classes requiring high precision such as buildings and roads.

Data standardization. In machine learning, one of the most commonly used data standardization approach is referred to as Z-score normalization and computed as:

$$x' = \frac{x - \mu}{\sigma}, \tag{1}$$

where x, μ, σ correspond to original data, mean value, and standard deviation. In addition, histogram equalization [11] is also a common pre-processing step. However, these approaches do not take into account the contextual information, they just follow certain heuristics. One may also think of applying color constancy algorithms [1] such as gray-world [3] and gamut [10] approaches. These algorithms assume that colors of the objects are highly affected by the color of the illuminant and try to remove this effect.

3. Method

In this section, we first explain how to perform style transfer between two domains. We then describe how StandardGAN standardizes two domains. Finally, we detail how we extend StandardGAN to multi-domain case.

StandardGAN consists of one content encoder, one decoder, one discriminator, and n style encoders, where n is the number of domains. Fig. 2 illustrates the generator to perform style transfer between two domains. The discriminator performs multi-task learning as in StarGAN [5] by adding an auxiliary classifier on top of the discriminator of CycleGAN [38]. The first task allows the fake source and the target domains to have as similar data distributions as possible, whereas the other task helps the discriminator to understand between which fake and real data it is discriminating. We provide detailed explanations for both tasks in *style transfer* and *classification loss* parts of the following sub-section.

3.1. Style Transfer Between Two Domains

We denote both domains by A and B. In the following, we explain the main steps that are required for style transfer between two domains.

Style Transfer. The goal of style transfer is to generate fake A with the style of B and fake B having a similar data distribution as real A. To perform style transfer, we use two types of encoders. One is domain agnostic content encoder, and the other one is domain specific style encoder. The content encoder is used to map the data into a common space, irrespective of which domain the data come from. On the other hand, the style encoder helps the decoder to generate

output with the style of its specific domain. We use adaptive instance normalization (AdaIN) [16] to combine the content of A with the style of B (or vice versa). AdaIN is defined as:

$$\text{AdaIN}(x, \gamma, \beta) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta, \qquad (2)$$

where x is the activation of the content encoder's final convolutional layer, and γ and β correspond to the parameters that are learned by the style encoder. As can be seen in Eq. 2, γ and β are used to scale and shift the activation, which results in changing the style of the output. After the activation is normalized by AdaIN, as depicted by Fig. 3, it is fed to the decoder to generate the fake data.

In order to force real A and fake B, and real B and fake A to have as similar data distributions as possible, we compute and minimize an adversarial loss between them. We use the adversarial loss functions described in LSGAN [21]. The discriminator adversarial loss between real A and fake B (or real B and fake A) is defined as:

$$\mathcal{L}_{adv\_D} = \mathbb{E}_{x \sim p(x)}[(D_{adv}(x) - 1)^2] + \\ \mathbb{E}_{y \sim p(y)}[(D_{adv}(G(y)))^2] \qquad (3)$$

where $\mathbb{E}$ denotes the expected value, G and D_{adv} stand for the generator and the adversarial output of the discriminator (the first task), and x and y correspond to data for both domains drawn from the distributions of $p(x)$ and $p(y)$. The generator adversarial loss is computed as:

$$\mathcal{L}_{adv\_G} = \mathbb{E}_{y \sim p(y)}[(D_{adv}(y) - 1)^2]. \qquad (4)$$

The overall generator adversarial loss $\mathcal{L}_{adv\_G}$ and the discriminator adversarial loss $\mathcal{L}_{adv\_D}$ are calculated by simply summing the adversarial losses between real A and fake B, and real B and fake A.

Classification loss. To force real A and fake B, and real B and fake A to have similar styles, normally, we need two discriminators. One is used for discriminating between real A and fake B, and the other is responsible for distinguishing between real B and fake A. However, as mentioned in Sec. 1, we want to keep the number of networks as small as possible to easily extend StandardGAN to multi-domain case. In order to use only one discriminator, we adopt the strategy explained in StarGAN [5]. Let us assume that A is the source and B is the target domain. We suppose that the labels of A and B are indicated by c_s and c_t (e.g., $c\_s = 0$ and $c\_t = 1$), and the image patch sampled from A is denoted by x. On top of the discriminator, we add a classifier. Both the discriminator and the generator have a role on this classifier. On the one hand, the discriminator wants the classifier to predict the label of A correctly. On the other hand, the generator tries to generate fake A in a way that

the classifier predicts it as B. The classification loss for the discriminator is defined as:

$$\mathcal{L}_{cls\_D} = \mathbb{E}[-\log D_{cls}(c\_s \mid x)], \qquad (5)$$

where $D_{cls}(c\_s \mid x)$ denotes the probability distribution over domain labels generated by D. By minimizing this function, D learns from which domain x come. The classification loss for the generator is computed as:

$$\mathcal{L}_{cls\_G} = \mathbb{E}[-\log D_{cls}(c\_t \mid G(x))]. \qquad (6)$$

Minimizing this function causes D to label fake A ($G(x)$) as B. We sum the classification losses between real A and fake B, and real B and fake A to compute the overall domain classification losses $\mathcal{L}_{cls\_D}$ and $\mathcal{L}_{cls\_G}$. In the training stage, minimizing Eqs. 5 and 6 allows the discriminator to understand whether it needs to distinguish between real A and fake B or between real B and fake A. As a result, the style transfer can be performed with only one discriminator. The classification loss is particularly useful when we extend StandardGAN to multi-domain adaptation case.

Semantic Consistency. As mentioned in Sec. 1, it is crucial to perform the style transfer without spoiling the semantics of the real data. Otherwise, the fake data and the ground-truth for the real data would not overlap. Thus, they cannot be used to train a model. For this reason, our decoder is architecturally quite simple. It consists of only one convolution and two deconvolution blocks (see Fig. 3). After scaling and shifting the content embedding of one domain with the AdaIN parameters learned by the style encoder from another domain, we directly decode the embedding, instead of adding further residual blocks. Moreover, we have additional constraints enforcing semantic consistency. As shown in Fig. 2, after we generate fake A with the style of B and fake B with the style of real A, we switch the styles once again to obtain A″ and B″. In an ideal case, A and A″, and B and B″ must be the same. Hence, we minimize the cross reconstruction loss $\mathcal{L}_{cross}$ that is the sum of L1 norms between A and A″, and between B and B″. Similarly, when we combine the content information of a domain with its own style information, we should be reconstructing itself (see A′ and B′ in Fig. 2). We also minimize the self reconstruction loss $\mathcal{L}_{self}$, which is computed by summing the L1 norms between A and A′, and between B and B′.

Training. The overall generator loss is calculated as:

$$\mathcal{L}_G = \lambda_1 \mathcal{L}_{cross} + \lambda_2 \mathcal{L}_{self} + \lambda_3 \mathcal{L}_{cls\_G} + \lambda_4 \mathcal{L}_{adv\_G}, \quad (7)$$

where λ_1, λ_2, λ_3, and λ_4 denote the weights for the individual losses. The discriminator loss is defined as:

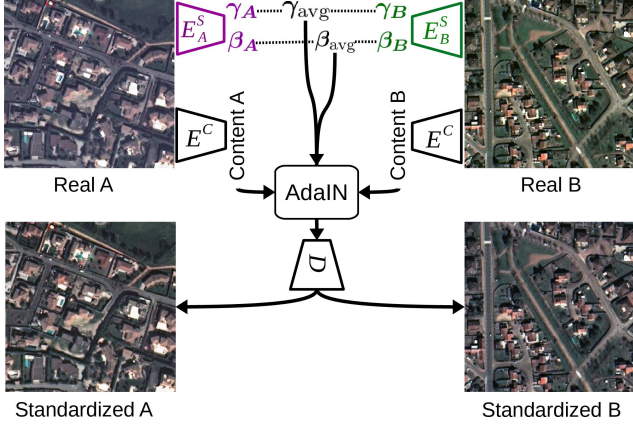$$\mathcal{L}_D = \lambda_3 \mathcal{L}_{cls\_D} + \lambda_4 \mathcal{L}_{adv\_D}. \qquad (8)$$

Figure 4. Standardizing two domains. Dashed lines correspond to arithmetic average.
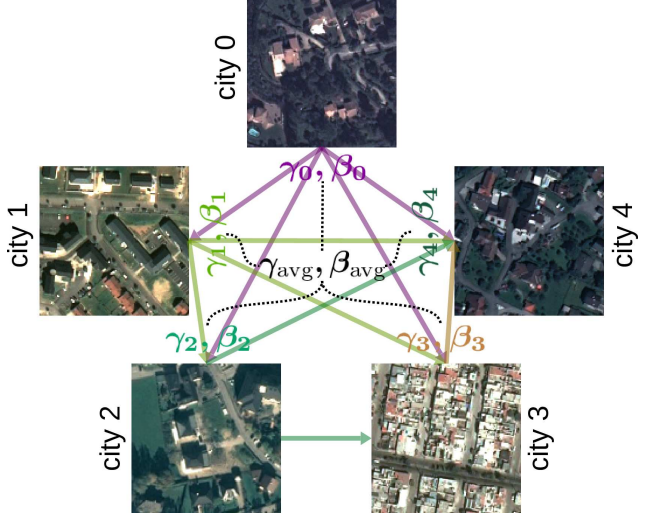

Figure 5. Standardizing multiple domains. Solid arrows represent adaptation between two domains. Dashed lines correspond to arithmetic average. γ_{avg} and β_{avg} are used for standardization.

Algorithm 1: The pseudocode for StandardGAN.

create 1 content encoder, 1 decoder, and 1 discrim.
foreach *domain* **do**
    init. domain specific AdaIN params. with zeros
    create a domain specific style encoder
end
foreach *training iteration* **do**
    $\mathcal{L}_G \leftarrow 0, \mathcal{L}_D \leftarrow 0$; // G and D losses
    for $i \leftarrow 0$ **to** *(# of domains - 1)* **do**
        for $j \leftarrow (i+1)$ **to** *(# of domains - 1)* **do**
            $\mathcal{L}_{G_-} \leftarrow$ G loss between dom. $i \& j$ (Eq. 7)
            $\mathcal{L}_{D_-} \leftarrow$ D loss between dom. $i \& j$ (Eq. 8)
            $\mathcal{L}_G \leftarrow \mathcal{L}_G + \mathcal{L}_{G_-}$
            $\mathcal{L}_D \leftarrow \mathcal{L}_D + \mathcal{L}_{D_-}$
        end
    end
    backprop. $\mathcal{L}_G$ and $\mathcal{L}_D$, $\mathcal{L}_G \leftarrow 0, \mathcal{L}_D \leftarrow 0$
    foreach *domain* **do**
        update dom. spec. AdaIN params. via Eq. 9
    end
    avg. AdaIN params. $\leftarrow$ arithmetic average of
        domain specific AdaIN parameters
end

We minimize $\mathcal{L}_G$ and $\mathcal{L}_D$ simultaneously.

As can be seen in Fig. 3, to generate fake data, content encoder, decoder, and the AdaIN parameters learned by the style encoder of the other domain are required. The issue is that the style encoder produces different AdaIN parameters for each image patch depending on the context of the patch. For instance, we cannot expect patches from a forest and an industrial area to have similar parameters, because they have different styles. For each domain, to capture the global AdaIN parameters, we first initialize domain specific γ and β parameters with zeros. We then propose to update them in each training iteration as:

$$p = 0.95 \times p + 0.05 \times p\_\mathrm{current}, \tag{9}$$

where p is the global domain specific AdaIN parameter (i.e., γ or β) and p_current is the parameter from the current training patch. After a sufficiently long training process, Eq. 9 estimates the global AdaIN parameters for each domain. These estimations can then be used in the test stage.

3.2. StandardGAN for Image Standardization

As mentioned previously, the domain agnostic content encoder learns to map domains into a common space. To generate target stylized fake source data, the content embedding extracted by the content encoder from the source domain is normalized with the global AdaIN parameters of the target domain. The normalized embedding is then given to the decoder to generate the fake data. We have discovered that instead of normalizing the embedding with the AdaIN parameters for one of the domains, if we normalize it with the arithmetic average of the global AdaIN parameters of both domains, StandardGAN learns to generate standardized data. The standardization process for two domains is depicted in Fig. 4. As shown in the figure, real A and real B have considerably different data distributions. On the other hand, standardized A and standardized B look quite similar,

and their data distributions are somewhere between the data distributions of real A and real B.

To standardize multiple domains, we propose Alg. 1. In multi-domain case, c_s and c_t in Eqs. 5 and 6 can range between 0 and n - 1, where n is the number of domains. As shown in Fig. 5, we perform adaptation between each pair of domains. We then take the average of the global AdaIN parameters of each domain and use the average to normalize

Table 1. The data set.

City (Country)	Class percentages (%)			Area (km²)
	building	road	tree	
Bad Ischl (AT)	5.51	6.0	35.38	27.71
Salzburg Stadt (AT)	9.44	8.69	23.88	134.71
Villach (AT)	9.26	10.63	19.91	43.59
Lienz (AT)	6.96	8.16	15.37	28.38
Sankt Pölten (AT)	6.68	6.39	25.13	87.17
Bourges (FR)	9.81	10.52	14.83	72.20
Lille (FR)	18.36	12.71	15.40	117.58
Vaduz (LI)	3.57	4.30	33.69	96.08

Table 2. Training time of StandardGAN for both experiments.

GPU	Exp.	# of patches	Tr. time (secs.)
Nvidia Tesla	1	5712	6077.82
V100 SMX2	2	8226	9929.52

the embeddings extracted by the content encoder from all the domains. We finally decode the normalized embeddings via the decoder to generate the standardized data.
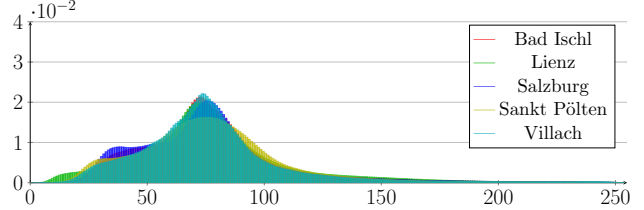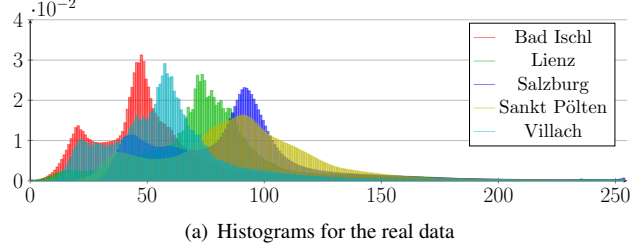
4. Experiments

In our experiments, we use Pléiades images captured from 5 cities in Austria, 2 cities in France, and 1 city in Liechtenstein. The spectral channels consist of red, green, and blue bands. The spatial resolution has been reduced to 1 m by the data set providers. The annotations for building, road, and tree classes have been provided [1]. Table 1 reports, for each city, the name of the city, percentage of the pixels belonging to each class, and the total covered area.

We have two experimental setups. In the first experiment, we use the images from Salzburg Stadt, Villach, Lienz, and Sankt Pölten for training and the image from Bad Ischl for test. In the second experiment, we choose Salzburg Stadt, Villach, Bourges, and Lille as the training cities and Vaduz as the test city. In the first experiment, we want to observe how well our method generalize to a new city from the same country. On the other hand, the goal of the second experiment is to investigate the generalization abilities of our approach when training and test data come from different countries. Let us also remark that, as confirmed by Table 1, classes in the test cities (i.e., Bad Ischl and Vaduz) are highly imbalanced, which makes the domain adaptation problem even more difficult. For example, in both cases, the number of pixels labeled as tree is significantly larger than the number of pixels labeled as building and road.

In the pre-processing step, we split all the cities into 256×256 patches with 32 pixels of overlap. We set $\lambda_1, \lambda_2, \lambda_3$, and λ_4 in Eqs. 7 and 8 to 10, 10, 1, and 1, respectively. We train StandardGAN for 20 epochs with Adam

(a) Histograms for the real data

(b) Histograms for the standardized data generated by StandardGAN

Figure 6. Histograms for green band of the cities used in the first experiment. (a) Before standardization, (b) After standardization.

Table 3. IoU scores for Bad Ischl (the first experiment).

	Method	building	road	tree	Overall
U-net on data by	U-net	45.36	18.81	**82.43**	48.87
	Gray-world	49.39	42.25	66.31	52.65
	Hist. Equaliz.	45.33	39.07	73.03	52.48
	Z-score norm.	51.22	46.56	77.62	58.47
	StandardGAN	**56.41**	**50.26**	80.59	**62.42**

Table 4. IoU scores for Vaduz (the second experiment).

	Method	building	road	tree	Overall
U-net on data by	U-net	29.83	26.42	46.25	34.16
	Gray-world	27.95	31.13	36.65	31.91
	Hist. Equaliz.	21.21	19.19	51.93	30.78
	Z-score norm.	29.94	29.87	41.98	33.93
	StandardGAN	**54.86**	**42.43**	**63.09**	**53.46**

optimizer, where the initial learning rate is 0.0002, the exponential decay rates for the moment estimates are 0.5 and 0.999, respectively. In each training iteration of Standard-GAN, we randomly sample 1 patch from each domain. After the 10th epoch, we progressively reduce the learning rate in each epoch as:

$$\text{learn. rate} = \text{init\_lr} \times \frac{\text{num\_epochs} - \text{epoch\_no}}{\text{num\_epochs} - \text{decay\_epoch}}, \quad (10)$$

where init_lr, num_epochs, epoch_no, and decay_epoch correspond to the initial learning rate (0.0002 in our case), the total number of epochs (we set it to 20), the current epoch no, and the epoch no in which we start reducing the learning rate (we determine it as 10). Table 2 reports the total number of training patches in both experiments and the training time of StandardGAN. We first standardize all the data. We then train a model on the standardized source data and classify the standardized target data. We compare

Figure 7. Real data used in the first experiment and the outputs generated by StandardGAN. Left column: the real data. Matrix on the right: The standardized data are highlighted by red bounding boxes. The rest of the cells depict the ith domain with the style of jth domain. The domain ids are indicated inside parentheses.
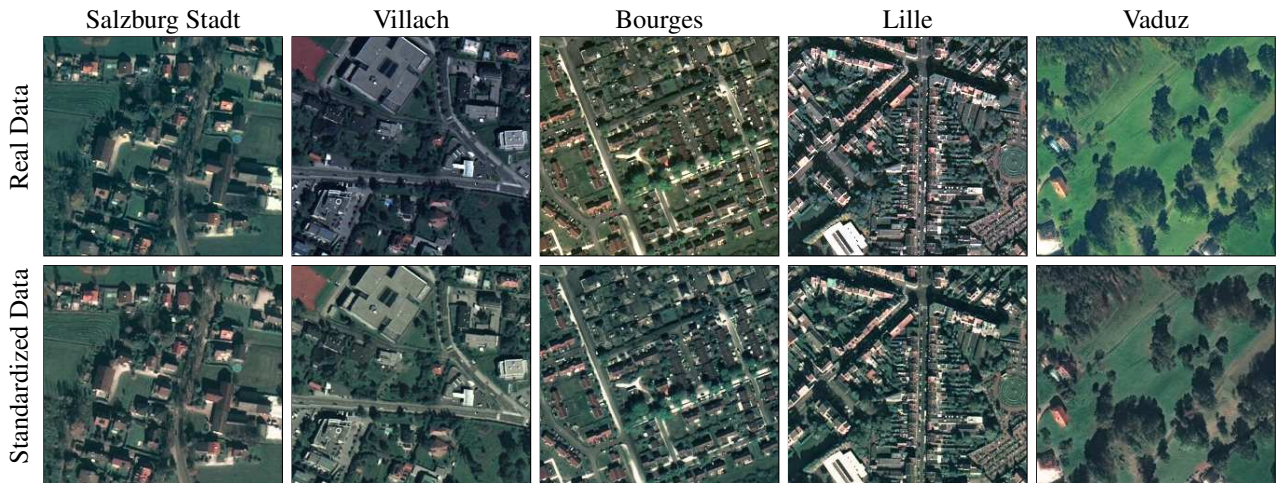

Figure 8. Real cities used in the second experiment, and the standardized data generated by StandardGAN.
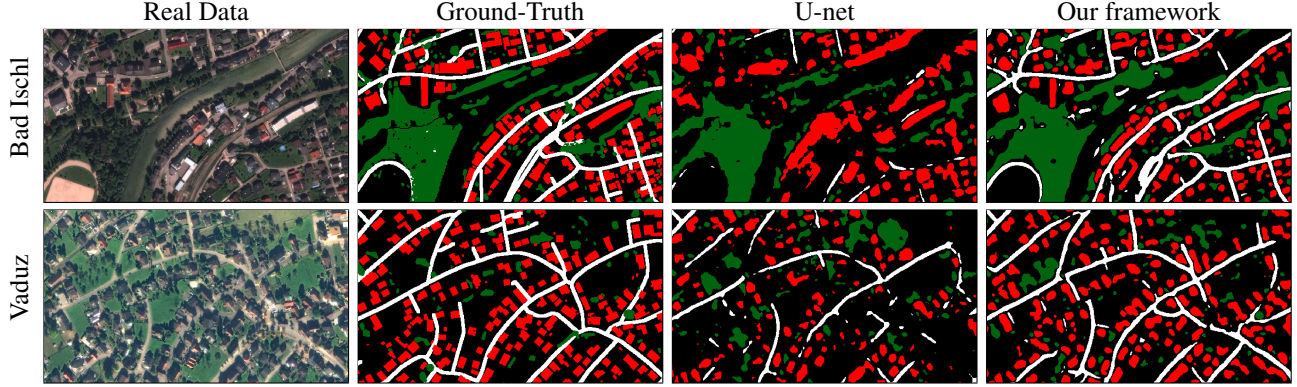
Figure 9. Comparison between the traditional U-net and our framework. Red, green, and white pixels represent building, road, and tree classes, respectively. The pixels in black do not belong to any class.

our approach with the other standardization algorithms described in Sec. 2, namely gray-world [3], histogram equalization [11], and Z-score normalization (Eq. 1). We use U-net [26] as the classifier. We also provide the experimental results for naive U-net without applying any domain adaptation methods. For each comparison, we train a U-net for 35 epochs via Adam optimizer with the learning rate of 0.0001 and the exponential decays rates of 0.9 and 0.999. In each training iteration of U-net, we use a mini-batch of 32 randomly sampled patches. We perform online data augmentation with random rotations and flips.

In Fig. 7, we depict close-ups from the cities used in the first experiment and the fake data generated by Standard-GAN. Note that to train a model, we do not use the target stylized source data, we use only the standardized data that are highlighted by red bounding boxes in the figure. The style transfer between each domain is the prior step to the standardization. We can clearly observe that there exists a substantial difference between the data distributions of the real data, whereas the standardized data look similar. Moreover, Fig. 6 verifies that color histograms of the standardized data are considerably closer to each other than those of the real data. Fig. 8 shows closeups from the cities in the second experiment and their standardized versions by StandardGAN. The standardized and the real data for Salzburg Stadt and Lille seem quite similar. The reason is the data distributions of these two cities are already somewhere between the distributions of all five cities. However, the radiometry of Villach, Bourges, and Vaduz significantly changes after the standardization process. Besides, all the standardized data have similar data distributions.

Tables 3 and 4 report the intersection over union (IoU) [6] values for both experiments. The training data acquired over a single country are usually more representative for a city from the same country than a city from another country. For this reason, the quantitative results for the first experiment are generally higher. Besides, in some cases, the representativeness of the samples belonging to different classes may vary. For instance, in the first experiment, the traditional U-net already exhibits a relatively good performance for tree class, as the tree samples from the source domains represent well the samples in the target data. For this class, the performance of our method is slightly worse. It is probably because of some artifacts generated by the proposed GAN architecture when standardizing the domains. On the other hand, for the other classes, our approach achieves a better performance than all the other methods. In the second experiment, unlike the first one, none of the class samples in the source domains are representative for the target domain. Hence, the performance of U-net is poor. In addition, the common heuristic based pre-processing methods do not help improving the results. However, the StandardGAN better allow the classifier to generalize completely different geographic locations. Fig. 9 illustrates the improvement of our framework against the naive U-net in terms of predicted maps.

5. Concluding Remarks

In this study, we presented novel StandardGAN, which is a new pre-processing approach proposed with the purpose of standardizing multiple domains. In our experiments, we verified that the standardized data generated by Standard-GAN enable the classifier to significantly better generalize to new Pléiades data. Note that StandardGAN has only one encoder, one discriminator, one decoder, and n style encoders. Although there are multiple style encoders, their architecture is fairly simple. Thus, it is feasible to use StandardGAN to standardize larger number of domains than the number of cities in our experiments. As future work, we plan to use StandardGAN for adaptation of more domains and for other types of remote sensing data such as Sentinel, aerial, and hyper-spectral images. In addition, we plan to investigate whether StandardGAN could be used for other real-world applications such as change detection.

References

[1] V. Agarwal, B. R. Abidi, A. Koschan, and M. A. Abidi. An overview of color constancy algorithms. *Journal of Pattern Recognition Research*, 1(1):42–54, 2006. 3

[2] B. Benjdira, Y. Bazi, A. Koubaa, and K. Ouni. Unsupervised domain adaptation using generative adversarial networks for semantic segmentation of aerial images. *Remote Sensing*, 11(11):1369, 2019. 2

[3] G. Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin institute*, 1980. 3, 8

[4] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin. Albumentations: fast and flexible image augmentations. *arXiv preprint arXiv:1809.06839*, 2018. 1

[5] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018. 3, 4

[6] G. Csurka, D. Larlus, and F. Perronnin. What is a good evaluation measure for semantic segmentation?. In *British Machine Vision Conference*, volume 27, page 2013, 2013. 8

[7] X. Deng, H. L. Yang, N. Makkar, and D. Lunga. Large scale unsupervised domain adaptation of segmentation networks with adversarial learning. In *IEEE International Geoscience and Remote Sensing Symposium*, pages 4955–4958, 2019. 2

[8] A. Elshamli, G. W. Taylor, and S. Areibi. Multisource domain adaptation for remote sensing using deep neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 2019. 3

[9] A. V. Etten, D. Lindenbaum, and T. Bacastow. SpaceNet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*, 2018. 2

[10] D. A. Forsyth. A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1):5–35, 1990. 3

[11] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Pearson International Edition, 2006. 2, 3, 8

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 1

[13] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017. 2

[14] J. Hoffman, D. Wang, F. Yu, and T. Darrell. FCNs in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 2

[15] H. Huang, Q. Huang, and P. Krahenbuhl. Domain transfer through deep activation matching. In *Proceedings of the European Conference on Computer Vision*, pages 590–605, 2018. 2

[16] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 4

[17] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision*, pages 172–189, 2018. 2

[18] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European Conference on Computer Vision*, pages 35–51, 2018. 2

[19] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017. 2

[20] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):645–657, 2016. 1

[21] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017. 4

[22] X. Pan, P. Luo, J. Shi, and X. Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision*, pages 464–479, 2018. 2

[23] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019. 2

[24] I. Redko, N. Courty, R. Flamary, and D. Tuia. Optimal transport for multi-source domain adaptation under target shift. *arXiv preprint arXiv:1803.04899*, 2018. 2

[25] R. Romijnders, P. Meletis, and G. Dubbelman. A domain agnostic normalization layer for unsupervised adversarial domain adaptation. In *Winter Conference on Applications of Computer Vision*, pages 1866–1875. IEEE, 2019. 2

[26] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, pages 234–241. Springer, 2015. 1, 8

[27] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Adversarial dropout regularization. *arXiv preprint arXiv:1711.01575*, 2017. 2

[28] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018. 2

[29] O. Tasar, S L Happy, Y. Tarabalka, and P. Alliez. ColorMapGAN: Unsupervised domain adaptation for semantic segmentation using color mapping generative adversarial networks. *IEEE Transactions on Geoscience and Remote Sensing*, 2020. 2

[30] O. Tasar, S L Happy, Y. Tarabalka, and P. Alliez. SemI2I: Semantically consistent image-to-image translation for domain adaptation of remote sensing data. *arXiv preprint arXiv:2002.05925*, 2020. 2

[31] O. Tasar, Y. Tarabalka, and P. Alliez. Incremental learning for semantic segmentation of large-scale remote sensing data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(9):3524–3537, 2019. 1

[32] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7472–7481, 2018. 2

[33] D. Tuia, C. Persello, and L. Bruzzone. Domain adaptation for the classification of remote sensing data: An overview of recent advances. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):41–57, 2016. 1

[34] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3964–3973, 2018. 2

[35] J. Zhang, C. Liang, and C.-C. J. Kuo. A fully convolutional tri-branch network (fctn) for domain adaptation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 3001–3005. IEEE, 2018. 2

[36] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon. Adversarial multiple source domain adaptation. In *Advances in neural information processing systems*, pages 8559–8570, 2018. 2

[37] S. Zhao, B. Li, X. Yue, Y. Gu, P. Xu, R. Hu, H. Chai, and K. Keutzer. Multi-source domain adaptation for semantic segmentation. In *Advances in Neural Information Processing Systems*, pages 7285–7298, 2019. 2

[38] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 2, 3

[39] X. Zhu, H. Zhou, C. Yang, J. Shi, and D. Lin. Penalizing top performers: Conservative loss for semantic segmentation adaptation. In *Proceedings of the European Conference on Computer Vision*, pages 568–583, 2018. 2

[40] Y. Zou, Z. Yu, B. V. K. Kumar, and J. Wang. Domain adaptation for semantic segmentation via class-balanced self-training. *arXiv preprint arXiv:1810.07911*, 2018. 2