CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

Journal Paper

Data-Efficient Domain Adaptation for Semantic Segmentation of Aerial Imagery Using Generative Adversarial Networks

Bilel Benjdira
Adel Ammar
Anis Koubaa
Kais Ouni

CISTER-TR-200208

Data-Efficient Domain Adaptation for Semantic Segmentation of Aerial Imagery Using Generative Adversarial Networks

Bilel Benjdira, Adel Ammar, Anis Koubaa, Kais Ouni

CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail:

https://www.cister-labs.pt

Abstract

Despite the significant advances noted in semantic segmentation of aerial imagery, a considerable limitation is blocking its adoption in real cases. If we test a segmentation model on a new area that is not included in its initial training set, accuracy will decrease remarkably. This is caused by the domain shift between the new targeted domain and the source domain used to train the model. In this paper, we addressed this challenge and proposed a new algorithm that uses Generative Adversarial Networks (GAN) architecture to minimize the domain shift and increase the ability of the model to work on new targeted domains. The proposed GAN architecture contains two GAN networks. The first GAN network converts the chosen image from the target domain into a semantic label. The second GAN network converts this generated semantic label into an image that belongs to the source domain but conserves the semantic map of the target image. This resulting image will be used by the semantic segmentation model to generate a better semantic label of the first chosen image. Our algorithm is tested on the ISPRS semantic segmentation dataset and improved the global accuracy by a margin up to 24% when passing from Potsdam domain to Vaihingen domain. This margin can be increased by addition of other labeled data from the target domain. To minimize the cost of supervision in the translation process, we proposed a methodology to use these labeled data efficiently.

applied sciences

MDPI

Article

Data-Efficient Domain Adaptation for Semantic Segmentation of Aerial Imagery Using Generative Adversarial Networks

Bilel Benjdira [1,2,*], Adel Ammar [1], Anis Koubaa [1,3] and Kais Ouni [2]

[1] Robotics and Internet of Things Laboratory, College of Computer and Information Sciences, Prince Sultan University, Riyadh 11586, Saudi Arabia; aammar@psu.edu.sa (A.A.); akoubaa@psu.edu.sa (A.K.)
[2] Research Laboratory Smart Electricity & ICT, SEICT, LR18ES44, National Engineering School of Carthage, University of Carthage, Tunis 2035, Tunisia; kais.ouni@enicarthage.rnu.tn
[3] CISTER, INESC-TEC, ISEP, Polytechnic Institute of Porto, 4200-465 Porto, Portugal
* Correspondence: bbenjdira@psu.edu.sa

Received: 17 December 2019; Accepted: 29 January 2020; Published: 6 February 2020

Abstract: Despite the significant advances noted in semantic segmentation of aerial imagery, a considerable limitation is blocking its adoption in real cases. If we test a segmentation model on a new area that is not included in its initial training set, accuracy will decrease remarkably. This is caused by the domain shift between the new targeted domain and the source domain used to train the model. In this paper, we addressed this challenge and proposed a new algorithm that uses Generative Adversarial Networks (GAN) architecture to minimize the domain shift and increase the ability of the model to work on new targeted domains. The proposed GAN architecture contains two GAN networks. The first GAN network converts the chosen image from the target domain into a semantic label. The second GAN network converts this generated semantic label into an image that belongs to the source domain but conserves the semantic map of the target image. This resulting image will be used by the semantic segmentation model to generate a better semantic label of the first chosen image. Our algorithm is tested on the ISPRS semantic segmentation dataset and improved the global accuracy by a margin up to 24% when passing from Potsdam domain to Vaihingen domain. This margin can be increased by addition of other labeled data from the target domain. To minimize the cost of supervision in the translation process, we proposed a methodology to use these labeled data efficiently.

Keywords: deep learning; domain adaptation; semantic segmentation; generative adversarial networks; convolutional neural networks; aerial imagery

1. Introduction

The semantic segmentation task provides for every pixel in the input image a label that defines its semantic class. In remote sensing context, the semantic segmentation of aerial images has an increasing potential for many tasks and applications, like analysis and management of road traffic, monitoring of urban and rural areas, fast interactions in case of emergency, and so on. The growing adoption of Unmanned Aerial Vehicles (UAVs) is behind this increasing potential. High-resolution images can be collected using UAVs from different points of view and processed by the semantic segmentation algorithms to promote the automatic analysis of surveyed scenes.

Since the emergence of Convolutional Neural Networks (CNNs), the area of image analysis algorithms has shown a considerable improvement in accuracy [1–7]. This affected directly the

area of semantic segmentation and paved the way towards a variety of CNN-based architectures, like SegNet [8], fully connected network (FCN), PSPNet [9], U-Net [10] and DeepLab [11]. Empirically, if we build a robust dataset and we train on it one of the state-of-the-art algorithms, we will get an accuracy that easily surpasses 80%.

Despite this exciting efficiency, a notable challenge is blocking the use of semantic segmentation algorithms in real cases. In fact, the accuracy of the model will be high only on images belonging to the same domain of the dataset used in training (object representation, resolution, sensor type, lighting conditions). If we test this model in another domain (images collected under conditions different from those of the training set), the accuracy decreases remarkably. This decrease is caused by the domain shift existing between the target and the source domain. Figure 1 shows a real case scenario in which we want to test the semantic segmentation model on a domain different from the source. A considerable shift is remarked between the two domains (location is changed, image coding is changed, resolution is changed).

The straightforward solution to mitigate this intriguing limitation is to train our model on a labeled dataset constructed from the target domain in a supervised or a semi-supervised way [12]. However, this method is costly and time-consuming. To illustrate this, we can note that pixel-labeling of an image from Cityscapes dataset takes nearly 90 minutes on average [13]. The size of the image is 2040 by 1016 pixels. To reduce the labeling time, we can benefit from human crowd intelligence by distributing the labeling task over a set of human crowds [14]. Every set will be allocated to the labeling of one class. This reduces the cost of generating the semantic labeled dataset on the target. Nevertheless, since such solutions are not always immediately available, it is still useful to search for a data-efficient solution of domain adaptation that uses only a minimal set of images for supervision of the domain transfer.

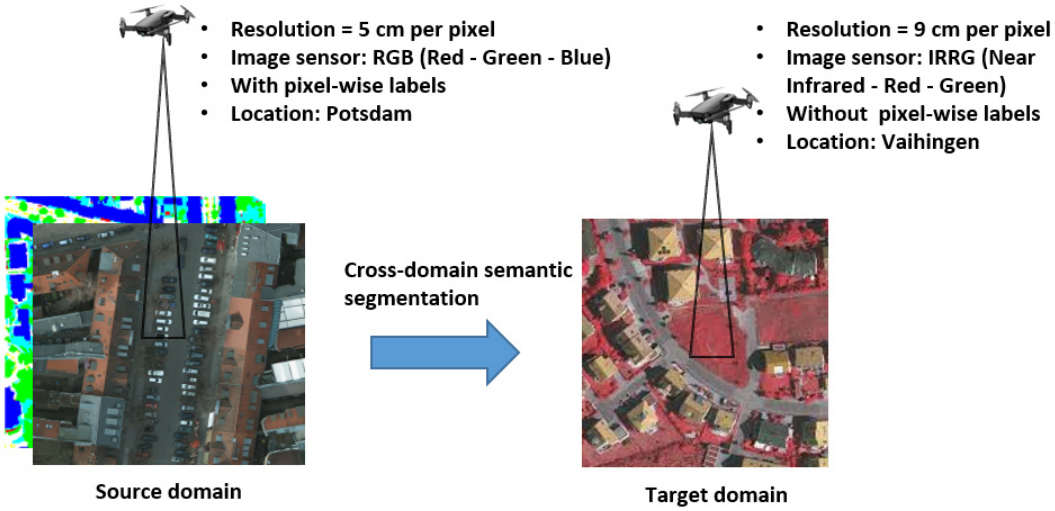

Figure 1. Cross-domains semantic segmentation of aerial imagery.

Domain adaptation is a separate area in machine learning whose objective is to learn how a model performance on a source data distribution can be improved on another target data distribution. Domain adaptation helps in mitigating the domain shift between the target domain data and the source domain data used in training. Typically, a mapping function is designed between the target domain and the source domain. Deep learning models have been used by the recent methods of domain adaptation for the training of such a mapping function [15–18].

Motivated by the current breakthrough made by GANs (Generative Adversarial Networks) [19,20], we developed a data-efficient domain adaptation algorithm based upon an architecture of two GAN networks. Our method aims at handling the case indicated in Figure 1 as well as other cases of the same nature. Our technique is based on converting the image that we want to segment from the

target domain to the source domain by passing it through two consecutive GAN networks. The two generative adversarial networks are trained separately. The first GAN network converts the chosen image from the target domain to a semantic segmentation label. The second GAN network converts this semantic label into the source domain. The generated image is proven to conserve the semantic map of the first image and to mimic the source domain characteristics. It will be used as an input to the segmentation model already trained on the source dataset and will generate a better segmentation label, as proven by the experiments we made. The accuracy of final segmentation using our algorithm can be increased by addition of labeled data from the target domain. A methodology is described to use the labeled data efficiently to minimize the cost of supervision in our algorithm. Our work has the following contributions: (1) Our approach is confirmed to mitigate the problem of domain shift for semantic segmentation by a significant margin that can increase with an efficient addition of labeled data from the target domain. (2) The method is validated using ISPRS semantic labeling dataset through the establishment of cross-domain semantic segmentation between Vaihingen and Potsdam datasets. (3) GANs are introduced as a favorable solution for analyzing aerial imagery.

The following is the organization of our paper: Section 2 provides a summary of the related works in the domain adaptation area within semantic segmentation. Section 3 will make an introduction to GANs. Section 4 will describe the different parts of our proposed method. Section 5 will present the experiments made to validate our method and discuss its effectiveness within domain adaptation of semantic segmentation in aerial imagery. Finally, our work is concluded in Section 6, which also deduces the possible extensions of our method.

2. Related Works

In this section, we will discuss the works that treated domain adaptation within semantic segmentation. Generally, it is assumed, in the machine learning context, that the test and the training datasets are belonging to the same distribution. However, there is substantial discordance between them in real cases. This discordance reduces the model's efficiency out of its training domain. Domain adaptation techniques are used mitigate this discordance and to make the model generalizes better over multiple domains.

In computer vision, efforts made on domain adaptation have been more focused on regression and classification tasks [21] not on semantic segmentation. For example, many works treated the training of the models on online images for the classification of objects in reality [22]. Recent works in this area are focusing on improving the adaptability of deep learning algorithms [15,16,23–25].

In semantic segmentation, most of the domain adaptation works are geared towards the use of simulated data [26–31]. In fact, domain adaptation was expected to be used by these works for the improvement of the efficiency of segmentation on real images through the training of the semantic segmentation model on pure synthetic data. FCNs in the wild [32] is one of the earliest works. It uses a pixel-level adversarial loss to guide the model to learn the domain-invariant properties. This aims at making the adversarial classifier not to differentiate between the target and the source domains. The goal is to make its performance equal on the two domains. Hoffman et al. [26] designed CyCADA, which is a model that changes synthetic data (source domain images) into real data style (target domain images) using CycleGAN. The segmentation model is then fed with the converted images to improve its accuracy on dealing with real photos. On the other hand, Zhang et al. [33] pointed out that a curriculum-style learning technique helps in reducing the domain shift. They deduced the target data properties by joining the learning of global label distribution with the learning of local distributions over landmark superpixels. The segmentation network was then trained through regularizing it to follow these features. Sankaranarayanan et al. [34] treated the problem differently by taking the source and the target images as input to an auto-encoder network that processes and regenerates them before feeding them to the segmentation model. CGAN architecture is another approach introduced by Tsai et al. [35] that makes addition of random noise into the source data giving it as input to the segmentation model. They proved that this technique enhances the model's efficiency on the target

domains. Huang et al. [36] proposed to train independently two networks for the target and the source domains. The training of the target model is done through its regression into the source model weights since the target domain is without labels. It should be noted that the calculation of an adversarial loss in each layer of both networks is done.

Aerial imagery has its peculiarities that should be considered. Therefore, a dedicated work that takes into consideration its peculiarities should be investigated. Recently, three works targeted the domain adaptation in semantic segmentation of aerial imagery. All of them used an unsupervised approach. The first work is the algorithm proposed by Benjdira et al. [4]. They introduced a GAN architecture to map images in an unsupervised way from the source domain to the target domain without the need for paired data. After that, they translated the source domain dataset into the target domain. Then, the translated dataset is used to fine-tune the segmentation model to enhance its ability to treat images from the target domain. The second work is the work proposed by Tasar et al. [37]. They adopted the same algorithm of Benjdira et al. [4] but changed the GAN architecture into another architecture named ColorMapGAN. This architecture converts the source images into images that the spectral distribution is similar to the target spectral distribution but conserves the semantic information of the source. The third work is introduced by Fang et al. [38]. They made a category-sensitive domain adaptation (CsDA) using a geometry-consistent GAN (GcGAN) embedded into a co-training adversarial learning network (CtALN). Their method is currently the state of the art in unsupervised domain adaptation in semantic segmentation of aerial imagery. Up to our knowledge, no one has treated the domain adaptation in semantic segmentation of aerial imagery using a supervised approach. We aim in this work to target this limitation and to study the implementation of a supervised domain adaptation algorithm based on the concatenation of two GAN networks. We only use a reduced amount of labeled data to guide the model during the training. To demonstrate our algorithm's efficiency, we tested it on the ISPRS semantic segmentation dataset [39]. We performed the domain adaptation for a semantic segmentation network from Potsdam (as source) to Vaihingen (as target).

3. Generative Adversarial Networks (GANs)

3.1. The Generator and the Discriminator

The popularity of GANs has continued to increase because they have many addressable applications. Goodfellow et al. [19] introduced GAN in 2014. A GAN is composed of two separate networks: the generator and the discriminator. During the training process, the generator learns how to generate data that mimics real data, whereas the discriminator learns how to differentiate real data from fake data produced by the generator. The training is operated jointly so that both are competing and playing an adversarial zero-sum game.

During the training, the generator is trying continuously to generate fake data that deceives the discriminator so that it classifies them as real. However, the discriminator is trying continuously to detect the fake data and not to classify them as real. To solve the adversarial zero-sum game during the training, game theory theorems are used. Figure 2 indicates the standard architecture of the GAN network.

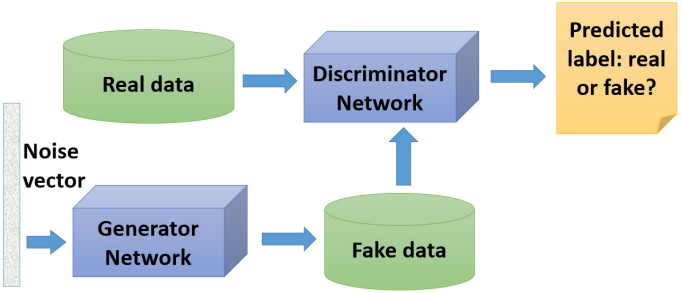

Figure 2. Standard architecture of the GAN.

During training, both networks engage in competition until a Nash equilibrium is reached. In game theory, Nash equilibrium is a status where no player is able improve or deviate his payoff [40].

Equation (1) shows the objective function of GAN:

$$min_G max_D V(Dis, Gen) = \mathbb{E}_{X \sim P_{real\_data}(X)}[logDis(X)] + \mathbb{E}_{z \sim P_z(z)}[log(1 - Dis(Gen(z)))] \quad (1)$$

where Gen denotes the generator cost function which is trained through the maximization of $Dis(Gen(z))$. On the other hand, Dis represents the discriminator cost function which is trained through the minimization of $Dis(Gen(z))$. While X is the image that is obtained from the distribution of real data p_{real_data}. The noise vector obtained from distribution p_z is denoted by z whereas $Gen(z)$ indicates the fake image that is produced by the generator. Additionally, $\mathbb{E}_{X \sim P_{data}(X)}$ represents the expectation on X, which is acquired by the distribution $P_{real\_data}(X)$. Both Dis and Gen play the two-player minimax game using the value function $V(Gen, Dis)$ [19].

Generative adversarial networks have many applications and implementations [41] with image-to-image translation being the application that may be the most attractive to be used in the domain adaptation's context. The use of GANs in the area of image-to-image translation is further discussed in the following subsection.

3.2. translation from Image-to-Image using GANs

The translation of an image from one domain to another has been the target for many GAN architectures [20,42–44]. Translation of images can be paired [20] or unpaired [45].

3.2.1. Paired Image-to-Image Translation

In this particular area, the GAN model must be trained using labeled pairs of images. With X being the source data, Y being the target data and N being the number of samples in each dataset, each pair of corresponding images $\{x_i\}_{i=0...N}$ and $\{y_i\}_{i=0...N}$ will be used by the model to learn in a supervised way how to translate between X and Y domains. Currently, Pix2pix [20] is the most known model for paired image-to-image translation.

3.2.2. Unpaired Image Translation

GAN is used in unpaired image translation to convert between two sets of images through an unsupervised training. With X being the source data, Y being the target data, N being the number of samples in the source dataset and M the number of samples in the target dataset, $\{x_i\}_{i=0...N}$ and $\{y_i\}_{i=0...M}$ do not correspond with each other and can randomly be obtained from the related domain set. Currently, CycleGAN [45] is the most known model for unpaired image-to-image translation.

4. Proposed Method

4.1. The GAN Architecture

Our method aims at translating images from the target domain to the source domain using two GAN networks as illustrated in Figure 3. The entire procedure makes the target domain images imitate the source domain characteristics, which include the quality of images, types of sensors and resolution, among others. The mapping process between the target and the source is done in two steps. First, the chosen image from the target is mapped into semantic label using the first GAN Network. Then, the generated label is mapped using the second GAN network into another image that looks like images from the source domain distribution. The two networks are trained separately using paired datasets. We used a modified architecture from the standard GAN that is inspired by some recent architectures [20,45].

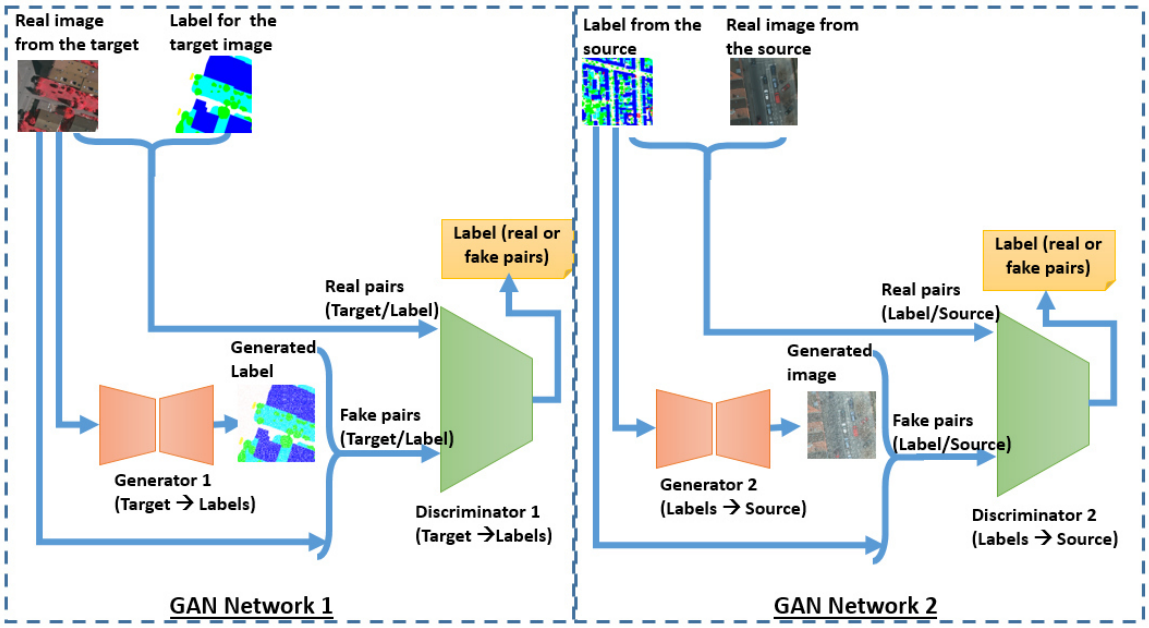

Figure 3. The GAN architecture.

The first GAN Network is designed by substituting the noise vector in the traditional GAN Network by images from the target distribution. The data generated by this GAN Network are the semantic label of the entry image. The training process is done using a small-sized dataset constructed from a few samples from the target domain and their semantic labels. The mapping function of the generator, $G : X \rightarrow L$, learns through an adversarial training how to produce the semantic label L of the target image X. Because the target is supposed to have a minimal set of labels, we assume that the provided labeled images are only a few significant samples from the target dataset. We mean by significant samples the existence of two constraints during the selection of the images. First, the sampled images should contain all the semantic classes of the dataset. Second, for each class, we should choose samples with the most available representations of the class inside the dataset. The discriminator D learns during the adversarial training how to differentiate between real pairs $(X, L_{original})$ and the fake pairs $(X, L_{generated})$. While the discriminator improves its ability to detect fake pairs from real pairs, the generator improves its ability to generate the true semantic labels of the input image. The architecture of the generator and the discriminator in the first GAN network is illustrated in Figure 4.

Concerning the generator, it is an encoder-decoder architecture with skip-connections. It is similar to U-Net [10] architecture. Eight convolutional layers are used for downsampling in the encoder part and similarly, eight deconvolutional layers are used for upsampling in the decoder part. Leaky ReLu [46] is used as the activation layer for all the layers of the encoder except the first one. Leaky ReLu is identical to the standard ReLu in the positive region. However, in the negative part, it has a small slope α. Explicitly, it is defined as $LeakyReLu(x) = x$, if $x >= 0$; and as $LeakyReLu(x) = \alpha x$, if $x < 0$. α has a small value, which gives small gradient where $x < 0$. The encoder part ends by giving us a small feature vector of size $(1 \times 1 \times 512)$. This feature vector will pass through the decoder part to rebuild the original feature vector. We used skip connections to concatenate every layer output in the encoder part with the corresponding layer in the decoder part. ReLu is used as the activation function for all the layers of the decoder. Batch normalization [47] is used after every layer of network except the first layer of the encoder and the last layer of the decoder. We used dropout [48] in the first three layer of the decoder to reduce overfitting. We apply *tanh* as an activation function to the last layer of the decoder to get the predicted label for the input image.

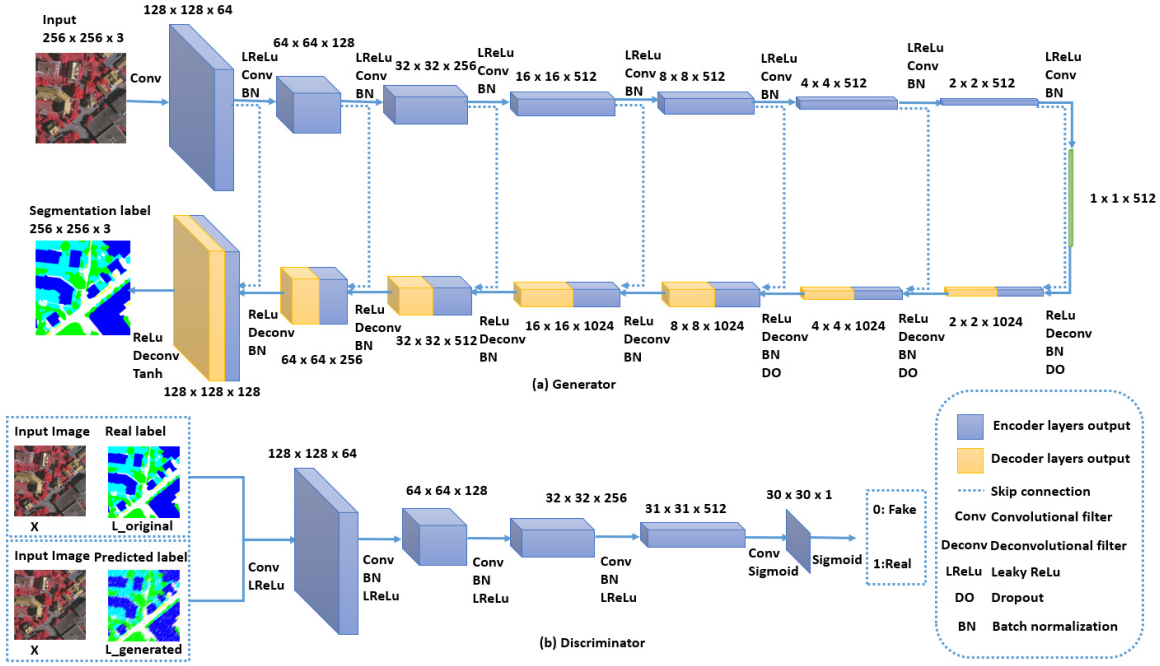

Figure 4. The architecture of the first GAN network: (**a**) the generator and (**b**) the discriminator.

Concerning the discriminator, it takes as input pairs of images from two sets. The first set is images from the target dataset associated with their real label $(X_{target}, L_{original})$. Data from this set should be classified by the discriminator as real pairs of data. The second set is images from the target dataset associated with the generated label from the generator network $(X_{target}, L_{generated})$. Data from this set should be classified by the discriminator as fake pairs of data. In the discriminator, five convolutional layers are used to encode the pair of images (X, L) into a feature vector of size $(30 \times 30 \times 1)$. Then we apply the Sigmoid activation function to this feature vector to get the final binary output of the discriminator {0: fake pairs, 1:real pairs}. As with the encoder part of the generator, we used Leaky ReLu [46] and Batch normalization [47] in every layer except the final one.

Concerning the second GAN Network, it is similar to the first GAN Network. It has a generator and a discriminator as detailed in Figure 3. The architecture of the generator is identical to the architecture of the first generator (See Figure 4) except that it has as input a semantic label and generates an image that imitates images taken from the source dataset. The architecture of the discriminator is also identical to the discriminator of the first GAN (See Figure 4) except that it takes as input pairs of images from two sets. The first set consists of semantic labels associated with their corresponding images from the source dataset $(L_{source}), X_{original})$. The second set consists of semantic labels associated with the generated images from the generator network $(L_{source}), X_{generated})$.

4.2. The Description of the Algorithm

The algorithm adopted to mitigate the domain shift between the source and the target is based on the GAN architecture provided in Figure 3. It is divided into five steps that are illustrated in Figure 5.

There are five steps in the algorithm. Step one begins by the training of a segmentation model on the source domain dataset. The accuracy of the segmentation model could easily get to over 80% if there is a well-structured dataset. In Step 2, we pick out some significant samples from the target domain and label them. The samples are significant if they meet two requirements. First, all the classes should exist in the samples. Second, the most common class representations should be presented. In Step 3, we use these samples to train the first GAN network. Step four makes the training of the second GAN network based on the source dataset provided with labels. Step five is the step where we use our proposed GAN to segment images from the target domain. This step is divided into five sub-steps. In the first sub-step, we pick out any image from the target domain that we want to segment.

Then, we pass it into the generator of the first GAN network to translate it into labels. After that, we pass the generated label into the generator of the second GAN network. This will convert it into an image that imitates images from the source domain. In the fourth sub-step, we pass this generated image into the segmentation network trained in Step 1. We will get then the semantic segmentation map of the image.

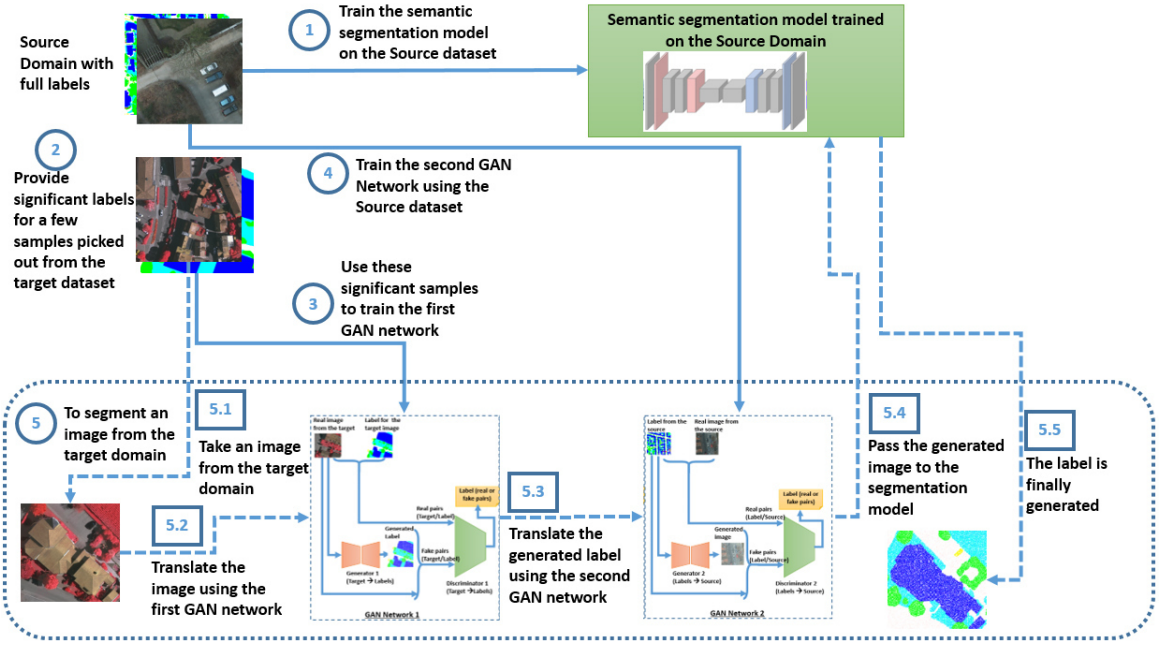

Figure 5. Flowchart of the domain adaptation algorithm.

4.3. Problem Formulation

This section presents the formal modelling of our algorithm. We consider the domain adaptation problem from the target data X_T to the source data X_S. The source data are provided with full semantic labels Y_S while the target data are not initially provided with labels. The first step of the algorithm is concerned with training a semantic segmentation model M_S on the source data. The following equation is correspondent to the source model M_S with the use of the cross-entropy loss:

$$L_{M_S}(M_S, X_S, Y_S) = -\mathbb{E}_{(x_s,y_s)\sim(X_S,Y_S)} \sum_{c=1}^{C} \mathbb{1}_{[c=y_s]} log(Softmax(M_S^{(c)}(x_s))) \quad (2)$$

where $\mathbb{E}_{(x,ys)\sim(X_S,Y_S)}$ denotes the expectation on x_s, y_s, which is drawn by the X_S and Y_S distribution. C is the number of classes of the segmentation task. $\mathbb{1}_{[c=y_s]}$ represents the loss for the class c separated from other classes. Due to the advancements in the semantic segmentation field, M_S could have a good accuracy if the source data are well constructed. However, when we use this model to segment images from the target domain, the accuracy will decrease because of the domain shift that exists between the source and the target. Hence, we will apply our proposed algorithm and begin by picking out significant samples from the target domain and labeling them. This small dataset will guide the mapping process from the target to the source. It will be used to train the first GAN Network, which will learn through an adversarial loss how to generate pixel-wise labels of the target images.

The mapping model from the target domain to the semantic labels $G_{T\longrightarrow L}$ is trained for a segmentation task in an adversarial manner. The discriminator $D_{T\longrightarrow L}$ will be trained on the other side to detect fake pairs of data from the real ones. This GAN model can be seen as a standard GAN where we applied three modifications. First, we substituted the noise vector by input images from the target domain. Second modification is to set the output of the GAN to semantic segmentation labels of

the input image. Third, is to set the input for the discriminator as pair of image for the target domain x_t, and semantic label real or generated by the discriminator (l_{t_real} or $l_{t\_generated}$).

Also, the GAN model can be treated as a modification of a conditional GAN [49]. Generally, conditional GANs or cGANs make a mapping between an input and an output. The input are formed by an image x from a distribution X concatenated to a noise vector z. The output is an image y from a distribution Y. The mapping is formulated by $G_{cGAN} : \{x, z\} \longrightarrow y$ and the loss function of a conditional GAN can be formulated as:

$$L_{cGAN}(G, D, X, Y, z) = \mathbb{E}_{(x\sim X, y\sim Y)}[logD(x, y)] + \mathbb{E}_{(x\sim X, z)}[log(1 - D(x, G(x, z))] \tag{3}$$

During the training, G (the generator) is trying to minimize this loss, whereas D (the discriminator) is trying to maximize it. The loss function can be expressed more clearly as:

$$L = arg\ min_G\ max_D\ L_{cGAN}(G, D, X, Y, z) \tag{4}$$

As proven in [20], mixing this GAN objective with $L1$ loss gives beneficial results. In fact, this helps the generator to be more able to fool the discriminator by being closer to the ground truth in an $L1$ sense. This loss does not affect the discriminator. The $L1$ loss related to the cGAN can be expressed as:

$$L_{L1} = \mathbb{E}_{(x\sim X, y\sim Y, z)}[\|y - G(x, z)\|_1] \tag{5}$$

The final objective of the cGAN can be expressed as:

$$L = \lambda_1 arg\ min_G\ max_D\ L_{cGAN}(G, D, X, Y, z) + \lambda_2 L_{L1} \tag{6}$$

where λ_1 and λ_2 are the weights for the original GAN loss and the $L1$ loss, respectively. Concerning the noise vector z, it is added to the cGAN to give stochasticity inside the given output. By removing it and passing only the input x_s, the generator tends more to give deterministic outputs and fails to grasp the whole entropy of the distribution they want to learn. Normally, Gaussian noise is used for the vector z. In the experiments, this noise vector does not prove high efficiency for capturing the high data variability. The model learns during the training to ignore the noise. Therefore, we did not use the noise vector. We used, instead, the dropout technique in the three first layers of the decoder part of the generator. The dropout has the effect to add some minor stochasticity in the generated output. Hence, the loss function of the first GAN network we implemented is:

$$\begin{gathered} Loss(GAN1) = \lambda_1 arg\ min_G\ max_D[\mathbb{E}_{(x_t\sim X_T, l_t\sim L_T)}[logD(x_t, y_t)]+ \\ \mathbb{E}_{(x_t\sim X_T)}[log(1 - D(x_t, G(x_t))]] + \lambda_2 \mathbb{E}_{(x_t\sim X_T, l_t\sim L_T)}[\|y_t - G(x_t)\|_1] \end{gathered} \tag{7}$$

where x_t are the input images sampled from the target distribution data X_T, l_t is the label associated with this input image and sampled from the target label distribution data L_T.

Similarly, the loss function of the second GAN network we implemented is defined as:

$$\begin{gathered} Loss(GAN2) = \lambda_1 arg\ min_G\ max_D[\mathbb{E}_{(l_s\sim L_S, x_s\sim X_S)}[logD(l_s, x_s)]+ \\ \mathbb{E}_{(l_s\sim L_S)}[log(1 - D(l_s, G(l_s))]] + \lambda_2 \mathbb{E}_{(l_s\sim L_S, x_s\sim X_S)}[\|x_s - G(l - s)\|_1] \end{gathered} \tag{8}$$

where x_s are the images sampled from the source distribution data X_s, l_s is the label associated with this image and sampled from the source label distribution data L_S. $GAN2$ is mapping from the semantic label to the source images.

To perform a segmentation of an image from the target domain X_T, we translate it to source domain X_S in two steps. Step one makes the translation from the target domain to semantic label domain using the generator of $GAN1$. Step 2 makes the translation from the semantic label domain to the source domain using the generator of $GAN2$. The source semantic segmentation model M_S will be

more able to segment the final translated image as it belongs more to the source domain on which it was trained. This will be more emphasized in the Experimental Section.

5. Experimental Results

This section aims at confirming the efficiency of our algorithm by describing the experiments that were implemented as well as discussing the findings.

5.1. The Datasets and the Evaluation Metrics

5.1.1. The Datasets

ISPRS (WGII/4) 2D semantic segmentation benchmark dataset [39] was used for validating our methodology. This dataset is provided by the ISPRS 2D semantic labeling challenge which provides a whole platform for the evaluation of semantic segmentation algorithms in aerial imagery context. We used Potsdam and Vaihingen datasets that are freely accessible by the community. Every image is provided with DSM (digital surface model) data. However, we used only the image data because we are targeting the domain adaptation using image data only. The two datasets comprise of very high-resolution photos with a 5 cm per pixel for Potsdam photos and 9 cm per pixel for Vaihingen photos. This difference of resolution represents one of the domain shift factors between the two domains. The VHR (Very High Resolution) helps in minimizing the interclass variance and maximizing the intraclass variance through the provision of more details about the objects represented in the images.

Every image in the two datasets is afforded with its semantic segmentation label that is categorized into six classes of ground objects: car, low vegetation, building, tree, clutter/background, and impervious surfaces. Clutter/background includes any ground object excluded from the five classes, while impervious surfaces show a paved area without any structure on it. In the Vaihingen dataset, there are 33 TOP images whose sizes are around 2000×2000 pixels. There are three channels in the TOP file: green, red, and the infrared bands. Out of the 33 TOP images, 27 were used for the training, and the remaining six were used for test. The Potsdam dataset contains 38 TOP images of size 6000×6000 pixels. These TOP files have three spectral channels: blue, green, and red. We subdivided these images into 32 TOP images for train, and the remaining 6 for test. For training of the segmentation model, the images were divided to squares of size 512×512 pixels used to feed the network. Samples from Vaihingen and Potsdam ISPRS datasets are shown in Figure 6.

Pixel distribution is not balanced across the six classes. Some classes like Buildings are more redundant than other classes like cars. Each class percentage in proportion to the whole number of pixels in the dataset is represented in Table 1.

Table 1. The distribution of pixels among categories.

Category	Potsdam	Vaihingen
Buildings	28.2%	26.9%
Impervious Surfaces	29.9%	29.3%
Low vegetation	20.9%	19.4%
Trees	14.4%	22.4%
Clutter	4.8%	0.7%
Cars	1.7%	1.3%

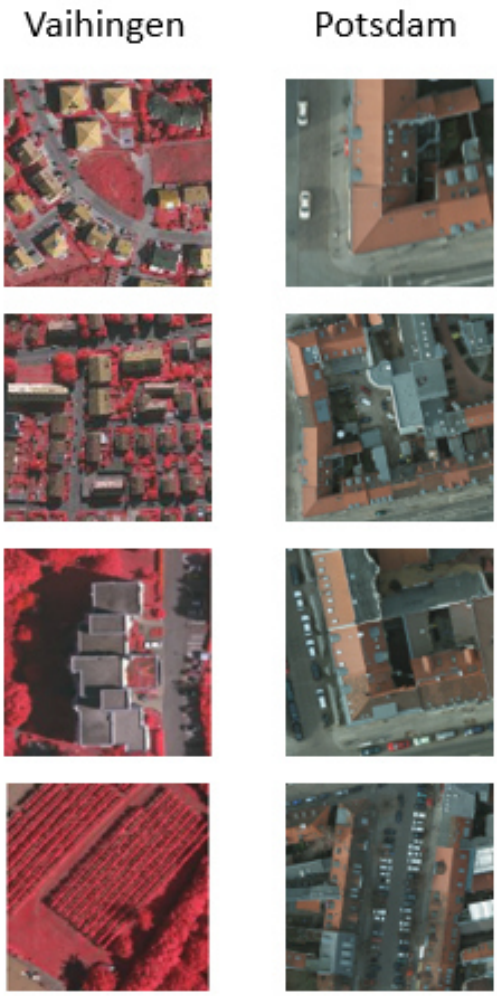

Figure 6. Image samples from Vaihingen and Potsdam ISPRS datasets.

5.1.2. The Analysis of the Domain Shift Factors

Three factors are responsible for the domain shift between Potsdam (the source domain), and Vaihingen (the target domain): Sensor variation, the variation of class representation, and the variation of resolution. Beginning by the sensor variation factor, Potsdam images are captured using RGB (Red-Green-Blue) sensor while Vaihingen images are captured using IRRG (Infrared, Red and Green) sensor. For example, green color is transformed to red in Vaihingen images. This makes the model which is trained on Potsdam images to fail in recognizing classes normally associated with green color like trees and low vegetation. Concerning the variation of resolution, we note that Vaihingen images are captured by using 9 cm per pixel resolution while those of Potsdam are captured using 5 cm per pixel resolution. This variation affects the ability of the model to recognize classes that are trained on a specific scale of resolution (like cars for example). Passing to the third factor, which is the difference of class representation, it is the most delicate factor to treat. To illustrate this domain shift factor, we can take the case of low vegetation class. Low vegetation in Potsdam are mostly grass areas in the modern town style. In Vaihingen, low vegetation class has different patterns and representations. In fact, they correspond to agricultural zones containing different types of vegetation. This difference of patterns on the same class affects the model ability when passing from a domain to another. We made a careful analysis of the domain between Potsdam and Vaihingen images for the six classes and the results are summarized in Table 2. This table helps us in studying the effect of our algorithm on every domain shift factor.

Table 2. Domain shift analysis when passing from Potsdam to Vaihingen.

Domain Shift Factor	Resolution	Sensor	Class Representation
Trees	low	high	medium
Cars	low	low	low
Clutter	low	high	high
Impervious Surfaces	low	low	low
Buildings	low	high	low
Low vegetation	low	high	high

5.1.3. Evaluation Metrics

To evaluate the semantic segmentation algorithms, four metrics are used: the accuracy, the recall, the precision and the F1 score. These metrics are calculated using TN (True Negatives), TP (True Positives), FN (False Negatives) and FP (False Positives). Considering a semantic segmentation class C, TP is the number of pixels of class C classified successfully as C. TN corresponds to the number of pixels that are not C and the algorithm did not classify them as C. FN corresponds to the number of pixels that belong to the class C but the algorithm did not classify them as C. FP is the count of pixels that are classified incorrectly as C while they do not belong really to C. These metrics are expressed below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = Sensitivity = \frac{TP}{TP + FN} \tag{11}$$

$$F1 = Dice = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{12}$$

We used also a fifth metric, which is the Intersection over Union (IoU) to evaluate the global segmentation efficiency. IoU is computed for every class separately before deducing the mean IoU for all the classes. Below is the expression of the IoU for two different sets of data A and B:

$$IoU(A, B) = \frac{size(A \cap B)}{size(A \cup B)} = \frac{TP}{TP + FP + FN} \tag{13}$$

5.2. Experimental Settings

5.2.1. Step 1: Training of the Semantic Segmentation Model

The algorithm begins by training the semantic segmentation model on the source dataset. Concerning the selection of the source dataset, we demonstrated in Section 5.2.6 (Discussion part) that our algorithm shows higher efficiency if we choose Potsdam domain as source and Vaihingen domain as target. In fact, Potsdam is far larger than Vaihingen (3800 images in Potsdam compared to 459 images in Vaihingen). Hence, to maximize the efficiency of our algorithm, we should select a large dataset as the source to learn the global patterns of the data. Then, our domain adaptation algorithm is used to treat domain shift that exists between the source and the target datasets. Consequently, we selected in our experiments Potsdam dataset as the source dataset and Vaihingen as the target dataset. Once the dataset was ready, we applied a state-of-the-art semantic segmentation model adapted to our requirements in aerial imagery. We chose the Bilateral Segmentation Network (BiseNet) [50], which is the fastest segmentation model tested on the Cityscapes dataset [13]. It achieves a speed of 65.5 frames per second with a mean IoU of 74.7 % on this dataset [51]. In the processing of aerial images, the speed factor is very important to have the ability to process video streams captured in real time from a drone. The architecture of BiSeNet is represented in Figure 7.

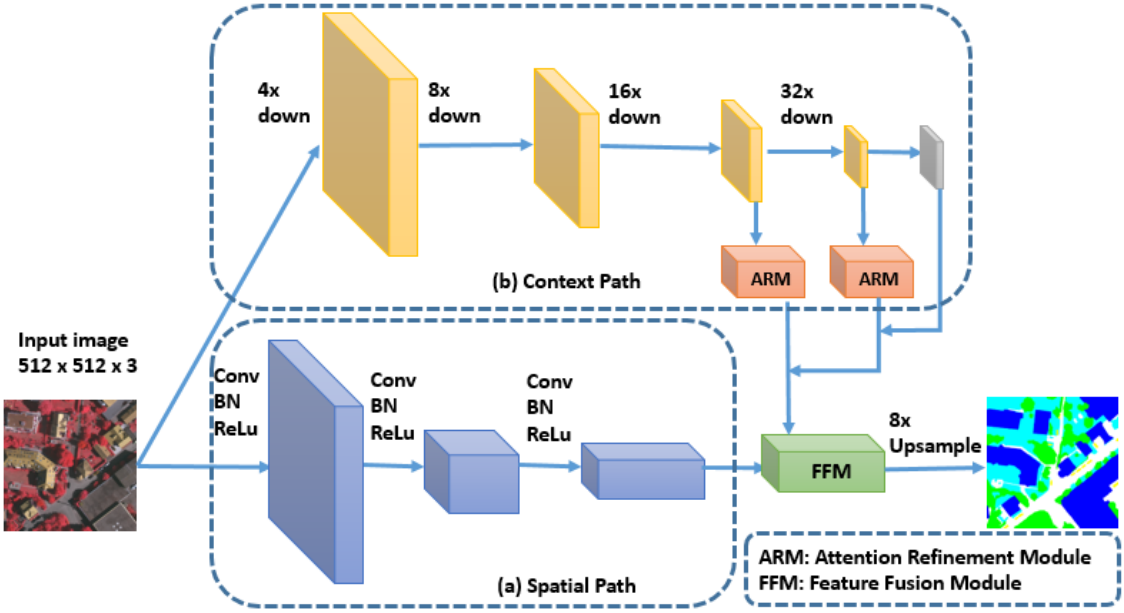

Figure 7. The Architecture of BiSeNet (Bilateral Segmentation Network).

A GPU machine containing the following features was used to carry out experiments associated with this study:

- Processor: Intel Core i9 (Coffee Lake architecture, 6 cores)
- GPU: Nvidia GTX 1080, 8GB dedicated
- Memory: 32 GB RAM
- Operating system: Windows 10 and Linux (Ubuntu 16.04)

To train BiSeNet on the Potsdam dataset, we used semantic segmentation Suite [52], which is an open-source framework where several segmentation models are implemented in Tensorflow [53]. ResNet101 [54] was used as the front end for the BiSeNet network. The training is run for 80 epochs, we set 1 image as the batch size. Image augmentation techniques are not used. ADAM [55] is used as an optimizer during the training with a learning rate 0.0001. As shown in Figure 8, the average segmentation accuracy surpasses 85% on the validation dataset in less than 15 epochs.

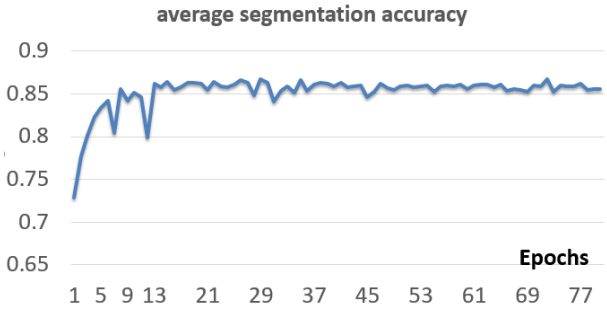

Figure 8. Progress of segmentation accuracy of BiseNet trained on Potsdam.

The Figure 9 shows the convergence of the loss of BiSeNet over the epochs.

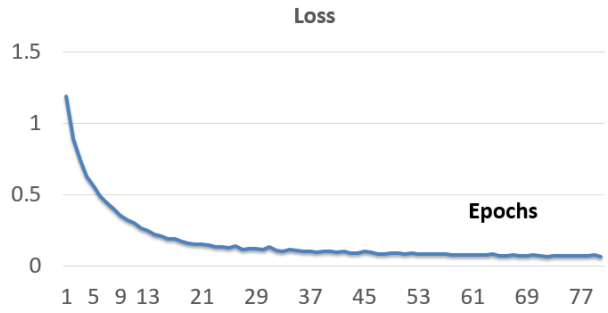

Figure 9. Curve of the loss during the training of BiseNet on Potsdam dataset.

After the training, model weights are saved to be used later in Step 5.

5.2.2. Step 2: Label Significant Data Samples from the Target Domain

To use data efficiently from the target domain, Step 2 consists of picking out significant samples from the target domain and semantically labeling them. The samples are significant if they respect two conditions. The first condition is that they contain pixels from all the classes on which the system is trained. Preferably, the class representation should be as balanced as possible, but this is not necessary. Sometimes, assuring balanced pixel distribution between classes on real aerial images is not possible. The second condition that should be met by the selected samples is that for every class, we should provide the most common patterns in the target domain. Once the samples are chosen, we semantically label them in respect to our targeted classes.

5.2.3. Step 3: Training the First GAN Network

We began by training the first GAN network using the set of labeled data we made in Step 2. We trained the GAN network many times using different sizes of sampled data to study the effect of the number of sampled images on increasing the accuracy of the segmentation. We tried 7 sizes of sampled images from the target domain (Vaihingen): 1, 3, 12, 23, 47, 94, 188. All the selected images are of size $512 * 512$. The GAN network learns during the training how to translate images from the target domain to the semantic label domain. The GAN architecture was implemented using Tensorflow [53]. We set the value 0.2 for the slope α of Leaky ReLu. We used ADAM optimizer with an initial learning rate of 0.0002 and momentum term β 0.5. The weight for the L_1 loss is set to 100 and the weight for the GAN weight is set to 1.

5.2.4. Step 4: Training the Second GAN Network

We passed to the fourth step of the algorithm, which is the training of the second GAN network that maps from the label domain to the source domain (Potsdam). We used the full provided source target for this training (3800 images of size 512×512). The GAN is implemented in Tensorflow [53]. We used the value 0.2 for the slope of Leaky ReLu. We used ADAM optimizer using an initial learning rate set to 0.0002 and a momentum term β set to 0.5. We run the training until convergence of the loss of the discriminator and the generator. Figure 10 shows some images generated in the source domain (Potsdam) from the semantic label. The generated images are mimicking the characteristics of real images from the source domain.

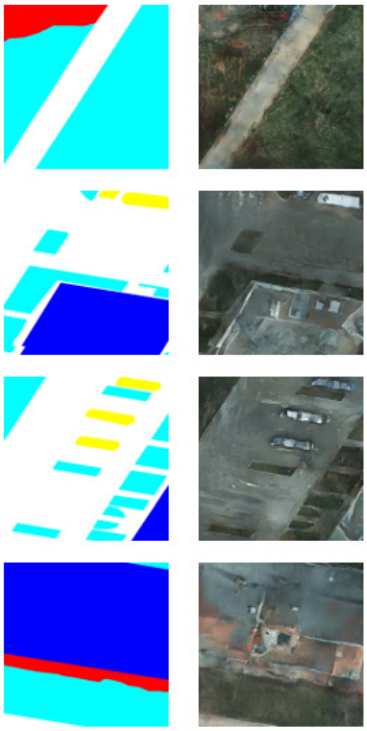

Figure 10. Translation of images from the label domain to the source domain using the second GAN network.

5.2.5. Step 5: Segment Images from the Target Domain

Once we finished the training of the first and the second GAN network, we apply Step 5 for segmentation of images from the target domain. We began by selecting the image we want to segment from the target domain (Vaihingen). Then we apply the first GAN network to translate this image to the label domain. The corresponding semantic label image will be passed into the second GAN network to generate the corresponding image in the source domain (Potsdam). The semantic segmentation model already trained on the source domain will be more able to segment this generated image because it is in the same domain it was trained on.

5.2.6. Discussion

The results of segmentation are always better using the generated image than the original one. Figure 11 shows the result of segmentation of some selected images from the target domain before and after application of our proposed algorithm. We can see clearly that our algorithm improves the quality of segmentation without needing too much data. The results in Figure 11 are generated using first GAN network trained only on 23 labeled images of size $512 * 512$ from the target domain.

The improvement in segmentation accuracy increases with the number of labeled images picked out from the target domain. We tested different trainings of the GAN 1 model using 1, 3, 12, 23, 47, 94 and 188 labeled images from the target domain.

To select the source dataset from Vaihingen and Potsdam, we applied our algorithm twice. The first is done using Potsdam as source and Vaihingen as target, results are shown in Table 3. The second is done using Vaihingen as source and Potsdam as target, results are shown in Table 4. Both tables show the improvement made by the algorithm in segmentation accuracy, precision, sensitivity, dice coefficient and IoU for the different numbers of sampled images from the target domain.

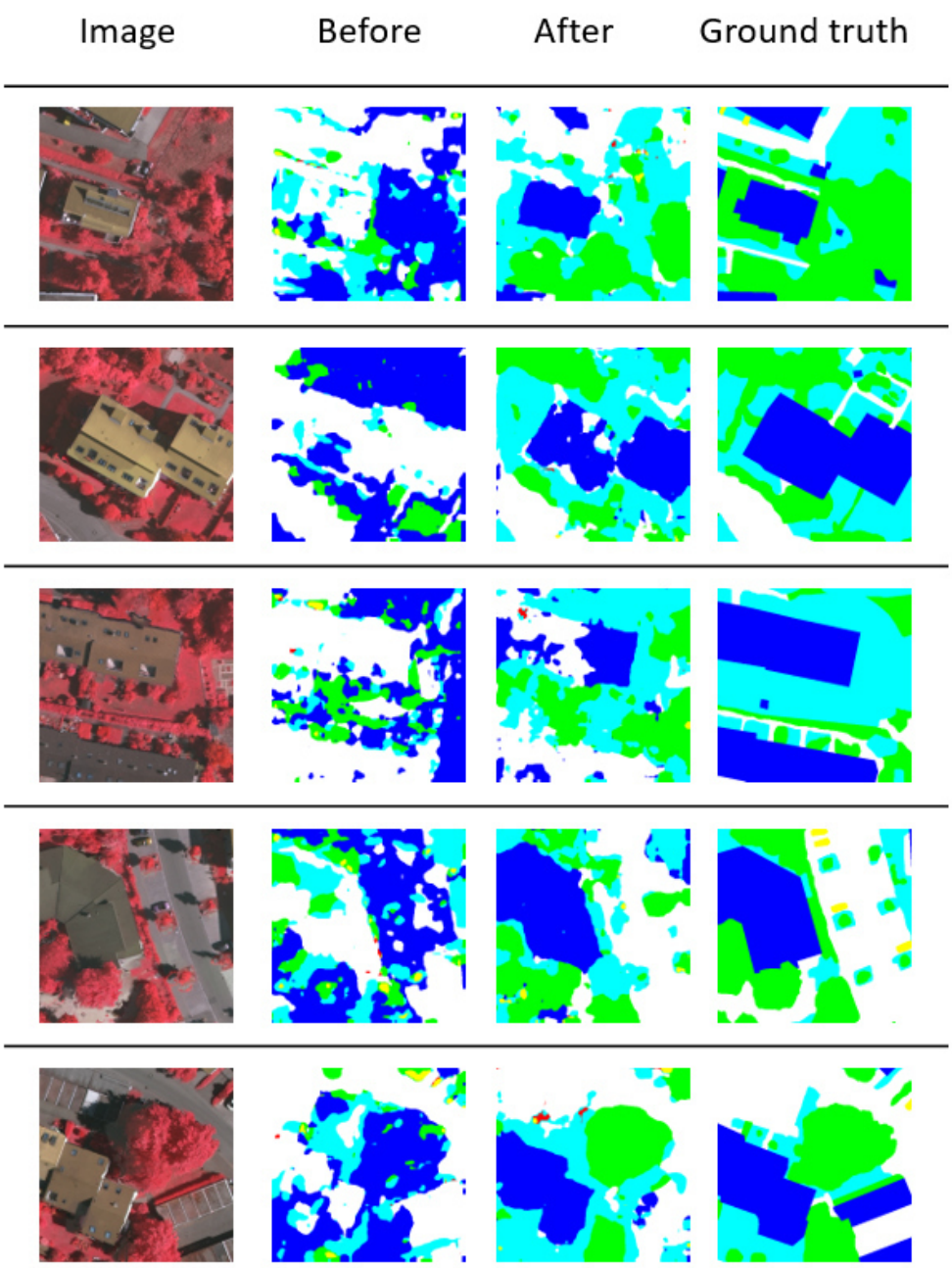

Figure 11. Segmentation of images from the target domain before and after application of our proposed algorithm.

Table 3. Segmentation Accuracy, Precision, Sensitivity, Dice Coefficient and IoU score for different numbers of sampled images from the target domain (Potsdam as source and Vaihingen as target).

Number of Images from Target	Average Accuracy	Precision	Sensitivity	Dice Coef	IoU
0	0.345	0.345	0.345	0.316	0.175
1	0.368	0.445	0.368	0.360	0.176
3	0.422	0.470	0.422	0.404	0.214
12	0.474	0.513	0.474	0.455	0.253
23	0.507	0.541	0.507	0.488	0.281
47	0.524	0.559	0.524	0.505	0.297
94	0.559	0.591	0.559	0.543	0.327
188	0.588	0.625	0.588	0.572	0.349

Table 4. Segmentation Accuracy, Precision, Sensitivity, Dice Coefficient and IoU score for different numbers of sampled images from the target domain (Vaihingen as source and Potsdam as target).

Number of Images from Target	Average Accuracy	Precision	Sensitivity	Dice Coef	IoU
0	0.334	0.335	0.334	0.288	0.169
192	0.363	0.365	0.363	0.318	0.179

As illustrated in Tables 3 and 4, the algorithm efficiency is far better if we select Potsdam as the source. For similar number of labeled images (188 images in first case and 192 images in second case), improvement in average accuracy is 0.243 in first case and 0.026 in the second case. In fact, Potsdam dataset is far larger than Vaihingen dataset (3800 images in Potsdam and 459 images in Vaihingen). Training the segmentation model on a large dataset helps to better capture the global patterns of the data. Then, our algorithm will be more efficient in minimizing the domain shift that exists between the source and the target. This can be seen when selecting Potsdam as source in Table 3. If the source dataset is small, the segmentation model will be less able to learn the global patterns of the data. In this case, our algorithm will not be able to treat efficiently the domain shift. This can be seen when selecting Vaihingen as source a in Table 4. Hence, our algorithm works more efficiently if the source dataset is large enough to make the segmentation model capture the global patterns of the data.

As shown in Table 3, without our algorithm, the total accuracy was 34.5%. Using only 1 labeled images, accuracy increases to 36.8%. The more labeled images are added, the more the accuracy increases until reaching 58.8% for 188 labeled images from the target domain. 188 images represents only 4.9% of the images that was needed to train the segmentation model on the source dataset (3800 images of size 512×512 which proves the data efficiency of our algorithm. Figures 12–16 respectively display the curves of accuracy, precision, sensitivity, F1 score and IoU for different numbers of sampled images from the target domain.

We deduced from the above curves that our algorithm increased the segmentation metrics by a significant margin, which explains the visible amelioration of the segmented map shown in Figure 11.

To judge the global efficiency of our algorithm, we compared it with two other domain adaptation algorithms. The first is FCNs in the wild [32]. It is a domain adaptation algorithm applied in general semantic segmentation. The second is the unsupervised algorithm introduced by Benjdira et al. [4]. It was designed specifically for domain adaptation in semantic segmentation of aerial imagery. We provided in Table 5 the comparison of average accuracy, dice coefficient (F1 score) and IoU in the task of domain adaptation from Potsdam to Vaihingen.

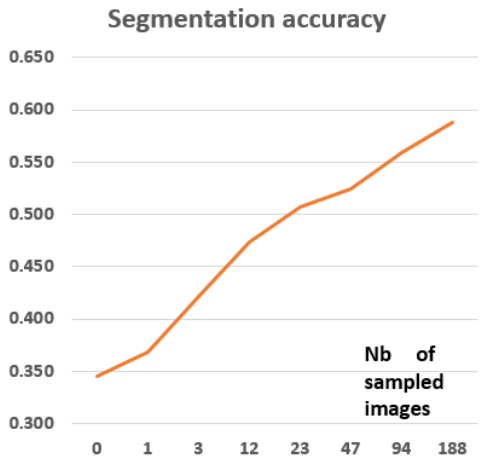

Figure 12. Segmentation accuracy for different numbers of sampled images from the target domain.

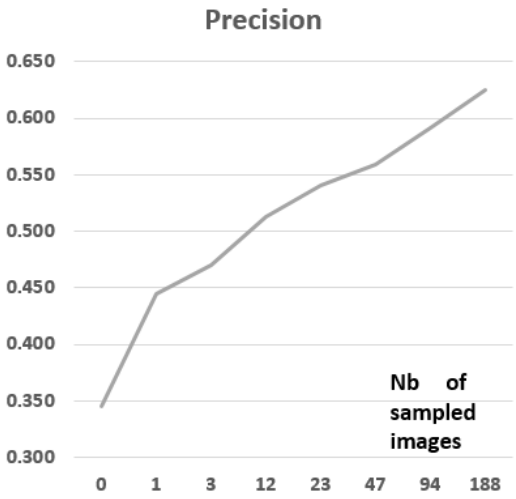

Figure 13. Segmentation precision for different numbers of sampled images from the target domain.

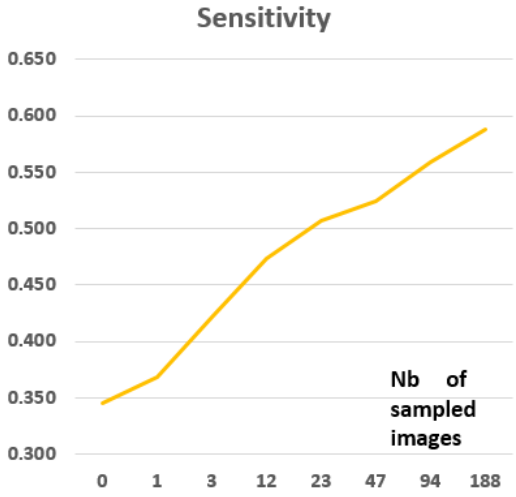

Figure 14. Segmentation sensitivity for different numbers of sampled images from the target domain.

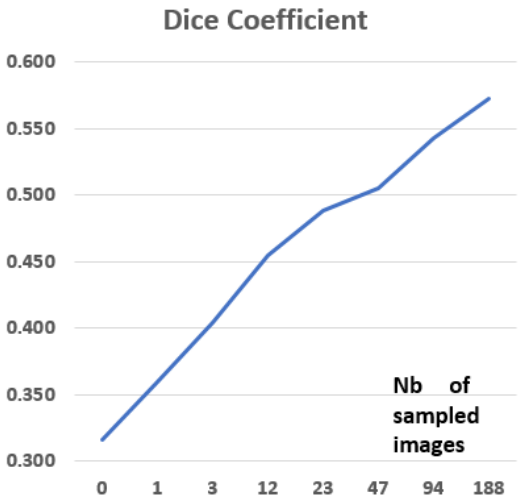

Figure 15. Segmentation dice coefficient for different numbers of sampled images from the target domain.

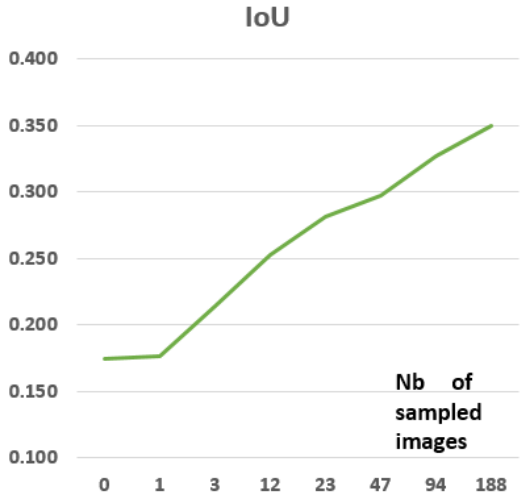

Figure 16. Segmentation IoU for different numbers of sampled from the target domain.

Table 5. Segmentation Accuracy, Dice Coefficient and IoU score for different domain adaptation algorithms (Potsdam as source and Vaihingen as target).

Method	Average Accuracy	Dice Coef	IoU
Without Domain Adaptation	0.345	0.316	0.175
FCNs in the wild	0.486	0.413	0.309
Unsupervised Method in [4]	0.520	0.490	0.300
Ours (188 images)	0.588	0.572	0.349

As shown in Table 5, our algorithm outperforms the other two algorithms in all measures. As a conclusion, the final user can choose between the unsupervised approach introduced by Benjdira et al. [4] and the current supervised approach. If the user does not want to invest time in labeling and wants a medium accuracy, he can choose the unsupervised approach. If he cares most about the accuracy, he can use the supervised approach presented in this paper. The choice will be a tradeoff between the labeling cost and the accuracy of domain adaptation.

Going deeper in the analysis of our algorithm, we studied its effect on the improvement of the segmentation accuracy for every class apart. Table 6 shows the improvement noted in each class for different numbers of sampled images. Moreover, we can validate that for every class, the more the size of labeled images increases, the more the segmentation accuracy is improved. This has some limited exceptions with some local decrease in the accuracy but did not affect the global improvement behavior. Only one class, the clutter background, knows a small decrease in accuracy and this is because there is no specific pattern to be learned for this class. It represents all the components that cannot be included within the other five classes, and the small number of samples is not sufficient for the first GAN network to capture the different patterns existing for this class. On the other side, classes Trees, Low vegetation and Building know a significant increase of accuracy by a margin of 47.7%, 21.9% and 31.6%, respectively. This is because the picked samples from the target domain were sufficient for the first GAN network to learn most of the patterns that exist for the class.

Table 6. Improvement of per-class accuracy for different numbers of sampled images from the target domain.

Nb of Images	Imp. Surf.	Building	Low Veget.	Tree	Car	Clutter Backgr.
0	0.583	0.227	0.383	0.062	0.400	0.935
1	0.591	0.265	0.368	0.311	0.323	0.893
3	0.477	0.303	0.553	0.417	0.323	0.893
12	0.538	0.439	0.559	0.402	0.325	0.894
23	0.602	0.467	0.573	0.421	0.338	0.894
47	0.605	0.416	0.575	0.520	0.385	0.893
94	0.634	0.463	0.618	0.509	0.405	0.893
188	0.685	0.543	0.602	0.539	0.408	0.893

Figure 17 shows the curves of improvement of segmentation accuracy for every class for different sizes of the labeled images from the target domain.

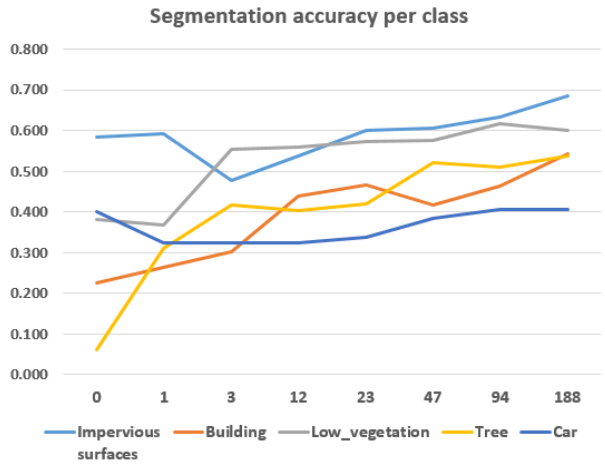

Figure 17. Segmentation accuracy per class for different sizes of the labeled images from the target domain.

Concerning the domain shift analysis of our algorithm, we note that it has no practical effect on classes that are not subject to domain shift (like class Impervious Surfaces and class Cars). In fact, our algorithm made an improvement of 0.8% for the class Car and 10.2% for the class Impervious Surfaces. On the other side, if the sensor domain shift factor is high on the class (See Table 2), the algorithm improves remarkably the segmentation accuracy. The improvement becomes smaller if it is combined with a high effect of other factors. For example, the class Trees and Buildings have a high effect of sensor variation factor and respectively medium and low effects of the class representation factor. The improvement is then 47.7% and 31.6%, respectively. We conclude, then, the efficiency of our algorithm in treating these cases of domain shift factors. On the other side, classes Low vegetation and clutter have a high effect of the sensor variation factor and a high effect of the class representation factor (the effect is much higher in the clutter class). The improvement is respectively 21.9% and −4.2%. We can conclude that the high effect of other domain shift factors reduces the impact of our algorithm, however, small to medium domain shift factors could be mitigated due to the supervision made by the labeled images. If the class patterns are presented in the labeled images (example: class Tree), the domain shift could be mitigated at a good degree. If the class patterns are so diversified and could not be provided in the small set of labeled images (example: class clutter background), the domain shift will be mitigated proportionally. This can be considered to be a limitation to our data-efficient approach. Hence, our algorithm is very efficient in treating the sensor variation factor. Concerning the class representation factor, our algorithm will be efficient in cases the effect is low to medium and most of the class patterns are provided within the labeled dataset.

6. Conclusions

In this study, we proposed a data-efficient supervised approach for domain adaptation in the context of semantic segmentation of aerial imagery. The algorithm is based on two GAN networks to translate images from the target to the source domain. This translation needs the provision of a small set of labeled images from the target domain. The method improves the segmentation accuracy by a margin up to 24% providing only 4.9% of the labeled data needed to train the segmentation model on the source dataset. Our method is confirmed to be efficient in treating the domain shift resulting from the sensor variation factor. It is also efficient in treating low to medium degrees of class representation factor if most of the class patterns are provided within the labeled data. Our work confirms the potential of GANs in aerial imagery analysis. Nevertheless, our algorithm should be combined with a solution to treat cases where we have high degree of class representation factor. Also, we should provide a remedy for cases where the class patterns are so diversified that they cannot be provided in the sampled images from the target domain. This can be solved using a semi-supervised approach to alleviate the manual work needed to label data.

Author Contributions: B.B. and A.A. designed the method. A.A. implemented the method. B.B. wrote the paper. A.K. and K.O. contributed to the supervision of the work. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Prince Sultan University.

Acknowledgments: This work is supported by Robotics and Internet of Things Lab (RIOTU), Prince Sultan University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alhichri, H.; Jdira, B.B.; Alajlan, N. Multiple Object Scene Description for the Visually Impaired Using Pre-trained Convolutional Neural Networks. In Proceedings of the International Conference on Image Analysis and Recognition, Póvoa de Varzim, Portugal, 13–15 July 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 290–295.
2. Benjdira, B.; Khursheed, T.; Koubaa, A.; Ammar, A.; Ouni, K. Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3. In Proceedings of the 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS), Muscat, Oman, 5–7 February 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
3. Al Rahhal, M.M.; Bazi, Y.; Al Zuair, M.; Othman, E.; BenJdira, B. Convolutional neural networks for electrocardiogram classification. *J. Med Biol. Eng.* **2018**, *38*, 1014–1025. [CrossRef]
4. Benjdira, B.; Bazi, Y.; Koubaa, A.; Ouni, K. Unsupervised Domain Adaptation Using Generative Adversarial Networks for Semantic Segmentation of Aerial Images. *Remote. Sens.* **2019**, *11*, 1369. [CrossRef]
5. Ammour, N.; Alhichri, H.; Bazi, Y.; Benjdira, B.; Alajlan, N.; Zuair, M. Deep learning approach for car detection in UAV imagery. *Remote. Sens.* **2017**, *9*, 312. [CrossRef]
6. Singh, V.K.; Rashwan, H.A.; Romani, S.; Akram, F.; Pandey, N.; Sarker, M.M.K.; Saleh, A.; Arenas, M.; Arquez, M.; Puig, D.; et al. Breast tumor segmentation and shape classification in mammograms using generative adversarial and convolutional neural network. *Expert Syst. Appl.* **2019**, *139*, 112855. [CrossRef]
7. Ammar, A.; Koubaa, A.; Ahmed, M.; Saad, A. Aerial Images Processing for Car Detection using Convolutional Neural Networks: Comparison between Faster R-CNN and YoloV3. *arXiv* **2019**, arXiv:1910.07234.
8. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
9. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]

10. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
11. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef]
12. Dong, L.Y.; Sui, P.; Sun, P.; Li, Y.L. Novel naive Bayes classification algorithm based on semi-supervised learning. *Jilin Daxue Xuebao (Gongxueban)/J. Jilin Univ. (Eng. Technol. Ed.* **2016**, *46*, 884–889. [CrossRef]
13. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223.
14. Sun, P.; Brown, C.; Beschastnikh, I.; Stolee, K.T. Mining Specifications from Documentation using a Crowd. In Proceedings of the 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), Hangzhou, China, 24–27 February 2019; pp. 275–286.
15. Tzeng, E.; Hoffman, J.; Darrell, T.; Saenko, K. Simultaneous deep transfer across domains and tasks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4068–4076.
16. Long, M.; Cao, Y.; Wang, J.; Jordan, M.I. Learning transferable features with deep adaptation networks. *arXiv* **2015**, arXiv:1502.02791.
17. Tzeng, E.; Hoffman, J.; Saenko, K.; Darrell, T. Adversarial discriminative domain adaptation. In Proceedings of the Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; Volume 1, p. 4.
18. Luo, Z.; Zou, Y.; Hoffman, J.; Fei-Fei, L.F. Label efficient learning of transferable representations acrosss domains and tasks. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 165–177.
19. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
20. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]
21. Patel, V.M.; Gopalan, R.; Li, R.; Chellappa, R. Visual Domain Adaptation: A survey of recent advances. *IEEE Signal Process. Mag.* **2015**, *32*, 53–69. [CrossRef]
22. Saenko, K.; Kulis, B.; Fritz, M.; Darrell, T. Adapting visual category models to new domains. In Proceedings of the European Conference on Computer Vision, Heraklion, Crete, Greece, 5–11 September 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 213–226.
23. Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **2016**, *17*, 2096–2030.
24. Ganin, Y.; Lempitsky, V. Unsupervised domain adaptation by backpropagation. *arXiv* **2014**, arXiv:1409.7495.
25. Bousmalis, K.; Silberman, N.; Dohan, D.; Erhan, D.; Krishnan, D. Unsupervised pixel-level domain adaptation with generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3722–3731.
26. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.Y.; Isola, P.; Saenko, K.; Efros, A.A.; Darrell, T. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv* **2017**, arXiv:1711.03213.
27. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A.M. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3234–3243.
28. Vazquez, D.; Lopez, A.M.; Marin, J.; Ponsa, D.; Geronimo, D. Virtual and real world adaptation for pedestrian detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 797–809. [CrossRef] [PubMed]
29. Peng, X.; Saenko, K. Synthetic to real adaptation with generative correlation alignment networks. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1982–1991.

30. Shrivastava, A.; Pfister, T.; Tuzel, O.; Susskind, J.; Wang, W.; Webb, R. Learning from simulated and unsupervised images through adversarial training. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2107–2116.
31. Shafaei, A.; Little, J.J.; Schmidt, M. Play and learn: Using video games to train computer vision models. *arXiv* **2016**, arXiv:1608.01745.
32. Hoffman, J.; Wang, D.; Yu, F.; Darrell, T. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv* **2016**, arXiv:1612.02649.
33. Zhang, Y.; David, P.; Gong, B. Curriculum domain adaptation for semantic segmentation of urban scenes. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2020–2030.
34. Sankaranarayanan, S.; Balaji, Y.; Jain, A.; Lim, S.N.; Chellappa, R. Unsupervised domain adaptation for semantic segmentation with gans. *arXiv* **2017**, arXiv:1711.06969.
35. Tsai, Y.H.; Hung, W.C.; Schulter, S.; Sohn, K.; Yang, M.H.; Chandraker, M. Learning to adapt structured output space for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7472–7481.
36. Huang, H.; Huang, Q.; Krahenbuhl, P. Domain transfer through deep activation matching. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 590–605.
37. Tasar, O.; Happy, S.L.; Tarabalka, Y.; Alliez, P. ColorMapGAN: Unsupervised Domain Adaptation for Semantic Segmentation Using Color Mapping Generative Adversarial Networks. *arXiv* **2019**, arXiv:1907.12859.
38. Fang, B.; Kou, R.; Pan, L.; Chen, P. Category-Sensitive Domain Adaptation for Land Cover Mapping in Aerial Scenes. *Remote. Sens.* **2019**, *11*, 2631. [CrossRef]
39. Gerke, M. *Use of the Stair Vision Library Within the ISPRS 2D Semantic Labeling Benchmark (Vaihingen)*; University of Twente: Enschede, The Nerthands, 2014.
40. Oliehoek, F.A.; Savani, R.; Gallego, J.; van der Pol, E.; Gross, R. Beyond Local Nash Equilibria for Adversarial Networks. *arXiv* **2018**, arXiv:1806.07268.
41. Goodfellow, I.J. NIPS 2016 Tutorial: Generative Adversarial Networks. *arXiv* **2016**, arXiv:1701.00160.
42. Liu, M.Y.; Breuel, T.; Kautz, J. Unsupervised Image-to-Image Translation Networks. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, NV, USA, 4–9 December 2017.
43. Zhu, J.Y.; Zhang, R.; Pathak, D.; Darrell, T.; Efros, A.A.; Wang, O.; Shechtman, E. Toward Multimodal Image-to-Image Translation. In Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, NV, USA, 4–9 December 2017.
44. Yi, Z.; Zhang, H.; Tan, P.; Gong, M. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2868–2876.
45. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017. [CrossRef]
46. Xu, B.; Wang, N.; Chen, T.; Li, M. Empirical Evaluation of Rectified Activations in Convolutional Network. *arXiv* **2015**, arXiv:1505.00853.
47. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
48. Srivastava, N.; Hinton, G.E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
49. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
50. Yu, C.; Wang, J.; Peng, C.; Gao, C.; Yu, G.; Sang, N. BiSeNet: Bilateral Segmentation Network for Real-Time Semantic Segmentation. In Proceedings of the European Conference on Computer Vision, Lecture Notes in Computer Science, Munich, Germany, 8–14 September 2018; pp. 334–349. [CrossRef]
51. Real-Time Semantic Segmentation on Cityscapes. Available online: https://paperswithcode.com/sota/real-time-semantic-segmentation-cityscap (accessed on 28 March 2019).
52. Semantic Segmentation Suite. Available online: https://github.com/GeorgeSeif/Semantic-Segmentation-Suite (accessed on 28 March 2019).

53. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
54. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016. [CrossRef]
55. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.