sensors

MDPI

Article

Recognizing New Classes with Synthetic Data in the Loop: Application to Traffic Sign Recognition

Gabriel Villalonga [1,2,*], Joost Van de Weijer [1,2] and Antonio M. López [1,2]

[1] Computer Vision Center (CVC), Universitat Autònoma de Barcelona (UAB), 08193 Bellaterra, Spain; joost@cvc.uab.es (J.V.d.W.); antonio@cvc.uab.es (A.M.L.)

[2] Computer Science Department, Universitat Autònoma de Barcelona (UAB), 08193 Bellaterra, Spain

* Correspondence: gvillalonga@cvc.uab.es

Received: 20 December 2019; Accepted: 16 January 2020; Published: 21 January 2020

Abstract: On-board vision systems may need to increase the number of classes that can be recognized in a relatively short period. For instance, a traffic sign recognition system may suddenly be required to recognize new signs. Since collecting and annotating samples of such new classes may need more time than we wish, especially for uncommon signs, we propose a method to generate these samples by combining synthetic images and Generative Adversarial Network (GAN) technology. In particular, the GAN is trained on synthetic and real-world samples from known classes to perform synthetic-to-real domain adaptation, but applied to synthetic samples of the new classes. Using the Tsinghua dataset with a synthetic counterpart, SYNTHIA-TS, we have run an extensive set of experiments. The results show that the proposed method is indeed effective, provided that we use a proper Convolutional Neural Network (CNN) to perform the traffic sign recognition (classification) task as well as a proper GAN to transform the synthetic images. Here, a ResNet101-based classifier and domain adaptation based on CycleGAN performed extremely well for a ratio $\sim$1/4 for new/known classes; even for more challenging ratios such as $\sim$4/1, the results are also very positive.

Keywords: CNNs; training with synthetic data; traffic sign recognition

1. Introduction

On-board computer vision is fundamental to perceiving the traffic environment around the ego-vehicle and, therefore, a crucial technology for assisting drivers or enabling fully autonomous vehicles (AVs). In the last decade, computer vision has been empowered by the use of Convolutional Neural Networks (CNNs), which allow the extraction of very detailed meaningful information from raw images. Over the last few years, we have witnessed unprecedented breakthroughs in tasks such as image-level classification [1,2], bounding-box-level 2D and 3D object detection [3–7], pixel-wise class and instance segmentation [8–14], skeleton-wise human pose estimation [15–17], and even dense monocular depth estimation [18–24]; all of them, among others, are essential visual capabilities for high levels of driving automation or assistance.

Indeed, CNNs have become very accurate models provided that there is sufficient data (in size and diversity) for their training [25,26]; *data* refers to both the raw images and the ground truth (GT) that we must associate with them as training supervision. In fact, since CNNs are data hungry and most GT comes from a (cumbersome and error-prone) manual labeling process, providing GT at scale, reducing manual intervention, and/or reusing previous knowledge have become emerging topics for computer vision research in general and for autonomous driving in particular. For instance, *active learning* techniques [27–30] focus on automatically finding the a priori best training images for their posterior manual labeling out of a large number of unlabelled ones; *self-labeling*

techniques [31–33] focus on progressively self-labeling images by refining increasingly accurate models; *transfer learning* techniques [34,35] focus on reusing existing models for re-training them to perform new tasks, so that only labels for such tasks are required; *domain adaptation* techniques [36–39] focus on reusing existing models in new domains, minimizing the labeling effort required in the new domains; while *self-supervision* [40–43] focuses on learning visual models without manual labeling, with the support of auxiliary simple (pretext) tasks for which it is possible to automatically define self-labels. Overall, all of these research topics are evidence that, due to CNNs, computer vision and its applications have become strongly data-dependent.

In this context, situations where a sudden lack of data can seriously limit the operational capabilities of a computer vision system can appear. For instance, imagine an AV or a driver assistance system that must *detect* (i.e., localize within an image) and *recognize/classify* (i.e., assign a specific meaning) traffic signs using on-board computer vision. While detection itself can be very robust due to the stability of the shape of traffic signs across the world (triangles, rectangles, circles), it can happen that the vehicles of a fleet deployed in a new world area suddenly detect traffic signs that they cannot recognize. These vehicles can automatically broadcast the unknown traffic signs to the company for opening an incidence. If the company decides that it is required to actually recognize the new traffic signs, it may take too much time to collect sufficient samples and perform their proper labeling, especially if the samples must be acquired under a variety of environmental conditions and viewpoints, which can be challenging if such new traffic signs are scarce (i.e., they are rare events). While this is just an illustrative example, we can imagine analogous situations for robots manipulating goods, traffic surveillance systems separating vehicles according to their functionality, etc.; readers can imagine other examples where perception-based systems may have such a kind of *unknown class* problem; after all, both CNNs as well as traditional (shallow) vision models work under the *closed-world* assumption.

Following a new tendency of recent years [44–49], in this paper, we propose the exploration of how synthetic data can help in such situations. In particular, assuming that we have a few samples of an unknown class (keeping with the example, these can be traffic sign images reported by the fleet), we propose a method consisting of the following steps: (1) To elaborate 3D graphical instances of the class and automatically place them all around in a virtual environment; (2) by varying the simulation conditions within the virtual environment, to automatically generate as many image samples as needed; (3) to domain-adapt these samples to close the visual gap with real-world images; (4) to retrain the corresponding CNNs for recognizing the new class. Then, the research question is how these results would differ from the results obtained by a counterpart procedure consisting of capturing samples of the new class with real-world cameras and labeling them by hand (i.e., instead of the steps (1)–(3)). Note that (3) must be a task-agnostic domain adaptation step. In particular, we propose the use of a Generative Adversarial Network (GAN) [50], previously trained in a generic manner to transform our synthetic images to look like real-world images captured by the cameras of our computer vision system. We remark that such a GAN has never seen samples of the unknown class before. As a very important and challenging use case, we assess the success of our proposal by applying it to traffic sign recognition in the wild [51], which can be seen as a fine-grain image classification task. We will elaborate exhaustive experiments by varying the known and unknown traffic signs according to different criteria, not only for training/retraining the classification CNNs, but also the task-agnostic GAN. In order to support this experimental part, we have created a synthetic dataset of traffic signs using an evolution of our SYNTHIA environment [44], which we call SYNTHIA-TS. Using the Tsinghua dataset [51] and SYNTHIA-TS, we ran an extensive set of experiments which show that the proposed method is indeed effective, provided that we use a proper CNN to perform the traffic sign classification task as well as a proper GAN to transform the synthetic images.

Here, a ResNet101-based classifier and a CycleGAN performed extremely well for a ratio of $\sim$1/4 for new/known classes; even for more challenging ratios such as $\sim$4/1, the results are also very positive. Therefore, synthesizing data following our proposal establishes a proper methodology to

minimize the lack of real-world labeled data when a computer vision system must be retrained to recognize new classes in a relatively short period.

The rest of the paper is organized as follows. Section 2 reviews the most related work to our proposal. Section 3 elaborates our proposal. Section 4 details the experimental protocol, the obtained results, and the conclusions deduced from them. Finally, Section 5 summarizes the work presented in this paper and suggests future directions of research in line with our conclusions.

2. Related Work

Learning to recognize new classes falls into the paradigm of *lifelong learning* [52–54], where a perception-based system has to continuously adapt to situations not previously experienced. We can think of three main components of such a learning capability (see Figure 1). The first one consists of the ability to identify unknown classes, which is not trivial, since CNNs tend to be overconfident about their classification decisions. In fact, this is an active research topic known as *out-of-distribution detection* (which includes *novelty* and *anomaly* detection; this paper relates to the novelty case) [55–58]. The second component consists of a data generation protocol to collect training samples of these unknown classes, provided that the CNN needs to take them into account in the future. Finally, the third component consists of a procedure that allows the CNN to recognize the new classes without deteriorating its accuracy in identifying the classes for which it was initially/previously trained; this is known as learning without forgetting [59–62] and still is an open and challenging research topic. In fact, the work of [58] focuses on out-of-distribution for our use case, i.e., traffic sign recognition.

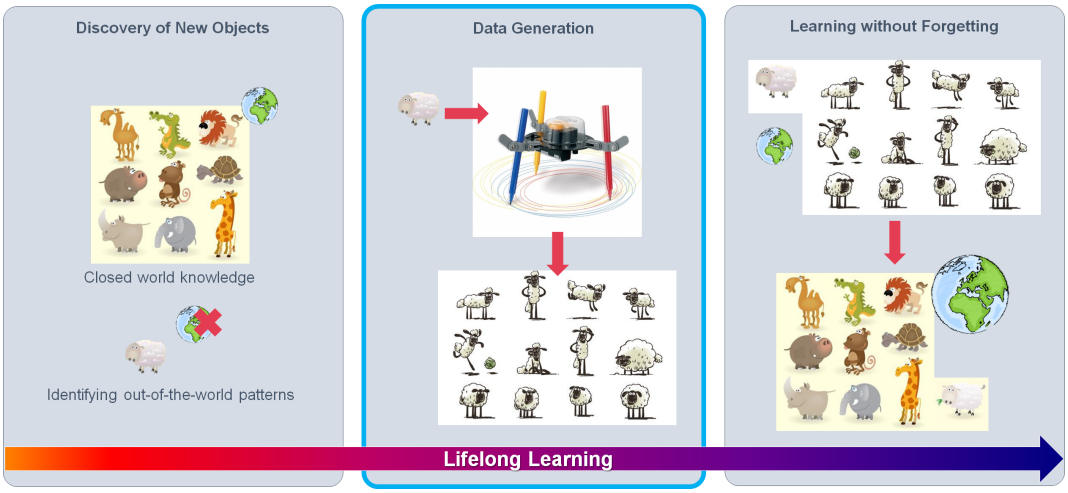

Figure 1. Lifelong learning setting. First, unknown classes are identified. Then, if we want to consider them in the future, we must collect diverse samples of these classes for posterior model retraining. Finally, we retrain the models to recognize the new classes without forgetting previous ones. In this paper, we focus on the second step, assuming that rather than collecting the samples from the real world, we generate them by using a virtual world.

In this paper, we focus on data generation. We require that we are given the unknown (novel) traffic signs in the form of a few on-board captured images. In addition, to avoid the forgetting problem when retraining the traffic sign recognition CNNs, we will just retrain using all of the available data (known and synthesized); in this way, the paper can really focus on assessing the usefulness of synthesizing samples of the unknown classes. Accordingly, the remainder of this section addresses the use of synthesized visual data for training computer vision models, as well as the use of GANs to perform task-agnostic domain adaptation.

Researchers such as Taylor et al. [63] pioneered the use of videogame data for testing vision-based tracking algorithms. Marin et al. [64] extended the use of this synthetic data to train object detectors performing in real images, while Vazquez et al. [65] raised the attention on the domain gap between virtual

and real world images. From there, the use of synthetic visual data generated from virtual environments has kept growing. We found works using synthetic data for object detection/recognition [66–69], object viewpoint recognition [70], re-identification [71], and human pose estimation [72]; building synthetic cities for autonomous driving tasks such as semantic segmentation [44,73], place recognition [74], object tracking [45,75], object detection [76,77], stixel computation [78], and benchmarking different on-board computer vision tasks [47]; building indoor scenes for semantic segmentation [79], as well as normal and depth estimation [80]; generating GT for optical flow, scene flow, and disparity [81,82]; generating augmented reality images to support object detection [83]; simulating adverse atmospheric conditions such as rain or fog [84,85]; even performing procedural generation of videos for human action recognition [86,87]. Moreover, since robotics and autonomous driving rely on sensorimotor models worthy of being trained and tested dynamically, in the last years, the use of simulators has been intensified beyond datasets [48,49,88,89].

In contrast to this literature, we can leverage the already-annotated real-world images conveying a set of classes known by our current CNN-based classifier, but we have to assess the possibility of using automatically generated synthetic images as samples of classes that are unknown for our current CNN. Therefore, these synthetic images must be used to retrain the CNN to properly classify previous and new classes. We will evaluate two different settings: First, when the synthetic images are used as they come from the virtual environment; second, when the synthetic images are transformed by a GAN to look like the real-world images, i.e., as a type of task-agnostic domain adaptation where, following the domain adaptation terminology, the synthetic world acts as the *source* domain and the real world as the *target* domain.

GANs use a *generator* CNN to transform the appearance of source images to look like target images, and a *discriminator* CNN which aims at distinguishing between the transformed and the original images in the target domain. The generator–discriminator system is trained until the discriminator is not able to distinguish the origin of the images, which is understood as the point when the source images are similar enough to the target ones. This application of GANs is known as *image-to-image translation*.

Isola et al. [90] proposed an encoder–decoder as the generator architecture, and a patch-based (patchGAN) approach for the discriminator. Since this approach was only able to work with low-resolution images, other approaches build upon this method to overcome this problem [91]. However, a relevant observation is that these proposals require pixel-level GT about how the generated images should look, which is termed as supervised image-to-image translation. In order to avoid this kind of supervision, Taigman et al. [92] designed an encoder–decoder generator in such a way that the encoder features are indistinguishable for original and transformed images. In other words, for the GT, it is only required to know if the images come from the source domain or the target one, which is always possible at training time. Liu et al. [93] also focused on generators' feature layers. Afterwards, other alternatives were proposed that did not require the mentioned supervision. Some approaches use an auxiliary task to define the loss between input and generated images; for instance, Bousmalis et al. [94] use image-level classification while Hoffman et al. [95] use semantic segmentation as auxiliary tasks. Other approaches focus on appearance of the input and generated images. Shrivastava et al. [96] proposed an identity loss between the input and generated images. One restriction of this approach is that the source and target domain images have similar appearances. Zhu et al. [97] and Kim et al. [98] followed the cycle idea; i.e., from source images, target-style ones are generated which, in turn, are the input to generate new source-style images. The source-to-target and the target-to-source are different generators. In each domain, we have different discriminators. The cycle idea is not only useful because it does not require image GT, but also because the input and transformed images can have a relatively different appearances, especially compared to the approach in [96]. In other words, in contrast to other GAN proposals, a GAN trained according to the cycle idea has the potential of properly transforming the appearance of source images showing content unseen during its training. Accordingly, in this paper, we follow the cycle idea. In particular, since [97] has publicly available code—called CycleGAN—we use it for the experiments in this paper.

Finally, the work with the most similar goal to that of this paper has recently been presented by Beery et al. [99]. The addressed application is animal detection and classification from static cameras. The paper evaluates the use of synthetic data for classifying animals for which it is difficult to have sufficient real-world image samples. Therefore, similarly to us, previous real-world image samples from known classes (animals) are leveraged for retraining their (animal) classifier together with the synthesized images containing the new class samples (they consider deer as a new class). In this paper, rather than focusing on one new class at a time, we also evaluate different balances between known and unknown classes. We also evaluate the difference between using the synthetic images as they come from the virtual environment in contrast to transforming them via GANs. In both cases, since our application falls into fine-grain classification, we also assess the dependency on common visual cues between seen and unseen classes.

3. Method

3.1. Overall Idea

Assume we need a classifier $\mathcal{C}$ such that, given an image (e.g., framing a traffic sign), it is able to assign to it a correct label from a given set $\mathcal{K}$ of known labels/classes (e.g., traffic sign classes). Let $\mathcal{I}$ be a set of images collected for training such a $\mathcal{C}$. For supervised training, we need to assign one class to each image, which is usually done offline by human annotators. Let $\mathcal{I}_\mathcal{K}$ be the corresponding annotated set of images. Then, we can run a supervised machine learning algorithm that uses $\mathcal{I}_\mathcal{K}$ to generate a classifier $\mathcal{C}_{\mathcal{I}_\mathcal{K}}$, which will be used (at testing time) to support the addressed application (e.g., on-board traffic sign recognition). The problem arises when, during the execution of such an application, we realize that there are classes of interest not included in $\mathcal{K}$ (e.g., after a warning from an out-of-distribution detection module also running as part of the application). Let us call $\mathcal{U}$ this set of new classes such that $\mathcal{K} \cap \mathcal{U} = \emptyset$. For training a new supervised classifier $\mathcal{C}_{\mathcal{I}_{\mathcal{K}\cup\mathcal{U}}}$ which takes into account all classes, we need to collect and manually annotate new images covering a sufficient number of instances for each class in $\mathcal{U}$. This may be difficult to do as quickly as we could wish, since we may be facing unusual classes (i.e., for which it is difficult to find corresponding instances by just randomly roaming in the real world) and there will also be a latency due to the manual annotation of the found instances.

Accordingly, the alternative method that we want to explore relies on automatically generating synthetic images to quickly obtain sufficient annotated instances of the new classes for training a new classifier. In addition, as we have already mentioned, training visual models using pure synthetic images can lead to a performance drop when performing in the real world. In order to reduce such a domain gap, GANs are a possible solution; i.e., by directly transforming the images of the source domain (e.g., synthetic world) to have a similar appearance to those of the target domain (e.g., real world). We follow this approach in this study.

Now, let $\widetilde{\mathcal{I}}_\mathcal{U}$ be a set of synthetically generated images automatically annotated according to the classes in $\mathcal{U}$, and $\mathcal{G}^{\widetilde{\mathcal{I}}_\mathcal{U}}$ the corresponding set of images transformed by a GAN. We aim to train a classifier $\mathcal{C}_{\{\mathcal{G}^{\widetilde{\mathcal{I}}_\mathcal{U}}, \mathcal{I}_\mathcal{K}\}}$ (ideally) performing in the real world as if $\mathcal{G}^{\widetilde{\mathcal{I}}_\mathcal{U}}$ would consist of real-world images annotated by humans. Yet another question is how the GAN is trained. From the point of view of generating synthetic images, generating images for classes in $\mathcal{U}$ is analogous to generating for classes in $\mathcal{K}$. Therefore, we require that besides the set of real-world images $\mathcal{I}_\mathcal{K}$, we also have a set $\widetilde{\mathcal{I}}_\mathcal{K}$ of (automatically) annotated synthetic images for the known classes; i.e., both sets cover the same classes, but for each class, it can be a different number of samples in each set. The GAN is trained to transform images from the set $\widetilde{\mathcal{I}}_\mathcal{K}$ into the set $\mathcal{I}_\mathcal{K}$, but without assuming a one-to-one pairing of the images from both sets. In other words, the GAN will learn to perform domain-to-domain transformations, but not class-specific transformations between domains. Therefore, when we need to transform synthetically generated instances for a new previously unknown class (i.e., in $\mathcal{U}$), we can apply the previously learned GAN even if it was not exposed to such a class during the training time and, in fact, it will

not be exposed at this time, due to the lack of real-world instances of this class. Figure 2 depicts the overall idea.

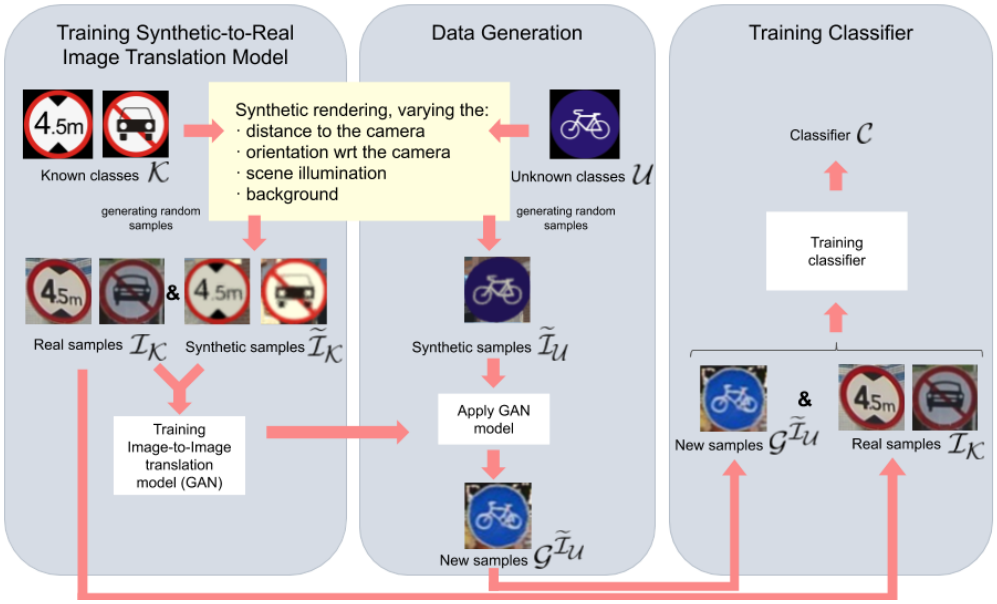

Figure 2. The proposed method for retraining a classifier, $\mathcal{C}$, to keep detecting previously known classes ($\mathcal{K}$)—for which we have labeled real-world samples—and, in addition, new previously unknown classes ($\mathcal{U}$) for which we do not have real-world samples. The key idea is to have synthetic samples for both the known and new classes. The real-world samples of the known classes ($\mathcal{I}_\mathcal{K}$) and the synthetic ones ($\tilde{\mathcal{I}}_\mathcal{K}$) are used to train a Generative Adversarial Network (GAN) with the aim of performing synthetic-to-real domain adaptation. In particular, the synthetic samples of the new classes ($\tilde{\mathcal{I}}_\mathcal{U}$) are transformed ($\mathcal{G}^{\tilde{\mathcal{I}}_\mathcal{U}}$) by this GAN. Then, the real-world samples of the previously known classes and the transformed synthetic samples of the new classes are used to train the desired classifier. The overall idea is illustrated for traffic sign recognition.

3.2. Data Generation

We start by generating synthetic images with automatic GT for each unknown class. We require a real-world example showing the appearance of an instance of each unknown class (i.e., the example already used to decide that the class must be considered in future versions of the classifier). Then, a designer can create a textured 3D model of it. This model can then be populated in a virtual environment that we have predefined. Next, we can capture as many images as we need containing instances of the new class along with automatic GT, which is done under predefined variations regarding the environmental and image capture conditions. For instance, for the traffic sign recognition study we address in this paper, we perform the following steps: (1) We create a traffic sign 3D model for a given unknown sign; (2) we use the SYNTHIA environment [44] to populate the 3D model in locations predefined for traffic signs; (3) we automatically aim the camera that captures images towards these locations, varying the capturing angle and distance between the camera and the traffic sign, as well as the scene illumination. This procedure ensures visual variability in the collected images due to the fact that environmental shadows influence the captures, as well as global illumination, resolution, etc. The same procedure can be used to capture synthetic images of known classes intended to be used in the training of the domain-adaptation GAN. In the case of the traffic signs, using the pixel-wise semantic segmentation GT provided by the virtual environment (SYNTHIA), we create corresponding 2D bounding boxes, which we crop to obtain the final synthetic image samples.

As we have mentioned before, synthetic images depicting instances of new classes must still be transformed by means of a GAN in order to alleviate domain shift effects. With this aim, we used the publicly available implementation of CycleGAN—as detailed in [97]—which we train using images

of known classes taken from the synthetic and real-world domains. The adversarial loss aiming at approaching the appearance of synthetic and real-world images is defined as follows:

$$\mathcal{L}_{adv}(G_{A\to B}, D_B, \mathcal{I}^A) = \mathbb{E}_{i\sim\mathcal{I}^A}[log(D_B(G_{A\to B}(i)))], \tag{1}$$

where A and B are different domains (synthetic or real in our case), $G_{A\to B}$ refers to the GAN generator from domain A to domain B, D_B refers to the GAN discriminator that distinguishes between images really coming from domain B and those put out by $G_{A\to B}$, and $\mathcal{I}^A$ is a set of images from domain A. The GAN discriminator is trained according to the following loss:

$$\mathcal{L}_{disc}(G_{A\to B}, D_B, \mathcal{I}^A, \mathcal{I}^B) = \mathbb{E}_{i^B\sim\mathcal{I}^B}[log(D_B(i^B))] + \mathbb{E}_{i^A\sim\mathcal{I}^A}[log(1 - D_B(G_{A\to B}(i^A)))]. \tag{2}$$

In addition, CycleGAN uses additional losses to force the image appearance to be transformed between domains without affecting the semantic content of the transformed images. In particular, the following cyclical reconstruction loss is used:

$$\mathcal{L}_{rec}(G_{A\to B}, G_{B\to A}, \mathcal{I}^A) = \mathbb{E}_{i\sim\mathcal{I}^A}[\|G_{B\to A}(G_{A\to B}(i)) - i\|_1], \tag{3}$$

which is complemented (regularized) with an additional loss aiming at not only ensuring in-domain content reconstruction, but also across-domain content similarity:

$$\mathcal{L}_{sim}(G_{A\to B}, \mathcal{I}^A) = \mathbb{E}_{i\sim\mathcal{I}^A}[\|G_{A\to B}(i) - i\|_1]. \tag{4}$$

Now, we can define the total loss function to train the GAN generator that transforms images from a synthetic domain $\mathcal{S}$ to a real domain $\mathcal{R}$ as follows:

$$\begin{aligned}\mathcal{L}(G_{\mathcal{S}\to\mathcal{R}}, G_{\mathcal{R}\to\mathcal{S}}, D_\mathcal{S}, D_\mathcal{R}, \mathcal{I}^\mathcal{S}, \mathcal{I}^\mathcal{R}) = {} & \mathcal{L}_{adv}(G_{\mathcal{S}\to\mathcal{R}}, D_\mathcal{R}, \mathcal{I}^\mathcal{S}) + \mathcal{L}_{adv}(G_{\mathcal{R}\to\mathcal{S}}, D_\mathcal{S}, \mathcal{I}^\mathcal{R}) \\ & + \mathcal{L}_{disc}(G_{\mathcal{S}\to\mathcal{R}}, D_\mathcal{R}, \mathcal{I}^\mathcal{S}, \mathcal{I}^\mathcal{R}) + \mathcal{L}_{disc}(G_{\mathcal{R}\to\mathcal{S}}, D_\mathcal{S}, \mathcal{I}^\mathcal{R}, \mathcal{I}^\mathcal{S}) \\ & + \mathcal{L}_{sim}(G_{\mathcal{S}\to\mathcal{R}}, \mathcal{I}^\mathcal{S}) + \mathcal{L}_{sim}(G_{\mathcal{R}\to\mathcal{S}}, \mathcal{I}^\mathcal{R}) \\ & + \mathcal{L}_{rec}(G_{\mathcal{S}\to\mathcal{R}}, G_{\mathcal{R}\to\mathcal{S}}, \mathcal{I}^\mathcal{S}) + \mathcal{L}_{rec}(G_{\mathcal{R}\to\mathcal{S}}, G_{\mathcal{S}\to\mathcal{R}}, \mathcal{I}^\mathcal{R}).\end{aligned} \tag{5}$$

At training time, we use $\mathcal{I}_\mathcal{K}$ and $\widetilde{\mathcal{I}}_\mathcal{K}$ as $\mathcal{I}^\mathcal{R}$ and $\mathcal{I}^\mathcal{S}$ image sets, respectively. Then, the learned generator $G_{\mathcal{S}\to\mathcal{R}}$ will be the CNN that we use to transform a set of synthetic images $\widetilde{\mathcal{I}}_\mathcal{U}$ into $\mathcal{G}^{\widetilde{\mathcal{I}}_\mathcal{U}}$.

4. Experimental Results

The experiments were designed to address two questions. Since we use synthetically generated instances of unknown classes to retrain the current classifier, we will have a domain shift problem. **(Q1)** *Can we reduce this domain shift by applying an image-to-image translation GAN to the samples of the unknown classes, provided that such a GAN was trained only with samples of the known classes?* and **(Q2)** *What are the overall classification results when training the classifier using the real-world data of the known classes with the data generated for the new classes following this GAN-based proposal?*

Note that question **Q1** focuses on classification results in terms of new classes in isolation, while **Q2** addresses the ultimate question, since we combine real-world samples from known classes with generated samples from unknown classes for training of the all-classes classifier. In the following, Section 4.1 introduces the synthetic and real-world datasets used in our experiments, and Section 4.2 elaborates the designed experiments to answer these questions, along with the obtained results and corresponding discussion.

4.1. Datasets

In order to perform our experiments, we need a dataset based on real-world images of traffic signs as well as another based on synthetic images. We selected the widely used Tsinghua traffic sign

dataset [51] and a synthetic analog that we created to perform the research in this paper, which we call SYNTHIA-TS. We briefly describe them in the following.

Tsinghua is a dataset composed of outdoor scenes captured in China while driving a car in urban scenarios. Following the approach proposed in [51], we cropped the traffic signs and removed all the classes with less than 100 samples. The resulting dataset is composed of 21,721 cropped images, representing 42 traffic sign classes. In terms of appearance, these classes can be hierarchically organized as shown in Figure 3, where the first criterion of splitting the dataset is the external shape of the traffic signs, and the second is the textual/graphical content of the signs. Both the shape and content define the semantics of each traffic sign, i.e., the class.

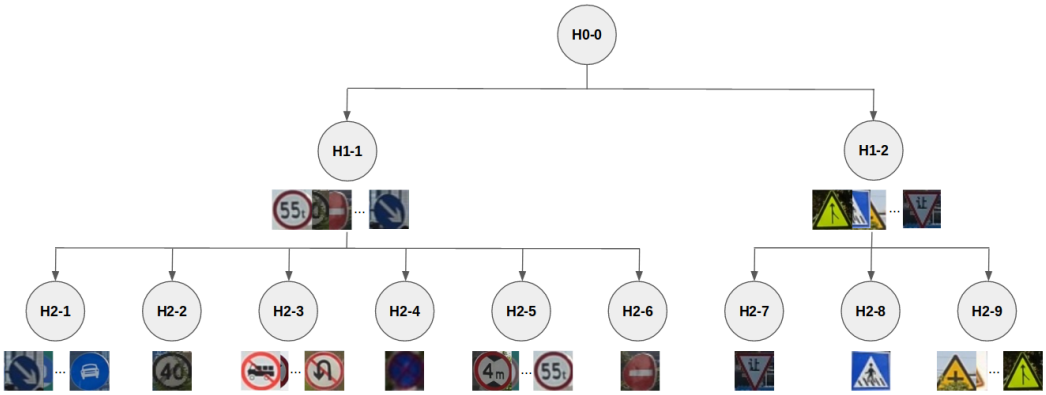

Figure 3. Hierarchy of Tsinghua traffic signs.

SYNTHIA-TS was created by mimicking the 42 classes considered from the Tsinghua dataset, using one textured 3D model per each of those classes. Then, following the protocol explained in Section 3.2, we acquired traffic sign images within the SYNTHIA environment. The generated data is balanced for all image acquisition conditions and classes. We generated 23,222 instances in total, covering the 42 classes. Since the SYNTHIA environment was previously created for multiple purposes, obtaining these instances from it took less than 2 h using a desktop PC based on an INTEL Core i7 CPU and one NVIDIA Geforce GTX 1080 GPU.

4.2. Experiments: Design, Results, and Discussion

We have not only considered the Tsinghua and SYNTHIA-TS datasets as a whole, i.e., H0-0 in terms of the hierarchy shown in Figure 3; instead, in order to perform a finer-grained analysis regarding questions **Q1** and **Q2**, we also conducted experiments based on different nodes of this hierarchy, which we call *splits*. Accordingly, our setup assumes that we have an existing split s_1 of real-world annotated images for training, and that we also want to learn a new split s_2, for which we have no access to a proper amount of corresponding real-world images and, therefore, we have to synthesize them. It is understood that s_1 and s_2 have no intersection between classes. On the other hand, for the purpose of performing comparative evaluations in our experimental setting, we do in fact have access to the real-world annotated images of split s_2.

Since we will be referring to splits coming from synthetic and real-world data, the former sometimes transformed by a GAN, and the latter sometimes used as training or testing data, we have defined the compact notation of Table 1, which will allow us to be precise and concise when describing the multiple experiments we report in this section. Using this notation and given two splits s_1 and s_2, an example of an experiment for **Q2** would consist of using $\mathcal{T}^T_{s_1}$ and $\mathcal{S}^T_{s_2}$ to train a traffic sign classifier for the known classes in split s_1 together with the new classes in split s_2, which we would like to be accurate when testing in $\mathcal{T}^C_{s_1 \cup s_2}$, i.e., accurate for all classes. Alternatively, if we use a GAN to transform the synthetic images, then the training of the classifier would be done with $\mathcal{T}^T_{s_1}$ and $\mathcal{G}^{s_2}{}_{s_1}$. In fact,

we transformed all of the synthetic images at once, which took less than 1.5 h using a desktop PC based on an INTEL Core i7 CPU and one NVIDIA GeForce TITAN X Pascal GPU.

Table 1. Basic notation for data subsets.

$\mathcal{S}_s^T$	Synthetic training data from split s
$\mathcal{T}_s^T$	Tsinghua training data in split s
$\mathcal{G}^{s_2}{}_{s_1}$	Dataset $\mathcal{G}^{s_2}$ generated by applying a GAN trained on splits $\mathcal{S}_{s_1}^T$ & $\mathcal{T}_{s_1}^T$ to split $\mathcal{S}_{s_2}^T$
$\mathcal{T}_s^C$	Tsinghua testing (classification) data from split s

As we can see in Figure 3, we have three hierarchical levels: (1) The whole data, (2) two splits based on external shape, and (3) given a shape, different data based on content. Each considered split is defined in Table 2, which specifies their features. We do not consider splits with only one class (i.e., H2-2, H2-4, H2-6, H2-7, and H2-8) since they would not allow the addressing of **Q1** (for which at least two classes are needed). However, note that although these splits are not considered in isolation, their data is considered when working with a split corresponding to their parent nodes in the hierarchy.

Table 2. Splits of Figure 3 used in appearance-driven experiments. Tsinghua samples are divided as training and testing tasks (details in main text). SYNTHIA-TS samples are used in training tasks only.

Split Code	Num. Classes	Samples Tsinghua	Samples SYN.-TS	Content
H0-0	42	21,721	23,222	All the traffic sign classes are considered
H1-1	35	20,256	19,507	Only circular traffic signs
H1-2	7	1465	3773	Only non-circular traffic signs
H2-1	6	3713	3248	Mandatory actions (circular, blue backg., white border).
H2-3	10	4012	5885	Prohibition actions (circular, white backg., red border, and diag. line)
H2-5	16	7316	8896	Information (circular, white backg., with red border)
H2-9	5	989	2713	Working areas (triangular, yellow backg., black border)

Now, we start the experiments by establishing the upper and lower bounds of different traffic sign classifiers. In these experiments, we use the full Tsinghua and SYNTHIA-TS datasets. Therefore, in this case, we use the split H0-0 (Figure 3) for Tsinghua, i.e., we use $\mathcal{T}_{H0-0}^T$ and $\mathcal{T}_{H0-0}^C$ for training and testing, respectively. Both sets have samples of all of the traffic signs that we consider. More specifically, for each class, 60% of the samples are used for training tasks (CycleGAN and traffic sign classifiers) and the remaining 40% for testing traffic sign classifiers. The per-class training/testing sampling is performed randomly and once. Training on $\mathcal{S}_{H0-0}^T$ and testing on $\mathcal{T}_{H0-0}^C$ acts as the lower bound, since we are using only synthetic images (as they come from the virtual environment); therefore, we must expect a domain shift. Training on $\mathcal{T}_{H0-0}^T$ acts as the upper bound, since we are using real-world images from the same distribution (camera and world area) as in the testing set. Table 3 shows these upper and lower bound results for the different architectures that we have considered, namely VGG16 and ResNet101. Moreover, since during the training of CNNs, there is certain amount of randomness (e.g., when sampling the datasets during a mini-batch), we repeat each training five times and report testing accuracy in terms of the mean and standard deviation of the F1 classification score (i.e., F1 = (2TP)/(2TP + FN + FP)) computed on the respective classification results. These results show that: (1) We can achieve a high classification accuracy with the appropriate real-world data; (2) using the synthetic data for training produces a reasonable accuracy (far from random), but there is a dramatic domain shift, with results dropping from 97.59% to 36.05% for VGG16, and from 98.76% to 58.74% for ResNet101.

Table 3. Lower and upper bounds for traffic sign classification on $\mathcal{T}^C_{H0-0}$. Average and standard deviation F1 scores for five training-testing runs are shown. The lower bound corresponds to training only with synthetic data ($\mathcal{S}^T_{H0-0}$), while the upper bound corresponds to training with real data ($\mathcal{T}^T_{H0-0}$).

Training Set	VGG16	ResNet101
$\mathcal{S}^T_{H0-0}$	36.05 ± 3.92	58.74 ± 2.04
$\mathcal{T}^T_{H0-0}$	97.59 ± 0.15	98.76 ± 0.14

Tables 4 and 5 report results to answer **Q1**. We consider paired splits—one is used as the set of known classes (s_k), and the other as the set of unknown classes (s_u). These splits do not intersect, but their union does not necessarily correspond to the full traffic sign hierarchy, because only splits from Table 2 are considered. The pairs were designed to force different global appearances between known and unknown classes. The $\mathcal{S}^T_{s_u}$ columns report the lower bound of classification accuracy for each experiment, i.e., training a classifier for classes in s_u with samples in $\mathcal{S}^T_{s_u}$ but testing on the real-world data $\mathcal{T}^C_{s_u}$. Columns $\mathcal{T}^T_{s_u}$ act as the upper bound, since training is done on real-world samples of s_u as if they were actually known. Columns $\mathcal{G}^{s_u}{}_{s_k}$ report the classification accuracies when training is done with the samples of $\mathcal{G}^{s_u}$, i.e., the samples of $\mathcal{S}^T_{s_u}$ transformed by a CycleGAN trained to perform image-to-image translation from $\mathcal{S}^T_{s_k}$ to $\mathcal{T}^T_{s_k}$. Therefore, the CycleGAN has not seen samples from classes in s_u at training time. Finally, we also include the case $\mathcal{G}^{s_u}{}_{s_u}$, where the CycleGAN has been trained using samples from the unknown set of classes. Obviously, this is not realistic in our application setting; however, it can be taken as an upper bound of the accuracy, which would be possible to achieve by using CycleGAN to transform the synthetic images. Figure 4 shows examples of the images involved in our experiments: Synthetic, real, and transformed by different CycleGANs.

Table 4. Experiments to support **Q1** (see main text). All tests are done in $\mathcal{T}^C_{s_u}$. Average and standard deviations of F1 scores are reported, since each experiment is performed five times. The column $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$ just stands for the subtraction of the means of the respective columns.

Known Classes (s_k)	Unknown Classes (s_u)	VGG16 $\mathcal{S}^T_{s_u}$	VGG16 $\mathcal{G}^{s_u}{}_{s_k}$	VGG16 $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$	VGG16 $\mathcal{G}^{s_u}{}_{s_u}$	VGG16 $\mathcal{T}^T_{s_u}$
H1-2	H1-1	39.20 ± 2.99	49.88 ± 3.68	10.68	74.55 ± 2.67	97.44 ± 0.23
H1-1	H1-2	65.44 ± 4.39	74.65 ± 4.71	09.21	94.35 ± 0.73	94.23 ± 6.11
H2-3			65.45 ± 4.05	−04.84		
H2-5	H2-1	70.29 ± 5.31	65.91 ± 9.28	−04.38	90.81 ± 0.71	99.22 ± 0.28
H2-9			74.01 ± 7.71	03.72		
H2-1			39.10 ± 2.08	−02.93		
H2-5	H2-3	42.03 ± 4.44	62.87 ± 4.11	20.84	88.15 ± 2.37	97.29 ± 0.44
H2-9			55.92 ± 0.72	13.89		
H2-1			54.51 ± 1.33	01.43		
H2-3	H2-5	53.08 ± 7.75	60.85 ± 4.90	07.77	85.25 ± 1.48	97.35 ± 0.25
H2-9			61.59 ± 2.88	08.51		
H2-1			72.51 ± 5.98	02.19		
H2-3	H2-9	70.32 ± 5.90	80.35 ± 4.09	10.03	94.77 ± 1.03	91.81 ± 7.60
H2-5			76.82 ± 4.12	06.50		

Table 5. Experiments to support **Q1** analogously to Table 4, but using ResNet101.

Known Classes (s_k)	Unknown Classes (s_u)	ResNet101 $\mathcal{S}^T_{s_u}$	ResNet101 $\mathcal{G}^{s_u}{}_{s_k}$	ResNet101 $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$	ResNet101 $\mathcal{G}^{s_u}{}_{s_u}$	ResNet101 $\mathcal{T}^T_{s_u}$
H1-2	H1-1	58.44 ± 1.92	63.00 ± 2.15	04.56	84.35 ± 1.08	98.70 ± 0.24
H1-1	H1-2	79.20 ± 4.06	88.80 ± 0.52	09.60	92.06 ± 1.66	96.26 ± 0.83
H2-3			76.66 ± 1.59	10.27		
H2-5	H2-1	66.39 ± 2.05	72.65 ± 2.90	06.26	89.71 ± 0.88	99.42 ± 0.13
H2-9			78.47 ± 3.26	12.08		
H2-1			48.12 ± 4.61	−07.01		
H2-5	H2-3	55.13 ± 2.81	68.86 ± 2.83	13.73	87.99 ± 2.86	97.84 ± 0.42
H2-9			52.70 ± 2.82	−02.43		
H2-1			58.73 ± 3.16	−02.49		
H2-3	H2-5	61.22 ± 1.87	63.61 ± 3.37	02.39	82.52 ± 0.76	97.08 ± 0.26
H2-9			65.11 ± 2.09	03.89		
H2-1			67.62 ± 5.63	−02.83		
H2-3	H2-9	70.45 ± 4.78	82.05 ± 2.29	11.60	92.83 ± 0.91	90.18 ± 0.91
H2-5			83.42 ± 2.10	12.97		

Figure 4. Sample images. The left block (4 columns) and right block (4 columns) rely on different criteria to generate their splits. Left block: Splits based on the hierarchy shown in Figure 3. Right block: Splits based on the balance between known and unknown (shown %) classes. Within each block, row-wise, we show samples from the same class of the unknown split. Within each block, from the left to the right column, we have: A SYNTHIA-TS sample from the unknown class, a SYNTHIA-TS sample from an unknown class transformed by a CycleGAN trained on SYNTHIA-TS-to-Tsinghua samples from known classes, as in the previous column but training a CycleGAN on the unknown classes, and a Tsinghua sample of an unknown class.

These results based on splits confirm the observations made for H0-0 according to Table 3; i.e., training and testing (for the unknown classes) with real-world data shows high classification accuracies, while training with the pure synthetic data and testing in the real-world data shows a significant drop of accuracy. Again, ResNet101 is more robust to domain shifts than VGG16. We can see how the gap gets larger as the number of classes based on synthetic data (unknown ones) increases. For instance, the gap for H1-1 is larger than for H1-2, both for VGG16 and ResNet101. Note that H1-1 contains 35 classes and H1-2 only seven (see Table 2). If we analyze the splits of the next hierarchical level (H2-X), the same observations hold; note that H2-3 and H2-5 (10 and 16 classes, respectively) show a larger gap than H2-1 and H2-9 (six and five classes, respectively), both for VGG16 and ResNet101.

On the other hand, CycleGAN indeed helps to significantly reduce the domain shift. When using the H1-1 split as known classes to train the CycleGAN, and applying this GAN to the synthetic images of the unknown classes—i.e., those in H1-2 split—we see ∼9 points of accuracy gain when testing in real-world images of the H1-2 split (9.21 for VGG16 and 9.60 for ResNet101). Changing the roles of these splits, the gain is 10.68 for VGG16 and 4.56 for ResNet101. However, in the two situations (H1-1/H1-2 as known/unknown and vice versa), ResNet101 reports significantly higher accuracies (more than 10 points) after the GAN-based domain adaptation of the synthetic images. In addition, for VGG16 and ResNet101, H1-2 as the split of unknown classes shows significantly higher accuracies (more than 20 points) than when it is H1-1, which is just a consequence of starting with similar accuracy differences before domain adaptation. Looking to the H2-X splits, we can see that the GAN-based domain adaptation reports significantly higher accuracy in most of the experiments. In fact, it is more interesting to analyze when it is not the case. For instance, when split H2-1 is used to train the CycleGAN, we obtain either very low accuracy gains (e.g., for VGG16, 1.43 when the unknown classes are in H2-5 and 2.19 for H2-9) or even negative adaptation (e.g., –2.93 for H2-3 with VGG16, and for H2-3/5/9 with ResNet101). We think that, when using H2-1 to train the CycleGAN, the learned image-to-image transform is too biased towards a blue background, which is a color not present in the rest of the considered H2-X splits (in the role of unknown classes). When exchanging the roles between H2-1 and the rest of the considered H2-X splits, the conclusion is the same for VGG16. However, ResNet101 is still able to extract the most from the domain-adapted images, showing significant accuracy gains with respect to using the synthetic images as they come directly from the virtual environment. Figure 5 presents some visual hints. For instance, when split H2-1 is used to train the CycleGAN, this adds a bluish color to the transformed images; when the CycleGAN is trained with the H2-9 split, the added color is yellowish. The former is more marked than the latter, which may be the reason behind some of the previously mentioned cases of poor domain adaptation. We can see other effects, like blue background images going to black backgrounds. According to the reviewed results, ResNet101 seems more robust to this effect than VGG16 (see the case of s_u = H2-1 in Tables 4 and 5).

Tables 4 and 5 help to analyze results in scenarios where there are significant visual differences among the known/unknown classes. We are also interested in analyzing different balances between known and unknown classes. Analogously to Table 1, we will define splits denoted by a percentage of classes; e.g., 100% would be H0-0. Each of these splits also has a complementary one with the remaining classes. When forming the new splits, in order to be sure that we do not degenerate in the previous hierarchy-based experiments, the classes are not sampled from H0-0, but they are proportionally and randomly sampled from all of the H2-X splits. For example, 50% would consider half of the H2-1, H2-3, H2-5, and H2-9 classes, added to H2-2, H2-6, and H2-8, which only have one class. Tables 6 and 7 present the corresponding results. We can see how previous observations are confirmed, namely: (1) Domain gap increases with the number of synthetic classes (the unknown ones) to be covered by the traffic sign classifier, but still, the obtained accuracies are reasonable; (2) CycleGAN is able to dramatically reduce the domain shift for the unknown classes, recovering from ∼10 to even ∼30 points of accuracy; (3) ResNet101 is able to produce the best results before and after domain adaptation.

Overall, to already answer **Q1**, we see that using known classes to train a GAN-based transformation from synthetic to real-world domains indeed helps to dramatically reduce the

classification accuracy gap due to the domain shift for synthetically generated new classes. However, there are scenarios more favorable than others, and there is still room for improvement.

Figure 5. Samples based on H2-X splits (Figure 3). Rows: Splits with classes in the role of unknown. Left-to-right columns: Samples from SYNTHIA-TS, SYNTHIA-TS samples transformed by a CycleGAN trained on SYNTHIA-TS, and Tsinghua samples of classes in H2-1 split—which play the role of known classes here—, analogous for H2-3 instead of H2-1, for H2-5 and H2-9, and samples from Tsinghua.

First, the CNN used matters. Here, Resnet101 shows significantly better classification accuracies than those of VGG16; i.e., ResNet101 is more robust to this kind of known/unknown class setting. We can see this by looking at Tables 4 and 5. Note how, for H2-X splits, when the split of classes used to train CycleGAN is the same split as that used to train the traffic sign classifier ($\mathcal{G}^{s_u}{}_{s_u}$ columns), then the classification accuracies of VGG16 and ResNet101 are similar, and VGG16 even outperforms ResNet101 several times. A similar effect can be appreciated in Tables 6 and 7. Hence, ResNet101 seems to be more robust to image imperfections introduced by CycleGAN. In this favorable but unrealistic setting, the domain-adapted images show fewer artifacts (see Figures 4 and 5).

Table 6. Experiments to support **Q1** (see main text). All tests are done in $\mathcal{T}^C_{s_u}$. Average and standard deviations of F1 scores are reported, since each experiment is performed five times. The column $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$ just stands for the subtraction of the means of the respective columns.

Known (s_k)/Unknown (s_u) % Classes ; Num. Classes	**VGG16** $\mathcal{S}^T_{s_u}$	**VGG16** $\mathcal{G}^{s_u}{}_{s_k}$	**VGG16** $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$	**VGG16** $\mathcal{G}^{s_u}{}_{s_u}$	**VGG16** $\mathcal{T}^T_{s_u}$
∼88%/12%; 37/05	82.58 ± 3.62	91.84 ± 0.92	09.26	99.08 ± 0.42	99.85 ± 0.18
∼76%/24%; 32/10	78.72 ± 1.57	88.74 ± 0.86	10.02	95.00 ± 0.54	99.30 ± 0.20
∼50%/50%; 20/21	41.80 ± 2.85	74.11 ± 1.36	32.31	85.85 ± 1.15	97.75 ± 0.46
∼24%/76%; 10/32	43.81 ± 2.22	71.31 ± 2.64	27.50	82.62 ± 2.47	97.53 ± 0.30
∼12%/88%; 05/37	40.35 ± 2.69	50.24 ± 1.25	09.89	76.26 ± 3.63	94.93 ± 0.34

Table 7. Experiments to support **Q1** analogously to Table 6, but using ResNet101.

Known (s_k)/Unknown (s_u) % Classes ; Num. Classes	**ResNet101** $\mathcal{S}^T_{s_u}$	**ResNet101** $\mathcal{G}^{s_u}{}_{s_k}$	**ResNet101** $\mathcal{G}^{s_u}{}_{s_k} - \mathcal{S}^T_{s_u}$	**ResNet101** $\mathcal{G}^{s_u}{}_{s_u}$	**ResNet101** $\mathcal{T}^T_{s_u}$
∼88%/12%; 37/05	97.54 ± 0.69	97.24 ± 2.27	09.70	99.47 ± 0.46	100.00 ± 0.00
∼76%/24%; 32/10	78.33 ± 3.08	88.12 ± 1.33	09.79	95.02 ± 0.84	99.69 ± 0.14
∼50%/50%; 20/21	63.78 ± 1.66	78.84 ± 2.24	15.06	87.11 ± 1.20	98.46 ± 0.24
∼24%/76%; 10/32	60.39 ± 3.27	77.68 ± 1.85	17.29	86.34 ± 1.05	98.75 ± 0.13
∼12%/88%; 05/37	60.17 ± 2.61	59.12 ± 2.61	−01.05	84.42 ± 0.76	96.22 ± 0.06

Second, the most adverse scenario is indeed when known and unknown classes show very different appearances combined with a low known/unknown class ratio. In a reasonable case, as for $\sim$(88%/12%) splits of randomly selected known/unknown classes, using ResNet101, we can see how we obtain a classification accuracy of 97.24 on average (Table 7) where training with real-world data reaches 100.00. In the most imbalanced case, $\sim$(12%/88%), ResNet101 reaches a classification accuracy of 77.85, still far from the 98.82 when training with real-world images. Note that in this scenario, there is room to improve GAN-based image-to-image translation, since even using the classification classes to train the CycleGAN, the obtained accuracy is 84.93, still far from 98.82.

Although in this paper, these are just intermediate results on our way to address **Q2**, this analysis is already useful if our goal is to perform transfer learning for the traffic sign classifier; i.e., if we want to train a classifier that only needs to operate in a new set of traffic signs for which we do not have enough samples, and we want to leverage knowledge from the known classes even if these are not going to be used for classification anymore—for instance, in particular environments with specialized traffic signs, like in some closed infrastructures or industrial facilities. In fact, the vision system does not need to be onboard a vehicle; it can even be on a humanoid or any other robotic platform. However, a probable requisite in this case would be to use the same camera sensor model to classify new classes as that used for collecting the real-world images involved in the training of the GAN-based domain adapter.

Finally, in order to address **Q2**, we performed the experiments shown in Tables 8 and 9. In these tables, columns $\mathcal{S}^T_{H0-0} + \mathcal{T}^T_{s_k}$ refer to training jointly with synthetic and real-world data. $\mathcal{S}^T_{H0-0}$ is the full SYNTHIA-TS set, i.e., covering known (k) and unknown (u) classes. Therefore, we use all of the synthetic data available for the classes we want to classify. On the other hand, $\mathcal{T}^T_{s_k}$ refers to the training set of all known real-world classes; in these experiments, $k \cup u$ are the 42 considered classes of split H0-0. Therefore, $\mathcal{T}^T_{s_k} + \mathcal{T}^T_{s_u}$ equals the full Tsinghua training set ($\mathcal{T}^T_{H0-0}$). Columns $\mathcal{G}^{H0-0}{}_{s_k} + \mathcal{T}^T_{s_k}$ are analogous to previous ones, but in this case, rather than using the synthetic data as gathered from the virtual environment, we use it transformed by a CycleGAN. These GANs are trained only using the known classes in each experiment, i.e., in this case, $\mathcal{T}^T_{s_k}$ and $\mathcal{S}^T_{s_k}$. Accordingly, $\mathcal{G}^{H0-0}{}_{s_k}$ is composed by $\mathcal{G}^{s_k}{}_{s_k}$, used as pure data augmentation, as well as $\mathcal{G}^{s_u}{}_{s_k}$, which is really needed, since we assume that we do not have real-world training samples for the split s_u.

Table 8. Experiments to support **Q2** (see main text), all done in $\mathcal{T}^C_{H0-0}$. Average and standard deviations of F1 scores are reported, since each experiment is performed five times. This is done for the all-classes classification problem, but we also show detailed results for known and unknown classes. Table 3 shows the lower and upper bounds for these experiments, i.e., training only on either SYNTHIA-TS or Tsinghua data. In terms of average F1, these bounds are 36.05 and 97.59, respectively.

Unknown Classes (s_u)	**VGG16, $\mathcal{S}^T_{H0-0} + \mathcal{T}^T_{s_k}$ H0-0 (s_k/s_u)**	**VGG16, $\mathcal{G}^{H0-0}{}_{s_k} + \mathcal{T}^T_{s_k}$ H0-0 (s_k/s_u)**
H1-1	37.12 ± 2.47 ($59.83 \pm 8.55/32.58 \pm 2.90$)	59.54 ± 3.60 ($84.97 \pm 2.89/54.45 \pm 3.82$)
H1-2	88.81 ± 1.70 ($95.92 \pm 0.74/53.22 \pm 9.91$)	92.88 ± 1.42 ($96.77 \pm 0.62/73.46 \pm 6.21$)
12%	87.53 ± 1.55 ($92.50 \pm 0.81/50.69 \pm 7.88$)	93.79 ± 1.16 ($94.73 \pm 1.16/86.83 \pm 2.11$)
24%	78.17 ± 1.81 ($88.21 \pm 0.68/46.05 \pm 5.79$)	89.91 ± 2.19 ($92.84 \pm 1.17/80.53 \pm 5.79$)
50%	56.52 ± 3.84 ($75.85 \pm 1.50/35.26 \pm 6.60$)	74.96 ± 1.40 ($85.01 \pm 1.26/63.90 \pm 2.73$)
76%	30.69 ± 2.13 ($49.09 \pm 1.67/24.94 \pm 3.10$)	67.45 ± 1.41 ($74.04 \pm 2.71/65.39 \pm 1.28$)
88%	31.46 ± 3.18 ($33.34 \pm 3.66/31.23 \pm 3.27$)	53.62 ± 2.26 ($64.44 \pm 4.74/50.70 \pm 2.46$)

Table 9. Experiments to support **Q2** analogously to Table 8, but using ResNet101. In this case, the lower and upper bounds are 58.74 and 98.76, respectively.

Unknown Classes (s_u)	ResNet101, $\mathcal{S}^T_{H0-0} + \mathcal{T}^T_{s_k}$ H0-0 (s_k/s_u)	ResNet101, $\mathcal{G}^{H0-0}{}_{s_k} + \mathcal{T}^T_{s_k}$ H0-0 (s_k/s_u)
H1-1	65.67 ± 0.66 ($95.42 \pm 0.80/59.72 \pm 0.68$)	70.65 ± 1.60 ($98.53 \pm 0.38/65.08 \pm 1.97$)
H1-2	95.55 ± 0.39 ($98.37 \pm 0.27/81.45 \pm 3.30$)	97.57 ± 0.31 ($98.77 \pm 0.11/91.55 \pm 1.33$)
12%	90.05 ± 1.67 ($94.11 \pm 0.62/60.00 \pm 9.71$)	95.54 ± 0.82 ($96.44 \pm 0.47/88.95 \pm 3.49$)
24%	82.70 ± 1.22 ($91.02 \pm 0.63/56.07 \pm 3.15$)	92.57 ± 0.36 ($95.00 \pm 0.50/84.78 \pm 0.77$)
50%	68.03 ± 2.03 ($81.92 \pm 0.95/52.74 \pm 3.32$)	81.08 ± 0.57 ($87.72 \pm 0.60/73.78 \pm 0.85$)
76%	49.23 ± 1.17 ($50.94 \pm 1.66/48.70 \pm 1.06$)	71.94 ± 1.76 ($71.50 \pm 2.38/72.08 \pm 1.75$)
88%	52.95 ± 2.31 ($43.77 \pm 1.99/54.06 \pm 2.53$)	56.56 ± 2.56 ($56.83 \pm 1.93/55.43 \pm 2.70$)

We see that the observations done in the context of **Q1** apply here too: (1) The domain gap increases with the number of synthetic classes (the unknown ones); (2) CycleGAN is able to significantly reduce the domain shift; (3) ResNet101 performs better than VGG16 before and after domain adaptation; even when using the full Tsinghua training set, they report similar upper bounds (97.59 for VGG16, 98.76 for ResNet101). In the case of ResNet101, the best cases almost reach the upper bound: (1) When the known classes are those in split H1-1 ($\sim$83% of classes) and the unknown in split H1-2 ($\sim$17% of classes), we obtain 97.57, which is very close to 98.76; (2) when classes are directly selected randomly on the H2-X hierarchy level, the case of 12% of unknown classes reaches 95.54, which is also a very high accuracy; even the 24% case still reports 92.57. Moreover, in all of these cases, the accuracy of the known classes keeps over 95 and over 84 for the unknown ones (91.55 for H1-2, 88.95 for the 12%, 84.78 for the 24%). Overall, we can conclude that with ResNet101, the proposed method works well when the ratio of unknown/known classes is of $\sim 1/4$. In order to reach the upper bound, we can investigate if we can still improve CycleGAN, but in this $\sim 1/4$ regime, the *last mile* can probably be covered by adding a small number of real-world collected and annotated samples from the unknown classes. As the vision system keeps performing in the real-world, the samples falling in the new classes can be kept; then, we can replace the synthesized and transformed samples by these self-annotated ones in a future retraining of the classifier. In fact, these self-annotation cycles can be also a good approach for more challenging ratios of unknown/known classes; note that for the 76% of unknown classes, the results are over 70, and over 50 for the 88%.

5. Conclusions

There are situations where a computer vision system may need to recognize new (previously unknown) classes, but the lack of samples from such classes (i.e., raw images with annotations) may seriously delay this possibility. In this paper, we have explored how to address this situation by using synthetic data and leveraging samples from the classes already known to the system. Since there is a domain shift between synthetic and real worlds, addressing the problem involves incorporating some kind of domain adaptation. To solve the problem of the lack of data, we have proposed to teach a GAN using (the already available) samples from the known classes, and to apply it to adapt synthetic samples from the new classes. As a proof of concept, we have focused on traffic sign recognition. We have used the publicly available Tsinghua dataset and we have created a synthetic dataset (SYNTHIA-TS) for designing the experiments presented in this paper. In particular, the experiments have been designed to address two questions. First, we addressed the intermediate question *can we reduce the synthetic-to-real domain shift by applying an image-to-image translation GAN to the unknown classes, provided that such a GAN was trained only with the known classes?* After an extensive set of experiments and results that we have presented and discussed, the answer was positive, which leads us to the main question, namely, *what are the overall classification results when training the classifier using the real-world data of the known classes with the data generated for the new classes following our proposal?* Again, the obtained results allow us to conclude that the proposed method is indeed effective, provided that we use a proper CNN to

perform the classifications task, as well as a proper GAN to transform the synthetic images. Here, a ResNet101-based classifier and domain adaptation based on CycleGAN performed extremely well for ratios of unknown/known classes of even ∼1/4. For more challenging ratios such as ∼4/1, the results are also very positive. As a matter of fact, instead of focusing on improving the components of the presented method, as future work, we plan to augment this method with complementary techniques such as self-annotation, i.e., using the classifier generated with our current method to self-annotate real-world samples of the new classes for a posterior retraining/fine-tuning of the classifier. The datasets of synthetic traffic signs used in this paper are publicly available at www.synthia-dataset.net.

Author Contributions: G.V. implemented the CNN-related code and ran all of the experiments (including the training and testing of all CNNs). He was also actively involved in the writing of the paper (including the discussion of the results), together with A.M.L. and J.V.d.W. A.M.L. and J.V.d.W. proposed and supervised the research. All authors have read and agreed to the published version of the manuscript.

Funding: Antonio and Gabriel acknowledge the financial support by the Spanish project TIN2017-88709-R (MINECO/AEI/FEDER, UE). Joost acknowledges the financial support by the Spanish project TIN2016-79717-R. Antonio thanks ICREA under the ICREA Academia Program for the financial support. Finally, as CVC members, the authors thank the Generalitat de Catalunya CERCA Program and its ACCIO agency.

Acknowledgments: We would like to thank Joan Serrat and Luis Herranz, as well as Idoia Ruiz, Xialei Liu, and Marc Masana, all them CVC/UAB members, for fruitful discussions on all kinds of topics related to lifelong learning, including data generation as presented in this paper.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. *Neural Inf. Process. Syst.* **2012**, *25*. [CrossRef]
2. Sharma, N.; Jain, V.; Mishra, A. An Analysis Of Convolutional Neural Networks For Image Classification. *Procedia Comput. Sci.* **2018**, *132*, 377–384. [CrossRef]
3. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *Neural Inf. Process. Syst.* **2015**, *1*, 91–99. [CrossRef] [PubMed]
4. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016.
5. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
6. Chabot, F.; Chaouch, M.; Rabarisoa, J.; Teuliere, C.; Chateau, T. Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
7. Mousavian, A.; Anguelov, D.; Flynn, J.; Kosecka, J. 3D bounding box estimation using deep learning and geometry. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
8. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
9. Noh, H.; Hong, S.; Han, B. Learning deconvolution network for semantic segmentation. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
10. Yu, F.; Koltun, V. Multi-scale context aggregation by dilated convolutions. *arXiv* **2016**, arXiv:1511.07122.
11. Uhrig, J.; Cordts, M.; Franke, U.; Brox, T. Pixel-level encoding and depth layering for instance-level semantic labelling. In Proceedings of the German Conference on Pattern Recognition (GCPR), Hannover, Germany, 12–15 September 2016.
12. Bai, M.; Urtasun, R. Deep watershed transform for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.

13. Liu, S.; Jia, J.; Fidle, S.; Urtasun, R. SGN: Sequential grouping networks for instance segmentation. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
14. Uhrig, J.; Rehder, E.; Fröhlich, B.; Franke, U.; Brox, T. Box2Pix: Single-Shot Instance Segmentation by Assigning Pixels to Object Boxes. In Proceedings of the Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018.
15. Cao, Z.; Simon, T.; Wei, S.; Sheikh, Y. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
16. Sun, K.; Xiao, B.; Liu, D.; Wang, J. Deep High-Resolution Representation Learning for Human Pose Estimation. *arXiv* **2019**, arXiv:1902.09212.
17. Li, J.; Wang, C.; Zhu, H.; Mao, Y.; Fang, H.; Lu, C. CrowdPose: Efficient Crowded Scenes Pose Estimation and A New Benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
18. Godard, C.; Aodha, O.; Brostow, G. Unsupervised monocular depth estimation with left-right consistency. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
19. Gurram, A.; Urfalioglu, O.; Halfaoui, I.; Bouzaraa, F.; Lopez, A.M. Monocular depth estimation by learning from heterogeneous datasets. In Proceedings of the Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018.
20. Gan, Y.; Xu, X.; Sun, W.; Lin, L. Monocular Depth Estimation with Affinity, Vertical Pooling, and Label Enhancement. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
21. Fu, H.; Gong, M.; Wang, C.; Batmanghelich, K.; Tao, D. Deep ordinal regression network for monocular depth estimation. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–21 June 2018.
22. Pillai, S.; Ambrus, R.; Gaidon, A. SuperDepth: Self-Supervised, Super-Resolved Monocular Depth Estimation. In Proceedings of the International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019.
23. Guizilini, V.; Li, J.; Ambrus, R.; Pillai, S.; Gaidon, A. Robust Semi-Supervised Monocular Depth Estimation with Reprojected Distances. *arXiv* **2019**, arXiv:1910.01765.
24. Ambrus, R.; Guizilini, V.; Li, J.; Pillai, S.; Gaidon, A. Two Stream Networks for Self-Supervised Ego-Motion Estimation. *arXiv* **2019**, arXiv:1910.01764.
25. Hestness, J.; Narang, S.; Ardalani, N.; Diamos, G.; Jun, H.; Kianinejad, H.; Patwary, M.; Yang, Y.; Zhou, Y. Deep Learning Scaling is Predictable, Empirically. *arXiv* **2017**, arXiv:1712.00409.
26. Sun, C.; Shrivastava, A.; Singh, S.; Gupta, A. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
27. Abramson, Y.; Freund, Y. SEmi-automatic VIsuaL LEarning (SEVILLE): a tutorial on active learning for visual object recognition. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 20–26 June 2005.
28. Settles, B. Active learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2012**, *6*, 1–114. [CrossRef]
29. Roy, S.; Unmesh, A.; Namboodiri, V. Deep active learning for object detection. In Proceedings of the British Machine Vision Conference (BMVC), Newcastle, UK, 3–6 September 2018.
30. H.H. Aghdam, A. Gonzalez-Garcia, J.W.A.L. Active Learning for Deep Detection Neural Networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
31. Xu, J.; Ramos, S.; Vázquez, D.; López, A. Domain Adaptation of Deformable Part-Based Models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2367–2380.
32. Xu, J.; Vázquez, D.; Mikolajczyk, K.; López, A. Hierarchical online domain adaptation of deformable part-based models. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweeden, 16–21 May 2016.

33. Zou, Y.; Yu, Z.; Kumar, B.; Wang, J. Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
34. Pan, S.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
35. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 24–27 June 2014.
36. Hoffman, J.; Wang, D.; Yu, F.; Darrell, T. FCNs in the Wild: Pixel-level Adversarial and Constraint-based Adaptation. *arXiv* **2016**, arXiv:1612.02649.
37. Zhang, Y.; David, P.; Gong, B. Curriculum Domain Adaptation for Semantic Segmentation of Urban Scenes. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
38. Chen, Y.; Li, W.; Sakaridis, C.; Dai, D.; Gool, L. Domain adaptive Faster R-CNN for object detection in the wild. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–21 June 2018.
39. Wang, M.; Deng, W. Deep Visual Domain Adaptation: A Survey. *Neurocomputing* **2018**, *312*, 135–153. [CrossRef]
40. Gidaris, S.; Singh, P.; Komodakis, N. Unsupervised Representation Learning by Predicting Image Rotations. In Proceedings of the International Conference on Learning Representation (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
41. Kim, D.; Cho, D.; Yoo, D.; Kweon, I. Learning image representations by completing damaged jigsaw puzzles. In Proceedings of the Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018.
42. Kolesnikov, A.; Zhai, X.; Beyer, L. Revisiting self-supervised visual representation learning. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
43. Xu, J.; Xiao, L.; López, A. Self-Supervised Domain Adaptation for Computer Vision Tasks. *IEEE Accesss* **2019**, *7*, 156694–156706. [CrossRef]
44. Ros, G.; Sellart, L.; Materzyska, J.; Vázquez, D.; López, A. The SYNTHIA Dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the IEEE InternationaL Conference on Computer Vision and Pattern Recognition (CVPR), Vegas, NV, USA, 27–30 June 2016.
45. Gaidon, A.; Wang, Q.; Cabon, Y.; Vig, R. Virtual Worlds as Proxy for Multi-Object Tracking Analysis. In Proceedings of the IEEE InternationaL Conference on Computer Vision and Pattern Recognition (CVPR), Vegas, NV, USA, 27–30 June 2016.
46. Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; Brox, T. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In Proceedings of the IEEE InternationaL Conference on Computer Vision and Pattern Recognition (CVPR), Vegas, NV, USA, 27–30 June 2016.
47. Richter, S.; Hayder, Z.; Koltun, V. Playing for Benchmarks. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
48. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robotics (FSR)*; Springer: Cham, Switzerland, 2017.
49. Dosovitskiy, A.; Ros, G.; Codevilla, F.; López, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. In Proceedings of the Conference on Robot Learning (CoRL), Mountain View, CA, USA, 13–15 November 2017.
50. 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 Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 8–13 December 2014.
51. Zhu, Z.; Liang, D.; Zhang, S.; Huang, X.; Li, B.; Hu, S. Traffic-Sign Detection and Classification in the Wild. In Proceedings of the IEEE InternationaL Conference on Computer Vision and Pattern Recognition (CVPR), Vegas, NV, USA, 27–30 June 2016.
52. Chen, Z.; Liu, B. *Lifelong Machine Learning*; Morgan & Claypool: San Rafael, CA, USA, 2017; ISBN 9781627055017.

53. Awasthi, A.; Sarawagi, S. Continual Learning with Neural Networks: A Review. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, Kolkata, India, 3–5 January 2019.
54. Parisi, G.; Part, R.K.J.; Kanan, C.; Wermter, S. Continual lifelong learning with neural networks. *Neural Netw.* **2019**, *113*, 54–71. [CrossRef] [PubMed]
55. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: a survey. *ACM Comput. Surv.* **2009**, *41*, 15:1–15:58. [CrossRef]
56. Pimentel, M.; Clifton, D.; Clifton, L.; Tarassenko, L. Review of novelty detection. *Signal Process.* **2014**, *99*, 215–249. [CrossRef]
57. Liang, S.; Li, Y.; Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. In Proceedings of the International Conference on Learning Representation (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
58. Masana, M.; Ruiz, I.; Serrat, J.; van de Weijer, J.; López, A. Metric Learning for Novelty and Anomaly Detection. In Proceedings of the British Machine Vision Conference (BMVC), Newcastle, UK, 3–6 September 2018.
59. Li, Z.; Hoiem, D. Learning without forgetting. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016.
60. Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 3521–3526. [CrossRef]
61. Aljundi, R.; Rahaf, C.; Tuytelaars, T. Expert gate: Lifelong learning with a network of experts. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
62. Liu, X.; Masana, M.; Herranz, L.; Weijer, J.; López, A.; Bagdanov, A. Rotate your Networks: Better Weight Consolidation and Less Catastrophic Forgetting. In Proceedings of the International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018.
63. Taylor, G.; Chosak, A.; Brewer, P. OVVV: Using virtual worlds to design and evaluate surveillance systems. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Minneapolis, MN, USA, 18–23 June 2007.
64. Marin, J.; Vázquez, D.; Gerónimo, D.; López, A. Learning appearance in virtual scenarios for pedestrian detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA, 13–18 June 2010.
65. Vázquez, D.; López, A.; Ponsa, D.; Marin, J. Cool world: domain adaptation of virtual and real worlds for human detection using active learning. In Proceedings of the Neural Information Processing Systems (NIPS) Workshop on Domain Adaptation: Theory and Applications, Granada, Spain, 2–14 December 2011.
66. Pepik, B.; Stark, M.; Gehler, P.; Schiele, B. Teaching 3D geometry to deformable part models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012.
67. Xu, J.; Vázquez, D.; López, A.; Marín, J.; Ponsa, D. Learning a part-based pedestrian detector in a virtual world. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2121–2131. [CrossRef]
68. Peng, X.; Sun, B.; Ali, K.; Saenko, K. Learning deep object detectors from 3D models. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
69. Hattori, H.; Lee, N.; Boddeti, V.; Beainy, F.; Kitani, K.; Kanade, T. Synthesizing a Scene-Specific Pedestrian Detector and Pose Estimator for Static Video Surveillance. *Int. J. Comput. Visionspec. Issue Synth. Vis. Data* **2018**, *126*, 1027–1044. [CrossRef]
70. Su, H.; Qi, C.; Li, Y.; Guibas, L. Render for CNN: viewpoint estimation in images using CNNs trained with rendered 3D model views. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
71. Barros, I.; Cristani, M.; Caputo, B.; Rognhaugen, A.; Theoharis, T. Looking Beyond Appearances: Synthetic Training Data for Deep CNNs in Re-identification. *Comput. Vis. Image Underst.* **2018**, *167*, 50–62.
72. Shotton, J.; Fitzgibbon, A.; Cook, M.; Sharp, T.; Finocchio, M.; Moore, R.; Kipmanand, A.; Blake, A. Real-time human pose recognition in parts from a single depth image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Colorado Springs, CO, USA, 20–25 June 2011.

73. Haltakov, V.; Unger, C.; Ilic, S. Framework for generation of synthetic ground truth data for driver assistance applications. In Proceedings of the German Conference on Pattern Recognition (GCPR), Saarbrücken, Germany, 3–6 September 2013.
74. Skinner, J.; Garg, S.; Sünderhauf, N.; Corke, P.; Upcroft, B.; Milford, M. High-Fidelity Simulation for Evaluating Robotic Vision Performance. In Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016.
75. Müller, M.; Smith, N.; Ghanem, B. A Benchmark and Simulator for UAV Tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016.
76. Johnson-Roberson, M.; Barto, C.; Mehta, R.; Sridhar, S.N.; Rosaen, K.; Vasudevan, R. Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks? *arXiv* **2017**, arXiv:1610.01983.
77. Tian, Y.; Li, X.; Wang, K.; Wang, F. Training and Testing Object Detectors with Virtual Images. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 539–546. [CrossRef]
78. Hernandez, D.; Schneider, L.; Espinosa, A.; Vázquez, D.; López, A.; Franke, U.; Pollefeys, M.; Moure, J. Slanted Stixels: Representing San Francisco's Steepest Streets. In Proceedings of the British Machine Vision Conference (BMVC), London, UK, 4–7 September 2017.
79. Handa, A.; Patraucean, V.; Badrinarayanan, V.; Stent, S.; Cipolla, R. Understanding Real World Indoor Scenes With Synthetic Data. In Proceedings of the IEEE InternationaL Conference on Computer Vision and Pattern Recognition (CVPR), Vegas, NV, USA, 27–30 June 2016.
80. Jiang, C.; Qi, S.; Zhu, Y.; Huang, S.; Lin, J.; Yu, L.; Terzopoulos, D.; Zhu, S. Configurable 3D Scene Synthesis and 2D Image Rendering with Per-pixel Ground Truth Using Stochastic Grammars. *Int. J. Comput. Visionspecial Issue Synth. Vis. Data* **2018**, *126*, 920–941. [CrossRef]
81. Butler, D.; Wulff, J.; Stanley, G.; Black, M. A naturalistic open source movie for optical flow evaluation. In Proceedings of the European Conference on Computer Vision (ECCV), Florence, Italy, 7–13 October 2012.
82. Mayer, N.; Ilg, E.; Fischer, P.; Hazirbas, C.; Cremers, D.; Dosovitskiy, A.; Brox, T. What Makes Good Synthetic Training Data for Learning Disparity and Optical Flow Estimation? *Int. J. Comput. Visionspec. Issue Synth. Vis. Data* **2018**, *126*, 942–960. [CrossRef]
83. Alhaija, H.; Mustikovela, S.K.; Mescheder, L.; Geiger, A.; Rother, C. Augmented Reality Meets Computer Vision: Efficient Data Generation for Urban Driving Scenes. *Int. J. Comput. Visionspec. Issue Synth. Vis. Data* **2018**, *126*, 961–972. [CrossRef]
84. Sakaridis, C.; Dai, D.; Gool, L. Semantic Foggy Scene Understanding with Synthetic Data. *Int. J. Comput. Visionspec. Issue Synth. Vis. Data* **2018**, *126*, 973–992. [CrossRef]
85. Bahnsen, C.; Vázquez, D.; López, A.; Moeslund, T. Learning to Remove Rain in Traffic Surveillance by Using Synthetic Data. In Proceedings of the International Conference on Computer Vision Theory and Applications (VISIGRAPP), Prague, Czechia, 25–27 February 2019.
86. Souza, C.; Gaidon, A.; Cabon, Y.; López, A. Procedural Generation of Videos to Train Deep Action Recognition Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
87. Varol, G.; Romero, J.; Martin, X.; Mahmood, N.; Black, M.; Laptev, I.; Schmid, C. Learning from Synthetic Humans. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
88. Chen, C.; Seff, A.; Kornhauser, A.; Xiao, J. DeepDriving: Learning affordance for direct perception in autonomous driving. In Proceedings of the International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
89. Savva, M.; Chang, A.; Dosovitskiy, A.; Funkhouser, T.; Koltun, V. MINOS: Multimodal Indoor Simulator for Navigation in Complex Environments. *arXiv* **2017**, arXiv:1712.03931.
90. Isola, P.; Zhu, J.; Zhou, T.; Efros, A. Image-to-Image Translation with Conditional Adversarial Nets. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
91. Wang, T.; Liu, M.; Zhu, J.; Tao, A.; Kautz, J.; Catanzaro, B. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.

92. Taigman, Y.; Polyak, A.; Wolf, L. Unsupervised Cross-Domain Image Generation. In Proceedings of the International Conference on Learning Representation (ICLR), Toulon, France, 24–26 April 2017.
93. Liu, M.; Breuel, T.; Kautz, J. Unsupervised Image-to-Image Translation Networks. In Proceedings of the Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017.
94. 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 (CVPR), Honolulu, HI, USA, 21–26 July 2017.
95. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.; Isola, P.; Saenko, K.; Efros, A.; Darrell, T. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. *arXiv* **2017**, arXiv:1711.03213.
96. 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 (CVPR), Honolulu, HI, USA, 21–26 July 2017.
97. Zhu, J.; Park, T.; Isola, P.; Efros, A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
98. Kim, T.; Cha, M.; Kim, H.; Lee, J.; Kim, J. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. In Proceedings of the Machine Learning Research, Sydney, Australia, 6–11 August 2017.
99. Beery, S.; Liu, Y.; Morris, D.; Piavis, J.; Kapoor, A.; Meister, M.; Joshi, N.; Perona, P. Synthetic Examples Improve Generalization for Rare Classes. *arXiv* **2019**, arXiv:1904.05916.