Reliability of GAN Generated Data to Train and Validate Perception Systems for Autonomous Vehicles

Weihuang Xu *
University of Florida
Gainesville, Florida
weihuang.xu@ufl.edu

Nasim Souly
Volkswagen Group
Innovation Center California
Belmont, California
nasim.souly@vw.com

Pratik Prabhanjan Brahma
Volkswagen Group
Innovation Center California
Belmont, California
pratik.brahma@vw.com

Abstract

Autonomous systems deployed in the real world have to deal with potential problem causing situations that they have never seen during their training phases. Due to the long-tail nature of events, collecting a large amount of data for such corner cases is a difficult task. While simulation is one plausible solution, recent developments in the field of Generative Adversarial Networks (GANs) make them a promising tool to generate and augment realistic data without exhibiting a domain shift from actual real data. In this manuscript, we empirically analyze and propose novel solutions for the trust that we can place on GAN generated data for training and validation of vision-based perception modules like object detection and scenario classification.

1. Introduction

The current surge in the development and industrialization of self-driving cars can be largely attributed to recent developments in the field of artificial intelligence. Several machine learning, especially deep learning, based modules in perception, prediction, and planning of autonomous vehicles learn and update themselves in a data-driven manner. These models typically need a lot of diverse and representative data in the development process. However, there are many real-world scenarios that either do not exist or exist in smaller quantities in finite datasets used for teaching these models. This is often referred to as the long tail problem. Another major challenge is the effort and cost associated with data labeling. In order to generate massive amounts of diverse data in a fully labeled manner for ADAS development and testing, simulation is considered a vital tool. One primary problem associated with such simulators is that of the domain shift. Even for high fidelity simulators, the way the rendering is done often does not match accurately with the physics of real-world sensors. This is where Generative Adversarial Networks (GANs) [8] have come up as an interesting alternative to generate *realistic* data.

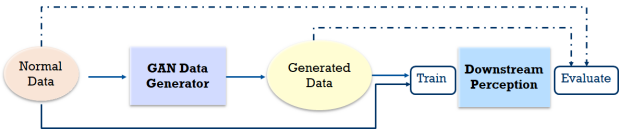

Figure 1. Exploring the usage of GAN generated out-of-distribution data for perception tasks in autonomous driving. The solid line shows the pipeline for augmenting training data using GANs. The dashed lines indicate the usage of real and generated data for evaluating downstream perception modules.

This paper aims to thoroughly study the usability and reliability of GAN generated data to tackle long-tail situations in particular, as shown in Figure 1. The major contributions of this paper can be summarized as follows:

1. Analysis of using GANs for out-of-distribution data for training two different perception tasks
2. Investigate the reliability of GAN generated data for testing or validating perception modules
3. Design GAN loss functions such that it can not only focus on visuals quality but also downstream usability of generated data.
4. Propose novel ways in using GAN for simulating corner case traffic scenarios like sharp cut-ins.

After a literature survey in Section 2, Section 3 introduces the overall system design of GAN based simulators. We explain our dataset in Section 4. Several experiments in Section 5 and 6 show how perception models behave when we consider GAN generated data. Our experiments cover the aspects of the impact of GANs for both training and testing

*This work was done during Weihuang Xu's summer internship at the Volkswagen Group Innovation Center California

of perception modules. Our conclusions are summarized in Section 7.

2. Related Work

Generative adversarial nets: GANs for image generation have generator and discriminator neural networks competing against each other. After reaching the equilibrium point through training, the generator can create *realistic* images that can somewhat confuse the discriminator in adjudging whether it is a real or fake image. The optimal GAN was proven to be the case when the generator's output distribution matches the training data distribution. Conditional GANs [22] follow a similar structure but rather focus on generating images based on a given conditional input. These conditional GANs can be useful as labeled data generation tools for autonomous perception tasks. A popular family of conditional GANs is the pix2pix [14] style models. The original pix2pix showed how realistic images could be generated from semantic segmentation masks as input. Latter works like pix2pixHD [33] have largely improved the quality and resolution. In the absence of paired data, unsupervised image to image translation networks, for instance, CycleGAN [38], and MUNIT [13], can help convert images from one domain to another. Moreover, methods such as [18] and [12] have been used to generate specific mask styles for road objects given just a context or a location. Similarly, there are also generative inpainting [36] networks that can delete specific objects and fill the cropper region by extrapolating neighborhood background.

Metrics for evaluating GANs: Several metrics like Inception score [28], Frechet Inception distance [11], Wasserstein Distance [9], Precision and Recall [17] and others [2, 30, 34] have been proposed to evaluate GANs. However, most of these metrics are only applicable when the aim is to generate new and diverse samples but still belonging to the training data distribution. These metrics also do not specify the subsequent impact on systems that are going to consume them.

GANs for data augmentation: There have been numerous papers [29, 7, 26] that show the advantages that GANs bring in. Yet, [27] recently showed that the usage of BigGAN [3] did not showcase any advantage in terms of improving image classification accuracy on ImageNet. Although this area is exciting, a relatively under-explored topic is the impact of GAN data to address rare or unseen data modes that are absent in the dataset.

Applications for autonomous systems: While GANs have been widely used to create images from semantic masks, there is also research on conversion between day and night [1], removing rain [20], fog addition [19] and several other applications as pointed out in [31] that can have potential usage for self-driving vehicles. There are also GAN based methods such as DeepRoad [37] which propose automated system testing of neural networks across multiple situations. Nevertheless, these works have not primarily touched on the trust and reliability aspects of using GANs as a substitute for real data collection, which is the main focus of this paper.

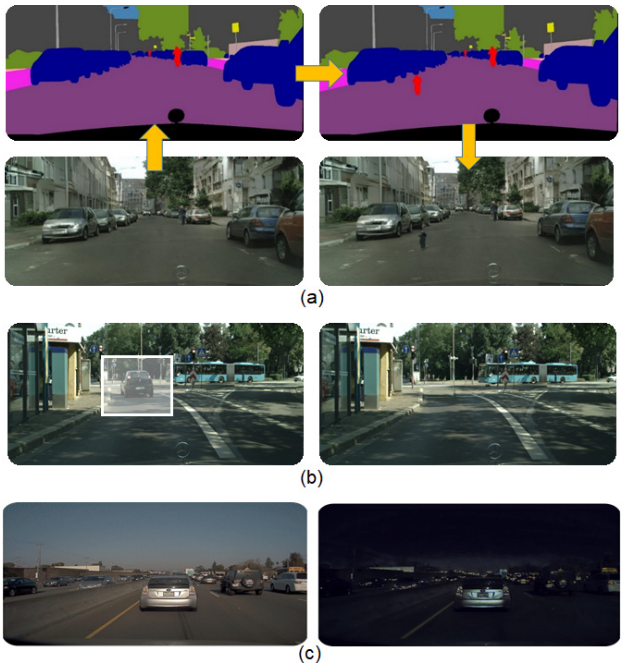

Figure 2. Example modules of GANipulator system: (a) Insertion, (b) Deletion, and (c) Style and Domain change.

3. GANs as Simulator Systems

In order for GANs to serve as a reliable substitute for graphic simulators, the overall system should be able to encompass all the features that can be achieved using conventional simulators used for perception tasks. Some of these are:

- **Insertion**: It is the ability to add an object or set of objects at a specified location in a user-friendly manner.
- **Deletion**: One should be able to remove objects from the scene and realistically fill it with the background.
- **Editing**: There should be one-to-many modules to come up with various shapes and structures of road objects and traffic agents using a friendly user interface
- **Motion and behavior**: It should simulate imagery for scenarios with different types of motion patterns of traffic agents.
- **Programmatic generation of variations**: Large GAN generated datasets can be sampled by programmatically varying conditional inputs or other features.

- **Style and domain**: One should be able to generate data as per various weather, visibility conditions, and sensor specifications.

Figure 2 illustrates some of these traits of a GAN based simulator, hereafter referred to as GANipulator, and how it can be used to create long tail situations, like inserting a child playing in the middle of the road. The subsequent experiment sections cover two pairwise combinations of a data generator and downstream perception task. The first one uses CycleGAN as the data generator and YOLOv5 to perform object detection in day and night time images. The next one applies pix2pixHD as the GANipulator module and a CNN-LSTM [6] as the perception task to detect sharp cut-in behavior of other vehicles in front of the ego vehicle. In the following sections, we report the implementation details and results of using GANs for two different tasks, namely object detection and scenario (sharp cut-in) detection. For these, we apply two different types of GANs, CycleGAN and pix2pixHD respectively, to evaluate whether GANs can be trusted in training and testing perception models.

4. Dataset

For our experiments, we were unable to find an open-source camera-based driving dataset that is pixel-wise semantically labeled as well as containing a few long-tail out-of-distribution situations such as sharp cut-ins, diverse lighting, and weather conditions. Thus, we collected and annotated about 8 hours of data by driving on California highways in a course of two months. Although the entire sensor suite comprises multiple front view and top view fisheye cameras, radars, and LiDAR, we only focus on perception modules using the high-resolution center front-facing camera (1920×1208) in this paper. 3959 discrete images were randomly sampled from the driving dataset to be pixel-wise annotated with instance-level semantic segmentation masks. Additionally, 600 short video clips were annotated based on the type of event (like lane change).

For our first experiment described in section 5, we used the segmentation masks to obtain bounding box ground truths for three categories of highway traffic agents - (1) cars, (2) motorbikes, and (3) big vehicles like trucks and buses. By virtue of the long tail distribution, although we have instances of fog, rain, and rare events like accidents in our dataset, the frequency is too small to do any sort of analysis with statistical significance. Therefore, we treat the night time domain as the out-of-distribution mode for which we assume training data is either rare or non-existent. Of the 3959 labeled images chosen for object detection experiments, only 299 of them are night images, and the rest are day images with the visible horizon. The experiments are designed to assess whether we can utilize GAN generated

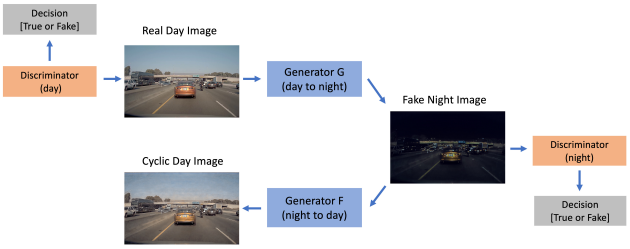

Figure 3. An illustration of CycleGAN architecture.

night images for downstream object detection training and evaluation.

For the second application, section 6, we extract 2 to 3 seconds short video clips from the aforementioned driving dataset where the images are sampled at 30 frames per second. We define sharp cut-in as an abrupt changing of lanes of a vehicle from an adjacent lane towards the front of the ego vehicle. We have many *normal* driving scenarios but only 11 events that can be categorized as sharp cut-ins; we use all of these events in the test set and thus have zero instances of real sharp cut-ins in the training set. Our training dataset consists of 94 normal real video clips, 97 GAN generated sharp cut-in clips, and 110 generated normal clips. Our validation set consists of 11 real sharp cut-ins and 11 normal video clips. It is worth noting that these clips are very diverse in terms of the time of the day, weather, and the length of the actual events.

5. GANs for Object Detection

We used CycleGAN [38] to transfer day images into the night domain. The samples of real day image, real night image, and GAN generated night image are shown in Figure 4. We followed four crucial steps to generate high-resolution imagery:

1. We apply transfer learning technique to fine-tune a pre-trained GAN [23] with 1800 pairs of unlabeled day-night images from our camera's driving dataset. The pre-trained model was trained on an unlabeled day and night pairs from BDD [35], and Mapillary [24] dataset.

2. We trained with random crops of size 360×360 to focus on local features instead of downsizing the whole images.

3. During inference, we directly pass the full resolution day images to the generator fully convolutionally to get night images of the same resolution.

4. After training CycleGAN for 200 epochs with learning rate $1e-6$ and decay factor 0.0002, we did a one-time visual check to pick the best checkpoint.

(a) Real Day Image (b) GAN Night Image (c) Real Night Image

Figure 4. Examples of (a) real day image, (b) corresponding CycleGAN generated night image, and (c) real night image.

We explored the performance of augmenting GAN data in object detection tasks using the recently released YOLOv5 [16] model. We did an ablation study of different combinations of real data and GAN augmented data to study the effects of GAN data in both the training and validation process. Stochastic gradient descent (SGD) was used with the initial learning rate of 0.01 with decay factor $5e-4$ and momentum 0.9. We also set class positive weight for motorbikes to be 4 to prevent the models from being heavily biased towards the majority classes of cars and big vehicles. Batch sizes of 16 were formed for 640×640 input resolution. The models were trained for 200 epochs, and the checkpoints with the best performance on a held-out validation set were used on the test set. Both CycleGAN and YOLOv5 models were implemented in Pytorch [25] and trained on a single Nvidia Titan X GPU.

5.1. Using GANs for Training

For all these experiments, the test set is balanced with 148 real day and 148 real night images. The training set starts with 3511 instances of real day images only and a varying amount of GAN generated night images or real night images are added. The mean Average Precision (mAP), both class-wise and overall, is reported separately on the test day and test night images.

For the first experiment, we assume we have no instances of the out-of-domain mode, i.e. night time, in our training data. Figure 5 shows the advantage obtained by progressively adding GAN generated night data. While the overall performance on the night-time test set improves with an increasing proportion of GAN night data in the training set, the performance on day time test set more or less remains stagnant. To analyze the statistical significance of results, in each experiment, we trained five models from scratch using the same settings of hyper-parameters. The mean and standard deviation of mAP values were shown in the bar plots. Overall mAP improvement by adding an equal amount of GAN night data to real day data was more than double, from 0.205 to 0.445, on night-time test data as compared to the model that was only trained with the real day images.

Table 1. mAP values calculated on real night test data for models trained on same amount of real night and GAN night images

	Training Data	
Classes	real day + real night[a]	real day + GAN night[b]
Car	0.382 ± 0.010	0.344 ± 0.013
Big Vehicle	0.372 ± 0.038	0.308 ± 0.036
Motobike	0.384 ± 0.078	0.377 ± 0.064
All	0.379 ± 0.038	0.343 ± 0.028

Only three significant digits for mAP values are shown in the table.
[a]: 3511 real day images and 150 real night images
[b]: 3511 real day images and 150 GAN night images

An intriguing question that we wanted to investigate is how good of a substitute is GAN as compared to using actual real data from the concerned domain. Table 1 compares the performance on night-time test set between adding 150 real night data versus adding an equal amount (150) of GAN generated night data to the training data. For both cases, the number of real training day images is 3511. The overall mAP gain is in the same range for both cases. However, one of the advantages of using GAN generated data is that we can add multiple versions of nighttime images into the training data and this leads to higher performance gain as seen in Figure 5.

Although GANs showcase performance advantage when used for data augmentation, there have been prior works [15] showing that it may sometimes lead to unintended bias into the dataset and training. GANs also suffer from mode collapse and hence are not able to represent some of the low probability regions of the distribution and also GAN data usually have specific unwanted patterns [21] like asymmetry, missing features, and semi-regular noise. In order to remove any sort of GAN specific bias towards the night domain, we propose a simple debiasing technique by generating GAN data across both in-distribution and out-of-distribution domains. We took the training data with 3511

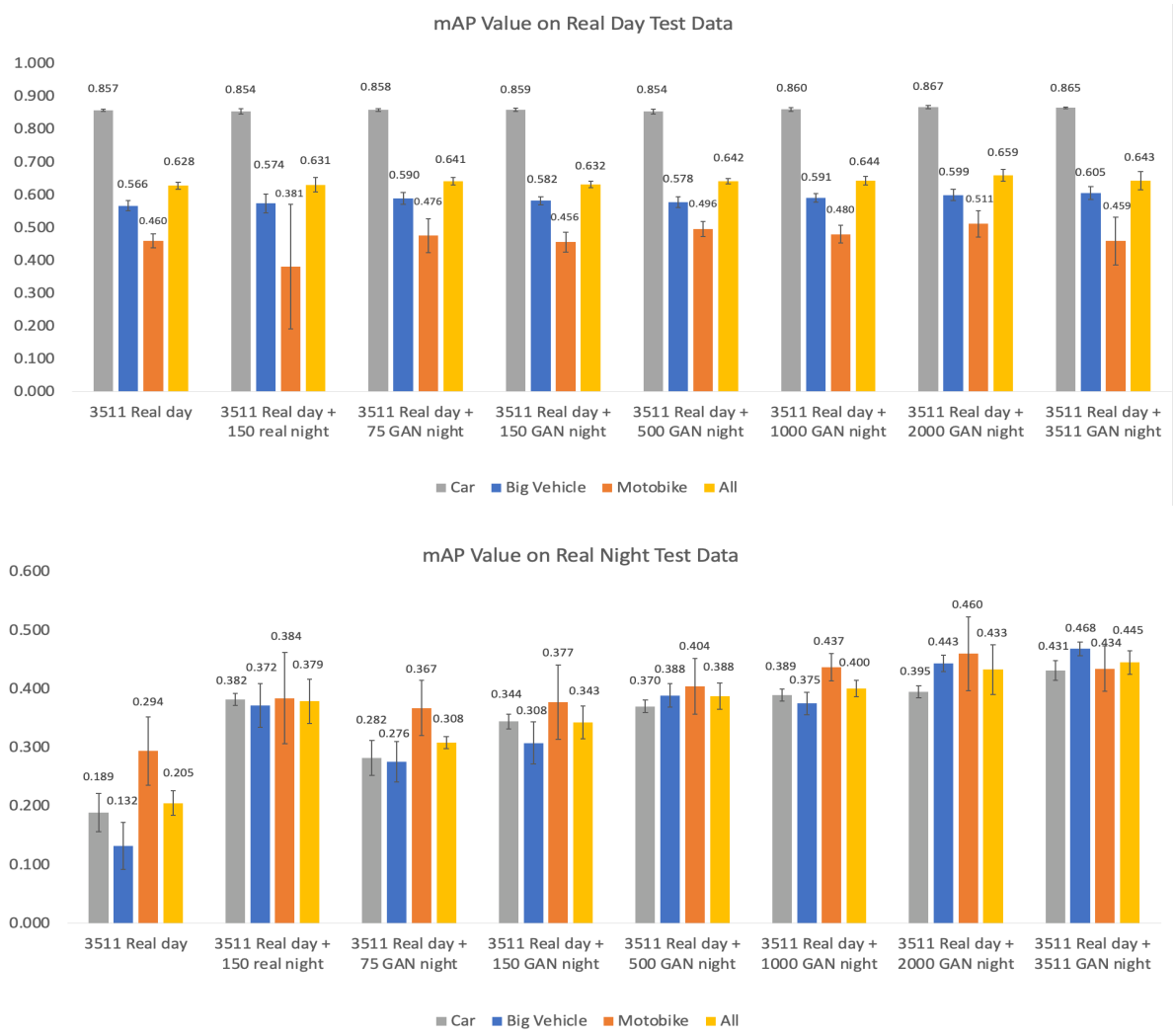

Figure 5. The models are trained with combinations of 3511 real day images and varying amounts of GAN night images and real night images. mAP values are calculated on the real day test set (top) and real night test set (bottom), respectively.

Table 2. Classwise and overall mAP values calculated on both real day test data and real night test data from models trained on different combinations of real data and GAN data to explore bias issue.

	Real Day Test Data				Real Night Test Data			
Training data	Car	Big Vehicle	Motobike	All	Car	Big Vehicle	Motobike	All
Real day + GAN night[a]	0.865	0.605	0.459	0.643	0.431	0.468	0.434	0.445
GAN day + GAN night[b]	0.875	0.600	0.507	0.660	0.445	0.470	0.350	0.422
Real day + GAN day + GAN night[c]	0.877	0.624	0.463	0.655	0.436	0.472	0.445	0.450

All GAN night images are generated from real day images using generator A in CycleGAN. GAN day images are generated by transforming real day images to GAN night images(generator A) to GAN day images(generator B) using CycleGAN.
[a]: 3511 real day images and 3511 GAN night images.
[b]: 3511 GAN day images and 3511 GAN night images.
[c]: 3511 real day images and 3511 GAN night images and 3511 GAN day images.

real day images and 3511 GAN night images as the baseline and compared it with two training sets (a) adding GAN day images to real day and GAN night, and (b) training only with GAN day and GAN night without any real data. As can be seen in Table 2, adding GAN data in the dominant in-distribution mode acts as a debiaser by asking the object detector not to focus on unwanted patterns and thus leads to better performance both on test day and night datasets.

5.2. Using GANs for Testing

A relatively less explored area of research is the safety and reliability of using GANs to evaluate models before deployment. This is similar to how simulators are employed to perform offline closed-loop and open-loop testing for perception and can thus be instrumental in capturing problems before the autonomous system hits the roads. For example, if we can conclude with high confidence that the self-driving software is unable to detect GAN-generated Halloween costumed pedestrians and the same conclusion also holds on real life pedestrians with such costumes, then it would be indeed a life-saving way of doing function validation. For this set of experiments, we either use the 149 real night images in the test set or translate the 149 real day images in the test set to create 149 GAN night images for testing. We use these two test sets to evaluate a YOLOv5 model which was trained using the real day images only. The results in Table 3 demonstrate that although overall mAP values remain in the same range, class-wise average precision varies a lot. We believe that this can be attributed to the difference in object distribution between the real day (and hence GAN night) and real night images in the test data. A similar problem is not that prominent during training since we go through the images iteratively in batches and epochs during the entire training process.

However, a qualitative eye test manifests similar patterns. Figure 6 shows how we can automatically figure out situations that can lead to potentially adverse results without really having to go for real data. The YOLOv5 trained with real day images is not supposed to behave well in the night domain. One such adverse pattern is that a truck either not being properly detected or only its lower half being partially detected as cars in the night domain. This is probably due to the fact that only the lower part of such bigger vehicles are illuminated by the front lights of the ego vehicle. Such disproportionate illuminance might be confusing the object detector to have poor performance in out-of-domain night time situations.

5.3. Using Perception Loss for Data Generation

The experiments so far have treated the data generator and actual downstream task in a decoupled manner. While

Figure 6. Predictions of bounding boxes for big vehicles from YOLOv5 model that was that was trained on real day images only, and tested on real night image (top) and GAN night image (bottom). In both cases, similar error patterns (lower half of truck is detected as cars) are observed.

Table 3. Object statics and mAP values on real night test data and GAN night test data.

	Number of Objects		mAP Values	
Classes	Real night	GAN night	Real night	GAN night
Car	2055	996	0.189 ± 0.033	0.392 ± 0.052
Big Vehicle	227	180	0.132 ± 0.040	0.051 ± 0.010
Motobike	12	21	0.294 ± 0.058	0.119 ± 0.038
All	2294	1197	0.205 ± 0.021	0.188 ± 0.026

the GANs can notably create visually indistinguishable imagery, the real usage of these data samples, of course, depends on the amount of value it adds to the subsequent learning process. In this experiment, we look for possible ways to combine the perception task of object detection in the GAN training process too in the hope of generating better applicable data samples even for out-of-distribution modes. While some previous literature has followed on the lines of adding auxiliary or perceptual losses on both domains, we cannot use the same since we assume that the target domain to be generated is a rare mode, and hence we neither have data or any trained models to work on it.

We start with the initial zero-shot situation where we assume we have no labeled data in the night domain. We

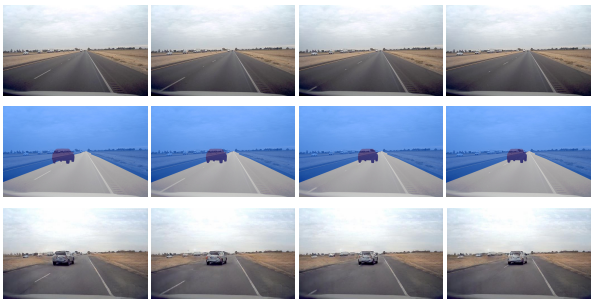

Figure 7. Frames from a real world video clip (1st row); simulated mask (2nd row) where gray, purple, and blue colors indicate road surface, vehicles, and others respectively; and GAN generated frames (3rd row) by manipulating the segmentation masks with the corresponding simulated masks.

only have 3511 real day images labeled with object detection bounding boxes. The pretrained CycleGAN was fine-tuned with 1800 unlabeled pairs of day and night images captured from the ego vehicle's raw driving data. The original CycleGAN comprises of two GAN style adversarial losses and two cycle consistency losses for both directions $day \rightarrow night \rightarrow day$ and $night \rightarrow day \rightarrow night$ each. In order to promote detection in low visibility situations like night time on highways, we rephrase the cycle consistency loss by putting more emphasis on using pixel-wise weights based on the presence and absence of foreground objects like vehicles in the images. As per Figure 3, the new cycle consistency loss for day image X is

$$L_{cyc}^{Day2Night} = \sum_{n} \sum_{i,j} w_{i,j} ||X_{i,j} - F(G(X))_{i,j}||_1, \quad (1)$$

where $w_{i,j} = 1$ if it's a background pixel and 10 for foreground objects of interest like cars, motorbikes, and bigger vehicles. We experimented with various weights and went for this uneven scheme as it promoted the performance of generating better imagery for the less popular class modes. Since the YOLOv5 was trained on real day images only, it is expected to perform well on daytime domain only and not on night images. Thus, the trained YOLOv5 was used to generate pseudo-labels for the daytime 1800 images used for CycleGAN training. And the cycle consistency revision was only applied one way, i.e on $day \rightarrow night \rightarrow day$. Although the cycle consistency loss is only applied on the reconstructed day images, it is intended to force both the generators F and G implicitly to put further attention on the foreground region containing vehicles. Table 4 shows that this led to a marginal increase in the overall performance on the real night test set.

Table 4. mAP values on real night test data reported by original CycleGAN and proposed CycleGAN with weighted consistency loss.

Models	Overall mAP
Original CycleGAN	0.445
Proposed CycleGAN	0.455

6. GANs for Scenario Classification

The application of GANs for modifying objects is relatively unexplored at the video level due to the difficulty of applying motion-related changes in it. Works like Pix2PixHD and Vid2Vid [32] generate color images and videos, respectively, using segmentation maps, but they do not specifically address the generation of rare or corner cases. Automatic video manipulation to create long-tail situations can be crucial in application areas such as autonomous driving. In this section, we present a novel approach to detect a rare scenario, specifically the sharp cut-in of a vehicle by hastily changing lanes into the ego lane in front of the ego vehicle. These can often lead to dangerous and unsafe situations since sharp cut-ins quickly reduce the safety gap between the ego vehicle and are often difficult to predict. We assume a complete zero-shot setting, which means there is no real-world training data available at all for the sharp cut-in scenario.

6.1. Modules

We train DeepLabv3 [4] model for semantic segmentation and Mask-R-CNN [10] for instance identification using the instance-level semantic segmentation annotations that we have for the 3959 images as mentioned in the previous section. We then apply the trained models to obtain segmentation masks for the remaining frames in the training videos.

Modifying the segmentation maps of videos is not a trivial task as careful attention must be paid to the orientation and perspective of cutting in vehicles. We thus leverage Unity3D game engine to manipulate segmentation maps. We use the calibration parameters of the real camera to mount a similar virtual camera in the simulation. Our dataset also contains standard maps and data from the vehicle bus like GPS location, acceleration, yaw, etc. We use this information to design roads and ego vehicle motion in the simulator. Assets of different types of vehicles are obtained from Unity Store and used to generate a sharp cut in behavior. Then, only the mask of the cutting-in vehicle, shown in purple in Figure 7, is overlaid onto the corresponding segmentation labels of real frames. We then use Pix2PixHD, which was also trained using the same 3959

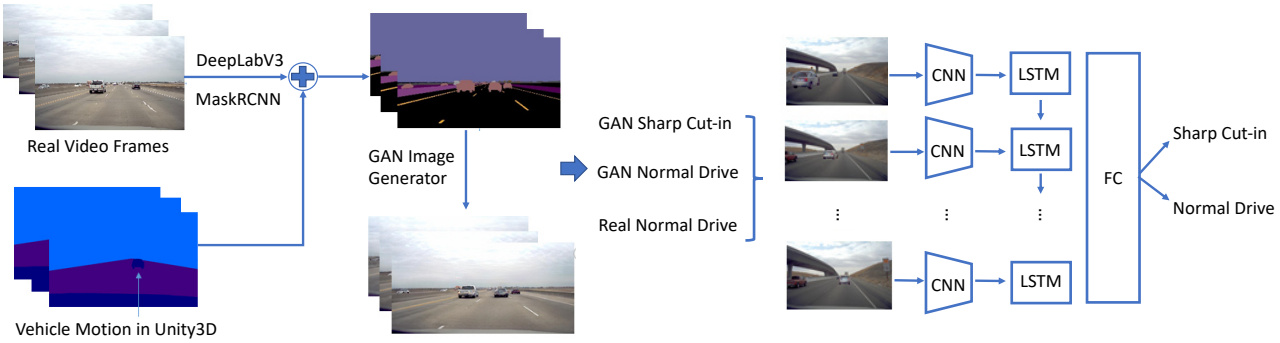

Figure 8. The architecture of sharp cut-in data generator and detector: Given a limited number of frames annotated, we train DeeplabV3 and MaskRCNN to obtain annotation for all frames and then we use Unity3D to insert moving vehicles into the masks. Afterward using Pix2PixHD trained on our data, we generate new clips for the intended scenarios. Lastly, we train a CNN+LSTM classifier using the real and generated data to classify the sharp cut-in vs normal driving on real videos.

image and segmentation mask pairs, to generate the realistic video frames for the intended cut-in event. Following the conclusion obtained from the GAN bias experiments in the previous section, we also end up generating both sharp cut-in and normal video clips using GANs, as depicted in Figure 8.

To perform video action classification while maintaining a low computational cost, we use the popular CNN+ LSTM model that applies the LSTM on the features obtained from applying a convolutional network on a sequence of video frames. We used the ImageNET [5] pretrained ResNet-152 weights as our CNN base to achieve visual features form video frames. We obtain a 512 dimensional feature vector for each frame and feed these features to a recurrent model that comprises 2 LSTM layers followed by a fully connected layer for binary classification. Although our video clips are 60 or 90 frames, our input sequence to the model is 30 frames. Thus we can get multiple data samples during batch training by randomly selecting the start and the end of videos to pick a 30 frame sequence.

6.2. Results for Scenario Classification

For each test video clip, the inference is performed by applying the model on its overlapping subsequences, and the decision is made by temporal averaging. We apply two different approaches to generate the sharp cut-in GAN data in the training set. In the first one, we generate the whole image sequence using pix2pixHD, and crop the pixels belonging to the cut-in vehicle only and paste that onto the corresponding frames belonging to the real normal data. In the second approach, we use the whole GAN generated frames as it is to create our training data. As reported in Table 5, using the whole generated images leads to better results. We believe this is because the homogeneity of the images in whole generated versions directs the network to correctly attend to motion changes rather than unnecessarily focusing on color or gradient change in cut-and-paste data.

Table 5. Results on zero shot detection of GAN generated cut ins using CNN and LSTM.

Method	Accuracy	Precision	Recall
Masks Pasted	0.56	0.6	0.35
Whole Generated	0.76	0.88	0.66

7. Conclusion

In this paper, we empirically study whether one can rely on GANs to generate data for training and evaluating vision-based perception modules. We solely focus on simulating data for out-of-training-distribution or long tail situations only. With our experiments on domain translation, we demonstrated how including GAN generated night data can substantially increase the performance of downstream object detection models on actual night time test sets. We also uncovered and suggested mitigation techniques for the bias introduced during GAN included training. Although results were inconclusive on whether one can quantitatively validate perception models by testing using GAN data only, we showed interesting detection patterns that also occur when tested with real data. We also proposed a novel loss function that helped CycleGAN to put focus on targeting the improvement of downstream object detection module. We also showed how GANs can be used to simulate rare traffic scenarios like sharp cut-ins and how action classification neural networks can then be trained, in the absence of any real world sharp cut-in training data, to detect such events in the real world.

References

[1] V. F. Arruda, T. M. Paixão, R. F. Berriel, A. F. De Souza, C. Badue, N. Sebe, and T. Oliveira-Santos. Cross-domain car detection using unsupervised image-to-image translation: From day to night. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[2] A. Borji. Pros and cons of gan evaluation measures, 2018.

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.

[7] F. H. K. dos Santos Tanaka and C. Aranha. Data augmentation using gans. *Proceedings of Machine Learning Research XXX*, 1:16, 2019.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[12] S. Hong, D. Yang, J. Choi, and H. Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7986–7994, 2018.

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

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[15] N. Jain, L. Manikonda, A. O. Hernandez, S. Sengupta, and S. Kambhampati. Imagining an engineer: On gan-based data augmentation perpetuating biases. *arXiv preprint arXiv:1811.03751*, 2018.

[16] G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, A. Hogan, lorenzomammana, tkianai, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, Hatovix, J. Poznanski, L. Yu, changyu98, P. Rai, R. Ferriday, T. Sullivan, W. Xinyu, YuriRibeiro, E. R. Claramunt, hopesala, pritul dave, and yzchen. ultralytics/yolov5: v3.0, Aug. 2020.

[17] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, pages 3927–3936, 2019.

[18] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz. Context-aware synthesis and placement of object instances. In *Advances in neural information processing systems*, pages 10393–10403, 2018.

[19] H. Machiraju and V. N. Balasubramanian. A little fog for a large turn. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2902–2911, 2020.

[20] T. Matsui and M. Ikehara. Gan-based rain noise removal from single-image considering rain composite models. *IEEE Access*, 8:40892–40900, 2020.

[21] K. McDonald. *How to recognize fake AI-generated images*, Dec. 5 2018 (accessed: 2020-09-29). https://medium.com/@kcimc/how-to-recognize-fake-ai-generated-images-4d1f6f9a2842.

[22] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[23] S. Nag, S. Adak, and S. Das. What's there in the dark. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2996–3000. IEEE, 2019.

[24] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4990–4999, 2017.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[26] P. Prabhanjan Brahma and A. Othon. Subset replay based continual learning for scalable improvement of autonomous systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1066–1074, 2018.

[27] S. Ravuri and O. Vinyals. Seeing is not necessarily believing: Limitations of biggans for data augmentation. 2019.

[28] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[29] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers. Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks. *Scientific reports*, 9(1):1–9, 2019.

[30] K. Shmelkov, C. Schmid, and K. Alahari. How good is my gan? In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229, 2018.

[31] M. Uřičář, P. Křížek, D. Hurych, I. Sobh, S. Yogamani, and P. Denny. Yes, we gan: Applying adversarial techniques for autonomous driving. *Electronic Imaging*, 2019(15):48–1, 2019.

[32] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems*, pages 1144–1156, 2018.

[33] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.

[34] Q. Xu, G. Huang, Y. Yuan, C. Guo, Y. Sun, F. Wu, and K. Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018.

[35] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020.

[36] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[37] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid. Deeproad: Gan-based metamorphic autonomous driving system testing. *arXiv preprint arXiv:1802.02295*, 2018.

[38] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.