Generation of Natural Traffic Sign Images Using Domain Translation with Cycle-Consistent Generative Adversarial Networks

Dominic Spata, Daniela Horn, Sebastian Houben

***Abstract*— Video-based traffic sign recognition poses a highly challenging problem due to the significant number of possible classes and large variances of recording conditions in natural environments. Gathering an appropriate amount of data to solve this task with machine learning techniques remains an overall issue.**

In this study, we assess the suitability of automatically generated traffic sign images for training corresponding image classifiers. To this end, we adapt the recently proposed cycle-consistent generative adversarial networks in order to transfer automatically rendered prototypical traffic sign images for which we control type, pose, and—to a degree—background into their true-to-life counterparts. We test the proposed system by extensive experiments on the German Traffic Sign Recognition Benchmark dataset [1] and learn that both a HOG-feature-based SVM classifier and a state-of-the-art CNN exhibit reasonable performance when solely trained on artificial data. Consequently, it is well suited as data augmentation method and allows for covering uncommon cases and classes.

I. Introduction

The detection and recognition of road signs in natural traffic scenes is a crucial ability for both advanced driver assistance systems and autonomous vehicles. Although video-based recognition is concerned with rigid, clearly defined objects designed for visual noticeability, in particular

- the high number of possible classes,
- their unbalanced distribution,
- the variance in diverting background, as well as
- numerous recording artefacts due to natural lighting and camera motion

pose unsolved challenges to be handled in today's research.

To complicate things further, it appears that most nations do not only own a number of unique traffic signs, but also at times very different versions of internationally used signs, including varying colors and even shapes. Consequently, the number of possible classes within given traffic sign categories, such as warning signs, is vast and the data acquisition, in particular when tending to appropriate variances regarding recording conditions and intra-class changes, is, hence, extensive.

However, over the past decades an increasing number of countries around the globe have agreed to align the visual appearance of traffic signs according to the Vienna Convention. The treaty encompasses, *inter alia*, a number of traffic sign designs for categories like danger, prohibitive, or mandatory signs. This uniformity enables us to straightforwardly render artificial images of classes from predefined categories by choosing the outer appearance, inserting the pictogram and transforming the resulting prototype with a random homography that mimics natural perspective and slight rotation.

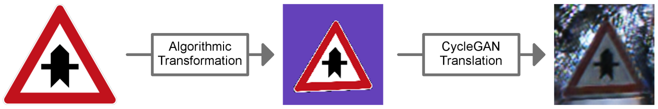

Fig. 1. A schematic of the generative process. Simple traffic sign diagrams are algorithmically transformed into prototype images, which are then translated into the final generated photograph by the CycleGAN's mapping network.

Recent advances in deep learning have produced a number of techniques to automatically generate life-like images. For these purposes the defining characteristics of the respective image domains are learned from example data. In particular, so called cycle-consistent generative adversarial networks allow for a style transfer between two image domains. We adapt this technique to translate the rendered images to true-to-life ones while preserving category, pose, pictogram, and, hence, the class identity of the traffic sign to generate.

We detail the predominant methods in traffic sign recognition and real-life image generation in Sec. II and describe prototype rendering and our adaptations to cycle-consistent generative adversarial networks in Sec. III. Sections IV and V focus on verifying the aptitude of the artificially generated training examples for well-established traffic sign classifiers. Sec. VI briefly summarizes our findings.

II. Related Work

Lifelike image generation has gained a lot of interest and inspired an entire body of literature initiated by the work of Goodfellow who introduced Generative Adversarial Networks [2] (GANs) which, when combined with the then established deep convolutional architectures [3], are capable of creating already highly convincing images. For these purposes, GANs relinquish modeling the underlying generative probability density but create example images that share dominant features with the training examples from which they emerge.

Image generation in GANs is based on white noise fed into the network. Gaining some control over the result has since been a vivid topic of research [4]. Class-consistency [5] is a predominant procedure for both creating only instances from one specific image category and stabilizing the overall training process. We, instead, make use of cycle-consistent GANs [6] (or CycleGANs for short), which enforce visual

The authors are with the Institute for Neural Computation, Ruhr University of Bochum, Universitätsstraße 150, 44801 Bochum sebastian.houben@ini.rub.de

and geometric similarity of an input image and its respective generated output. Thus, CycleGANs can be likened to other methods of style transfer such as [7], [8], [9], [10].

Both video-based traffic sign recognition and detection have been studied extensively in the past years [11], [12], [13], [14], [15] – not least because of publicly available large-scale datasets like the Swedish Traffic Sign Dataset [16], the Belgian Traffic Sign Dataset [17], the LISA dataset [15] presenting American traffic signs, and the Russian Traffic Sign Dataset [18]. While these examples clearly cover the variance in visual appearance and recording conditions as well as intra-class variance, the number of traffic sign classes (currently led by the Russian Traffic Sign Dataset with 157 classes) only represents a fraction of all traffic sign categories presently deployed in streets worldwide.

The need for the augmentation of these datasets has therefore been studied, on the one hand in order to also handle cases not present in the dataset and, on the other hand, as a well-established regularization method, in particular when training deep convolutional neural networks. Moiseev et al. [19] present an extensive but purely algorithmic augmentation setup in which they vary the pose of traffic sign pictograms and add motion blur, image noise, and background from real images. Their approach shows on par results when training convolutional neural networks on synthetic data only, but falls short for feature-based classifiers.

A recent paper by Luo et al. [20] deploy a GAN that transforms a pictogram into a lifelike traffic sign image in front of a given background. Likewise, they show results on a Spatial Transformer Network and achieve state-of-the-art results, even with purely synthetic training data. However, training the network requires a suited regularization term in order to retain the class identity during image generation.

Our system uses a CycleGAN straightforwardly controlling the generation process. Furthermore, the background is learned and created by the same GAN. For evaluation we use a support vector machine (SVM) trained on Histogram-of-Oriented-Gradients (HOG) features which, despite its simplicity, showed good performance during the German Traffic Sign Recognition Benchmark. We consider this the most objective comparison: Using a fixed feature extraction step, we can attribute the performance to the quality of the generated data only and not to the ability of a deep network to generalize and transfer possibly missing features well. In order to compare against the state-of-the-art, we also perform parallel experiments with a deep convolutional neural network (CNN).

III. Method

In our approach, the data generation process is addressed as an image-to-image translation problem, allowing for a high degree of intermediate control over properties of the generated images. Image-to-image translation systems characteristically learn mappings that perform translation between two domains of images. In the case at hand, one domain consists of real-life image data, the other comprises a form of image prototype that is simplistic enough as to allow for efficient sampling. For the latter, natural images are then generated by a learned domain mapping of the previously sampled prototype images, as depicted in Fig. 1.

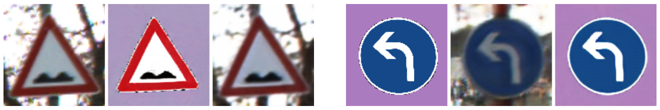

Fig. 2. Cyclic translations between image domains $\mathcal{X}$ and $\mathcal{Y}$. Each image triplet shows the original source image, its translation, and the cyclic reconstruction. *Left:* Translation from photos to icons. *Right:* Translation from icons to photos.

This approach offers three distinct advantages over more traditional data generation. Firstly, when the prototype domain is wisely chosen, its algorithmic sampling procedure already provides much of the salient information required for a natural image. The generative model then merely needs to close the remaining textural and stylistic gap between the prototype image and its natural counterpart, which is a much simpler task than generation from scratch. This is especially effective for a dataset of multiple classes with standardized features, such as the German Traffic Sign Recognition Benchmark (GTSRB) dataset.

Secondly, since the appearance of the generated image is tightly coupled to the appearance and features of the corresponding prototype image, it is possible to exert direct control over the data generation via appropriate adjustments to the sampling process for the prototype images. In particular, class distribution, scale and perspective of the traffic sign within the image, and even certain background details are easily customizable.

Lastly, the following approach allows image generation for types of traffic signs which are not presented to the generative model during training, provided that prototype images can be created for them. This may be used to supplement natural images of traffic signs for which no real photographs are available. The following subsections outline some technical specifics of our approach.

A. Cycle-Consistent Generative Adversarial Networks

CycleGANs naturally excel at textural and stylistic transformation. However, the fact that they can also be trained using unpaired datasets enforces their suitability for the given task. This characteristic is critical in our application in order to simplify the prototype sampling process, as it is sufficient to create the prototype image training set independently from the real image training set.

In CycleGANs, the mappings $G : \mathcal{X} \rightarrow \mathcal{Y}$ and $F : \mathcal{Y} \rightarrow \mathcal{X}$ for image domains $\mathcal{X}, \mathcal{Y}$ are trained via optimization of a variant of the adversarial loss known from generative adversarial networks [2]. G and F each enter a minimax game with a discriminator that aims to distinguish real images from those created by the mapping. Optimization steps alternate their modifications of mappings and discriminators, such that the discriminators' performance is either decreased or increased to eventually produce plausible image results w.r.t. their respective domain.

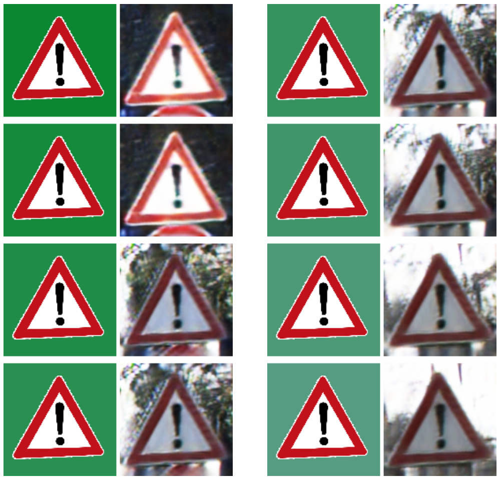

Fig. 3. A demonstration of the learned association that maps smoothly from background colors in the image prototypes to background and illumination styles in the generated images. Each image pair shows an image prototype on the left and the corresponding generated image on the right.

In order to enforce the resulting mappings to be actual translators rather than being arbitrary, CycleGANs are additionally constrained to be *cycle-consistent*, which requires their mappings to be the inverse of one another. This is formulated as an additional loss based on the L1-norm of the difference between images from the training sets and the result of running them through the chain of the two mappings. Fig. 2 shows examples of cyclic translations between the two domains in both directions.

B. Prototype Domain

The image prototypes used in our approach have been derived from standardized traffic sign diagrams[1] by modifying them in two significant ways. Firstly, we apply randomized perspective transforms that supply the geometric information required in a natural image. The CycleGAN is designed primarily for stylistic and textural translations and therefore cannot effectively contribute such information itself. Initial tests showed that prototype images with blank background tend to produce background which is highly dependent on the traffic sign class and hardly varies for similar traffic signs. We therefore, secondly, replaced the transparent background with a random homogeneous color enriching the variations oft he simplistic image domain $\mathcal{Y}$. In this manner we remove the focus from the only source of variation present – the traffic sign itself – to seed the generation of varied scenic details. This is similar to conditioning the mapping on an additional three-dimensional latent vector with the advantage that the information is present locally where it is used in the image. The model can learn an association between these colors and certain realistic background styles. Fig. 3 shows the smooth transitions of background colors in the prototype images and their respectively changing background styles in the generated images.

IV. Experiments

Our objective is to assess the ability of generated traffic sign images to substitute for or supplement natural images in the context of multi-class classification tasks. We therefore conduct a range of experiments designed to determine the influence of our generated data on classification accuracy when used in the training set of a traffic sign classifier. The classifier model we use is a simple multi-class SVM as described in [21] trained on the HOG features [22] of traffic sign images.

In order to establish a frame of reference for the classification accuracy values we obtain, each experiment produces and compares three SVM classifier models, which differ in their data input. SVM_{Base} is trained purely on real data and provides the baseline accuracy for each experiment. SVM_{Gen} is trained on generated images as resulting by our CycleGAN. $\text{SVM}_{\text{Proto}}$ has prototype images as training input. As these images are already high in information, we mean to distinguish the individual influences of the prototype sampling and CycleGAN translation on classification accuracy.

We mirror these experiments with a deep CNN, analogously naming the models CNN_{Base}, CNN_{Gen}, and $\text{CNN}_{\text{Proto}}$. For training we use batch normalization and dropout with a rate of 0.5. Details on the used architecture can be found in Table II.

The GTSRB dataset is provided with a predetermined split into a training and a test set. We further subdivide the training set into two halves, using one half for training the CycleGAN model and the other for training the aforementioned SVM_{Base}. This assures maximum independence between the generated data and the real data used to obtain the accuracy baseline. Likewise, the accuracy values reported in the next section are calculated on one half of the GTSRB test set, as the other half was used as a validation set to fine tune our generative process.

Under this general protocol, we further distinguish two types of experiments relative to the exact composition of the training sets. Their nature and purpose is outlined in the following.

A. Training Scenarios

Two types of experiments were conducted in order to validate our approach. The first one is intended to evaluate the overall quality of the generated images. For this purpose, we test the two classifiers SVM_{Gen} and $\text{SVM}_{\text{Proto}}$ along the previously described baseline classifier SVM_{Base}, as well as CNN_{Gen} and $\text{CNN}_{\text{Proto}}$ against CNN_{Base}, respectively.

In the second experiment, we remove examples for one traffic sign class from the original CycleGAN training set and use the resulting generative model to generate images for that traffic sign class. We train two additional classifier models on similar training sets of real-world images in which all training samples for the previously selected traffic sign class are completely replaced by generated images ($\text{SVM}_{\text{GenClass}}$, $\text{CNN}_{\text{GenClass}}$) and prototype images ($\text{SVM}_{\text{ProtoClass}}$, $\text{CNN}_{\text{ProtoClass}}$), respectively. Finally, their performances are compared to SVM_{Base} and CNN_{Base}.

[1] Available in the public domain, for example from: https://commons.wikimedia.org/wiki/Road_signs_of_Germany

Category	**Characterization**	**Examples**
Warning Signs	upright triangular shape, red border, white background, black content	
Restriction Signs	circular shape, red border, white background, mostly black content	
Derestriction Signs	circular shape, white background, diagonal bars, gray content	
Direction Signs	circular shape, blue background, white arrows	
Miscellaneous Signs	no common features	

TABLE I

CATEGORIZATION OF TRAFFIC SIGNS GIVEN IN THE GTSRB DATASET. EACH OF THE 43 TRAFFIC SIGN CLASSES WAS SORTED INTO ONE OF THE FIVE CATEGORIES BASED ON THEIR SHARED VISUAL FEATURES.

Layer Type	**Filters**	**Size**	
Convolution	32	3×3	
Convolution	32	1×1	
Convolution	32	1×1	
Strided MaxPooling		2×2	
BatchNorm			
Convolution	64	3×3	4x
Convolution	64	1×1	
Convolution	64	1×1	
BatchNorm			
Fully Connected		256	
Fully Connected		128	
Fully Connected		43	

TABLE II

ARCHITECTURE OF THE CNN CLASSIFICATION NETWORK. EACH LAYER IS FOLLOWED BY A RELU ACTIVATION FUNCTION, THE LAST LAYER BY A SOFTMAX ACTIVATION.

B. Dataset Preparation

The GTSRB dataset consists of images with strongly varying sizes and aspect ratios, which complicates the use of mini-batches during the training of the generative model. The CycleGAN training set is therefore preprocessed to yield images of size 128×128. Firstly, we discard all images below the size of 64×64. Afterwards, preserving aspect ratio, we scale all images such that the smaller spatial dimension has a size of 128 pixels and then centrally crop the larger spatial dimension down to 128 pixels. We found that the CycleGAN is sensitive to class distribution and tends to introduce artefacts into generated images of underrepresented classes. Hence we further resample the images of the dataset to balance the class distribution. This creates a dataset of 12,212 examples.

Note that this preliminary selection of images skews the distribution of geometric information, as smaller images tend to possess a larger border around the traffic sign. Both the CycleGAN and both classifiers are sensitive to the distribution of geometric information, and we must therefore account for this circumstance during the creation of the prototype images. We choose the parameters of the randomized perspective transforms differently for the prototype images used during CycleGAN training and for those used to generate images during our experiments, such that the distribution roughly matches that of the preprocessed and the original GTSRB data, respectively.

In the course of the experiments we will sometimes refer to traffic sign "categories", which organize traffic sign classes into groups based on shared visual features. The 43 classes of the GTSRB dataset were divided into five categories, as shown in Table I.

C. Implementation Details

We adopt the implementation details in [6], using fully convolutional neural networks for all learned functions of the CycleGAN. All layers consist of a variant of convolution, instance normalization [23], and ReLU activation. The discriminator networks use the leaky ReLU variant with a slope of 0.2 and produce an output matrix of patchwise classifications, rather than a single classification for the entire image. We raise the power of the mapping networks compared to the original implementation by increasing the number of residual blocks and doubling the number of filters in all layers in order to account for the highly multimodal nature of our multi-class dataset. The resulting mapping and discriminator architectures are displayed in Tables III and IV. Note that for our experiments we require only the mapping that translates prototype images into their real-to-life counterparts, while the inverse mapping is kept purely for regularization purposes during training.

The networks are trained using an L2-variant of the adversarial loss [25] and an Adam optimizer [26] with parameters $\beta_1 = 0.5, \beta_2 = 0.999$. The learning rate is set to 0.0002 for the first half of epochs and is linearly decayed to zero over the second half. We train the CycleGAN for a total of 24 epochs. The mini-batches for training the discriminator networks are partly sampled from a buffer of generated images to combat model oscillation [27]. Contrary to the original implementation, we use mini-batches of size five. We also adopt a slightly modified version of the data augmentation scheme used in the original implementation. Instead of upsampling the images to the static size 143×143 and randomly recropping them, we choose the upsample size uniformly at random from the range $[128, 143]$, so as to reduce the impact on the distribution of geometric

Layer Type	**Filters**	**Kernel Size**
Convolution	64	7 × 7
Strided Convolution	128	3 × 3
Strided Convolution	256	3 × 3
9 × Residual Block	256	3 × 3
Fractionally Strided Convolution	128	3 × 3
Fractionally Strided Convolution	64	3 × 3
Convolution	3	7 × 7

TABLE III

ARCHITECTURE OF THE MAPPING NETWORKS. CONVOLUTION LAYERS CONSIST OF CONVOLUTION, INSTANCE NORMALIZATION, AND RELU ACTIVATION. RESIDUAL BLOCKS FOLLOW THE DESIGN RECOMMENDED BY [24]. THE LAST LAYER USES NO NORMALIZATION STEP AND tanh INSTEAD OF RELU AS ACTIVATION FUNCTION.

Layer Type	**Filters**	**Kernel Size**
Strided Convolution	64	4 × 4
Strided Convolution	128	4 × 4
Strided Convolution	256	4 × 4
Convolution	512	4 × 4
Convolution	1	4 × 4

TABLE IV

ARCHITECTURE OF THE DISCRIMINATOR NETWORKS. CONVOLUTION LAYERS CONSIST OF CONVOLUTION, INSTANCE NORMALIZATION, AND LEAKY RELU ACTIVATION WITH A SLOPE OF 0.2. THE FIRST LAYER DOES NOT USE A NORMALIZATION STEP AND THE LAST LAYER DOES NOT USE AN ACTIVATION FUNCTION.

information. We further forgo random horizontal flips, as they may produce images depicting nonexistent signs.

The method's inherent cycle-consistency stabilizes the training process, such that it hardly produces unusable CycleGANs. The resulting models show an overall fairly small variance. Losses alternately increase and decrease by design and, thus, do not converge.

Our CycleGAN and CNN implementations are based on Tensorflow[2]. Training our CycleGANs took 10 to 12 hours each on a GTX 1070. Furthermore, we use the SVM implementation contained in scikit-learn[3] for all classifiers. Our code is available online[4].

V. RESULTS

The results of our experiments demonstrate that the generated images are reasonably, though not perfectly, realistic. Fig. 4 depicts examples of generated traffic signs in direct comparison to real-world samples. SVM$_{\text{Gen}}$ und SVM$_{\text{GenClass}}$ consistently show increased classification errors compared to the real-data baseline. However, they significantly outperform SVM$_{\text{Proto}}$ and SVM$_{\text{ProtoClass}}$, respectively. These results are in line with those of the CNN-based classifiers.

The results of the individual experiment types are discussed in greater detail below.

	Classification Accuracy (%)		
Category	**SVM$_{\text{Base}}$**	**SVM$_{\text{Gen}}$**	**SVM$_{\text{Proto}}$**
Warning	78.08	75.76 (−2.32)	45.36
Restriction	87.40	72.21 (−15.19)	47.40
Derestriction	80.33	86.34 (+6.01)	84.70
Direction	94.37	85.86 (−8.51)	61.95
Miscellaneous	98.65	96.62 (−2.03)	84.82
Total	87.97	79.27 (−8.70)	56.17

TABLE V

PER-CATEGORY CLASSIFICATION ACCURACIES OF REAL, GENERATED, AND PROTOTYPE TRAINING INPUT. NUMBERS IN PARENTHESES STATE DIFFERENCES TO SVM$_{\text{BASE}}$.

	Classification Accuracy (%)		
Category	**CNN$_{\text{Base}}$**	**CNN$_{\text{Gen}}$**	**CNN$_{\text{Proto}}$**
Warning	92.38	88.24 (−4.14)	16.33
Restriction	97.09	85.65 (−11.44)	6.00
Derestriction	95.08	88.52 (−6.56)	0.00
Direction	94.71	85.63 (−9.08)	20.34
Miscellaneous	95.55	93.42 (−2.13)	25.44
Total	95.42	87.57 (−7.85)	13.24

TABLE VI

PER-CATEGORY CLASSIFICATION ACCURACIES OF REAL, GENERATED, AND PROTOTYPE TRAINING INPUT. NUMBERS IN PARENTHESES STATE DIFFERENCES TO CNN$_{\text{BASE}}$.

Each classifier was trained with roughly $19,300$ samples of real-world, generated, and prototype images, respectively (cf. Sec. V-A), or a mixture of real-world and synthetic image data (see Sec. V-B). Initial tests on bigger sized synthetic datasets showed no significant improvement on accuracy, while being incommensurately time-consuming. Extensive experiments are therefore postponed to future research.

A. Training on Fully Generated Data

Table V displays a comparison of the classification accuracies of SVM$_{\text{Base}}$, SVM$_{\text{Gen}}$, and SVM$_{\text{Proto}}$. The use of generated data lowers the classification accuracy by almost nine percentage points, revealing imperfections in our generation process. SVM$_{\text{Proto}}$, meanwhile, appears insufficient to produce a useful traffic sign classifier.

An investigation into the per-class accuracies shows that SVM$_{\text{Gen}}$ performs better than SVM$_{\text{Base}}$ for certain traffic sign types. In rare cases, this is even true for SVM$_{\text{Proto}}$. The traffic sign classes that appear to benefit most from the alternate sources of data are those underrepresented in the GTSRB dataset. Experiments attempting to create a classifier that improves over the baseline in terms of overall classification accuracy yielded only modest success.

Similarly, training a CNN classifier on an entirely generated dataset leads to a comparable drop in performance (cf. Table VI). Using only prototype images instead inhibits proper feature extraction during training and results in extremely poor performance.

[2] See: `https://www.tensorflow.org/`

[3] See: `https://scikit-learn.org`

[4] See: `https://github.com/Spataner/trafficsign-cyclegan`

Class	Classification Accuracy (%) SVM_{Base}	$\text{SVM}_{\text{GenClass}}$		$\text{SVM}_{\text{ProtoClass}}$
Experiment 1: replacing class “no entry (trucks)”				
No entry (trucks)	97.18	88.73	(−8.45)	0.00
Speed limit 100	74.44	73.99	(−0.45)	74.44
Roundabout	70.46	72.73	(+2.27)	70.46
Total	87.97	87.87	(−0.10)	86.86
Experiment 2: replacing class “slippery road”				
Slippery road	67.11	60.53	(−6.58)	0.00
(Non-substituted classes exhibited no performance change for $\text{SVM}_{\text{GenClass}}$)				
Total	87.97	87.89	(−0.08)	87.17
Experiment 3: replacing class “pass right”				
Pass right	95.87	78.47	(−17.40)	14.16
Stop	92.36	93.06	(+0.70)	93.06
Forward or right	96.92	98.46	(+1.54)	98.46
Total	87.97	87.06	(−0.91)	83.61

TABLE VII

Per-class classification accuracies for different substitute classes (highlighted). Numbers in parentheses state differences to SVM_{Base}. Most classes do not exhibit any change in classification accuracy when the chosen substitute class is replaced with generated images and are, hence, not shown.

Class	Classification Accuracy (%) CNN_{Base}	$\text{CNN}_{\text{GenClass}}$		$\text{CNN}_{\text{ProtoClass}}$
Experiment 1: replacing class “no entry (trucks)”				
No Entry (Trucks)	100.0	100.0	(+0.00)	15.49
Speed Limit 20	51.52	100.0	(+48.48)	96.97
Left Hand Curve	75.86	100.0	(+24.14)	100.0
Pass Left	78.18	94.55	(+16.37)	96.36
Danger	90.06	71.93	(−18.13)	80.70
Caution Snow	91.43	70.00	(−21.43)	74.29
Total	95.42	94.41	(−1.01)	91.12
Experiment 2: replacing class “slippery road”				
Slippery Road	98.68	98.68	(+0.00)	0.00
Speed Limit 20	51.52	93.94	(+42.42)	78.79
Left Hand Curve	75.86	100.0	(+24.14)	96.55
Pedestrian Crossing	53.12	75.00	(+21.88)	46.88
Pass Left	78.18	98.18	(+20.00)	100.0
Caution Snow	91.43	44.29	(−47.14)	75.71
Total	95.42	95.15	(−0.27)	92.49
Experiment 3: replacing class “pass right”				
Pass Right	93.51	81.12	(−12.39)	42.18
Speed Limit 20	51.52	96.97	(+45.45)	100.0
Left Hand Curve	75.86	100.0	(+24.14)	93.10
Caution Snow	91.43	65.71	(−25.72)	70.00
Pass Left	78.18	27.27	(−50.91)	69.09
Derestrict Overtaking	100.0	27.27	(−72.73)	100.0
Total	95.42	93.65	(−1.77)	89.50

TABLE VIII

Per-class classification accuracies for different substitute classes (highlighted). Numbers in parentheses state differences to CNN_{Base}. Apart from the substitute class itself, the 5 classes with most deviating accuracies w.r.t. CNN_{Base} are shown.

B. Training on Partially Generated Data

Several experiments have been conducted in which one traffic sign class was replaced by either generated or prototype images. We show the results for one example traffic sign class each from the categories of warning signs, restriction signs, and direction signs. Table VII displays comparisons between SVM_{Base}, $\text{SVM}_{\text{GenClass}}$, and $\text{SVM}_{\text{ProtoClass}}$ when examples for traffic sign classes “no entry (truck)”, “slippery road”, and “pass right” are substituted, respectively.

Performance on all untouched traffic classes is largely unaffected by the example substitution conducted for the chosen class. Classification accuracy for selected classes mirror the findings of the previous experiment in that use of generated data decreases performance, while use of prototype data collapses it dramatically. Furthermore, the corresponding per-class accuracy of the classifier from the previous experiment correlates weakly with the accuracy on the substituted class in this configuration.

When training CNN classifiers, the story is different. While the replaced sign class is usually recognized with comparable performance, other, often unrelated, classes partly exhibit strong deviations in classification performance (cf. Table VIII). Obviously, the feature extraction during several training sessions yields different filters, but at the time of writing it is unclear whether this can be attributed to the generated data that may obstruct the training or whether the net has converged to a different local optimum. We will therefore defer this examination to future research.

It is important to note that this experiment is only successful for traffic sign types that belong to a broader traffic sign category and thus share visual features with other classes. If the traffic sign class whose examples are removed from the CycleGAN training set is visually distinct to all remaining traffic signs, then the generative model is unable to create convincing images for that class.

VI. Conclusion

We have presented a flexible system for the generation of traffic sign images in which the pose and, to a degree, the background can be controlled by the user. This facilitates data acquisition substantially. In a fair comparison, however, it has become apparent that using real-world data results in better and more stable classifiers. We advocate to use the method as a data augmentation technique and, in particular, for cases and classes in which no real data is available. In future work our main focus will therefore be to extend these cases by enriching the generation pipeline with further characteristics and recording artefacts like motion blur, dirt, damages and overexposure.

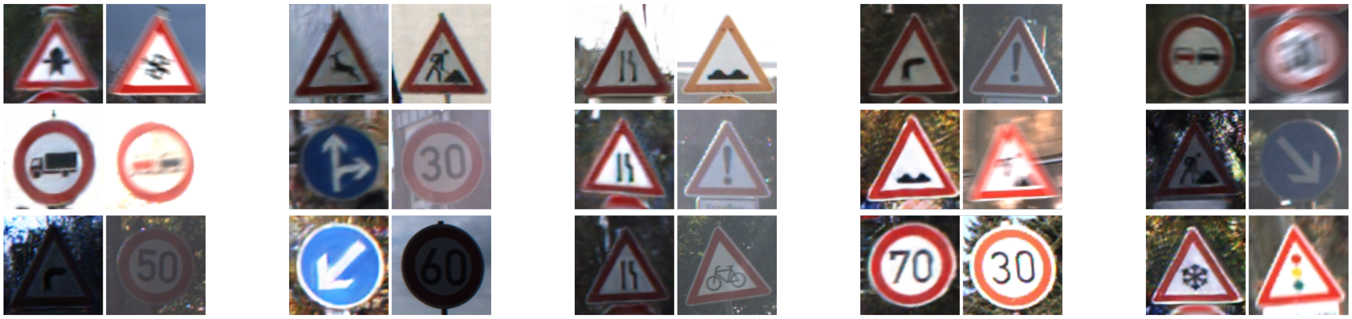

Fig. 4. A side-by-side comparison of generated and real traffic sign images. Each image pair shows a generated image on the left and its nearest neighbor in terms of Euclidean distance from the CycleGAN training set on the right, demonstrating that the generated images are both plausible and unique.

References

[1] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” *Neural Networks*, vol. 32, pp. 323–332, 2012.

[2] 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 27*. Curran Associates, Inc., 2014, pp. 2672–2680.

[3] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv:1511.06434*, 2015.

[4] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” in *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 2172–2180.

[5] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier gans,” *arXiv:1610.09585*, 2016.

[6] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *arXiv:1703.10593*, 2017.

[7] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2414–2423.

[8] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 694–711.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *arXiv:1611.07004*, 2017.

[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 105–114.

[11] P. Sermanet and Y. LeCun, “Traffic sign recognition with multi-scale convolutional networks,” in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 2809–2813.

[12] F. Zaklouta and B. Stanciulescu, “Real-time traffic sign recognition in three stages,” *Robotics and Autonomous Systems*, vol. 62, no. 1, pp. 16–24, 2014.

[13] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool, “Traffic sign recognition—how far are we from the solution?” in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–8.

[14] A. Gudigar, S. Chokkadi, and R. U, “A review on automatic detection and recognition of traffic sign,” *Multimedia Tools and Applications*, vol. 75, no. 1, pp. 333–364, 2016.

[15] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund, “Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey,” *Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012.

[16] F. Larsson and M. Felsberg, “Using fourier descriptors and spatial models for traffic sign recognition,” in *Proceedings of the Scandinavian Conference on Image Analysis (SCIA)*, 2011, pp. 238–249.

[17] R. Timofte, K. Zimmermann, and L. Van Gool, “Multi-view traffic sign detection, recognition, and 3d localisation,” *Machine Vision and Applications*, vol. 25, no. 3, pp. 633–647, 2014.

[18] A. Chigorin and A. Konushin, “A system for large-scale automatic traffic sign recognition and mapping,” *Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, City Models, Roads and Traffic*, vol. II-3/W3, pp. 13–17, 2013.

[19] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin, “Evaluation of traffic sign recognition methods trained on synthetically generated data,” in *Proceedings of the International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS)*. Springer, 2013, pp. 576–583.

[20] H. Luo, Q. Kong, and F. Wu, “Traffic sign image synthesis with generative adversarial networks,” in *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2540–2545.

[21] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, 1995.

[22] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 886–893.

[23] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” *arXiv:1607.08022*, 2016.

[24] S. Gross and M. Wilber, “Training and investigating residual nets,” *Facebook AI Research, CA. [Online]. Available: http://torch.ch/blog/2016/02/04/resnets.html*, 2016.

[25] X. Mao, Q. Li, H. Xie, R. Y. Lau, and Z. Wang, “Multi-class generative adversarial networks with the l2 loss function,” *arXiv:1611.04076*, 2016.

[26] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980*, 2014.

[27] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, “Learning from simulated and unsupervised images through adversarial training,” *arXiv:1612.07828*, 2016.