2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

Traffic Sign Image Synthesis with Generative Adversarial Networks

Hengliang Luo*‡, Qingqun Kong†‡ and Fuchao Wu*
*National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China 100190
†Research Center for Brain-inspired Intelligence, Institute of Automation,
Chinese Academy of Sciences, Beijing, China 100190
‡University of Chinese Academy of Sciences, Beijing, China 100049
Email: hengliang.luo@nlpr.ia.ac.cn, qingqun.kong@ia.ac.cn, fcwu@nlpr.ia.ac.cn

***Abstract*—Deep convolutional neural networks (CNN) has achieved state-of-the-art result on traffic sign classification, which plays a key role in intelligent transportation system. However, it usually requires a large number of labeled training data, which is not always available, to guarantee a good performance. In this paper, we propose to synthesize traffic sign images by generative adversarial networks (GANs). It takes a standard traffic sign template and a background image as input to the generative network in GANs, where the template defines which class of traffic sign to include and the background image controls the visual appearance of the synthetic images. Experiments show that our method could generate more realistic traffic sign images than the conventional image synthesis method. Meanwhile, by adding the synthesis images to train a typical CNN for traffic sign classification, we obtained a better accuracy.**

I. Introduction

Traffic sign recognition aims to detect and classify any traffic sign existed in an image, and plays a vital role in advanced driver assistance systems and autonomous driving. After detecting localizations of traffic signs in an image, the final step is to classify these signs into pre-defined classes (e.g., *speed limit of 80*, *stop sign*, etc.) based on their local appearance. Such an image classification problem has been extensively studied in the past decades and convolutional neural networks (CNN) offers an excellent solution to this problem [1], [2], [3]. In other words, it is not hard to build a CNN-based traffic sign classification system if a large number of labeled training data is available.

Usually, the traffic sign images can be collected via car-mounted cameras or from street view images provided by map service providers. Then, the locations and class labels of traffic signs are annotated by human labelers for each collected image. However, it is not easy to collect a large number of data with high quality for the following reasons: 1) Different countries have different designs of traffic signs, which means that it has to collect and label different data for different countries. 2) To increase the robustness of the recognition system, the training data should cover different background, illumination and weather conditions as much as possible. 3) The distribution of traffic signs is unbalanced in the real scenario, which further increases the labors devoted for collecting images of those rare traffic signs. For example, *speed limit signs* are ubiquitous, while the signs like *falling rocks* may only exist in some special roads. As a result, it usually needs to go over more roads to collect as many images of *falling rocks* as those of *speed limit signs*. To acquire more training data with diversity and low cost, traffic sign images synthesized from standard templates have been widely used to train machine learning based classification algorithms [4], [5], [6], [7]. These kinds of methods can be roughly described as follow: Geometry and color transformations are first applied on the standard traffic sign templates. Then, the changed templates are added to background images. Finally, the generated images are blurred with Gaussian blur of different kernel sizes. However, the synthetic images are not as real as the collected data in visual appearance as shown in Fig. 7, where the background and the traffic sign do not merge well. Moreover, the classification result of using these synthetic data is also not as good as that of using the real data.

Recently, Generative Adversarial Networks (GANs) [8], [9] have achieved very remarkable results in image generation. While the original GANs generate images from random vectors, the conditional GANs [10], image-to-image GANs[11], [12], [13], have been proposed to extend the original GANs to generate images under some certain conditions. GANs include two networks, one generative network and one discriminative network, which are trained with competitive goals. The goal of discriminative network is to classify the real data and synthesized data correctly, while the objective of generative network it to generate images which can be classified as real by discriminative network. At last, the generative network can generate realistic images similar to the real data. Due to these merits, GANs have been widely studied and used in various applications [14]. To the best of our knowledge, it has not been used in generating traffic sign images. For this task, the image-to-image GANs seem to be a good choice by generating traffic sign from standard traffic template. However, image-to-image GANs convert an image to an almost fixed style without much diversity, while the purpose of traffic sign image synthesis is to generate images with different styles such as lighting and background. Alternately, we propose in this paper to generate traffic sign images by incorporating a background image and a standard template of traffic sign by a generative

978-1-5386-3788-3/18/$31.00 ©2018 IEEE

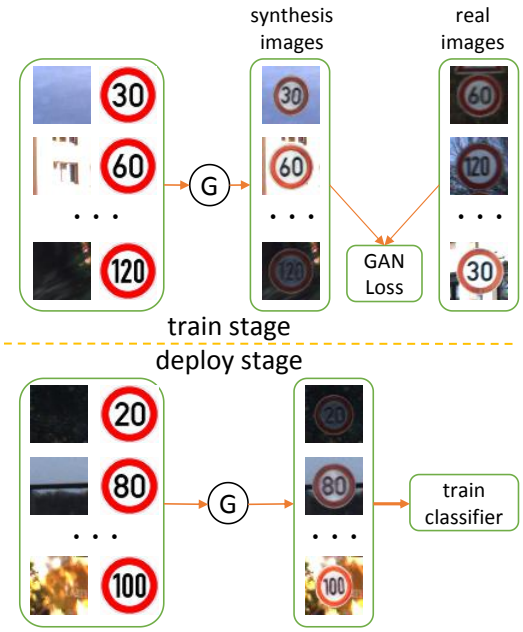


Fig. 1. Overview of the proposed method for traffic sign image synthesis.

network, which is trained in the framework of GANs (cf. Fig. 1). The background image is taken as input to transfer the illumination style to the synthesized traffic sign so as to incorporate the given background and the foreground (traffic sign) more naturally. As shown in Fig. 8, our generated traffic sign images look more realistic than the ones generated by the traditional method (Fig. 7). To sum up, the main contributions of this paper are as follows: 1) We propose a new generative network with inputs of standard traffic sign templates and background images, in which the background images control the visual appearance and illumination of the synthetic traffic signs. 2) Our method achieves better results than the traditional traffic sign synthesis method, both in terms of visual appearance and usefulness in training good classifier.

The rest of this paper is organized as follows. In Section II, we describe the proposed method for traffic sign image synthesis. Then, in section III, we report experiments and analysis the results. Finally, conclusions are drawn in Section IV.

II. Proposed Methods

Traditionally, to generate traffic sign images from standard templates, it includes four steps [5], [7]: geometry transformation, illumination rendering, background fusion and image blur. For geometry transformation, affine transformation is used to simulate the viewpoint change in the real scene as the traffic signs are always on planes. The affine transformation can be denoted by [15]

$$\mathbf{x}' = \mathbf{H}\mathbf{x} = \begin{bmatrix} \mathbf{A} & \mathbf{d} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{x} \tag{1}$$

where $\mathbf{x}'$ and $\mathbf{x}$ are homogeneous image coordinates. $\mathbf{d}$ is a 2D translation vector $[d_x, d_y]^T$, and $\mathbf{A}$ is a 2×2 non-singular matrix which can be decomposed as

$$\mathbf{A} = sR(\theta)R(-\phi)\begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} R(\phi) \tag{2}$$

where $R(\alpha)$ denotes the rotation matrix of α. There are six independent parameters $(dx, dy, s, t, \theta, \phi)$ in an affine transformation. For illumination rendering, it is achieved by changing the values of S and V channels in the HSV color space of the template image. While the step of background fusion is simple, where one background patch is randomly sampled from real traffic scene images, and the geometric and illumination changed template image is added to the background to get synthesized images. Finally, Gaussian blur with random kernel size is applied to the synthesized images so as to obtain the final generated images. As can be seen, in these steps (i.e., geometry transformation, illumination rendering, image blur), there are some parameters need to be specified, which are manually set by sampling from hand-crafted distributions. The illumination rendering method is also too simple to simulate the complicated real scene.

In this paper, we propose to generate traffic sign images using GANs, which can learn the generation parameters from real data instead of the manually specified parameters in the traditional method. Instead of directly using image-to-image GANs to generate traffic sign images from standard template which we found problematic according to our previous experiments, we use GANs to seamlessly incorporate background and geometric transformed traffic sign template as shown in the bottom of Fig. 1. We have also tried to generate geometric transformation in the framework of GANs, but we found that in this case the network can not converge properly. Therefore, we use GANs to synthesize the visual appearance of the merged background and the traffic sign, while the geometric transformation of the traffic sign is manually set as in the traditional method.

There are a generative network G and a discriminative network D in GANs. In our method, the structure of generative network G is shown in Fig. 2a. The input of the generative network G includes 3 parts, i.e., a standard traffic sign template $\mathbf{x}$, six affine transformation parameters $\mathbf{a}$ and a background image patch $\mathbf{b}$. For the output of generative network G, it also includes 3 parts, a synthesized traffic sign $\tilde{\mathbf{x}}$, a synthesized traffic sign without background $\mathbf{x}^B$, and a standard template after applying the affine transformation $\mathbf{x}^A$. $\tilde{\mathbf{x}}$ is the generated traffic sign image, while $\mathbf{x}^A$ and $\mathbf{x}^B$ are useful in training GANs as regularizers. Mathematically, the generative network can be denoted by $(\tilde{\mathbf{x}}, \mathbf{x}^B, \mathbf{x}^A) := G(\mathbf{x}, \mathbf{a}, \mathbf{b})$.

In our generative network G, one input in size of $48 \times 48 \times 4$ is taken as the input standard template $\mathbf{x}$, which contains 3 RGB color channels and one mask channel. Then, with the supplied affine parameters $\mathbf{a}$ (another input), an affine transformation is applied on $\mathbf{x}$, obtaining a transformed template named as *temp-affine* $\mathbf{x}^A$ and a transformation mask as *temp-mask*. For the transformed template *temp-affine*, a convolutional layer followed by LeakyReLU and a residual block [16] is used to extract mid-level features of the template.

For simplicity, we denote these steps as *block2* in Fig. 2. To seamlessly incorporate background and template, the third input (i.e., background image patch) with size of $48 \times 48 \times 3$ is transformed to mid-level convolutional features using operation *block1*, which has two convolutional layers followed by the mean and replication operations (see Fig. 2b for details). Note that in *block1*, the background image patch is first reduced to the spatial size of 1×1, and then replicated to the spatial size of 48×48. The purpose is to keep the illumination information while eliminating the structure information of the background. The two kinds of mid-level convolutional features extracted from the transformed template and the background image patch are then concatenated in the channel dimension. After that, the operation denoted as *block3* is applied to generate a traffic sign image with new visual appearance, from which the traffic sign target *sign-mask* $\mathbf{x}^B$ is selected using the mask *temp-mask*. The *block3* contains one convolutional layer, two residual blocks and one convolutional layer sequentially as shown in Fig. 2b. The residual blocks in *block2 and block3* are used to keep structure of the traffic sign. Finally, the background region *sign-mask* is cropped out using the inverse mask of *temp-mask*, and added to the traffic sign target *sign-mask* to get the synthesized result $\tilde{\mathbf{x}}$. In this network structure, all the three outputs ($\tilde{\mathbf{x}}$, $\mathbf{x}^B$, $\mathbf{x}^A$) are in size of $48 \times 48 \times 3$.

The structure of the discriminative network D is shown in Fig. 2c. The size of input is $48 \times 48 \times 3$, and the output size is 6×6. Similar to [12], [11], we use the local adversarial loss when learning D.

As in classical GANs, the goal of G is to generate images which can be classified as real by D, while the goal of D is to correctly distinguish the generated images and real images. Due to these objectives, the loss function for training G contains two parts:

$$L_G = L_{GAN} + \lambda L_{Reg}, \tag{3}$$

$$L_{GAN} = -\sum_i \log(D(\tilde{\mathbf{x}}_i)) \tag{4}$$

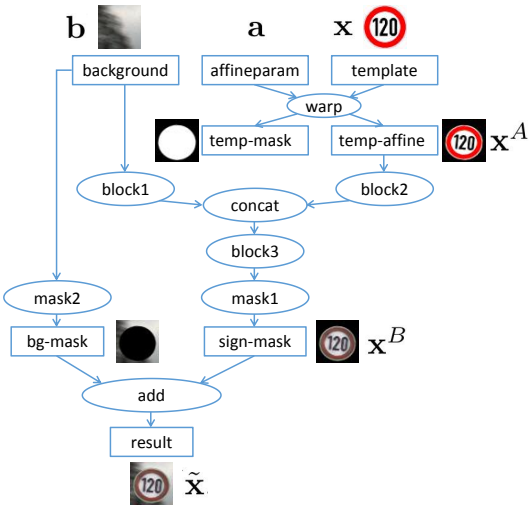
$$L_{Reg} = -\sum_i \mathbf{x}_i^A \log(\mathbf{x}_i^B) \tag{5}$$

L_{GAN} is to restrict the output of G so as to be classified as real traffic sign by D. L_{Reg} is the cross entropy loss between $\mathbf{x}_i^A$ and $\mathbf{x}_i^B$, whose aim is to regularize $\mathbf{x}_i^B$ using $\mathbf{x}_i^A$ since we want to keep the structure and color information in the synthetic traffic signs. λ is a weight to balance the two losses. It is worth noting that $D(\cdot)$, $\mathbf{x}_i^A$ and $\mathbf{x}_i^B$ are not scalars, and the average operation in loss functions (Eq. 4 and Eq. 5) is omitted for clarity.

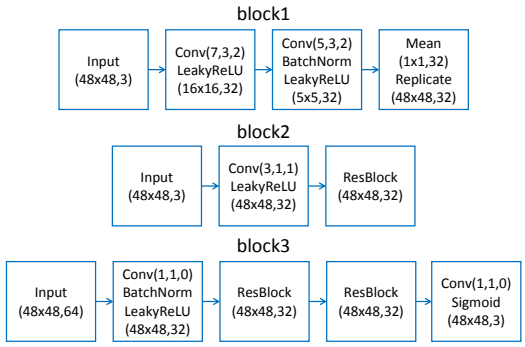
According to the objective of discriminative network, the loss function for training D is as simple as:

$$L_D = -\sum_i \log(1 - D(\tilde{\mathbf{x}}_i)) - \sum_i \log(D(\mathbf{y}_i)), \tag{6}$$

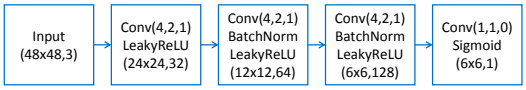
where $\tilde{\mathbf{x}}_i$ is a generated image, and $\mathbf{y}_i$ is a real image.



(a) Generative network.



(b) Three blocks in the generative network.



(c) Discriminative network.

Fig. 2. The structures of our generative and discriminative networks. The *warp* denotes warp affine operation. The *mask1* means mask operation using *temp-mask*. The *mask2* means mask operation using *1 - temp-mask*. The *Conv(k,s,p)* denotes a convolutional layer with kernel size $k \times k$, stride size s and pad size p. *LeakyReLU* denotes a leaky ReLU layer with slope of 0.2. BatchNorm denotes a batch normalization layer. *Mean* denotes mean operation on each spatial channel. *Replicate* denotes replicate operation on each spatial channel. *ResBlock* means a residual block *(Input-Conv(3,1,1)-BatchNorm-LeakyReLU-Conv(3,1,1)-BatchNorm)* $\oplus$ *Input-LeakyReLU*, where the number of channels are the same in all convolutional layers. *Sigmoid* means a sigmoid layer.

When train the GANs, G is first pre-trained with L_{Reg} to make G can roughly reconstruct itself. Then G and D are trained alternately. The detailed training procedure is shown in Algorithm 1.

During the training, the weight and model of the learned generative network G are saved every a certain number of iterations. It is expected that the styles of synthesized images using different models are different. As a result, when using G to generate data, it is straightforward to use multiple models to increase diversity of the synthesized data.

Algorithm 1: Training procedure of GANs

Input: Sets of traffic sign templates $\mathbf{x}_i$, background images $\mathbf{b}_i$, affine transformation parameters $\mathbf{a}_i$, real images $\mathbf{y}_i$, max number of pre-training steps (T_1), max number of training steps (T_2).

Output: Generative network G and discriminative network D.

for $t = 1, \ldots, T_1$ **do**

1. Sample a mini-batch of traffic sign templates $\mathbf{x}_i$, background images $\mathbf{b}_i$, affine transformation parameters $\mathbf{a}_i$.
2. Compute $(\tilde{\mathbf{x}}_i, \mathbf{x}_i^B, \mathbf{x}_i^A) := G(\mathbf{x_i}, \mathbf{a_i}, \mathbf{b_i})$.
3. Update parameters in G with Adam algorithm on mini-batch loss L_{Reg} in (5).

end

for $t = 1, \ldots, T_2$ **do**

1. Sample a mini-batch of traffic sign templates $\mathbf{x}_i$, background images $\mathbf{b}_i$, affine transformation parameters $\mathbf{a}_i$, real images $\mathbf{y}_i$.
2. Compute $(\tilde{\mathbf{x}}_i, \mathbf{x}_i^B, \mathbf{x}_i^A) := G(\mathbf{x_i}, \mathbf{a_i}, \mathbf{b_i})$.
3. Update parameters in D with Adam algorithm on mini-batch loss L_D in (6).
4. Update parameters in G with Adam algorithm on mini-batch loss L_G in (3).

end

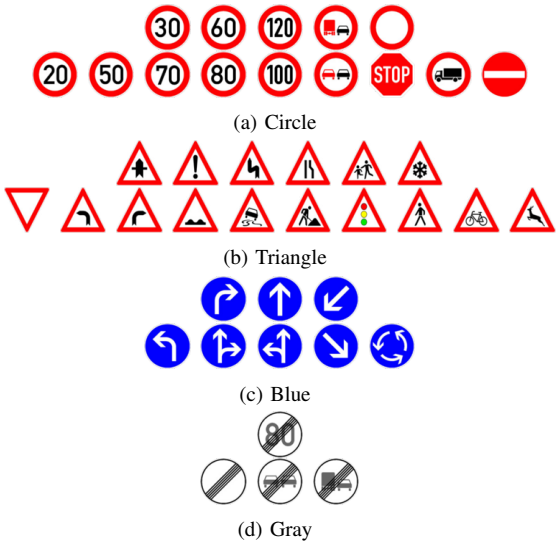
Fig. 3. Some samples in GTSRB data set.

III. Experiments

In this section, we evaluate the performance of the proposed method quantitatively and qualitatively. After briefly introducing the used dataset and implementation details, we show how the generated images can benefit a state of the art traffic sign classifier as well as its superiority to the previous method. We also show the qualitative results of using generative networks to synthesize traffic sign images.

A. Data Set

German Traffic Sign Recognition Benchmark(GTSRB) data set is used in our experiments. It includes 43 classes of traffic signs, where the numbers of training samples and test samples are 39209 and 12630, respectively. Fig. 3 shows some samples in GTSRB.



(a) Circle

(b) Triangle

(c) Blue

(d) Gray

Fig. 4. GTSRB data set is split into 4 categories according to the shapes and colors of the signs. The top signs in each category are used to train GANs, while the bottom signs are used for test. A traffic sign in GTSRB with totally different color and shape from the 4 categories is not used.

Fig. 5. Background image samples in GTSDB.

For the convenience of analysis, we further group these 43 different traffic sign classes into 4 categories based on their shapes and colors as shown in Fig. 4. For each category, the traffic signs are randomly divided into two parts, one for training GANs while the other for testing (i.e., using the generative network in the learned GANs to generate images of these classes), as shown in the top and bottom rows of Fig. 4 respectively. In this way, we make sure that there is no overlap between training and testing classes of GANs so as to evaluate the generalization ability of our method.

The backgrounds used in GANs are randomly sampled from real traffic scene images. In this experiment, we select and crop patches from traffic scene images in German Traffic Sign Detection Benchmark(GTSDB)[17], where the total number of traffic scene images is 100. Some examples are shown in Fig. 5.

B. Implementation Details

Our method is implemented in Torch[18]. The generative network and discriminative network are trained with Adam[19] optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$, and learning rate of 0.00002. The batch size is 50, and the hyperparameter λ in Eq. 3 is set to 0.5. The iterations for pre-training and training are set as 50 and 4000, respectively

TABLE I
NUMBERS OF TRAINING AND TEST SAMPLES IN EACH CATEGORY OF GTSRB DATASET

	Train				Test
	5%	10%	50%	All	
Circle	578	1152	5760	11520	3810
Triangle	353	702	3510	7020	2220
Blue	174	345	1725	3450	1080
Gray	36	72	360	720	210

For comparison, we implemented a traditional traffic sign synthesis method. As described before, the traditional method includes geometry transformation, illumination rendering, background fusion and image blur. Obviously, both of our method and the traditional method use same strategies for geometric transformation, background fusion and image blur. The difference lies in how to synthesize illumination changes. To render with different illuminations, the traditional method resorts to adjusting the HSV values of the template which is independent of the incorporated background. Specifically, the values of S and V channels are multiplied by a random number sampled from uniform distributions $U[0.5, 1.0]$ and $U[0.1, 1.0]$, respectively. The affine parameters $dx, dy, s, t, \theta, \phi$ are randomly sampled from $G[-3, 3]$, $G[-3, 3]$, $U[0.55, 0.85]$, $U[0.8, 1.05]$, $G[-\pi/20, \pi/20]$, $U[-\pi/30, \pi/30]$ respectively, where $G[a, b]$ denotes $\frac{b+a}{2} + \frac{b-a}{5}\mathcal{N}(0, 1)$. Note that the affine parameters in our method and the implemented traditional method follow the same sampling strategy. Finally, the generated images are Gaussian blurred with randomly selected kernel size from $\{1 \times 1, 3 \times 3, 5 \times 5, 7 \times 7, 9 \times 9\}$ as in our method.

C. Classification Result

We first show how different synthesis methods can help to improve classification performance when the number of labeled real images is limited. For this purpose, we implemented a state of the art classifier for GTSRB dataset [3]. To be specific, we used different training data to train the classifier, whose performance were then evaluated by the standard GTSRB test dataset. Therefore, only the training data is different for different methods, while the classifier and the test data are all identical.

As we have stated before, our GANs are trained with data in the top rows of Fig. 4 and used for generating images of classes in the corresponding bottom rows. Therefore, the classification experiments are conducted on these classes too. For neatness, we report the mean accuracy of these classes for each category. The total numbers of training and test samples in each category are listed in Table I.

Besides using synthesized data only, we also mix synthesized data with part of real data to train the classifier. The aim is to test whether the synthesized data can improve the performance of classifier when only a few real data is available.

The classification results are shown in Table II. Let us first look at the results of using only synthesized data (i.e., with 0% real data) in the first row of Table II, using GANs (especially the multi-model GANs) demonstrates a clear advantage over the traditional method. The results also show that using synthesized images is good enough to train a practical classifier. On the other hand, when the labeled training samples are limited, augmenting training set with synthesized images consistently improves the classification accuracy as shown in the other rows of Table II. These results also confirm that it is necessary to synthesize traffic sign images as in some cases that some traffic sign images are hard to collect. For example, for the 'Gray' category, the number of training images is small and when only 5% or 10% data is available, the classification performance degrades significantly. In this case, by using synthesized images for this category, the classifier performs as well as that when using enough real data. While both the traditional and the proposed image synthesis methods have a positive effect on classifier learning, using GANs as proposed in this paper usually leads to a better performance than the traditional method. While for GANs-based method, it is necessary to generate images using multiple different models so as to increase diversity of the generated data. Fig. 6 shows the output of generative networks with different models. It is clear that the synthesized images are different when using different models, thus the data diversity can be improved by using multiple models.

Fig. 6. The synthesis results with different models saved during the training iterations. Each row shows images generated by one model.

D. Visual effect of the Synthesis Images

To qualitatively compare the effectiveness of the proposed method and the traditional image synthesis method. We show images synthesized by traditional and our GANs-based method in Fig. 7 and Fig. 8, respectively. The traffic signs and backgrounds in Fig. 8 are more harmonious than those in Fig. 7. Fig. 8 also confirms that the background input in generative network G can influence the visual appearance of the generated traffic sign, which is exactly what we hope for when designing and using GANs. Although the traffic sign images synthesized by generative network look more realistic than those generated by the traditional method, however, when compared with the real traffic sign images in Fig. 3, the results of using generative network still need to be further improved.

IV. CONCLUSION

In this paper, we propose a new traffic sign image synthesis method. A generative network with inputs of a standard traffic sign template and a background image patch is designed for

TABLE II

CLASSIFICATION RESULTS WHEN USING DIFFERENT TRAINING SAMPLES, FROM ALL SYNTHETIC SAMPLES TO ALL REAL DATA. REAL MEANS USING REAL IMAGES FOR TRAINING, SYN MEANS USING IMAGES GENERATED BY TRADITIONAL SYNTHESIS METHOD, WHILE GAN(S) AND GAN(M) REPRESENT USING IMAGES BY OUR METHOD WITH SINGLE AND MULTIPLE MODELS. SEE TEXT FOR DETAILS.

Part of real		Real	Real+Syn	Real+GAN(s)	Real+GAN(m)
0%	Circle	-	94.86	96.30	**96.33**
	Triangle	-	96.58	96.17	**97.66**
	Blue	-	97.96	98.89	**98.98**
	Gray	-	**100.0**	**100.0**	**100.0**
5%	Circle	96.17	98.71	**99.69**	**99.69**
	Triangle	95.14	97.97	97.84	**98.24**
	Blue	98.89	99.26	**99.44**	99.35
	Gray	77.14	99.52	**100.0**	**100.0**
10%	Circle	98.61	99.40	99.79	**99.87**
	Triangle	97.07	97.88	**98.47**	98.37
	Blue	99.07	**99.54**	99.44	99.44
	Gray	86.67	99.52	**100.0**	**100.0**
50%	Circle	99.58	99.74	99.84	**99.87**
	Triangle	97.97	98.42	**98.56**	98.51
	Blue	99.44	**99.54**	99.44	**99.54**
	Gray	**100.0**	**100.0**	**100.0**	**100.0**
all	Circle	99.82	-	-	-
	Triangle	97.97	-	-	-
	Blue	99.44	-	-	-
	Gray	100.0	-	-	-

Fig. 7. Traffic sign images synthesized by traditional method.

Fig. 8. Traffic sign images synthesized by GANs.

this purpose. In this architecture, the background image patch controls the visual appearance and illumination condition of the synthetic traffic signs. This network is trained in the framework of GANs so as to generate realistic images. Experimental results show that our method not only generates more realistic images than the traditional method, but also achieves better classification results when used in the case of limited real labeled training data.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (61672032, 61573352, 61472119).

REFERENCES

[1] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3642–3649.

[2] M. Haloi, "Traffic sign classification using deep inception based convolutional networks," *arXiv preprint arXiv:1511.02992*, 2015.

[3] A. Desmaison, "The power of Spatial Transformer Networks," http://torch.ch/blog/2015/09/07/spatial_transformers.html, 2015, [Online; accessed 12-January-2018].

[4] J. Greenhalgh and M. Mirmehdi, "Real-time detection and recognition of road traffic signs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1498–1506, 2012.

[5] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin, "Evaluation of traffic sign recognition methods trained on synthetically generated data," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2013, pp. 576–583.

[6] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2110–2118.

[7] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 4, pp. 1100–1111, 2018.

[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*, 2014, pp. 2672–2680.

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

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

[11] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

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

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

[14] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *arXiv preprint arXiv:1710.07035*, 2017.

[15] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The german traffic sign detection benchmark," in *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, 2013, pp. 1–8.

[18] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.

[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.