Traffic Sign Repositories: Bridging the Gap Between Real and Synthetic Data

Diogo Lopes da Silva[1] and António Ramires Fernandes[2][0000−0002−3680−572X]

[1] Universidade do Minho, Braga, Portugal
[2] Algoritmi Centre/Department of Informatics/LASI, Universidade do Minho, Braga, Portugal arf@di.uminho.pt

Abstract. Creating a traffic sign dataset with real data can be a daunting task. We discuss the issues and challenges of real traffic sign datasets, and evaluate these issues from the perspective of creating a synthetic traffic sign dataset. A proposal is presented, and thoroughly tested, for a pipeline to generate synthetic samples for traffic sign repositories. This pipeline introduces Perlin noise and explores a new type of noise: Confetti noise. Our pipeline is capable of producing synthetic data which can be used to train models producing state of the art results in three public datasets, clearly surpassing all previous results with synthetic data. When merged or ensemble with real data our results surpass previous state of the art reports in three datasets: GTSRB, BTSC, and rMASTIF. Furthermore, we show that while models trained with real data datasets perform better in the respective dataset, the same is not true in general when considering other similar test sets, where models trained with our synthetic datasets surpassed models trained with real data. These results hint that synthetic datasets may provide better generalization than real data, when the testing data is outside of the distribution of the real data.

Keywords: Synthetic data · Traffic sign classification · Convolutional Neural Networks.

1 Introduction

Creating a representative recognition traffic sign repository is an appreciable challenge, requiring gathering and labelling samples for a large number of classes, under a variety of lighting and deterioration conditions. To aggravate things further, we must also consider the potential intraclass variability due to multiple versions of the same sign. Furthermore, pictograms and fonts vary from country to country, implying that the usage of a dataset across different countries will come with an associated performance cost. Building a representative multinational dataset will imply that the number of classes explodes, which in turn implies a much larger dataset, and potentially a larger supporting model.

Inspecting publicly available European traffic signs we find a reduced number of classes, and a significant number of classes with a low number of samples. Given a traffic sign dataset, some works address the accuracy problem from the

model architecture perspective. Complex architectures, namely Spatial Transformer Networks (STN), Inception modules, and Generative Adversarial Networks (GAN), have been able to achieve considerable accuracies. It is important to note, however, that the test sets for each particular traffic sign repository are commonly similar in terms of lighting and atmospheric conditions to respective the training sets. Hence, the accuracies reported in these works do not necessarily generalize to new samples captured under different conditions.

Another approach is to focus on the data. The usage of synthetic data for traffic sign repositories has been addressed multiple times in the literature [19], [11], [17], [5], [6], [9], [10]. Synthetic datasets eliminate several of the issues mentioned above. Nevertheless, the issue of getting a representative dataset remains relevant. While in synthetic datasets, the gathering and labelling processes are no longer required, simulating lighting and deterioration conditions remains a challenge.

The only required input for a synthetic dataset in [19, 1, 10] is a set of templates for each class. A set of geometric and colour operations are then applied to provide sample diversity, in an attempt to achieve a truly representative dataset. Other approaches require the existence of real data, and mostly use GANs in an attempt to generate synthetic samples that belong to the same distribution as real data [11, 17, 5, 9].

In [10] a pipeline for the generation of synthetic data for traffic sign repositories was proposed. The results obtained provide some hope that the gap between real and synthetic data can be closed for traffic sing repositories. The proposal does not require real data, but it shows how, given the availability of such data, performance can be increased for some datasets.

The present work is an extended version of [10], where we expand the discussion relating to real vs. synthetic data, detailing the issues that each approach faces. We also provide extended testing, adding detailed testing on the new operators proposed, adding new datasets, and extended our analysis of the results, allowing us to consolidate our conclusions.

2 Synthetic vs. Real Traffic Sign Datasets

Gathering enough data for a real traffic sign dataset is both a time and resource consuming task, with well over 100 different traffic signs classes requiring collecting a significant number of samples per class.

The following issues concerning real traffic sign datasets were identified during this work:

- Scarcity: some signs are rare in particular countries. For instance, the diamond shape yield sign is uncommon in Portugal. Collecting a significant number of these signs might be unfeasible.
- Placement: some signs can only be found at certain regions, for instance, the warning sign for snow. This implies that travelling is required to those regions to gather samples.

- Lighting: lighting varies along the day, and seasonally as well. Even if we dismiss the seasonal variation, the intraday variation (including cloudy skies) can affect the accuracy of a model.
- Weather exposure, graffiti and stickers: these are all elements that can degrade significantly the ability to recognise a sign. See Figure 1 for some examples from Belgium.
- Adverse atmospheric conditions: the presence of rain, snow and fog should be taken into account when gathering samples to provide high accuracy in what are probably the most demanding situations for the driver.
- Intraclass variation: Traffic signs within the same class can have different pictograms, or use different fonts. This is mostly due to the introduction of new versions over time for some classes. Furthermore, manufacturing issues can also cause differences in the pictograms. See Figure 2 for some examples from German traffic signs.
- Maintenance: When new traffic signs are introduced, it will take some time until sufficient samples are gathered to retrain the model.
- Camera equipment: different sensors will produce samples which will differ in contrast, brightness and hue.

Fig. 1. Sample of signs from Belgium that show some degradation such as weathering, graffiti, and stickers.

Fig. 2. Sample of intraclass variations per country in GTSRB. Letf and middle: variation due to new templates being introduced; Right: variation due to manufacturing.

Another issue is the absence of an international observed standard for traffic sign pictograms. Traffic signs can vary significantly from country to country as can be seen when inspecting the ETSD, which contains samples from six European countries. Some examples are provided in Figure 3. This implies that simply using a traffic sign dataset across countries is not an option, although some classes can be reused.

While all these issues do not impair our ability to correctly interpret a traffic sign, they can have a severe effect on a deep learning model accuracy.

Fig. 3. Sample of intraclass variations that can be found across different countries.

Regarding synthetic traffic signs, and considering building a synthetic dataset for a country, scarcity is not an issue, as there is no inherent limitation on the number of samples to be synthesised. Similarly, there is no issue with particular signs only being found in certain regions. Intraclass variation is also easily tackled as multiple templates can be used per class. Maintenance is required each time a new template is added to the dataset, but for synthetic signs only retraining the model is needed, and this can be done as early as when the new sign is designed.

The real issues when building synthetic traffic sign datasets are related to lighting, degradation, atmospheric conditions, and differences in sensors.

Being able to simulate these conditions will allow for a well designed dataset. This is not a simple task, however, once achieved, building traffic sign datasets for any country is only a matter of gathering the respective templates and train a model.

3 Related Work

This section focuses on the European traffic sign repositories used in this work, and on research relating to synthetic traffic sign datasets.

3.1 Traffic Sign Datasets

Only for a few countries have traffic sign samples been collected, labelled, and released as a public dataset. The volume and quality of samples varies greatly across countries.

This works focuses mainly on three European datasets: GTSRB[3] [18] (Germany); BTSC[4] [20] (Belgium); and rMASTIF[5] [23] (Croatia).

Table 1 presents some statistics for these datasets.

In [3], Serna and Ruichek propose the European Traffic Sign Dataset (ETSD). This dataset merges existing datasets from six European Countries: Belgium, Croatia, France, Germany, Netherlands, and Sweden. For Croatia and Germany, besides using the previously identified datasets the authors also took advantage of the respective detection datasets to gather more samples. Regarding France, Netherlands, and Sweden, the datasets used were:

[3] http://benchmark.ini.rub.de/
[4] https://btsd.ethz.ch/shareddata/
[5] http://www.zemris.fer.hr/ kalfa/Datasets/rMASTIF/

Table 1. Statistics for the German, Belgian, and Croatian datasets.

	GTSRB	BTSC	rMASTIF
class #	43	62	31
train #	39209	4575	4044
test #	12630	2520	1784
min res	25x25	22x21	17x16
max res	232x266	674x527	185x159

- Stereopolis dataset (France) [14];
- UTBM (France [3]
- RUG Traffic sign image database (Netherlands) [2]
- STS (Sweden) [8]

The final ETSD dataset has 164 classes, with a training/test split of 60546/21930 samples. Data from this dataset is used for cross-testing, see Section 5.3.

Other datasets are available, for instance the Italian or DITS dataset [21], and the Tsinghua-Tencent 100K benchmark [22].

3.2 Synthetic Traffic Signs

Stergiou et al. [19] proposed the generation of synthetic training dataset based on traffic sign templates. Templates for 50 classes of British traffic signs were gathered, and composed with background images of British roads, both from rural and urban areas.

Processing the templates consisted in both colour and geometric processing. Regarding colour, the goal was to simulate different lighting conditions, in order to approximate real life scenarios, with the final dataset containing 4 brightness variations. Considering the geometric transformations, 20 distinct affine transformations for shearing were applied, alongside rotations, scaling, and translations.

This dataset was evaluated with a CNN model with 6 convolutional layers achieving a peak accuracy of 92.20%. However, since the test dataset has not been provided, no comparisons with other works can be made.

Luo et al. [11] approached the synthetisation of the dataset based on Generative Adversarial Networks (GAN). The authors also implemented a conventional pipeline to generate synthetic samples using both colour and geometric transformations and claim to achieve more realistic imagery with the GAN approach.

The main purpose in using GANs is that the GAN itself will learn the generation parameters from real data, as opposed to the conventional pipeline where the parameters are manually tuned. On the down side, this approach requires existing real data to train the GAN, and unless the real data is truly representative, the synthetic data will inherit biases from the real data.

As input, the algorithm receives a sign template, an affine transformation, and a background. The GAN is responsible for the synthesis of the merging the background and the traffic sign template. The geometric transformations are applied independently as in Stergiou et al. [19].

An accuracy of 97.24% was achieved on a subset of the GTSRB test set, not including the yield diamond shade sign. For comparison purposes Luo et al. also presents the model accuracy when training with only real data: 99.21%. A dataset consisting of merged real and synthetic data was also tested achieving an accuracy of 99.41%, using 50% of the real training data.

Extending the proposal of Luo et al. [11], Spata et al. [17] propose to generate the background itself with a GAN. The geometric transformations are still applied independently. As the authors state, "the CycleGAN is designed primarily for stylistic and textural translations and therefore cannot effectively contribute such information itself". The reported accuracy result with synthetic datasets is 95.15%.

Horn and Houben [5] further explore the generation of synthetic data with CycleGANs, however, results are only provided for selected classes.

Araar et al. in [1] propose a conventional pipeline with geometric transformations and image processing techniques, as in Stergiou et al. [19]. With a DenseNet architecture, an accuracy of 97.83% in GTSRB is reported using only synthetic data.

Liu et al. [9] explore the generation of synthetic data using a DCGAN trained on real data. Their work shows that it is possible to create images with a high degree of similarity based on the SSIM metric.

Horn et al. [6] propose assessing the quality of generated synthetic samples using four different measures. The main purpose is to evaluate if a synthetic image is significantly different from the distribution of real images.

Luo and Wang [12] propose a pipeline to label real images. A synthetic dataset is produced using conventional geometric and colour operations on templates, which are then merged to real backgrounds. This dataset is then used to train a model that will be used in real unlabelled data in order to provide the labels. Their results are not directly comparable to previous works since the authors are more focused on recall and not on the performance of the synthetic dataset per se. This process is repeated and a recall of 98.6% is achieved with all images being correctly classified.

A relevant note is that all works strived to generate synthetic samples as close to real samples as possible.

4 Synthetic Traffic Signs Generation Algorithm

As in previous works, a synthetic sample is a composition of a background image with a foreground template that undergoes a set of operations.

The traffic sign synthesising algorithm is a pipeline of geometric transformations, colour transformations, and image disturbances in the form of noise and blur. To define the set of operations for our pipeline we examined real traffic sign datasets to identify the relevant operations to include.

To define the set of templates used we inspected only the training set of each dataset. A sample of the gathered templates for GTSRB is shown in Figure 4.Some classes have multiple templates due to the presence of older versions of a

sign, or even manufacturing differences (an example can be seen in the templates for $120km/h$ speed limit). This is common, but not exclusive to speed limit classes. Some templates are rotated to accommodate for the real sign placement, for instance the roundabout sign.

Fig. 4. Sample templates for the GTSRB.

An interesting issue arises when traffic signs share the same pictogram, yet belonging to different classes. And example can be found in the rMASTIFF dataset. The main difference between these signs is the shape and colour of the outer area of the sign. This naturally results in a number of inter-class misclassifications. To deal with this issue multiple templates were used for each class varying the hue and luminance channels, see Figure 5. This approach successfully reduces the number of misclassified samples from both classes.

Fig. 5. Templates for classes with same central pictogram (rMASTIFF). Source [10]

In the remainder of this section we first discuss the background options, the usage of information from real data distribution, new operators to synthesise samples, concluding with the full pipeline presentation.

4.1 Background

As discussed in Section 3.2, the usage of real scenario backgrounds in synthetic samples is the common approach. In our work, the real imagery backgrounds come from signless images from Google Street View.

As depicted in Figure 6, for our synthetic data generation we further tested an alternative background approach: random solid colour per sample.

While real backgrounds provide more realistic imagery they may introduce a bias in the training set. Different regions share different architectural trends, and even rural areas can be very diverse. Thus, it can be challenging to find a

Fig. 6. Real vs. solid colour backgrounds. Source [10]

suitable set of backgrounds covering a significant number of scenarios. Furthermore, adding weather conditions, and lighting variations due to time of day or even seasons, only aggravates this quest for having representative backgrounds.

Random solid colour backgrounds, on the other hand, avoid all the previously discussed issues, and "force" the network to focus on the traffic sign since there are no features outside of the traffic sign. This approach has been tested previously in [1], but Araar et al. discarded this option due to poor results.

In our pipeline, we explore both real ans solid colour backgrounds.

4.2 Brightness Distribution

Real image data can be modelled by statistical data distributions for some of its parameters. Brightness is the example explored in this work. The availability of real data allows to compute brightness for synthetic samples based on the real data brightness distribution.

To find a distribution that fits the real dataset brightness distribution of the real datasets, the Kolmogorov–Smirnov test (K-S test) was performed. Running the K-S test on all available samples from the three datasets, we found that the Johnson distribution with bounded values was able to closely fit the real sample data. The histogram plot of the distribution for the three main datasets explored in our work is depicted in Figure 7. Table 2 presents the parameters of the distributions.

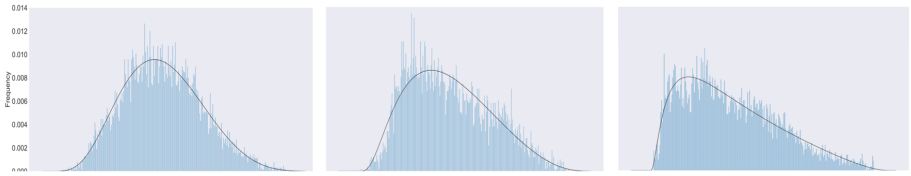

Fig. 7. Brightness frequency distribution for BTSC, rMASTIF, and GTSRB. The curve represents the Johnson fitted distribution. The horizontal axis represents the sample average brightness.

To adjust the brightness of a synthetic sample, a brightness value was sampled from the Johnson distribution, and the average brightness of the template is adjusted to match the sampled brightness. Examples of the end result can be seen in Figure 8.

Table 2. Fitted Johnson brightness distribution parameters for the German, Belgian, and Croatian datasets. Source [10].

	Parameter			
dataset	γ	δ	ξ	λ
GTSRB	0.747	0.907	7.099	259.904
BTSC	0.727	1.694	2.893	298.639
rMASTIF	0.664	1.194	20.527	248.357

Fig. 8. Brightness variation in synthetic samples of class 7 for the GTSRB dataset. Source [10].

While this approach may provide trained models with higher accuracy in samples with the same brightness distribution, the usage of a brightness distribution also contaminates the synthetic dataset with the biases present in the training set. As can be seen in Figure 7 the distribution for the three datasets varies significantly. Furthermore, this approach can only be used if real data is available. Note, however, that to compute a brightness distribution an unlabelled set of real traffic signs is sufficient.

To create a synthetic dataset from scratch we propose the usage of exponential Equation 1, where the desired brightness is computed considering a uniform random variable u in $[0, 1]$, and a variable $bias$ that determines the minimum brightness. Brightness B can be defined in the range $[bias, 255]$ as:

$$B = bias + u^{\gamma} \times (255 - bias) \tag{1}$$

In our tests we set $bias = 10$, and $\gamma = 2$.

For both approaches the process to adjust the template brightness is identical, being performed in HSV colour space. The first step is to compute the average V component in HSV representation. A ratio between the desired brightness and the mean V value is computed, and multiplied by V for every pixel.

4.3 Confetti Noise

A significant portion of the smaller samples from the real datasets have abrupt pictogram colour variations. Some examples are presented in Figure 9.

Our approach to simulate this phenomena is based on impulsive noise. This noise, which we named Confetti Noise, modifies the value of pixels in a random fashion, being applied only to the smaller samples.

Confetti noise is only applied to smaller samples and has 3 parameters. The kernel size ratio (3% of the original template dimension), the probability of

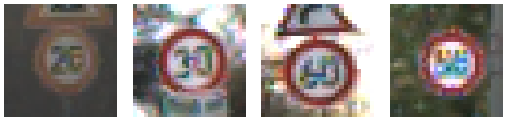

Fig. 9. Examples of noisy traffic sign samples of classes 0, 1, 2, and 3 from the GTSRB dataset, respectively. Source [10].

updating the window under the kernel (set at 3%), and the stride (set at 1.5% of the template dimensions).

Finally, the template is resized to the desired resolution. The effect on a template and a comparison with an actual traffic sign is depicted in Figure 10.

Fig. 10. Confetti noise. From the left: original template, resized sample, resized template after applying confetti noise, and sample from GTSRB. Source [10].

4.4 Perlin Noise

Perlin Noise [15] has been used in Computer Graphics as a technique to produce natural appearing textures. In the context of creating a synthetic dataset, Perlin Noise can be used to simulate the heterogeneity found in real traffic signs due to uneven light exposure, colour fade, or deterioration due to exposure. Hence, Perlin noise was added to our pipeline.

The process of applying Perlin noise to the templates consists first in random cropping of a large noise texture, and alpha blending the crop with the template, see Equation 2 . The Perlin noise parameters are as follows: 6 octaves, a persistence of 0.5, and a lacunarity of 2.0. Figure 11 shows examples of Perlin noise applied to different templates.

$$final = (1 - \alpha) \times template + \alpha \times noise \tag{2}$$

4.5 Synthetic generation pipeline

Most of the operations we use to generate synthetic samples are similar to those in [19] and [1]. The geometric operations used were resizing, translation, rotation, and perspective transforms. Regarding colour we also perform hue and saturation jitter.

Fig. 11. Perlin noise sample (left) applied to classes 1, 36, and 41 from the GTSRB dataset with $\alpha = 0.4$. Source [10].

The full pipeline[6] is depicted in Figure 12. On the side of each box the probability of applying the respective transformation to a sample is displayed. Branching occurs to provide a different treatment to small and large samples. Depicted in bold are the items that represent the novelty and were discussed above.

Figure 13 presents samples of synthesised samples considering solid colour backgrounds. As opposed to previous works our samples are clearly not photo-realistic.

5 Evaluation

As this work is focused on the datasets an not on fine tuning an architecture, we opted for a plain vanilla CNN. A summary of the CNN architecture employed in this work can be seen in Table 3.

This model consists of three convolution blocks with a kernel size of 5×5 pixels and one fully connected layer. The activation function used in all convolutional layers is LeakyReLU, while the fully connected layer uses ReLU. Batch size is set to 64, and the Adam optimizer is used with a learning rate of 0.0001.

Most tests are performed on three of the datasets presented in Section 3.1, namely GTSRB, BTSC, and rMASTIF. All values reported are averages of 5 runs, each completing 40 epochs.

On real datasets data augmentation has been performed prior to training to achieve a more balanced dataset, with each class ending with at least 2000 samples. To take advantage of the available data we first performed horizontal flipping where applicable. This is particularly useful when the flipped image ends up in another class, as is the case with turning signs. After this step, and for those classes still having less than 2000 samples, we performed common geometric operations, namely, translations (up to four pixels in each direction), and rotations ($-10°$ to $10°$).

Dynamic data augmentation consisting of geometric and colour operations is applied to the dataset during training. The operations involved are: rotations with a maximum of $5°$ in each direction; shear in the range of $[-2, 2]$ pixels followed by rotations; translations in a range of $[-0.1, 0.1]$ percent, also followed by rotations; and centre cropping of 28×28 pixels. The colour transformation

[6] source code for the generation of synthetic datasets available at https://github.com/Nau3D/bridging-the-gap-between-real-and-synthetic-traffic-sign-datasets

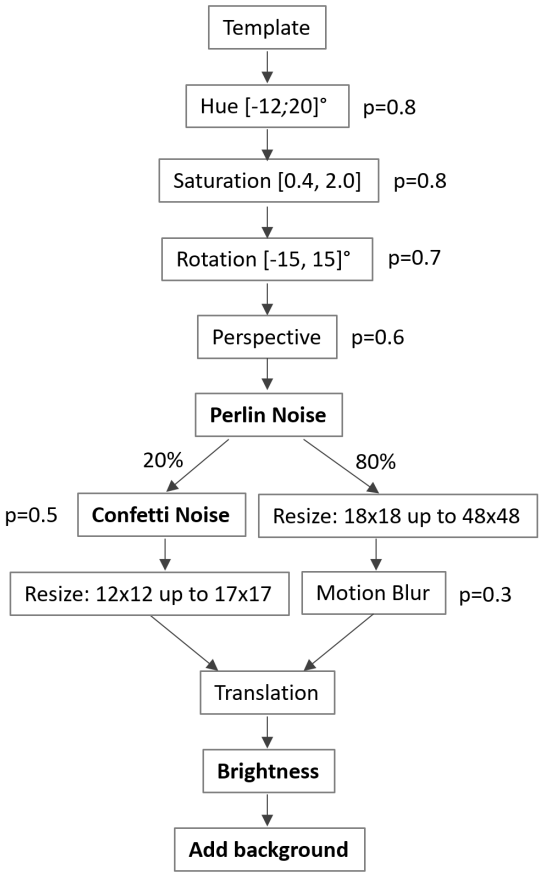

Fig. 12. Synthetic template transformation pipeline.

consists in jittering of the brightness, saturation, contrast, each multiplied by a random value in the range $[0, 3]$, and hue jittered in the range $[-0.4, 0.4]$. As mentioned in Section 4, when classes have similar or even identical pictograms, and only the outer colour differs, it is preferable to have multiple templates for those classes and a smaller range for hue jittering. For the Croatian rMASTIF dataset the range $[-0.2, 0.2]$ produces better results.

To increase the diversity, these transformations are applied independently, i.e., for each sample in the original dataset, eight more samples are produced in the data augmentation process.

Accuracy results for real data are reported in Table 4, including state-of-the-art results, to put in context the results obtained with synthetic data presented in the following subsections. Our results are an average of 5 runs per dataset.

Fig. 13. Samples of generated synthetic traffic signs with solid colour backgrounds for each class of the GTSRB dataset. Source [10].

Table 3. Neural network model with a total of approximately 2.7 million trainable parameters. The number of outputs is set according to the number of classes of the dataset. Source [10]

Layer type	Filters	Size
Input		32×32
Convolution + LeakyReLU	100	5×5
Batch Norm.		
Dropout ($p = 0.05$)		
Convolution + LeakyReLU	150	5×5
Max Pooling		2×2
Batch Norm.		
Dropout ($p = 0.05$)		
Convolution + LeakyReLU	250	5×5
Max Pooling		2×2
Batch Norm.		
Dropout ($p = 0.05$)		
Fully Connected + ReLU		350
Fully Connected + ReLU		# classes (c)

Table 4. Accuracy results when training with real data. Number of parameters is 10^6.

Dataset	Model	Input	Params	Acc (%)
GTSRB	Ours	32×32	2.7	99.64 ± 0.02
	Mahmoud and Guo. [13]	64×64	–	99.80
	Haloi [4]	128×128	10.5	**99.81**
BTSC	Saha et al. [16]	56×56	6.3	99.17
	Ours	32×32	2.7	99.30 ± 0.03
	Mahmoud and Guo. [13]	64×64	–	**99.72**
rMASTIF	Jurišić et al. [7]	48×48	6.3	99.53
	Ours	32×32	2.7	**99.71** ± 0.05

Note that we are using a smaller input when compared to reference works. This results not only in a smaller memory footprint, but also a faster evaluation due to the lower number of convolution operations required.

5.1 Solo Synthetic Dataset Evaluation

Synthetic datasets have 2000 samples per class, and the dynamic data augmentation procedure is as described for real data datasets. According to Section 4.2 and 4.1 we have two variations for both brightness and background.

We prepared synthetic datasets to contemplate these variations resulting in 4 distinct synthetic dataset types. Brightness can be set according to the exponential equation (Equation 1), or sampling from the respective Johnson distribution. Backgrounds can be crops of signless real images, or just a solid colour square.

To identify the dataset type, we use **R** for real data datasets. Synthetic datasets are identified by a three letter abbreviation, always starting with **S** for synthetic. The second letter relates to brightness option and the third to the background used. The synthetic dataset types are referred to as:

- **SES** - **E**xponential brightness and **S**olid bgs;
- **SJS** - **J**ohnson brightness dist. and **S**olid bgs;
- **SER** **E**xponential brightness and **R**eal bgs;
- **SJR** - **J**ohnson brightness dist. and **R**eal bgs;

To provide more meaningful results, for each type we created five datasets varying a random seed.

Perlin noise was the first feature to be tested. Regarding our pipeline we turned off Confetti noise, and experiment with different intensities of Perlin noise, i.e., with different α values in Equation 2. The **SES** synthetic datasets were used for this test. Results reported in Table 5, clearly show the advantage of using Perlin noise, with a very significant increase in accuracy when comparing $\alpha = 0.0$ (no Perlin) with $\alpha = 0.4$. Even with 60% noise in the sample, the models outperform those trained with datasets without noise.

Note that the best setting, $\alpha = 0.4$, produces samples that look too "dirty", as can be seen in Figure 11. Smaller settings will produce more realistic values, however, higher accuracy is obtained with this more saturated version.

Table 5. Results on GTSRB for SES datasets for Perlin noise blending.

$\alpha = 0.0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$
98.60 ± 0.17	98.95 ± 0.12	$\mathbf{99.31} \pm 0.12$	99.06 ± 0.09	89.34 ± 0.23

Confetti noise was tested in combination with Perlin noise to evaluate its usefulness regarding accuracy, see Table 6.

Table 6. Combined Perlin and Confetti accuracy results for models trained with synthetic datasets for GTSRB. In the base datasets neither Perlin nor Confetti is applied.

datasets	C + P	P	C	base
SES	99.24 ± 0.12	$\mathbf{99.31} \pm 0.12$	98.70 ± 0.17	98.60 ± 0.17
SER	$\mathbf{99.34} \pm 0.09$	$\mathbf{99.34} \pm 0.08$	99.24 ± 0.10	99.06 ± 0.10
SJS	$\mathbf{99.25} \pm 0.09$	99.23 ± 0.08	98.70 ± 0.18	98.60 ± 0.17
SJR	$\mathbf{99.49} \pm 0.04$	99.44 ± 0.07	99.09 ± 0.07	99.08 ± 0.17

Regarding datasets with only Confetti noise (no Perlin noise) vs. the base dataset (no Confetti nor Perlin Noise), the usage of Confetti noise brings consistent improvement. Comparing the effect of Perlin noise vs. Confetti noise it is clear that the former has a significantly greater impact than the latter. This is to be expected as Confetti noise is only applied to smaller samples (20% of the dataset).

Combining both noise algorithms brings a somewhat unexpected result for the SES datasets. While all the other variations improve or at least maintain their accuracy when using both noise algorithms, there is a significant decrease in accuracy in the models trained with the SES datasets.

At the moment we don't have a justification for this behaviour, and further study is required to evaluate the Confetti noise impact when combined with Perlin noise. Nevertheless, as Confetti noise provides better results in the majority of cases, we will preserve it in all datasets for the sake of consistency with our previous work.

Results for all types of synthetic datasets in the three datasets used throughout this work can be found in Table 7. The results for all models trained with synthetic data are within less than 0.5% of the accuracy obtained with real data. This represents a clear step in bridging the gap between real and synthetic data.

The best previously reported accuracy for a synthetic dataset was by Araar et al. in [1], presenting an accuracy for GTSRB of 97.83%. Our results, in all

Table 7. Test dataset accuracy for models trained with synthetic data. The accuracy obtained with the real dataset is presented inside parenthesis.

	SES	SER	SJS	SJR.
GTSRB (99.64)	99.24 ± 0.12	99.34 ± 0.09	99.25 ± 0.09	$\mathbf{99.49} \pm 0.04$
BTSC (99.30)	$\mathbf{99.12} \pm 0.04$	98.86 ± 0.12	99.11 ± 0.09	98.92 ± 0.09
rMASTIF (99.72)	$\mathbf{99.47} \pm 0.09$	99.27 ± 0.14	99.26 ± 0.17	99.37 ± 0.08

the variants clearly surpass previous results. Is it noteworthy to point out that our work is the only one which does not pursue photo-realism when generating samples, as can be seen in Figure 13. This hints that pursuing photo-realism may not be a requirement, or even the best option.

Another interesting result is that, as opposed to the results reported by Araar et al. in [1], we managed to obtain very good results with solid colour backgrounds.

Considering our results, for both BTSC and rMASTIF the best result is obtained with the SES dataset. This is the most agnostic dataset, as it does not incorporate neither background or brightness information from real data. On the other hand, for GTSRB the best result was obtained almost in the opposite scenario: using both brightness information and real backgrounds.

When using brightness information for GTSRB the result is consistently better. This can be due to the fact that this dataset has the darkest samples on average and the narrowest brightness distribution curve, thereby benefiting from a more tailored brightness distribution. The fact that both BTSC and rMASTIF datasets offer better performance with solid backgrounds could be interpreted as a hint that the negative background dataset we are using is biased towards the German dataset.

5.2 Combining Real and Synthetic Data

Assuming real data is available, we can combine it with synthetic data in two ways: merging datasets and ensembling. This section describes both options.

Merging Real and Synthetic Data

To evaluate the benefits of merging both types of data, two synthetic versions were selected to be merged with real data: SES and SJS, i.e., solid background synthetic datasets with both brightness options. Based on the previously built datasets, we created 5 merged datasets for each brightness option.

As expected, the results show a clear improvement over previous results, see Table 8. Results also show a slight advantage when using brightness information from the real dataset.

Ensembles

Ensembling is a known technique in deep learning when multiple models are available. To take advantage of the diversity of data available, our approach is to

Table 8. Average accuracy results for merged datasets.

	Real + SES	- Real + SJS
GTSRB	99.70 ± 0.04	$\mathbf{99.75} \pm 0.02$
BTSC	99.36 ± 0.05	$\mathbf{99.40} \pm 0.05$
rMASTIF	99.81 ± 0.04	$\mathbf{99.84} \pm 0.07$

consider a synthetic dataset, a merged dataset, and the real dataset. This results in three trained models that we combine in an ensemble.

Considering that the trained merged models have solid colour backgrounds, the synthetic dataset will have real backgrounds. To increase diversity we will use the SER dataset, as it is more agnostic of the dataset than SJR since the latter includes brightness information from the training set.

Therefore, our ensemble has three models trained with the folloing datasets: SER, Real, and Merged (Real + SJS). This provides a diverse ensemble with both brightness and background options.

The ensemble was evaluated 5 times, selecting the i^{th} model of each type to build the i^{th} ensemble.

Table 9. Average accuracy results for ensembles. Source [10]

GTSRB	BTSC	rMASTIF
99.82 ± 0.02	99.38 ± 0.02	99.79 ± 0.05

Ensembling provides mixed results regarding accuracy. Both in BTSC and rMASTIF the results are worse compared to merging only. On a closer examination of the BTSC individual model results we can observe that there is a significant set of samples that are misclassified by most models. Hence, ensembling is unable to improve over the individual models accuracy. Note that the difference in percentage translates to a single sample as the test dataset is relatively small. In rMASTIF a similar situation was found.

On the other hand, for GTRSB the result is above the state of the art (99.81% from Haloi [4]), with the best ensemble for GTSRB achieved an accuracy of 99.85%. Although it may seem unfair to compare an ensemble to a single network, note that our input is only 32×32, compared to Haloi's 128×128. This implies that, although we are considering 3 models, it is likely that inference with our ensemble will be faster than with Haloi's model. Furthermore, our ensemble memory footprint is also smaller than Haloi's, see Table 4 for the number of parameters on both models.

5.3 Cross-testing

In Section 2 we mentioned the existing intraclass variety that can be found when considering the same traffic sign for different countries. We also discussed how lighting and camera sensors can be a relevant issue when pursuing the best accuracy performance.

This test aims at exploring these issues. It is also an assessment on the generalization capabilities of the models trained with different datasets.

The test consists of evaluating a model trained with a dataset for a particular country in a dataset of a different country. For instance, models trained with GTSRB datasets will be evaluated on test datasets from Croatia and Belgium. Compared to [10], we have extended this test to include two more test countries: France and Sweden. Data for this datasets came from the ETSD, where we include all classes from France and Sweden that semantically overlapped classes in the three main datasets used previously in this work: GTSRB, BTSC, and rMASTIF. Furthermore, this test includes more classes than in [10]. Figure 14 show the classes that were considered and provides samples for the respective datasets. For this test we included all classes that have similar pictograms and fonts, although templates may vary slightly.

Note that for GTSRB, BTSC, and rMASTIF, we use only the test datasets for evaluation since this also provides for a direct comparison with the previous reported results. However, since the French and Swedish datasets are smaller and have not been used previously in this work, we opted to use the full dataset (tranining + test). Results are presented in Table 10.

Based on the results presented it is fair to say that using a dataset designed for a country in another country is not advisable, with the reported accuracies for models trained with real data falling as low as 60.66%. This low performance suggests that the test data does not belong to the distribution of the real data. Although, this test has been presented as a test relating to the usage of a dataset built for a country being used in another country, this test can also be seen as evaluating how different camera sensors, lighting, and national intraclass variation would impact accuracy. From this perspective, it confirms the difficulty in gathering a truly representative dataset, as discussed in Section 2.

Yet, perhaps the most surprising result is the fact that synthetic datasets achieve higher accuracies in 9 out of 12 tests. In particular SES achieved the highest accuracy in 8 out of 12 scenarios. When considering the sum over each training dataset used, SES datasets provided the highest accuracies in all three cases, with real datasets offering the lowest accuracies. The predominance of SES over SER, the only difference being in the backgrounds, again hints that the usage of real backgrounds has the potential to introduce an undesirable bias. SES is the most agnostic dataset, not including lighting information from the respective real counterpart dataset. Together with the poor results obtained with the real datasets, this suggests that these real datasets are in fact not representative of the global population of traffic sign imagery.

Not only does SES provide the best results, but the differences towards the results obtained with real data can be very significant. The worst case is when

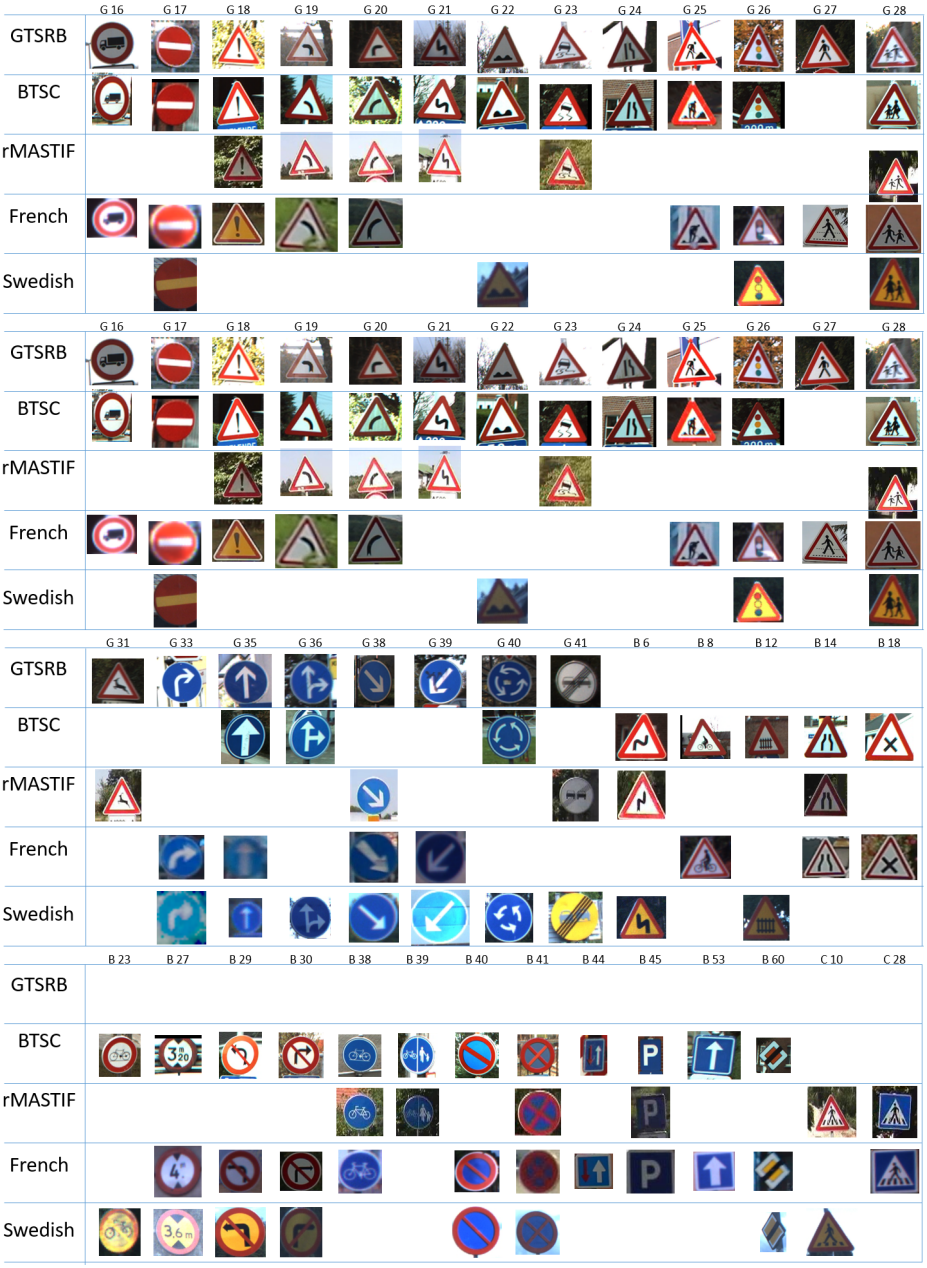

Fig. 14. Sample signs from classes with the same semantic meaning.

Table 10. Accuracy results for cross-testing. Each group describes the results obtained with models trained with the respective datasets; Rows indicate datasets used for evaluation purposes. Second column reports on the number of samples from the test datasets.

Test set	#	R	SER	SES
		Trained for GTSRB (Germany)		
Belgium	942	**97.98** ± 0.43	97.47 ± 0.52	96.58 ± 1.06
Croatia	1067	95.99 ± 0.38	99.18 ± 0.35	**98.78** ± 0.32
France	670	**81.94** ± 0.93	80.24 ± 0.79	81.79 ± 1.33
Sweden	4258	60.66 ± 1.10	62.81 ± 1.64	**64.03** ± 2.14
Total	6937	73.22 ± 0.76	74.64 ± 1.01	**75.51** ± 1.32
		Trained for BTSC (Belgium)		
Croatia	1019	66.48 ± 0.62	84.26 ± 1.29	**84.91** ± 1.60
German	5699	83.98 ± 0.97	**95.16** ± 0.39	93.69 ± 0.89
France	1100	77.95 ± 0.44	79.97 ± 0.90	**79.69** ± 0.88
Sweden	2395	**73.53** ± 0.81	65.57 ± 1.89	70.81 ± 1.22
Total	10213	79.13 ± 0.61	85.47 ± 0.62	**85.94** ± 0.74
		Trained for rMASTIF (Croatia)		
Belgium	1169	76.92 ± 1.71	84.36 ± 1.01	**85.37** ± 0.09
German	7109	91.65 ± 0.79	96.32 ± 0.72	**97.21** ± 0.59
France	685	80.93 ± 0.77	77.69 ± 1.21	**81.02** ± 1.27
Sweden	3601	70.83 ± 0.97	75.11 ± 1.15	**77.97** ± 1.26
Total	12564	83.73 ± 0.42	88.11 ± 0.54	**89.71** ± 0.26

comparing models trained for BTSC and tested on the Croatia test set, where the difference between accuracies reaches 18.43%. On the other hand, in the 3 scenarios where R beats SES the differences are significantly smaller: 1.4%, 0.15%, and 2.72%.

This difference between results obtained synthetic and real datasets indicates that the former may provide better generalization than the latter. This is particularly significant as SES is the most agnostic dataset.

5.4 Unleashing synthetic datasets

Up until now we gathered templates for each dataset observing only the training dataset. The set of templates for each class may therefore not be fully representative. This section unleashes the synthetic datasets in the sense that we are now free to examine the test datasets to seek for new templates.

Upon this exploration of the test sets it became clear that both in BTSC and GTSRB there are multiple signs whose templates are not found in the previous datasets. This is to be expected in real scenarios as discussed in Section 2, as

gathering real samples for all pictogram or font variations is an immensely time and resource consuming task.

This section reports on this exploration for two datasets: BTSC and GTSRB.

BTSC: In the Belgium dataset we found a set of six images from the test set that are misclassified by the majority of models, both trained with synthetic and real data. This problem was identified when we analysed the ensemble results. Figure 15 shows these samples. These samples belong to the same class, class 45, however, in the training set there are no samples with such templates. Figure 16 presents the initial templates considered, and the new templates added for this test.

Fig. 15. BTSC - Set of images misclassified by the majority of the models, both trained on real and synthetic data. Source [10].

Repeating the previous test with the SES dataset we found that not only we got 100% accuracy for class 45 but also that there were no adversarial effects on the other classes. Adding this templates we got an average accuracy of 99.31%, surpassing the results we got with the real dataset.

Although the difference to the acccuracy obtained with the real dataset is relatively small (0.01%) the synthetic dataset behaves considerably better than its real counterpart when considering the average accuracy obtained during training, as seen in Figure 17.

Fig. 16. BTSC - left: templates from the training set; right: new templates from the test set (note: the text was added just for the template and does not correspond to any real sign). Source [10].

As a final test we merged one of these datasets with the real BTSC dataset, achieving 99.76% accuracy, with only 6 misclassified images out of 2520. This result surpasses the current state of the art result of 99.72% reported in [13], with inputs that have a quarter of the pixels and a model with a lighter architecture. The best epoch achieved an accuracy of 99.84%.

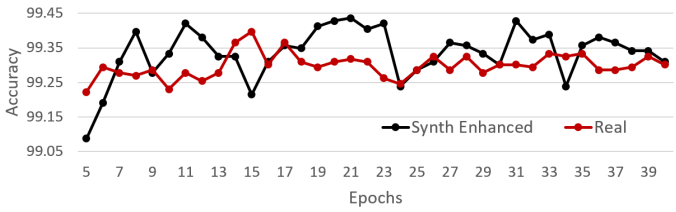

Fig. 17. BTSC - accuracy for the test set. Comparison between models trained with the enhanced synthetic training sets vs models trained with real data. Source [10].

GTSRB: In this dataset misclassifications are spread over several classes. Examining the templates from the test set we ended adding the templates in Figure 18.

These templates represent variations that are missing from the training set. Technically, the third sign in Figure 18 is not a traffic sign, but it is a common combination present in both the training and test sets.

Fig. 18. Added templates for GTSRB

Repeating the previous tests with the unleashed dataset we noticed a decrease in the number of misclassified samples in classes where templates were added without relevant adversarial side effects. Table 11 provides a comparison between the previously obtained accuracies and the results for the unleashed datasets, showing a significant improvement in all synthetic models.

Table 11. Results for unleashed GTSRB datasets. Source [10].

	Prev Results	Unleashed
SES	99.24 ± 0.12	$\mathbf{99.41 \pm 0.08}$
SER	99.34 ± 0.09	$\mathbf{99.40 \pm 0.10}$
SJS	99.25 ± 0.09	$\mathbf{99.50 \pm 0.09}$
SJR	99.49 ± 0.04	$\mathbf{99.57 \pm 0.04}$

As previously done for BTSC, we also tested merging real data with one of the SES and SJR models. The accuracy obtained was 99.80% for Real + SES,

and 99.79% for Real + SJR. These are marginally below the state of the art results reported in [4] (99.81%). As noted previously in Section 5.1, our models are much lighter than Haloi's due to the difference in image input size (32×32 vs. 128×128).

6 Conclusion

A new proposal for synthetic data generation for traffic sign repositories was presented. Two operators were introduced: Perlin and Confetti noise. While Perlin noise provides a clear advantage Confetti noise still requires some further study to evaluate its usefulness. We also explored taking advantage of the brightness distribution of real data when available. While using real data distributions for some variable can provide an increase in accuracy in some scenarios it also may introduce undesirable bias to the dataset. Finally, we also explored solid colour vs. real imagery backgrounds. Again, while real background imagery may provide some benefits in some scenarios, it is also a potential source of bias. A noteworthy point is that, unlike previous works, our synthetic samples are clearly not realistic, suggesting either that the level of realism achieved in previous works is not enough, or that pursuing realism may not be the best option.

We showed that our synthetic datasets clearly surpass any previous attempts regarding the accuracy obtained, with the worst results falling within 0.5% of the results obtained with real data with the same model architecture. In conventional tests such as merging and ensembling we surpassed state of the art results in three public traffic sign datasets, clearly showing the potential of synthetic data.

While surpassing state of the art results is always a rewarding, we strongly believe that the most relevant results are those when synthetic data was unleashed, and the cross-testing experiments.

Unleashing the synthetic dataset provided excellent results at a minimal cost. All that is required is to retrain the model. The set of templates for unleashed datasets is the natural set of templates when no real data is available. In this scenario all templates for older versions should be collected to achieve the full potential of a trained model.

Cross-testing was presented as in inter-country test, but, as discussed, it can also be seen as a broader national test with a very diverse test set. We obtained an interesting result with the most agnostic dataset, without real backgrounds and without using the brightness distribution of real data, clearly surpassing the results obtained with real data. This is a clear indication of the generalization potential when using synthetic data.

We strongly believe that our work shows the potential of synthetic data in the domain of traffic sign repositories. Nevertheless, there is still work to be done. The pipeline can be fine tuned, and new operators can be explored to deal with shadows, rain, fog, and night time, amongst other common occurrences in traffic sign imagery.

Acknowledgements

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the RD Units Project Scope: UIDB/00319/2020

References

1. Araar, O., Amamra, A., Abdeldaim, A., Vitanov, I.: Traffic sign recognition using a synthetic data training approach. International Journal on Artificial Intelligence Tools **29** (05 2020). https://doi.org/10.1142/S021821302050013X
2. Grigorescu, C., Petkov, N.: Distance sets for shape filters and shape recognition. IEEE Transactions on Image Processing **12**(10), 1274–1286 (2003). https://doi.org/10.1109/TIP.2003.816010
3. Gámez Serna, C., Ruichek, Y.: Classification of traffic signs: The european dataset. IEEE Access **6**, 78136–78148 (2018). https://doi.org/10.1109/ACCESS.2018.2884826
4. Haloi, M.: Traffic sign classification using deep inception based convolutional networks. ArXiv **abs/1511.02992** (2015)
5. Horn, D., Houben, S.: Fully automated traffic sign substitution in real-world images for large-scale data augmentation. In: 2020 IEEE Intelligent Vehicles Symposium (IV). pp. 465–471 (2020). https://doi.org/10.1109/IV47402.2020.9304547
6. Horn, D., Janssen, L., Houben, S.: Automated selection of high-quality synthetic images for data-driven machine learning: A study on traffic signs. In: 2021 IEEE Intelligent Vehicles Symposium (IV). pp. 832–837 (2021). https://doi.org/10.1109/IV48863.2021.9575337
7. Jurišić, F., Filković, I., Kalafatić, Z.: Multiple-dataset traffic sign classification with onecnn. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR). pp. 614–618 (Nov 2015). https://doi.org/10.1109/ACPR.2015.7486576
8. Larsson, F., Felsberg, M.: Using fourier descriptors and spatial models for traffic sign recognition. In: Heyden, A., Kahl, F. (eds.) Image Analysis. pp. 238–249. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
9. Liu, Y.T., Chen, R.C., Dewi, C.: Generate realistic traffic sign image using deep convolutional generative adversarial networks. In: 2021 IEEE Conference on Dependable and Secure Computing (DSC). pp. 1–6 (2021). https://doi.org/10.1109/DSC49826.2021.9346266
10. Lopes da Silva., D., Ramires Fernandes., A.: Bridging the gap between real and synthetic traffic sign repositories. In: Proceedings of the 3rd International Conference on Deep Learning Theory and Applications - DeLTA,. pp. 44–54. INSTICC, SciTePress (2022). https://doi.org/10.5220/0011301100003277
11. Luo, H., Kong, Q., Wu, F.: Traffic sign image synthesis with generative adversarial networks. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 2540–2545 (2018)
12. Luo, J., Wang, Z.: A low latency traffic sign detection model with an automatic data labeling pipeline. Neural Computing and Applications pp. 1–14 (2022)
13. Mahmoud, M.A.B., Guo, P.: A novel method for traffic sign recognition based on dcgan and mlp with pilae algorithm. IEEE Access **7**, 74602–74611 (2019). https://doi.org/10.1109/ACCESS.2019.2919125

14. Paparoditis, N., Papelard, J.P., Cannelle, B., Devaux, A., Soheilian, B., David, N., Houzay, E.: Stereopolis II: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology. Revue Française de Photogrammétrie et de Télédétection (200), 69–79 (apr 2014). https://doi.org/10.52638/rfpt.2012.63,
15. Perlin, K.: An image synthesizer. SIGGRAPH Comput. Graph. **19**(3), 287–296 (Jul 1985). https://doi.org/10.1145/325165.325247, https://doi.org/10.1145/325165.325247
16. Saha, S., Kamran, S.A., Sabbir, A.S.: Total recall: Understanding traffic signs using deep hierarchical convolutional neural networks. CoRR **abs/1808.10524** (2018), http://arxiv.org/abs/1808.10524
17. Spata, D., Horn, D., Houben, S.: Generation of natural traffic sign images using domain translation with cycle-consistent generative adversarial networks. In: 2019 IEEE Intelligent Vehicles Symposium (IV). pp. 702–708 (2019)
18. Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. Neural Networks (0), – (2012). https://doi.org/10.1016/j.neunet.2012.02.016, http://www.sciencedirect.com/science/article/pii/S0893608012000457
19. Stergiou, A., Kalliatakis, G., Chrysoulas, C.: Traffic sign recognition based on synthesised training data. Big Data and Cognitive Computing **2**(3) (2018). https://doi.org/10.3390/bdcc2030019, http://www.mdpi.com/2504-2289/2/3/19
20. Timofte, R., Zimmermann, K., Gool, L.V.: Multi-view traffic sign detection, recognition, and 3d localisation. In: 2009 Workshop on Applications of Computer Vision (WACV). pp. 1–8 (Dec 2009). https://doi.org/10.1109/WACV.2009.5403121
21. Youssef, A., Albani, D., Nardi, D., Bloisi, D.D.: Fast traffic sign recognition using color segmentation and deep convolutional networks. In: Blanc-Talon, J., Distante, C., Philips, W., Popescu, D., Scheunders, P. (eds.) Advanced Concepts for Intelligent Vision Systems. pp. 205–216. Springer International Publishing, Cham (2016)
22. Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., Hu, S.: Traffic-sign detection and classification in the wild. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2110–2118 (2016)
23. Šegvic, S., Brkić, K., Kalafatić, Z., Stanisavljević, V., Ševrović, M., Budimir, D., Dadić, I.: A computer vision assisted geoinformation inventory for traffic infrastructure. In: 13th International IEEE Conference on Intelligent Transportation Systems. pp. 66–73 (Sep 2010). https://doi.org/10.1109/ITSC.2010.5624979