27^{th} Computer Vision Winter Workshop
Terme Olimia, Slovenia, February 14–16, 2024

Weather-Condition Style Transfer Evaluation for Dataset Augmentation

Emir Mujić, Janez Perš
University of Ljubljana, Faculty of Electrical Engineering
Tržaška 25, 1000 Ljubljana
janez.pers@fe.uni-lj.si, em4593@student.uni-lj.si

Darko Štern
AVL List GmbH
Hans-List-Platz 1, 8020 Graz
darko.stern@avl.com

Abstract.

In this paper, we introduce a framework for evaluating style transfer methods that simulate desired target weather conditions from source images, acquired in fair weather. The resulting images can be used for targeted augmentation of datasets geared toward object detection. Our approach diverges from traditional measures that focus on human perception only, and importantly, does not rely on annotated datasets. Instead, we operate on statistical distribution of outcomes of the inference process (in our case, object detections).

The proposed evaluation measure effectively penalizes methods that preserve features and consistencies in object detection, and awards those, which generate challenging cases more similar to the target style. This is counteracted by the requirements that the generated images remain similar to the images acquired in target weather conditions.

This shift enables a more relevant and computationally practical assessment of style transfer techniques in the context of weather condition generation. By reducing the dependency on annotated datasets, our methodology offers a more streamlined and accessible approach to evaluation.

1. Introduction

In the rapidly evolving field of computer vision, the enhancement and adaptation of datasets through style transfer, particularly under varying weather conditions, is of paramount importance [20]. The research, presented in this paper, is primarily motivated by the desire to improve the performance of advanced driver assistance systems (ADAS) through targeted learning of hard examples from challenging weather conditions. Importantly, the method itself does not directly enhance ADAS; rather, it provides a quantitative measure of quality of synthetic samples in context of ADAS tasks, which could poten-

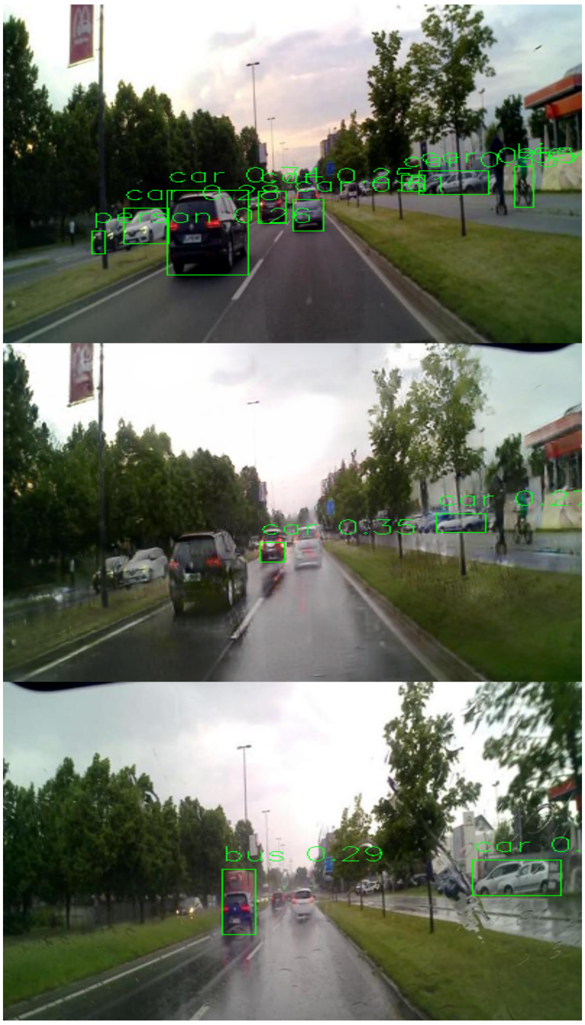

Figure 1: Successful style transfer mimicking real rain's impact on vehicle detection. Top: input image; middle: simulated rainy image; bottom: style reference (rainy weather). Vehicle detection using YOLOv8 [11] shows significant performance drop from top (clear) to middle (rainy) image.

tially be used to refine and improve already existing algorithms. The collection of images under adverse weather is often hindered by their rarity, seasonal dependence, and increased risk to vehicles. Thus, synthesizing weather conditions in existing images recorded in fair weather, that not only look realistic but also can correctly challenge or impair the performance of computer vision algorithms, is a key point of interest in the modern automotive industry. Assessing the quality of style-transfer model is a difficult task and still remains an open issue [23].

In this paper, we propose a novel evaluation measure that quantitatively shows how successful style transfer is by visual quality of generated images while keeping the statistics of object detection similar to the targeted (conditional) style. Our proposed method should help object detection by determining if a dataset of images is challenging enough and at the same time has similar core characteristics to target style, for it to be used to improve object detection in specific weather conditions. The approach of this paper is tailored towards ADAS applications. Example result of successful transfer where detection is hindered by the added style in the same way it would be by the style of a rainy image is shown in Figure 1. Additionally, if we are in possession of annotated fair weather images, then the style transfer preserves the annotations since they show the same scene. This significantly simplifies the process of obtaining examples difficult for object detection since we can analyze and extract those examples and based on the statistical difference of annotations and detections.

2. Related Work

We split our related work section into three parts: in the first (i) we look into type of generative models we use in our paper, the second (ii) covers the work on the most popular evaluation measures and the third (iii) the practical computer vision applications we will focus on.

Generative models. Ever since the introduction of generative adversarial networks (GANs) [6], the idea of translating images from one domain to another has been a keen topic of research. Early works in this were done by Isola *et al.* in [9] where they showed it was possible to preform image-to-image (I2I) translation using conditional GANs. For evaluation measures they used Amazon Mechanical Turk (AMT) and "semantic interpretability" [24] of the generated images to see how well can an off-the-shelf image semantic segmentation network such as [15], segment generated images. Early research in I2I didn't consider unpaired translation, the issue of not having image pairs from two domains, since it focused on automatic segmentation, coloring and label→image tasks. Amongst the first to solve this problem, Park *et al.* introduced cycle consistency to GANs [25], creating CycleGAN. Their methods for evaluation are the same as in [9]. More recently, work by Park *et al.* in [17] and Hu *et al.* [8] created unpaired I2I translation based on contrastive learning, reaching current state-of-the-art performance in style transfer tasks. In our tasks we look at weather condition translation and first to create a bespoke network for this are Li *et al.* [12]. They employed attention and segmentation modules to the generator. More recently Piazzati *et al.* in [18], found that using physics-informed network to guide the effects of weather proved to be state-of-the-art in weather translation.

Evaluation measures. The first, and one of the most used, measures for quantitative score of GANs is Inception Score (IS) [21]. It uses a deep network Inception v3 [22] pre-trained on ImageNet [5] to extract relevant features from generated images and calculates the average KL-Divergence between the conditional label distribution and generated samples distribution. It shows correlation with human scoring on CIFAR-10 dataset. Barratt *et al.* in [1] showed that IS has issues with both theory and use in practice. More modern measure is Fréchet inception distance (FID), introduced by Heusel *et al.* in [7]. Similar to IS, FID uses Inception v3 network pre-trained on ImageNet, but now the generated and real samples are embedded to tensors before calculating the statistical distance, in this case 2-Wasserstein, between them. Chong *et al.* in [4] prove that both IS and FID are functions of the generator and the number of samples, therefore we can't fairly compare two generators and even the same generators evaluated on different number of images. They propose a new measure of effectively unbiased FID and IS called FID_∞ and IS_∞ respectfully. Besides the bias issue FID also assumes a Gaussian distribution of samples which is not necessarily true; to solve this, Binkowski *et al.* [3] introduce kernel inception distance (KID) where the kernel can be customized to accommodate different tensor distributions. Betzalel *et al.* in [2] state that the same problems that IS has of Inception v3 being trained on ImageNet are present in FID as well, and suggest using CLIP [19] basis instead of the Inception model. They also state that evaluation measures in general might benefit from multiple measures such as FID$_\infty$ + KID.

Applications. One of the practical applications of computer vision is in the world of advanced driver assistance systems (ADAS). Nidamanuri *et al.* in [16] state that the camera sensor is useful for multiple functions such as object detection, blind spot monitoring, parking assist, lane keeping and traffic sign recognition, with moderate accuracy. Liu *et al.* [14] state that classical object detectors often fail when faced with adverse weather conditions.

Our research is similar to Li *et al.* [12] where they employ a weather classifier trained on real weather images, to check if images generated by their model are good enough to fool the classifier into giving the image a label of the target condition.

3. Methods

Our framework is comprised of multiple elements. First is the ADAS algorithm (e.g. object detection) that we wish to improve by additional learning on target weather examples. The second component is a trained generator of target weather conditions, which takes non-annotated fair-weather image and transforms it into target-weather image. The third component is an image quality assessment measure, which guarantees the similarity of generated images to the real world target weather images. The fourth and critical component of our proposed framework is evaluation measure of performance degradation of the chosen ADAS algorithm that *does not rely on image annotations.*

3.1. Object detection with YOLOv8

Since we don't have access to actual ADAS algorithms used on vehicles, we believe that YOLOv8 as an example of a state-of-the-art object detector is a good approximation. YOLOv8 is a deep neural network [11], developed as an upgrade on YOLOv5 [10] architecture in both speed and performance. In this paper we use it as a default object detector. It's trained on COCO Dataset [13] with 272 categories. For our use case (simulation of ADAS), many of these categories are not interesting, hence we filter all results and look only for a few categories. In no particular order these are: car, truck, bus, train, person and bicycle. COCO has additional categories associated with driving such as van and motorcycle, but we treat these all these as cars in case of motorcycles and truck for vans. These particular categories were chosen based on the most common vehicles found in our custom driving dataset. As detection result, YOLOv8 returns a bounding box and a label (category). We use this to compare to the ground truth. Ground truth was done by manually labeling and drawing bounding boxes and object categories same ones as taken in YOLO, on test set of 92 images from both dry and rainy weather conditions.

3.2. Image quality assessment with FID

Despite its issues with bias, most generative models are evaluated using the FID measure for image assessment. A few methods have been developed after FID, however it remains the most used measure in practice. This is due to the fact that it's relatively easy to calculate and there are numerous implementations. To calculate it we run inference on a pre-trained Inception v3 model and calculate the 2-Wasserstein distance from the $N \times 2048$ dimensional vector we get as a result from the inference. N is the number of images from each label (real or generated). FID assumes a Gaussian distribution on the feature vector with mean μ and covariance matrix Σ. FID score is calculated as the square of the 2-Wasserstein distance (1) between tensors X and Y:

$$\begin{aligned}\text{FID}_{X,Y} = \|\mu_X - \mu_Y\|^2 + \\ + \operatorname{tr}\left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X\Sigma_Y}\right). \quad (1)\end{aligned}$$

Since FID assumes Gaussian distribution, μ_x and μ_Y are the means of tensors X and Y, and Σ_X and Σ_X their respective covariance matricies. FID is known to be biased ([2] [4]), however it is the most used measure of generative model quality, even in tasks such as conditioned style-transfer ([8] [12] [18]). Despite objectively better measures existing such as FID_∞ [4], KID [3] and CLIP [19], in this paper we decided on using FID due to its ease of implementation and popularity. We will discuss the possible issues with this choice in section 4.

3.3. Quality of detection measure

To measure quality of detections without annotated images, we rely on statistics of results from YOLOv8, on the entire test set. The assumption underpinning this approach is that *statistically, YOLOv8 detections should have similar distribution shape to the actual annotations on the same driving route, regardless of the weather.* This statement will be backed up in section 4.1. Detection is run on all images included in the test: input image, conditional style image and generated image. For consistency with our use case we refer to these as “dry”, “rainy” and “generated” (images generated from dry to have the style of rainy), respectively. From detection results on an image we take two values, the horizontal and vertical coordinate of the bounding box and the size of the bounding box in pixels in both horizontal and vertical directions. From these we create 2D discrete histograms which, when computed for the whole set of a single image style, show us statistical feature that we want to emphasize during translation. In our case this is object detection. We can compare histograms using an adequate measure like Bhattacharyya distance. Bhattacharyya distance tells us how “far“ two discrete data distributions are one to another. Before computing, all histograms are normalized to a range $[0, 1]$. This makes our method invariant to actual number of detected objects and focuses only on positions. Currently, this is an advantage since we expect the number of vehicles for example, to be different during data collection in different weather conditions. However, with careful data acquisition (ensuring that the streets are equally busy) the absolute frequencies could be another feature to include. Computing the Bhattacharyya distance is quite straightforward, in this case as we follow equation (2) where P and Q are discrete data distributions and p_i and q_i are their respective bins:

$$d_B(P,Q) = -ln\left(\sum_{i=1}^{n}\sqrt{p_i q_i}\right) \tag{2}$$

For computing equation (2) on 2D histograms, we simply transform a $n \times m$ matrix into a $1 \times (n \times m)$ vector. This approach, along with equation (2), measures the effectiveness of YOLOv8 in object detection across various images. A requirement for using this measure is that dry and rainy images need to contain similar image content across the dataset, but for tasks such as style-transfer this is fulfilled in most cases.

3.4. Combining the measures

As a result of FID we get a single number that should indicate the “visual distance“ between two images consistent with human evaluation. With detection this is more complicated and we propose a method described in section 3.3.

We compute Bhattacharyya distances between histograms for both position and size to get a sense how close the distributions of detected objects are for dry, rainy and generated images. Normalizing all of the calculated values for both FID and Bhattacharyya distance so the smallest is 0 and largest 1, and can combine them into a weighted sum to give us a score shown in equation (3):

$$\begin{aligned} s(X,Y) = -\alpha \cdot FID_{X,Y} + \\ + \beta \cdot d_B\left(H_{size}(X), H_{size}(Y)\right) + \\ + \gamma \cdot d_B\left(H_{position}(X), H_{position}(Y)\right) \end{aligned} \tag{3}$$

where $s(X, Y)$ is the measure score between a set of images X and Y, $FID(X, Y)$ is the FID score between those two sets, $H_{size}(X)$, $H_{size}(Y)$ and $H_{position}(X)$, $H_{position}(Y)$ are the notations for histograms computed for detection sizes and positions of detection for a set of images X and Y, $d_B(H(X), H(Y))$ is the Bhattacharyya distance between sets of histograms. Parameters α, β and γ are hyperparameter weights ($\alpha, \beta, \gamma \geq 0$) for FID, $d_B\left(H_{size}(X), H_{size}(Y)\right)$ and $d_B\left(H_{position}(X), H_{position}(Y)\right))$ respectively, that describe the contribution to the total measure score.

3.5. Dry-to-rainy translation: QS-Attn Model

In this paper, we utilized the query-selected attention (QS-Attn) model [8] for I2I translation tasks. QS-Attn enhances contrastive learning [17] by selectively focusing on significant anchor points within images. This model employs an attention mechanism that prioritizes important queries in the source domain, creating a condensed attention matrix. This matrix is pivotal in routing features across both source and target domains, ensuring that relational structures from the source are retained in the translated images.

4. Experiments

4.1. Dataset and training

For our dataset, we recorded 1242 images on a route in dry and the same amount in rainy weather. This makes our dataset weakly-paired, meaning pairs of images do not exist since the recording environment (the road) is dynamic, but images are still similar enough to be considered “location pairs“. Examples of this are shown in Figure 2. Recording the dataset like this, twice on the same route with the camera fixed in the same place on the windscreen, ties in with the discussion of weather the statistics of detections are the same. These statistics are very route specific and we choose a route that has main streets with a good flow of traffic as well as residential areas with less active traffic and more passive traffic such as parked vehicles to try and cover what most vehicles see in day to day city driving. Of the complete dataset, 1140 images are training images, 10 validation images and 92 test images. Images in the dataset cover urban driving scenes in central European cities, in dry conditions and rain such is shown in Figure 2. All images are resized from original 3840×2160 pixels resolution, to 400×400 pixels to accommodate the model input and to make it possible to train the model on a single Nvidia RTX3090 GPU.

For our model, we used an official implementation[1] of GAN described in section 3.5. Training hyperparameters are default, except for “QS_Mode“ set to *global*, “crop_size“ and “load_size“ hyperparameters are set to 400 to accommodate hardware limitations. Model was trained for 400 total epochs, of which the first 200 are with the default learning rate (“n_epochs“ hyperparameter) and the latter 200 with linear learning rate decay (“n_epochs_decay“ hyperparameter).

4.2. YOLOv8 detection on our data

To benchmark YOLOv8, we annotated our test data with bounding boxes and labels for objects of interest. Annotation statistics are shown in Table 1, the numbers represent the number of bounding boxes for that label in absolute value and as a percentage of all labels. Results on 92 test images per weather condition are shown in Table 2. Looking at normalized histograms in Figure 3 of detections for both dry and rainy conditions, we can get a sense how well YOLOv8 does in a more practical sense. Since histograms in Figure 3 and 4 are normalized to a range $[0, 1]$, the shape of the distribution is for now much more relevant for us than the values at any particular point. Figures 3 and 4 are the distributions we are basing our evaluation on. We can see that they are similar in shape. This results needs to be additionally verified with more annotated images for both dry and rainy conditions.

4.3. Evaluation procedure

Our evaluation relies on the fact that during training, model creates more and more realistic rainy images as epochs tend towards the final one. We can sample the training weights at certain points during training to obtain a sub-optimal model and run inference with test image set to obtain intermediate results. We then follow method described in section

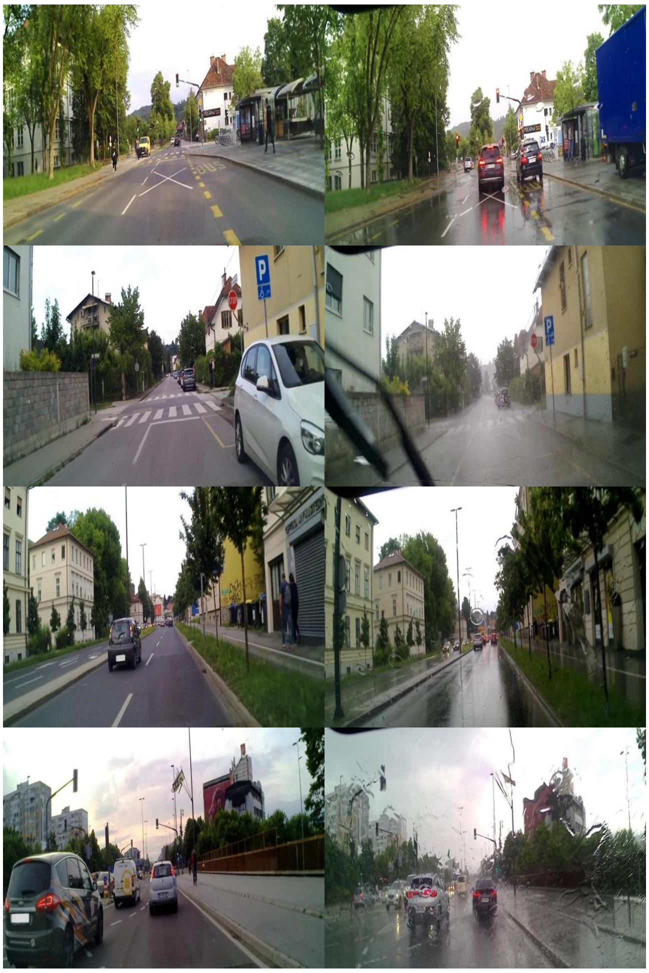

Figure 2: Sampled images from our weakly-paired dataset depicting scenes of urban driving

	Dry	Rain
Car	364 (78.45%)	322 (82.11%)
Person	69 (14.78%)	28 (7.05%)
Bicycle	13 (2.8%)	17 (4.28%)
Bus	9 (1.94%)	10 (2.52%)
Truck	9 (1.94%)	20 (5.04%)

Table 1: Annotation results for our dataset

	Dry	Rain
Precision	0.734	0.377
Recall	0.492	0.153
F1 Score	0.589	0.218

Table 2: YOLOv8 benchmark results for our dataset

[1] https://github.com/sapphire497/query-selected-attention

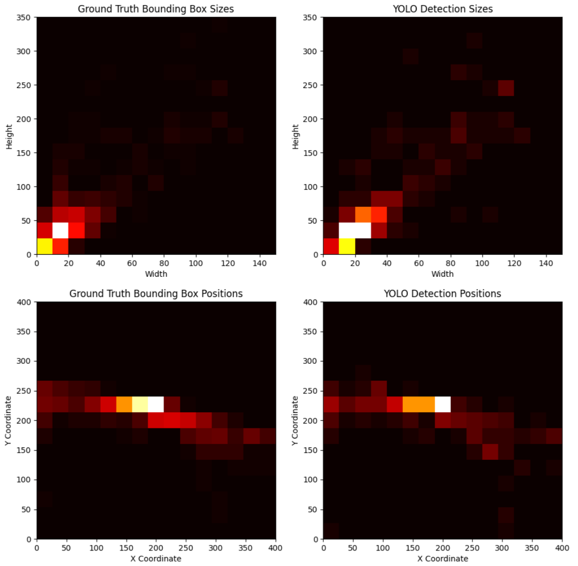

Figure 3: 2D histogram comparison of detections on dry images.

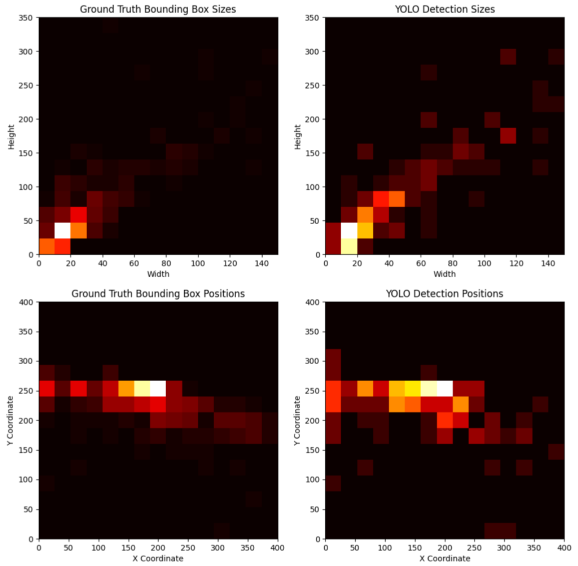

Figure 4: 2D histogram of detections on rainy images.

3.4, and test our combined evaluation measure. One thing we need to make sure is to correctly sample results from dry and generated set. The reason for this is, because of the nature of style-transfer tasks, there is a possibility of high correlation between detection scores from these dry and generated images since they depict the exact same scene only in different weather. To get an accurate measure of performance over different samples, we take every even numbered image from the test set of dry samples and odd numbered sample from the generated set, to make sure the results are correctly calculated. Rainy images contain different scenes so sampling can be done either way. From this we create histograms for all sets of images: dry, rainy and generated. Example of histograms from the last training epoch is shown in Figure 5. Normalization to range $[0, 1]$ is done and finally we compute Bhattacharyya distance according to eq. (2) between dry and generated, and dry and rainy histograms. FID is then computed between rainy and fake images to give us a FID score. We note we used FID primarily for its ease of computation, implementations of other measures, such KID and FID$_\infty$, are less common.

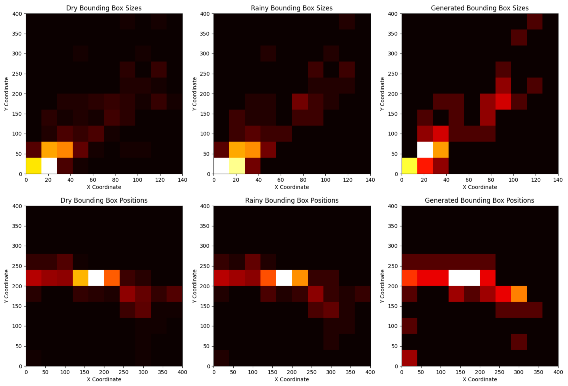

Figure 5: 2D histogram of samples taken on last training epoch

For our tests, following the described method, our measure rewards generated samples that have a similar (according to eq. (2)) histogram distribution to rainy samples, and low FID score between rainy and generated samples. For clarity, in our experiments we compared dry to rainy and dry to generated samples to show that the measure for generated images goes from being more similar to dry towards being more similar to rainy. Theoretically the best score a model can achieve is 2. This is because we need to make sure all values are scaled to the same size, therefore we normalize both FID and Bhattacharyya distances to $[0, 1]$. Setting all of the weights in eq. (3) to $\alpha, \beta, \gamma = 1$ gives us a maximum score of 2.

4.4. Results

We sample the model at every 5th epoch and evaluate the results according to our method. We first look at graphs for all influential measures over sampled epochs separately and not normalized. In Figure 6, we can see that the measure for similarity of histograms between dry and fake samples drifts quite rapidly from values closer to dry vs. dry, towards dry

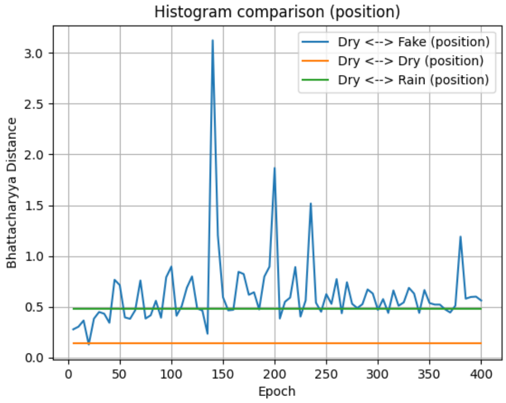

Figure 6: Bhattacharyya distance comparison for positional histograms over training epochs

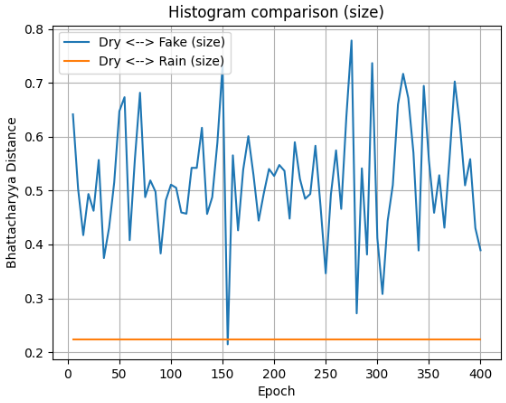

Figure 7: Bhattacharyya distance comparison for size histograms over training epochs

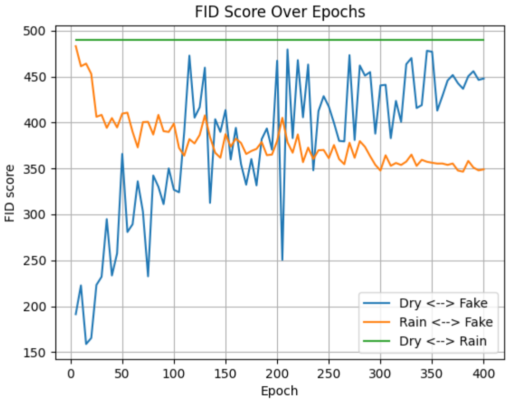

Figure 8: FID score over training epochs

vs. rainy quite rapidly at the beginning of the training process.

Spikes in the Figure 6 are due to the random nature of training generative models and we can interpret it as follows: the model suddenly learns how to represent a new feature from the rainy set such as windscreen wipers found on training images, so suddenly on all generated images form a certain epoch there are simulated wipers represented as a back line across the screen. Example of this is in Figure 10. These interfere with possible detections and make the histogram of generated samples dissimilar to that of dry samples. From the section 4.3 we know our measure has a theoretical best value so going over this is not wanted, just as much as not reaching this value in the first place. Spikes can tell us that something drastically changed during training, and needs to be visually examined. Windscreen wipers are actually *a valid distortion* on our images, if the camera is placed in a way that is occasionally covered with wipers and we can use this epoch to obtain difficult training samples that simulate wipers *if* that is our goal.

Looking at size comparisons, things are more difficult to assess. Graph showing comparisons over epochs is shown in Figure 7. Detection sizes are noisy over epochs and general trend is difficult to see. This is due to the fact that over different epochs images go through various phases of added artifacts and effects by the model, making the detections that are present, inconsistent.

Analyzing the graph of FID score over epochs in Figure 8, we get a sense how does well does the translation work. We can see that the score comparing dry and generated samples trends up towards the value of dry vs. rainy and, more relevant for us, the score comparing rainy and generated trends down over epochs. This, to a certain degree, ensures us that the style-transfer seems to be working correctly.

Now normalizing these values and summing them according to equation (3), gives us our measure how good the style transfer is. Measure is shown in Figure 9.

We also fitted a trend line using least squares to the results to get a better trend estimate. We can see that the measure value goes up with training epochs, reassuring us the model is doing style-transfer correctly according to both FID (as proxy for human perception) and at the same time making the images challenging for an object detector in a similar direction to that of a rainy image. Different weights would emphasize different aspects of style-transfer and therefore give us different looking graphs for a model, depending on what component is most important for any given task. Example image sampled at different epochs where our measure shows higher values are shown in Figure 10.

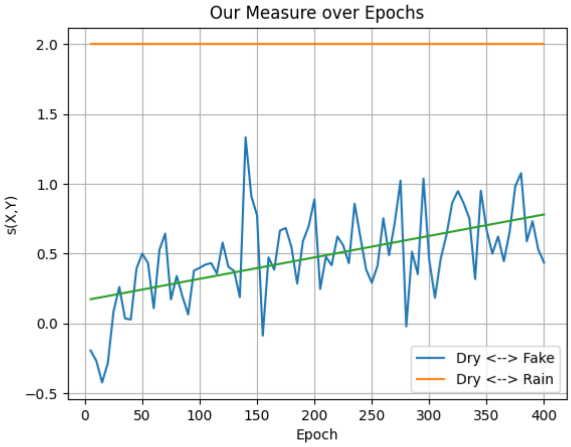

Figure 9: Our model evaluation measure over training epochs. Hyperparameters are set to values $\alpha, \beta, \gamma = 1$. $s(X, Y)$ represents the score between two sets of images.

One important fact to mention is that all of the presented results were done on our test dataset that has 92 images from each weather condition. This means that looking at raw values of the measure is not reliable enough since we can't with certainty state that the results of even Bhattacharyya distance are unbiased, let alone FID. Therefore, for current results we propose looking at only the trend is it rising or falling and based on that determine is the model working in the wanted "direction".

5. Conclusion

This study developed a framework for evaluating style transfer in weather-conditioned image generation, addressing the challenge of maintaining key features for object detection while accurately simulating weather conditions. This has implications for dataset augmentation in fields like ADAS. Future goals include testing with a larger dataset, both for training and evaluation, and further research on the statistical consistency of the proposed measure. Plans also include adapting this measure as a loss function for training style transfer models for specific computer vision tasks. Additionally, alternatives to FID and other histogram distances for image similarity will be explored.

Acknowledgement

This work was financed by the Slovenian Research Agency (ARIS), research program [P2-0095], and research project [J2-2506].

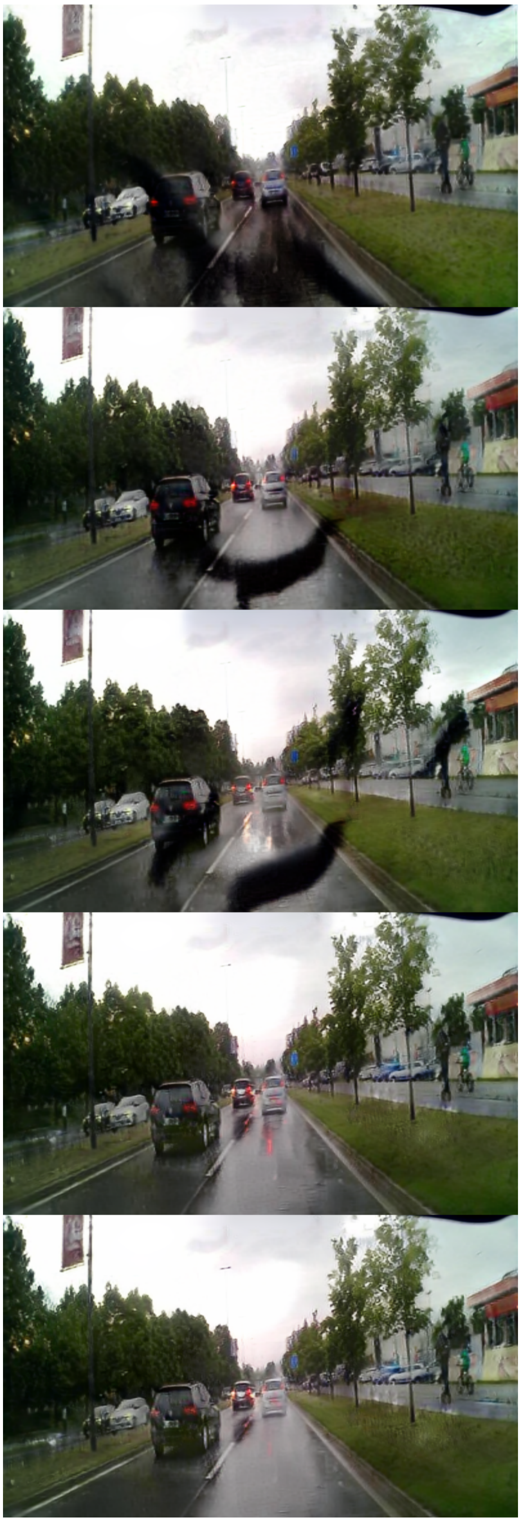

Figure 10: Example generated images by QS-Attn [8] from the test dataset on epoch numbers 130, 230, 270, 360 (roughly corresponding to local maxima of the proposed measure) and 400 (final epoch), in order from top to bottom. In first three, an attempt to add wipers is clearly visible.

References

[1] S. Barratt and R. Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018. 2

[2] E. Betzalel, C. Penso, A. Navon, and E. Fetaya. A study on the evaluation of generative models. *arXiv preprint arXiv:2206.10935*, 2022. 2, 3

[3] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 2, 3

[4] M. J. Chong and D. Forsyth. Effectively unbiased fid and inception score and where to find them. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6070–6079, 2020. 2, 3

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[7] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 2

[8] X. Hu, X. Zhou, Q. Huang, Z. Shi, L. Sun, and Q. Li. Qs-attn: Query-selected attention for contrastive learning in i2i translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18291–18300, 2022. 2, 3, 4, 8

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[10] G. Jocher. Ultralytics yolov5. `https://github.com/ultralytics/yolov5`, 2020. 3

[11] G. Jocher, A. Chaurasia, and J. Qiu. YOLO by Ultralytics. `https://github.com/ultralytics/ultralytics`, Jan. 2023. 1, 3

[12] X. Li, K. Kou, and B. Zhao. Weather gan: Multi-domain weather translation using generative adversarial networks. *arXiv preprint arXiv:2103.05422*, 2021. 2, 3

[13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 3

[14] W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang. Image-adaptive yolo for object detection in adverse weather conditions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1792–1800, 2022. 3

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2

[16] J. Nidamanuri, C. Nibhanupudi, R. Assfalg, and H. Venkataraman. A progressive review: Emerging technologies for adas driven solutions. *IEEE Transactions on Intelligent Vehicles*, 7(2):326–341, 2021. 3

[17] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu. Contrastive learning for unpaired image-to-image translation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 319–345. Springer, 2020. 2, 4

[18] F. Pizzati, P. Cerri, and R. de Charette. Physics-informed guided disentanglement in generative networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2, 3

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[20] C.-G. Roh, J. Kim, and I.-J. Im. Analysis of impact of rain conditions on adas. *Sensors*, 20(23):6720, 2020. 1

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 2

[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 2

[23] Z. Wang, L. Zhao, H. Chen, Z. Zuo, A. Li, W. Xing, and D. Lu. Evaluate and improve the quality of neural style transfer. *Computer Vision and Image Understanding*, 207:103203, 2021. 2

[24] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016. 2

[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2