University of British Columbia

Advanced Machine Learning Tools for Engineers

EECE 571T

Group 20 – Generative Adversarial Network Based Image Deraining With Contrastive Learning

---

Author: Zhongze Chen
Author: Han Wang

April 2023

Abstract

Different rain conditions like rain streaks and accumulations cause severe occlusion and low visibility, which is a big challenge while restoring the details making single-image de-raining. Single-image de-raining aims to remove rain degradations from images, important in many practical applications like surveillance security, transportation and remote sensing. In this paper, we proposed a supervised, one-sided translation, which is generative-adversarial-network-based contrastive learning for unpaired image-to-Image translation, mapping images from the source rainy images domain to target clean images domain to remove rain degradations. We also preprocessed the challenge GT-rain dataset and reorganized the data so that it fits our model. By using both qualitative and quantitative measures, the significant effects of single-image de-draining can be observed. To optimize the model, various evaluation metrics are employed, and the loss function is adjusted accordingly to determine the best parameters.

[Our code is available at: https://github.com/czz1997/EECE571T]

1) The Goal of the Report

The quality of outdoor photographs can be impacted by various weather from rain to snow. Primarily due to an increase in the frequency and intensity of extreme precipitation events in Vancouver [1], rain has a greater impact on image quality than other weather conditions due to its physical nature and affects both the foreground and background details in the image, hence, the process of restoring diffuse reflection caused by rain patterns from a single image is challenging, affecting some fields from automatic driving to traffic monitoring [2]. We proposed a generative adversarial network (GAN) [3] based model with contrastive learning to remove degradations induced by rain streaks and accumulation from the rainy image. The model will be trained and refined using pre-determined loss functions, including a reconstruction loss and Structure Similarity Index Measure (SSIM) [5] loss until the optimal configurations are attained using the GT-RAIN dataset [4] collected from the real world as our training and testing dataset. The final model will be evaluated using the Fréchet inception distance (FID), peak-signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM), to minimize a rain-invariant loss between rainy and clean images.

2) Previous work done

A lot of attention is focused on eliminating the structural rain patterns from real-time rain photos [6]. Prior-based de-raining techniques are not effective in delivering optimal results when images are impacted by various shapes and orientations of rainfall [7,8]. Numerous de-raining networks based on deep learning have been suggested in recent years. It's worth noting that many current techniques are supervised, and they are trained on synthetic datasets using pairs of rainy and clean images (ground truth) in a supervised manner. Supervised models typically yield favourable de-raining results by establishing a rigid relationship between the rainy image and its ground truth. The improved performance is due to the fact that the models have already been trained on the rain streaks in the rainy images. This means that the models already know the rain streaks in the input rain images. As a result, supervised methods typically have excellent generalization capabilities when using paired data. However, for real-world rainy images without ground truth (i.e., unpaired data), many existing supervised de-raining models may fail because of the irregular and non-uniform rain streaks that are frequently encountered in real-world rainy scenes [9-11].
It is important to mention that most real-world rainy images lack ground truth, making them unsuitable for use by supervised networks. In such scenarios, unsupervised or semi-supervised methods are typically employed, as they can perform de-raining without the need for paired images or with only a small number of paired images. However, due to the absence of prior knowledge, research on semi-supervised and unsupervised networks for the task of SID has progressed much more slowly than that of supervised networks. This is primarily because: (1) the rain streaks in real-world rainy images have highly irregular shapes and directions, such as streaks, drops, and veils. Even for synthetic datasets, it remains challenging to accurately map the relationship between rainy and rain-free images without strict pairwise constraints; (2) even with existing synthetic datasets such as Rain800 [12] and Rain1400 [13], fully supervised methods still cannot achieve optimal recovery results, leaving ample room for improvement.
A deep rain streak removal CNN (DeRainSRCNN)-based de-raining framework [14] provides higher PSNR and SSIM values for both synthetic and real-time images compared with other existing de-raining networks but faces some training errors and poor image visual quality for images with high rain patterns.
GANs are deep neural network architectures consisting of two sub-nets that compete against each other in an "adversarial" manner and are effective in generating more realistic images. However, most existing GAN-based models require paired training data, which can be expensive to obtain in practice. To address this issue, an unsupervised GAN called CycleGAN [15] has been proposed, which can use unpaired images. While CycleGAN has been shown to be effective in various low-level tasks, it is still challenging to apply it to solve the rain removal problem because the domain knowledge of rainy and rain-free images is asymmetrical. Specifically, the rainy image contains both the background and rain streaks (or drops), whereas the rain-free image only includes the background. Therefore, directly utilizing CycleGAN for the

task of single image de-raining may result in colour or structure distortion issues, and furthermore, it may not be able to fully recover the image blurs [16], dehaze [17], and de-raining.
DerainCycleGAN [14], which is constructed on the foundation of RR-GAN [18], has integrated an attention-based rain line extractor into CycleGAN, enabling it to utilize both rainy and clean image information and enhancing its rain line extraction performance. The objective of image de-raining is to generate an output that resembles the ground truth image without rain while preserving the structure or content of the input scene. This involves a disentanglement problem, where the content that must be retained across domains is separated from the appearance that requires modification. Generally, an adversarial loss is employed to enforce the desired appearance, while CycleGAN is used to ensure cycle-consistency and preserve content. However, the assumption of a bijection relationship between the two domains made by cycle-consistency can be too rigid and difficult to achieve, especially when images from one domain (i.e., multiple rainy images) have additional information compared to the other domain (i.e., clean images) [19], and the time required for training may increase when regenerating the target image back to the source domain [20]. Despite this, the method still suffers from weaknesses as the rain line is extracted solely from the rainy image, resulting in a decrease in background information [21]. Thus, we will adopt a GAN-based model with contrastive learning that is designed for unpaired image-to-image translation, for improved de-raining performance with less reliance on paired ground truth.

3) Strategy

First, we preprocess the challenge dataset and reorganize the data so that it fits our model. We divide the dataset into two parts, rainy images and clean images (ground truth). We retain the folder by scene structure in each part and put rainy images in the scene folder in the first part and clean images in the scene folder in the second part. For each iteration in training, we randomly select a scene and a rainy image A in that scene as the input for the generator; then randomly select another scene and a clean image B in that scene as the real sample for the discriminator and as negative samples for contrastive learning; finally, we use the corresponding ground truth clean image gt for L1 and SSIM loss. We perform the same data augmentation on each input pair (A, B, gt), including colour jitter, random zoom, random crop and random horizontal flip.

Then, we base our model on an unpaired translation GAN model, CUT. For each iteration, we optimize the discriminator first, then the generator. At the start of the iteration, we feed rainy image A to the generator which translates it to the fake clean image fake_B. To compute the loss for the discriminator, we feed B as real and fake_B as fake to the discriminator to classify and apply GAN loss. The parameters of the discriminator are then updated with the loss. Next, we compute the GAN loss for the generator by feeding fake_B to the discriminator and penalize the generator if the image is detected as fake by the discriminator. Then, we compute the patch-wise, multi-layer contrastive loss by using B to draw negative samples. Finally, we compute the SSIM and L1 difference between fake_B and gt. We apply different weights to GAN loss, PatchNCE loss, SSIM loss and L1 loss to compute the final loss for the generator.

4) What Methods Worked

a. Pseudo-code description:

The pseudocode for the methods used is listed below in Figure 1.

Algorithm 1 GAN-based Image Deraining with Contrastive Learning

for number of training iterations **do**
  Randomly select a batch of rainy images A from some random scenes
  Randomly select a batch of clean images B from some random scenes
  Fetch corresponding groundtruth images gt from the same scenes as A
  Feed A to the generator G to generate translated images $fake\_B = G(x)$
  procedure UPDATE DISCRIMINATOR D
    Classify B and $fake\_B$: $D(B)$ and $D(fake\_B)$
    Compute GAN loss: $\mathcal{L}_{\text{GAN}} = \text{MSE}(D(B), 1) + \text{MSE}(D(fake\_B), 0)$
    Update the parameters of the discriminator D by gradient descent
  end procedure
  procedure UPDATE GENERATOR G
    Recompute $D(fake\_B)$ to calculate GAN loss: $\mathcal{L}_{\text{GAN}} = \text{MSE}(D(fake\_B), 1)$
    Compute patch-wise multi-layer contrastive loss $\mathcal{L}_{\text{PatchNCE}}(fake\_B, B)$
    Compute SSIM loss $\mathcal{L}_{\text{SSIM}}(fake\_B, gt)$ and L1 loss $\mathcal{L}_1(fake\_B, gt)$
    Apply loss weights λ to each loss and compute total loss
    Update the parameters of the generator G by gradient descent
  end procedure
end for

Figure 1: Pseudocode Description.

b. Explanation of Method Used and Results:

- Method and Dataset Used

We model image de-raining as a supervised, one-sided image-to-image translation (I2I) problem. There are two domains which appear in pairs in the dataset, rain and clean images and we want to learn from the paired images the mapping from rainy scenes (input) to clean images (output). Since the input and output have the same size, this is often trained with an encoder-decoder structure, typically a GAN. For our de-raining problem, we will base our algorithm on a novel GAN called CUT [20] which enables one-sided translation in the unpaired setting. Since there may be minor imperfections in paired images due to the limitations of data collection, we chose an unpaired I2I translation model so that we can relax the reliance on the paired ground truth during training and make it more tolerant to the subtle inconsistency inside each image pair. By comparing different examples of rainy images and their corresponding clean images in a contrastive manner, the model can learn to identify the common features that are shared across different rainy conditions. Once the model has learned this shared structure, it can use this knowledge to map any new rainy image into a clean image, even if it has not seen that particular rainy condition before. This is particularly useful in the context of mapping multiple rainy conditions into one clean image because it allows the model to generalize across different types of rainy conditions and create a single, high-quality clean image. Making relationships present in the input be analogously reflected in the output, the distance information in the image can be better preserved. TraVeLGAN [22], DistanceGAN [23], and GcGAN [24] are examples of models that allow one-way image translation without relying on cycle-consistency. However, they typically require knowledge of the entire image or use pre-defined distance functions. In this work, we replaced cycle-consistency with a cross-domain similarity function that learns to maximize information between input and output patches, without depending on a pre-determined distance measure. The main structure of the network is shown in Figure 2 below.

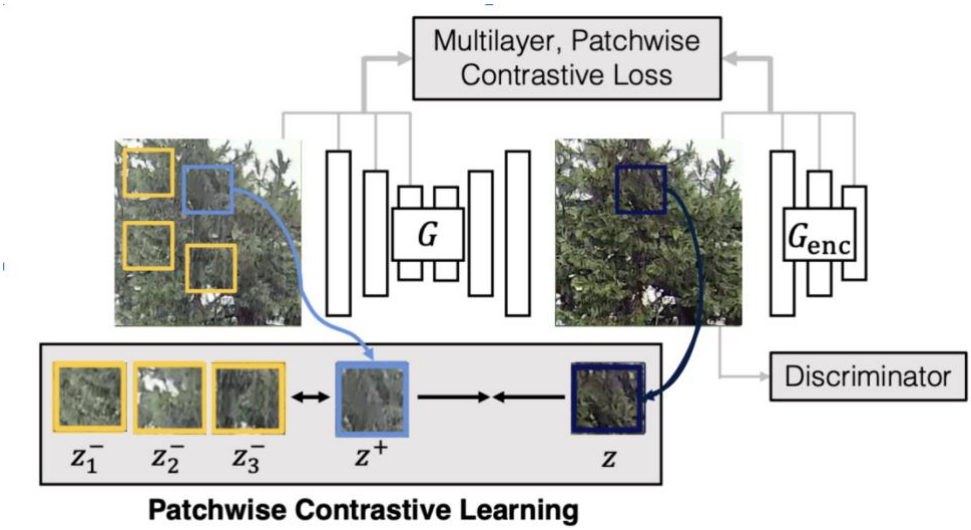

Figure 2: Network structure.

A common issue for the deep network is no ideal real pairs of rain and clean images. Distinguishing from the lack of ideal real ground truth pairs of rain and clean images that exist in current deep-learning-based single image de-raining datasets, the GT-RAIN dataset [4] consists of pairs of real rainy images and ground-truth images captured moments after the rain had stopped -- the negative performance impact induced by short temporal interval is much less than that of synthetic data which leads to unwanted artifacts. We use the real-paired GT-RAIN dataset to train and test our model. GT-RAIN features diverse and challenging scenarios, including various types of rain conditions, a large variety of background scenes from urban locations (i.e. buildings, streets, cityscapes) to natural scenery (i.e. forests, plains, hills), and varying degrees of illumination from different times of day, and all of which are captured by cameras that cover a wide array of resolutions, noise levels, and intrinsic parameters, helping us overcome the challenge of building a robust model.

In image de-raining, we wish to change the appearance of the scene (i.e., from rainy to clean), while preserving the content (objects in the scene). In the past, this was typically achieved by the adversarial loss [3] for appearance transformation and cycle-consistency [15] for content preservation. However, cycle-consistency is usually too restrictive as it assumes the mapping between domains is a bijection. In our case, a single clean image may be mapped to multiple rainy images (e.g., with light rain and with heavy rain). CUT proposed a multiplayer, patch-based contrastive learning to replace cycle-consistency, which enforces correspondence in content between input and output.

Given the source domain X (rainy scenes) and target domain Y (clean scenes), our task is to learn the one-directional mapping $G: \mathrm{X} \rightarrow \mathrm{Y}$. In CUT, G is a generator consisting of an encoder G_{enc} and a decoder G_{dec}. They are sequentially applied on the input rainy image X to generate clean output image $\hat{y} = G_{dec}(G_{enc}(x))$. A discriminator D is applied on the generator output whose goal is to distinguish synthetic clean images from the generator from real clean images. This encourages the generator G to produce realistic clean images. Together G and D are playing a min-max game where G attempts to generate realistic clean images to fool D while D devotes to identifying synthetic clean images from G. This leads to the first loss in CUT, adversarial loss, as shown in Eq. 1.

$$\mathcal{L}_{GAN}(G, D, X, Y) = \mathbb{E}_{y \sim Y} \log D(y) + \mathbb{E}_{x \sim X} \log\Big(1 - D\big(G(x)\big)\Big) \tag{1}$$

As shown in Figure 3, the feature tensor encodes both images, X and $\hat{y}$, after which a query patch is sampled from the output $\hat{y}$ and compared to the input patch located at the same position. We create a classification problem with (N+1) classes, where the positive class is the query patch and N negative patches are sampled from different locations in the same input image. To accomplish this, we incorporate a two-layer MLP network into the encoder part G_{enc} of our generator, which projects the input and output patches to a common embedding space.

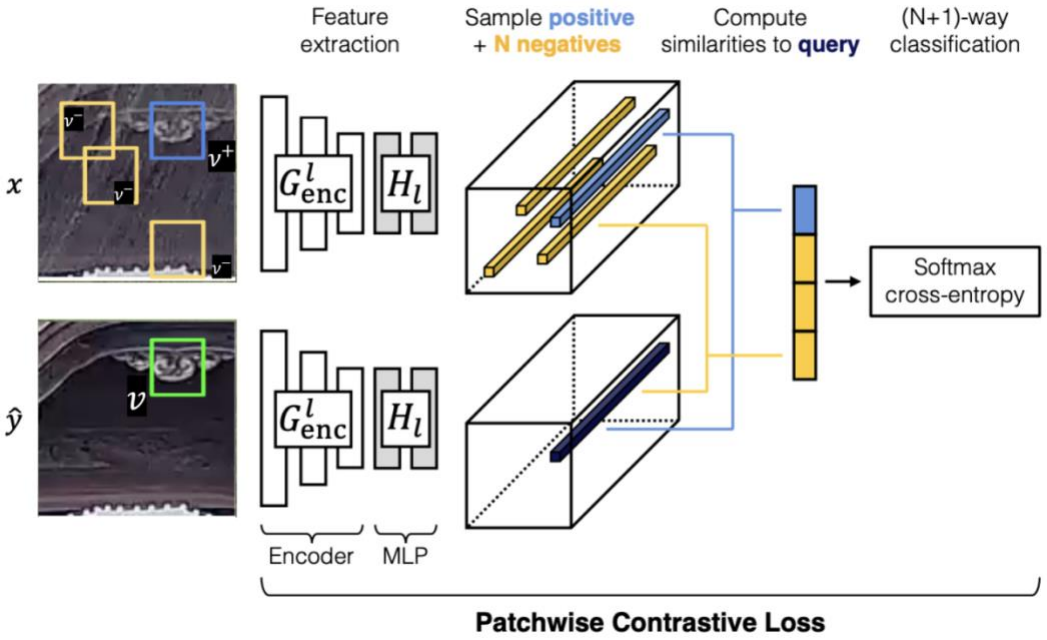

Figure 3: Network Methods.

Unsupervised learning involves learning a compressed code to reconstruct input data. Data imputation, such as denoising or context prediction, is more effective but still relies on a pre-specified loss function. A newer approach involves maximizing mutual information through noise contrastive estimation to group related signals together. This includes images with themselves or with transformed versions, neighbouring patches, and multiple views of the input image. Our approach uses PatchNCE loss in a multilayer, patch-based contrastive learning approach, considering crucial factors like several negatives, hyperparameters, and data augmentations to design an effective InfoNCE loss [25-26].

In addition to adversarial loss, CUT introduces contrastive loss in a patch-wise, multi-layer manner. In contrastive learning, there are three entities: a query v (an output rainy image patch), a positive example v^+ (corresponding input clean image patch), and some negative examples v^- (other input image patches). They are first mapped into vectors by a small network H and then normalized to calculate the distances between samples. Our goal here is to make the generated clean image patch v as close to the corresponding input rainy image patch v^+ as possible and also far from other patches v^- in the input rainy image. This ensures that the model preserves mutual information between the input rainy image and the output clean image. To minimize the distance between the query and the positive sample, contrastive loss maximizes the probability of the positive sample being chosen over negative samples by applying cross-entropy loss with a temperature τ, as shown in Eq. 2.

$$\ell(v, v^+, v^-) = -\log\left[\frac{e^{v\cdot\frac{v^+}{\tau}}}{e^{v\cdot\frac{v^+}{\tau}}+\sum e^{v\cdot\frac{v^-}{\tau}}}\right] \tag{2}$$

CUT applies contrastive loss to multiple layers $l \in \{1,2,\dots,L\}$ and multiple spatial locations $s \in S_t$ for each layer. This is called PatchNCE loss, as shown below, where $\hat{z}$ and z is decoder and encoder features respectively and S/s denotes spatial locations other than s, as shown in Eq. 3.

$$\mathcal{L}_{PatchNCE}(G, H, X) = \mathbb{E}_{x\sim X}\sum_{l}^{L}\sum_{s}^{S_t}\ell\left(\hat{z}_l^s, z_l^s, z_l^{\frac{S}{s}}\right) \tag{3}$$

Finally, since we have paired images, we can apply $\mathcal{L}_1$loss to the generated clean image $\mathcal{Y}$ and ground truth $\mathcal{Y}^{gt}$. We will apply a small weight to this loss so that our model will not focus on the subtle difference of an image pair in content.

Therefore, our overall objective function can be written as Eq. 4 below:

$$\mathcal{L}(G, D, H) = \mathcal{L}_{GAN}(G, D, X, Y) + \lambda_{NCE}\mathcal{L}_{PatchNCE}(G, H, X) + \lambda_{L1}\mathcal{L}_1(G(x), \mathcal{Y}^{gt}) + \lambda_{SSIM}SSIM(G(x), \mathcal{Y}^{gt}) \tag{4}$$

where λ_1 and $\lambda_2$are weights for different loss.

The challenge of defining a "perceptual" distance function for high-dimensional signals, such as images, has long been a problem in computer vision and image processing. Most image translation work relies on per-pixel reconstruction metrics, such as L1 loss, which can lead to blurry results and do not reflect human perceptual preferences. The VGG classification was repurposed as a "perceptual loss" for paired image translation tasks, surpassing traditional metrics like SSIM and FSIM in human perceptual tests. However, this approach cannot adapt to new data and domains. In contrast, our method, which uses mutual information, allows the cross-domain similarity function to adapt to specific input and output domains, avoiding the need for a pre-defined similarity function. Traditional unsupervised learning methods aim to learn a compressed code to reconstruct the input, but data imputation methods require a pre-designed loss function to measure predictive performance. In order to train our image-to-image translation model effectively, we employ a combination of several loss functions. Specifically, to ensure that our model can be trained effectively for the task of unpaired image-to-image translation from rainy images to clean images, we utilize a combination of three loss functions: GAN loss, PatchNCE loss, L1 loss, and SSIM loss.

The GAN loss encourages the generator network to produce images that are similar to the target domain (clean images) and fool the discriminator network into believing they are real. This helps to capture the high-level features and structures of the target domain, which may not be easily captured by the L1 loss alone. The PatchNCE loss is to have the output query patch be similar to the input patch at the same location, but not to the input patches from other areas. The L1 loss measures the difference between the generated images and the target domain at the pixel level. By minimizing this loss, we ensure that the generated images are as close to the target domain as possible in terms of pixel values. The SSIM loss measures the similarity between the generated and target images in terms of both structure and human perception. By increasing the weight of SSIM, we aim to make the inter-image structures and human perception as consistent as possible, in addition to ensuring pixel-level similarity. By combining these four loss functions, we are able to train our model to generate high-quality clean images from rainy images, while maintaining consistency with inter-image structures and human perception. In the case of determining the above methods and models, we also permuted different hyperparameters and loss functions and tested the results of model training with different combinations. The model training results are shown in the following table.

- Results
  - Quantitative Result

Metrics / Method	FID↓	SSIM↑	PSNR↑
Baseline	185	0.6744	19.08
CUT+L1	168	0.6640	19.04
CUT+SSIM	169	0.6880	**19.80**
CUT+SSIM+L1	**160**	**0.6918**	19.72

Table 1: Quantitative result table

The table shows the quantitative results from our experiments with the average of all outcome evaluation metrics. The baseline model is trained using original hyper-parameters and data augmentation settings. All other methods are trained with enhanced data augmentation and respective optimal hyper-parameters. The best results are highlighted in bold. We measure the performance of our model using the following three metrics: FID measures the distance between generated image distributions and target image distribution. SSIM measures the similarity between generated image and the ground truth image. PSNR quantifies reconstruction quality for images. As we can see from Table 1, only applying L1 loss can hurt the performance as it has lower SSIM and PSNR than the baseline. However, using SSIM loss greatly improves the performance and when it is used together with L1 loss, we get the best results. For our best result, the weight of the SSIM loss is 1.0, and that of the L1 loss is 0.1. The batch size is 2. For training, the number of epochs is 200 while the number of epochs with gradually decayed learning rate is 400.

  - Qualitative Result

In order to better demonstrate the qualitative results, we randomly select the results from some scenes in both the test set and validation set respectively and select a certain degree of rainy images in each scene. In that case, it's convenient for us to visually compare the different rain removal effects of the same rainy image under different model parameter settings, and to verify whether the best combination of model parameters is suitable for many different rainy scenes.
From Figure 4 below, we can see that the best combination of model parameters in the quantitative results also performs best in the quantitative results, which is almost the same as the ground truth images.
The basic baseline network causes a significant bluish image generation problem while adding only the L1 loss function causes a significant image edge distortion problem, and adding only the SSIM loss causes both problems, but both are relatively minor.

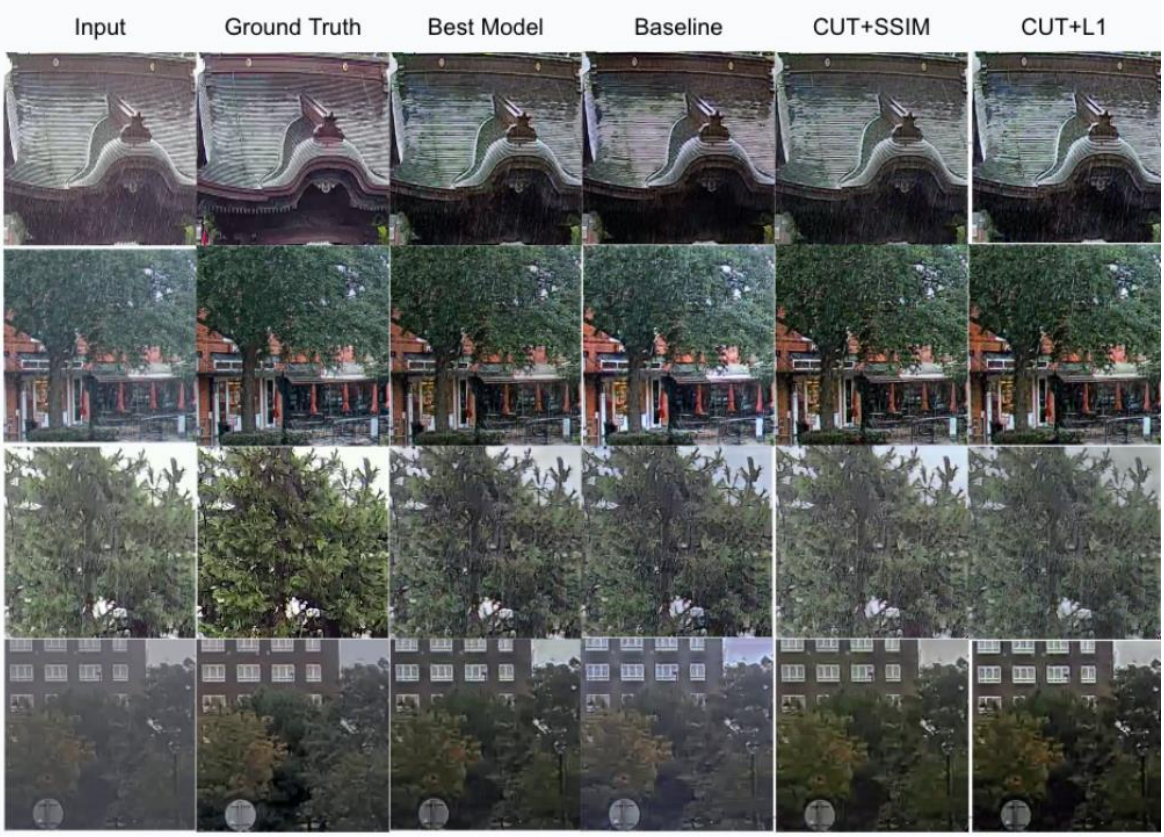

Figure 4: Qualitative results.

5) What did not work?

During our experiments, we attempted multiple strategies to achieve the best performance, but some of them do not work as expected. In the final strategy, we select a clean image B from a random scene that may be different from A's scene. However, at first, we were using the clean image from the same scene, i.e., the ground-truth image, which does not yield good performance. It may be because in contrastive loss we use B as negative samples and when B is gt, we are requiring the corresponding image patch in fake_B and gt to be far away while we want them to be the same.

An important part of the experiments is hyperparameter tuning. The original codes from CUT use the minimal batch size of one, which is very small. However, it is known that large batch sizes can make learning smoother since the gradients are averaged and are more likely to point in the right direction. Increasing batch size often results in faster convergence and better performance, whereas in our experiments we get the best results with batch size only 2. In addition, we encountered unstable training and model crashes as we increased the batch size. It may be worth researching and applying some latest techniques to stabilize GAN training and redo the experiment with different batch sizes and

correspondingly scaled learning rates. Apart from the batch size, we also attempted to increase the number of epochs and increase the strength of data augmentation so that we can train the model longer without overfitting. However, we observed that the model took longer to converge and when it converged it did not have better performance. We suspect that this may be limited by the backbone of the model since stronger data augmentation makes it more difficult to learn. For future work, we will replace the backbone with a deeper and more complex one and redo the experiments with more epochs and augmentations.

In our model, SSIM loss plays an important role as it provides paired information to make the training easier for the generator without introducing unwanted changes of content as L1 loss does. In some related tasks like de-fogging, weights for SSIM loss can be as large as 10. However, we observed a noticeable dip in the image quality when we use large weights for SSIM in our model. We suspect that this is because the model is focusing mostly on reducing SSIM loss as it is the dominant loss while ignoring the GAN loss which ensures image quality.

6) Conclusions and Future Work

In conclusion, we proposed an image de-raining model based on the unpaired translation model CUT with enhanced data augmentation and various ways to utilize paired information. The network has a significant rain removal effect. It achieved the best result combination of the average of metrics with refined hyper-parameters and the combination of the loss functions.

During our experiments, we observed that increasing batch size and increasing the number of epochs with stronger data augmentation did not bring any improvements. As we discussed before in section 5, in future research, we plan to enhance the backbone by using a deeper and more intricate architecture and conduct additional experiments with an increased number of epochs and stronger data augmentations like random rotation, colour distortion, and rain mask. Take colour distortion as an example, modifying the colour space will help the network to overcome the overfitting problem by simply relieving the reliance on colour histograms.

The FID is a popular evaluation metric used to assess the similarity between the distributions of real and generated images in GANs. In addition to evaluation, FID loss can also serve as a training objective to improve the quality of generated images. FID loss measures the distance between the feature representations of real and generated images, extracted by a pre-trained deep neural network, usually the Inception network. By minimizing the FID loss during GAN training, the generator is encouraged to generate images that are more similar in distribution to the real images, leading to higher quality and more realistic generated images. We will also try to add FID loss to our existing combination of loss functions or replace it with existing loss functions to explore better results.

For future work, we also consider style transfer with adaptive instance normalization. Meanwhile, we will also make comparisons with other well-known networks such as SOTA-specialized GANs.

7) References

[1] Bimal K Chhetri, Eleni Galanis, Stephen Sobie, Jordan Brubacher, Robert Balshaw, Michael Otterstatter, Sunny Mak, Marcus Lem, Mark Lysyshyn, Trevor Murdock, et al. Projected local rain events due to climate change and the impacts on waterborne diseases in vancouver, british columbia, canada. Environmental Health, 18(1):1–9, 2019.
[2] Qin Qin, Jingke Yan, Qin Wang, Xin Wang, Minyao Li, and Yuqing Wang. Etdnet: An efficient transformer deraining model. IEEE Access, 9:119881–119893, 2021.
[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. Communications of the ACM, 63(11):139–144, 2020.
[4] Yunhao Ba, Howard Zhang, Ethan Yang, Akira Suzuki, Arnold Pfahnl, Chethan Chinder Chandrappa, Celso M de Melo, Suya You, Stefano Soatto, Alex Wong, et al. Not just streaks: Towards ground truth for single image deraining. In European Conference on Computer Vision, page 723–740. Springer, 2022.
[5] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4):600–612, 2004.
[6] Thiyagarajan Jayaraman and Gowri Shankar Chinnusamy. Performance analysis of a dual stage deep rain streak removal convolution neural network module with a modified deep residual dense network. International Journal of Applied Mathematics and Computer Science, 32(1):111–123, 2022.
[7] Li-Wei Kang, Chia-Wen Lin, and Yu-Hsiang Fu. Automatic single-image-based rain streaks removal via image decomposition. IEEE transactions on image processing, 21(4):1742–1755, 2011.
[8] Fei Kou, Weihai Chen, Changyun Wen, and Zhengguo Li. Gradient domain guided image filtering. IEEE Transactions on Image Processing, 24(11):4528–4539, 2015.
[9] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng. Progressive image deraining networks: a better and simpler baseline. In CVPR, 2019.
[10] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. H. Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In CVPR, 2019.
[11] Y. Wei, Z. Zhang, H. Zhang, R. Hong, and M. Wang. A Coarse-to-Fine Multi-stream Hybrid Deraining Network for Single Image Deraining. In ICDM, 2019.
[12] H. Zhang, V. Sindagi, and V. Patel. Image de-raining using a conditional generative adversarial network. In CVPR, 2017.
[13] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley. Removing Rain from Single Images via a Deep Detail Network. In CVPR, pp.1715-1723, 2017.
[14] Yanyan Wei, Zhao Zhang, Yang Wang, Mingliang Xu, Yi Yang, Shuicheng Yan, and Meng Wang. Deraincyclegan: Rain attentive cyclegan for single image deraining and rainmaking. IEEE Transactions on Image Processing, 30:4788–4801, 2021.
[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei 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.
[16] B. Lu, J. Chen, and R. Chellappa. Unsupervised DomainSpecific Deblurring via Disentangled Representations. In CVPR, 2019.

[17] D. Engin, A. Genç, and H. Ekenel. Cycle-Dehaze: Enhanced CycleGAN for Single Image Dehazing. In CVPR Workshops: NTIRE, 2018.
[18] Changgang Zheng, Shufan Yang, Juan Marcelo Parra-Ullauri, Antonio Garcia-Dominguez, and Nelly Bencomo. Reward-reinforced generative adversarial networks for multi-agent systems. IEEE Transactions on Emerging Topics in Computational Intelligence, 6(3):479–488, 2021.
[19] Yi, Zili, et al. "Dualgan: Unsupervised dual learning for image-to-image translation." Proceedings of the IEEE international conference on computer vision. 2017.
[20] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan 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.
[21] ZhaoKang Guo, Mingzheng Hou, Mingjun Sima, and ZiLiang Feng. Derainattentiongan: Unsupervised single-image deraining using attention-guided generative adversarial networks. Signal, Image and Video Processing, 16(1):185–192, 2022.
[22] Amodio, Matthew, and Smita Krishnaswamy. "Travelgan: Image-to-image translation by transformation vector learning." Proceedings of the ieee/cvf conference on computer vision and pattern recognition. 2019.
[23] Thrun, Sebastian, L. Saul, and Bernhard Schölkopf. "Advances in Neural Information Processing Systems 16." Proceedings of the 2003 Conference. illustrated edition. London, England: The MIT Press. 2004.
[24] Fu, Huan, et al. "Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.
[25] Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding." arXiv preprint arXiv:1807.03748 (2018).
[26] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In Proceedings of the European conference on computer vision (ECCV), pages 172–189, 2018.