A General Approach To Single Image Snow Removal

Riley Cooper
Electrical and Computer Engineering
Queen's University
Kingston, ON K7L 3N6
15mrc5@queensu.ca

Jason Harris
Electrical and Computer Engineering
Queen's University
Kingston, ON K7L 3N6
harris.jason@queensu.ca

Liam Horton
Mechanical and Materials Engineering
Queen's University
Kingston, ON K7L 3N6
liam.horton@queensu.ca

Francesco Marrato
Electrical and Computer Engineering
Queen's University
Kingston, ON K7L 3N6
15fram@queensu.ca

Abstract

In this paper, we propose an alternative method for single image snow removal using a more general and less task specific architecture than most state of the art methods. The proposed architecture aims to learn a general transition map between two images and is commonly used for tasks like transforming an aerial view image to a map. Current state of the art architectures involve multiple layers of snow identification and other network structures specific to the snow removal task. Using a general transition map architecture allows for easier implementation and a wider field of applications. A general adversarial network (GAN) and more specifically a pix2pix GAN architecture trained on the Snow100K dataset was implemented to achieve this goal. This solution showed promising results with a peak signal-to-noise ratio (PSNR) of 23.2839 and a structural similarity index (SSIM) 0.7697 and warrants additional investigation to see the limit in its applicability to the task of removing snowfall in images.

1 Introduction

In any form of advertisement or presentation, it is important to have clear and high quality images to assist in effectively ideas. This necessity encourages the development of various post processing techniques that can improve image quality and in turn the effectiveness of various machine vision strategies. One issue being tackled is weather conditions obstructing the view of cameras. Any outdoor photography may deal with obstructed vision due to conditions like rain, fog, and snow. Strategies to deal with these weather conditions are crucial to maintain a high level of quality for these outdoor images.

Some approaches to the weather obstruction problem are classical machine vision based and attempt to remove the high frequency noise created by these conditions [1]. Others are machine and deep learning oriented including a variety of convolutional neural network (CNN) based architectures. These architectures leverage calculated and assumed priors of the obstructed images to improve results[2, 3, 4]. Generative adversarial networks (GANs) have also been used for weather condition removal. The GAN structure allows for some of these CNN schemes, that play the role of the generator, to be more effectively trained[5]. In some cases, the GAN is fed snowy or rainy images and the GAN attempts to remove the weather and generate the obstructed pixels based on the rest of the image.

Preprint. Under review.

The proposed research is a further investigation into GAN implementation for removing these visual obstructions in images. The focus is on desnowing as it presents a more difficult task since snow can vary in size and opacity across a single image. The work builds off of previously implemented solutions and compares the results to pre-existing data from other research endeavours such as DerainNet[2], DesnowNet[3], DehazeNet[4], and DesnowGan[5]. PSNR and SSIM are used as comparison metrics as they are already used for comparisons between the previously mentioned models[5]. The proposed solution was trained and tested on the Snow100K dataset which comprises of synthetically made snowy images using snow masks[3].

The GAN implemented is a pix2pix GAN which is designed for image-to-image translations. The pix2pix GAN is a conditional GAN (cGAN) architecture frequently used for taking in an image as input and applying some learned transformation to create a different output. Some examples of previous operations that have been performed using a pix2pix GAN are converting maps to satellite images, transforming sketches into photos, and black and white images into coloured images[6]. This architectures success with a variety image transformations begged the question of it's effectiveness on transformations such as removing snowfall (which is more akin to noise removal).

The contribution of this paper is to continue exploring methods of snowfall removal from images. Where other models use priors and knowledge of the problem to achieve good results, here a similar model which performs well for different applications is tested and shown to have comparable results based on training alone with little to no additional context of the problem. Most other generator structures such as DesnowGAN, have components designed for snow removal, followed by residual generation to fill the image[5]. The pix2pix GAN is designed for general image translations and learns the translation necessary based on the provided images[6]. The results show that simply learning the translation provides decent performance and further research into learning complex image translations could be combined with existing methods to further improve results.

The remainder of this paper begins with Section 2 which discusses related works to the problem including previous strategies for removing weather obstructions in images and different structures involved in the pix2pix GAN. Section 3 goes over the architecture of the implemented GAN for this paper. Section 4 and 5 go over the experiments conducted and results obtained which are compared to previous models. Finally Section 6 shows conclusions drawn from the experiments and suggestions of future work to continue the research.

2 Related works

In this section strategies used to combat various weather patterns that hinder machine vision performance and machine learning architecture used in the proposed solution are investigated.

2.1 Rain removal

Rain in images can cause many of the same issues for machine vision systems as snow. It can obstruct and obfuscate important image features in a scene that are key to vision system accuracy. Many strategies have been developed for specifically rain removal as well as combined snow and rain removal.

A vision based approach involves dividing an image into low and high frequency components and removing the rain from the high frequency component before recombining it with the low frequency[1]. However, learning based approaches are becoming more widely used and show very promising results. DerainNet is one such example which uses a CNN on a detail layer of rainy images to achieve improved deraining[2]. DerainNet along with other algorithms base their understanding of the problem in priors and assumptions about the quantitative effects the weather pattern has on the image. Other learning implementations involve multi-step networks, each with a different goal such as identifying image features or generating a clear image such as MERNet[7].

2.2 Haze removal

Many haze removal strategies rely on the creation of a medium transmission map to effectively dehaze images. These transmission maps estimate the the changes in light due to the haze and ideally provide a simple means of removing the haze using simple image operations. DehazeNet is one such network

which uses a CNN architecture and existing priors to generate a transmission map from a given hazy image[4]. There are many strategies to transmission map generation including combining multiple transmission maps into a fusion net for increased robustness[8]. The understanding of haze removal is still evolving and priors which describe properties of hazy images continue to be created[9]. These priors can be incorporated into future dehazing algorithms with the possibility of improved results.

2.3 Snow removal

Removing snow from images presents it's own issues compared to rain and haze. Mainly, that snow varies in size within the same image much more than other weather patterns. Snow also varies in opacity from translucent to opaque particles. Many approaches have been taken towards desnowing single images using machine learning techniques. One notable network is DesnowNet which combines feature descriptors to detect the snow and recovery modules to generate an estimate of the hidden space[3]. DesnowNet later evolved into DesnowGan which demonstrated improved performance over it's predecessors thanks to it's incorporation of a GAN structure[5]. DesnowGAN keeps a similar structure of descriptors and generators, but with multiple structure updates to DesnowNet to address computational complexity and network interpretability. They use a network similar to the DesnowNet network as the generator in the GAN structure to allow for improved residual generation and output image quality. DDMSNet is a recently developed network involving a coarse CNN structure for snow removal and priors derived from image depth and segmentation data to improve image quality[10]. A composition GAN solution was recently explored that generates both a clean image and a snow mask based on a provided snowy image. The original snowy image is then discriminated against the combination of these 2 generated images. The structure showed comparable performance in clean image generation and trained a structure to generate snow masks for future data set augmentation[11].

2.4 U-nets

U-nets are a variation of a CNN where a copy of the CNN is appended to the network that upscales the result. The copy uses a form of upscaling instead of pooling to create a 'U' shape in the data sizes produced (first decreasing in size, then increasing). This structure allows the features found in the CNN to be propagated in the upsampling layers to give more information on the image. This information can be used by a successive CNN to improve it's performance. This network is capable of learning off of fewer training examples by using severe data augmentations like elastic deformation. This allows the network to become invariant under these augmentations without the need for additional training data[12].

U-nets were originally designed for biomedical imaging applications where data sets are limited and the ability for a network to learn from fewer training examples is crucial[12]. U-nets are often used in combination with other networks, MADNet is a network that combines u-nets with GANs to improve the quality of images taken of the surface of mars[13].

2.5 GANs

Generative adversarial networks or GANs are a form of machine learning network used primarily to generate data. The basic structure of a GAN consists of a generator and a discriminator. Both are neural networks and are each trained in tandem during the GAN training process. The generator's goal is to produce images that resemble the ground truth while the discriminator's goal is to distinguish between ground truth or generated images. The discriminator is continually shown images from the dataset and the generated images and decides if each is generated or not while the generator eventually becomes better at fooling the discriminator. This game is mathematically presented as a minimax game where the generator and discriminator are adversarial players and the end goal is to reach a Nash equilibrium. Intuitively this means that the generator is producing almost identical images to the ground truth and the discriminator cannot distinguish between generated and real images by the end of the training[14].

3 Proposed solution

3.1 GAN structure

The proposed desnow cGAN is an evolution of the Pix2Pix image translation GAN proposed in [6]. and was outlined in [15]. The cGAN implementation uses the Keras deep learning framework available with TensorFlow, which matches what was outlined in the original paper. Our approach applies the capabilities of a translation cGAN with the availability of synthetic snowy images from the Snow100K dataset.

The cGAN includes modifications to change aspects of the generator discriminator pair between training runs. Adjustments to batch size, epoch count, downsampling strategy, batch normalization layers, and dropout can be adjusted with boolean values. This addition allowed the team to run back to back training runs, changing the architecture slightly each time and compiling the results of each change.

The code base can be found on GitHub here.

3.1.1 Generator

The generator uses layers of encoders and decoders in a U-Net architecture. The 256x256 pixel (px) input image is downsampled through a seven layer encoder structure. Each encoding layer is comprised of a two-dimensional convolution and a leaky rectified linear activation function (ReLU). Optional max pooling and batch normalisation layers can be activated through boolean switches. The encoder layers increase in size from 128 connections at the first layer to 512 at the seventh layer. Transitioning to the seven decoding layers, a similar combination of two-dimensional convolution and regular relu activation is used. The decoding layers have sizes that mirror those of the encoding layers, beginning at a size of 512 with the final layer of size 128. Located between the encoding and decoding layers is a bottleneck. The bottleneck consists of two, two-dimensional convolution layers of size 512 with a relu activation with a 2x2 pool size. Final layers of the generator include a two-dimensional transpose with a 2x2 stride followed by a tanh activation layer.

The U-net architecture places skip connections between corresponding layer of the encoder and decoder sections. These skip connections are merged using a concatenation step at the end of each generator decoding layer. The generator is trained in an adversarial process where the weights are updated in relation to adversarial loss and the L1 (Manhattan) loss between the generated image and the expected output.

3.1.2 Discriminator

The discriminator uses a deep CNN to perform image classification. Unlike a standard GAN, the discriminator takes a pair of images consisting of a source and target image and outputs the predicted likelihood of the target image being a real of fake translation of the source image. In our application this means the discriminator is provided with a pair of images consisting of either two identical ground truth images (source and target) or a ground truth image paired with an image created by the generator from a synthetic snowy image.

The discriminator consists of six two-dimensional convolution layers with patch sizes of 64, 128, 256, 512, and 1 respectively. Optional two-dimensional max pooling layers and batch normalisation layers can be activated through boolean switches and are applied on layers one through four with an additional batch normalisation performed after layer five. The discriminator uses a leaky relu activation function between each layer with an alpha (the negative portion of the slope) of 0.2. Updating the discriminator is done directly using Adam optimisation.

3.2 Data set

The Snow100K dataset was the single source of training, testing, and validation data for this research. The DesnowGAN presented in [5] also used the Snow100K dataset, presenting an opportunity for a quantitative comparison between the snow removal capabilities of DesnowGan and the proposed GAN architeture. Snow100K consists of 100 thousand images that have been synthetically obstructed by a snow mask [3]. Each image is part of a pair, consisting of a ground truth image without any

obstruction and a synthetically created image which has had a false snow mask applied over top of the ground truth image (Figure 1).

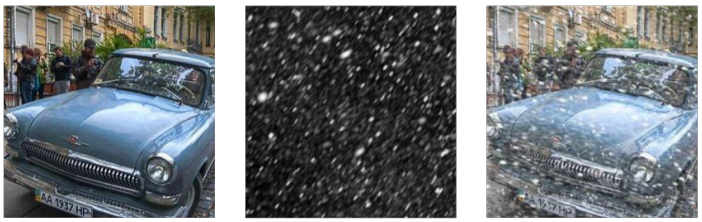

Figure 1: Example files from Snow100K dataset.

The dataset is a 50/50 split of training and validation sets, with each consisting of three levels of snow obstruction (Snow100K-S: small, Snow100K-M: medium, Snow100K-L: large snow particles). The proposed GAN architecture was trained on a subset of 2500 images from the large snow obstruction training set (Snow100K-L). The large snow obstruction mask was selected as it includes a combination of all three levels of snow obstruction and would give the best opportunity for significant image interference. A slice of 2500 images was chosen because of limitations in computing resources and training time. The images in Snow100K are of various rectangular sizes and had to be cropped to fit a square input size of 256x256 px. The cropped area was taken from the image center, and did not exceed the maximum dimensions of any of the input images.

After training, quantitative testing (PSNR and SSIM analysis) was performed on a second slice of 2500 images from the validation set of the Snow100K dataset. Qualitative testing was performed on a 1329 image subset of the Snow100K dataset that is composed entirely of realistic snowy images. Qualitative testing included both 256x256 px center cropped images and the full size original images. This subset is reserved for testing the generators ability to remove snow from images similar to what would be seen in winter environments. These images include snow on the ground, and are taken entirely outdoors. The testing subset of the Snow100K dataset comprises images never before presented to our implementation and are therefore a good test of its generalization.

4 Experiments

4.1 Computing resources

The proposed cGAN model was trained on three separate Graphics Processing Unit (GPU) accelerated systems. The first, was a remote cluster whose access was provided by Queen's University. The remote cluster was comprised of four Nvidia GeForce GTX 1080Ti graphics cards, of which only a single card was used. The second system was a personal workstation which utilised a single Nvidia GeForce RTX 3060Ti graphics cards. The third, was a Nvidia Tesla P100 provided using the premium subscription service of Google Colab notebooks. Single card GPU acceleration was chosen because of its simple integration with TensorFlow. Attempts were made to train the cGAN model on lower-end laptop grade GPUs. Laptops proved not powerful enough and were estimating training completion times measured in days instead of hours.

Training was done in parallel, with each system training approximately a third of the experimental GAN models (permutations of the base GAN). All systems demonstrated similar training speed when using a batch size of one (which covered six of the eight training runs). An average training speed of 13 images per second was observed, with GPU utilization never surpassing 75%. Under utilisation of GPU resources was attributed to the communication overhead required to load/offload small batches ranging from 1 to 32 images. The frequent loading and offloading of images to the GPU's on board memory was taking more cycles than the computation steps. Most often, GPU utilization did not surpass 50%, using a small batch size of a single image. A summary of the computing resources is provided in Table 1 , outlining hardware and associated CUDA/cuDNN versions.

Table 1: Outline of computing resources used for training.

Resource	GPU	Memory	CUDA Cores	CUDA Version	cuDNN Version
Cluster	Nvidia GeForce GTX 1080Ti	11GB GDDR5X	3584	10.1.0	7.6.5
Work Machine	Nvidia GeForce RTX 3060Ti	8GB GDDR6	4864	11.2.0	8.1.0
Google Colab	Nvidia Tesla P100	16GB CoWoS HBM2	3584	11.1.0	8.1.0

4.2 Experiment list

To demonstrate the efficacy of the proposed cGAN architecture for desnowing applications, a base case and six experiments have been developed which are summarized in Table 2. These experiments involve sequentially adjusting the model's hyperparameters with respect to the base case to identify the optimal network architecture.

All experiments were completed using mean normalized 256x256 px synthetic snowy images and the corresponding ground truth images. The outputs were RGB desnowed images with equivalent pixel size. The network for each experiment was trained from scratch and the weights were randomly initialized using a Gaussian Distribution with a mean of 0 and standard deviation of 0.02.The Adam optimization algorithm was used in all cases and initialized with a learning rate of 0.0002 and momentum parameter $\beta_1 = 0.5$.

The base case was developed using many of the recommended hyperparameters from [6]. This includes a batch size of 1, a dropout rate of 50% in the decoder block, and batch normalization in both the discriminator and the generator. The base case employed downsampling using strided convolutions in order to reduce the computational complexity and the number of parameters to limit the risk of overfitting. Finally, 100 epochs were used to maintain a reasonable training time while ensuring the convergence of comparison metrics.

The proposed experiments are a mixture of hyperparameter and network architecture adjustments. Experiments 1 and 2 consist of adjusting the batch size to 16 and 32, respectively to determine the effect on the model's performance and training time. For Experiment 3, the number of epochs was doubled compared to the base case. Experiment 4 utilizes maxpooling layers rather than strided convolutions for downsampling which further reduces the computational complexity since it is a fixed operation and cannot be learned. Experiment 5 eliminates batch normalization from the hidden layers of the discriminator and Experiment 6 adjusts the dropout rate to 25%.

Table 2: Experiment list for training the proposed cGAN.

	Base Case	**Experiment 1**	**Experiment 2**	**Experiment 3**	**Experiment 4**	**Experiment 5**	**Experiment 6**
Dataset Size	2500	2500	2500	2500	2500	2500	2500
Batch Size	1	16	32	1	1	1	1
Epochs	100	100	100	200	100	100	100
Downsampling	Strided Convolution	Strided Convolution	Strided Convolution	Strided Convolution	Max Pooling	Strided Convolution	Strided Convolution
Batch Normalization	Yes	Yes	Yes	Yes	Yes	No	Yes
Dropout (Decoder)	0.5	0.5	0.5	0.5	0.5	0.5	0.25
Input	Crop	Crop	Crop	Crop	Crop	Crop	Crop

4.3 Comparison metrics

To judge the performance of each variation of the model the PSNR and SSIM of were calculated every 10 epochs for each experiment. These values were used to compare the best performing model of those tested to previously published implementations.

PSNR is the ratio between a signal's power and it's noise. In practice it represents how much an image has degraded after being compressed or transformed in some way. It is based on the mean square error (MSE) between the altered and original image. This metric is more indicative of image degradation and is considered outdated by some in the research community. However, many of the previous networks reviewed that perform desnowing and weather obstruction removal tasks use it as a comparison metric [16]. For this reason it has been included in the result analysis.

SSIM is a metric based on visible structures in images. It is a better indication of what a human would consider similar compared to PSNR. It is frequently used in addition to PSNR for image quality and similarity comparisons.

5 Results and Analysis

The proposed cGAN was evaluated via four different methodologies. The first two methodologies are quantitative and deploy the PSNR and SSIM metrics that were discussed in Section 4.3. In the first quantitative comparison we evaluate the different experiments (see Table 2). We then discuss our results relative to other deep learning architectures. The third and fourth methodologies are qualitative in nature. We subjectively evaluate the performance of our implementation on four realistic snowy images (one selected by each group member) and then wrap-up the Results and Analysis section by comparing our implementation relative to the outputs of other published implementations.

5.1 Quantitative evaluation of the base case implementation

The Pix2Pix cGAN implementation deployed for our desnowing project is based heavily on the original Image-to-Image Translation with Conditional Adversarial Networks architecture. However, our team wanted to explore the effects of adjusting different hyper parameters and modifications to the architecture to verify themes published in previous papers [6, 5, 12] and identify any potential improvements. The team performed a total of 11 experiments in addition to running an analysis on the synthetic image data and the original Pix2Pix cGAN implementation (referred to as our Base Case).

Ideally, the team would have performed many runs on each experiment and published the average results. However, as mentioned in Section 4.1 our team was limited by computing resources. Therefore, the team elected to train the model on each experiment until they were successfully trained. This strategy resulted in three training runs for Experiment 1 and two training runs for Experiments 5 and 6. We also chose to train the model exhibiting the best PSNR values a second time to verify consistency of results. Experiments 1B and 4B have shorter training cycles because of timeouts between the client and GPU cluster (which happened frequently).

Figures 2 and 3 show the resultant PSNR and SSIM scores for each trained experiment as training progressed. Multiple runs for the same experiment are shown as lines with the same colour but different line types. Results of experiments with similar performance are directionally the same with both the PSNR and SSIM metrics. However, the order of best performing models differs depending on the metric deployed. For instance Experiment 4A performs well with both metrics but ranks higher with PSNR (when compared with SSIM).

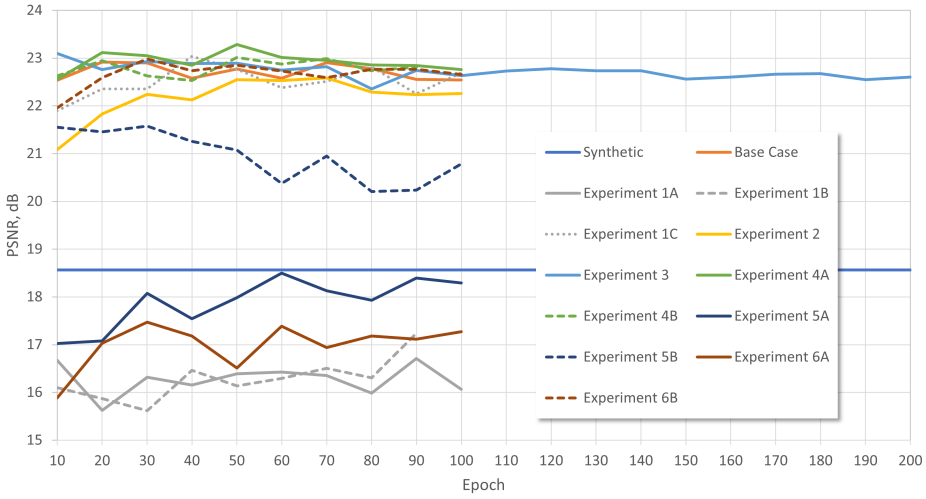

Figure 2: PSNR evaluated on cropped Snow100K-L subset

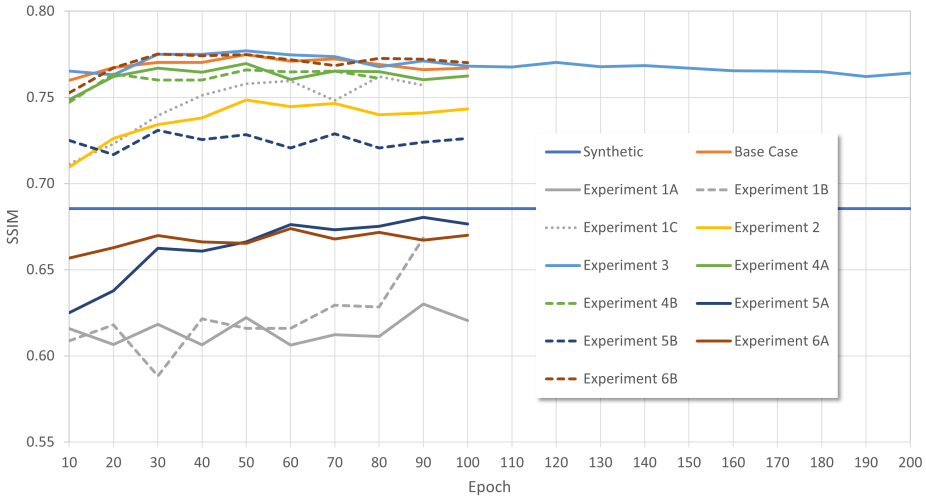

Figure 3: SSIM evaluated on cropped Snow100K-L subset

Several themes can be observed in these graphs. Specifically, all successfully trained models reached their maximum PSNR and SSIM scores between 10 and 70 epochs. Experiment 3 shows a slow decline in PSNR and SSIM values when the training is extended. This reinforces previous observations by [6] that more training does not necessarily result in better outcomes. Experiments with larger batch sizes (Experiments 1C and 2), batch normalization disabled (Experiment 5B) all experience slower convergence. It is plausible that these experiments may suffer from a poorer initial weights within the networks.

Experiment 4 was a modification of the the Pix2Pix cGAN architecture where all Conv2D layers with a stride of 2 were modified to have a single stride. Downsampling was achieved by the addition of a Maxpool layer with a kernel size of 2x2 and a stride of 2. This modification resulted in the best PSNR scores. Interestingly, the SSIM score for this experiment wasn't as good when compared to other experiments where the original model architectures were left intact. All subsequent results were assessed using scores derived from or images generated from the model that was generated at 50 epochs in Experiment 4A.

5.2 Quantitative comparison with other models

Table 3: Performances of various methods on Snow100K-L's test set

Dataset	Snow100K-L	
Metric	PSNR	SSIM
Synthesized data	18.6777	0.7332
Synthesized data (cropped subset)	**18.5651**	**0.6855**
Zheng *et. al.* [1]	19.9562	0.7221
DerainNet [2]	19.1831	0.7495
DehazeNet [4]	22.6175	0.7975
DeepLab [17]	21.2931	0.7747
Jorder [18]	23.4085	0.8091
DuRN-S-P [19]	27.2131	0.8891
DeSnowNet [3]	27.1699	0.8983
DS-GAN [5]	28.0677	0.9211
Our Implementation	**23.2839**	**0.7697**

As previously discussed, there are multiple deep learning implementations of single image rain, haze and snow removal. We compare our results with those published in Table 3. It is important to note that, due to the computing resource limitations, a direct comparison was not feasible. Similar to our

training methodology our model validation was done on a 2500 center cropped image subset of the Snow100k validation set. The average PSNR and SSIM scores for the subset aren't significantly different from that of the full dataset. Results in *Synthesized Data* row(s) denote the similarities between the synthesized snowy image x and the snow-free ground truth y; the scores represent averages over the entire test set for published methods and a cropped subset for our implementation. The maxpool modifications that we implemented in Experiment 4 result in a model that performs well relative to the bulk of the existing deep learning implementations (more-so when we consider that our test subset starts with lower PSNR and SSIM scores).

5.3 Qualitative comparison and applications

The group has selected four realistic snowy images to evaluate the performance of our selected implementation. The first comparison will discuss image output characteristics in general terms. The second comparison evaluates the implementation for potential use in automotive applications. A third comparison evaluates how the algorithm may function in the context of ongoing research on infrastructure projects. Finally, we present a failure case.

Figure 4 showcases the abilities of our implementation. The model successfully removed the light snowfall at different depths within the image while leaving all major structural elements in place. Some snow flakes/particles appear to be removed from the surface of the cardinal's feathers demonstrating the difficult task of differentiating between snowfall and snow that is in place. Colour reproduction is excellent but there is a loss of contrast in some areas of the image. It is unclear if the loss of contrast is a trained feature of the model to deal with the snow removal. This loss is most prominently seen on the cardinal's eye and in the region above its right foot and below its right feather. There are minor losses of detail in the generated image where it appears to be slightly pixelated. This is most prominently seen in the striations on the cardinal's wing.

Figure 4: General image characteristics - realistic snowy image (left) and desnowed image (right)

The use of a Pix2Pix cGAN for automotive applications shows merit when considering the results of Figure 5. Figure 5a being the ground truth, with Figures 5b to 5d being increasing passes through the desnowing generator. The generator demonstrates a strong ability to remove low frequency large snow particles from the images but struggles with higher frequency smaller snow particles. The high contrast areas of the image contribute to its ability to produce a strong result. The areas being replaced when snow particles are removed are of a single color (red, black, or white) making replacement easier. Passing the output through the generator multiple times (seen in Figure 5c and Figure 5d) begins to remove the high frequency small snow particles with the trade off being the decreasing quality of the image. Decreasing image quality is especially evident in Figure 5d, where the red panelling of the vehicles is beginning to show qualities similar to image compression artifacts. Also in Figure 5d, the details of the vehicle such as its taillights are beginning to be blended with the rest of the vehicle. Applying this desnowing generator to autonomous driving (the original application considered for this project) would have to consider the trade off between snow removal and image quality. Desnowed images could be better interpreted by classification algorithms, but could cause

further issues if object details are lost as the snow particles are removed. Figure 5c is an excellent example of snow removal without significant detail loss.

(a) Ground truth (b) Generated desnow image - 1 Pass

(c) Generated desnow image - 2 Passes (d) Generated desnow image - 4 Passes

Figure 5: Desnow with multiple passes

In addition to promising results for the application of computer vision in autonomous vehicles, Figure 6, which shows a realistic snowy image on the left and corresponding desnowed image on the right, demonstrates the proposed architecture's capability to offer improved vision-based structural inspection under adverse weather conditions. High quality images are crucial for a successful inspection system and weather conditions can obstruct the view of defects possibly leading to an inaccurate conditional assessment.

Figure 6: Research applications for infrastructure - realistic snowy image (left) and desnowed image (right).

The generator's output was able to remove all of the larger snow particles but failed to remove some of the smaller snow particles over the water. This region in the original image was already somewhat blurry due to the shadow of the bridge which could have made it difficult for the network to detect the finer snow particles. The region under the bridge where a significant portion of particles were removed has a similar quality output compared to the original realistic image. When analyzing the image for potential structural inspection, although this image does not offer millimeter resolution to begin with, the generated image offers a much clearer view of the structure demonstrating the success of the network.

Finally, there were cases where the proposed architecture was unable to remove most of the snow. As shown in Figure 7, there are still many snow particles which can be seen in the desnowed image on the right. Some snow particles have been removed such as the large snowflake slightly above the letter *'r'* in the word *'Grill'* on the sign or slightly above the letter *'B'* in *'Blizzard'*. However, the majority have been left untouched.

Figure 7: Failure case - realistic snowy image (left) and desnowed image (right).

There are many reasons why this may have occurred. One possibility may be that the high variance in snow particle size in the image made it difficult for the translation map to incorporate all sizes and remove them. The proposed architecture was also observed to have trouble on images with monotone sections which may have contributed to presented output. The most likely case was determined to be due to the inherent randomness of using a GAN and machine learning architectures in general. This is likely one of the weaknesses of using a more generalized model for this task.

5.4 Qualitative comparison with published implementations

Subjective qualitative evaluation between different models is achieved by examination of Figure 8. Our implementation performs well in comparison to other published techniques. Limitations with the format of the report limit the size of images that can be presented and the images displayed in Figure 8 benefit from downsampling of images from their original sizes. Generally our implementation performs similarly to DehazeNet and DesnowNet. Our implementation outperforms all other implementations on snowfall removal on the tree image in the second row. However, in full size views our generated image suffers from some pixelation.

Figure 8: Qualitative comparison - First column (left): realistic snowy image. Second column: DerainNet [2]. Third column: DehazeNet [4]. Fourth column: DeepLab [18]. Fifth column: DesnowNet [3] Sixth column (right): Our Implementation.

The images in Figure 8 were generated by downsampling realistic snowy images to 256x256 px for processing by our model. The output was then upscaled to the original resolution. Interestingly, the portion of the generated image representing the center crop is quite different from those generated by our model using the original center crop. Consider Figure 9. The left image was generated by taking a 256x256 px center-crop of the original snowy image. This image has replaced some snow with black artifacts. However, when the whole image is processed and a center crop of the upscaled image is analyzed the artifacts are not present. Image quality suffers because of losses from down and upscaling processes.

Figure 9: Qualitative comparison - Left: center crop. Right: full size.

6 Conclusion

This paper presented a new and more general architecture for image desnowing. The pix2pix GAN architecture allowed for the transition map between the snowy and desnowed image to be learned as a general translation map rather than having the architecture be designed for the purpose of desnowing in advance. The more general and simplistic approach showed promising results and in theory could be used not only for desnowing, but also for removal of other obstructions in images. In comparison with state of the art models in use, our more general implementation was able to provide results both quantitatively and qualitatively similar to some previously used methods. Overall, this study has shown that more general models with less knowledge of the desnowing application itself can be effective and their incorporation into existing networks warrants further investigation.

6.1 Future works

Due to the time constraints of this study, there remain many other tests and improvements that can be done to the above research. The majority of the network permutations were only tested once which is not representative of their average performance. Running each experiment multiple times would result in a more accurate depiction of the performance of the network architecture and ensure that the initial values generated have little effect on the outcome. Structure modifications to allow for experimentation on larger images of varying size would allow for more widespread applications.

6.2 Group member contributions

All group members contributed equally to the term project.

Riley Cooper: Researched different methods for performing desnowing and related problems such as dehazing and deraining, wrote auxiliary helper functions to assist in simulation validation such as calculating the psnr and ssim, wrote the introduction, related works, conclusion, and assisted with results sections of the final report. Co-authored the initial presentation, final report formatting adjustments.

Jason Harris: Researched GAN architectures for the desnow problem, modified the pix2pix cGAN for experiments, attempted initial runs on GPU enabled laptop, wrote code to generate images for all models and experiments, modified Riley Cooper's PSNR and SSIM code to evaluate all models and co-authored the initial presentation and the final report.

Liam Horton: Identified a suitable dataset for snow removal, researched GAN architectures for the desnow problem, wrote code to implement these architectures and completed preliminary training with the Celeb dataset. I attempted to transfer the preliminary model to work on our application of desnowing images. After having difficulty with this task I helped to ensure the modified pix2pix cGan would work for our application. I ran a significant portion of the GAN training runs and co-authored the initial presentation and the final report.

Francesco Marrato: Researched GAN architectures for the desnow problem, wrote code for data pre-processing for network training, organized the project onto a git repository, cleaned/summarized all code used for the project, ran a good portion of the GAN training runs and provided summaries of the results, wrote "Dataset" and "GAN structure" sections of the report.

References

[1] X. Zheng, Y. Liao, W. Guo, X. Fu, and X. Ding, "Single-image-based rain and snow removal using multi-guided filter," in *Neural Information Processing*, M. Lee, A. Hirose, Z.-G. Hou, and R. M. Kil, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 258–265.

[2] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2944–2956, 2017.

[3] Y.-F. Liu, D.-W. Jaw, S.-C. Huang, and J.-N. Hwang, "Desnownet: Context-aware deep network for snow removal," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 3064–3073, 2018.

[4] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, "Dehazenet: An end-to-end system for single image haze removal," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5187–5198, 2016.

[5] D.-W. Jaw, S.-C. Huang, and S.-Y. Kuo, "Desnowgan: An efficient single image snow removal framework using cross-resolution lateral connection and gans," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1342–1350, 2020.

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *CVPR*, 2017.

[7] Q. Wang, Y. Cheng, Y. Yang, C. Wang, and C. Yang, "Multi-stage enhanced rain image restoration network," *Journal of Physics: Conference Series*, vol. 2035, no. 1, p. 012040, sep 2021. [Online]. Available: https://doi.org/10.1088/1742-6596/2035/1/012040

[8] A. Dudhane and S. Murala, "Ryf-net: Deep fusion network for single image haze removal," *IEEE Transactions on Image Processing*, vol. 29, pp. 628–640, 2020.

[9] Q. Zhu, J. Mai, and L. Shao, "A fast single image haze removal algorithm using color attenuation prior," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3522–3533, 2015.

[10] K. Zhang, R. Li, Y. Yu, W. Luo, and C. Li, "Deep dense multi-scale network for snow removal using semantic and depth priors," *IEEE Transactions on Image Processing*, vol. 30, pp. 7419–7431, 2021.

[11] Z. Li, J. Zhang, Z. Fang, B. Huang, X. Jiang, Y. Gao, and J.-N. Hwang, "Single image snow removal via composition generative adversarial networks," *IEEE Access*, vol. 7, pp. 25 016–25 025, 2019.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[13] Y. Tao, S. Xiong, S. J. Conway, J.-P. Muller, A. Guimpier, P. Fawdon, N. Thomas, and G. Cremonese, "Rapid single image-based dtm estimation from exomars tgo cassis images using generative adversarial u-nets," *Remote Sensing*, vol. 13, no. 15, 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/15/2877

[14] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

[15] J. Brownlee, "How to develop a pix2pix gan for image-to-image translation," August 2019.

[16] C. Thomas, "Deep learning image enhancement insights on loss function engineering," February 2020.

[17] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[18] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan, "Deep joint rain detection and removal from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1357–1366.

[19] X. Liu, M. Suganuma, Z. Sun, and T. Okatani, "Dual residual networks leveraging the potential of paired operations for image restoration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7007–7016.