Residual-Guide Network for Single Image Deraining

Zhiwen Fan[†], Huafeng Wu[†], Xueyang Fu, Yue Huang, Xinghao Ding[*]
School of Information Science and Engineering,
Xiamen University, China.
[*]Corresponding author.
[†]Co-first author contribute equally.
{waynefan,whfeng,fxy}@stu.xmu.edu.cn,{yhuang2010,dxh}@xmu.edu.cn

ABSTRACT

Single image rain streaks removal is extremely important since rainy condition adversely affects many computer vision systems. Deep learning based methods have great success in image deraining tasks. In this paper, we propose a novel residual-guide feature fusion network, called ResGuideNet, for single image deraining that progressively predicts high-quality reconstruction while using fewer parameters than previous methods. Specifically, we propose a cascaded network and adopt residuals from shallower blocks to guide deeper blocks. We can obtain a coarse-to-fine estimation of negative residual as the blocks go deeper with this strategy. The outputs of different blocks are merged into the final reconstruction. We adopt recursive convolution to build each block and apply supervision to intermediate de-rained results. ResGuideNet is detachable to meet different rainy conditions. For images with light rain streaks and limited computational resource at test time, we can obtain a decent performance even with several building blocks. Experiments validate that ResGuideNet can benefit other low- and high-level vision tasks.

ACM Reference Format:
Zhiwen Fan, Huafeng Wu, Xueyang Fu, Yue Huang, Xinghao Ding. 2018. Residual-Guide Network for Single Image Deraining. In *2018 ACM Multimedia Conference (MM '18), October 22–26, 2018, Seoul, Republic of Korea.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240508.3240694

1 INTRODUCTION

Rain streaks degrade visual quality on images and videos. Due to the block and blurred effect to objects in a rainy image, undesirable result of many outdoor computer vision applications like object detection [22] will be adversely affected. However, most existing algorithms are trained with well-controlled conditions. Thus, designing an effective method for removing rain streaks is desirable for a wide range of practical applications. Deep learning has been introduced for this problem since Convolutional Neural Networks (CNN) have proven powerful for a variety of vision tasks.

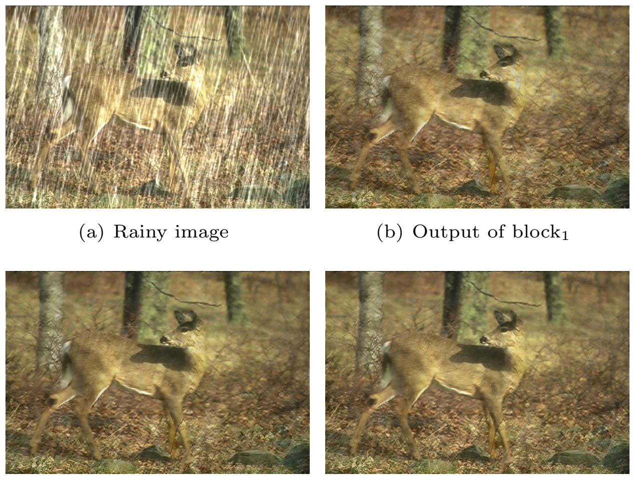

(a) Rainy image (b) Output of block$_1$ (c) Output of block$_3$ (d) Output of block$_5$

Figure 1: Progressive high-quality result as blocks go deeper, the SSIM of the output of block$_1$, block$_3$, block$_5$ is 0.927, 0.935, 0.943, respectively.

However, existing models in rain streaks removal tasks tend to learn negative residual within a single model, these models have to be carefully designed with tones of parameters to capture different modalities of rain streaks. Also most methods optimized with Euclidean distance that will inevitably generate blurry predictions since the per-pixel losses do not close to perceptual difference between output and ground-truth images as human visual perception [14]. Further, it is wasteful to utilize a resource-hungry model to meet all kinds of demands for rain streak removal tasks. For example, under light rainy conditions, a simple model can obtain a decent derain result, whereas a heavy rainy image should be handled with a computationally intensive model to detect rain streaks with different shapes and scales.

To address above drawbacks, we propose the residual-guide feature fusion network (ResGuideNet) in a cascaded architecture. Each block contains a global shortcut to predict residual [8] which can make the learning process much easier. However, a simply cascaded basic building blocks is of difficulty to improve the reconstructed quality in deeper blocks. We conjecture that it is because a cascaded architecture may lost

MM '18, October 22–26, 2018, Seoul, Republic of Korea
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5665-7/18/10...$15.00
https://doi.org/10.1145/3240508.3240694

valuable intermediate reconstruction features which makes the deeper blocks difficult to learn new rain streak pattern. We then proposed to concatenate the predicted residuals from shallower to the deeper blocks. By using this simple operation, the shallower residuals can guide deeper predictions to generate a finer estimation as shown in Figure 1.

In addition, we apply supervision to all intermediate outputs which can obtain a coarse-to-fine rain streak residual as the blocks go deeper. The basic rain streak removal block is based on recursive computations with a proper shortcut strategy to reduce the number of network parameter while keeping good deraining performance. The final recovered image is obtained by merging all outputs of intermediate reconstruction which can be viewed as an ensemble learning.

The contributions of our paper are three-fold:

(1) We build a single and separable network that can handle different rainy conditions. By maintaining negative residual features in shallow blocks to deep blocks, a coarse-to-fine estimation of negative rain streaks residual can be obtained. As the application scenario changes, the user can detach a portion of our model to meet varying computational requirements at test.
(2) We apply supervision to all the intermediate and final reconstructions with a combined loss function. The proposed model combines all intermediate results to obtain the final result, which can be viewed as ensemble learning.
(3) We discuss how ResGuideNet can be applied to other low-level vision tasks including denoising and the reconstructed images could benefit down-stream applications such as object detection.

All our code and real-world dataset will be released soon.[1]

2 RELATED WORKS

Depending on the input format, existing rain streak removal algorithm can be roughly categorized into video-based methods and single-image methods. For video-based methods [1] [2] [17] [9] [25], inter-frame information between adjacent frames is leveraged to identify rainy region and remove rain.

Removing rain streaks from single-image is more challenging since less information can be utilized. [15] attempts to extract rain streaks and background details from high-frequency layer by sparse-coding based dictionary learning. [21] proposes a framework to rain removal based on discriminative sparse coding. [19] learn background from pre-collected natural images and rains from rainy images by utilizing two Gaussian mixture models (GMMs).

Deep learning has also introduced for restoration problems and convolutional neural networks (CNN) have found great success in processing many computer vision problems. The first CNN-based method for single image deraining was introduced by [7]. The authors build a relative shallow network with 3 layers to learn the mapping function. In [8], combining with ResNet [11] [10], the authors present a deep detail network (DDN) to learn residual with the high frequency part of rainy images. In [33], the author proposed a conditional GAN-based algorithm for removal of rain streak from a single image. [30] learn binary rain region mask rand remove the rain streaks simultaneously through a multi-scale network (JORDER). [32] utilize the rain density information with a multi-stream densely connected network (DID-MDN) for jointly rain-density estimation and deraining. Further, single image dehaze [4] [24] [18] [31] achieving promising result by introducing deep learning models.

In image restoration field, achieving good performance with a moderate number of network parameters is an important goal for designing a deep neural network, [26] [6] [20] proposed to reuse the same convolutional filter weight to learn hierarchical feature representation. In order to avoid gradient vanishing problems and reduce the total parameters for very large deep models, [16] [27] proposed to use recursive computation with proper supervision and shortcut to achieve state-of-the-art performance in single-image super resolution while using few parameters.

3 METHOD

3.1 Motivation

Since rain streaks are always overlapped with background texture, most methods tend to learn the negative residual of its input with a complex or carefully designed model. However, this may lead to an over-smoothed result and need tons of parameters to optimize. Also, it is infeasible to apply a resource-hunger model to process video frame-by-frame for its time-consuming processing. On the other hand, to meet different kind of demands in practical applications, a light weight or detachable network is desirable since their huge number of parameters will limit their application in mobile device, automatic driving and video survillence. However, existing methods use a fixed computational budget to handle both "easy" and "hard" application scenarios. This is less flexible for a model to implement in real-world application.

As is evident in Figure 3, we test our ResGuideNet under heavy and light rain streaks conditions. We can observe our method has a progessively better reconstruction as blocks go deeper. However under light rainy condition, the SSIM [29] does not improve much since block$_2$ to block$_5$, as shown in Figure 4.

Thus, we would like to build a model that receives good results on all devices, with varying computational constraints of all devices. Furthermore, users can improve the average reconstruction quality by reducing the amount of computation that spent on light rain condition to save up computation for heavy cases.

Motivated by the prior work that has a resouce-efficient implementation [12], we aim to construct CNN that is able to slice the network to meet the computational limitation to process rain streaks under different rainy conditions. Unfortunately, deep neural network is inherently related with the early-existed features. Thus, we build a model that incorporates a series of deraining sub-networks and progressively generate a cleaner estimation given a rainy input. We can

[1]zhiwenfan.github.io

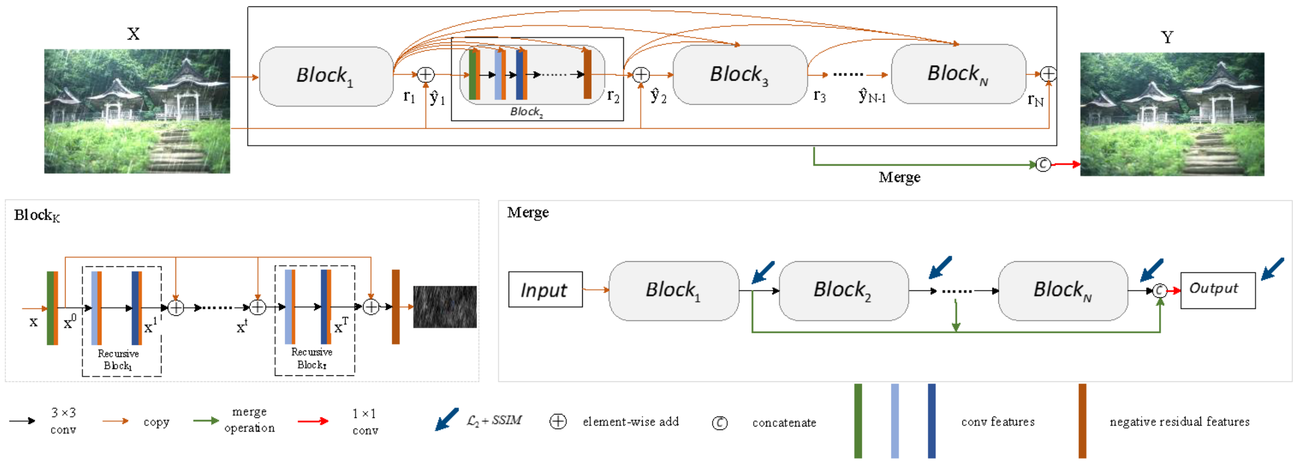
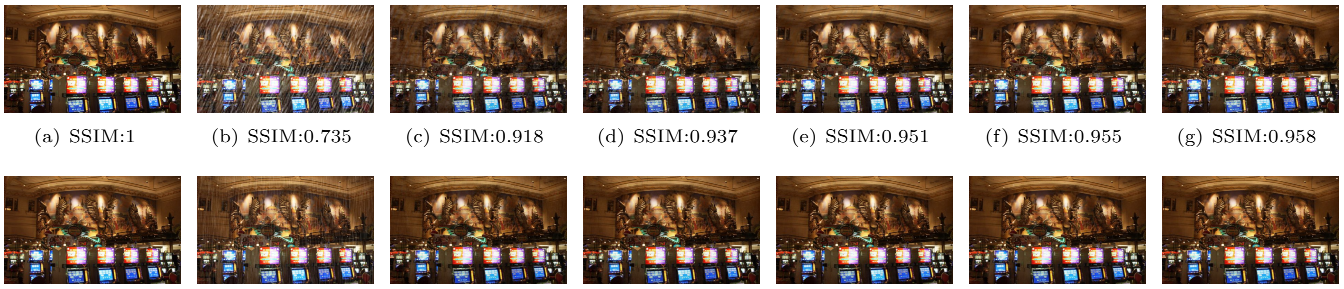


Figure 2: The proposed structure of our rain streak residual-guide network(ResGuideNet)



(a) SSIM:1 (b) SSIM:0.735 (c) SSIM:0.918 (d) SSIM:0.937 (e) SSIM:0.951 (f) SSIM:0.955 (g) SSIM:0.958

(h) SSIM:1 (i) SSIM:0.885 (j) SSIM:0.968 (k) SSIM:0.971 (l) SSIM:0.973 (m) SSIM:0.975 (n) SSIM:0.976

Figure 3: Comparasion under heavy and light rainy conditions using our ResGuideNet, (a) and (h) are clean image. (b)and (i) is the synthetic rainy image under heavy rain condition and light rain condition, respectively. (c)-(g) (j)-(n) are the results from block$_1$ to block$_5$ of ResGuideNet under heavy and light rainy conditions, respectively.

also use a portion of the whole model to handle different rainy conditions.

3.2 Residual Feature Reuse

A major challenge for deep learning models is its optimization. To address the gradient vanish problem in back propagation, shortcuts have been proposed to stabilize the gradient flow in deep residual networks (ResNet). By assuming that the residual mapping is much easier to learn than the original unreferenced mapping, residual network explicitly learns a residual mapping for a few stacked layers. With such strategy, deep neural networks can be easily trained and therefore, ResNet has achieved very impressive performance on the a number of tasks. Also, [13] proposed to concatenate feature maps densely from lower to deeper layers which can alleviate the gradient vanishing problem and reduce the number of model parameters. It may be interpreted as there is no need to relearn redundant features. [28] has introduced dense connection in regression tasks and has shown densely connections could benefit the long-term memories and the restoration of mid/high frequency information.

In this paper, we adopt global residual learning with a long shortcut in each block to ease the learning process. Each block consists of several convolutional layers using Leaky Rectified Linear Units, we refer this architecture as **Baseline model**. However, simply cascaded blocks cannot obtain promising results. We conjecture that deeper blocks is difficult to extract new rain streak patterns and the intermediate reconstructions from lower blocks contain valuable information have lost. In order to deal with this problem, we suggest to integrate information from previous blocks to deeper ones, compensating information and further enhance high-frequency signals.

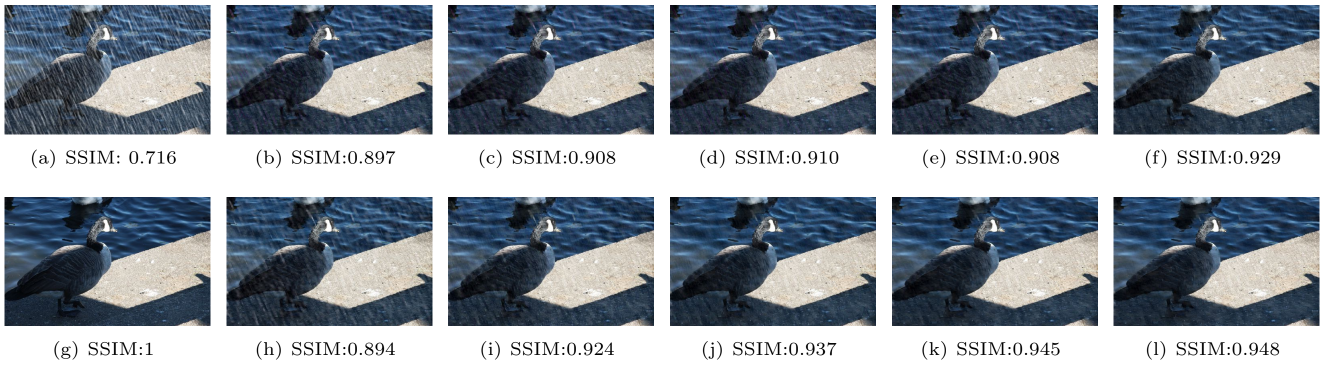


Figure 5: Comparasion between Baseline model and Baseline model with residual reuse (Baseline-RR) on a single test image, (a) is the synthetic rainy image , (g) is clean image , (b)-(f) are the results from block$_1$ to block$_5$ of Baseline, (h)-(l) are the result from block$_1$ to block$_5$ of Baseline-RR.

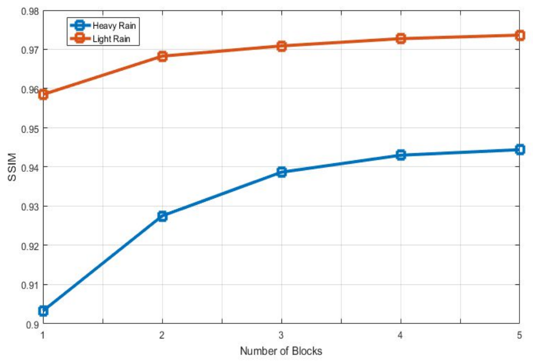


Figure 4: Deraining result with ResGuideNet under heavy and light rain conditions, we obtain the result on the test datasets and averaged them. We can observe the reconstruction does not improve much for light rain condition since block$_2$ to block$_5$.

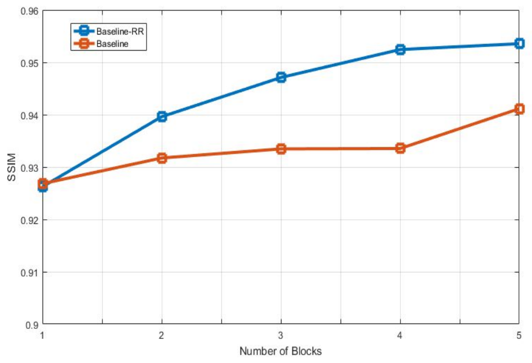


Figure 6: Comparison of SSIM between Baseline model and Baseline with residual reuse (Baseline-RR).

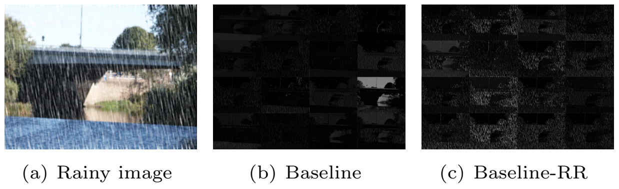


Figure 7: Features with/without residual reuse.

We evaluate the benefit of the transition from natively cascading deraining blocks (**Baseline**) to our adopted negative residual reuse (**Baseline-RR**) by feature fusion strategy using 5 blocks. For fair comparison, we increase the number of feature maps in each building block of Baseline model to have the same parameters with Baseline-RR. We conduct the experiments on the dataset provided by [8]. As is clear from the visual quality of reconstruction in Figure 5, Baseline-RR obtain a more eye-pleasing reconstruction and a higher SSIM value as the blocks go deeper. In Figure 6, Baseline-RR obtains a gradual increase on SSIM as the block becomes deeper, whereas the Baseline model does not possess this property, the SSIM value is based on averaging all test images.

We further show the 16 feature maps of Baseline-RR and Baseline model in Figure 7, we can observe the 16 feature maps of Baseline-RR have larger activations on rain streaks in the 2nd layer of Block$_2$ than Baseline model. This is because Baseline-RR incorporates rain streaks?residual of Block$_1$ to suppress image contents in features for the residual learning process.

3.3 Loss Function

Since rain streaks are blend with object edges and background scene, it is hard to distinguish between rain streaks and

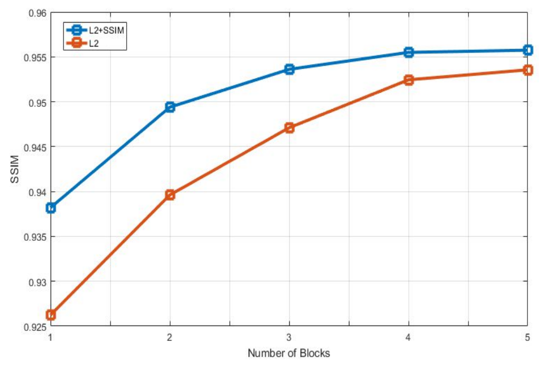


Figure 8: Comparison of Baseline-RR using different loss function.

objects' structure by simply optimizing ℓ_2 loss function. Per-pixel losses cannot capture perceptual difference between output and ground-truth images as human visual perception. A model with ℓ_2 loss tend to result in a blurred reconstruction. Therefore, for each block we adopt ℓ_2+SSIM loss [29] which can preserve global structure better as well as keeping per-pixel similarity. We minimize the combination of those loss functions in training stage. Figure 8 shows the effectiveness of the implementation of SSIM loss with ℓ_2 loss and proves that the supervision to intermediate outputs could benefit the whole model. Note that, the above experimental result is obtained by averaging 100 test images of dataset [8].

The overall loss function for block$_k$ is

$$\begin{aligned} L_{MSE_k} &= \frac{1}{N}\sum_{i=1}^{N}(\|f_k(X_i,W,b)-Y\|_2^2. \\ L_{SSIM_k} &= \frac{1}{N}\log(1.0/g(f_k(X_i,W,b),Y)+1e^{-4}). \\ L_{B_k} &= L_{MSE_k}+\lambda * L_{SSIM_k}. \end{aligned} \tag{1}$$

where N is the number of training rainy patches, k indicate the index of block. X, Y and X_i indicate rainy patches, corresponding clean patches and the input of block$_i$, respectively. W and b are the parameters in our model that need to tune. f denotes function mapping of each block. g denotes the function of SSIM. λ is the hyperparameter that balance the MSE loss and SSIM loss, we set λ as 1 via cross-validation that achieving satisfying result.

Note the overall ResGuideNet loss that containing $M+1$ loss function terms if the ResGuideNet contains M blocks

$$L = \frac{1}{M+1}(\sum_{i=1}^{M} L_{B_k} + L_{Merge}). \tag{2}$$

where L$_{Merge}$ is the final reconstruction merged by all previous intermediate outputs, with the same format of L_{B_k}.

3.4 Recursive Computation

As we mentioned above, the trade-off between the number of parameters and the model performance can be overcame using recursive strategy where the the nonlinear mapping operator is shared within each block. We adopt two convolutional operation in each recursive unit. We can write the structure of the input and output relationship in the t^{th} and $(t+1)^{th}$ recursion $(1 \le t < T)$ within each block as

$$\mathrm{x}^{\mathrm{t}} = \mathrm{g}\left(\mathrm{x}^{\mathrm{t}-1}\right), \quad \mathrm{x}^{\mathrm{t}+1} = \mathrm{g}\left(\mathrm{x}^{\mathrm{t}}\right), \tag{3}$$

where g indicates each recursive unit within one block.

However, as the recursions continue, the network depth increases, which introduces a severe gradient vanish problem that makes training difficult. To solve the gradient vanish problem as the recursion continues and to propagate information more easily, the output feature map of first feature extraction Conv+LReLU structure is fed into all subsequent outputs of recursive blocks. We can reformulate the structure as

$$\mathrm{x}^{\mathrm{t}} = \mathrm{g}\left(\mathrm{x}^{\mathrm{t}-1}\right)+\mathrm{x}^0, \quad \mathrm{x}^{\mathrm{t}+1} = \mathrm{g}\left(\mathrm{x}^{\mathrm{t}}\right)+\mathrm{x}^0, \tag{4}$$

The recursive computation is shown in bottom-left of Figure 2. We evaluate the benefit of transitioning from ResGuideNet without recursion (ResGuideNet-NRecur) to our adopted ResGuideNet using 5 recursions in each block. We show the quanlitative result in Figure 9. As is eveident, kernel reuse and propagate all information forward directly, from output of the first layer within each block, benefit the restoration process of image content.

Table 1: Performance of ResGuideNet$_5$ and ResGuideNet.

	ResGuideNet$_5$	ResGuideNet
SSIM	0.960	0.961
PSNR	29.92	30.11

3.5 Inter-block Ensemble

[3] first well studied the idea of ensemble learning which combines predictors instead of selecting a single predictor, ensemble learning has also introduced in neural networks to improve performance. [5] arranged a committee of neural networks in a simple voting scheme, and the final output predictions is based on the averaged result. Recently, [10] [13] using deep neural networks to deel with several computer vision tasks also use the ensemble technique.

Motivating by ensemble idea, we integrate all intermediate reconstruction of each block to form the best reconstruction which is aggragated by concatenation. As is shown in the bottom-right of Figure 2, the final reconstruction is obtained from the fusion of all intermediate reconstructions by a 1×1 convolution. Note that, we only use the merged result in section 4 since it is convenient for comparison in other sections. We refer the output with merging operation as ResGuideNet while the output of block$_i$ as ResGuideNet$_i$. We can observe an improved result from Table 1. The experiment is conducted on the test dataset of [8].

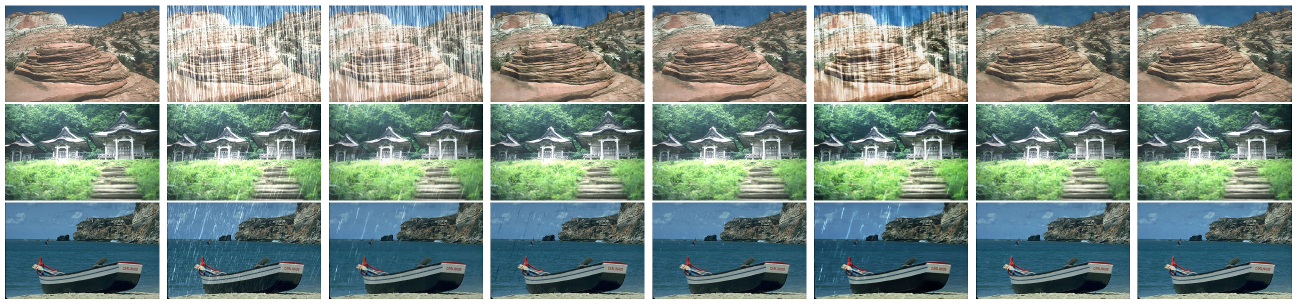

(a) Clean image (b) Rainy imag (c) GMM (d) DDN (e) JORDER (f) DID-MDN (g) Ours B_3 (h) ResGuideNet

Figure 10: Three results on synthetic images.

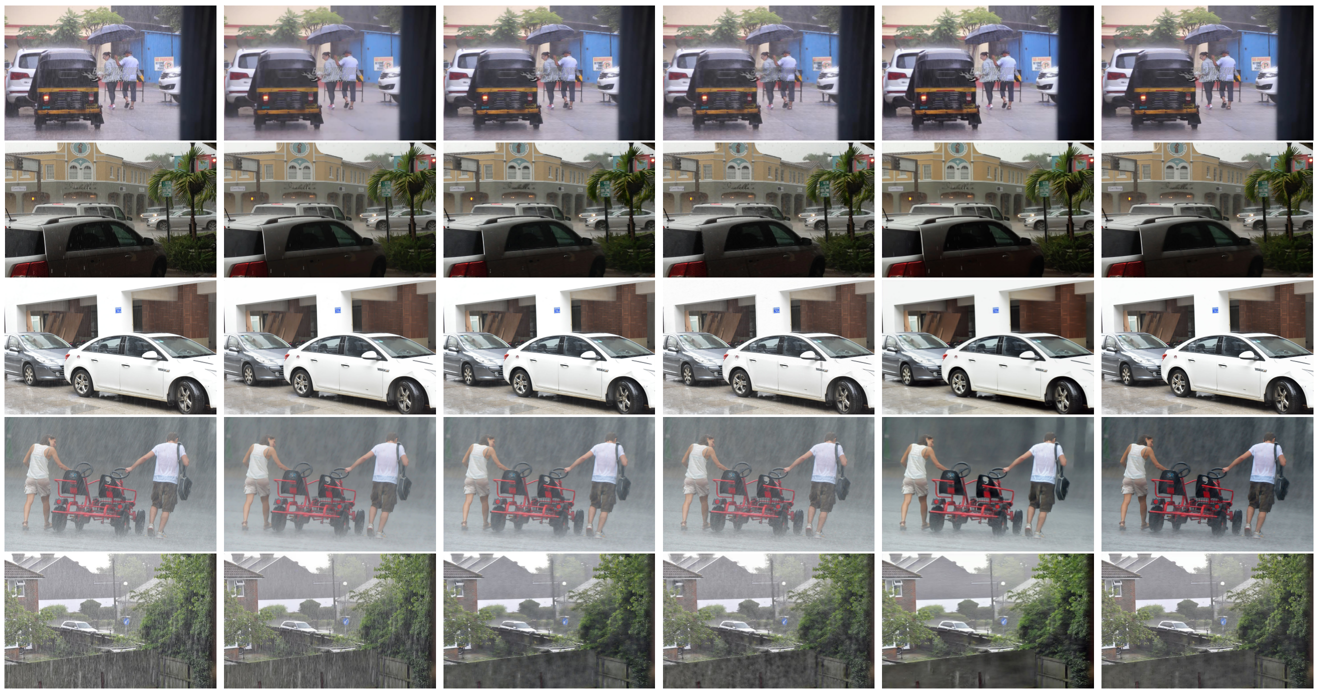

(a) Rainy image (b) GMM (c) DDN (d) JORDER (e) DID-MDN (f) ResGuideNet

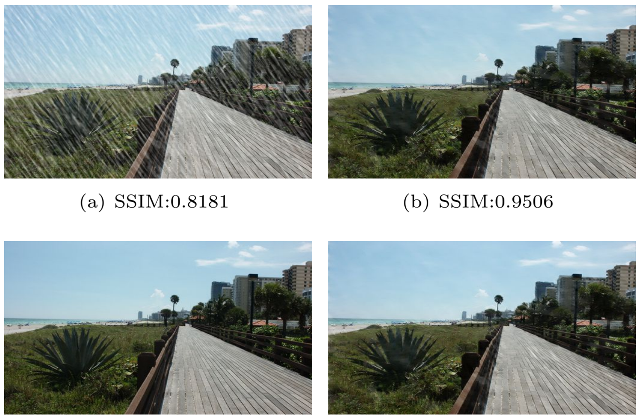
Figure 11: Five results on real-world rainy images with different rain magnitudes and shapes.

Table 2: Averaged SSIM and PSNR value on JORDER [30]'s dataset with their parameter number. Red indicates the best and blue indicates the second best performance.

	Rainy images	GMM [19]	DDN [8]	JORDER [30]	DID-MDN [32]	ResGuideNet$_3$	ResGuideNet
Rain100H	13.56/0.379	15.05/0.425	21.92/0.764	26.54/0.835	24.53/0.799	24.74/0.815	25.25/0.841
Rain100L	26.90/0.838	28.66/0.865	32.16/0.936	36.63/0.974	29.50/0.907	32.82/0.960	33.16/0.963
Rain12	30.14/0.855	32.02/0.910	31.78/0.900	33.92/0.953	28.25/0.906	29.19/0.936	29.45/0.938
Parameters	-	-	58,175	369,792	≈135,800	19,404	37,065

3.6 The Proposed Architecture

As discussed, the proposed ResGuideNet consists of repeated blocks. Each block includes several convolutional kernels and a global shortcut. The ResGuideNet propagates rain streak residual information from shallow blocks into deeper ones. The network architecture is shown in Figure 2. The final reconstruction is obtained by concatenating all intermediate



(a) SSIM:0.8181 (b) SSIM:0.9506

(c) SSIM:1 (d) SSIM:0.9585

Figure 9: Comparasion between Baseline and Baseline-RR, (a) rainy image, (c) clean image, (b) is the result of ResGuideNet without Recursion(ResGuideNet-NRecur), (d) is the result of ResGuideNet .

Table 3: Evaluation on DDN's synthetic dataset.

Metric	GMM	DDN	JORDER
PSNR/SSIM	26.33/0.838	31.12/0.926	32.95/0.921
Metric	DID-MDN	ResGuideNet$_3$	ResGuideNet
PSNR/SSIM	31.35/0.941	30.79/0.939	31.38/0.950

outputs and compressed them into the final rain-streak residual. ℓ_2+SSIM supervision is applied to guide each blocks and the final merged output.

Our basic network structure can be expressed as:

$$\begin{aligned} &F_1(X) = \hat{r}_1, \hat{y}_1 = X + \hat{r}_1. \\ &F_2(\hat{y}_1; \hat{r}_1) = \hat{r}_2, \hat{y}_2 = X + \hat{r}_2. \\ &F_3(\hat{y}_2; \hat{r}_2, \hat{r}_1) = \hat{r}_3, \hat{y}_3 = X + \hat{r}_3. \\ &\cdots\cdots \\ &F_5(\hat{y}_4; \hat{r}_4, \hat{r}_3, \hat{r}_2, \hat{r}_1) = \hat{r}_5, \hat{Y} = X + \hat{r}_5. \end{aligned} \tag{5}$$

where F indicates different blocks that consist of several convolutional layers using Leaky Rectified Linear Units. X and Y indicate rainy and clean pairs. $\hat{r}$ indicates negtive residual that is the output of each block. Block$_i$'s input is expressed as $\hat{y}_{i-1}$. Note that the left side of the semicolon indicates input of each block while the right side indicates residual features to guide each block. It is shown that more guidance provided when the blocks go deeper. $\hat{r}_1$, $\hat{r}_2$, $\hat{r}_3 \cdots \hat{r}_N$ all should be approximated to $Y - X$ in training stage as indicated in Equation 1, thus it is easier for deeper blocks to learn new rain streaks information with the guidance of rain streaks residual in shallow blocks.

4 EXPERIMENTS

We compare our algorithm with several state-of-the-art deep and non-deep techniques on synthetic and real-world datasets.

4.1 Implementation details

We train and test the algorithm using TensorFlow for the Python environment on a NVIDIA GeForce GTX 1080 with 8GB GPU memory. We use the Xavier method to initialize the network parameter and RMSProp for parameter learning. We select the initial learning rate to be 0.001. We set the size of training batch to 16. 50,000 iterations of training were required to train ResGuideNet. For all experiments we set the filter size to be 3×3 except the merge convolution and each convolution layer has 16 feature maps.

4.2 Dataset

Since clean and rainy image pairs from real-world is hard to obtain, four synthetic datasets are aviable for comparison. [30] provide $Rain100H$ and $Rain100L$ that is synthesized with heavy and light rain, each of them contains 100 images for test. The third dataset called $Rain12$ collected by [19] which contains 12 syhthetic images. The last one is provided by [8] constains 10K pairs of rainy/clean images with different orientations and magnitudes of rain streaks. For fair comparision, we conduct experiment that train the deep learning-based models and test them on $Rain100H$ and for $Rain100L$ datasets, the model trained on $Rain100L$ is used to test $Rain12$. Besides, we use the training dataset provide by [8] to train all models and test them on [8]'s test dataset. During training stage, We randomly generate 0.8 million rainy/clean patch pairs with size of 128×128 in the training stage.

4.3 Evaluation on Synthetic dataset

We train and test all the methods with the same dataset [30] [19]. SSIM [29] and PSNR are adopted to perform quantitative evaluations shown in Table 2 and Table 3. Our method has a comparable SSIM values with JORDER while outperforming other methods, which is in consistent with the visual result. We can observe the intermediate result in the third block (ResGuideNet$_3$) even has a decent result compared with other methods in Figure 10. However, our ResGuideNet contains far fewer parameters than others and can be sliced into a smaller network to meet light rain condition with limited resources, potentially making ResGuideNet easily implemented in varying real-world applications.

4.4 Evaluation on Real-world dataset

In this section, we show that ResGuideNet trained on synthetic training data still works well on real-world application. We implement other methods according to their optimal setting. Figure 11 shows visual results on real-world rainy images. Since no ground truth exists, we only show their qualitative result. As shown, ResGuideNet generate a less blurred result and have promising results on multiple kind of rain streaks.

Table 4: Running time of different methods.

	GMM	DDN	JORDER	DID-MDN	ResGuideNet$_3$	ResGuideNet
CPU	1990	1.51	295	4.20	1.26	3.15
GPU	-	0.16	0.18	0.14	0.06	0.11

4.5 Running Time

To illustrate the efficiency of implementation of ResGuideNet in practical application, we show the average running time of 100 test images in Table 4, all the test are conducted with a 500×500 rainy image as input. The GMM is non-deep method that is run on CPUs according to the provided code, while other deep-based methods are tested on both CPU and GPU. All experiments are performed with the same environment described in implementation details. The GMM has the slowest running time since it has complicated inference at test time. Our method has a fast computational time on GPU compared with other methods. In a light rain condition, we can use the third block as final output for testing that has a even faster running time. This experiment shows ResGuideNet has a promising practical value.

5 EXTENSION

5.1 Image Denosing

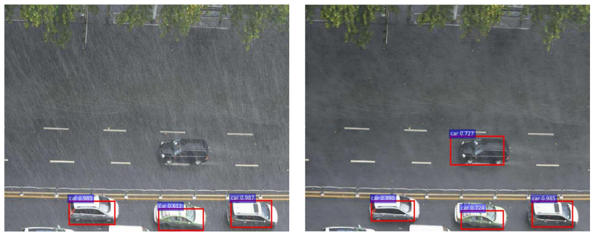
In this section, we show more evaluations for other general image processing tasks. We trained our ResGuideNet and [34] with the train and val set of berkeley segmentation dataset 500(BSD500). We use the training set which contains 300 images for training and the test set of BSD500 contains 200 images for testing. We apply Gaussian noise with different standard deviation to both train and test datasets. The averaged SSIM is shown in Table 5. This experiment demonstrates that ResGuideNet can generalize to similar image restoration problems.

Table 5: Denosing results.

	σ=10	σ=30	σ=70
DnCNN [34]	0.968	0.912	0.818
ResGuideNet	0.963	0.914	0.831

5.2 Pre-processing for high-level vision

Most exsting models for high-level tasks is trained with a well scenario, the performance will be degraded in rainy conditions since rain streaks block and blur the key structure of objects, Figure 12 show a case that under heavy rainy condition, the pre-trained Faster R-CNN [23] model trained on a well condition of KITTI dataset that failed to capture some objects and produce a low recognition confidence. We incorporate our ResGuideNet as a pre-process model for the Faster R-CNN, the detection performance has a great improvement over the naive Faster R-CNN input with a degraded image. We also



(a) Rainy image (b) Our result

Figure 12: An example of detection.

add experiments on vehicle detection before and after deraining using our model on both synthetic dataset and real-world rainy images, the experiments conducted on 480 synthetic rainy images from KITTI dataset and 20 real-world rainy images with manually annotation. The AP improves from 0.725 to 0.833 and from 0.591 to 0.657, respectively.

6 CONCLUSION

We presented the ResGuideNet, a novel convolutional network architecture for single image deraining which is easy to implement in a number of practical applications. We build our model with several deraining sub-networks in a cascaded manner. By propagating negative residuals in shallow blocks to deeper ones, the deeper blocks effectively extract new information of negative rain streak residuals to generate rain residual in a coarse to fine fashion. The final reconstruction takes all intermediate outputs into account to leverage more informations across blocks which can be viewed as ensemble learning. With our proposed architecture, ResGuideNet has $37K$ and ResGuideNet$_3$ has less than $20K$ parameters while still achieving good performance. For different rain conditions and computational resources, we can detach ResGuideNet into a smaller size can still achieve decent reconstruction. Moreover, extensive experiments have shown that our ResGuideNet can generalize to other low-level tasks and has potential value for high level vision problems.

7 ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China grants 61571382, 81671766, 61571005, 81671674, U1605252, 61671309 and 81301278, Guangdong Natural Science Foundation grant 2015A030313007, Fundamental Research Funds for the Central Universities grants 20720160075 and 20720180059, the CCF-Tencent Open Fund, and the Natural Science Foundation of Fujian Province of China (No.2017J01126)

REFERENCES

[1] Peter C Barnum, Srinivasa Narasimhan, and Takeo Kanade. 2010. Analysis of Rain and Snow in Frequency Space. *International Journal of Computer Vision* 86, 2-3 (2010), 256.

[2] Jérémie Bossu, Nicolas Hautière, and Jean-Philippe Tarel. 2011. Rain or snow detection in image sequences through use of a histogram of orientation of streaks. *International journal of computer vision* 93, 3 (2011), 348–367.

[3] Leo Breiman. 1996. Stacked regressions. *Machine learning* 24, 1 (1996), 49–64.

[4] Bolun Cai, Xiangmin Xu, Kui Jia, Chunmei Qing, and Dacheng Tao. 2016. Dehazenet: An end-to-end system for single image haze removal. *IEEE Transactions on Image Processing* 25, 11 (2016), 5187–5198.

[5] Harris Drucker, Corinna Cortes, L. D. Jackel, and Yann Lecun. 1989. Boosting and Other Ensemble Methods. *Neural Computation* 6, 6 (1989), 1289–1301.

[6] David Eigen, Jason Rolfe, Rob Fergus, and Yann LeCun. 2013. Understanding deep architectures using a recursive convolutional network. *arXiv preprint arXiv:1312.1847* (2013).

[7] Xueyang Fu, Jiabin Huang, Xinghao Ding, Yinghao Liao, and John Paisley. 2017. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing* 26, 6 (2017), 2944–2956.

[8] Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. 2017. Removing Rain from Single Images via a Deep Detail Network. In *CVPR*.

[9] Kshitiz Garg and Shree K. Nayar. 2004. Detection and Removal of Rain from Videos. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European Conference on Computer Vision*. Springer, 630–645.

[12] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q Weinberger. 2017. Multi-scale dense convolutional networks for efficient prediction. *arXiv preprint arXiv:1703.09844* (2017).

[13] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Vol. 1. 3.

[14] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*. Springer, 694–711.

[15] L. W. Kang, C. W. Lin, and Y. H. Fu. 2012. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing* 21, 4 (2012), 1742.

[16] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1637–1645.

[17] J. H. Kim, J. Y. Sim, and C. S. Kim. 2015. Video Deraining and Desnowing Using Temporal Correlation and Low-Rank Matrix Completion. *IEEE Transactions on Image Processing* 24, 9 (2015), 2658–2670.

[18] Boyi Li, Xiulian Peng, Zhangyang Wang, Jizheng Xu, and Dan Feng. 2017. An All-in-One Network for Dehazing and Beyond. *arXiv preprint arXiv:1707.06543* (2017).

[19] Yu Li, Robby T Tan, Xiaojie Guo, Jiangbo Lu, and Michael S. Brown. 2016. Rain Streak Removal Using Layer Priors. In *CVPR*. 2736–2744.

[20] Ming Liang and Xiaolin Hu. 2015. Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3367–3375.

[21] Yu Luo, Yong Xu, and Hui Ji. 2015. Removing Rain from a Single Image via Discriminative Sparse Coding. In *IEEE International Conference on Computer Vision*. 3397–3405.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*.

[24] Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang. 2016. Single image dehazing via multi-scale convolutional neural networks. In *European conference on computer vision*. Springer, 154–169.

[25] Varun Santhaseelan and Vijayan K. Asari. 2015. Utilizing Local Phase Information to Remove Rain from Video. *International Journal of Computer Vision* 112, 1 (2015), 71–89.

[26] Richard Socher, Brody Huval, Bharath Bath, Christopher D Manning, and Andrew Y Ng. 2012. Convolutional-recursive deep learning for 3d object classification. In *Advances in Neural Information Processing Systems*. 656–664.

[27] Ying Tai, Jian Yang, and Xiaoming Liu. 2017. Image Super-Resolution via Deep Recursive Residual Network. In *Proceeding of IEEE Computer Vision and Pattern Recognition*. Honolulu, HI.

[28] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. 2017. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4539–4547.

[29] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

[30] Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. 2017. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1357–1366.

[31] He Zhang and Vishal M Patel. 2018. Densely Connected Pyramid Dehazing Network. In *CVPR*.

[32] He Zhang and Vishal M Patel. 2018. Density-aware Single Image De-raining using a Multi-stream Dense Network. In *CVPR*.

[33] He Zhang, Vishwanath Sindagi, and Vishal M Patel. 2017. Image de-raining using a conditional generative adversarial network. *arXiv preprint arXiv:1701.05957* (2017).

[34] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. 2017. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing* 26, 7 (2017), 3142–3155.