A Decoder-free Transformer-like Architecture for High-efficiency Single Image Deraining

Xiao Wu , **Ting-Zhu Huang*** , **Liang-Jian Deng*** and **Tian-Jing Zhang**
University of Electronic Science and Technology of China, Chengdu, 611731
wxwsx1997@gmail.com, tingzhuhuang@126.com, liangjian.deng@uestc.edu.cn,
zhangtianjinguestc@163.com

Abstract

Despite the success of vision Transformers for the image deraining task, they are limited by computation-heavy and slow runtime. In this work, we investigate Transformer decoder is not necessary and has huge computational costs. Therefore, we revisit the standard vision Transformer as well as its successful variants and propose a novel *Decoder-Free Transformer-Like* (DFTL) architecture for fast and accurate single image deraining. Specifically, we adopt a cheap linear projection to represent visual information with a lower computational cost than previous linear projections. Then we replace standard Transformer decoder blocks with designed *Progressive Patch Merging* (PPM), which attains comparable performance and efficiency. Finally, DFTL could significantly alleviate the computation and GPU memory requirements through proposed modules. Extensive experiments demonstrate the superiority of DFTL compared with competitive Transformer architectures, e.g., ViT, DETR, IPT, Uformer, and Restormer. The code is available at https://github.com/XiaoXiao-Woo/derain.

1 Introduction

Image deraining is a classical problem in computer vision, which is highly desired in consumer photography and image processing. However, it is a challenging task since distant rain streaks usually combine with water droplets to form a veil over the backdrop, substantially reducing the visibility of the image. Recently, with the emergence of convolutional neural networks (CNNs), the state-of-the-art (SOTA) methods for image deraining are dominated by CNNs. However, the receptive field of convolution operations is limited, and CNNs usually have bulky structures to boost performance.

Transformer, first applied in natural language processing, is increasingly being used to replace CNNs in various vision tasks. Pioneering works in Transformer [Vaswani *et al.*, 2017], [Liu *et al.*, 2021] widely involve self-attention mechanisms as basic blocks to realize a strong feature representation ability. Most Transformer-based methods consist of three parts: a) patch generator; b) MLP; c) multi-head self-attention (MSA). In general, Transformer can be written in the following form:

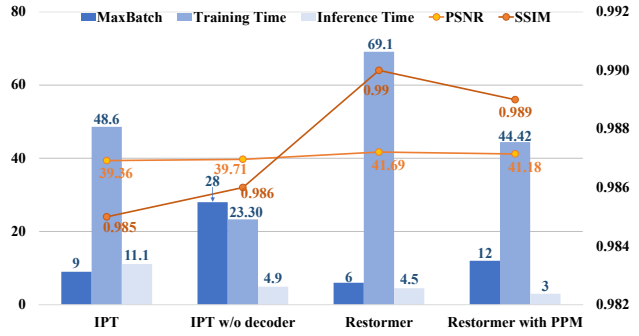

Figure 1: Ablation of decoders for two Transformer architectures. Transformer decoders have limited performance and higher complexity than a decoder-free Transformer-like architecture. Our methods achieve competitive results compared with standard Transformer with less GPU memory and faster runtime for image deraining.

$$
\begin{aligned}
&\mathbf{X} = \left[\mathbf{X}^1; \mathbf{X}^2; \cdots; \mathbf{X}^N\right], \\
&\mathbf{Q} = \mathbf{X}\mathbf{W}_Q^T, \mathbf{K} = \mathbf{X}\mathbf{W}_K^T, \mathbf{V} = \mathbf{X}\mathbf{W}_V^T, \\
&\mathbf{Z} = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}})\mathbf{V} + \mathbf{X}, \\
&\mathbf{Z} = \text{FFN}(\mathbf{Z}) + \mathbf{Z},
\end{aligned} \tag{1}
$$

where $\mathbf{X} \in \mathbb{R}^{N \times C}$ denotes the input features, N is the number of patches and C is channels. $\mathbf{W_Q}$, $\mathbf{W_K}$, $\mathbf{W_V}$ are weights of *query* ($\mathbf{Q}$), *key*($\mathbf{K}$) and *value* ($\mathbf{V}$) in the linear projections (LP). FFN$(\cdot)$ is the feed-forward network. The complete Transformer is achieved by stacking Eq. 1 to build Transformer encoder block (TEB) and Transformer decoder block (TDB). Therefore, the computational complexity of vanilla Transformer is $\mathcal{O}(N^2 + NC)$ that is more cumbersome than CNNs. Meanwhile, we could build cross-covariance attention (XCA) by matrix multiplication $\mathbf{V}^T\mathbf{K}\mathbf{Q}^T$, which is $\mathcal{O}(NC + C^2)$. The computational complexity is quadratic with the channels on the input images.

By analyzing previous works such as IPT [Chen *et al.*, 2021], Uformer [Wang *et al.*, 2021b], and Restormer [Za-

* Corresponding authors.

mir *et al.*, 2021] (please refer to Fig. 1), we find redundancy that they just simply apply TEB-to-TEB Transformer as the encoder-decoder architecture and decoders perform feature interaction across inputs of encoders rather than encoders and TDB, as shown in Fig. 2 (a). In other words, these decoders actually result in a slight or no improvement in their performance, but with high costs in terms of the parameter number, FLOPs, and training/inference time. It motivates us to build a decoder-free Transformer-like (DFTL) framework.

In this paper, we first adopt a cheap linear projection (LP) to generate intrinsic feature maps with lower complexity. Then, we introduce the Transformer encoder block (TEB) to represent desired features. Finally, Progressive Patch Merging (PPM) is proposed to restore rich features with different spatial resolution representations. Meanwhile, the architecture is still under the self-attention mechanism, which attains the powerful contextual representation ability of the vanilla Transformer. The advances of DFTL can be summarized as 1) **Efficient Transformer:** propose cost-effective operations for better results. 2) **Flexible structure:** a scalable framework for the restoration of rainy images.

The contributions of this work are as follows:

1. We propose a new DFTL architecture that could achieve competitive performance with less GPU memory and hold satisfying complexity/performance trade-offs compared to other self-attention-based techniques.
2. We adopt a Cheap LP and MSA/XCA to capture the multi-scale contextual details. Based on these modules optimized for Transformer, we develop two variants named DFTL-W and DFTL-X, which consistently achieve comparable performance to the prior arts.
3. A new hybrid loss is designed for more effective training, which could favor the potential convergence and improve the final testing performance expectedly.

2 Related Works

In general, existing single image deraining methods can be roughly divided into two categories, i.e., optimization-based and deep learning-based methods.

2.1 Optimization-based Deraining Methods

Optimization-based deraining methods view the rainy image as components assembled with the background image $\mathcal{B}$ and the rain streaks $\mathcal{R}$. The whole process can be expressed as the following formula:

$$\mathcal{Y} = \mathcal{B} + \mathcal{R}. \tag{2}$$

To remove $\mathcal{R}$ and obtain a clean image $\mathcal{Y}$, several works are proposed for single image deraining by founding effective optimization models based on image priors, e.g., directional sparsity prior [Jiang *et al.*, 2019; Jiang *et al.*, 2017; Deng *et al.*, 2018]. With the development of deep learning, optimization-based techniques may be insufficient. They are only capable of dealing with particular situations.

2.2 Deep Learning-based Deraining Methods

CNN-based techniques. With the powerful representation and extraction ability of CNN, diverse CNN-based structures are designed to improve the performance of image deraining. In [Li *et al.*, 2018], the authors utilize dilated CNN and squeeze-and-excitation blocks to deal with the task of image deraining, obtaining superior outcomes compared to traditional optimization-based methods. PReNet given by [Ren *et al.*, 2019] is to deepen shallow ResBlock via recurrent operations progressively, achieving competitive results. Wang *et al.* introduce a convolutional dictionary learning mechanism to remove rain streaks [Wang *et al.*, 2020] effectively. In addition, Fu *et al.* firstly attempt to use the so-called graph convolution network to extract relation-aware features and exploit pixel-level global spatial relationship [Fu *et al.*, 2021], getting the SOTA deraining results. However, all the techniques mentioned above are based on the characteristics of CNN, which requires stacking several convolution blocks to enlarge the receptive field and improve performance, lacking long-range interactions and ignoring geometric details extraction.

Vision Transformer techniques. In [Dosovitskiy *et al.*, 2021], it introduces Vision Transformer (ViT) to treat input image as 16×16 words and attains excellent results on image recognition. Besides, IPT [Chen *et al.*, 2021] pretrains the model on ImageNet dataset and achieves SOTA on several low-level visual tasks. SwinIR [Liang *et al.*, 2021] builds a single way structure based on Swin Transformer to achieve image restoration. More recently, Restormer [Zamir *et al.*, 2021] designs Transformer with convolution projections and cross-covariance across feature channels, further improving the structures of Transformer.

3 Proposed Method

3.1 Pipeline

An overview of the DFTL structure is presented in Fig. 2 (b). Given an image $I \in \mathbb{R}^{H \times W \times 3}$, we first progressively divide it into 4-level patches by Patchify module. These multi-scale patches are $\{1, 1/2, 1/4, 1/8\}$ relative to H, W, termed as H^l, W^l, abbreviated as P^l standing for patch size P at the specific scale l by the following Eq. 3:

$$N = \left\lfloor \frac{(P - k + 2 \cdot pad)}{s} + 1 \right\rfloor, \tag{3}$$

where N is the number of patches, k is patch/kernel size, pad means padding, and s denotes stride. In what follows, we describe proposed modules: (a) Cheap Linear Projection (LP); (b) Transformer Encoder Block (TEB); (c) Progressive Patch Merging (PPM).

3.2 Cheap Linear Projection

Linear projection (LP) could represent patch-level features into long range dependencies. However, according to Eq. 1, we can find one of complexity of Transformer is LP. Thus, we sort out several classic LP variants and show Cheap LP in Fig. 3. Convolution LP embeds spatial relationships into feature maps based on strided convolution operations. ViT

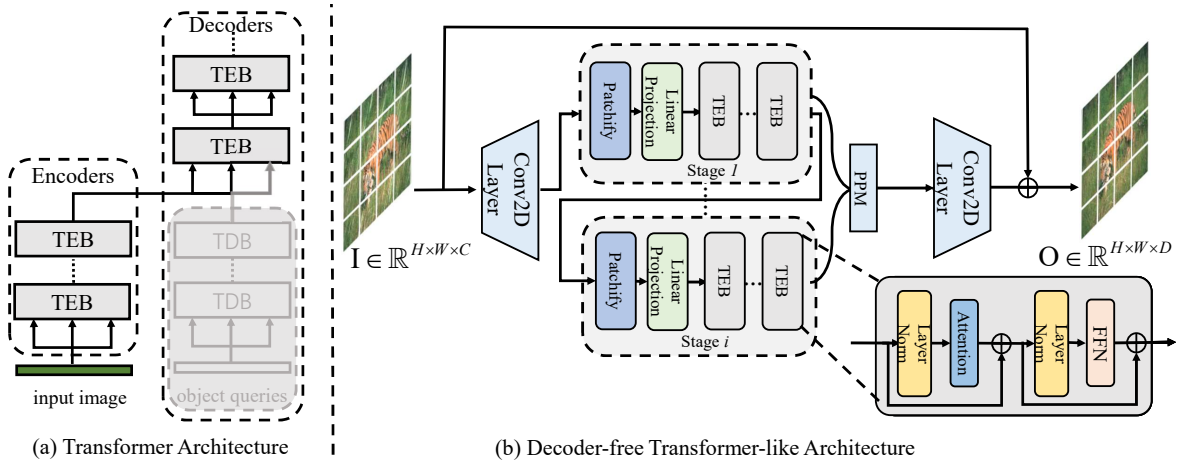

Figure 2: Comparisons of different architectures, where Conv2D layer is convolution, and TEB/TDB stands for Transformer encoder/decoder block. (a) Many vision Transformers employ encoder-decoder structure for various vision tasks. However, recent vision Transformers adopt TEB-to-TEB to construct encoder-decoder for restoration of rainy image. They do not calculate TDB by object queries. (b) A decoder-free Transformer-like architecture (DFTL). Progressive Patch Merging (PPM) is used to replace standard Transformer decoders. The input channel and the output channel are C and D, respectively.

LP projects the flattened patches and window based LP represents window based self-attention (W-MSA) along channel direction. Compared to depthwise convolution, they would occupy more memory and FLOPs. Hence, given the input $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$, $\mathbf{X}$ could be partitioned into patches $\{\mathbf{X}_p \in \mathbb{R}^{k\times k\times C}\}_{p=1}^{P}$ by Eq. 3, where $\mathbf{X}_p$ is pth patch ($p = 1, \cdots, P$). Cheap LP can be expressed as,

$$\mathbf{X}_p = \text{Concat}(\sum_{c=0}^{C}\sum_{i=0}^{k}\sum_{j=0}^{k}\mathbf{X}_p \cdot \mathbf{W}^{\mathbf{T}}, \mathbf{Y}_p), \tag{4}$$

where $\mathbf{W} \in \mathbb{R}^{Ck^2\times D}$ denotes learnable weights, $\mathbf{Y}_p$ is the output of ViT LP. As shown in Fig. 3 (d), Cheap LP generates patches with the same channels, then employs kernel size $k \times k$ of linearly depthwise operations to expand channels.

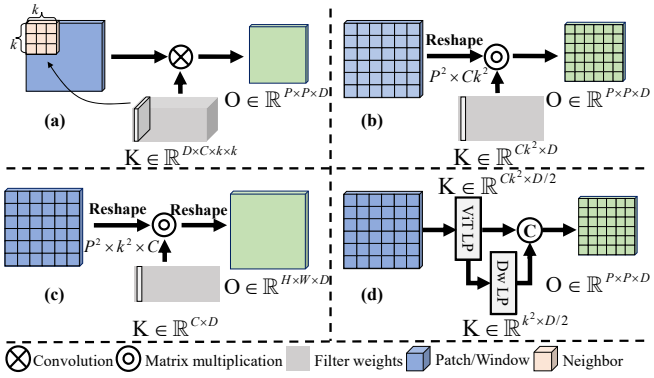

Figure 3: Flowchart of LP in PVT [Wang *et al.*, 2021a], ViT, SwinT and Cheap LP. (a) Convolution LP; (b) ViT LP; (c) Window-based LP; (d) Cheap LP. Cheap LP is used to extract features in DFTL. Note that the dimension of input is $H \times W \times C$ and LPs have different ways to share weights $\mathbf{K}$. P, k, C, D denote *patch size*, *kernel size (window size)*, *in_channel*, *out_channel*, respectively.

3.3 Transformer Encoder Block

Self-attention (SA) is the core of Transformer but is infeasible to GPU overheads. Thus, we analyze two SA mechanisms in this section, including 1) W-MSA through introducing one local window to consistent with the efficiency and performance, 2) XCA by channel correlation with a lower spatial cost.

Window Based Self-Attention (W-MSA). To reduce cost in spatial resolution, W-MSA encodes local pixel similarity in a window $M \times M$, whose computational complexity is $\mathcal{O}(M^2 + MC)$, which reduces the computational/memory overhead. With W-MSA followed by FFN module, TEB can be computed as follows,

$$\begin{aligned}\mathbf{Z}_p &= \text{W-MSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) + \mathbf{X}_p \\ &= \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}})\mathbf{V} + \mathbf{X}_p, \\ \mathbf{Y}_p &= \text{FFN}(\mathbf{Z}_p) + \mathbf{Z}_p,\end{aligned} \tag{5}$$

where $\mathbf{X}_p$ and $\mathbf{Y}_p$ denote the input and output of TEB. $\mathbf{Z}_p$ indicates the intermediate feature maps, and d is a scalar. DFTL-W is built by the sequential form of LP and W-MSA.

Cross-Covariance Attention (XCA). Considering the feature may have more channels, DFTL-W may produce limited performance because W-MSA only encodes feature vectors in each pixel. We introduce XCA to compute channel correlation along channel dimension:

$$\begin{aligned}\mathbf{Z}_p &= \text{XCA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) + \mathbf{X}_p \\ &= \mathbf{V}^T\,\text{Softmax}(\frac{\mathbf{K}^T\mathbf{Q}}{\sqrt{d}}) + \mathbf{X}_p,\end{aligned} \tag{6}$$

where $\mathbf{K}^T\mathbf{Q}$ is also termed as the attention matrix with $\mathbb{R}^{C\times C}$. The model with XCA is named as DFTL-X, which is the sequential form of TEB and LP.

Methods \ Datasets	*Rain12*		*Rain200L*		*Rain200H*		*DID-Data*		*DDN-Data*		*MaxBatch*
Metrics	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	-
Input	30.14	0.8553	26.71	0.8438	13.08	0.3733	23.63	0.7324	25.23	0.7901	-
DSC [Luo *et al.*, 2015]	30.07	0.8664	27.16	0.8663	14.73	0.3815	24.24	0.8279	27.31	0.8373	-
GMM [Li *et al.*, 2016]	32.14	0.9145	28.66	0.8652	14.50	0.4164	25.81	0.8344	27.55	0.8479	-
JCAS [Gu *et al.*, 2017]	33.10	0.9302	31.42	0.9173	14.69	0.4999	25.16	0.8509	26.81	0.8632	-
Clear [Fu *et al.*, 2017b]	31.15	0.9358	30.51	0.9361	13.90	0.7091	24.10	0.8518	25.86	0.8781	100
DDN [Fu *et al.*, 2017a]	32.71	0.9291	34.37	0.9578	25.99	0.8006	28.95	0.8619	29.64	0.8913	**240**
RESCAN [Li *et al.*, 2018]	36.63	0.9527	38.01	0.9796	27.46	0.8485	34.15	0.9294	32.87	0.9294	180
PReNet [Ren *et al.*, 2019]	36.54	0.9606	36.81	0.9767	27.78	0.8717	33.47	0.9252	32.24	0.9257	96
FBL [Yang *et al.*, 2020]	37.69	0.9651	39.02	0.9827	30.07	0.9021	34.26	0.9320	33.05	0.9334	8
RCDNet [Wang *et al.*, 2020]	37.59	0.9608	39.17	0.9885	30.24	0.9048	34.08	0.9532	33.04	0.9472	21
FuGCN [Fu *et al.*, 2021]	37.38	**0.9674**	39.61	0.9860	29.77	0.8991	34.37	**0.9620**	33.01	**0.9489**	32
IPT [Chen *et al.*, 2021]	37.12	0.9629	39.36	0.9850	29.41	0.8909	34.32	0.9354	33.21	0.9366	9
DFTL-W	37.60	0.9632	39.99	0.9873	30.02	0.9050	34.37	0.9574	33.27	0.9375	180
DFTL-X	**38.09**	0.9670	**41.27**	**0.9890**	**31.81**	**0.9271**	**34.73**	0.9604	**33.70**	0.9424	16
Ideal value	$+\infty$	1	$+\infty$	1	$+\infty$	1	$+\infty$	1	$+\infty$	1	$+\infty$

Table 1: Quantitative experiments evaluated on the five datasets. The best and the second best results are boldfaced and underlined.

3.4 Progressively Patch Merging (PPM)

It is redundant that decoders perform feature interaction across the *Query*, *Key* and *Value* from encoders. Considering generations of patch embeddings, we develop PPM to replace decoders of Transformer. Specifically, we use bilinear interpolation to upsample the patch-level features. Then we restore detailed information and shrink the channels to obtain the outputs by two convolution layers. In this process, we use skip-connection to achieve concatenation of patches progressively. The outputs of PPM are then added with the original inputs to remove rain streaks and produce clean images. The whole process is presented as follows:

$$\begin{aligned} \mathbf{F}^l &= \text{Upsample}(P^l \times P^l)(\mathbf{F}^{l-1}), \\ \mathbf{F}^l &= \text{Conv}(C^l,\ C^l/2)(\text{Concat}(\mathbf{F}^l)), \\ \mathbf{O} &= \text{Conv}(C^l,\ 3)(\mathbf{F}^l) + \mathbf{I}, \end{aligned} \tag{7}$$

where $\mathbf{F}^l$ represents l th-level feature maps, $\mathbf{I}$ and $\mathbf{O}$ are the input and output of our DFTL, respectively.

3.5 Hybrid Loss Function

Our work proposes a novel gradient-based hybrid loss function (GBHL) to achieve better results. We empirically present the Eq. 8, in which the significant numerical value is provided by SSIM [Wang *et al.*, 2004] or mean square error (MSE), etc. Mathematically, we impose a regularizer as a constraint for SSIM or MAE as follows:

$$\begin{aligned} \mathcal{L}_{ssim} &= \mathcal{L}_{ssim} \bigodot \sum_{k \neq ssim} |\frac{\mathbf{L_k}}{\|\mathbf{L_k}\|_F}|, \\ \mathcal{L}_{mse} &= \mathcal{L}_{mse} \bigodot \sum_{k \neq mse} |\frac{\mathbf{L_k}}{\|\mathbf{L_k}\|_F}|. \end{aligned} \tag{8}$$

Here $\bigodot$ represents hadamard product. $\|\mathbf{L_k}\|_F$ is set to $requires\_grad = False$. $\|\cdot\|_F$ is Frobenius norm, and $k = [ssim, mae, mse]$.

4 Experiments

In this section, we demonstrate the advantages of proposed method via comprehensive experiments on both synthetic and real datasets. In particular, we also compare our DFTL with other Transformer architectures to prove the efficiency of our methods. Please refer to the supplementary materials for more details, e.g., datasets, implementation details and discussion of model structures and complexity.

4.1 Comparison on Synthetic and Real Datasets

We evaluate our model on five synthetic datasets to compare quantitative results. Besides, we list the maximum batch (MaxBatch) of each model that can be trained simultaneously on single GPU. Also, we perform tests on two real scenarios.

Synthetic cases. The performance of all compared methods on five synthetic datasets is reported in Table 1. It can be observed that our models could exceed the most advanced techniques. Besides, we show visual comparisons on Rain200H, DDN, and DID datasets, see Fig. 4. For rainy streaks in different directions, our methods could preserve more details and attain better disentanglement from complex scenes. We also visualize the trade-off analysis between latency and performance among these deraining models in Fig. 5.

Real cases. To demonstrate the generalization ability, we perform the visual evaluation on the real-world rainy images in Fig. 6. It can be observed that DFTL-W/-X could remove more rainy streaks than other methods and have better visual qualities on the whole image.

4.2 Comparison with other Transformers

In this section, we perform a comparison with existing vision Transformers to verify the effectiveness of DFTL targeting the specific image deraining task. We select seven classic Transformer architectures, i.e., ViT_{256}, DETR [Carion *et al.*,

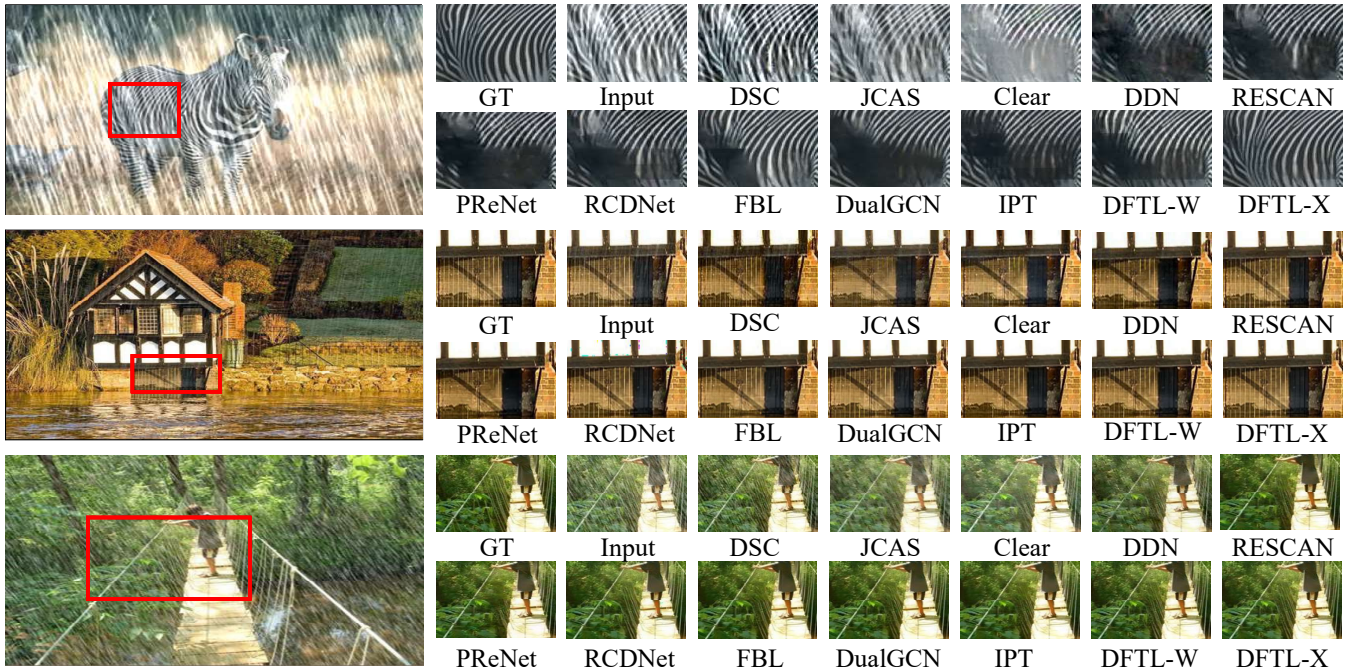

Figure 4: Visual comparisons on Rain200H, DDN, DID datassets with synthetic different rain streaks. The first row is from Rain200H dataset, the second/last row shows results on DDN/DID dataset.

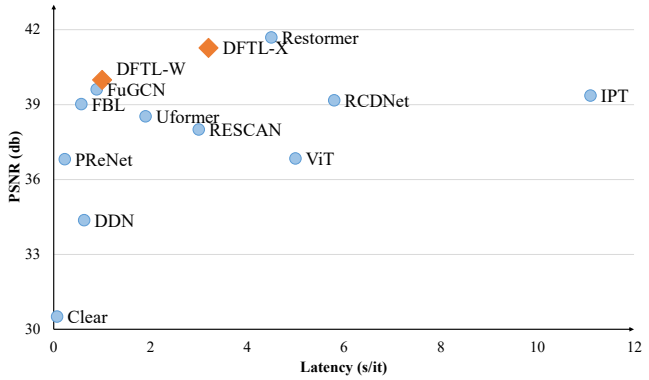

Figure 5: Comparison between latency and performance.

2020], DeformDETR [Zhu *et al.*, 2021], Uformer [Wang *et al.*, 2021b], IPT [Chen *et al.*, 2021] and Restormer [Zamir *et al.*, 2021]. All models are trained in the same framework with default settings as their original codes. In Table 2, we summarize the PSNR and SSIM of all outcomes produced by different architectures on Rain200L dataset. Besides, we report parameter number (Param) and FLOPs, demonstrating that our DFTL can significantly reduce computational costs. Furthermore, we show the comparisons on PSNR and latency in Fig. 5. Our methods could obtain better PSNR in an efficient manner.

4.3 Ablation Study

Ablation on Cheap LP. In Table 3, we compare different locations replacing LP with Cheap LP in DFTL-X over Rain200H dataset. Cheap LP can achieve competitive performance and high efficiency. On the one hand, benefiting from

Methods	PSNR	SSIM	Param	FLOPs
Input	26.71	0.8438	-	-
VIT_{256} [Dosovitskiy *et al.*, 2021]	36.84	0.9652	159.9M	15.3G
DETR [Carion *et al.*, 2020]	27.84	0.7144	39.4M	1.6G
DeformDETR [Zhu *et al.*, 2021]	25.64	0.7092	40.1M	1.5G
Uformer [Wang *et al.*, 2021b]	38.53	0.9816	146.7M	148.1G
IPT [Chen *et al.*, 2021]	39.36	0.9850	64.3M	18.0G
Restormer [Zamir *et al.*, 2021]	**41.69**	**0.9903**	35.9M	9.7G
DFTL-W	39.99	0.9873	**17.6M**	**1.8G**
DFTL-X	<u>41.27</u>	<u>0.9890</u>	29.3M	6.8G

Table 2: Comparison with various Transformer architectures.

DFTL-X	PSNR	SSIM	Param	FLOPs
Network + LP	31.76	0.9243	30.2M	6.9G
Patchify + Cheap LP	**31.81**	**0.9271**	29.3M	6.8G
Network + Cheap LP	31.56	0.9238	20.8M	4.6G

Table 3: Comparison of different locations of Cheap LP in DFTL-X.

Model	Sturcture	PSNR	SSIM	Latency	Memory	Param	FLOPs
SA	Single Scale	-	-	-	Crack	1.7G	229.9G
XCA		**41.39**	**0.9902**	6.4	14187M	60.9M	51.7G
SA	Multi-Scale	<u>41.30</u>	<u>0.9890</u>	5.1	14845M	161.8M	25.6G
W-MSA		40.47	0.9871	1.9	7459M	161.8M	5.3G
XCA		41.18	<u>0.9890</u>	3.0	5365M	30.2M	6.9G

Table 4: Comparison of different self-attention in DFTL-X.

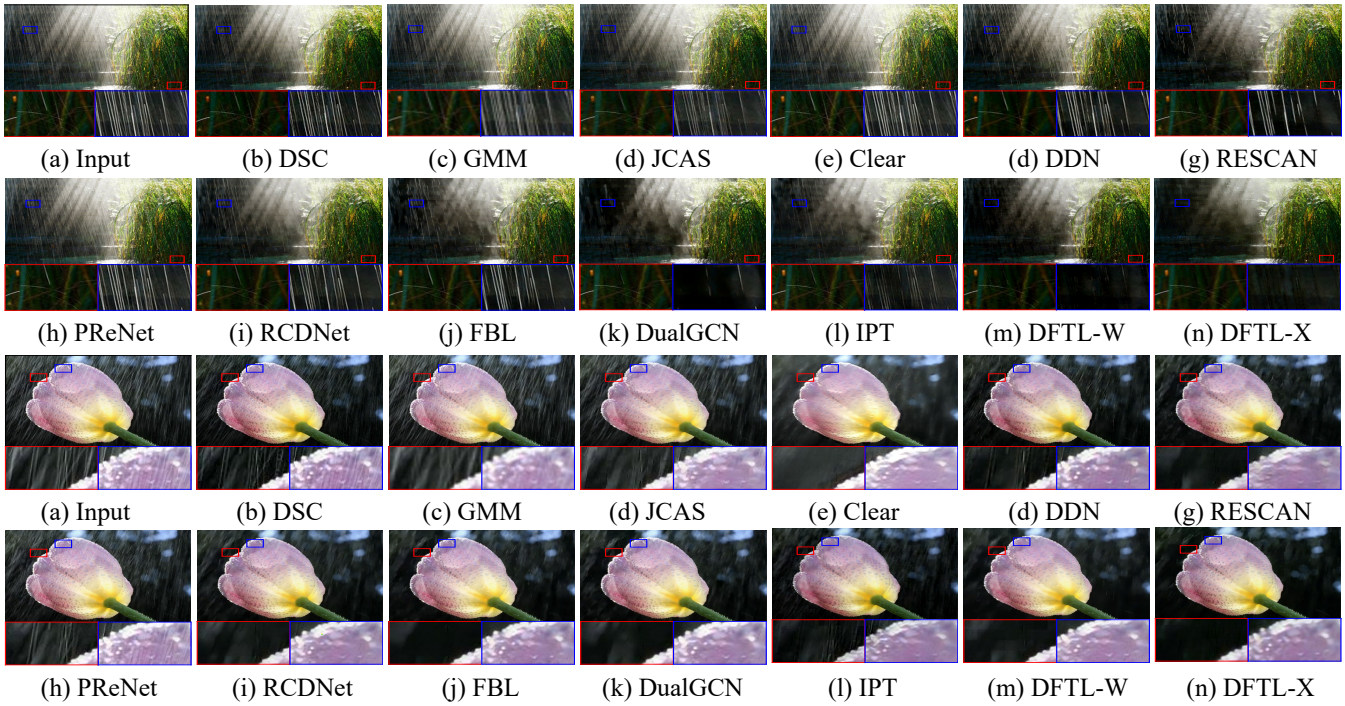

Figure 6: Visual comparisons on two real-world datasets obtained by [Fu *et al.*, 2017a], [Zhang *et al.*, 2017]

depthwise operation, Patchify with Cheap LP could generate more feature maps to improve performance with lower complexity. On the other hand, the network with Cheap LP leads to weak performance because of poor feature extraction.

Ablation on DFTL. We compare the effects of different self-attention in DFTL-X on Rain200L dataset. In Table 4, SA and XCA significantly surpass other compared counterparts. However, SA has higher computational loads and better results than XCA. W-MSA has the same parameter number as SA and larger memory than XCA. Compared to it, DFTL-W avoids the problem by employing the sequential form of LP and W-MSA (see more details in the supplementary material). Thus, it attains competitive results and handles larger MaxBatch with limited resources in Table 1.

Ablation on PPM. This part investigates the effects of conventional decoders and proposed PPM on two representative Transformers, i.e., IPT and Restormer. Since IPT is a single way architecture to restore images, PPM is used for multi-scale architecture and can't be directly integrated into IPT. Thus we only ablate the decoders of IPT. Restormer's decoders are replaced with PPM to validate the effectiveness. Through the comparisons in Fig. 1, decoders of Transformer have a slight performance gain but lead to colossal GPU memory occupations (2x$\sim$3x) and costs of latency ($\sim$1.5x). Thus, it is significant for an efficient Transformer to build a decoder-free Transformer-like architecture.

Ablation on Hybrid Loss. We compare the results between different weighted combinations and proposed GBHL. Due to the numerical differences between $\mathcal{L}_{MAE}$ and $\mathcal{L}_{SSIM}$ are about 10 times in the initial training phase, α is set to 10. Two groups shown in Table 5 are adopted to conduct a comparison on Rain200H. Experimental results show that our GBHL makes the highest value for both PSNR and SSIM.

Loss combinations	Weights $\{\alpha, \beta, \gamma\}$	PSNR	SSIM
MAE + 1-SSIM	$\{10, 1\}$	29.76	0.8955
MAE + MSE + 1-SSIM	$\{10, 1, 1\}$	29.77	0.8968
GBHL	-	30.02	0.9050

Table 5: The table shows the performance of DFTL-W for different combinations of weights evaluated on Rain200H dataset.

5 Conclusion

This paper presents a decoder-free Transformer-like architecture (DFTL) for image deraining to analyze popular Transformer architectures from a new perspective. It reveals that decoders are redundant to Transformer. Proposed modules are more computationally efficient compared with standard Transformer modules. The comparisons with several competitive Transformers show our methods have good feature representation ability at low computational costs. Moreover, we propose a novel gradient-based hybrid loss function to produce more reasonable results. Extensive experiments demonstrate DFTL can achieve the comparable to SOTA methods.

Acknowledgments

This research is supported by NSFC (12171072, 61876203, 61702083), Key Projects of Applied Basic Research in Sichuan Province (Grant No. 2020YJ0216), and National Key Research and Development Program of China (Grant No. 2020YFA0714001)

References

[Carion *et al.*, 2020] N Carion, F Massa, G Synnaeve, N Usunier, A Kirillov, and S Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.

[Chen *et al.*, 2021] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, pages 12299–12310, 2021.

[Deng *et al.*, 2018] Liang-Jian Deng, Ting-Zhu Huang, Xi-Le Zhao, and Tai-Xiang Jiang. A directional global sparse model for single image rain removal. *Applied Mathematical Modelling*, 59:662–679, 2018.

[Dosovitskiy *et al.*, 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[Fu *et al.*, 2017a] Xue-Yang Fu, Jia-Bin Huang, Zeng De-Lu, Yue Huang, Xing-Hao Ding, and John Paisley. Removing rain from single images via a deep detail network. In *CVPR*, pages 1715–1723, 2017.

[Fu *et al.*, 2017b] Xue-Yang Fu, Jia-Bin Huang, Xing-Hao Ding, Ying-Hao Liao, and John Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[Fu *et al.*, 2021] Xue-Yang Fu, Qi Qi, Zheng-Jun Zha, Yu-Rui Zhu, and Xing-Hao Ding. Rain streak removal via dual graph convolutional network. In *AAAI*, pages 1352–1360, 2021.

[Gu *et al.*, 2017] Shu-Hang Gu, De-Yu Meng, Wang-Meng Zuo, and Lei Zhang. Joint convolutional analysis and synthesis sparse representation for single image layer separation. In *ICCV*, pages 1717–1725, 2017.

[Jiang *et al.*, 2017] Tai-Xiang Jiang, Ting-Zhu Huang, Xi-Le Zhao, Liang-Jian Deng, and Yao Wang. A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors. In *CVPR*, pages 2818–2827, 2017.

[Jiang *et al.*, 2019] Tai-Xiang Jiang, Ting-Zhu Huang, Xi-Le Zhao, Liang-Jian Deng, and Yao Wang. Fastderain: A novel video rain streak removal method using directional gradient priors. *IEEE Transactions on Image Processing*, 28(4):2089–2102, 2019.

[Li *et al.*, 2016] Yu Li, Robby T Tan, Guo Xiao-Jie, Lu Jiang-Bo, and Michael S Brown. Rain streak removal using layer priors. In *CVPR*, pages 2736–2744, 2016.

[Li *et al.*, 2018] Xia Li, Jianlong Wu, Zhouchen Lin, Hong Liu, and Hongbin Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *ECCV*, pages 262–277, 2018.

[Liang *et al.*, 2021] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCV*, pages 1833–1844, 2021.

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021.

[Luo *et al.*, 2015] Yu Luo, Yong Xu, and Hui Ji. Removing rain from a single image via discriminative sparse coding. In *ICCV*, pages 3397–3405, 2015.

[Ren *et al.*, 2019] Dong-Wei Ren, Wang-Meng Zuo, Qing-Hua Hu, Peng-Fei Zhu, and De-Yu Meng. Progressive image deraining networks: A better and simpler baseline. In *CVPR*, pages 3932–3941, 2019.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeuIPS*, page 6000–6010, 2017.

[Wang *et al.*, 2004] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[Wang *et al.*, 2020] Hong Wang, Qi Xie, Qian Zhao, and De-Yu Meng. A model-driven deep neural network for single image rain removal. In *CVPR*, pages 3100–3109, 2020.

[Wang *et al.*, 2021a] Wen-Hai Wang, En-Ze Xie, Xiang Li, Deng-Ping Fan, Kai-Tao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[Wang *et al.*, 2021b] Zhen-Dong Wang, Xiao-Dong Cun, Jian-Min Bao, and Jian-Zhuang Liu. Uformer: A general u-shaped transformer for image restoration. *arXiv preprint arXiv:2106.03106*, 2021.

[Yang *et al.*, 2020] Wenhan Yang, Shiqi Wang, Dejia Xu, Xiaodong Wang, and Jiaying Liu. Towards scale-free rain streak removal via self-supervised fractal band learning. In *AAAI*, volume 34, pages 12629–12636, 2020.

[Zamir *et al.*, 2021] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. *arXiv preprint arXiv:2111.09881*, 2021.

[Zhang *et al.*, 2017] He Zhang, Sindagi Vishwanath, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[Zhu *et al.*, 2021] Xi-Zhou Zhu, Wei-Jie Su, Le-Wei Lu, Bin Li, Xiao-Gang Wang, and Ji-Feng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.