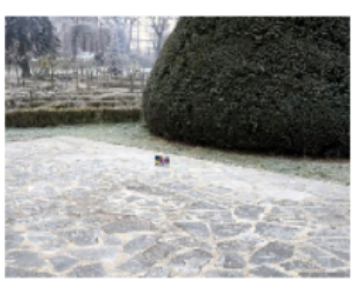
基于Vision Transformer的去雾算法研究与实现

李昕 2201821 自然语言处理大作业

> 本次实验是**基于已有模型代码Uformer（Uformer: A General U-Shaped Transformer for Image Restoration (CVPR 2022)）进行改进的**
>
> 主要针对Uformer中出现的：1.注意力操作复杂度较高；2.对数据集没有重复利用 问题进行解决
>
> > **针对每个问题，都进行了改进，并附有相应的结果：**
> >
> > 问题1的改进是用到了Informer提出的概率系数自注意力机制，**虽然恢复效果有所降低，但是大幅减少了自注意力复杂度**；
> >
> > 问题2的改进使用到了对比正则化**让恢复的图像接近无雾图像，并远离带雾图像**，以达到更好的恢复结果
>
> **详细的运行流程在第2章描述；实验结果以及消融实验分别在第3章和第4章描述；而在研究过程中，还发现了修改后的不足之处，在第5章有描述**
>
> **数据集的下载地址（第1章），对数据集的预处理（第2章） 以及 训练好的模型权重（第3章）都有详细的操作解释**

> 在我提交的代码中，所有的关键代码都有详细的注释，包括但不限于：
>
> **详细的维度注释，每个关键代码块的具体含义，函数的输入输出，特殊类和函数的输入输出的示例**，这样在阅读的时候会特别方便

1. 数据集




NH-HAZE --- IEEE CVPR 2020 NTIRE 研讨会


Dense-HAZE --- IEEE CVPR 2019 NTIRE研讨会

1.1 NH-HAZE

数据集下载： https://competitions.codalab.org/competitions/22236#participate-get_data

> Train：1-40；Test：41-45

我们引入了NH-HAZE，一个非均匀的真实数据集，有成对真实的模糊和相应的无雾图像。因此，非均匀雾霾数据集的存在对于图像去雾场是非常重要的。

它代表第一个真实的图像去模糊数据集与非均匀的模糊和无模糊（地面真实）配对图像

为了补充之前的工作，在本文中，我们介绍了NH-HAZE，这是第一个具有非均匀模糊和无雾（地面真实）图像的真实图像去模糊数据集。

1.2 NTIRE 2019

DENSE-haze是一个真实的数据集，包含密集（均匀）模糊和无烟雾（地面真实）图像

官方地址：

https://data.vision.ee.ethz.ch/cvl/ntire19/#:~:text=Datasets%20and%20reports%20for%20NTIRE%202019%20challenges

https://data.vision.ee.ethz.ch/cvl/ntire19//dense-haze/

另一个下载地址：

https://www.kaggle.com/rajat95gupta/hazing-images-dataset-cvpr-2019?select=GT

> Train：1-45；Test：51-55

2. 模型运行过程

2.0 模型介绍

> 在文件夹 `/Uformer_ProbSparse/` 下存放模型代码

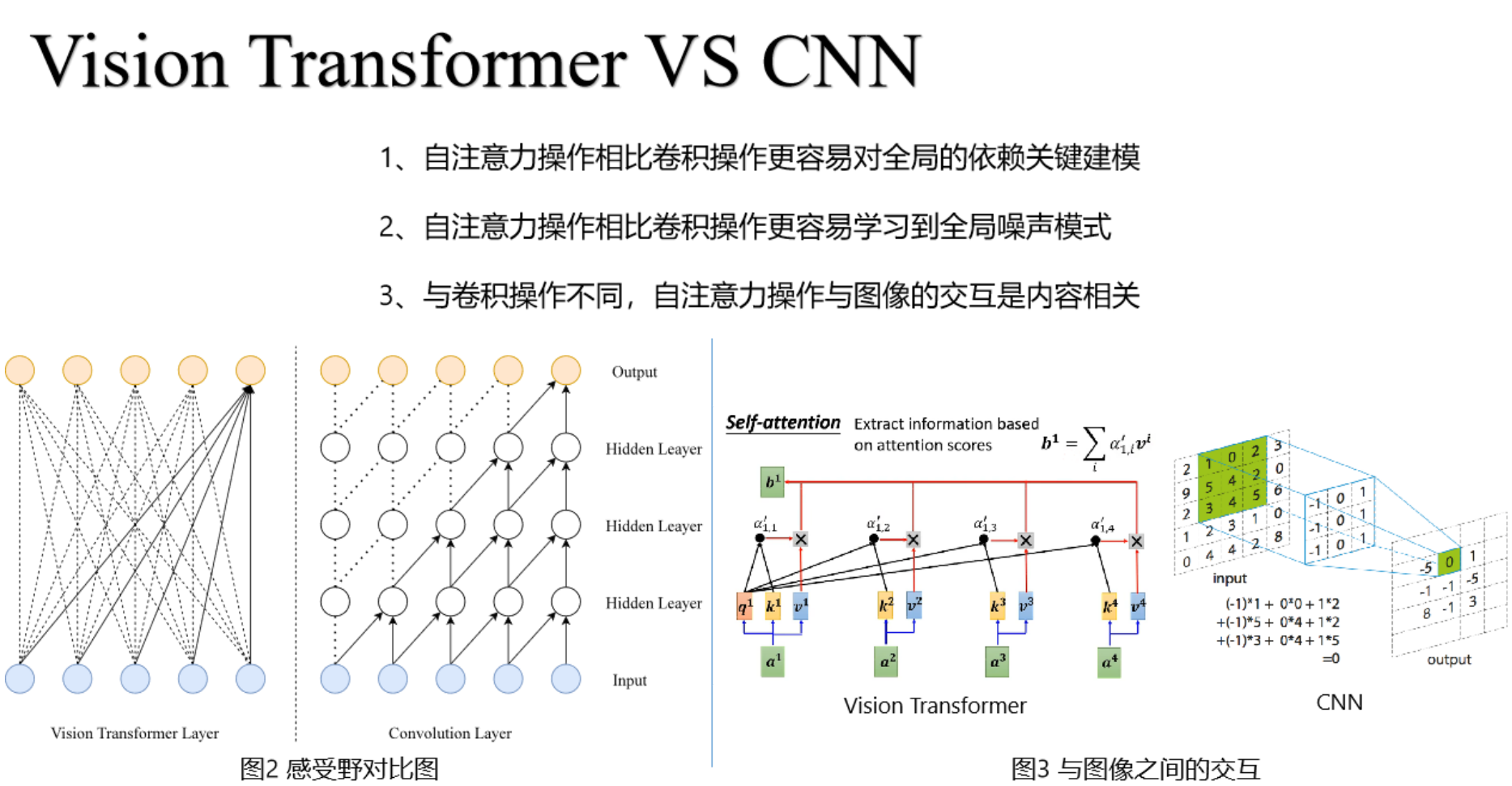


图2 感受野对比图

图3 与图像之间的交互

研究目标

2021年11月，在计算机视觉的Low-level中，相继提出了基于Vision Transformer的IPT模型，**Uformer模型**[1]；因模型在图像去噪，超分辨率等任务上有着极佳的性能，而轰动了计算机视觉领域。

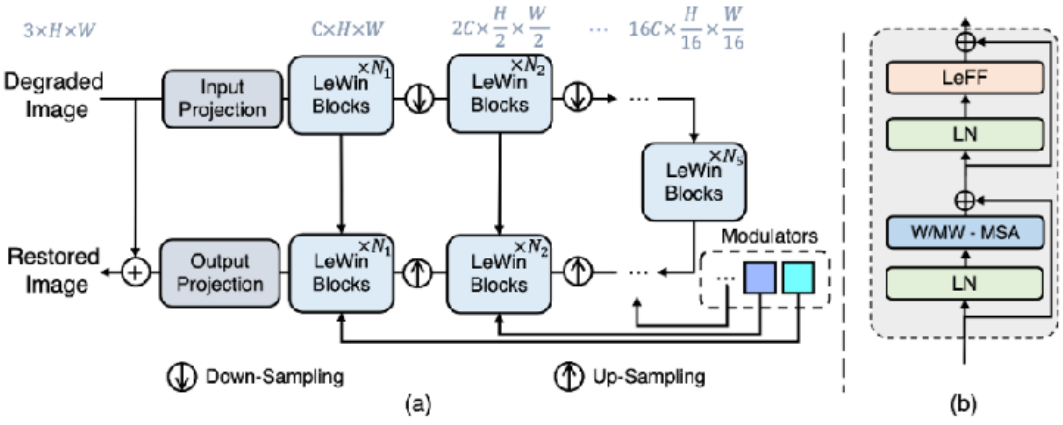


图4 Uformer模型

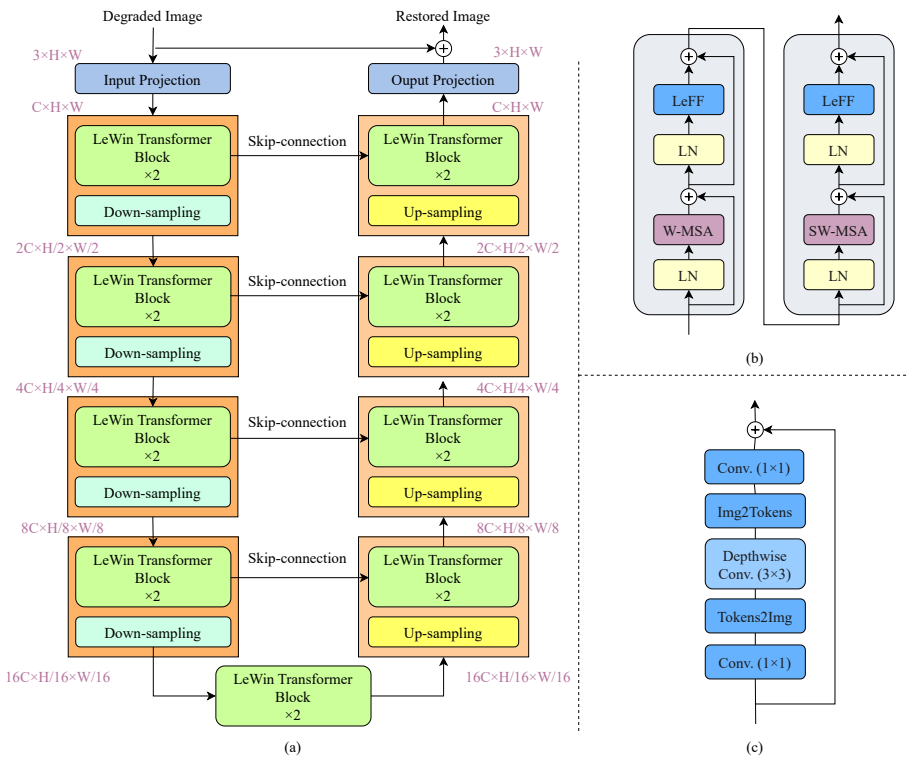
Uformer模型引入后问题：

1. 自注意力机制按照标准的 Transformer 架构会计算所有像素之间的自注意权重，**导致模型复杂度是对输入像素数量的二次计算代价**，这种开销在较大特征图输入时是巨大的，需要减少复杂度

2. 绝大部分的去雾模型仅仅让恢复的图像不断接近无雾图像，**该策略并未充分利用数据集。**

参考文献：

[1] Wang Z, Cun X, Bao J, et al. Uformer: A General U-Shaped Transformer for Image Restoration. arXiv preprint arXiv:2106.03106, 2021.



> 参考代码：https://github.com/ZhendongWang6/Uformer

问题1 – 自注意力机制复杂度高

为了减少自注意力机制复杂度使用长时间序列领域的Informer模型提出的**概率稀疏自注意力机制**[2] --- AAAI'21 Best Paper

Algorithm 1 概率稀疏自注意力

Require: Tensor $Q \in \mathbb{R}^{L_Q \times C}$，$K \in \mathbb{R}^{L_K \times C}$，$V \in \mathbb{R}^{L_V \times C}$，$L_Q = L_K = L_V = M^2$ // 输入的维度，M²表示像素个数，C表示像素维度

1: 设置超参数：采样因子c，$u = c \ln L_Q$和$U = L_Q \ln L_K$

2: 为了选取了U个点积对计算$\bar{M}(q_i, K)$，随机从K中选择$\ln L_K$个 key 组成$\bar{K}$

3: 计算Q与$\bar{K}$的注意力分数$\bar{S} = Q\bar{K}^T/\sqrt{d}$，得到的结果以行为单位

4: 计算度量值$\bar{M}(q_i, K) = \max_j \left\{\frac{q_i k_j^T}{\sqrt{d}}\right\} - \frac{1}{L_K}\sum_j^{L_K} \frac{q_i k_j^T}{\sqrt{d}} = \max(\bar{S}) - \mathrm{mean}(\bar{S})$ // **KL 散度推导的最大平均测量值来衡量点积对重要性**

5: 从$\bar{M}$中选择前u个 query 作为$\bar{Q}$

6: 对于选择的前u个 query 对应的自注意力，计算为$S_0 = \mathrm{softmax}(\bar{Q}K^T/\sqrt{d}) \cdot V$ // 选取部分(q, k)点积对计算自注意力

7: 其他 query 对应的自注意力，计算为$S_1 = \mathrm{mean}(V)$

8: 根据原始的 query 所在行对自注意力结果进行排列得$S = \{S_0, S_1\}$

Ensure: 自注意力特征图S

表1 概率稀疏自注意力机制与标准自注意力机制复杂度对比

方法	时间复杂度	空间复杂度
概率稀疏自注意力	$O(L_K \ln L_Q)$	$O(L_K \ln L_Q)$
标准的自注意力	$O(L_K L_Q)$	$O(L_K L_Q)$

[2] Zhou H, Zhang S, Peng J, et al. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting //Proceedings of the AAAI Conference on Artificial Intelligence. Menlo Park: AAAI, 2021.

> 参考代码：https://github.com/zhouhaoyi/Informer2020

问题2 – 并未充分利用数据集

为了进一步为了对数据集充分利用，**不仅让恢复的图像接近与无雾图像，还让恢复图像远离带雾图像**
我们使用对比正则化(Contrastive Regularization，CR)[3]来对参数进行约束 --- CVPR2021

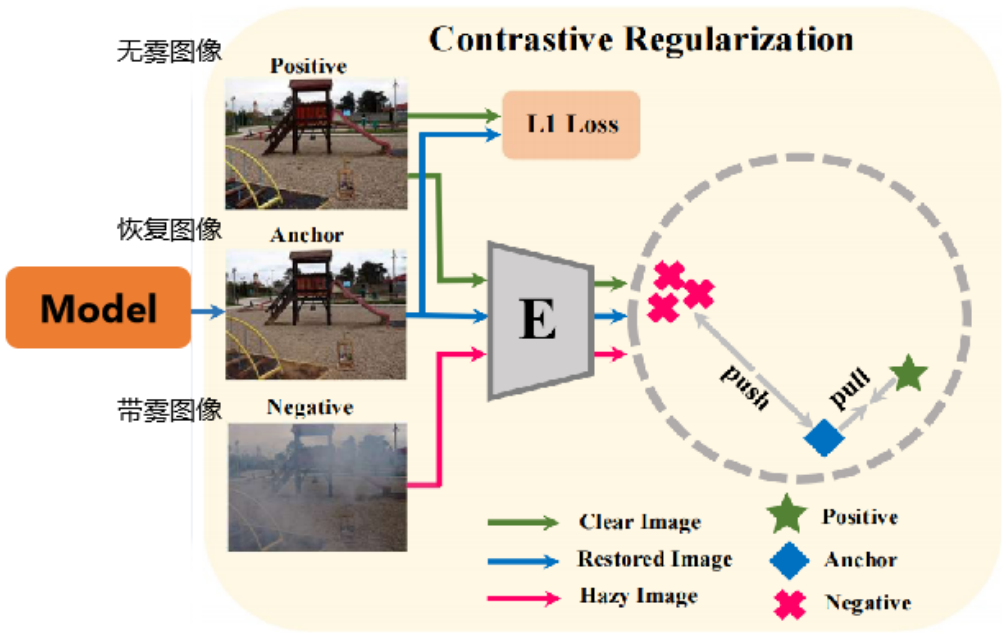


图5 对比正则化过程

参考文献：
[3] Wu H, Qu Y, Lin S, et al. Contrastive Learning for Compact Single Image Dehazing //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2021.

> 参考代码：https://github.com/GlassyWu/AECR-Net

2.1 预处理数据 --- 把训练数据图像切分成大小为256*256的小图

下载数据集存放在：

`/home/dell/桌面/TPAMI2022/Dehazing/#dataset/NH_haze/`

内含两个文件夹：`train` `test`

对训练数据集处理：

```
python3 generate_patches_SIDD.py --src_dir /home/dell/桌面/TPAMI2022/Dehazing/#dataset/NH_haze/train --tar_dir /home/dell/桌面/NLP/Datasets/NH-HAZE/train_patches
```

2.2 训练代码My_train.py

```
python3 ./My_train.py --arch Uformer --nepoch 270 --batch_size 32 --env My_Infor_CR --gpu '1' --train_ps 128 --train_dir /media/dell/fd6f6662-7e38-4427-80c6-0d4fb1f0e8b9/work_file/NLP/Datasets/NH-HAZE/train_patches --val_dir /media/dell/fd6f6662-7e38-4427-80c6-0d4fb1f0e8b9/work_file/NLP/Datasets/NH-HAZE/test_patches --embed_dim 32 --warmup
```

如果要继续对模型进行训练：`--pretrain_weights` 设置预训练权重路径，我的模型预训练权重在My_best_model文件夹下，以数据集划分不同预训练权重

并添加参数 `--resume`

训练所有参数设置在option.py文件种，主要的参数含义：

- `--train_ps` 训练样本的补丁大小，默认为128，指多大的patches输入到模型中
- `--train_dir` `--val_dir` 训练和测试文件夹，文件夹下包含两个文件夹gt和hzay，分别包含无雾图片集和带雾图片集
- `--batch_size` 设置Batch_size，默认为3
- `--is_ab` **是否使用n a对比损失，默认为False（使用）
- `--w_loss_vgg7` 对比损失使用的权重，默认为1
- `--w_loss_CharbonnierLoss` CharbonnierLoss 所占权重，默认为1**

2.3 测试代码test_long_GPU.py和预训练权重

> 训练权重：
>
> 链接：https://pan.baidu.com/s/1a1YPTGSNa0R6I-qiTNir0A
> 提取码：y422
>
> 模型预训练权重：将百度网盘中的 `Uformer_ProbSparse/My_best_model` 文件夹放到 `Uformer_ProbSparse` 文件夹下，里面包含4大数据集下的权重

```
python3 ./test_long_GPU.py
```

测试流程：

在My_train.py文件中，为了训练速度考虑，我们是在每个patch上进行的测试，但patch上测试结果不等于在整图上测试的结果，因此该文件是对模型在整图上结果进行测试，论文中的结果与该测试结果一致

由于代码的特殊设置，需要让输入的图片的长和宽为 `--train_ps` 的整数倍，如果不够足，则要进行扩展

主要参数解释：

- `--input_dir` **设置测试的文件夹，文件夹下包含两个文件夹gt和hzay，分别包含无雾图片集和带雾图片集**
- `--train_ps` 训练样本的补丁大小，默认为128，指多大的patches输入到模型中
- 代码中的: L表示图像需要拓展长和宽为多大

  例如：输入是1200 * 1600，patch size = 128时，L = 1664

  L需要为128倍数，且要大于输入图像的长和宽，需要根据输入图像进行调整，例如：NH-HAZE数据集上的为L = 1664

3. 实验结果

NH-HAZE和Dense-HAZE数据集上对比

表2 NH-HAZE和Dense-HAZE数据集结果对比

方法	NH-HAZE		Dense-Haze	
	PSNR	SSIM	PSNR	SSIM
(TPAMI'10) DCP[16]	10.57	0.5196	10.06	0.3856
(TIP'16) DehazeNet[17]	16.62	0.5238	13.84	0.4252
(ICCV'17) AOD-Net[49]	15.40	0.5693	13.14	0.4144
(ICCV'19) GridDehazeNet[50]	13.80	0.5370	13.31	0.3681
(AAAI'20) FFA-Net[51]	19.87	0.6915	14.39	0.4524
(CVPR'20) MSBDN[22]	19.23	0.7056	15.37	0.4858
(CVPR'20) KDDN[52]	17.39	0.5897	14.28	0.4074
(CVPR'21) AECR-Net[24]	19.88	0.7173	**15.80**	0.4660
Ours	**20.09**	**0.7845**	14.98	**0.5346**

- PSNR指标：是一种对图像**像素质量**衡量的客观评价指标；
- SSIM指标：从**亮度，对比度和结构相似度**三方面对两张图进行衡量

两者都是越大越好

NH-HAZE和Dense-HAZE数据集上对比

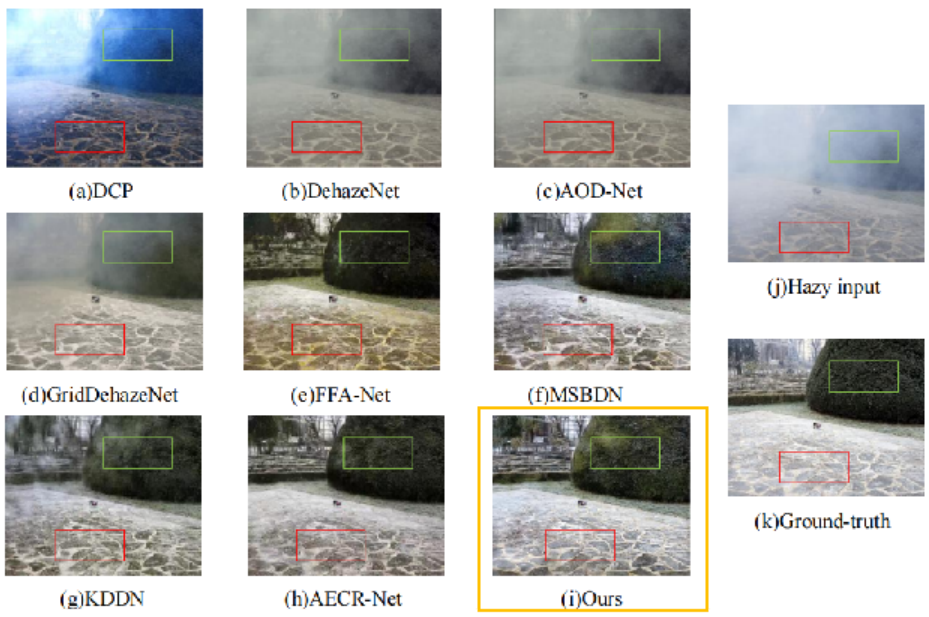


图10 NH-HAZE恢复图对比

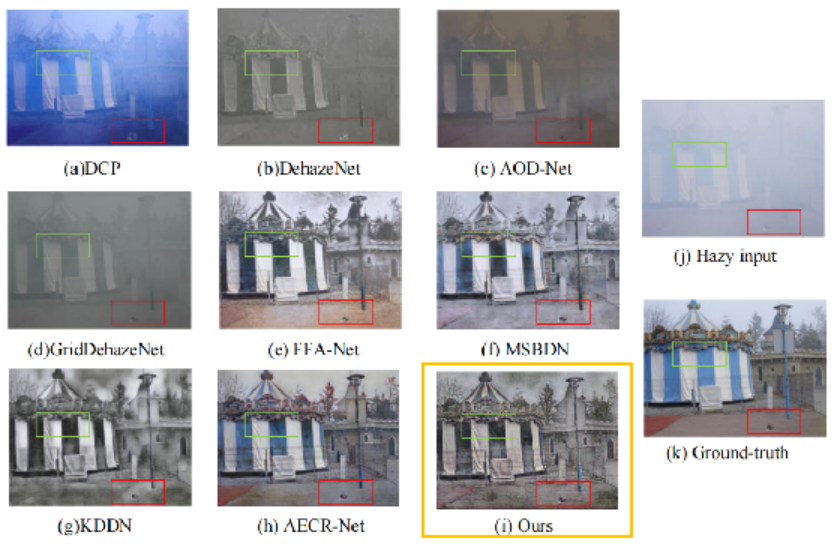


图11 Dense-HAZE恢复图对比

结果分析：

> **根据恢复图的结果，我们发现在部分图上的效果并不是特别优异**
>
> ***可以很好的反应Vision Transformer的劣势：*该架构虽然全局建模能力强，但局部建模能力没有CNN强，因此当输入某物体占大部分空间时，恢复结果容易受到其影响；因此可以在之后改进中使用CNN和Transformer组合模型，共同对全局和局部进行建模。**

4. 消融实验

NH-HAZE数据集上消融实验

将**对比正则化**引入后：PSNR 和 SSIM 指标都有提升

- 因此证明了让恢复图像的特征空间接近无雾图像的，远离带雾图像的对比正则化有益于图像的恢复，这与前人工作的结论一致

将**概率稀疏自注意力机制**引入后：虽然减少了自注意力机制的复杂度，但PSNR 指标略有下降

- 这可能是因为该机制为了极大减少时间和空间复杂度，而**选用少量的点积对来计算自注意力所造成的信息损失**，从而降低恢复图像的质量；但因为复杂度的大幅降低，这种损失是可以接受的。

因此，我们提出的各个改进部分都能够对模型性能进行了一定程度上的提升。

表4 NH-HAZE数据集上消融实验

模型	对比正则化	概率稀疏自注意力机制	自注意力所用总内存空间	PSNR	SSIM
Uformer	—	—	267.97MB	18.89	0.7507
	√	—	267.97MB	**20.20**	0.7781
	√	√	**263.00**MB	20.09	**0.7845**

5. 总结展望

总结展望

本文将基于 Vision Transformer 模型引入了图像去雾任务中，虽然针对一些问题进行改进，但仍存在一些缺点：

1. 在大量任务中，一般会将 Transformer 模型用**大量的数据进行预训练**，然后在下游任务中再进行微调，这种做法会对模型结果带来很大提升。而在我们的实验中使用了真实数据集，但这些数据集都是少量的，因此没有进行预训练。未来可以在大量的生成的数据集上先进行预训练，然后再在少量的真实数据集上进行微调。

2. **根据之前对数据集的分析**，且近期有相关的工作表明：以**多尺度表示不同大小物体的特征和区域是非常重要的**，因此可以用不同大小的卷积操作标记不同区域来学习到细粒度特征，再用 Vision Transformer 执行全局注意力学习粗粒度特征，最后融合细粒度和粗粒度特征。未来可以对模型架构进行改进，利用粗粒度和细粒度特征对带雾图像进行恢复，尝试减少恢复的无雾图中的伪影，提高恢复质量。

3. 虽然改进后的模型相较于原始的 Uformer 的参数量有所减少，但是相较于之前的去雾模型而言仍然较大，**因此需要对模型架构进一步优化以减少参数量**