



浙江工业大学

# 人工智能导论

## 大作业实验报告

传统图像处理方式的去雾、去雨算法与  
深度学习框架下的去雾、去雨算法的比较

姓 名 杨 骝

学 号 201906160125

专业班级 自动化 1902

指导教师 付 明 磊

学 院 信息工程学院

提交日期 2022 年 5 月 29 日

# 目 录

第 1 章	图像去雾算法.....	2
1.1	传统处理方式的去雾算法.....	2
1.1.1	有雾图片模型介绍.....	2
1.1.2	暗通道先验方法.....	2
1.2	深度学习方式的去雾算法.....	9
1.2.1	背景模型建立.....	9
1.2.2	DehazeNet 模型.....	10
1.2.3	实验和结果.....	14
第 2 章	图像去雨算法.....	18
2.1	传统处理方式的去雨算法.....	18
2.1.1	稀疏编码图像去雨方法介绍 <sup>[15]</sup> .....	18
2.1.2	实验与结果.....	21
2.2	深度学习方式的去雨算法.....	22
2.2.1	背景模型建立.....	22
2.2.2	JORDER 模型.....	22
2.2.3	实验和结果.....	26
2.2.4	深度学习算法与传统图像算法的比较.....	28
第 3 章	总 结.....	29
参 考 文 献	.....	32

## 第 1 章 图像去雾算法

### 1.1 传统处理方式的去雾算法

#### 1.1.1 有雾图片模型介绍

在计算机视觉中，下面这个模型被广泛地用于描述有雾的图像：

$$I(x) = J(x)t(x) + A(1 - t(x)) \quad (1-1)$$

其中 $I$ 表示了观测得到的图片亮度， $J$ 表示图片原本的亮度， $A$ 表示全球大气光成分， $t$ 表示从物体到相机的折射率。如图 1.1-1，有雾图像生成经过模型说明了拍摄物体的光经过了散射、折射和太阳光同时进入相机，形成了最终拍摄图像。由于 $I$ 是已知量，那么只需要去计算或者估计大气光成分和折射率就能够还原出图像原本的亮度，即完成了去雾操作。

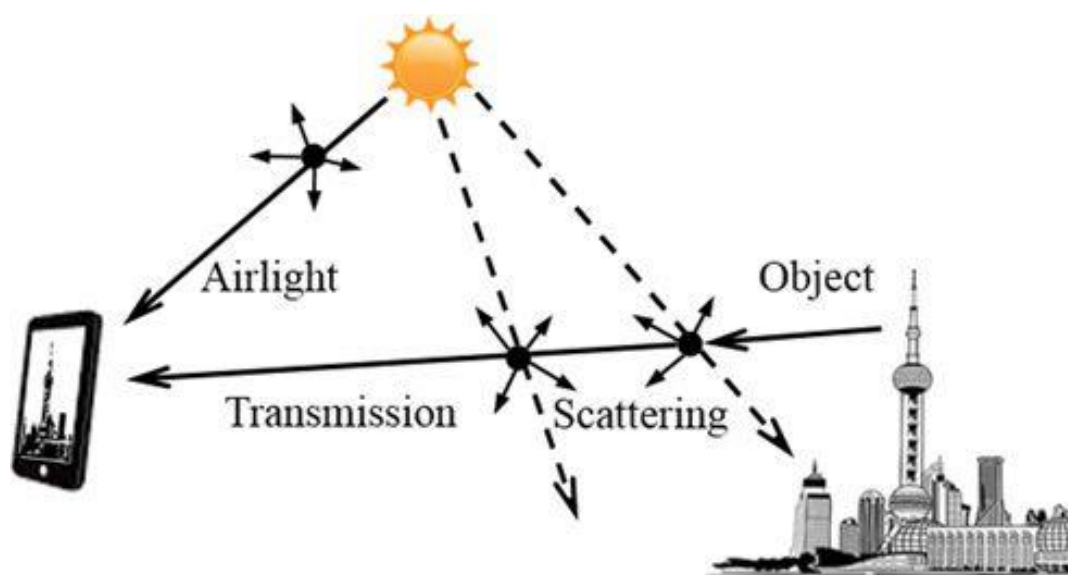


图 1.1-1 有雾图像生成经过

#### 1.1.2 暗通道先验方法

暗通道去雾<sup>[1]</sup>是由何恺明于 2009 年提出的一种图像去雾方式。基于大量的图片观测得到，在图像的无雾部分中至少有一个颜色通道内的像素点数值（亮度）非常低。因此对于一个图像 $J$ 而言,我们可以引入一个新的定义：

$$J^{dark}(x) = \min_{c \in \{r, g, b\}} \left( \min_{y \in \Omega(x)} (J^c(y)) \right) \quad (1-2)$$

其中 $J^c$ 是指图像 $J$ 的其中一个颜色通道。 $\Omega(x)$ 是指中心为 $x$ 的一个方框。经过对大量图片的观测， $J^{dark}$ 在除了天空区域之外，基本上的数值为0。将这样经过大量图片验证的信息称为暗通道先验。下面将通过一个实际的例子，来具体了解暗通道的计算方法。

14	14	18	24	30	32	29
19	18	16	31	50	50	50
22	19	21	31	48	65	70
23	24	25	38	51	62	71
35	32	33	42	53	57	63
58	54	51	53	56	59	49
62	63	69	66	61	66	46

图 1.1-2 (a) 三通道像素值(红色通道)

29	26	29	34	39	43	45
34	30	31	41	56	58	58
39	37	35	42	55	71	80
40	40	40	49	56	67	87
49	45	43	53	59	62	75
72	67	63	63	65	59	54
73	77	77	75	67	64	47

图 1.1-2 (b) 三通道像素值(绿色通道)

7	7	8	7	9	13	10
11	10	10	14	24	29	24
15	15	14	18	29	40	43
14	16	16	25	32	40	49
22	22	23	31	38	35	41
39	37	34	39	42	39	27
40	42	46	48	42	41	23

图 1.1-2 (c) 三通道像素值(蓝色通道)

7	7	8	7	9	13	10
11	10	10	14	24	29	24
15	15	14	18	29	40	43
14	16	16	25	32	40	49
22	22	23	31	38	35	41
39	37	34	39	42	39	27
40	42	46	48	42	41	23

图 1.1-2 (d) 三通道像素最小值

如图 1.1-2 (a) ~图 1.1-2 (c) 所示为彩色图像的三通道像素值，需要计算每个像素值的 2 邻域暗通道。首先计算出每个通道中像素最小值，如图 1.1-2 (d) 所示。然后先对图像进行扩充，随后选择合适的方框大小，如图 1.1-3 所示。选择方框内像素值最小的数值作为该像素点暗通道的数值，如图 1.1-4 所示。

7	7	7	7	8	7	9	13	10	13	10
11	10	11	10	10	14	24	29	24	29	24
7	7	7	7	8	7	9	13	10	13	10
11	10	11	10	10	14	24	29	24	29	24
15	15	15	15	14	18	29	40	43	40	43
14	16	14	16	16	25	32	40	49	40	49
22	22	22	22	23	31	38	35	41	35	41
39	37	39	37	34	39	42	39	27	39	27
40	42	40	42	46	48	42	41	23	41	23
39	37	39	37	34	39	42	39	27	39	27
40	42	40	42	46	48	42	41	23	41	23

图 1.1-3 像素扩充图

7	7	7	7	7	7	7
7	7	7	7	7	7	7
7	7	7	7	7	7	7
7	7	7	7	7	7	7
7	7	7	7	7	7	7
23	23	23	23	23	8	8
23	23	23	23	23	8	8

图 1.1-4 暗通道数值图

暗通道中的像素值非常低主要是由于三个因素:阴影。例如,城市景观图像中的汽车、建筑和窗户内部的影子,或者风景图像中的树叶、树木和岩石的阴影;其次,彩色物体或表面。任何物体在任何颜色通道中缺乏颜色都会导致暗通道值偏低;最后,黑暗物体表面。例如,黑色的树干和石头。由于自然的户外图像通常是充满阴影和色彩丰富的。而这些物体确实非常暗才导致暗通道中的像素值非常低。



图 1.1-5 原始图像图



图 1.1-6 暗通道图(窗口大小为 15)

对于一个实际图像如图 1.1-5 所示,其对应的暗通道处理效果如图 1.1-6 所示。经过作者对将近 5000 张图片的暗通道进行观测得到,暗通道内大约 75%的像素点的亮度为 0, 90%的像素点的亮度小于 25, 如图 1.1-7 所示为 5000 张图片的平均灰度直方图。

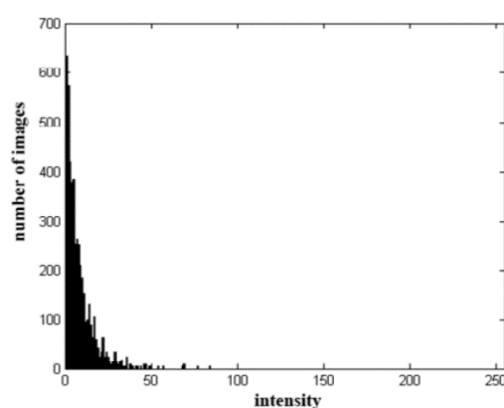


图 1.1-7 图像平均灰度值直方图

由于大气太阳光的影响，有雾图像比对应的无雾图像的折射率要低，所以有雾图像的亮度要比对应的无雾图像的亮度要高。在下一节，将给出折射率的估计方式。

## ● 折射率估计

我们将中的有雾图片模型进行最小化操作，可以得到如下的公式：

$$\min_c \left( \min_{y \in \Omega(x)} \left( \frac{I^c(y)}{A^c} \right) \right) = \tilde{t}(x) \min_c \left( \min_{y \in \Omega(x)} \left( \frac{J^c(y)}{A^c} \right) \right) + (1 - \tilde{t}(x)) \quad (1-3)$$

其中 $\tilde{t}(x)$ 是某像素的折射率。而根据暗通道先验知识，暗通道 $J^{dark}(x)$ 在无雾图像中的亮度为 0，可以得到如下公式：

$$J^{dark}(x) = \min_{c \in \{r, g, b\}} \left( \min_{y \in \Omega(x)} (J^c(y)) \right) = 0 \quad (1-4)$$

由于全球大气光成分 $A^c$ 一直为正数，所以可以推导出：

$$\min_c \left( \min_{y \in \Omega(x)} \left( \frac{J^c(y)}{A^c} \right) \right) = 0 \quad (1-5)$$

因此最终的折射率估计公式如下：

$$\tilde{t}(x) = 1 - \min_c \left( \min_{y \in \Omega(x)} \left( \frac{I^c(y)}{A^c} \right) \right) \quad (1-6)$$

而实际上，公式中 $\min_c \left( \min_{y \in \Omega(x)} \left( \frac{I^c(y)}{A^c} \right) \right)$ 就是 $\frac{I^c(y)}{A^c}$ 图像的暗通道，因此可以通过图像 $\frac{I^c(y)}{A^c}$ 计算暗通道得到。

上述的暗通道先验知识是排除了天空区域，但是一张图像往往具有天空部分，不能够排除。幸运的是，天空的颜色往往与大气光成分 $A^c$ 相近，因此我们可以得到：

$$\min_c \left( \min_{y \in \Omega(x)} \left( \frac{I^c(y)}{A^c} \right) \right) \rightarrow 1, \tilde{t}(x) \rightarrow 0 \quad (1-7)$$

即经过去雾处理后的原本的天空部分会接近于大气成分光。所以折射率的估计公式对于天空区域和非天空区域都适用，无需对天空区域做额外的处理。

值得注意的是，即使对于普通的图片在晴天下，也不是完全无雾的。在我们看向远处的物体时也存在一定的雾，因此我们需要保留一定的雾通过引入一个常数：

$$\tilde{t}(x) = 1 - \omega \min_c \left( \min_{y \in \Omega(x)} \left( \frac{I^c(y)}{A^c} \right) \right) \quad (1-8)$$

$\omega$ 通常设置为 0.95。 $\omega$ 也可以根据实际图片的效果进行改变，如果 $\omega = 0$ ，根据式 1-8，所得图片与原始图片一致。

通过直接估计得到的折射率还比较粗糙，这里还可以通过原片的灰度图进行导向滤波<sup>[2]</sup>对折射率图片进一步地精细化。导向滤波具有速度快，能够保留梯度信息等优势，导向滤波将在最后一节进行讲解。这里先采用原本的折射率进行图像还原，折射率估计图如图 1.1-8 所示：



图 1.1-8 折射率估计图

## ● 图像还原

在折射率估计部分，假设大气光成分 $A^c$ 已知，在这一部分，我们给出一种简单方法用于估计大气光。如图 1.1-5 和图 1.1-6 所示，图片暗通道中的有雾区域与实际图像的有雾区域非常近似，因此我们可以将暗通道中最亮的 0.1% 像素的平均值当作大气光成分 $A^c$ 。

估计出大气光成分 $A^c$ 和折射率 $t(x)$ 之后，可以利用有雾图片的模型进行图像的还原。但是当折射率 $t(x)$ 接近于 0 的时候。公式中的 $J(x)t(x)$ 项也会接近于零。比较容易产生噪声，因此为折射率 $t(x)$ 设置一下下界 $t_0$ ，意味着在雾非常密集的区域保留了少量的雾。最后还原公式如下：

$$J(x) = \frac{I(x) - A}{\max(t(x), t_0)} + A \quad (1-9)$$

最后还原效果如下图 1.1-9 所示：



图 1.1-9 图像去雾效果图

### ● 折射率导向滤波

在折射率估计部分中，通过暗通道先验得到了较为粗糙的折射率估计，实际上可以采用导向滤波对折射率做进一步的精细化操作。下面将介绍导向滤波，以及使用经过导向滤波后的折射率图进行图像还原的原本图像。

导向滤波可以像流行的双边滤波器一样作为边缘保持平滑算子，但是在边缘附近具有更好的行为。引导滤波器还具有快速的非近似线性时间算法，其计算复杂度与滤波核大小无关。经证明导向滤波器在各种计算机视觉和计算机图形学应用中既有效又高效。

导向滤波假设是引导图 $I$ 和滤波器输出 $q$ 之间是一个局部线性模型。因此我们可以假设其中的线性变换的基本公式如下：

$$q_i = a_k I_i + b_k, \forall i \in \omega_k \quad (1-10)$$

其中 $a_k$ ， $b_k$ 是固定窗口下的线性变换参数，使用的窗口半径为 $r$ 。并且由于 $\nabla q = a \nabla I$ ，所以导向滤波可以在边缘处有较好的行为。我们认为输入图像 $p_i$ 是由原始图像 $q_i$ 和噪声 $n_i$ 组合得到的。可以得到下述公式：

$$q_i = p_i - n_i \quad (1-11)$$

为了确认线性变换的参数，使 $q$ 和滤波器输入 $p$ 之间的差异（噪声 $n_i$ ）最小化。具体而言，我们使窗口中的以下成本函数最小化：

$$E(a_k, b_k) = \sum_{i \in \omega_k} \left( (a_k I_i + b_k - p_i)^2 + \epsilon a_k^2 \right) \quad (1-12)$$

其中 $\epsilon$ 是正则项参数，避免 $a_k$ 参数过大。最后得到的求解结果如下：



$$a_k = \frac{\frac{1}{|\omega|} \sum_{i \in \omega_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \epsilon} \quad (1-13)$$

$$b_k = \bar{p}_k - a_k \mu_k$$

其中 $\mu_k$ 和 $\sigma_k^2$ 是窗口 $\omega_k$ 内导向图 $I$ 的均值和方差， $|\omega|$ 是窗口 $\omega_k$ 内的像素个数， $\bar{p}_k$ 窗口 $\omega_k$ 内输出图像的像素均值。

至此，我们可以给出导向滤波的算法步骤：

$$\begin{aligned} \text{mean}_I &= f_{\text{mean}}(I) \\ \text{mean}_p &= f_{\text{mean}}(p) \\ \text{corr}_I &= f_{\text{mean}}(I * I) \\ \text{corr}_{Ip} &= f_{\text{mean}}(I * p) \\ \text{var}_I &= \text{corr}_I - \text{mean}_I * \text{mean}_I \\ \text{cov}_{Ip} &= \text{corr}_{Ip} - \text{mean}_I * \text{mean}_p \\ a &= \frac{\text{cov}_{Ip}}{\text{var}_I + \epsilon} \\ b &= \text{mean}_p - a * \text{mean}_I \\ \text{mean}_a &= f_{\text{mean}}(a) \\ \text{mean}_b &= f_{\text{mean}}(b) \\ q &= \text{mean}_a * I + \text{mean}_b \end{aligned} \quad (1-14)$$

其中 $p$ 表示， $f_{\text{mean}}$ 表示窗口半径为 $r$ 的均值滤波。

最后给出经过导向滤波之后的折射率估计图片与原本粗糙的折射率估计图片，以及对于的还原图像 1.1-10 和 1.1-11：

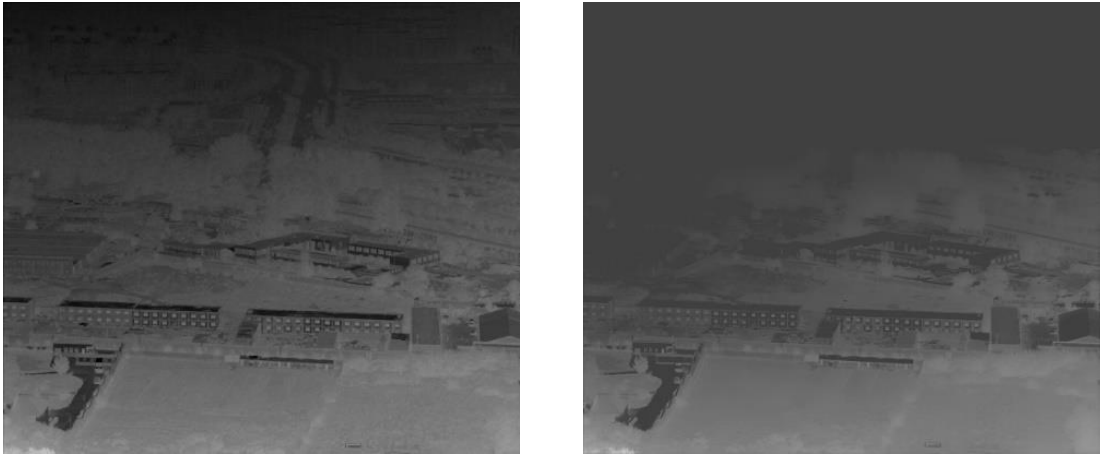


图 1.1-10 折射率估计图对比（左侧为粗糙估计，右侧经过导向滤波）



图 1.1-11 还原图像对比（左侧为粗糙估计折射率，右侧经过导向滤波的折射率）

可以看到使用经过导向滤波之后的折射率图片的去雾图片更加接近真实的图片，而使用仅通过暗通道先验的还原图像，颜色偏深，远处的图片没有得到很好的还原。

## 1.2 深度学习方式的去雾算法

### 1.2.1 背景模型建立

为了处理通过数字图像处理技术去除有雾图片中的雾，首先需要对有雾图片进行数学建模。McCartney et al.<sup>[3]</sup>提出了 atmospheric scattering model 来对其进行描述，Narasimhan et al.<sup>[4]</sup>和 Nayar et al.<sup>[5]</sup>对其进行优化，最终有雾图片的数学模型可以写作：

$$I(x) = J(x)t(x) + \alpha(1 - t(x)) \quad (1 - 15)$$

其中， $I(x)$ 是被观测到的有雾图片， $J(x)$ 是我们希望得到的真实图片，即去雾的图片。 $t(x)$ 是一个媒介传递函数， $\alpha$ 是一个全局的环境光照参数。式子中的 $x$ 是图片中像素的索引。去雾任务的目标是得到真实场景的图片，即式中的 $J(x)$ ，因此我们还需要知道媒介传递函数 $t(x)$ 以及 $\alpha$ 。

媒介传递函数 $t(x)$ 描述了镜头离物体的远近对进光量的影响，它被写作：

$$t(x) = e^{-\beta d(x)} \quad (1 - 16)$$

其中 $d(x)$ 表示场景中一点到镜头的距离， $\beta$ 用来表示大气对光线削弱程度。如果距离 $d(x)$ 趋于无穷远，那么根据式 1-16 可以得到 $\alpha = I(x)$ 。但显然距离不会趋向于零，但是如果距离很远，就会导致一个非常小的转换变量 $t_0$ 。为了估计出 $\alpha$ ，此文提出了一种更加稳定的估计方法：

$$\alpha = \max_{y \in \{x | t(x) \leq t_0\}} I(y) \quad (1-17)$$

由此可以得到：如果要从一张有雾图像中复原出清晰的场景，最关键的一点便是准确估计媒介传递映射，即  $t(x)$ 。

### 1.2.2 DehazeNet 模型

为了准确的估计出媒介传递映射来复原一张有雾图像，此文提出了一种端到端的深度学习框架，即 DehazeNet。下面介绍 DehazeNet 的结构框架。下图展示了 DehazeNet 框架的示意图。

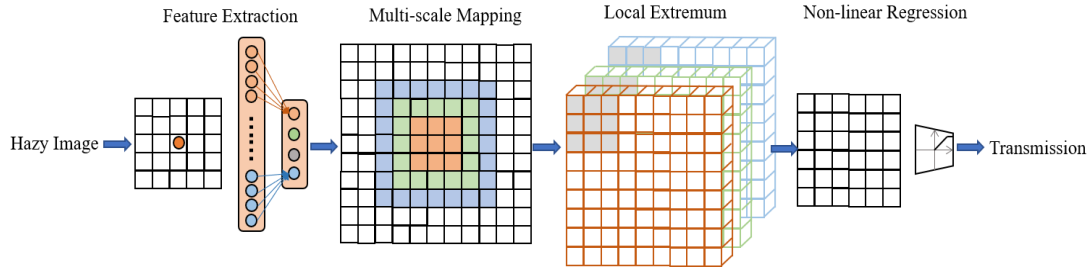


图 1.2-1 DehazeNet 深度去雾模型的框架总览

#### ● Feature Extraction

为了解决去雾任务中的图像的 ill-posed 问题，传统的图像处理工作提出了各种假设并基于这些假设提取出关于雾的特征，特征的提取在整张图像上完成，这样的操作其实等价于用一个合适大小的卷积核对输入的图像进行卷积操作，然后对卷积后的输出做一个非线性映射。这一层中，选用一个 Maxout 输出单元作为这个线性映射。因此，DehazeNet 的第一层可以表示为

$$\begin{aligned} F_1^i(x) &= \max_{j \in [1, k]} f_1^{i, j}(x) \\ f_1^{i, j} &= W_1^{i, j} * I + B_1^{i, j} \end{aligned} \quad (1-18)$$

其中， $W_1 = \{\omega_1^{i, j}\}_{(i, j)=(1, 1)}^{(n_1, k)}$ ， $B_1 = \{b_1^{i, j}\}_{(i, j)=(1, 1)}^{n_1, k}$  分别表示卷积核中的权重以及偏差。式子中的 \* 表示卷积操作。Maxout 映射单元将多个高维的矩阵映射为只有一个维度的向量，并由此方式自动学习、提取图像中的有关于雾的特征。

## ● Multi-scale Mapping

多尺度特征是从图像的多个空间尺度来计算图像的特征，多尺度特征已经被证明能够有效地帮助我们去雾<sup>[6]</sup>。并且多尺度特征的提取也保证了从图像中提取的特征的尺度不变性质。多尺度特征提取在广泛应用于深度学习框架中，例如 GoogLeNet<sup>[7]</sup>、ILSVRC14<sup>[8]</sup>等。受这些框架的启发，DehazeNet 使用平行卷积操作作为 DehazeNet 的第二层。此处的卷积核的大小可以从  $3 \times 3$ 、 $5 \times 5$  以及  $7 \times 7$  中选择。那么 DehazeNet 的第二层输出可以写作：

$$F_2^i = W_2^{\lceil \frac{i}{3} \rceil (i \setminus 3)} * F_1 + B_2^{\lceil \frac{i}{3} \rceil (i \setminus 3)} \quad (1-19)$$

其中， $W_2 = \{\omega_2^{p,q}\}_{(p,q)=(1,1)}^{(3, \frac{n_2}{3})}$ ， $B_2 = \{b_2^{p,q}\}_{(p,q)=(1,1)}^{(3, \frac{n_2}{3})}$ ， $n_2$  对参数被分割为 3 组并且  $n_2$  表示 DehazeNet 第二层的输出维度。式 5 中的  $\lceil * \rceil$  和  $\setminus$  分别表示向上取整函数以及取余函数。

## ● Local Extremum

为了实现空间不变形，采用一个线性特征整合来处理第二层输出的结果。在经典的 CNN 模型中，邻域最大池用来克服像素层面上的位置灵敏性。除此之外，local extremum 也符合媒介传递映射是在邻域上不变的性质，同时他也能抑制对估计传递映射时产生的误差带来的噪声。因此，DehazeNet 的第三层为一个 local extremum 操作，即

$$F_3^i(x) = \max_{y \in \Omega(x)} F_2^i(y) \quad (1-20)$$

其中， $\Omega(x)$  是一个  $f_3 \times f_3$  的邻域，第三层输出的维度  $n_3 = n_2$ 。不同于 CNN 中的池化层，local extremum 操作应用于像素层面的特征映射，从而能够保留分辨率用于图像复原。

## ● Non-linear Regression

在深度学习网络中，常常用来作为激活函数的有 Sigmoid 和 Rectified Linear Unit (ReLU) 函数。前者非常容易产生梯度消失现象，这会使网络在训练时收敛过慢或者到达一个局部极小值；后者虽然能够解决前者的问题，但后者常常用于分类任务，并不适合于回归任务，例如图像复原任务。ReLU 只在值小于零时抑制值。这可能会导致响应溢出，特别是在最后一层，因为在图像恢复时，最后一层的输出值应该在一个小范围内同时处于上下边界。为了解决这一问题，本文提出了 Bilateral Rectified Linear

Unit (BReLU) 激活函数。受到 Sigmoid 和 ReLU 激活函数的启发, BReLU 作为一种新颖的线性单元保持了双边约束和局部约束线性。在此基础上, 将第四层特征映射定义为

$$F_4 = \min(t_{\max}, \max(t_{\min}, W_4 * F_3 + B_4)) \quad (1-20)$$

这里  $W_4 = \{\omega_4\}$  包含了一个大小为  $n_3 \times f_4 \times f_4$  的卷积核,  $B_4$  则是一个偏差。  $t_{\min}$  和  $t_{\max}$  分别为 BReLU 的上下界, 这此文中  $t_{\min} = 0, t_{\max} = 1$ 。这一激活函数的梯度可以被写成:

$$\frac{\partial F_4(x)}{\partial F_3} = \begin{cases} \frac{\partial F_4(x)}{\partial F_3}, & t_{\min} \leq F_4(x) < t_{\max} \\ 0, & otherwise \end{cases} \quad (1-21)$$

以上四层被级联在一起形成一个基于 CNN 的端到端可训练系统, 其中与卷积层相关的滤波器和偏差是需要学习的网络参数。DehazeNet 具体的模型框架以及模型中每一层结构的具体信息汇总在表 1。我们注意到, 这些层的设计可以与现有图像去雾方法的专业知识相联系。

表 1-1 DehazeNet 模型的结构框架

Formulation	Type	Input Size	Num $n$	Filter $f \times f$	Pad
Feature Extraction	Conv	$3 \times 16 \times 16$	16	$5 \times 5$	0
	Maxout	$16 \times 12 \times 12$	4	-	0
Multi-scale Mapping	Conv	$4 \times 12 \times 12$	16	$3 \times 3$	1
			16	$5 \times 5$	2
			16	$7 \times 7$	3
Local Extremum	Maxpool	$48 \times 12 \times 12$	-	$7 \times 7$	0
Non-linear Regression	Conv	$48 \times 6 \times 6$	1	$6 \times 6$	0
	BReLU	$1 \times 1$	1	-	0

## ● Training of DehazeNet

**Training Data:** 训练深度神经网络, 需要较多的带雾图像以及他们原始无雾图像。但实际上带雾的图像很难找到对应的无雾的图像, 为了解决这一问题, 可以使用自己合成的带雾的图像来对 DehazeNet 进行训练。

考虑到一张有雾图像可以被建模为如式 1-15 的模型, 因此我们可以合成一对有雾

图像和无雾图像通过无雾图像上增加媒介传播函数。这样的做法基于两个假设：1) 图像内容独立于媒介传输函数；2) 媒介传播函数具有领域不变的性质，即图像中两个相邻像素点离镜头的距离相同，并且穿过的媒介相同。基于这样的假设，我们可以为一张无雾图像生成对应的带雾的图像：考虑一张无雾图像  $J^P(x)$ ，大气自然光参数为  $\alpha$ ，以及一个随机生成的传播函数  $t \in (0, 1)$ ，相对应的带雾图像  $I^P(x)$  可以由下面的式子来合成：

$$I^P(x) = J^P(x)t + \alpha(1 - t) \quad (1 - 22)$$

为了减少训练时式 1-22 中参数对训练得到地模型的影响，我们将大气自然光参数  $\alpha$  设置为 1 不变。训练集中的无雾图像从网络上获得，并按照式 1-22 来生成对应的带雾图像，这些图像中包含的内容各不相同，包括人们的日常生活、自然风光、城市风光等。

### Training Method

DehazeNet 是一个监督训练模型，他的目标是最小化 DehazeNet 输出的媒介传播与真实的媒介传播之间的区别。考虑一个  $\mathcal{F}(I^P; \Theta)$  表示 DehazeNet 网络输出的媒介传播值，它和网络参数  $\Theta = \{W_1, W_2, W_4, B_1, B_2, B_4\}$  有关。这里我们设计此深度学习框架的损失函数  $\mathcal{L}(\Theta)$  网络输出媒介传播值  $\mathcal{F}(I^P; \Theta)$  和真实的媒介传播值  $t$  之间的均方误差，即

$$\mathcal{L}(\Theta) = \frac{1}{N} \sum_{i=1}^N \|\mathcal{F}(I_i^P; \Theta) - t_i\|^2 \quad (1 - 23)$$

为了使这个损失函数最小，我们使用 Stochastic gradient descent (SGD) 来训练 DehazeNet。

### Changing Parameters（调参过程）

DehazeNet 模型有很多参数，其参数数量可达 8240 个，包括暗通道窗口尺寸、暗通道亮度、去雾程度、最小透射率、大气光值等参数，但其中可调，并且能够明显影响模型效果的参数并不多，本文通过实验最终选取模型学习率、训练次数、每次训练的样本量、去雾程度和最小透射率这 5 个参数进行调整。关于模型的学习率参数、训练次数以及每次训练的样本量这三个参数，是在之前的人工智能算法训练中非常熟悉的参数。这三个参数都能够较大程度地影响模型的精度。

通过之前那么多的实验也明白了学习率参数不宜过小，但是也不宜过大。因为当学习率过大，每次梯度下降的步数过大，导致常常无法找到全局最优解；但当学习率参数过小时，会使得模型陷入局部最优解，无法实现全局最优解的效果。另外关于模型训练轮数

也是相同的道理，当训练轮数过大，不仅会使得模型的训练时间显著增加，可能还会导致模型过拟合，但是当模型的训练轮数过小，往往会导致模型训练的精度很低，欠拟合。还有模型每次训练的样本数据量，因为模型的训练是一个迭代的过程，所以每次模型训练数据集的数量也会对模型的精度和准确率产生影响。上述这三个参数在之前的实验中已经很熟悉了，因此在这里不对其做过多介绍。通过多次的调试，本文最终使用 GPU 训练模型，选择模型的学习率为 0.0006，训练次数为 50000 次，每次迭代的数据样本为 128 个。另外 DehazeNet 模型还有扫描半径参数  $\text{eps}$  和初始半径参数  $r$ ，这里本文也实现调试后，确定最终模型的扫描半径  $\text{eps}$  为 0.0001，初始半径  $r$  为 60。

而另外两个参数去雾程度  $w$  和最小透射率  $t$  是 DehazeNet 模型特有的参数信息。去雾程度  $w$  参数在模型中用于估算透射率图，去雾程度  $w$  的参数是一个归一化参数，其值分布在 0 至 1 之间，该值越小，模型最终的去雾效果越不明显；最小透射率值  $t$  在模型中用于图像颜色的整体调整，通常来说，模型最小透射率值  $t$  越小，显示的图像会整体向白场过度，显示越白。关于 DehazeNet 模型的去雾程度参数通常在 0.9 左右，最小透射率值通常在 0.2 左右。下图 1.2-2 将展示了以上面确定的最终学习率、轮数和迭代样本量为不变量参数的模型效果图

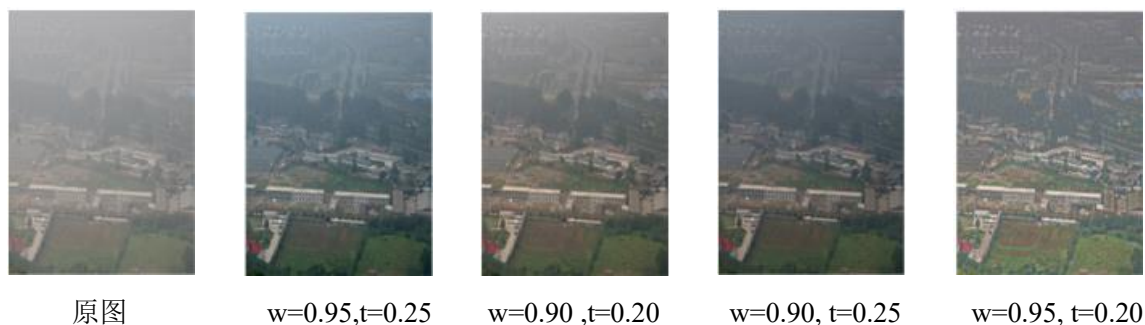


图 1.2-2 不同去雾模型参数在相同真实图下的去雾效果比较

最终通过数次调参确定了模型最终的去雾程度参数为 0.95，最小透射率值为 0.25。

### 1.2.3 实验和结果

#### 1. 去雾算法对比

本文将 DehazeNet 生成的结果与 6 种去雾方法进行对比，这 6 种方法分别为 ATM [9]、BCCR [10]、FVR [11]、DCP [12]、CAP [13]、RF [14]。我们首先直观地观察这 7 种方法复原得到的去雾图像之间的差异来定性分析。





图 1.2-3 不同去雾方法在相同真实图像 Street 下的去雾效果比较

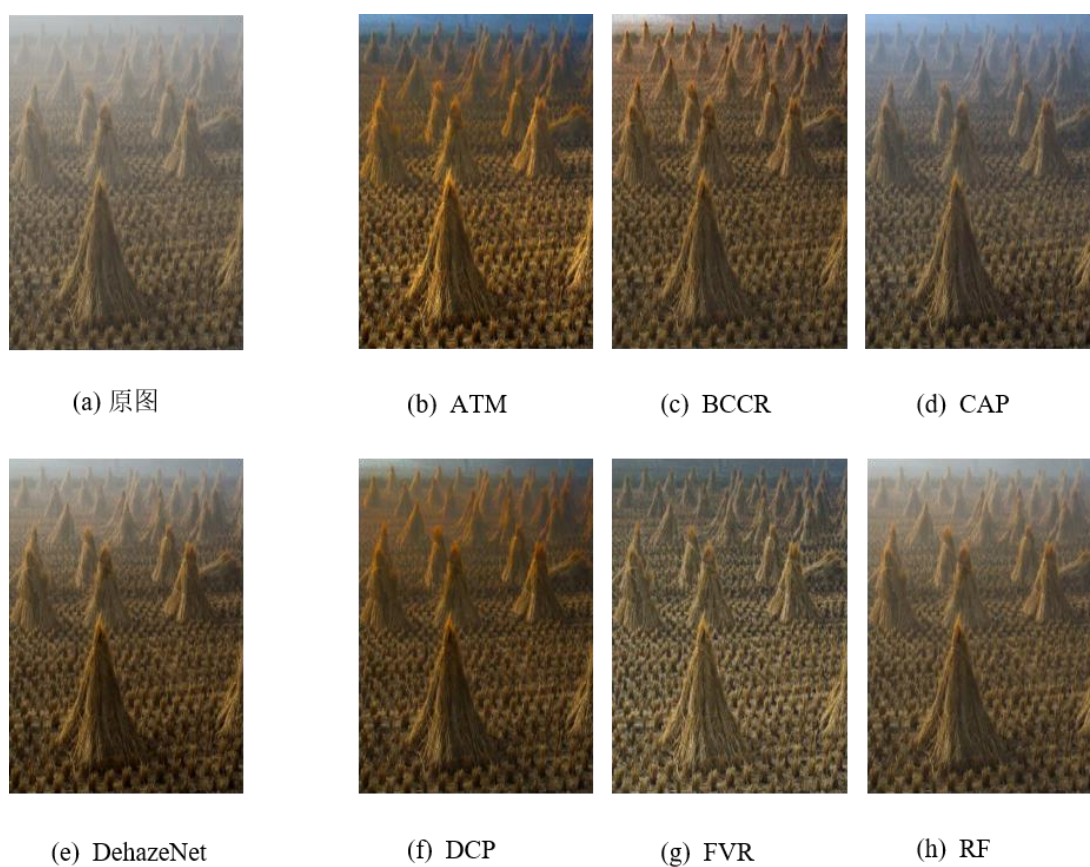


图 1.2-4 不同去雾方法在相同真实图像 Cones 下的去雾效果比较



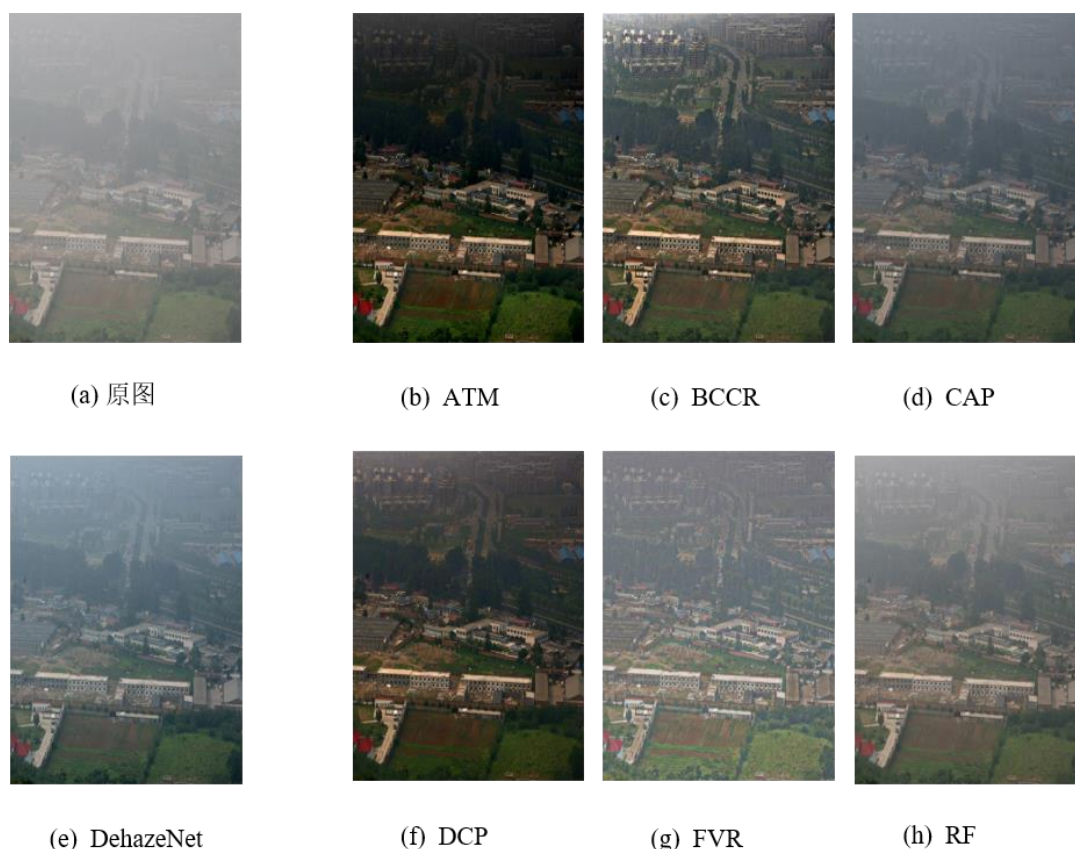


图 1.2-5 不同去雾方法在相同真实图像 Canon 下的去雾效果比较

几乎所有的去雾方法在户外的图像上面都取得了令人信服的、足够良好的结果，我们很难用肉眼比较他们的结果并对这些方法进行排序。为了比较这 7 种方法，我们更加关注于 3 张非常有挑战性的带雾的图片。这些图片上存在大量的白色或者灰色的区域，这些区域对去雾任务来说是很难去处理的，因为大多数的去雾算法都对白色或者灰色非常敏感。图 1.2-3、图 1.2-4 和图 1.2-5 定性地对比了 7 种不同的效果最好的去雾算法在真实世界图像中的去雾效果。两幅图的图(a)表示带雾的原图，图(e)是采用 DehazeNet 复原的图像，图(b)~图(h)（除了图 e）分别展示了 ATM、BCCR、CAP、DCP、FVR 以及 RF 的图像复原结果。

有雾图像中的天空区域是去雾任务中的难点，因为云和雾存在相似的大气散射模型（atmospheric scattering model）。如图 1.2-3、图 1.2-4 和图 1.2-5 所示，绝大多数去雾算法都很好地实现了去雾的任务，场景和物体的细节都被很好地还原，然而这些算法的结果都存在不同程度上的过度增强的现象，特别是天空区域的复原。这些算法得到的复原之后的天空区域都比它应有的黑暗，或者出现过饱和或者颜色改变的现象。雾气通常来说只存在于大气表层，因此天空区域几乎不用处理。基于学习框架的两种

方法 CAP 和 RF 虽然在一定程度上避免了天空区域颜色的改变，但画面中非天空区域的颜色却被糟糕地增强了，这可能是因为他们使用的是非基于内容的回归模型。相对而言，DehazeNet 能够找到天空区域，并使他们保留原来的色彩，同时确保其他区域能够获得有效的去雾效果。

## 2. 深度学习算法与传统图像算法的对比

Dark Channel Prior (DCP) 所使用的基于先验的传输估计是一种统计的方法，对某些特定情况下的图像可能并不起作用，无法达到预期的去雾效果。其论文也指出，当场景物体与大气光具有内在相似性时，基于暗通道先验的传输估计并不可靠。DCP 方法本身就存在固有的过高估计传输的问题。下面将举出具体的图片例子来说明 DCP 这一问题，并使用 DehazeNet 去雾算法对这一图片进行去雾，比较两种方法对这一图像去雾后的效果来说明 DehazeNet 相对于传统图像处理方法的优势。

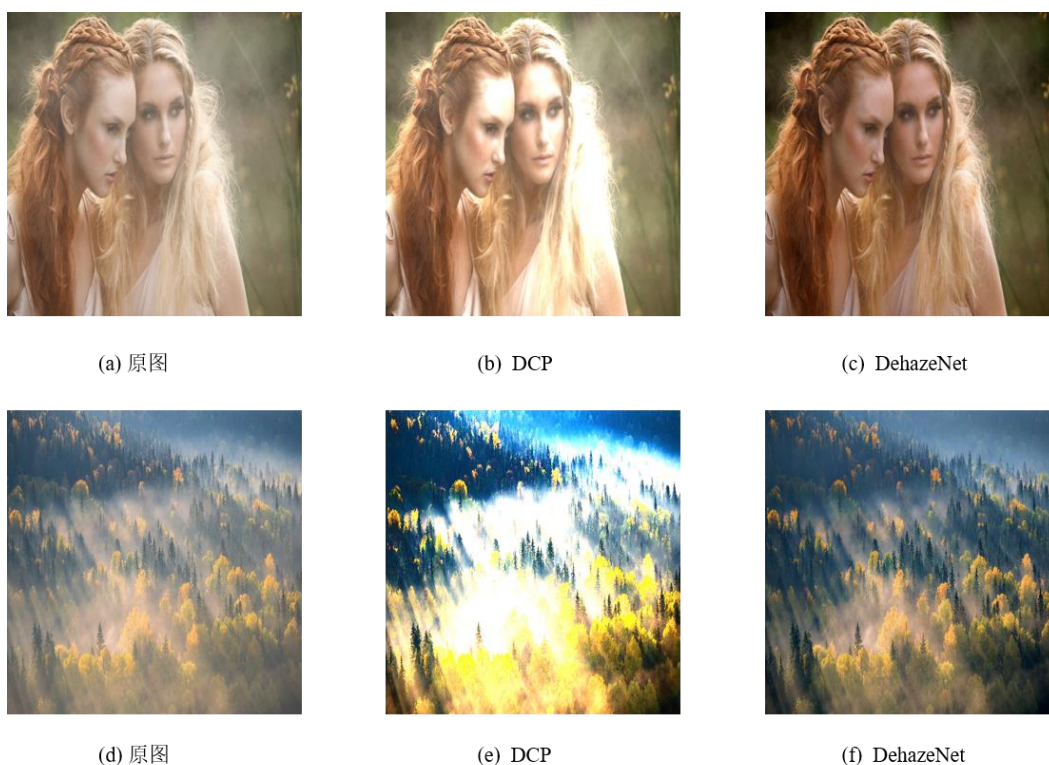


图 1.2-6 DehazeNet 与暗通道先验方法对比

图 1.2-6 中 (a) 和 (d) 是需要去雾的输入图片，他们都有一个共同的特点：场景物体与大气光具有内在相似性。例如图 (a) 中女孩白皙的皮肤和图 (b) 中透过树林的光线。图 (b) 和图 (e) 为 Dark Channel Prior 去雾方法获得的复原图像。从图中可

以很明显地看到此方法得到的图像出现了过度曝光的现象（图（b）女孩的头发处以及图（e）中颜色偏黄的树木处），这显然地反映了 DCP 算法过高估计传输的问题。图（c）和图（f）是采用 DehazeNet 方法得到的复原图像，与图（b）和图（e）相比较可以很明显地看出 DehazeNet 避免了图像的过饱和，同时也有效地去除了图像中的雾。通过与传统图像处理方法的比较，我们可以很明显地得到基于深度学习框架的去雾算法的优势所在。

## 第 2 章 图像去雨算法

### 2.1 传统处理方式的去雨算法

#### 2.1.1 稀疏编码图像去雨方法介绍<sup>[15]</sup>

现在已经有非常多的含雨的模型，最常见的是线性加性复合模型，一种流行的加法合成模型如下所示：

$$J = I + R \quad (2-1)$$

其中  $J$  表示雨图像， $I$  表示去雨图像层， $R$  表示雨层。

然而，通过合成的雨图像缺少在真实雨图像中经常出现的一些特征，例如内部反射的效果。这促使我们需要采用非线性复合模型。在 Photoshop 中会使用屏幕混合模型来渲染更真实的雨图像。它通过反转、相乘和再次反转两层来复合两层：

$$J = 1 - (1 - I) * (1 - R) = I + R - I * R \quad (2-2)$$

由屏幕混合模型渲染的图像通常看起来比由简单的加法合成模型渲染的图像更真实。例如，对于亮度接近 0 的暗背景，雨层将主导雨图像的外观。相比之下，当背景相当亮，亮度接近 1 时，雨图像以图像层为主，几乎看不到雨层。对于均匀的灰色区域，使用屏幕混合模型相当于在正常模式下使用该灰度值作为混合白色区域的不透明度，类似于雨滴的透明效果。

基于稀疏性的先验被广泛应用于图像恢复任务中，具有令人印象深刻的性能。因此在该方法中，也考虑了学习字典下图像层和雨层的稀疏先验。更具体地说，定义  $Y_1$  向量形式的图像层的斑块集合和  $Y_2$  雨层的斑块集合，而在字典下可以进行近似即：

$$Y_1 := PI \approx DC_1 \quad (2-3)$$

$$Y_2 := \mathcal{P}R \approx DC_2 \quad (2-4)$$

其中,  $\mathcal{P}$ 表示将图层映射到数组的线性运算符,  $D$ 为非监督设置中的学习字典,  $C_1$ 和 $C_2$ 分别表示了稀疏编码。

为了进一步加强区分性, 我们假设可以根据两个图层的区域部分, 从稀疏码中得到的标签将其分成两个不同的类别。下面可以提出降噪模型。

$$\min_{I, R, D, C_I, C_R} \|PI - DC_I\|_F^2 + \|PR - DC_R\|_F^2 \quad (2-5)$$

需要满足

$$\begin{cases} J = I + R - I * R; \\ 0 \leq I \leq 1, 0 \leq R \leq 1; \\ \|C_I[:, j]\|_0 \leq T_1, \|C_R[:, j]\|_0 \leq T_2, \text{对于所有的 } j; \\ \mathcal{B}(C_I)^T \mathcal{B}(C_R) \leq \varepsilon \end{cases}$$

其中 $T_1$ 和 $T_2$ 分别是稀疏码 $C_I$ 和 $C_R$ 的每一列的稀疏约束。

最小化问题是一个非常具有挑战性的非凸优化问题。解决这类问题的一种广泛使用的技术是在每次迭代中交替更新五个变量。这种多块交替迭代收敛到最终的答案相当慢。考虑到多块交替迭代, 该方法提出了一个贪婪追踪算法, 它在经验上比交替迭代方案的简单实现表现得更好。算法具体如图 3-1 所示, 其基本思路如下: 在每次迭代中, 首先追求图像层块的稀疏逼近。其次, 如果相关的稀疏码尽可能地有区别性, 追求雨层碎片加上图像层剩余部分的稀疏近似。最后, 在开始下一次迭代之前, 我们投影估计的补丁以适应屏幕混合模型并再次学习字典 $D$ 。

根据大量经验的观察, 序列 $I^k$ 、 $R^k$ 确实收敛。可以看出除了关于互斥的惩罚项 $\mathcal{B}(C_I)^T \mathcal{B}(C_R) \leq \varepsilon$ , 所有其他约束都在算法中被明确地执行。事实上, 算法以贪婪的方式试探性地找到了

$$\min_{I, R, D, C_I, C_R} \|PI - DC_I\|_F^2 + \|PR - DC_R\|_F^2 \quad (2-6)$$

近似解。更具体地说, 算法中有两个辅助变量 $\omega$ 和 $r$ 。变量 $\omega$ 是维数与字典大小相同的指示向量, 对应于值为 1 的原子表示这些原子用于雨层, 否则用于图像层。变量 $r$ 表示图像层中可能残留的雨层, 它将被另一轮雨层和图像层的分离进一步抑制。假设所提出的算法确实收敛, 我们有当 $k \rightarrow \infty$ 时,  $r_k$ 会收敛到零, 这意味着惩罚项 $\mathcal{B}(C_I)^T \mathcal{B}(C_R)$ 收敛到零, 也就是说最后一个约束在迭代次数足够多的情况下也能成立。

该方法提供了一个变分模型, 加上一个简单的迭代数值方法, 并且不涉及任何其他图像处理模块。相较于其余传统方法十分简单, 实验还表明, 该方法在视觉质量方

面始终优于当时的其他方法。

当然该方法也存在一定的局限性。一个是当输入图像具有许多与雨滴相似的结构时，所提出的方法可能不能很好地工作，另一个是所提出的方法不适用于雨滴被放大的雨图像。

```

Input : rain image  $J$ 
Output: de-rained image  $I^K$  and rain layer  $R^K$ 
Initial : dictionary  $D^0$ , sparse codes  $C_1^0, C_2^0$ , two
           patch arrays  $Y_1^0, Y_2^0$ , exclusive indicator  $w^0$ 
for  $k = 0, 1, \dots, K$ , do
    (1) sparse coding:

         $C_1^k := \operatorname{argmin}_C \|Y_1^{k-1} - D^k C\|_F^2$ 
        subject to  $\|C[:, j]\|_0 \leq T_1$ ;

         $r^k := D^k \operatorname{diag}(\omega^{k-1}) C_1^k$ ;
         $C_2^k := \operatorname{argmin}_C \|Y_2^{k-1} + r^k - D^k C\|_F^2$ 
        subject to  $\|C[:, j]\|_0 \leq T_2$ .

    (2) updating the patches of rain layers:

         $Y_2^k := D^k C_2^k$ .

    (3) updating both layers:

         $R^k \Leftarrow Y_2^k$  (clipped to  $[0, 1]$ );

         $I^k := (J - R^k) ./ (1 - R^k)$ .          (9)

    (4) sampling patches:

         $Y_1^k := \mathcal{P} I^k; Y_2^k := \mathcal{P} R^k$ ;

         $\omega^k[j] = \begin{cases} 1 & \text{if } \mathcal{B}(C_2^k)[j] \neq 0 \\ 0 & \text{otherwise.} \end{cases}$ 

         $w^k = w^{k-1} \cdot \omega^k$ .

    (5) updating dictionary:

         $D^k := \operatorname{argmin}_D \|Y_1^k - D C_1^k\|_F^2 + \|Y_2^k - D C_2^k\|_F^2$ .

        subject to  $\|D_j\|_2 = 1$ , for all  $j$ .
end
    
```

图 2.1-1 稀疏图像编码去雨方法伪代码



## 2.1.2 实验与结果



图 2.1-2 去雨效果对比图（建筑）

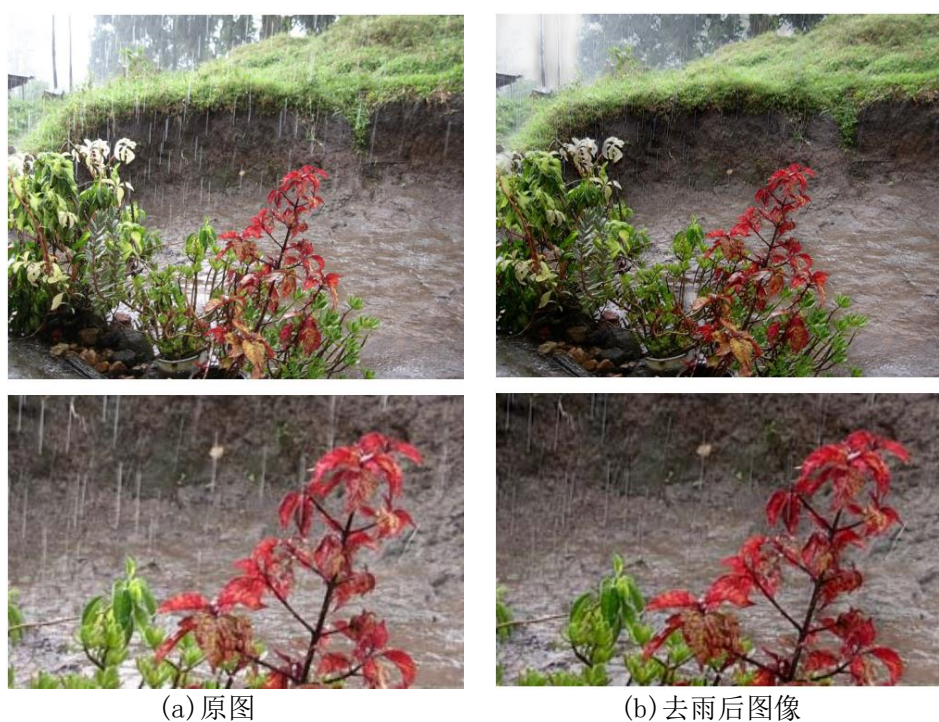


图 2.1-3 去雨效果对比图（植物）

如图 2.1-2 所展示的是建筑图像的去雨效果图，可以看到，去雨操作可以去除大部

分的雨条纹，但是还是会保留一部分的伪影子。但是总体来说，图像的大部分信息都能够被还原。而图 2.1-3 所示的植物图像的去雨效果图，除了去除大部分的雨条纹之外，还保留了图片中红叶的纹理。证明了这个方法的还原效果还是比较良好的。

## 2.2 深度学习方式的去雨算法

### 2.2.1 背景模型建立

在去雨方面，广泛运用的数学模型：

$$O = B + \tilde{S} \quad (2-7)$$

其中 $B$ 表示背景图， $\tilde{S}$ 表示雨条纹图， $O$ 表示带有雨条纹的图像。

Huang 等人<sup>[38]</sup>试图通过稀疏编码和 HOG 中学习到的特征将雨纹从高频层中分离出来，Kim 等人<sup>[39]</sup>首先检测雨条纹，然后用非局部均值滤波器去除它们。Luo 等人<sup>[42]</sup>提出了一种判别稀疏编码方法，用于从背景图像中分离降雨条纹。最近的一项工作<sup>[41]</sup>利用高斯混合模型来分离雨条纹，实现了最先进的性能，然而，仍然略有平滑的背景。

以上的方法都将去雨看作是信号分离的问题，基于背景层和雨水层各自的独特特征对其进行分离。然而以上方法存在以下几个缺陷：

- 1) 由于图像雨纹与背景纹理存在内在重叠，大多数方法都倾向于在非雨区去除纹理细节，这样就会导致区域过度平滑。
- 2) 现有模型对于复杂雨图效果不好，比如说雨条纹积累以及不同形状和方向的雨条纹。
- 3) 现有模型对于雨水特征的提取较为简单，没有利用到空间背景信息。

由此可以看出：如何更好地提取背景层与雨水层的特征以及如何有效处理雨条纹积累对于去雨来说至关重要。

### 2.2.2 JORDER 模型

前面提到原来公认的雨图模型中， $\tilde{S}$ 需要同时包含两个信息：雨水位置和该位置上的雨水密度信息。但是我们知道图像中不同区域的雨的疏密程度不同，很难用统一的稀疏度对 $\tilde{S}$ 进行建模。而且不区分雨区和非雨区，由于图像雨纹与背景纹理存在内在重叠，在进行去雨过程中也将造成非雨区过于平滑。

针对上述的问题，作者提出了新的雨图模型，在原有模型基础上加入了一个 region-

dependent 的变量  $R$ ，表征单个雨线的位置。注意  $R$  是 binary map，其中 1 表示该像素点位于雨区，0 则反之。

$$O = B + SR \quad (2-8)$$

这样做不但给网络学习雨区提供了额外的信息，而且有助于构建一个新的检测流程：先检测雨区，再对雨区和非雨区进行不同操作，避免了对非雨区细节的损失。

在现实世界中，雨的外观不仅是由个别的雨条纹形成的，而且是由雨条纹的积累形成的。当雨积密度大时，个别的条纹不能清楚地观察到。这种雨条纹的积累，其视觉效果类似于雾。

为了适应这两种现象(即不同方向的雨纹积累和重叠雨纹)，作者提出了包含雾效和多层雨线的新模型：

$$O = \alpha \left( B + \sum_{t=1}^s \tilde{S}_t R \right) + (1 - \alpha) A \quad (2-9)$$

其中  $\tilde{S}_t$  表示一层拥有相同方向的雨线，对不同方向雨线求和，可以得到雨线累计的效果； $\alpha$  表示传输率，即大气散射中的成像模型中表示该点的光强衰减之后，进入成像设备的比例； $A$  表示环境光。

在加入传输率与环境光之后，该模型可以有效地表示雨线积累形成的雾效效果。而且我们可以看出这种雾效是通过不同方向上的雨线模型叠加而成，所以可以迭代地对图像进行雨线去除。

作者构建了一个多任务网络来执行 JOint Rain DEtection and Removal (JORDER)，该网络通过端到端学习来解决公式。下图 2.2-1 展示了 JOint Rain DEtection and Removal 的流程。

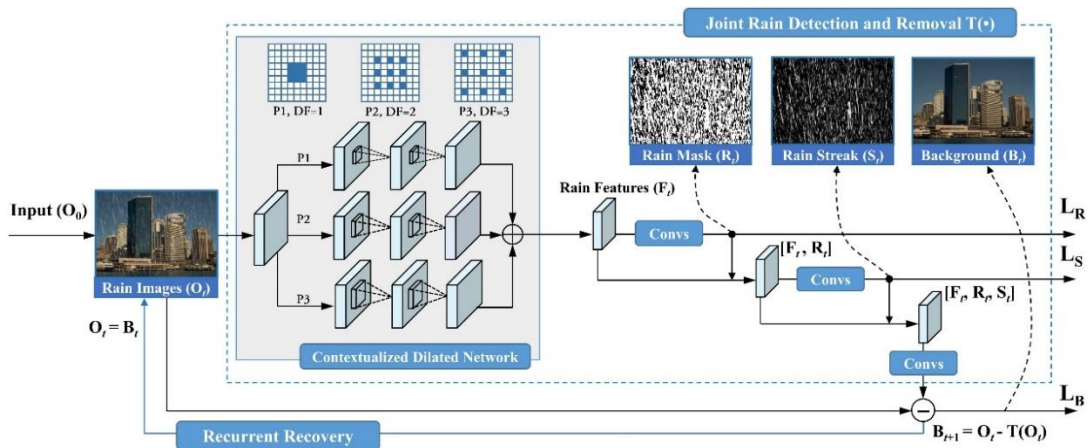


图 2.2-1 JOint Rain DEtection and Removal



根据公式(2-9),我们知道通过给定降雨图像 $O$ ,我们的目标是估计 $B$ 、 $S$ 和 $R$ 。作者采用了最大后验(MAP)估计:

$$\arg \min_{B,S,R} ||O - B - SR||_2^2 + P_b(B) + P_s(S) + P_r(R) \quad (2-10)$$

其中,  $P_b(B)$ 、 $P_s(S)$ 、 $P_r(R)$ 分别为 $B$ 、 $S$ 、 $R$ 上的先验概率,从训练数据中学习 $B$ 、 $S$ 、 $R$ 的先验概率并隐式嵌入到网络中。

$B$ 、 $S$ 、 $R$ 的估计是内在相关的。因此,  $B$ 的估计受益于预测的 $\hat{S}$ 和 $\hat{R}$ 。所以作者提出了一个多任务架构,该架构可以根据 $R$ 、 $S$ 和 $B$ 使用多个损失函数进行训练(见图 2.2-1 中的破折号)。

如图 2.2-1 所示,作者首先利用 Contextualized Dilated Networks 来提取雨的特征表示 $F$ 。随后,对 $R$ 、 $S$ 和 $B$ 按顺序进行预测,这意味着这是一个雨条纹检测、估计和去除的连续过程。每一个都是基于 $F$ :

1.  $R$ 由 $F$ 的卷积过程估计;
2.  $S$ 是由 $F$ ,  $\hat{R}$ 串联卷积过程进行预测;
3.  $B$ 是由 $F$ ,  $\hat{R}$ ,  $\hat{S}$ ,  $O - \hat{R}\hat{S}$ 串联卷积过程进行预测。

作者提出 Contextualized Dilated Networks 来更好的表征雨水的特征。该模型有两个特征:

- 1) 它是一个循环网络,可以提供一个更大的接受域;
- 2) 该模型将初步提取到的特征经过三个感知野逐渐扩大的卷积路径,利用扩张卷积挖掘不同尺度的区域性情境信息,为检测雨水提供更好的表征。

这一部分的模型结构如下图:

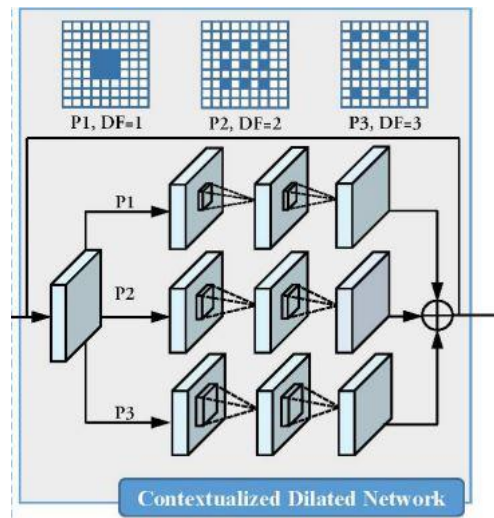


图 2.2-2 Contextualized Dilated Networks

我们可以看出扩张卷积用放大因子的步长对像素进行加权,从而在不失去分辨率的情况下增加其接受域。三个扩张路径由两个卷积组成相同的内核大小  $3 \times 3$ 。然而,随着不同的扩张因素,不同的路径有自己的接受域。比如上图中路径P2由两个带扩张因子的卷积组成,它的卷积核表示为  $DF = 2$  的情况。因此,通过两个卷积,这三条路径的可分辨域分别为  $5 \times 5$ 、 $9 \times 9$  和  $13 \times 13$ 。

设  $F_{rr}(\cdot)$ ,  $F_{rs}(\cdot)$ ,  $F_{bg}(\cdot)$  是由学习网络生成的逆恢复函数模型,分别估计基于输入的雨图像  $O$  的雨条纹二元图  $\hat{R}$ , 雨条纹图  $\hat{S}$  和背景图像  $\hat{B}$ 。使用  $\Theta$  集合表示模型的所有网络参数。

我们使用  $n$  组相应的降雨图像,背景图像,降雨区域图和降雨条纹图  $\{(o_i, g_i, r_i, s_i)\}_i^n$  进行训练,这  $n$  组训练数据是使用自己合成的带雨的图像来进行训练。我们采用以下联合损失函数对  $\Theta$  参数化的网络进行训练从而能够根据降雨图像  $o_i$  联合估计  $r_i$ 、 $s_i$  和  $g_i$ 。

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^n (\|F_{rs}(o_i; \Theta) - s_i\|^2 + \lambda_1 \|F_{bg}(o_i; \Theta) - g_i\|^2 - \lambda_2 (\log \hat{r}_{i,1} r_{i,1} + \log(1 - \hat{r}_{i,2}) (1 - r_{i,2})) \quad (2-11)$$

其中

$$\hat{r}_{i,j} = \frac{\exp \{F_{rs,j}(o_i; \Theta)\}}{\sum_{k=1}^2 \exp \{F_{rs,k}(o_i; \Theta)\}}, j = \{1, 2\}$$

参数  $\lambda_1$  和  $\lambda_2$  为权重因子。

### Changing Parameters (调参过程)

关于 JORDER 模型和 DehazeNet 模型一样,也有很多参数,但它是 DehazeNet 模型的进一步提升。因为随着降雨量增大会带来雾效,因此 JORDER 模型中有运用到 DehazeNet 模型,以消除降雨所带来的雾效。在这里,模型的参数选择和略有不同,取其学习率为 0.0006,训练次数为 50000 次,每次迭代的数据样本为 128 个,扫描半径  $\epsilon$  为 0.001,初始半径  $r$  为 50,以及去雾程度  $w=0.95$ ,最小透射率值  $t=0.25$ 。

除了 JORDER 模型中关于 DehazeNet 模型部分需要进行参数选择以外。它主要需要进行的参数调整还有 SRCNN 模型中的学习率、训练轮数和迭代数量。JORDER 模型会将 SRCNN 模型中确定这三个参数训练后的输出量作为其输入数据,根据 JORDER 模型自身的损失函数和优化方法,利用学习率计算下降梯度的平滑率,来对学习率进行自适应调整,最后当模型的准确率达到要求以后,再利用 DehazeNet 模型消除降雨带来的雾效。

而 SRCNN 模型有三层网络结构,它需要分别设置学习率,给出的参考学习率是前两层为 0.0001,最后一层为 0.00001,此外还给出了模型的参考训练轮数 15000 和每次迭代的样本量 128。

因此本文通过调整 JORDER 模型的学习率、模型训练轮数以及模型每轮迭代训练中的样本量,最终选择最合适的模型参数。下图 2.2-3 展示了调整模型学习率所得到的效果图,这里采用的训练轮数和迭代样本量是已经完成训练得到的最优解,跟模型给出的参考量一致,训练轮数为 15000,每次迭代的样本量为 128。

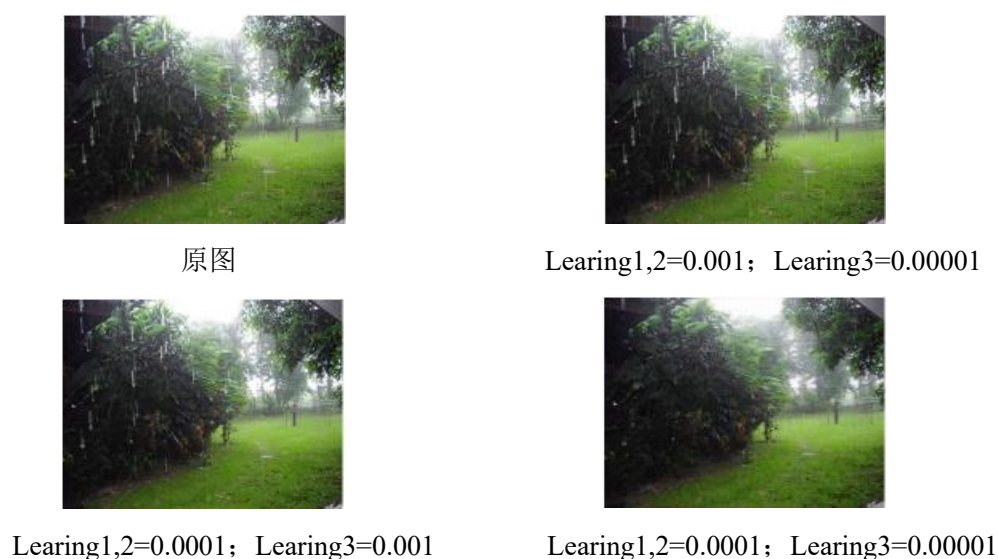


图 2.2-3 不同去雨模型参数在相同真实图像下的去雨效果比较

最终通过数次调参发现还是参考学习率的效果最佳,其余两种的学习率效果很差,跟原图几乎没有任何区别,因此最终选定模型前两层网络的学习率为 0.0001,最后一层网络的学习率为 0.00001,训练轮数为 15000,每次迭代样本量为 128。

### 2.2.3 实验和结果

#### ● Datasets

我们在一些基准数据集上比较了我们的方法和最新的方法:(1)Rain12<sup>[41]</sup>,包括 12 张合成雨图像,其中只有一种类型的雨条纹;Rain100L,仅包含一种雨纹的合成数据集;(2)Rain20L,是补充材料中用于测试竞争网络架构的 Rain100L 的子集;(3) Rain100H,这是我们有五个条纹方向的合成数据集。请注意,虽然真正的雨图像很少包含多个不同方向的雨条纹,但合成这类图像进行训练可以提高网络的容量。

#### ● 去雨算法结果比较

我们比较了 JORDER 和两种最先进的方法:鉴别稀疏编码(DSC),层先验(LP),训练



后进行去雨。图 2.2-4 是一张真实雨图经过几种方法去雨后的效果图。我们可以看出 JORDER 算法的去雨效果非常好，成功去除大部分雨纹。



图 2.2-4 不同方法的去雨效果图（一）

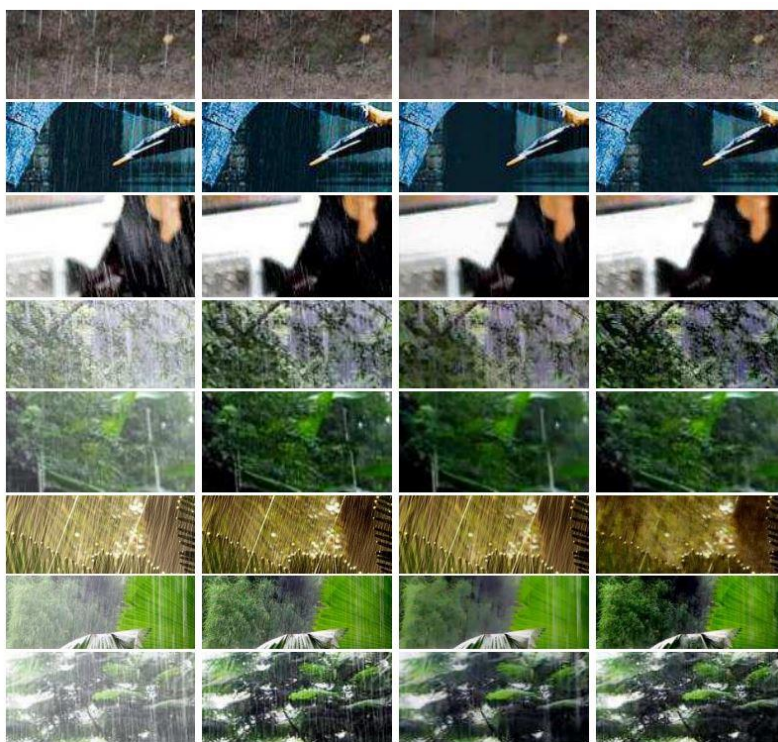


图 2.2-5 各种方法去雨效果图（二）

图 2.2-5 展示了更多去雨图的结果，从左到右分别是真实雨图，DSC，LP，JORDER。可以看出我们的方法明显优于其他方法，JORDER 可以成功去除大部分雨纹。



## 2.2.4 深度学习算法与传统图像算法的比较

传统去雨算法 DSC 是基于字典学习的单幅图像去雨算法。基本思想是在具有强互斥性的学习字典上，采用高分辨率稀疏逼近双层补丁。该鉴别稀疏编码可以从非线性组合中精确分离两层。我们比较深度学习算法 JORDER 和传统去雨算法 DSC 对图像去雨后的效果来说明深度学习算法的优点与不足。



图 2.2-6 JORDER 和 DSC 去雨效果图。从左到右分别为原图、DSC 复原图和 JORDER 复原图



图 2.2-7 JORDER 和 DSC 去雨效果图。从左到右分别为原图、DSC 还原图和 JORDER 还原图

图 2.2-6 和图 2.2-7 是 JORDER 和 DSC 对雨图去雨后的效果，从左到右分别是真实雨图，DSC，JORDER。我们可以看出，无论是暴雨还是小雨，JORDER 算法的去

雨效果更加明显，成功去除了大部分雨纹；而 DSC 算法对于小雨环境下的图像(图 2.2-6 第一行，图 2.2-7 第一行图像)的去雨效果比较明显，但是对于大颗粒雨滴（图 2.2-6 第二行图像）以及暴雨环境下的图像（图 2.2-7 第二行图像）几乎达不到去雨的效果。所以可以知道相对于 DSC，JORDER 去雨效果是更好的。但是我们也会发现，尽管 JORDER 去雨效果很好，但从其去雨的效果图可以看出，它容易使得图像变暗，而且会出现图像细节模糊的现象；而 DSC 算法对于整体图像的亮度不会有太大影响，而且图像信息会更加清晰。

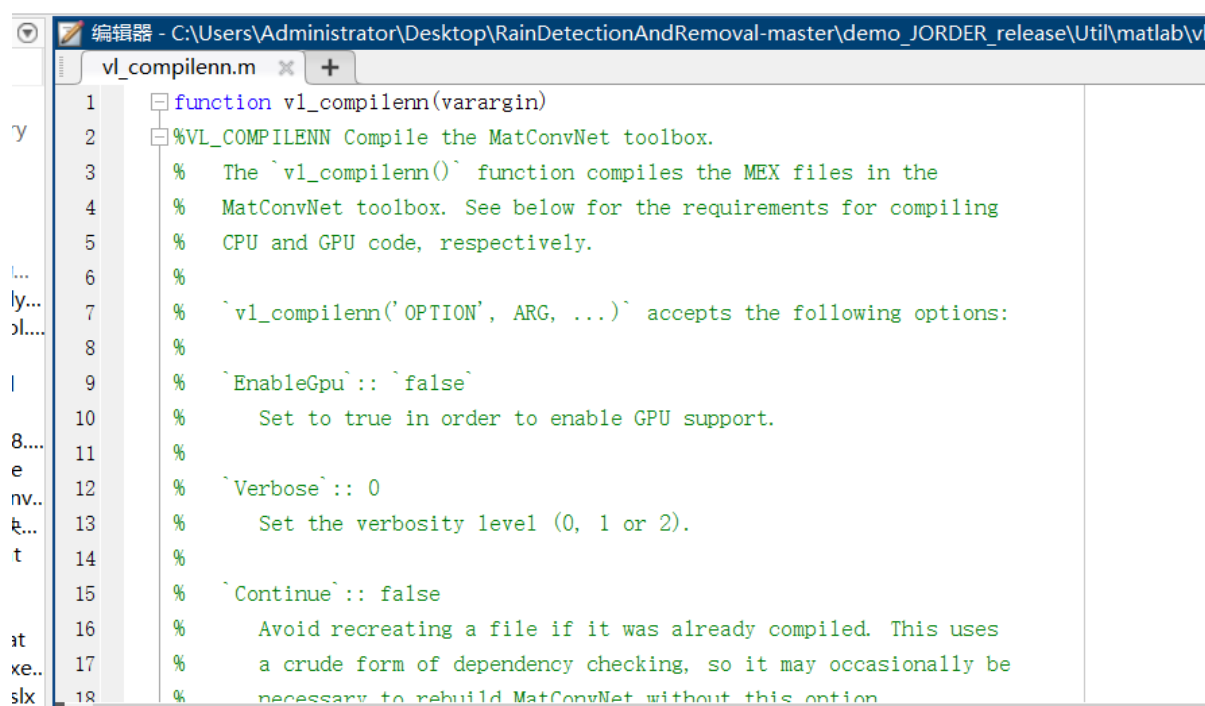
### 第 3 章 总结

这次的课程设计中，我复现了去雾、去雨两个任务中非常经典的 4 种算法。其中去雾任务中我们分别复现了暗通道估计方法和 DehazeNet 深度学习方法，去雨任务中我们复现了稀疏编码图像去雨方法和 JORDER 深度学习方法。这两种方法的前者为传统图像处理方法，后者为基于深度学习框架的处理方法。通过比较他们在实际图像上的去雾或者去雨效果，总结了基于传统图像处理方法和基于深度学习的方法各自的优势和劣势。

实验发现深度学习和传统图像处理方式他们各有优劣，例如传统图像处理方法普遍能在短时间内取得很好的去雾、去雨效果，并且他们的可解释性更强，但他们的劣势在于他们的方法过于依赖建立模型的优劣程度，如果对图像的先验知识并不符合某一张图像，那么他们复原的图像就变得非常糟糕。深度学习方法虽然在多数图像上都取得了很好的复原效果，并且并不需要很多、很准确的先验模型，但深度学习模型的方法对训练数据的数量和质量要求较高，并且普遍需要较长的训练过程。

我在这次复现的实验过程中，学习到了很多关于参数调整的知识，是之前在 PaddlePaddle 平台上做实验没有学习到的知识点。这里的模型更加多元化、更灵活、精度也更好，是在前人的基础上修改了参数，优化了模型。我在学习 JORDER 模型和 DehazeNet 模型的过程中让我掌握了很多知识也解决了很多调参问题。我起初想调整 JORDER 模型中学习率、训练轮数等参数，但是详细阅读基础的经典代码后发现，并没有任何参数信息的显示，都是直接调用了某个 model 来实现的，我明白了这个模型应该是封装的，于是一步一步的进去寻找参数，当我找完所有关于 JORDER 模型的封装代码后依旧没有找到这几个参数，但我发现了这里有调用另外一个文件夹中的

DehazeNet 以及 SRCNN 模型，于是我结合之前阅读的文献资料，明白了 DehazeNet 模型是 JORDER 模型用于处理降雨量过大图片中雾气效果的，而 SRCNN 模型是用于处理图片的，处理完降雨图片以后再进入 JORDER 的调用 model 中。于是我接下去进入 SRCNN 模型中，便很快找到了模型的参数信息，内有学习率、训练轮数等一系列我之前在做实验中非常常见的参数信息，并且也是我熟悉的，于是我就通过调整 SRCNN 中的参数信息，最后观察 JORDER 模型的去雨效果图片，寻找最优的参数，以此达到实验效果。另外在这次实验中因为 JORDER 部分代码是需要通过 MATLAB 来实现的，而这个模型的训练量非常大，需要使用服务器的 GPU 来跑，因此我还需要自己来书写在 MATLAB 中使用服务器 GPU 跑 JORDER 中部分模型的训练代码。



```
1 function vl_compilenn(varargin)
2 %VL_COMPILENN Compile the MatConvNet toolbox.
3 % The `vl_compilenn()` function compiles the MEX files in the
4 % MatConvNet toolbox. See below for the requirements for compiling
5 % CPU and GPU code, respectively.
6 %
7 % `vl_compilenn('OPTION', ARG, ...)` accepts the following options:
8 %
9 % `EnableGpu`:: `false`
10 % Set to true in order to enable GPU support.
11 %
12 % `Verbose`:: 0
13 % Set the verbosity level (0, 1 or 2).
14 %
15 % `Continue`:: false
16 % Avoid recreating a file if it was already compiled. This uses
17 % a crude form of dependency checking, so it may occasionally be
18 % necessary to rebuild MatConvNet without this option
```

我在网络上有找到关于这部分的开源代码，但是不同的 MATLAB 版本不同会造成不一样的情况，另外还需要如果要在 MATLAB 中采用服务器 GPU 跑代码还需要加装工具箱，这部分还是消耗了我很多时间，但我也学到了好多知识点，学会如何修改 GPU 代码在不同版本 MATLAB 上，因为在 MATLAB 不是像 PaddlePaddle 平台上，直接 device 一下然后选择 GPU 就可以实现的，MATLAB 的信息传输还是跟别的有所不同。

总得来说这次大作业让我学习到了很多知识，有些是之前没有遇到的，因为之前在 PaddlePaddle 平台上做的实验大部分都是搭建好环境，直接调整参数运行。这次自己复现，特别是 JORDER 代码，有一部分是需要通过 MATLAB 来实现的，这是我从

未有过的尝试，因为在此之前从来没有想过用 MATLAB 来跑深度学习的模型，因为以前常用 MATLAB 做机器人仿真或者矩阵运算。但是通过这次的学习让我更加多元化的认识了人工智能中的各种机器学习算法，给我拓宽了视野，让我在未来的科研之路上能够更好地发挥主观能动性，培养自身的发散思维，运用自己在本科中所学知识解决问题。



## 参 考 文 献

- [1] He K, Sun J, Tang X. Single image haze removal using dark channel prior[J]. IEEE transactions on pattern analysis and machine intelligence, 2010, 33(12): 2341-2353.
- [2] He K, Sun J, Tang X. Guided image filtering[C]//European conference on computer vision. Springer, Berlin, Heidelberg, 2010: 1-14.
- [3] E. J. McCartney, "Optics of the atmosphere: scattering by molecules and particles," New York, John Wiley and Sons, Inc., vol. 1, 1976.
- [4] S. K. Nayar and S. G. Narasimhan, "Vision in bad weather," in IEEE International Conference on Computer Vision, vol. 2, 1999, pp. 820–827.
- [5] S. G. Narasimhan and S. K. Nayar, "Contrast restoration of weather degraded images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 6, pp. 713–724, 2003.
- [6] K. Tang, J. Yang, and J. Wang, "Investigating haze-relevant features in a learning framework for image dehazing," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 2995–3002.
- [7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9.
- [8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, pp. 1–42, 2014.
- [9] M. Sulami, I. Glatzer, R. Fattal, and M. Werman, "Automatic recovery of the atmospheric light in hazy images," in IEEE International Conference on Computational Photography (ICCP), 2014, pp. 1–11.
- [10] G. Meng, Y. Wang, J. Duan, S. Xiang, and C. Pan, "Efficient image dehazing with boundary constraint and contextual regularization," in IEEE International Conference on Computer Vision (ICCV), 2013, pp. 617–624.
- [11] J.-P. Tarel and N. Hautiere, "Fast visibility restoration from a single color or gray level image," in IEEE International Conference on Computer Vision, 2009, pp. 2201–2208.
- [12] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 12, pp. 2341–2353, 2011.
- [13] 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.
- [14] K. Tang, J. Yang, and J. Wang, “Investigating haze-relevant features in a learning framework for image dehazing,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 2995–3002.
- [15] Luo Y, Xu Y, Ji H. Removing rain from a single image via discriminative sparse coding[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 3397-3405.
- [16] F. Agostinelli, M. R. Anderson, and H. Lee. Adaptive multi-column deep neural networks with application to robust image denoising. In Proc. Annual Conf. Neural Information Processing Systems. 2013. 2
- [17] C. Ancuti, C. O. Ancuti, T. Haber, and P. Bekaert. Enhancing underwater images and videos by fusion. In Proc. IEEE Int’l Conf. Computer Vision and Pattern Recognition, pages 81–88, June 2012. 3
- [18] P. C. Barnum, S. Narasimhan, and T. Kanade. Analysis of rain and snow in frequency space. Int’l Journal of Computer Vision, 86(2-3):256–274, 2010. 1
- [19] J. Bossu, N. Hautiere, and J.-P. Tarel. Rain or snow detection in image sequences through use of a histogram of orientation of streaks. International journal of computer vision, 93(3):348–367, 2011. 1
- [20] N. Brewer and N. Liu. Using the shape characteristics of rain to identify and remove rain from video. In Joint IAPR International Workshops on SPR and SSPR, pages 451–458, 2008. 1
- [21] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising with multi-layer perceptrons, part 1: comparison with existing algorithms and with bounds. arXiv:1211.1544. 2
- [22] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising with multi-layer perceptrons, part 2: training trade-offs and analysis of their mechanisms. arXiv:1211.1552. 2
- [23] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. IEEE Trans. on Image Processing, PP(99):1–1, 2016. 2, 6
- [24] Y.-L. Chen and C.-T. Hsu. A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In Proceedings of the IEEE International Conference on Computer Vision, pages 1968–1975, 2013. 1, 2
- [25] Z. Cui, H. Chang, S. Shan, B. Zhong, and X. Chen. Deep network cascade for image super-

- resolution. In Proc. IEEE European Conf. Computer Vision. 2014. 2
- [26] C. Dong, C. Loy, K. He, and X. Tang. Image superresolution using deep convolutional networks. TPAMI, 2015. 2
- [27] C. Dong, C. C. Loy, K. He, and X. Tang. Image superresolution using deep convolutional networks. In ECCV. 2014. 2
- [28] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In Proc. IEEE Int’l Conf. Computer Vision, December 2013. 1, 2, 6, 8
- [29] K. Garg and S. K. Nayar. Detection and removal of rain from videos. In Proc. IEEE Int’l Conf. Computer Vision and Pattern Recognition, volume 1, pages I– 528, 2004. 1
- [30] K. Garg and S. K. Nayar. Photorealistic rendering of rain streaks. In ACM Trans. Graphics, volume 25, pages 996–1002, 2006. 1, 6
- [31] K. Garg and S. K. Nayar. Vision and rain. Int’l Journal of Computer Vision, 75(1):3–27, 2007. 1
- [32] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. arXiv:1508.06576, 2015. 2
- [33] D.-A. Huang, L.-W. Kang, Y.-C. F. Wang, and C.-W. Lin. Self-learning based image decomposition with applications to single image denoising. IEEE Transactions on multimedia, 16(1):83–93, 2014. 1
- [34] D.-A. Huang, L.-W. Kang, M.-C. Yang, C.-W. Lin, and Y.-C. F. Wang. Context-aware single image rain removal. In Proc. IEEE Int’l Conf. Multimedia and Expo, pages 164–169, 2012. 2, 3, 4
- [35] Q. Huynh-Thu and M. Ghanbari. Scope of validity of psnr in image/video quality assessment. Electronics letters, 44(13):800–801, 2008. 6
- [36] V. Jain and S. Seung. Natural image denoising with convolutional networks. In Proc. Annual Conf. Neural Information Processing Systems. 2009. 2
- [37] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In ACM Trans. Multimedia, pages 675–678, 2014. 6, 8
- [38] L. W. Kang, C. W. Lin, and Y. H. Fu. Automatic single-image-based rain streaks removal via image decomposition. IEEE Trans. on Image Processing, 21(4):1742–1755, April 2012. 1, 2, 4, 6, 8
- [39] J.-H. Kim, C. Lee, J.-Y. Sim, and C.-S. Kim. Single image deraining using an adaptive nonlocal means filter. In IEEE Trans. on Image Processing, pages 914– 917, 2013. 1
- [40] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown. Rain streak removal using layer priors.

- In Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, pages 2736–2744, 2016. 2, 3, 4, 6, 7, 8
- [41] K. G. Lore, A. Akintayo, and S. Sarkar. Llnet: A deep autoencoder approach to natural low-light image enhancement. arXiv preprint arXiv:1511.03995, 2015. 2
- [42] Y. Luo, Y. Xu, and H. Ji. Removing rain from a single image via discriminative sparse coding. In Proc. IEEE Int'l Conf. Computer Vision, pages 3397–3405, 2015. 1, 2, 3, 4, 6, 7, 8
- [43] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proc. IEEE Int'l Conf. Computer Vision, volume 2, pages 416–423, July 2001. 6
- [44] S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. Int'l Journal of Computer Vision, 48(3):233–254, 2002. 3
- [45] C. Osendorfer, H. Soyer, and P. van der Smagt. Image super-resolution with fast approximate convolutional sparse coding. In Neural Information Processing. 2014. 2
- [46] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Scholkopf. Learning to deblur. " arXiv:1406.7444, 2014. 2
- [47] S.-H. Sun, S.-P. Fan, and Y.-C. F. Wang. Exploiting image structural similarity for single image rain removal. In Proc. IEEE Int'l Conf. Image Processing, pages 4482–4486, 2014. 1
- [48] Y. Tian and S. G. Narasimhan. Seeing through water: Image restoration using model-based tracking. In Proc. IEEE Int'l Conf. Computer Vision, pages 2303– 2310, Sept 2009. 3
- [49] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 2010. 2
- [50] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE Trans. on Image Processing, 13(4):600–612, 2004. 6
- [51] H. Z. J. P. X. C. Wenqi Ren, Si Liu and M.-H. Yang. Single image dehazing via multi-scale convolutional neural networks. In Proc. IEEE European Conf. Computer Vision, pages 914–917, October 2016. 2
- [52] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In Proc. Annual Conf. Neural Information Processing Systems. 2012. 2
- [53] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image

- deconvolution. In Proc. Annual Conf. Neural Information Processing Systems. 2014. 2
- [54]Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu. Automatic photo adjustment using deep neural networks. ACM Trans. Graphics, 2015. 2
- [55]W. Yang, J. Feng, J. Yang, F. Zhao, J. Liu, Z. Guo, and S. Yan. Deep Edge Guided Recurrent Residual Learning for Image Super-Resolution. ArXiv, April 2016. 5
- [56]F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In International Conference on Learning Representation, 2016. 5
- [57]K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. ArXiv e-prints, August 2016. 2
- [58]X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng. Rain removal in video by combining temporal and chromatic properties. In Proc. IEEE Int’l Conf. Multimedia and Expo, pages 461–464, 2006. 1