arXiv:2303.00755v1 [eess.IV] 1 Mar 2023

用于图像去噪的 Cloud K-SVD

ChristianMarius Lillelund https://orcid.org/0000-0002-2520-37741, 1*†, Henrik BaggerJensen[1, 1†] 和 Christian Fischer Pedersenhttps://orcid.org/0000-0002-1266-5857[1,1]

1*丹麦奥胡斯 N 区 Finlandsgade 22 号，邮编 DK-8200，奥胡斯大学电气与计算机工程系。

通讯作者。电子邮箱：cl@ece.au.dk；其他作者：
201304157@post.au.dk；cfp@ece.au.dk；
†这些作者对本研究的贡献相同。

摘要

Cloud K-SVD 是一种字典学习算法，可在多个节点上进行训练，从而生成一个共享字典，用于表示图像数据中的低维几何结构。我们展示了该算法的一项新应用：利用它从重叠图像块中恢复无噪声图像和噪声图像。我们使用 Kubernetes 中的 Docker 容器实现了节点网络，以支持 Cloud K-SVD。结果表明，Cloud K-SVD 能够近似恢复图像，并从基准灰度图像中去除可量化的噪声，同时不影响恢复精度；在噪声水平为（$\mu = 0$、 $\sigma^2 = 0.01$、0.005、0.001）时，干净图像与恢复图像之间的结构相似性指数分别达到 0.88、0.91 和 0.95，与该领域当前最先进水平相近。显然，Cloud K-SVD 能够在多个节点上学习共享字典，并去除图像中的加性高斯白噪声。该共享字典可用于在网络中的任意节点恢复特定图像。

关键词： Cloud K-SVD，字典学习，分布式系统，图像去噪

1 引言

我们所处的大数据时代，从广泛的数据源中产生了海量的宝贵数据 [1]。仅由社交网络收集的数据就包括数百万张图像、视频和消息。每天产生的社交数据几乎达到500TB [2]。随着数据的激增，科学界对信号的稀疏近似越来越感兴趣 [3]，并将其应用于压缩、去噪、复原和分类等领域 [3–7]。稀疏近似中的一个核心问题是设计一个包含基本信号原子的字典，利用该字典可以描述信号通常是这些原子的线性组合。对于这项任务，我们可以使用预定义的固定字典，例如过完备小波字典或 Gabor字典 [8]，，也可以通过将随机字典适配到一组训练信号来构建自己的字典，这种做法称为字典学习 [3–7, 9]，例如 Aharon、Elad 和 Bruckstein 于 2006年提出的广受认可的 K-SVD 算法 [5]。此后，K-SVD 被广泛采用，例如利用时间相关性恢复受损的云和阴影图像 [10]，或利用 K-SVD 训练全局字典以去除不需要的噪声，从而提升羽毛图像的质量 [11]。2016年，Raja 和 Bajwa [12] 发表了 Cloud K-SVD，这是一种扩展

相比传统 K-SVD，在数据量庞大或涉及隐私问题时，Cloud K-SVD 具有多项优势：它可以在多个本地节点或分布式节点上进行训练，从而生成共享字典，用于表示所有数据合并后的低维几何结构。整个过程无需在节点之间交换完整或部分数据集。当数据涉及隐私问题或数据量巨大时，这一特性尤为理想。事实上，数据在本地处理，节点之间交换的是以残差形式呈现的（中间）训练结果，而非原始数据，因此无需交换敏感信息 [13]。节点通过平均共识就每个字典原子的重要性达成一致，并协同交换和平均各自本地字典中的原子，直至它们与全局字典一致。我们展示了 Cloud K-SVD 的一种新颖应用：在一个真实的节点网络中实现该算法，并用它去除加性白高斯噪声（AWGN）（图1）。每个节点负责对图像的特定部分进行去噪，同时与其他节点协作，在此过程中就全局字典达成共识。由此生成一幅稀疏、噪声减少的图像和一个全局字典，其中汇集了所有节点上的信息。实践中，我们使用 Docker 容器实现该算法，并在 Kubernetes 中运行实验。我们将恢复图像的峰值信噪比（PSNR）和结构相似性（SSIM）与 Ghosh、Chaudhury 和 Mandal 提出的剪枝非局部均值（PNLM）算法进行比较，相关算法见 [14]。源代码可在 GitHub 上获取 [15]。



图 1 左：原始图像。中：噪声图像。右：去噪图像。来源： [8]

第 2节介绍所用方法的理论背景。第 3 节描述我们的实际配置和所用硬件。第 4节包含所有实验及其结果。最后，第 5节总结了这项工作，并给出一些最后的评述。

2 信号处理理论

本节介绍主要概念：稀疏近似、字典学习、共识平均以及 Cloud K-SVD算法。

2.1 稀疏近似

在稀疏近似中，目标是将信号 $\boldsymbol{x}$表示为字典 $\boldsymbol{D} \in \mathbb{R}^{M\times N}$ [3]中尽可能少的列（即原子）的线性组合，从而提供对该信号最简单的可能近似。该字典通常是超完备的（$M < N$），这会导致欠定线性系统 $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$，其中 $\boldsymbol{y} \in \mathbb{R}^M$ 是观测信号，而 $\boldsymbol{x} \in \mathbb{R}^N$ 是未观测的信号向量。由于 $\boldsymbol{D}$ 是欠定的，因此存在无穷多个解；在这些解中，目标是找到满足 $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$的最稀疏（非零元素最少）表示 $\boldsymbol{x}$，即 $\min \boldsymbol{x}_0 \ s.t. \ \boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$，其中 $\|\cdot\|_0$ 表示 ℓ_0伪/计数范数。通常，观测信号$\boldsymbol{y}$ 含有噪声，这会在最小化问题中引入噪声预算，并放宽等式约束：$\min \ \boldsymbol{x}_0 \ s.t. \ \boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}_2^2 \leq \varepsilon^2$。这是一个NP难问题；通常将范数从 ℓ_0 改为 ℓ_1，并通过追踪算法求得近似解，例如正交匹配追踪 [16]。用矩阵表示法可以将其表述为一个等价的最小化问题：

$$\min_{\boldsymbol{X}\in\mathbb{R}^{N\times Q}} \boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}_2 \leq \epsilon \ \text{ s.t. } \ \forall i : \boldsymbol{x}_{i0} \leq K \quad (1)$$

其中，$\boldsymbol{E}, \boldsymbol{Y} \in \mathbb{R}^{M\times Q}$, $\boldsymbol{X} \in \mathbb{R}^{N\times Q}$ K 是 $\boldsymbol{X}$各列中的非零元素数量。

2.2 字典学习

我们求解线性逆问题 $\boldsymbol{Y} = \boldsymbol{D}\boldsymbol{X}$，以 $\boldsymbol{X}$ 从 $\boldsymbol{Y}$ 采样测量值中恢复信号 $\boldsymbol{X}$的近似值。字典$\boldsymbol{D}$ 指导我们的恢复过程，帮助我们选用最佳的非零变换系数，因为它包含感兴趣数据中的几何特征数据库。字典学习是一种用于从采样测量值中记录这些特征的框架

[17]。这假设样本（例如图像块）可以由合适的超完备基中的少数几列进行线性组合，从而实现稀疏近似。令 $\boldsymbol{Y} \in \mathbb{R}^{M\times Q}$ 为向量化图像块构成的矩阵，那么如果 $\boldsymbol{Y} \approx \boldsymbol{D}\boldsymbol{X}$，每个图像块 $\boldsymbol{y}_i \in \mathbb{R}^M$在字典 $\boldsymbol{D} \in\mathbb{R}^{M\times N}$中都有稀疏表示；稀疏信号矩阵$\boldsymbol{X} \in \mathbb{R}^{N\times Q}$的每一列至多包含 K 个非零元素 [17]。因此，字典 $\boldsymbol{D}$ 是一个降维步骤，可以预先定义，也可以从数据 [4, 5, 18]中学习得到。

2.3 平均共识

平均共识 [19] 是一种让节点协作共享其状态的方法，例如字典学习场景中的当前残差向量，然后通过求平均引导节点集合达成共同的平均共识。考虑一个由一组 H 节点构成的多跳网络，该网络建模为有向图 $\mathcal{G} = (\mathcal{V}, \varepsilon)$，其中 $\mathcal{V} =1, 2, ..., H$ 表示这些 H 节点，边集 $\varepsilon = (i, j): i, j \in \mathcal{V}, i \neq j$ 由顶点对组成。顶点对(i, j)表示从 i 到 j 的有向边，即节点 i 可以向节点 j发送数据。平均协议假设每个节点i 持有某个值 $z_i \in \mathbb{R}$，并通过线性分布式迭代计算平均值$\bar{z} = \sum_{i=1}^{H} z_i/H$。每个节点都有一个局部状态变量$x_i(t = 0) = z_i$，并通过邻居状态变量的加权平均值迭代更新该值。在一定条件下，状态变量 $x_i(t)$收敛到 $\bar{z}$ [19]。接收到邻居的状态后，节点按如下方式更新其状态变量：

$$x_i(t+1) = x_i(t) + \sum_{j=1, j\neq i}^{H} \boldsymbol{W}_{ij}(t)\Big(x_j(t) - x_i(t)\Big) \tag{2}$$

其中，$\boldsymbol{W}(t) \in \mathbb{R}^{H\times H}$ 是定义为$\boldsymbol{W}(t) := \boldsymbol{I} - \varepsilon \boldsymbol{L}(t)$ 的权重矩阵，其中 $\boldsymbol{I}$是单位矩阵，$\boldsymbol{L} = [l_{ij}]$ 是网络 [19]的图拉普拉斯矩阵。

2.4 Cloud K-SVD

我们假设采用分布式设置，本地数据存储在不同的节点上。每个节点用 H_i 表示，其本地数据为 $\boldsymbol{Y}_i \in \mathbb{R}^{M\times Q_i}$，总计

Algorithm 1 The cloud K-SVD algorithm. Originally by [12]. Modifications made by authors.

Input: Local data $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_H$, parameters N and K and doubly-stochastic matrix $\boldsymbol{W}$

Initialize: Generate $\boldsymbol{d}^{ref} \in \mathbb{R}^M$ and $\boldsymbol{D}^{init} \in \mathbb{R}^{M\times Q}$ randomly, set $t_d \leftarrow 0$ and $\hat{\boldsymbol{D}}_i^{(t_d)} \leftarrow \hat{\boldsymbol{D}}^{init}$, $i = 1, 2, \ldots, H$.

1 **while** *stopping criteria not satisfied* **do**
2 $\quad t_d \leftarrow t_d + 1$.
3 $\quad$ The ith site solves $\forall q, \hat{\boldsymbol{x}}_{i,q}^{(t_d)} \leftarrow \arg\min_{\boldsymbol{x}\in\mathbb{R}^K} \| \boldsymbol{y}_{i,q} - \hat{\boldsymbol{D}}_i^{(t_d-1)}\boldsymbol{x} \|_2^2$ s.t. $\| \boldsymbol{x} \|_0 \leq K$ using SOMP
4 $\quad$ **for** $n = 1$ **to** N **do** `// Dictionary Update`
5 $\quad\quad \hat{\boldsymbol{E}}_{i,n,R}^{(t_d)} \leftarrow \boldsymbol{Y}_i\hat{\boldsymbol{\Omega}}_{i,n} - \sum_{j=1}^{n-1} \hat{\boldsymbol{d}}_{i,j}^{(t_d)}\hat{\boldsymbol{x}}_{i,j,T}^{(t_d)}\hat{\boldsymbol{\Omega}}_{i,n}^{(t_d)} - \sum_{j=n+1}^{N} \hat{\boldsymbol{d}}_{i,j}^{(t_d-1)}\hat{\boldsymbol{x}}_{i,j,T}^{(t_d)}\hat{\boldsymbol{\Omega}}_{i,n}^{(t_d)}$
6 $\quad\quad \hat{\boldsymbol{M}}_i \leftarrow \hat{\boldsymbol{E}}_{i,n,R}^{(t_d)}\hat{\boldsymbol{E}}_{i,n,R}^{(t_d)^T}$
7 $\quad\quad$ Generate $\boldsymbol{q}^{init}$, $t_p \leftarrow 0$ and $\hat{\boldsymbol{q}}_i^{(t_p)} \leftarrow \boldsymbol{q}^{init}$
8 $\quad\quad$ **while** *stopping rule* **do** `// Power Method`
9 $\quad\quad\quad t_p \leftarrow t_p + 1$
10 $\quad\quad\quad$ Set $t_c \leftarrow 0$ and $\boldsymbol{z}_i^{(t_c)} \leftarrow \hat{\boldsymbol{M}}_i\hat{\boldsymbol{q}}_i^{t_p-1}$
11 $\quad\quad\quad$ **while** *stopping rule* **do** `// Consensus`
12 $\quad\quad\quad\quad t_c \leftarrow t_c + 1$
13 $\quad\quad\quad\quad \boldsymbol{z}_i^{(t_c)} \leftarrow \sum_{j\in\mathcal{N}_i} \boldsymbol{w}_{i,j}\boldsymbol{z}_i^{(t_c-1)}$ $\iff$ the jth site has $\boldsymbol{z}_j^{(t_c-1)}$
14 $\quad\quad\quad$ **end**
15 $\quad\quad\quad \hat{\boldsymbol{q}}_i^{(t_p)} \leftarrow \hat{\boldsymbol{v}}_i^{(t_p)} / \| \hat{\boldsymbol{v}}_i^{(t_p)} \|_2$
16 $\quad\quad$ **end**
17 $\quad\quad \hat{\boldsymbol{d}}_{i,n}^{(t)} \leftarrow \mathrm{sgn}\Big(\langle \boldsymbol{d}^{ref}, \hat{\boldsymbol{q}}_i^{(t_p)} \rangle\Big)\hat{\boldsymbol{q}}_i^{(t_p)}$
18 $\quad\quad \hat{\boldsymbol{x}}_{i,n,R}^{(t)} \leftarrow \hat{\boldsymbol{d}}_{i,n}^{(t_d)^T}\hat{\boldsymbol{E}}_{i,n,R}^{(t_d)}$
19 $\quad$ **end**
20 **end**

Result: $\hat{\boldsymbol{D}}_i^{t_d}$ and $\hat{\boldsymbol{X}}_i^{t_d}$, $i = 1, 2, \ldots, H$

个 H 节点，$Q = \sum_{i=1}^{H} Q_i$ 因此数据总量为，并可表示为一个矩阵$\boldsymbol{Y} \in \mathbb{R}^{M\times Q}$。我们的算法基于 [12]，发表于1。作者所做的修改包括在 [12] （第 3 行）中使用同时正交匹配追踪（SOMP）代替常规正交匹配追踪（OMP），以及计算残差向量 $\boldsymbol{z}^{(t_c)}{}_i$ 时加入一个条件：相邻的 j^{th} 站点必须

可达并持有 $z^{(t_c-1)}{}_j$ （第 13 行），否则在更新 $\hat{}d_{i\,n}^{(t)}$,字典元素时不计入其残差。下面我们将回顾该算法。第一步是使用任意稀疏近似算法计算局部信号矩阵$\hat{}X^{(t_d)}{}_i$。此步骤如下：

$$\forall q, \hat{\boldsymbol{x}}_{i,q}^{(t_d)} = arg \min_{\boldsymbol{x}\in\mathbb{R}^N} \| \boldsymbol{y}_{i,q} - \hat{\boldsymbol{D}}_i^{(t_d-1)}\boldsymbol{x} \|_2^2 \quad \text{subject to } \| \boldsymbol{x} \|_0 \leq K \tag{3}$$

其中 $\boldsymbol{y}_{i,\ q}$ 和 $\hat{}\boldsymbol{x}^{(t_d)}{}_{i,\ q}$ 分别表示节点 i处的 q^{th} 样本和信号向量。

只要 $\hat{}\boldsymbol{D}^{(t_d-1)}{}_i$ 中的字典原子彼此保持接近，[12]在每次迭代中于各节点本地计算稀疏系数是可接受的。接下来是字典更新步骤。为此，Cloud K-SVD 使用分布式幂迭代模型来寻找平方半正定残差矩阵 $\boldsymbol{M}$的主特征向量，该向量记为 $\boldsymbol{q}$ ；该矩阵被定义为不含小于或等于零的特征值。幂方法表示为 $\hat{}\boldsymbol{q}^{(t_d)} = \boldsymbol{M}\hat{}\boldsymbol{q}^{(t_d-1)}$，其中$\hat{}\boldsymbol{q}^{(t_d)}$ 是主特征向量的估计值，并且在每次迭代中（$t_d \to \infty$）都趋近于 $\boldsymbol{q}$ 。我们将 $\boldsymbol{q}$ 的初始值记为 $\boldsymbol{q}^{init}$ ，它可以是非零向量。所有节点都以初始向量 $\boldsymbol{q}^{init}$ 和矩阵 $\boldsymbol{M}_i$开始。在幂方法的每次迭代中，各节点就矩阵 $\boldsymbol{M}$的 $\boldsymbol{q}$ 估计值达成平均共识，其定义为 $\boldsymbol{M} = \sum_{i=1}^{H} \boldsymbol{M}_i$，其中 H 表示节点数。如果我们将幂方法应用于 $\boldsymbol{M}$，则可观察到：

$$\hat{\boldsymbol{q}}^{(t_d)} = \boldsymbol{M}\hat{\boldsymbol{q}}^{(t_d-1)} = \Big(\sum_{i=1}^{H} \boldsymbol{M}_i\Big)\hat{\boldsymbol{q}}^{(t_d-1)} = \sum_{i=1}^{H} \boldsymbol{M}_i\hat{\boldsymbol{q}}_i^{t_d-1} \tag{4}$$

每个节点都维护一个$\hat{}\boldsymbol{q}^{(t_d-1)}$，用于保存之前的误差。随后使用共识平均来求上述方程中的最后一项求和。由于输出$\hat{}\boldsymbol{q}^{(t)}$ 需要共识平均和幂方法步骤，因此每个节点的局部估计都会受到这两种算法误差的影响。如果我们执行足够多的共识迭代 t_c 和幂迭代 t_p，每个节点对主特征向量的估计都将收敛到残差 $\boldsymbol{M}$ [20]的真实值。为了计算局部字典的估计，其表示为$\hat{}\boldsymbol{D}_i^{t_d}\forall i = 1, 2, \ldots, H$，并在迭代 t_d时形成全局字典估计$\hat{}\boldsymbol{D}^{t_d}$，我们通过协作求出需要最小化的总误差，在每个节点更新原子$\boldsymbol{d}_{i,n}$。这要求我们求出局部误差矩阵，记作$\hat{}\boldsymbol{E}^{(t_d)}{}_{i,n,R}$。为此，我们需要找出每个节点字典所使用的索引，记作 $\boldsymbol{\omega}_n$。在此情况下，$\boldsymbol{\omega}_i^{t_d}{}_{,n}$ 表示迭代 t_d时字典原子$\boldsymbol{D}_{i,n}$所使用的索引：

$$\boldsymbol{\omega}_{i,n}^{(t_d)} = \left\{ q \parallel 1 \leq q \leq Q_i, \boldsymbol{x}_{i,n,T}^{(t_d)}(q) \neq 0 \right\} \forall i \tag{5}$$

其中，$\boldsymbol{x}^{(t_d)}{}_{i,\ n,\ T}(q)$ 表示 $\boldsymbol{x}^{(t_d)}{}_{i,\ n,\ T}$ 的第 q^{th} 个元素。

我们将 $\Omega^{(t_d)}{}_{i,n} = Q \times \boldsymbol{\omega}_i^{(t_d)}{}_{,n}$ 定义为一个二进制矩阵，其在 $(\boldsymbol{\omega}^{(t_d)}{}_{i,n}\ (q)\ q)$ 处的元素为 1，用于存储 $\boldsymbol{\omega}^{(t_d)}{}_{i,n}$。因此，$\hat{}\boldsymbol{E}^{(t_d)}{}_{i,n,R} = \hat{}\boldsymbol{E}_{i\,n}^{(t)}{}, \Omega^{(t_d)}{}_{i,n}$。$\hat{}\boldsymbol{E}^{(t_d)}{}_{i,n}$ 由以下公式给出：

$$\hat{\boldsymbol{E}}_{i,n}^{(t_d)} = \boldsymbol{Y}_i - \sum_{j=1}^{n-1} \hat{\boldsymbol{d}}_{i,j}^{(t_d)}\hat{\boldsymbol{x}}_{i,j,T}^{t_d} - \sum_{j=n+1}^{N} \hat{\boldsymbol{d}}_{i,j}^{(t_d-1)}\hat{\boldsymbol{x}}_{i,j,T}^{t_d-1} \tag{6}$$

其中，j 是 1，2、.....、N 对应的字典原子，但不包括 $j = i$。

这里我们按数值顺序更新原子。已更新的原子属于迭代 t_d，未更新的原子属于 $t_d - 1$。下一步是像 K-SVD 中那样，对所有节点的 $\hat{}\boldsymbol{E}^{(t_d)}{}_{n,R} = \sum_{i=1}^{N} \hat{}\boldsymbol{E}^{(t_d)}{}_{i,n,R}$ 计算奇异值分解，以求得 $\boldsymbol{U}_{1,n}$ 和 $\boldsymbol{V}_{1,n}$。我们进行如下更新：$\left(\hat{}\boldsymbol{d}_{i,n},\ \hat{}\boldsymbol{x}_{i,n,R}\right) = (\boldsymbol{U}_{1,n}, \boldsymbol{\Delta}_{(1,1)}\boldsymbol{V}_{1,n})\forall i$根据定义，我们知道：

$$\hat{\boldsymbol{x}}_{i,n,R} = \hat{\boldsymbol{d}}_{i,n}\hat{\boldsymbol{E}}_{i,n,R}^{(t_d)} = \boldsymbol{U}_{1,n}\hat{\boldsymbol{E}}_{i,n,R}^{(t_d)} = \boldsymbol{\Delta}_{(1,1)}\boldsymbol{V}_{1,n}\forall i \tag{7}$$

为了求得$\hat{}\boldsymbol{E}^{(t_d)}{}_{i,n,R}$的近似值，我们使用分布式幂法求其主特征向量，即$\hat{}\boldsymbol{E}^{(t_d)}{}_{n,R}$ 必须是方阵

且为半正定矩阵。这可以通过将矩阵与其转置相乘$\hat{\boldsymbol{E}}^{(t_d)}_{n,R}\hat{\boldsymbol{E}}^{(t_d)T}_{n,R}$来实现，然后对所得矩阵应用幂方法，定义如下：

$$\hat{\boldsymbol{M}}^{(t_d)} = \sum_{n=1}^{H} \hat{\boldsymbol{M}}_n^{(t_d)} = \sum_{n=1}^{H} \hat{\boldsymbol{E}}_{n,R}^{(t_d)} \hat{\boldsymbol{E}}_{n,R}^{(t_d)^T} \quad (8)$$

然后，我们求出$\hat{\boldsymbol{M}}^{(t_d)}_n$的主特征向量。根据定义，它是$\hat{\boldsymbol{E}}^{(t_d)}_{n,R}$的$U_1$，因为对于任意实矩阵$\boldsymbol{A}$，$\boldsymbol{A}$的左奇异向量都是$\boldsymbol{A}\boldsymbol{A}^T$的特征向量。由此可知，$\hat{\boldsymbol{d}}_{i,n}$是我们分布式幂方法的结果。我们在本地计算$\hat{\boldsymbol{d}}^{(t_d)}_{i\,n}$、$\hat{\boldsymbol{E}}^{(t_d)}_{i,n,R}$，以设置$\hat{\boldsymbol{x}}_{i,n,R}$，最后设置$\hat{\boldsymbol{x}}_{i,n} = \hat{\boldsymbol{x}}_{i,n,R}\Omega_{i\,n}^{(t_d\ T)}$），将零置入系数所在的行。至此，一个原子的字典更新完成；当所有原子都更新完毕后，字典更新步骤也就完成了。每次字典学习迭代（t_d）都会重复此循环。

3 节点网络设置

我们搭建了一个由四个 4 Rasp- berry Pi 2 Model B 节点组成的 Kubernetes 网络，用于开展实际实验。在 Kubernetes 中，为 P 工作器 Pod 配置了部署控制器。每个 Pod 都可以作为一个独立自主的单元接收、处理并发送数据。我们手动调整部署中的 Pod 数量，以展示随着更多 Pod 加入工作池、输入数据进一步分散时，Cloud K-SVD 算法的效率。所有 Pod 共用相同的网络介质，因此结果中包含了潜在的网络延迟。

节点 P_i 接收 $\boldsymbol{Y}_i \in \mathbb{R}^{M\times Q_i}$ 数据信号，并通过先计算信号向量来协同进行字典学习 $\boldsymbol{X}_i$，再通过共识迭代估算残差误差。我们采用共识平均来降低残差误差，并利用局部字典恰当地估算信号向量$\boldsymbol{D}_i$，以适配节点 i 中的数据。所有实验均使用一个预处理器和一个后处理器 Pod（见图 2）。这些 Pod 在实验中并未启用，仅用于拆分和汇总数据。

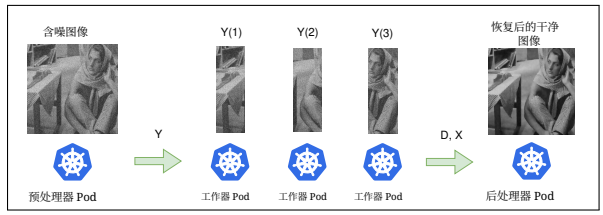

图 2 Kubernetes 编排：例如，运行 Cloud K-SVD 的三个工作 Pod 副本。预处理器将图像数据分成大小相同的三部分。工作 Pod 分别处理图像中对应的部分，后处理器再将其合并。

图3 展示了预处理器、工作节点和后处理器 Pod 之间交互的简要时序图。图中展示了我们方案中的所有步骤实现及 Cloud K-SVD 协议的概况。图示经过了简化，因此数据是由预处理器直接发送给各个工作节点的，而不是看起来那样在它们之间依次传递。最后保存数据时也是如此。

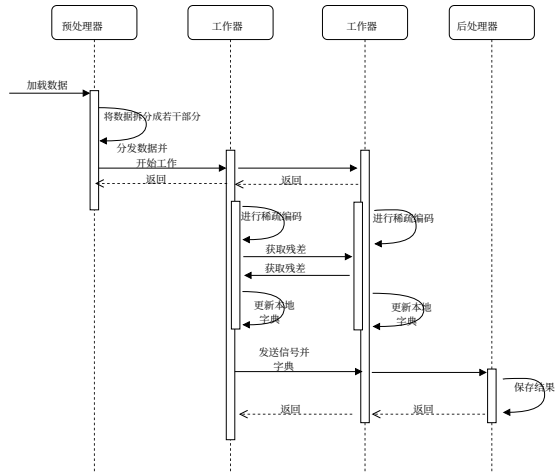

图3 在此示例中，数据被加载到两个工作器 Pod 中。随后启动 Cloud K-SVD，最终生成一个信号矩阵和一个局部字典，由后处理器 Pod 收集。

4 实验与结果

我们的实验旨在证明 Cloud K-SVD 能够从完全未经探索的干净和含噪图像块中学习新字典，并将这些图像块恢复为稀疏近似。恢复精度是选定的性能指标，它是迭代次数的函数，

t_d，信号长度（图像块作为列向量），M，信号数量（图像块），Q，以及字典中的原子数量 N。我们考虑这样一种情况：图像块受到自然图像中常见的加性高斯白噪声污染。

4.1 K-SVD 的变体

我们考虑 K-SVD 的两种变体：

• 局部 K-SVD：节点 P_i 接收 $\boldsymbol{Y}_i \in \mathbb{R}^{M\times Q_i}$ 个数据信号，并在本地执行字典学习任务，对等节点之间不进行任何协作共识迭代。该模型分发数据信号，但由于没有共识迭代，节点 P_i 不会了解节点 P_j 的数据。该模型模拟了传统的 K-SVD 方案，但将数据分发到多个工作 Pod。

• Cloud K-SVD：节点 P_i 接收 $\boldsymbol{Y}_i \in \mathbb{R}^{M\times Q_i}$ 个数据信号，并通过协作进行字典学习：首先经由稀疏近似计算信号向量 $\boldsymbol{X}_i$，然后通过共识迭代估计残差误差。在此过程中，所有工作 Pod P 通过平均或校正共识协同工作，以降低残差误差，并利用适配节点 j 数据的局部字典 $\boldsymbol{D}_i$ 正确估计信号向量。

4.2 参数设置

训练数据的完整尺寸为 $M \times Q$，其中M 是 $\boldsymbol{Y} \in \mathbb{R}^{M\times Q}$中数据信号的长度，$Q$ 是$\boldsymbol{Y}$ 中信号列向量的数量，且 $\boldsymbol{X}$。我们将使用图像块数据（列）来训练分布式字典；因此，输入块的尺寸为 M。由此得到的数据向量汇集在单个矩阵中 $\boldsymbol{Y} \in \mathbb{R}^{M\times Q}$。随后，我们在预处理器 Pod 上向 $\boldsymbol{Y}$ 中的所有训练数据添加加性高斯白噪声。接着，将数据划分为大小相等的部分，并分发给 P 个工作器 Pod（见图2）。工作器 Pod 现在负责进行稀疏近似，以减少ˆ所得 $\boldsymbol{Y}$ 数据矩阵中的加性高斯白噪声。每次实验都会生成一个独特的随机字典（元素独立同分布且服从均匀分布，ℓ_2 列归一化至范围 $[0;\ 1]$）$\boldsymbol{D}^{M\times N}$，并将其分发给每个工作器 Pod。随后，网络中的所有工作器 Pod 都负责各自的本地字典，并在训练过程开始时对其进行演化开始。我们使用自行实现的同步正交匹配追踪（SOMP）算法 [20]，并将权重标量 w 的取值设在范围 $[0;\ 1]$内。以下参数设置适用于所有实验：

• Cloud K-SVD 执行三种类型的嵌套迭代：

– t_d：通过 SOMP 计算稀疏信号向量，并在字典更新步骤中更新相应原子的云端迭代次数。**设置：** $t_d = 10$。– t_p：针对每个原子 N 执行的幂迭代次数。该次数需要在执行时间与共识程度之间进行权衡。**设置：** $t_p = 3$。– t_c：通过平均共识从相邻 Pod 获取残差误差的共识迭代次数。该次数需要在执行时间与共识程度之间进行权衡。**设置：** $t_c = 5$。

• N：字典 $\boldsymbol{D} \in \mathbb{R}^{M\times N}$ 中原子列向量的数量，以及信号行向量 $\boldsymbol{X} \in \mathbb{R}^{N\times Q}$ 的长度。由于实验中评估的是过完备字典，因此 $N \gg M$。Pod 上的初始字典使用随机生成的数据进行初始化，包含固定数量的原子 N，维度为 M。**设置：** $N = 50$，100，200。

• K：$\boldsymbol{X}$的信号向量中非零系数的数量。**设置：** $K = 3$，5、7。

• Pod 权重 w：一个标量，乘以共识步骤中残差向量 q_i 和 q_j 之间的差值。它定义了在范围 $[0, 1]$内，节点 i 对向量 q_j 赋予多大权重。**设置：** $w = 1$。

• P：在单次实验中运行 Cloud K-SVD 的 Kubernetes 工作器 Pod 副本数量。**设置：** $P = 4$。

• PS：图像块大小是从训练图像中提取的图像块的分辨率；我们要求$N \gg M$。**设置：**

– 图像块 5×5： $M = 25$ 和 $N = 50$– 图像块 6×6： $M = 36$ 和 $N = 100$– 图像块 7×7： $M = 49$ 和 $N = 100$– 图像块 8×8： $M = 64$ 和 $N = 150$– 图像块 9×9： $M = 81$ 和 $N = 200$

4.3 实验：字典共识

这项使用合成数据的实验旨在展示 Cloud K-SVD 中的共识元素。图 4 展示了字典 $\boldsymbol{D}_i \ \forall i = 1$、2、.....之间的平均均方误差随 t_d 次迭代的变化情况。这里，我们在所有实验中固定参数 t_c，并观察增大 t_p时误差值的变化。我们发现，无论 $t_c = 1$ 的数量是多少，都无法收敛；不过，当 t_c 较高时，字典间的误差会共同趋近于零（对数尺度），从而呈现出清晰的变化趋势。

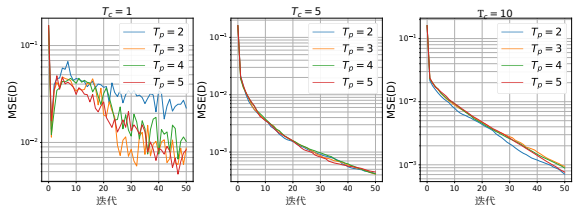

图 4 在不同幂迭代次数、 t_p 和共识迭代次数 t_c 下，Pod 字典的平均均方误差。我们设置云端迭代次数 $t_d = 0$，1，.....，50、原子数 $N = 50$、信号长度 $M = 20$、信号数 $Q = 2000$、非零元素数 $K = 3$ 和 Pod 数 $P = 4$。

表1 列出了使用合成数据并针对协作式 K-SVD 步骤进行跟踪的 $P = 4$ 个节点的平均执行时间。其目的是展示共识迭代和幂迭代如何影响算法平均完成 K-SVD 步骤所需的时间，单位为秒。此处不包括信号近似步骤，因为它不受共识迭代次数或幂迭代次数影响，因此在这里的所有实验中都保持不变。不过，趋势很明显：每次幂迭代执行的共识迭代越多，算法完成 K-SVD 步骤所需的时间就越长。

表 1 在不同 t_p 和 t_c 下，一次 K-SVD 迭代的平均执行时间。 $t_d = 50$, $M = 20$, $N = 50$ $K = 3$ $P = 4$, $Q = 2000$, ,.

t_c	$t_p = 2$	$t_p = 3$	$t_p = 4$	$t_p = 5$
1	4.5s	6.6s	8.3s	9.8s
2	18.4秒	26.9秒	35.2秒	43.1秒
3	34.5秒	51.2秒	67.9秒	84.2秒

4.4 实验：图像块学习

第二组实验评估了 Cloud K-SVD 在 cleanimages 上的表现。该数据集包含八幅 8 位灰度图像，这些图像根据其维度和轮廓的多样性进行选择。表2 展示了原始分辨率、按 $\alpha = 2$倍缩小后的分辨率、图像均值和方差等属性。

表2 用于图像块学习的测试图像及其相应属性。按抽取因子α缩小会生成大小为 $\alpha \times \alpha$ 的图像块，并将每个图像块中的元素缩减为局部均值。

Name	原始图像 res	分辨率/$\alpha = 2$	均值/方差
Castle	304×200	152×100	115/11.18
Lenna	480×512	240×256	124/8.96
中国	427×640	214×320	114/26.95
Flower	427×640	214×320	68 /9.97
切尔西	300×451	150×226	117/4.05
相机	512×512	256×256	118/15.07
宇航员	512×512	256×256	112/22.27
Face	150×150	75×75	139/15.52

为了提高性能，我们在处理前对部分图像采用范围为 $[1-5]$ 的抽取因子 α。 $\alpha = 1$ 表示不进行缩小，而 $\alpha = 2$ 表示与原始图像相比，像素总数在列和行方向上均减半。对于图像块学习，我们引入变量 PS，表示提取的图像块的分辨率，例如 5×5 或 8×8。对于这两种情况，都会从每张训练图像中提取所有可能的重叠图像块，大小为 PS；例如，图像块大小为 $PS = 8 \times 8$、分辨率为 256×256 且 $M = 64$ 时，总共提取 $Q = (256-8+1)^2 = 62001$ 个图像块。于是，图像块大小为 PS 时，一个输入数据样本的维度为 M（垂直堆叠的列），训练数据总量为 $T_t = M \times Q$。随后，将所得数据向量汇集到单个矩阵 $Y \in \mathbb{R}^{M \times Q}$ 中，再拆分成大小相等的部分，分发给 P 个 Pod 进行学习。

对于每个实验，都会为每个节点生成一个大小为 $\boldsymbol{D}_i$ 的初始字典，其中的条目为随机独立同分布均匀分布的值，并且其归一化的 ℓ_2 列位于范围 $[0,1]$ 内。每个字典都会在开始前分发到节点 P_i。随后，所有工作 Pod 负责各自的局部字典。此外，我们将权重标量 w 的值设定在范围 $[0,1]$ 内。

- 图像块大小为 PS $= 5 \times 5$、6×6、7×7、8×8、9×9。
- 数据维度 (M) $= 25$，36、49、64、81。
- 原子数 (N) $= 50$，100、150、200。
- 稀疏度（K）$= 3$，5，7。
- 迭代次数（t_d）$= 10$。
- 节点数量（P）$= 4$。
- 幂迭代次数（t_p）$= 3$
- 共识迭代次数（t_c）$= 5$。

图 5 显示了字典之间的平均均方误差 $\boldsymbol{D}_i\ \forall i = 1,\ 2,\ \ldots\ldots,\ P = 4$ 随 t_d 迭代次数的变化，其中共识功能启用或禁用。可以看出，与使用分布式数据的常规 K-SVD（局部 K-SVD 变体）相比，启用共识功能时（Cloud K-SVD 变体），字典之间的均方误差明显下降。换句话说，Pod 通过这种方式学习彼此的数据，从而生成更为共享的字典，可用于更好地表示 $\boldsymbol{Y}$，而不仅仅是 $\boldsymbol{Y}_i$。

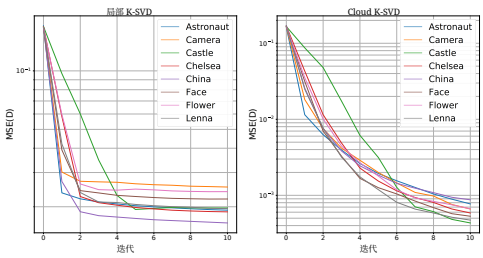

图 5 不同无噪图像下各节点字典之间的平均均方误差。$t_d = 0,\ 1,\ \ldots\ldots,\ 10,\ t_p = 3$ $t_c = 5\ \ M = 25\ \ N = 50\ \ K = 3\ \ P = 4$ ，，，，。

图6 和7 展示了在迭代ˆ过程中，使用 Cloud K-SVD 或局部 K-SVD 将恢复的 $\boldsymbol{Y}$ 与 $\boldsymbol{Y}$ 进行比较时的均方误差/结构相似性得分 t_d。我们观察到ˆ这两种变体的表现相似，这主要是因为 $\boldsymbol{Y}_i$ 由相应的字典 $\boldsymbol{D}_i$ 和信号 $\boldsymbol{X}_i$ 构成。中国在均方误差/结构相似性方面表现较差，而我们能更好地恢复 *Flower*。算法似乎偏好包含大量重复轮廓、特定结构较少的图像，但对于包含许多细小且独特细节的图像，性能较差。

图8 展示了将非零元素的数量设为 ℓ_2 时的均方误差/ $K = 3$、5、7 时，均方误差/ t_d -范数/峰值信噪比/结构相似性得分随迭代次数的变化。我们

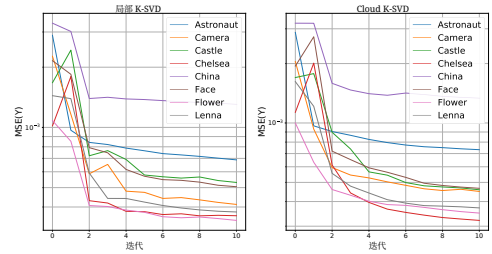

图 6 不同无噪图像的原始信号 $\boldsymbol{Y}$ 与恢复信号ˆ $\boldsymbol{Y}$ t_d =之间的均方误差。，，，，，，，，，，0 1 ... 10 $t_p = 3$ $t_c = 5$ $M = 25$ $N = 50$ $K = 3$ $P = 4$

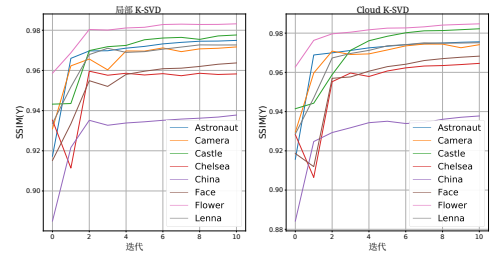

图 7 不同无噪图像的原始信号 $\boldsymbol{Y}$ 与恢复信号ˆ $\boldsymbol{Y}$ t_d =之间的结构相似性。,,,,0 1 ... 10 $t_p = 3$ $t_c = 5$ $M = 25$ $N = 50$ $K = 3$ $P = 4$ ，，，，，，。

注意到，较高的 K 会带来较低的均方误差/ℓ_2-范数，以及较高的峰值信噪比/结构相似性得分。

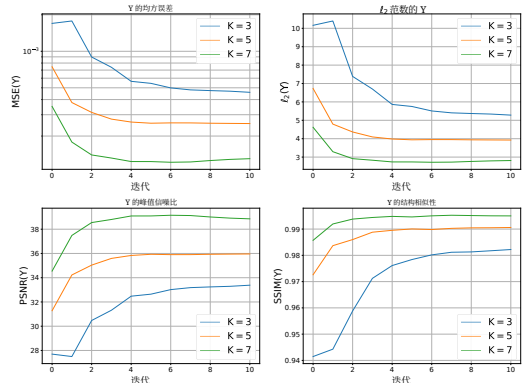

图 8 不同 K下 Castle 图像的原始信号 $\boldsymbol{Y}$与恢复信号 $\boldsymbol{Y}$之间的均方误差、ℓ_2 范数误差、峰值信噪比和结构相似性ˆ。$t_d = 0,\ 1,\ \ldots,\ 10,\ t_p = 3,\ t_c = 5\ \ M = 25\ \ N = 50\ \ P = 4$ ，，，。

图9 展示了均方误差/ℓ_2-范数/峰值信噪比/结构相似性分数随迭代次数的变化 t_d，此时图像块大小增大。由于 PS 是以像素数表示的数据维度 M，并且 $N \gg$

M，我们也相应增大 N。可以看到，当图像块大小较小时，例如 $PS = 5 \times 5/M = 25$，与较大的图像块大小（例如 $M = 49$）相比，MSE^和 ℓ_2-范数在 $\boldsymbol{Y}$ 与 $\boldsymbol{Y}$ 之间更低。随着图像块大小增大，峰值信噪比和结构相似性也呈现类似的变化趋势。理想情况下，我们希望均方误差和 ℓ_2-范数较低，而峰值信噪比和结构相似性较高。

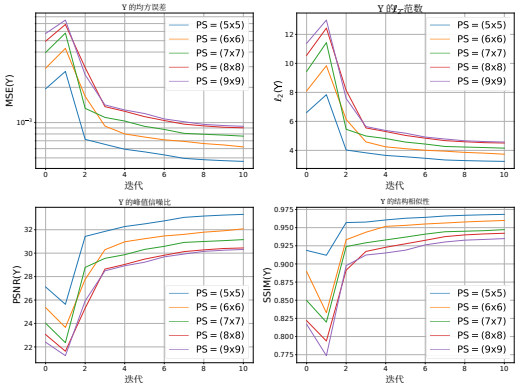

图 9 不同 M 和 N 下，$Face$ 图像的原始信号 $\boldsymbol{Y}$ 与恢复信号 $\boldsymbol{Y}$ 之间的均方误差、误差的 ℓ_2-范数、峰值信噪比和结构相似性^。 $t_d = 0$, 1,..., 10,$t_p = 3$ $t_c = 5$ $K = 3$ $P = 4$, , , 。

表3、4和5 展示了四个节点在正交匹配追踪和K-SVD步骤中的平均执行时间，对应的取值为 K = 3，5，7，M =25，36，49，64，81 和 α=1，分别对应局部 K-SVD 和 Cloud K-SVD 运行中的 2。正如预期，正交匹配追踪步骤似乎会受到较大 K的显著影响，而当 K 设定较高时，K-SVD步骤的表现尚可。使用较大的图像块维数 M 不会使正交匹配追踪变慢，但会使 K-SVD变慢。这是因为通常 $N \gg M$的大小约为其两倍，因此字典大小 N 会随着 M 的增大而增大。由于 K-SVD 会逐个原子更新字典，因此更新大型字典自然比更新小型字典耗时更长。

4.5 实验：去噪

现在，我们向所有训练图像中添加均值为 μ = 0 、方差为 σ^2 =0.001、0.005和0.01 的高斯分布噪声（加性高斯白噪声），然后提取 Q 含噪图像块。图10、11 和12展示了人脸图像的干净版本和含噪版本。

图13 展示了^$\boldsymbol{Y}$ 的均方误差/峰值信噪比/结构相似性得分随 t_d 迭代次数变化的情况，并与

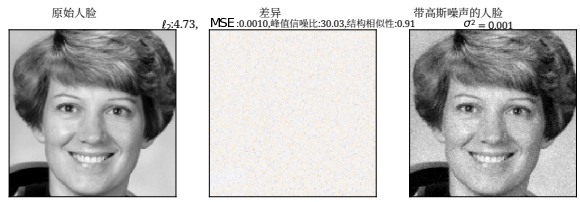

图 10 左：原始图像。中：差异。右：噪声图像（$\mu = 0$, $\sigma^2 = 0.001$）。

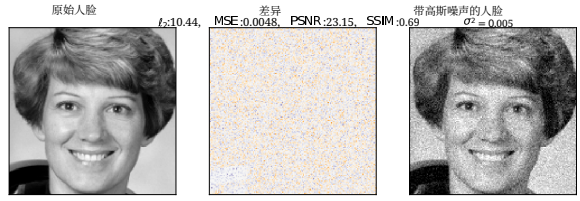

图 11 左：原始图像。中：差异。右：噪声图像（$\mu = 0$, $\sigma^2 = 0.005$）。

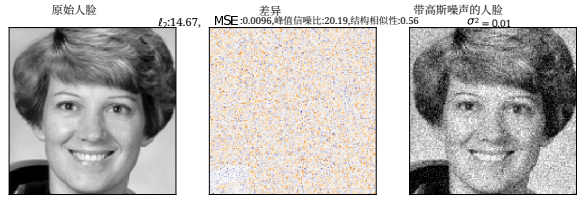

图 12 左：原始图像。中：差异。右：噪声图像（$\mu = 0$, $\sigma^2 = 0.01$）。

原始无噪声图像 $\boldsymbol{Y}$是 Cloud K-SVD 从含噪图像块中恢复人脸图像后的结果。图14展示了恢复图像的拼贴图。这里，方差设为 σ^2 = 0.01 ，并采用了 PS、 K和 N的各种配置。我们发现，在大多数情况下，大约到第四次迭代时效果就达到持平点；有些配置甚至随着迭代次数增加而效果变差。由于加性高斯白噪声的强度相当高，经过一定次数的迭代后，算法开始恢复噪声，因为它只接触过这些含噪测量值。与 [14], 进行比较时，按照 $\sigma^2 = 0.01$的噪声方差，使用 [14]的记法相当于 $\sigma = 25.5$，因为 $\sqrt{0.01 * 255} = 25.5$。我们在人脸图像上取得的最佳峰值信噪比/结构相似性约为 27.98/88.05，使用的是 $PS = 9 \times 9$、$K = 3$、$N = 200$ 和 $\sigma = 25.5$（按 [14]的记法）。 [14] 在对山景图像去噪时，针对 $\sigma = 20$取得的峰值信噪比/结构相似性为 29.21/85.64，针对 $\sigma = 30$ 取得的分数为 27.67/79.93 [21]。因此，我们取得了与 [14], 相当的峰值信噪比/结构相似性，不过请注意，

表 3 使用无噪声图像块时，局部 K-SVD（$t_p = 0, t_c = 0$）和 Cloud K-SVD（$t_p = 3, t_c = 5$）的正交匹配追踪和 K-SVD步骤的平均执行时间。 $K = 3$、 $M = 25$、 $N = 50$。 *Castle*和 *Face*具有 $\alpha = 1$，其余具有 $\alpha = 2$

变体/步骤	宇航员	相机	Castle	切尔西	中国	Face	Flower	Lenna
局部/正交匹配追踪	28s	29.4秒	27.3秒	15.1秒	30.6秒	9.8s	30.7秒	27.6秒
局部/K-SVD	7.5s	7.1s	6.7s	3.3s	8s	2.5s	7.6s	6.8s
云端/OMP	34.2秒	32.6秒	31.6秒	18.5秒	34.2秒	12.5秒	35.6秒	31.2秒
Cloud/K-SVD	42.3秒	43.1秒	43.5秒	33.9秒	44.9秒	31.9秒	46.4秒	42.8秒

表 4 使用不同的 K 值时，局部 K-SVD（$t_p = 0$， $t_c = 0$）和云K-SVD变体（$t_p = 3$， $t_c = 5$）中正交匹配追踪和 K-SVD步骤的平均执行时间。 *Castle* $M = 25$、 $N = 50$、 $\alpha = 1$。

变体/步骤	Castle $K = 3$	Castle $K = 5$	Castle $K = 7$
局部/正交匹配追踪	27.3 秒	45.2 秒	64.4 秒
局部/K-SVD	6.7s	8.7s	11.1 秒
云端/OMP	31.6秒	51.9秒	74.2秒
Cloud/K-SVD	43.5秒	45.4秒	43.1秒

图像块大小不同（我们使用 $PS = 9 \times 9$，他们使用 $PS = 3 \times 3$）。

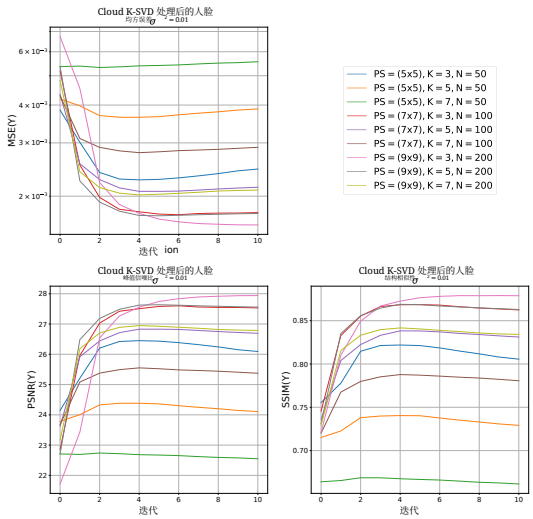

图13 在不同的 M N K $t_d = 0\ 1 \ldots 10$ $t_p = 3$ $t_c = 5$ $P = 4$ 下，含有加性高斯白噪声（$\sigma^2 = 0.01$）的 *Face* 图像的原始 ˆ 信号 $\boldsymbol{Y}$ 与恢复信号 $\boldsymbol{Y}$之间的均方误差、峰值信噪比和结构相似性，以及 . ,,,、 , , , 。

图15 展示了 $\sigma^2 = 0.005$的均方误差/峰值信噪比/结构相似性得分。显然，较小的图像块大小和较高的稀疏度（绿色线条）难以降低加性高斯白噪声，而较低稀疏度的配置效果更好。根据结果，较为合适的配置是图像块大小为 7×7 因此

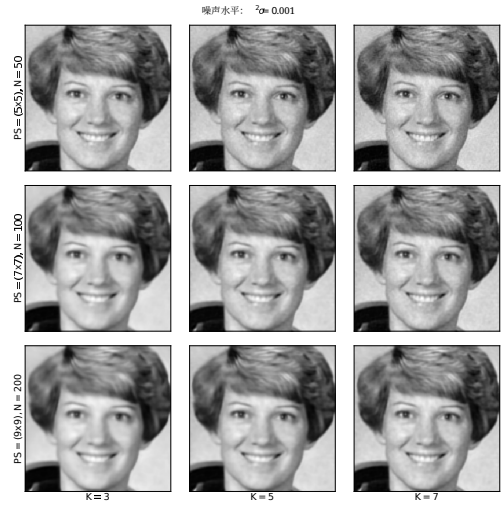

图14 使用 Cloud K-SVD 在不同的 M、 N 和 K 条件下恢复的 *Face* 图像拼贴图。 $\sigma^2 = 0.001$、 $t_d = 10$、 $t_p = 3$ $t_c = 5$ $P = 4$, , 。

$M = 49$，原子数量为 100，稀疏度为 $K = 3$。图16 是一张恢复图像的拼贴图。作为比较， $\sigma^2 = 0.005$ 等于 $\sigma = 18$在 [14]的记法中的数值。我们获得的最佳峰值信噪比/结构相似性得分约为**29.50/91.05** ，对应 $PS = 9x9$、 $K = 5$、 $N = 200$和 $\sigma = 18$。 [14]得到**29.21/85.64** ，对应 $\sigma = 20$。同样，我们所用的参考图像和图像块大小不同（人脸与*Hill* [21], $PS = 9 \times 9$与 3×3）。

图17 展示了 $\sigma^2 = 0.001$下的均方误差/峰值信噪比/结构相似性得分。较大的像素尺寸 PS 搭配较低的稀疏度 K，在加性高斯白噪声方差较低时已不再是理想设置。当 K 较低而 PS较高时（粉色线），Cloud K-SVD 难以恰当地恢复图像。若输入信号中几乎没有噪声或完全没有噪声，Cloud K-SVD 在高稀疏度下表现更好。图 18 展示了恢复图像的拼贴图。作为比较， σ^2 =0。001 等于 $\sigma = 8$ 在 [14]的记法中的值。我们得到的最佳峰值信噪比/结构相似性得分约为 **32.90/95.10**，对应 $PS = 5x5$、 $K = 7$、 $N = 50$

表5 使用具有不同 M 和 N 值的 $Face$ 时，局部 K-SVD（$t_p = 0$，$t_c = 0$）和 Cloud K-SVD（$t_p = 3$，$t_c = 5$）变体的正交匹配追踪和 K-SVD步骤的平均执行时间。 $K = 3$。 $\alpha = 1$。

变体/迭代	**Face** $M = 25$ $N = 50$	**Face** $M = 36$ $N = 100$	**Face** $M = 49$ $N = 100$	**Face** $M = 64$ $N = 150$	**Face** $M = 81$ $N = 200$
局部/正交匹配追踪	9.8s	10s	10.1秒	10.6秒	10.9秒
局部/K-SVD	2.5s	4.8s	6.6s	12.9秒	17.7秒
云端/OMP	12.5秒	14.3秒	13.3秒	16.3秒	15.9秒
Cloud/K-SVD	31.9秒	63.4秒	71.5秒	120.5秒	157.9秒

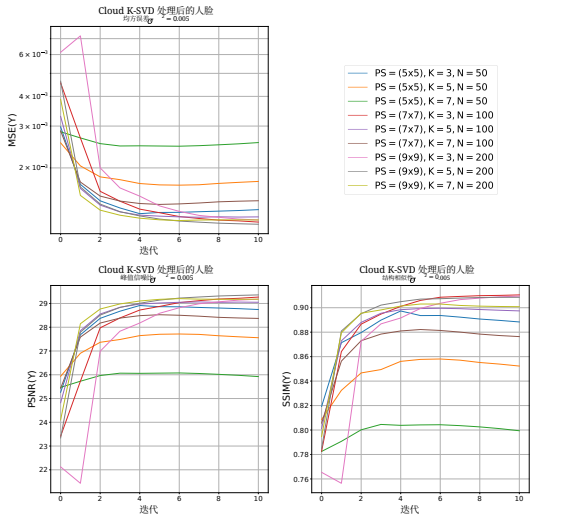

图 15 在不同的 M N K $t_d =$ 下，$0\ 1\ \ldots\ 10$ $t_p = 3$ $t_c = 5$ $P = 4$ 带有加性高斯白噪声（$\sigma^2 = 0.005$）的 $Face$ 图像的原始^信号 $\boldsymbol{Y}$ 与恢复信号 $\boldsymbol{Y}$ 之间的均方误差、峰值信噪比和结构相似性，以及。、、、、、、。

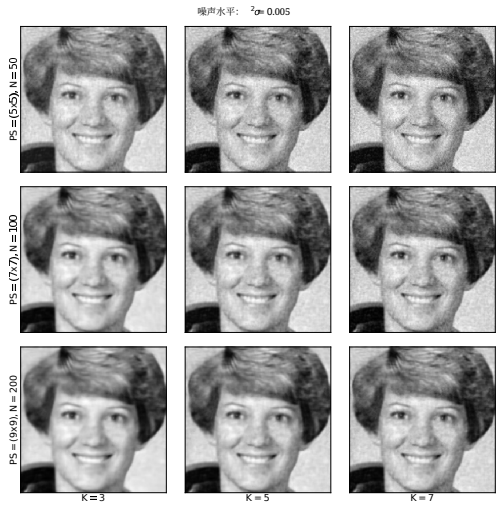

图 16 使用 Cloud K-SVD 在不同的 M、 N 和 K 下恢复的 $Face$ 图像拼贴图。 $\sigma^2 = 0.005$、 $t_d = 10$、 $t_p = 3$ $t_c = 5$ $P = 4$ 、 、 。

以及 $\sigma = 8$。 [14] 得到 **32.15/94.01** ，对应 $\sigma = 10$。同样，我们使用的参考图像和图像块大小不同（人脸 与 $Hill$ [21]， $PS = 5 \times 5$与 3×3）。

表 6 展示了针对正交匹配追踪和 K-SVD步骤所跟踪的四个节点的平均执行时间。

5 结论

为了成功执行加性高斯白噪声去噪，需要相应地调整 Cloud K-SVD 的参数。噪声严重时，必须去除大量细节；噪声较轻时，则只需去除较少细节。减小数据大小和原子数量， M 和 N，并增加非零元素的数量， K，可以更好地恢复图像中的细节。Cloud K-SVD 最初是为图像分类而设计的，但我们对其进行了创新应用，即该算法，并进行了一些实际改进，例如采用同步正交匹配追踪（SOMP），并调整算法以应对节点故障。我们证明，改进后的 Cloud K-SVD 能够成功地对图像进行分布式加性高斯白噪声去噪，同时学习全局字典。我们将结果与 [14]，中提出的剪枝非局部均值（PNLM）算法进行了比较，结果支持我们的观点：Cloud K-SVD 的恢复精度可与该领域当前最先进水平相媲美。

利益冲突

作者声明，他们不存在已知的竞争性财务利益或个人关系，这些利益或关系可能会影响本文所报告的工作。

表6 在不同稀疏度 K、数据大小 M 和加性高斯白噪声等级 σ^2下，使用 Cloud K-SVD（$t_p = 3$，$t_c = 5$）从含噪图像块中恢复 *Face* 时，正交匹配追踪和 K-SVD步骤的平均执行时间。我们设置 $\alpha = 1$。

步骤/σ^2	**Face** $K = 3$ $M = 25$	**Face** $K = 3$ $M = 49$	**Face** $K = 3$ $M = 81$	**Face** $K = 5$ $M = 25$	**Face** $K = 5$ $M = 49$	**Face** $K = 5$ $M = 81$	**Face** $K = 7$ $M = 25$	**Face** $K = 7$ $M = 49$	**Face** $K = 7$ $M = 81$
正交匹配追踪/0.001	11.9秒	12.4	14.1	20s	20.1秒	22s	28s	28.3秒	33.2秒
正交匹配追踪/0。005	12.5	12.8秒	14.9秒	19s	21.5秒	23.7秒	27.4秒	29.8秒	31.7秒
正交匹配追踪/0。01	12.3	12.2秒	14.7秒	20.5秒	22.4秒	22.2秒	28.5秒	28.8秒	32.5秒
K-SVD/0。001	31.5秒	72.1	178.1秒	33s	69.5秒	178.9秒	33.7秒	70.3秒	179.6秒
K-SVD/0。005	32.1秒	68.7秒	179s	32.5秒	67.8秒	175.9秒	33.3秒	67.5秒	178.5秒
K-SVD/0。01	31.8秒	70s	160.3秒	32.7秒	70.9秒	159.5秒	33.3秒	73.9秒	158.4秒

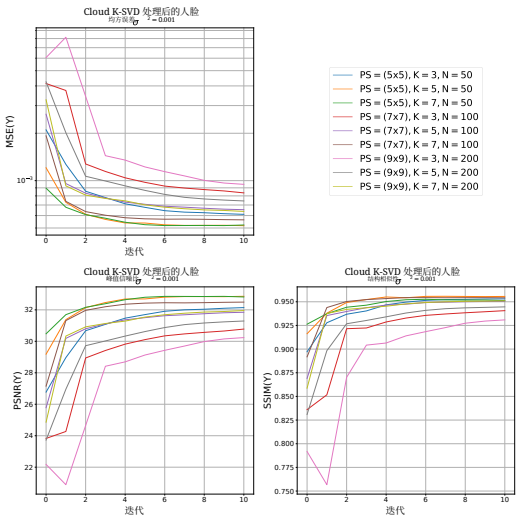

图17 原始ˆ信号 $\boldsymbol{Y}$与在不同

M N K $t_d =$ 下带有加性高斯白

0 1 ... 10 $t_p = 3$ $t_c = 5$ $P = 4$
噪声（$\sigma^2 = 0.001$）的 *Face*图像的恢复信号 $\boldsymbol{Y}$之间的均方误差、峰值信噪比和结构相似性，以及。,,,,,,.

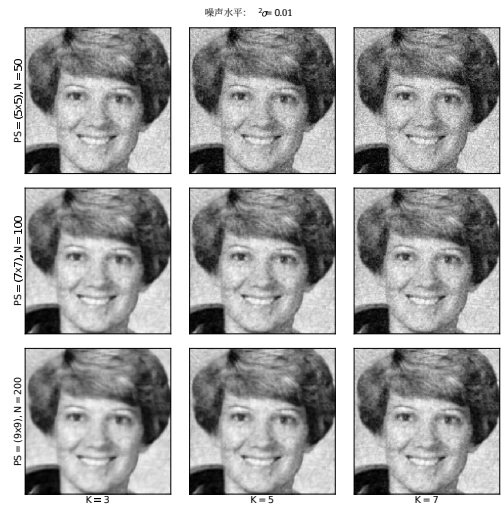

图18 使用 Cloud K-SVD 在不同 M、N和 K 下恢复的 *Face* 图像拼贴图。$\sigma^2 = 0.01$、$t_d = 10$、$t_p = 3$、
$t_c = 5$ $P = 4$ ，。

参考文献

[1]Taylor-Sakyi，K. 大数据：理解大数据。*CoRR* **abs/1601.04602**，8（2016）。网址 http://arxiv.org/abs/1601.04602。 https://arxiv.org/abs/1601.04602。

[2] Li, C.等。 面向数据中心网络社交大数据计算的分布式私有在线学习。 *2016 IEEE*国际通信会议，*ICC 2016*6（2016）。 https://doi.org/10.1109/ICC.2016.7510723, https://arxiv.org/abs/1602.06489 。

[3] Elad，M. 稀疏与冗余表示 第1版（ Springer-Verlag New York，2010年）。网址 https://www.springer.com/gp/book/9781441970107。

[4]Elad, M. 与 Aharon, M. 基于学习字典的稀疏冗余表示进行图像去噪。 *IEEE*图像处理汇刊 **15**（12），3736–3745（2011）。网址 https://ieeexplore.ieee.org/document/4011956 。

[5] Aharon, M.、Elad, M. 和 Bruckstein, A. K-SVD：一种用于设计过完备字典的稀疏表示算法 Michal。*IEEE* 信号处理汇刊**54**（11），4311–4322（2006年）。 https://doi.org/10.1109/TSP.2006.881199 。

[6] Mairal, J.、Bach, F. 和 Ponce, J. 任务驱动的字典学习。*IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**（4），791–804（2012）。https://doi.org/10.1109/TPAMI.2011.156，https://arxiv.org/abs/1009.5358 。

[7]Theodoridis, S. 机器学习：贝叶斯与优化视角（Elsevier Science and Technology，2015）。网址：https://www.sciencedirect.com/book/9780128188033/machine-learning。

[8]Mallat, S. 《信号处理中的小波之旅：稀疏方法》第3版（爱思唯尔公司，2009）。网址 https://www.sciencedirect.com/book/9780123743701/a-wavelet-tour-of-signal-processing。[9] Tosic, I. & Frossard, P. 字典学习。*IEEE*《信号处理杂志》**28**(2), 27–38（2011）。https://arxiv.org/abs/arXiv:1402.1515v1 。[10] Li, X., Shen, H., Member, S., Zhang, L. & Member, S. 利用多时相字典学习恢复受浓云和阴影污染的定量遥感产品。*IEEE*地球科学与遥感汇刊 **52** (11), 7086–7098（2014）。[11]Yan, X., Yang, B., Zhang, W., Liu, C. & Wang, Y. 一种基于KSVD的改进型羽毛和羽绒图像去噪算法。论文集——*2016*年第*8*届医学与教育信息技术国际会议（*ITME 2016*）419–423（2017）。https://doi.org/10.1109/ITME.2016.0100 。[12] Raja, H. & Bajwa, W. U. Cloud K-SVD：一种适用于大型分布式数据的协同字典学习算法。*IEEE*《信号处理汇刊》**64**(1), 173–188（2016）。https://doi.org/10.1109/TSP.2015.2472372 。[13] Clifton, C., Kantarcioglu, M., Vaidya, J., Lin, X. & Zhu, M. Y. 隐私保护分布式数据挖掘工具。*ACM SIGKDD*探索通讯**4** (2), 28–34（2002）。https://doi.org/10.1145/772862.772867 。

[14]Ghosh, S., Mandal, A. K. & Chaudhury, K. N. 剪枝非局部均值。*IET*图像处理**11**, 317–323（2017）。https://doi.org/10.1049/iet-ipr.2016.0331 。

[15]Lillelund, C. M.、Jensen, H. B. 和 Peder-sen, C. F. 源代码（2021年）。网址：https://github.com/thecml/cloud-ksvd。

[16] Mallat, S. G. 和 Zhang, Z. 基于时频字典的匹配追踪。*IEEE*信号处理汇刊 **41**（12），3397–3415（1993年）。https://doi.org/10.1109/78.258082 。

[17]Dumitrescu, Bogdan, Irofti, P. 字典学习算法与应用第1版（施普林格国际出版，2018年）。网址：https://www.springer.com/gp/book/9783319786735。

[18] Mairal, J., Bach, F., Ponce, J. & Sapiro, G. 矩阵分解与稀疏编码的在线学习。《机器学习研究杂志》**11**，19–60（2010年）。网址：https://arxiv.org/abs/0908.0050 。

[19]Chen, Y., Tron, R., Terzis, A. & Vidal, R. 非对称无线链路下的校正共识。*IEEE*决策与控制会议论文集6660–6665（2011）。https://doi.org/10.1109/CDC.2011.6161272 。

[20] Tropp,J. A., Gilbert, A. C. & Strauss, M. J. 同时稀疏近似算法。第一部分：贪心追踪。《信号处理》**86**(3), 572–588（2006年）。网址：http://dx.doi.org/10.1016/j.sigpro.2005.05.030。https://doi.org/10.1016/j.sigpro.2005.05.030 .

[21]BM3D图像数据库。网址 https://webpages.tuni.fi/foi/GCF-BM3D/。