UE COM

Learning Deep CNN Denoiser Prior for Image Restoration

Adrien Zabban

8 janvier 2024

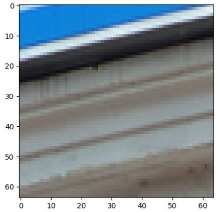
Le Problème inverse

But

On a une image observée dégradée y et l'on veut retrouver l'image d'origine x. On sait que cette image a été dégradée de la façon suivante :

$$y = Hx + v$$

où H est la matrice de dégradation que l'on connait, et v est un bruit gaussien d'écart-type σ inconnu.



$$\xrightarrow{f_{H,\sigma}}$$

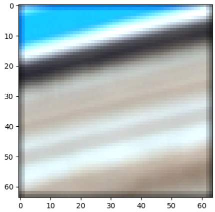


Figure: image d'origine x (à gauche) et l'image dégradée y (à droite).

Maximiser la log-likelihood

$$
\begin{aligned}
\max_x \log(p(x|y)) &= \max_x \log(p(x,y)) \quad \text{car } p(x|y) = p(x,y) \times p(y) \\
&= \max_x \log(p(y|x)) + \log(p(x)) \\
&\quad \text{or } (y|x) = (v + Hx|x) \sim \mathcal{N}(Hx, \sigma^2) \\
&= \max_x -\frac{\|y - Hx\|^2}{2\sigma^2} + \log(p(x)) \\
&= \min_x \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(x) \quad \text{avec } \Phi = -\frac{\log \circ p}{\lambda}
\end{aligned}
$$

Maximiser la log-likelihood

$$\begin{aligned}
\max_x \log(p(x|y)) &= \max_x \log(p(x,y)) \quad \text{car } p(x|y) = p(x,y) \times p(y) \\
&= \max_x \log(p(y|x)) + \log(p(x)) \\
&\quad \text{or } (y|x) = (v + Hx|x) \sim \mathcal{N}(Hx, \sigma^2) \\
&= \max_x -\frac{\|y - Hx\|^2}{2\sigma^2} + \log(p(x)) \\
&= \min_x \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(x) \quad \text{avec } \Phi = -\frac{\log \circ p}{\lambda}
\end{aligned}$$

But

On veut donc trouver $\hat{x}$ tel que : $\hat{x} = \arg\min_x \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(x)$

Une première méthode : ISTA

On veut minimiser : $F = f + g$ avec f dérivable et g pas forcément continue. On pose L la constante de Lipschitz de f.
En posant p_L tel que :

$$p_L(y) = \arg\max_x \left\{ g(x) + \frac{L}{2} \left\| x - \left(y - \frac{1}{L} \nabla f(y) \right) \right\|^2 \right\}$$

On peut montrer que l'on peut approximer $\hat{x} = \min_x F(x)$ en itérant :

$$x_{k+1} = p_L(x_k)$$

c'est-à-dire : $\hat{x} = \lim_{k \to \infty} x_{k+1}$

Une première méthode : ISTA

image de l'implémentation d'ISTA (si j'ai le temps)

Une deuxième méthode : Half Quadratic Splitting (HQS)

Idée

Diviser la variable x pour découpler le terme de fidélité et le terme de régularisation.

Idée

Diviser la variable x pour découpler le terme de fidélité et le terme de régularisation.

On a l'équivalence entre :

$$\min_x \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(x)$$

$$\Leftrightarrow \min_{x,z} \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(z) \quad \text{tel que} \quad z = x$$

En rajoutant un paramètre μ:

$$\Leftrightarrow \min_{x,z,\mu} \frac{1}{2}\|y - Hx\|^2 + \lambda\Phi(z) + \frac{\mu}{2}\|z - x\|^2$$

On appelle $\mathcal{L}_\mu(x, z)$, le terme que l'on doit minimiser.

On va approximer $\min_{x,z} \mathcal{L}_\mu$ par :

$$\lim_{k\to\infty} \min_z \min_x \min_z \ldots \min_x \mathcal{L}_\mu(x,z)$$

On peut alors trouver le minimum sur x et z en itérant :

$$\begin{cases} x_{k+1} = \arg\min_x & \|y - Hx\|^2 + \mu\|z_k - x\|^2 \\ z_{k+1} = \arg\min_z & \frac{\mu}{2}\|z - x_{k+1}\|^2 + \lambda\Phi(z) \end{cases}$$

Les systèmes de plug and play

Définition

Un système de plug and play est un système qui pour un problème donné le résout avec à la fois une méthode d'optimisation et une méthode d'apprentissage.

L'article propose de résoudre les 2 équations de la HQS comme ceci :

$$\begin{cases} x_{k+1} = \arg\min\limits_{x} \quad \|y - Hx\|^2 + \mu\|z_k - x\|^2 & \rightarrow \text{calcule de gradient} \\ z_{k+1} = \arg\min\limits_{z} \quad \dfrac{\mu}{2}\|z - x_{k+1}\|^2 + \lambda\Phi(z) & \rightarrow \text{Denoiser (CNNs)} \end{cases}$$

Pour x_{k+1} :

$$x_{k+1} = \arg\min_x \quad \|y - Hx\|^2 + \mu\|z_k - x\|^2$$
$$\Leftrightarrow x_{k+1} = (H^T H + \mu I)^{-1}(H^T y + \mu z_k)$$

Pour z_{k+1} :

$$z_{k+1} = \arg\min_z \quad \frac{\mu}{2}\|z - x_{k+1}\|^2 + \lambda\Phi(z)$$
$$\Leftrightarrow z_{k+1} = \arg\min_z \quad \frac{1}{2(\sqrt{\lambda/\mu})^2}\|z - x_{k+1}\|^2 + \Phi(z)$$
$$\Leftrightarrow z_{k+1} = Denoiser(x_{k+1}, \sqrt{\lambda/\mu})$$

Denoiser

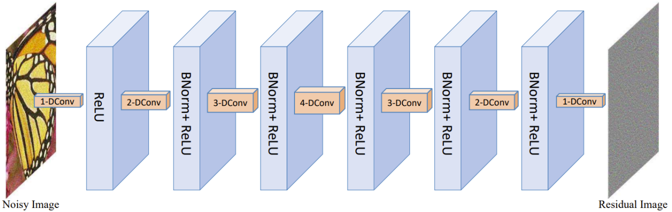


Figure 1. The architecture of the proposed denoiser network. Note that "s-DConv" denotes s-dilated convolution [63], here $s = 1, 2, 3$ and 4; "BNorm" represents batch normalization [32]; "ReLU" is the rectified linear units ($\max(\cdot, 0)$).

Figure: Denoiser utilisé dans l'article

Dans l'article

Entraînement de 25 Denoiser avec un écart-type de bruit constant de $2k$ où $k \in [1, 25]$.

Denoiser

Mon Denoiser

modèle : 5 couches CNNs au lieu de 7. (dilatation de 1, 2, 3, 2, 1)
sur des images en couleur de tailles : 64×64.
2 entraînements :

- **Const**: appris sur un écart-type de bruit constant de 5, image viennent d'un maxpooling d'image de 256×256.
- **Rand**: appris sur un écart-type de bruit uniforme sur $[1, 20]$, image qui vienne d'un crop.

Le Denoiser

Les résultats de tests :

	MSE	PSNR	MSSSIM
Baseline	3.74×10^4	34.3	0.998
Const	7.05×10^4	31.7	0.999

Table: Image avec un écart-type de bruit constant de 5.

	MSE	PSNR	MSSSIM
Baseline	18.7×10^4	27.38	0.990
Rand	9.01×10^4	30.6	0.996

Table: Image avec un écart-type de bruit variant entre 1 et 20.

Le Denoiser

inférence

Plug and play

résultats

Références

- **acticle:** Kai Zhang, Wangmeng Zuo, Shuhang Gu and Lei Zhang, *Learning Deep CNN Denoiser Prior for Image Restoration*. http://arxiv.org/abs/1704.03264
- **FISTA:** Amir Beck and Marc Teboulle, *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems*. https://www.ceremade.dauphine.fr/~carlier/FISTA
- **Lien de mon implémentation :** https://github.com/Highdrien/CNN-Denoiser-Prior