Supplementary Material to "Convolutional Sparse Coding for Image Super-resolution"

Shuhang Gu[1], Wangmeng Zuo[2], Qi Xie[3], Deyu Meng[3], Xiangchu Feng[4], Lei Zhang[1,*]
[1]Dept. of Computing, The Hong Kong Polytechnic University, Hong Kong, China
[2]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[3]School of Mathematics and Statistics, Xi′an Jiaotong University, Xi′an, China
[4]Dept. of Applied Mathematics, Xidian University, Xi'an, China
{cssgu, cslzhang}@comp.polyu.edu.hk, cswmzuo@gmail.com,
qixie.liwu@stu.xjtu.edu.cn, dymeng@mail.xjtu.edu.cn, xcfeng@mail.xidian.edu.cn

In this file, we first provide the details of the optimization methods for filter and mapping function training. Then, we present more visual examples of the SR results obtained by the competing methods.

1. SA-ADMM for filter and mapping function training

In this section, we first briefly introduce the algorithm of stochastic average alternating direction method of multipliers (SA-ADMM) [34]. Then, we provide the details of how to use SA-ADMM for filter and mapping function training in our CSC-SR model.

1.1. A Brief Introduction to SA-ADMM

Here we briefly introduce the algorithm of SA-ADMM. For more details of the algorithm and the convergence analysis, please refer to the original paper [34].

The ADMM algorithm is proposed to solve the following problem:

$$\min_{x,y} \phi(x) + \psi(y) \quad s.t. Ax + By = c. \tag{1}$$

In many real applications, $\phi(x)$ in (1) can be written as

$$\phi(x) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(x) + \Omega(x), \tag{2}$$

where ℓ_i is the contribution from the i-th sample, and n is the number of samples. For such a problem, the original ADMM algorithm suffers from a heavy computation burden in the step of updating x when n is a large number.

If the function $\ell_i(x)$ in (2) is L-smooth[1], SA-ADMM algorithm can be used to solve the problem (1). More specifically, in our case $\Omega = 0$. For problem (1) with $\phi(x) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(x)$, the SA-ADMM algorithm updates the variable x, y and the Lagrangian variable α alternatively:

$$\begin{aligned}
&x_{t+1} \leftarrow argmin_x \ \frac{1}{n}\sum_{i=1}^{n} \ell_i(x_{\tau_i(t)}) + \bigtriangledown \ell_i(x_{\tau_i(t)})^T(x - x_{\tau_i(t)}) + \frac{L}{2}\|x - x_{\tau_i(t)}\|^2 + \frac{\rho}{2}\|Ax + By_t - c - \alpha_t\|^2,\\
&y_{t+1} \leftarrow argmin_y \ \psi(y) + \frac{\rho}{2}\|Ax_{t+1} + By - c + \alpha_t\|^2,\\
&\alpha_{t+1} \leftarrow \ \alpha_t + Ax_{t+1} + By_{t+1} - c,
\end{aligned} \tag{3}$$

*This research is supported by the HK RGC GRF grant (PolyU5313/13E).

[1] Let $\|\cdot\|$ be the Euclidean norm. For a differentiable function f, we use $\bigtriangledown f$ to denote its gradient. A function f is L-smooth if $\| \bigtriangledown f(x) - \bigtriangledown f(y)\| \leq L\|x - y\|$.

where L is the scalar in the L-smooth definition, and $\tau_i(t)$ is defined as

$$\tau_i(t) = \begin{cases} t & i = k(t) \\ \tau_i(t-1) & otherwise \end{cases} . \tag{4}$$

The updating strategy of y and α is the same as the standard ADMM algorithm. For the x subproblem, by letting its derivative to zero, we have:

$$x_{t+1} \leftarrow (\rho A^T A + LI)^{-1}[L\overline{x}_t - \rho A^T(By_t - c + \alpha_t) - \overline{\bigtriangledown \ell}_t], \tag{5}$$

where $\overline{x}_t = \frac{1}{n}\sum_{i=1}^n x_{\tau_i(t)}$, and $\overline{\bigtriangledown \ell}_t = \frac{1}{n}\sum_{i=1}^n \bigtriangledown \ell_i(x_{\tau_i(t)})$.

Denote by (x^*, y^*) the optimal solution of (1) , Zhong et al. have proved that

$$\mathbb{E}[\Phi(\overline{x}_T, \overline{y}_T) - \Phi(x^*, y^*) + \gamma\|A\overline{x}_T + B\overline{y}_T - c\|] \leq \frac{1}{2T}\{\|x^* - x_0\|^2_{H_x} + nL\|x^* - x_0\|^2 + \|y^* - y_0\|^2_{H_y} + 2\rho(\frac{\gamma^2}{\rho^2} + \|\alpha_0\|^2)\}, \tag{6}$$

where $\|x\|_H = x^T H x$ for a positive semi-defined matrix H, $H_x = L_A I - \rho A^T A$ and $H_y = \rho B^T B$. Eq. (6) shows that the $\{\overline{x}_T, \overline{y}_T\}$ generated by SA-ADMM will converge with speed $\mathcal{O}(\frac{1}{T})$.

1.2. Filter training by SA-ADMM

The filter training problem in our CSC model aims to optimize the following problem:

$$\boldsymbol{f} = \arg\min_{\boldsymbol{f}} \sum_{k=1}^{K} \| \boldsymbol{Y}_k - \sum_{i=1}^{N} \boldsymbol{f}_i \otimes \boldsymbol{Z}_{k,i} \|^2_F, \ s.t. \|\boldsymbol{f}_i\|^2_F \leq e. \tag{7}$$

Note that here we do not omit the index k for the training image. $\boldsymbol{Y}_k$ is the kth training image and $\boldsymbol{Z}_{k,i}$ is the feature map produced by the ith filter $\boldsymbol{f}_i$ on $\boldsymbol{Y}_k$. Based on the properties of convolution and Kronecker product, we have the following equation:

$$vec(\boldsymbol{f} \otimes \boldsymbol{Z}) = \boldsymbol{F} * vec(\boldsymbol{Z}) = (\boldsymbol{I} \odot vec(\boldsymbol{Z}))vec(\boldsymbol{F}^T) = \mathbb{Z}^T * vec(f), \tag{8}$$

where $\odot$ and $vec(\bullet)$ denote the Kronecker product and the vectorization operation, respectively. $\boldsymbol{I}$ is the Identity matrix and $\boldsymbol{F}$ is the BCCB matrix corresponding to filter $\boldsymbol{f}$. $\mathbb{Z} = Image2Patch(\boldsymbol{Z})$ is the output of an *Image2Patch* operation on matrix $\boldsymbol{Z}$ with the size of filter $\boldsymbol{f}$, e.g., extracting all the patches from $\boldsymbol{Z}$ with the same size of $\boldsymbol{f}$.

Based on the above equations, the filter learning problem can be transformed to

$$\boldsymbol{f} = \arg\min_{\boldsymbol{f}} \sum_{k=1}^{K} \| vec(\boldsymbol{Y}_k) - [\mathbb{Z}^T_{k,1}, \mathbb{Z}^T_{k,2}, \ldots, \mathbb{Z}^T_{k,N}] * [vec(\boldsymbol{f}_1)^T, vec(\boldsymbol{f}_2)^T, \ldots, vec(\boldsymbol{f}_N)^T]^T \|^2_F, \ s.t. \|\boldsymbol{f}_i\|^2_F \leq e. \tag{9}$$

For the purpose of simplicity, we denote $[\mathbb{Z}^T_{k,1}, \mathbb{Z}^T_{k,2}, \ldots, \mathbb{Z}^T_{k,N}]$ by $\mathbb{Z}$ and $[vec(\boldsymbol{f}_1)^T, vec(\boldsymbol{f}_2)^T, \ldots, vec(\boldsymbol{f}_N)^T]^T$ by $\boldsymbol{f}$, the filter training problem with a large number of training images can be written as:

$$\boldsymbol{f} = \arg\min_{\boldsymbol{f}} \sum_k \|\boldsymbol{y}_k - \mathbb{Z}_k * \boldsymbol{f}\|^2 \quad s.t. \|\boldsymbol{f}_i\|^2 \leq e. \tag{10}$$

By introducing an augmented variable $\boldsymbol{s} = \boldsymbol{f}$, we can solve (10) by the SA-ADMM algorithm in (3):

$$\begin{aligned} &\boldsymbol{f}_{t+1} = [L\bar{\boldsymbol{f}}_t - \rho(\boldsymbol{d}_t - \boldsymbol{s}_t) - \frac{1}{K}\sum_{k=1}^{K} \mathbb{Z}^T_k(\mathbb{Z}_k \boldsymbol{f}_{\tau_j(t)} - \boldsymbol{Y}_k)]/(\rho + L) \\ &\boldsymbol{s}_{t+1} = argmin_{\boldsymbol{s}} \frac{\rho}{2}\|\boldsymbol{f}_{t+1} + \boldsymbol{d}_t - \boldsymbol{s}\|^2, \ \ s.t. \|\boldsymbol{s}_i\|^2 \leq e \\ &\boldsymbol{d}_{t+1} = \boldsymbol{d}_t + \boldsymbol{f}_{t+1} - \boldsymbol{s}_{t+1} \end{aligned} \tag{11}$$

For our square loss function in Eq. (7), a general scalar L which satisfies the L-smooth condition is the upper bound on the eigenvalues of $\mathbb{Z}^T\mathbb{Z}$. The $\boldsymbol{s}$ problem is a proximal problem with ℓ_2-norm ball constraint, which has a closed-form solution. Please note that, here $(\rho + L)$ is a scalar, and the updating of $\boldsymbol{f}$ does not need any matrix inverse calculation.

1.3. Mapping Function Learning by SA-ADMM

The mapping function learning problem in our CSC model aims to optimize the following problem:

$$\{\boldsymbol{W}\}=\arg\min_{\boldsymbol{W}}\sum_{k=1}^{K}\|\boldsymbol{X}_k-\sum_{j=1}^{M}\boldsymbol{f}_j^h\otimes g(\boldsymbol{Z}_{k,:}^l;\boldsymbol{w}_j)\|_F^2,\quad s.t.\ \boldsymbol{w}_j\succeq 0,|\boldsymbol{w}_j|_1=1. \tag{12}$$

Denote by $\tilde{\boldsymbol{Z}}_i^l$ the upsampling of LR feature map

$$\tilde{\boldsymbol{Z}}_{k,i}^l(x',y')=\begin{cases}\boldsymbol{Z}_{k,i}^l(x,y) & if\ mod(x',factor)=0\ and\ mod(y',factor)=0\\ 0 & otherwise\end{cases}, \tag{13}$$

then we have

$$[vec(\boldsymbol{Z}_{k,1}^h),vec(\boldsymbol{Z}_{k,2}^h),\dots,vec(\boldsymbol{Z}_{k,M}^h)]=[vec(\tilde{\boldsymbol{Z}}_{k,1}^l),vec(\tilde{\boldsymbol{Z}}_{k,2}^l),\dots,vec(\tilde{\boldsymbol{Z}}_{k,N}^l)]*\boldsymbol{W}, \tag{14}$$

where $\boldsymbol{W}=[\boldsymbol{w}_1,\boldsymbol{w}_2,\dots,\boldsymbol{w}_M]$ is the linear mapping function matrix, and $\boldsymbol{w}_j=[w_{j,1},w_{j,2},\dots,w_{j,N}]^T$ is the linear transform vector used to predict the jth HR feature map. Utilizing the BCCB matrix corresponding to the HR filters, the original problem (12) can be rewritten as

$$\{\boldsymbol{W}\}=\sum_{k=1}^{K}\arg\min_{\boldsymbol{W}}\|vec(\boldsymbol{X})-[\boldsymbol{F}_1^h,\dots,\boldsymbol{F}_M^h]*\begin{bmatrix}[vec(\tilde{\boldsymbol{Z}}_{k,1}^l),\dots,vec(\tilde{\boldsymbol{Z}}_{k,N}^l)] & & \\ & \cdots & \\ & & [vec(\tilde{\boldsymbol{Z}}_{k,1}^l),\dots,vec(\tilde{\boldsymbol{Z}}_{k,N}^l)]\end{bmatrix}*vec(\boldsymbol{W})\|_F^2$$

$$s.t.\ \boldsymbol{w}_j\succeq 0,|\boldsymbol{w}_j|_1=1.$$

Let

$$\boldsymbol{A}=\left\{\boldsymbol{F}_1^h*[vec(\tilde{\boldsymbol{Z}}_1^l),vec(\tilde{\boldsymbol{Z}}_2^l),\dots,vec(\tilde{\boldsymbol{Z}}_N^l)],\dots,\boldsymbol{F}_M^h*[vec(\tilde{\boldsymbol{Z}}_1^l),vec(\tilde{\boldsymbol{Z}}_2^l),\dots,vec(\tilde{\boldsymbol{Z}}_N^l)]\right\}, \tag{15}$$

and then the mapping function training problem has the form

$$\{\boldsymbol{W}\}=\sum_{k=1}^{K}\arg\min_{\boldsymbol{W}}\|vec(\boldsymbol{X})-\boldsymbol{A}*vec(\boldsymbol{W})\|_F^2\quad s.t.\ \boldsymbol{w}_j\succeq 0,|\boldsymbol{w}_j|_1=1. \tag{16}$$

We solve (16) by the SA-ADMM algorithm

$$\begin{aligned}&vec(\boldsymbol{W})_{t+1}=[Lvec(\bar{\boldsymbol{W}})_t-\rho(\boldsymbol{T}_t-\boldsymbol{S}_t)-\frac{1}{K}\sum_{k=1}^{K}\boldsymbol{A}_k^T(\boldsymbol{A}_k vec(\boldsymbol{W}_{\tau_j(t)})-\boldsymbol{X}_k)]/(\rho+L)\\&\boldsymbol{S}_{t+1}=argmin_{\boldsymbol{S}}\ \frac{\rho}{2}\|\boldsymbol{W}_{t+1}+\boldsymbol{T}_t-\boldsymbol{S}\|^2,\quad s.t.\ \boldsymbol{s}_j\succeq 0,\sum\boldsymbol{s}_j=1\\&\boldsymbol{T}_{t+1}=\boldsymbol{T}_t+\boldsymbol{W}_{t+1}-\boldsymbol{S}_{t+1}\end{aligned} \tag{17}$$

Different from the ℓ_2-norm proximal problem in (11) which has a closed-form solution, the second optimization problem in (17) is a proximal problem with nonnegative simplex constraint. Although it does not has a closed-form solution, we have the following **Remark** to show that each column of $\boldsymbol{S}$ can be solved very efficiently.

Remark 1. *Let* $\mathbf{e}=(1,1,\dots,1)^T$*; problem*

$$\min_{\mathbf{a}\in\Re^n}\|\mathbf{a}-\mathbf{b}\|_F^2\quad s.t.\mathbf{e}^T\mathbf{a}=1,-\mathbf{a}\leq 0, \tag{18}$$

has a globally optimal solution

$$\mathbf{a}^*=\mathbf{b}-\frac{\sum_{i=1}^{r}\mathbf{b}_{\tau_i}-1}{r}+[\frac{\sum_{i=1}^{r}\mathbf{b}_{\tau_i}-1}{r}\mathbf{e}-\mathbf{b}]_+, \tag{19}$$

where $\{\tau_1,\tau_2,\dots,\tau_n\}$ *is an index sequence which satisfies* $b_{\tau_1}\geq b_{\tau_2}\geq\cdots b_{\tau_n}$*.* r *is an integer which satisfies* $b_{\tau_r}>\frac{\sum_{i=1}^{r}\mathbf{b}_{\tau_i}-1}{r}$ *and* $b_{\tau_{r+1}}\leq\frac{\sum_{i=1}^{r}\mathbf{b}_{\tau_i}-1}{r}$.

Proof. The Lagrange function of (18)

$$\mathfrak{L}(\boldsymbol{a}, \lambda, \boldsymbol{v}) = \|\boldsymbol{a} - \boldsymbol{b}\|_F^2 + \lambda(\boldsymbol{e}^T\boldsymbol{a} - 1) - \boldsymbol{v}^T\boldsymbol{a} \quad s.t. \boldsymbol{v} \geq 0 \tag{20}$$

is a convex function. Let the partial derivative of $\mathfrak{L}$ w.r.t. $\boldsymbol{a}$ equal to zero, we have the optimal solution of $\boldsymbol{a}$:

$$\boldsymbol{a}^* = \boldsymbol{a} - \frac{\lambda}{2}\boldsymbol{e} + \frac{1}{2}\boldsymbol{v}. \tag{21}$$

Substitute (21) into (20), we have the dual function of (18)

$$\begin{aligned} g(\lambda, \boldsymbol{v}) &= \| - \frac{\lambda}{2}\boldsymbol{e} + \frac{1}{2}\boldsymbol{v}\|_F^2 + \lambda(\boldsymbol{e}^T(\boldsymbol{b} - \frac{\lambda}{2}\boldsymbol{e} + \frac{1}{2}\boldsymbol{v}) - 1) - \boldsymbol{e}^T(\boldsymbol{e} - -\frac{\lambda}{2}\boldsymbol{e} + \frac{1}{2}\boldsymbol{v}) \\ &= \frac{n}{4}\lambda^2 + \frac{1}{4}\|\boldsymbol{v}\|_F^2 - \frac{1}{2}\lambda\boldsymbol{e}^T\boldsymbol{v} + \lambda\boldsymbol{e}^T\boldsymbol{b} - \frac{n}{2}\lambda^2 + \frac{1}{2}\lambda\boldsymbol{e}^T\boldsymbol{v} - \lambda - \boldsymbol{v}^T\boldsymbol{b} + \frac{\lambda}{2}\boldsymbol{v}^T\boldsymbol{e} - \frac{1}{2}\|\boldsymbol{v}\|_F^2 \\ &= -\frac{n}{4}\lambda^2 - \frac{1}{4}\|\boldsymbol{v}\|_F^2 + \frac{1}{2}\lambda\boldsymbol{e}^T\boldsymbol{v} + \lambda\boldsymbol{e}^T\boldsymbol{b} - \lambda - \boldsymbol{v}^T\boldsymbol{b} \\ &= -\frac{1}{4}\|\lambda\boldsymbol{e} - \boldsymbol{v}\|_F^2 + \boldsymbol{b}^T(\lambda\boldsymbol{e} - \boldsymbol{v}) - \lambda. \end{aligned} \tag{22}$$

Thus, the dual problem

$$\begin{aligned} max_{\lambda,\boldsymbol{v}} g(\lambda, \boldsymbol{v}) &= max_{\lambda,\boldsymbol{v}} - \frac{1}{4}\|\lambda\boldsymbol{e} - \boldsymbol{v}\|_F^2 + \boldsymbol{b}^T(\lambda\boldsymbol{e} - \boldsymbol{v}) - \lambda \\ &= \boldsymbol{b}^T\boldsymbol{b} + max_{\lambda,\boldsymbol{v}} - \frac{1}{4}\|\lambda\boldsymbol{e} - \boldsymbol{v} - 2\boldsymbol{b}\|_F^2 - \lambda \end{aligned} \tag{23}$$

is a concave function, and the optimal solution can be achieved by letting the partial derivative equal to zero. The optimal solution of $\boldsymbol{v}$ is

$$\boldsymbol{v}^* = [\lambda\boldsymbol{e} - 2\boldsymbol{b}]_+ = max(\lambda\boldsymbol{e} - 2\boldsymbol{b}, 0). \tag{24}$$

We then substitute (24) into (23), and calculate its derivative w.r.t. λ:

$$\frac{\partial\{-\frac{1}{4}\|\lambda\boldsymbol{e} - 2\boldsymbol{b} - [\lambda\boldsymbol{e} - 2\boldsymbol{b}]_+\|_F^2 - \lambda\}}{\partial\lambda} = -\frac{1}{2}\mathfrak{I}(\lambda\boldsymbol{e} < 2\boldsymbol{b})^T(\lambda\boldsymbol{e} - 2\boldsymbol{b}) - 1, \tag{25}$$

in which

$$(\mathfrak{I}(\boldsymbol{x}))_i = \begin{cases} 1 & x_i \ is \ true \\ 0 & x_i \ is \ false \end{cases}, i = 1, 2, \ldots, n. \tag{26}$$

Let $-\frac{1}{2}\mathfrak{I}(\lambda\boldsymbol{e} < 2\boldsymbol{b})^T(\lambda\boldsymbol{e} - 2\boldsymbol{b}) - 1$ be zero, we have

$$\lambda = \frac{2(\sum_{i=1}^r \boldsymbol{b}_{\tau_i} - 1)}{r}, \tag{27}$$

where $\{\tau_1, \tau_2, \ldots, \tau_n\}$ is an index sequence which satisfies $b_{\tau_1} \geq b_{\tau_2} \geq \cdots b_{\tau_n}$, and r is an integer which satisfies $b_{\tau_r} > \frac{\sum_{i=1}^r \boldsymbol{b}_{\tau_i} - 1}{r}$ and $b_{\tau_{r+1}} \leq \frac{\sum_{i=1}^r \boldsymbol{b}_{\tau_i} - 1}{r}$.

Based on (21), (24) and (27), the optimal solution for problem (18) is

$$\boldsymbol{a}^* = \boldsymbol{b} - \frac{\sum_{i=1}^r \boldsymbol{b}_{\tau_i} - 1}{r} + [\frac{\sum_{i=1}^r \boldsymbol{b}_{\tau_i} - 1}{r}\boldsymbol{e} - \boldsymbol{b}]_+.$$

□

2. More Results

In the main paper, we have presented the SR results of 3 images by the competing methods. In Figs. 1-6 of this supplementary, we present more visual comparisons of the SR outputs by the competing methods.

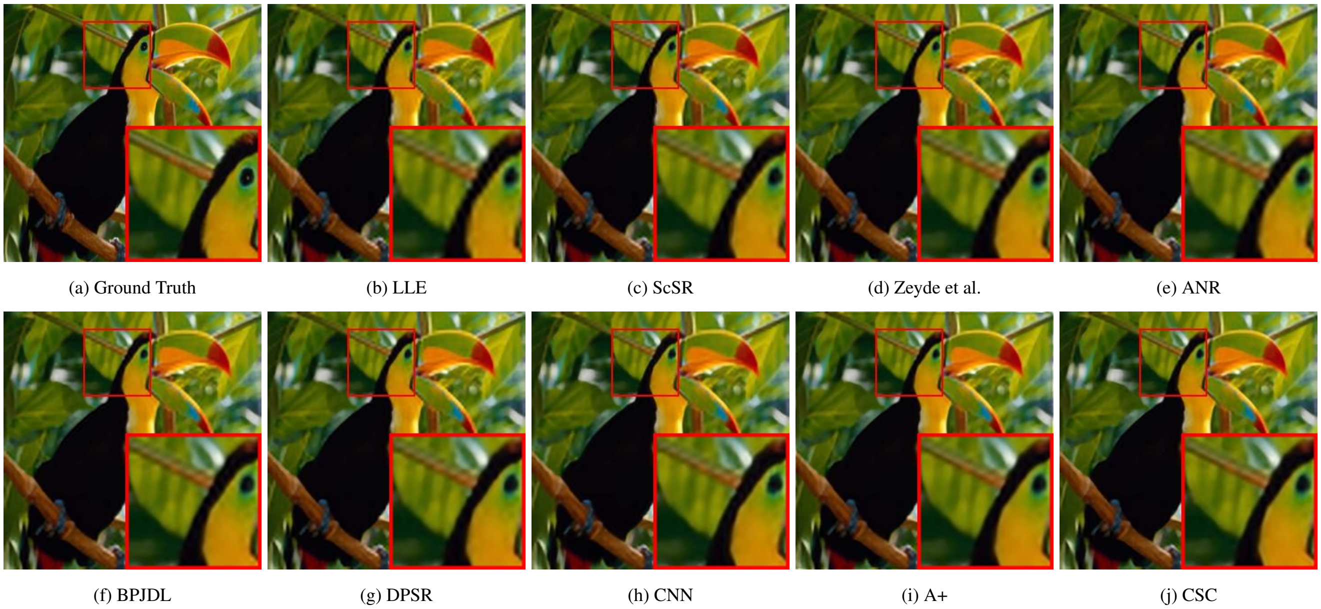


Figure 1. Super resolution results on image *Bird* by different algorithms (zooming factor 3).

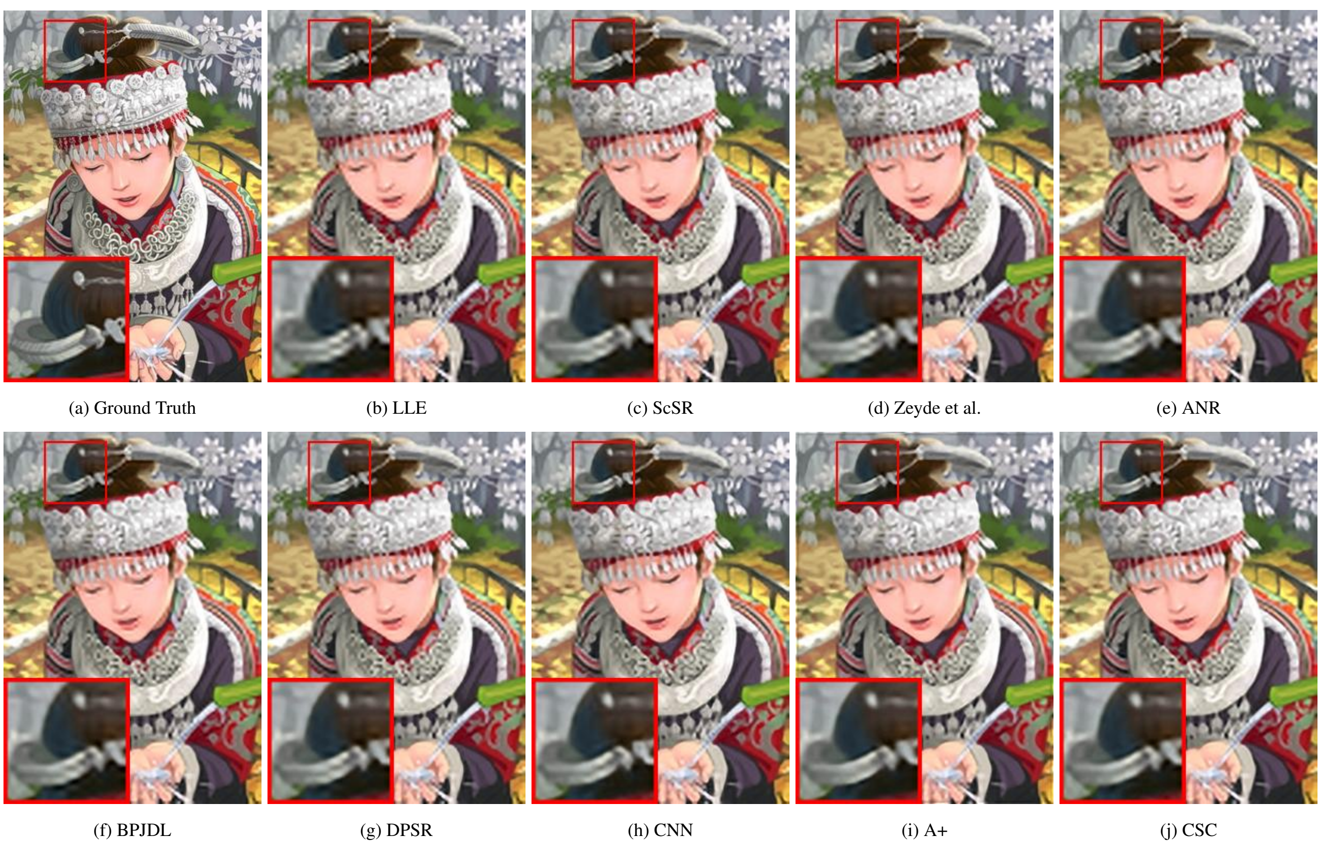
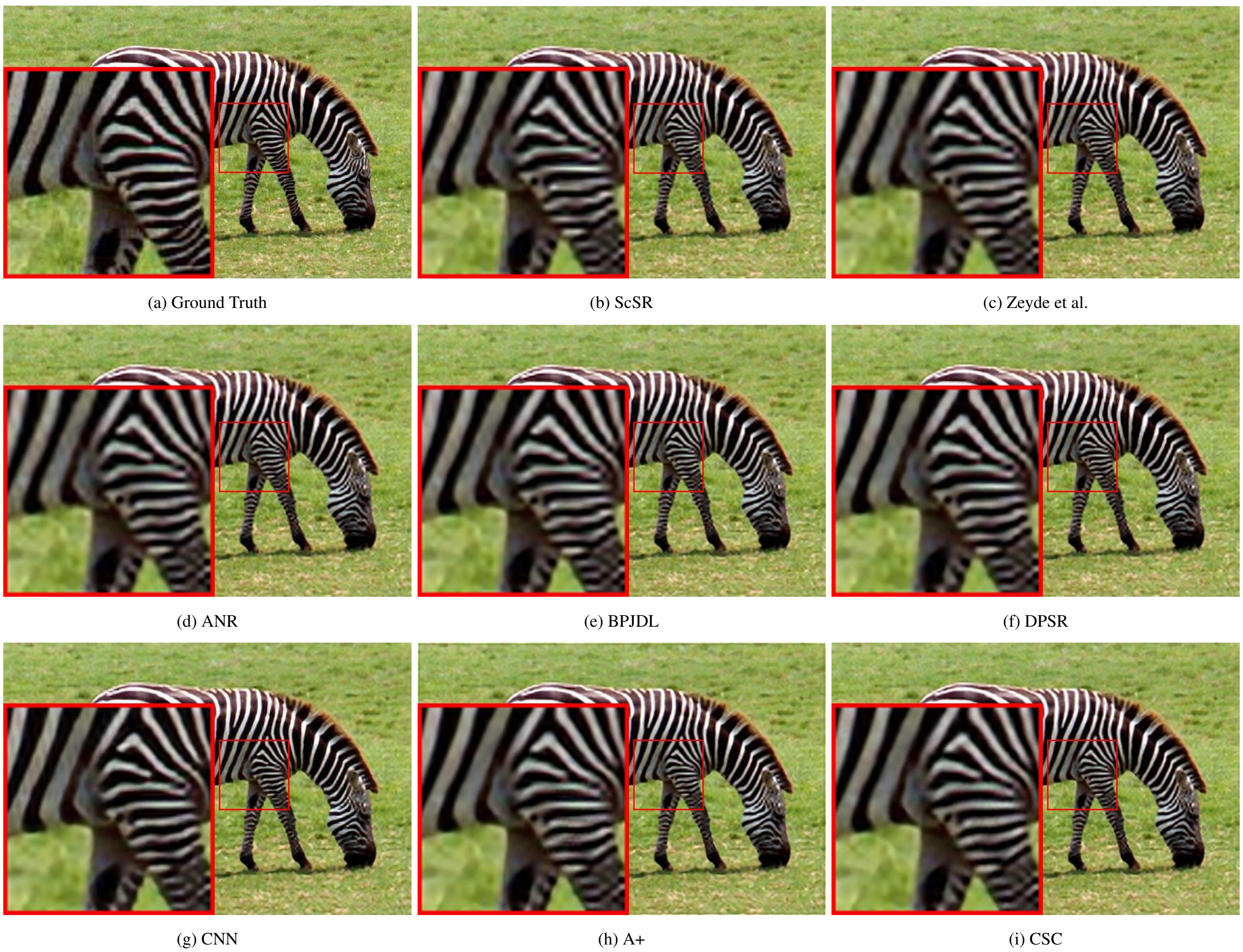


Figure 2. Super resolution results on image *Comic* by different algorithms (zooming factor 3).

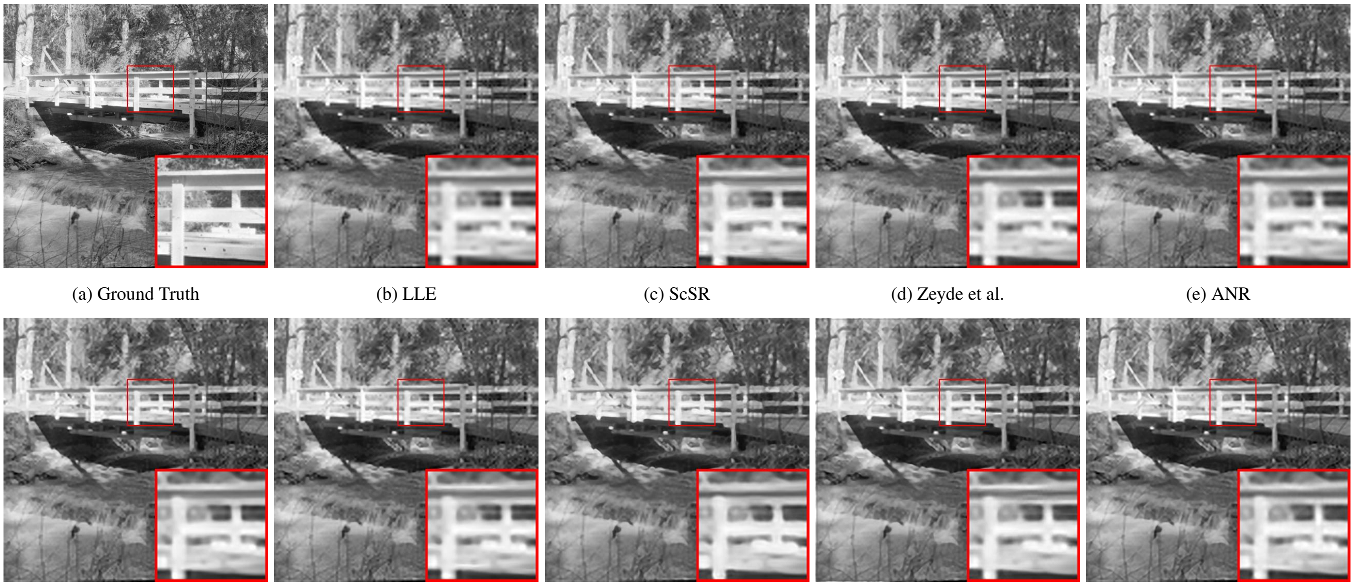


(a) Ground Truth (b) ScSR (c) Zeyde et al.

(d) ANR (e) BPJDL (f) DPSR

(g) CNN (h) A+ (i) CSC

Figure 3. Super resolution results on image *Zebra* by different algorithms (zooming factor 3).



(a) Ground Truth (b) LLE (c) ScSR (d) Zeyde et al. (e) ANR

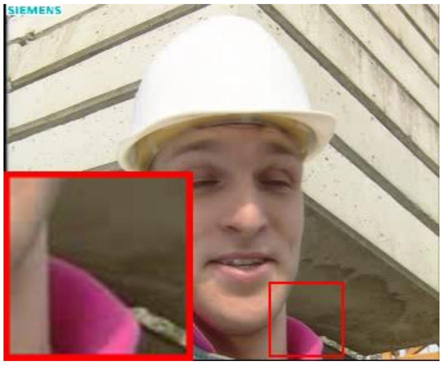
(f) BPJDL (g) DPSR (h) CNN (i) A+ (j) CSC

Figure 4. Super resolution results on image *Bridge* by different algorithms (zooming factor 4).

(a) Ground Truth (b) LLE (c) ScSR (d) Zeyde et al. (e) ANR

(f) BPJDL (g) DPSR (h) CNN (i) A+ (j) CSC

Figure 5. Super resolution results on image *Woman* by different algorithms (zooming factor 4).



(a) Ground Truth

(b) ScSR

(c) Zeyde et al.

(d) ANR

(e) BPJDL

(f) DPSR

(g) CNN

(h) A+

(i) CSC

Figure 6. Super resolution results on image *Foreman* by different algorithms (zooming factor 4).