Partially Supervised Neighbor Embedding for Example-Based Image Super-Resolution

IEEE Journal of Signal Processing,
Vol. 5, No. 2, 2011
Kaibing Zhang, Xinbo Gao, Xuelong Li, and Dacheng Tao

Presented by In-Yong Song

School of Electrical Engineering and Computer Science
Kyungpook National Univ.
Abstract

- Proposed method
  - Novel example-based image super-resolution reconstruction algorithm
    - Assumption for Textures
      - Containing multiple manifolds
    - Use of clustering and supervised neighbor embedding (CSNE)
      - Class predictor for low-resolution (LR) patches
        » Use of unsupervised Gaussian mixture model
      - Estimating high-resolution (HR) patches
        » Use of supervised neighbor embedding
        » Utilizing class label information of each patch
Introduction

◆ Existing super-resolution approaches

  – Interpolation-based method
    • Using base function or interpolation kernel function
      – Intuitional and highly efficient
      – Use of less additive information
        » Generating high resolution image of less perceptual version
  
  – Degrading model-based method
    • Three step
      – Registration with a reference image, deblurring and fusion
    • Use of a priori knowledge
      – Making SR reconstruction well-posed
• Tikhonov’s regularization
  – The most representative regularization based algorithms
• Farsiu et al.
  – Proposing bilateral total variation (BTV) operator
• Li et al.
  – Presenting locally adaptive bilateral total variation (LABTV) operator

• Problem of algorithms
  – Achieving good result
    » Under the limited imaging model and correct selection of regularization parameters
  – Difficult to practical application
– Example-based or learning-based method
  • Single-frame super-resolution method
    – Predicting high-frequency details of low-resolution
    – Learning co-occurrence relationship between LR patches and their corresponding HR patches
  • Freeman *et al.*
    – Modeling relationship between local regions of image and scenes by using Markov network
      » Sensitive to training examples
  • Chang *et al.*
    – Assuming LR image and its corresponding HR image
      » Similar local geometry
    – Use of locally linear embedding
      » Estimating HR patch by minimizing reconstruction error and optimal weights
    – Problem of neighbor sizes
• Chan et al.
  – Use of histogram matching
    » Problem of neighbor embedding

• Extended Chan et al.
  – Considering edge and neighborhood sizes
    » Important role of reconstruction quality of HR
  – Treating edge patches and non-edge patches differently in neighbor size
  – Problem of algorithm
    » Highly depending edges detection
    » Selecting neighborhood size

– Proposed method
  • Novel example-based image super-resolution
    – Using clustering and partially supervised neighbor embedding (CSNE) algorithm
    – Use of Gaussian mixture model
      » Clustering LR patches into different categories to guide neighbor embedding
Review of neighbor embedding for super-resolution reconstruction

◆ Neighbor embedding for image super-resolution

– Brief formulation

- \( X_s = \{ x_s^i \}_{i=1}^m \): a training patches of low resolution images
- \( Y_s = \{ x_s^i \}_{i=1}^m \): the corresponding high-resolution patches
- \( X_t = \{ x_t^j \}_{j=1}^m \): a set of low-resolution patches
- \( X_t = \{ x_t^j \}_{j=1}^m \): their corresponding high-resolution patches

- Three step
  - 1) Find k-nearest neighbors \( N_t^j = \{ x_t^{j(1)}, x_t^{j(2)}, \ldots, x_t^{j(K)} \} \) of each \( x_t^j \) among all patches from \( X_s \), i.e., \( N_t^j \subset X_s \). The distance measure matrix between \( x_s^j \) and \( x_t^j \) is defined as \( D_{ij}, i=1,2,\ldots,m, \ j=1,2,\ldots,n. \) The \( j \)th column of \( D \) is the distance between \( x_t^j \) and all patches from \( X_s \).
– 2) Compute weights $\omega_{ij}$ which best reconstruct each $x_t^j$ from its neighbors $N_t^j$

$$\epsilon^j = \min \left\| x_t^j - \sum_{x_s^i \in N_t^j} \omega_{ij} x_s^i \right\|$$

(1)

where $\omega_{ij}$ is the weights for $x_s^i$, subject to the following constraints:

$$\sum_{x_s^i \in N_t^j} \omega_{ij} = 1 \text{ and } \omega_{ij} = 0 \text{ if } x_s^i \notin N_t^j$$

– 3) Compute each $y_t^j$ as follow:

$$y_t^j = \sum_{x_s^i \in N_t^j} \omega_{ij} y_s^i$$

(2)

• First, to find $k$-nearest neighbors of the input LR patch
• Second, to compute optimal weights by minimizing reconstruction error
• Last, to compute a HR patch with linear combination of the HR patches corresponding to the $k$-nearest LR patches
Proposed algorithm

◆ Describing general idea of the proposed algorithm
– Framework of algorithm

Fig. 1. Illustration of neighbor embedding for image super-resolution.
Image patches classification with Gaussian mixture model

– Gaussian mixture model (GMM)

• Set of parameters describing the mean and variation of the random vectors

\[
p(x_i | \theta^*) = \sum_{c=1}^{C} p(x_i | \theta^*_c)
\]

where \( \theta^*_c = (\alpha^*_c, \mu^*_c, \Sigma^*_c) \) is the parameter set of the \( c \)th sub-calss.

• The probability density function for random vectors \( x_i \)

\[
p_{x_i|c}(x_i | c, \theta^*) = \frac{1}{(2\pi)^{D/2} \sqrt{|\Sigma^*_c|}} \exp\left\{ -\frac{1}{2} (x_i - \mu^*_c)^T (\sum^*_c)^{-1} (x_i - \mu^*_c) \right\}
\]

here \( D \) represents the dimension of random vector \( x_i \).
• The conditional probability

\[ p_{x_i}(x_i \mid \theta^*) = \sum_{c=1}^{C} p_{x_i \mid c}(x_i \mid c, \theta^*) \alpha_c^* \]  

(5)

• Thus, log of the probability of the \( X_N = \{x_i\}_{i=1}^{N} \)

\[ \log p_{x_i}(x_i, C, \theta^*) = \sum_{i=1}^{N} \log \sum_{c=1}^{C} p_{x_i \mid c}(x_i \mid c, \theta^*) \alpha_c^* \]  

(6)

• For the model parameters \( \theta^* \), a maximum of the fitness in terms of maximum likelihood(ML)

\[ \hat{\theta}_{ML}^* = \arg \max_{\theta^*} \log p_{x_i}(x_i \mid C, \theta^*) \]  

(7)
Clustering and supervised neighbor embedding for SR reconstruction

- Partially supervised neighbor embedding for SR reconstruction
  - Taking account of the low-resolution patches containing multiple manifolds, corresponding to classes
  - Using class label information of input LR patches
    - Tuning the distance between samples in different classes
      » Embedding $k$-neighbors of the patches from the same class
– Algorithm

1) The training samples \( X_s \) are divided into \( C \) clusters by using Gaussian mixture model.

2) Find \( k \)-nearest neighbors \( N_t^j = \{x_t^{j(1)}, x_t^{j(2)}, \ldots, x_t^{j(K)}\} \) of each \( x_t^j \) among all patches from \( X_s \). The distance measure matrix \( D_{ij} \) between \( x_s^i \) and \( x_t^j \) is tuned as follows:

\[
D_{ij} = D_{ij} + \alpha \max(D) M_{ij} \quad i = 1, 2, \ldots, m, j = 1, 2, \ldots, n
\]

where \( M_{ij} = 0 \) if \( x_s^i \) and \( x_t^j \) belong to the same class, otherwise 1.

– Supervised strength parameter \( \alpha \) in (8)
  – Control of Using how much class information
    » Finding \( k \)-nearest neighbors
  – Value range from [0, 1]
    » Calling partially supervised neighbor embedding within (0, 1)
Proposed example-based image super-resolution algorithm

– Pseudo-code of proposed algorithm

---

**Algorithm 1** Example-based image super-resolution

**Objective:** Estimate HR $Y_t$ of an input LR $X_t$

**Input:**
- Low-resolution input image $X_t$;
- Neighborhood size $k$;
- Patches size $q \times q$;
- Scaling factor $s$ (an integer more than 1);
- Training image $Y_s$ and its downsampled (by $sX$) version $X_s$;
- Supervised strength parameter $\alpha$.

**Output:**
- High-resolution image $Y_t$. 
• 1) Partition $X_t$ and $X_s$ into patches of size $q \times q$ with overlapping by one or two pixels.

• 2) Partition $Y_s$ into patches of size $sq \times sq$ with overlapping by $s$ or $2s$ pixels accordingly, thus generating high-low training patch-pairs from $X_s$, i.e., $\{y_s^i : x_s^j\}_{i=1}^m$.

• 3) For all low-resolution patches $\{x_s^i\}_{i=1}^m$, given initial cluster $C$, a class predictor (called CP) is learnt through unsupervised Gaussian mixture model. After CP constructed, all $\{x_s^i\}_{i=1}^m$ are divided into its corresponding class.

• 4) For each patch $x_t^j$ in $X_t$, find $k$-nearest neighbors among all $\{x_s^i\}_{i=1}^m$, and the distance between $x_s^i$ and $x_t^j$ is computed as

$$D_{ij} = D_{ij} + \alpha \max(D)M_{ij} \quad i = 1, 2, \ldots, m, j = 1, 2, \ldots, n$$

\[ \text{if} \quad \text{CP}(x_s^i, x_t^j) = 1 \quad \text{then} \]

$$M_{ij} = 0$$

\[ \text{else} \]

$$M_{ij} = 1$$

end if
• 5) Compute the optimal weights $W_j$ by minimizing the error of reconstruction for $x_i^j$;
• 6) Compute the high-resolution patch to be estimated $y_t^j$ from the weights sum of $W_j$ with the $k$ patches in $Y_s$ corresponding to the $k$-nearest neighbors found in $X_s$;
• 7) Merge all $y_t^j$ s to obtain $Y_t$
Computational complexity

- Considering each step of algorithm 1
  - \( m \) and \( n \) : the number of patches in the training set and the input image
  - \( q^2 \) : the size of each LR patch
  - \( C \) : the number of clusters of the Gaussian mixture model
  - \( k \) : the number of neighbors

Table I. Computational complexity of the proposed algorithm in each step

<table>
<thead>
<tr>
<th>Step</th>
<th>The computational complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step (1)</td>
<td>( O(m + n) )</td>
</tr>
<tr>
<td>Step (2)</td>
<td>( O(m) )</td>
</tr>
<tr>
<td>Step (3)</td>
<td>( O(Cq^2m + m) )</td>
</tr>
<tr>
<td>Step (4)</td>
<td>( O(kq^2m^2n) )</td>
</tr>
<tr>
<td>Step (5)</td>
<td>( O(q^2nk^3) )</td>
</tr>
<tr>
<td>Step (6)</td>
<td>( O(kn) )</td>
</tr>
<tr>
<td>Step (7)</td>
<td>( O(n) )</td>
</tr>
</tbody>
</table>
Experimental results and evaluations

- Training and testing images
  - Six images
    - Including human, plants and animals

Fig. 2. Training images. From left to right, top to bottom are labeled No.1 to No. 6.
– Getting input LR images
  • Degrading by averaging blurring operation
    – within 4 x 4 neighbors
  • Down-sampling with factor 4
    – product a testing input image

Fig. 3. Input LR images.
Experimental results

- Comparing performance of the CSNE with SRNE and NeedFS
  - 3 x 3 size of low-resolution patches
    - Two pixels overlapped between adjacent patches
  - 12 x 12 size of high-resolution patches
    - Eight pixels overlapped between adjacent patches
  - Parameter $\alpha$
    - Set up as 0.1

\[
PSNR = 10 \log_{10} \frac{255^2}{MSE}
\]  \quad (9)
• Objective assessment
  – Use of peak-signal-to-noise ratio (PSNR)

\[ PSNR = 10 \log_{10} \frac{255^2}{MSE} \] (9)

**Table II.** Performance of PSNR (dB) using different algorithms

<table>
<thead>
<tr>
<th>Testing image</th>
<th>Bicubic</th>
<th>SRNE</th>
<th>NeedFS</th>
<th>CSNE</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.1</td>
<td>32.1108</td>
<td>32.8328</td>
<td>32.8763</td>
<td>32.8500</td>
</tr>
<tr>
<td>No.2</td>
<td>33.2212</td>
<td>33.3149</td>
<td>33.4700</td>
<td>33.4333</td>
</tr>
<tr>
<td>No.3</td>
<td>30.6383</td>
<td>30.7247</td>
<td>30.9269</td>
<td><strong>30.9484</strong></td>
</tr>
<tr>
<td>No.4</td>
<td>33.4183</td>
<td>33.7501</td>
<td>33.5985</td>
<td><strong>34.1458</strong></td>
</tr>
<tr>
<td>No.5</td>
<td>31.4471</td>
<td>31.7679</td>
<td>32.3715</td>
<td>32.2758</td>
</tr>
<tr>
<td>No.6</td>
<td>30.8027</td>
<td>30.9385</td>
<td>31.1828</td>
<td><strong>31.2046</strong></td>
</tr>
<tr>
<td>Average</td>
<td>31.9397</td>
<td>32.2215</td>
<td>32.4043</td>
<td><strong>32.4763</strong></td>
</tr>
</tbody>
</table>
– Resulting image

• No. 1

**Fig. 4.** 4X recovery of No. 1 using different methods. From left to right, top to bottom: the low resolution image; the original image; Bicubic; SRNE; NeedFs; the proposed method.

**Fig. 5.** Local magnification of No. 1. From left to right: the original image; Bicubic; SRNE; NeedFs; the proposed method.
Fig. 6. 4X recovery of No. 4 using different methods. From left to right, top to bottom: the low resolution image; the original image; Bicubic; SRNE; NeedFs; the proposed method.

Fig. 7. Local magnification of No. 4. From left to right: the original image; Bicubic; SRNE; NeedFs; the proposed method.
• No. 6

**Fig. 8.** 4X recovery of No. 5 using different methods. From left to right, top to bottom: the low resolution image; the original image; Bicubic; SRNE; NeedFs; the proposed method.

**Fig. 9.** Local magnification of No. 1. From left to right: the original image; Bicubic; SRNE; NeedFs; the proposed method.
Part clustering results and the weight of neighbor patches changed based on cluster information

- Demonstrating clustering results
  - Effect on the weight of neighbor patches for reconstruction

**Table III.** Number of patches of each cluster of the No. 3 Training set.

<table>
<thead>
<tr>
<th>Cluster No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Patches</td>
<td>3051</td>
<td>1457</td>
<td>1551</td>
<td>2507</td>
<td>2753</td>
<td>3007</td>
<td>4486</td>
<td>3736</td>
<td>1913</td>
<td>1056</td>
</tr>
</tbody>
</table>
Fig. 10. Part of clustering results for LR patches in the No. 3 training set.
– Showing how the weight of neighbor patches changed

• Based on clustering information

**Table IV.** Comparison of weights of neighbor patches between SRNE (five neighbors) and the proposed algorithm (ten neighbors) for the 2000th patch

<table>
<thead>
<tr>
<th>SRNE</th>
<th>Index</th>
<th>Patches</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5853</td>
<td>25175</td>
<td>0.3384</td>
</tr>
<tr>
<td></td>
<td>5365</td>
<td>-0.0133</td>
<td>0.2459</td>
</tr>
<tr>
<td></td>
<td>14360</td>
<td>0.1336</td>
<td></td>
</tr>
<tr>
<td>The proposed algorithm</td>
<td>Index</td>
<td>Patches</td>
<td>Weights</td>
</tr>
<tr>
<td></td>
<td>5853</td>
<td>25175</td>
<td>0.3138</td>
</tr>
<tr>
<td></td>
<td>14360</td>
<td>-0.2762</td>
<td>0.3859</td>
</tr>
<tr>
<td></td>
<td>25238</td>
<td>-0.3170</td>
<td>-0.1695</td>
</tr>
<tr>
<td></td>
<td>12170</td>
<td>0.2847</td>
<td>0.1314</td>
</tr>
<tr>
<td></td>
<td>11227</td>
<td>0.2132</td>
<td></td>
</tr>
</tbody>
</table>

**Table V.** Comparison of weights of neighbor patches between SRNE (five neighbors) and the proposed algorithm (ten neighbors) for the 3000th patch

<table>
<thead>
<tr>
<th>SRNE</th>
<th>Index</th>
<th>Patches</th>
<th>Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6893</td>
<td>6789</td>
<td>0.1215</td>
</tr>
<tr>
<td></td>
<td>6429</td>
<td>0.2778</td>
<td>0.2832</td>
</tr>
<tr>
<td></td>
<td>14095</td>
<td>0.0349</td>
<td></td>
</tr>
<tr>
<td>The proposed algorithm</td>
<td>Index</td>
<td>Patches</td>
<td>Weights</td>
</tr>
<tr>
<td></td>
<td>6893</td>
<td>6789</td>
<td>0.4600</td>
</tr>
<tr>
<td></td>
<td>6429</td>
<td>-0.4124</td>
<td>0.3381</td>
</tr>
<tr>
<td></td>
<td>16384</td>
<td>-0.4547</td>
<td>0.3427</td>
</tr>
<tr>
<td></td>
<td>20651</td>
<td>-0.2812</td>
<td>0.0541</td>
</tr>
<tr>
<td></td>
<td>12645</td>
<td>-0.0564</td>
<td></td>
</tr>
</tbody>
</table>
**Effect of supervised strength parameter**

- Affecting how much class-information for neighbor embedding
  
  - If $\alpha = 0$, no any class information of patches used
    - Equivalent to the algorithm SRNE
  
  - If $\alpha = 1$, a fully supervised neighbor embedding

**Fig. 11.** Performance of PSNR versus different number of neighbors for NO. 3. The supervised strength alpha ranges from 0 to 0.3.

**Fig. 12.** Performance of PSNR versus different number of neighbors for NO. 3. The supervised strength alpha ranges from 0.4 to 1.0.
Fig. 13. Performance of PSNR versus different number of neighbors for NO. 6. The supervised strength alpha ranges from 0 to 0.4.

Fig. 14. Performance of PSNR versus different number of neighbors for NO. 6. The supervised strength alpha ranges from 0.5 to 1.0.
Conclusions

◆ Proposed method
  – Novel example-based image super-resolution reconstruction algorithm
    ● Use of clustering and supervised neighbor embedding (CSNE)
      – Use of unsupervised Gaussian mixture model
        » Class predictor for low-resolution (LR) patches
      – Utilizing class label information of each patch
      – Use of supervised neighbor embedding
        » Estimating high-resolution (HR) patches
  – Experiment of proposed method
    ● Evaluating proposed algorithm
– An Improved Super-Resolution with Manifold Learning and Histogram Matching
Locally Linear Embedding

1. Select neighbors

2. Reconstruct with linear weights

3. Map to embedded coordinates
Two key steps

1. Find weight matrix $W$ of linear coefficients:
   \[
   \varepsilon(W) = \sum_i \left| \tilde{x}_i - \sum_j W_{ij} \tilde{x}_j \right|^2
   \]
   Enforce sum-to-one constraint with the Lagrange Multiplier:
   \[
   W_j = \frac{1 - \sum_k C_{ik}^{-1}(\tilde{x} \cdot \eta_k)}{\sum_k C_{jk}^{-1}} \sum_k C_{jk}^{-1}(\tilde{x} \cdot \eta_k + \eta_k)
   \]

2. Find projected vectors $Y$ to minimize reconstruction error
   \[
   \Phi(Y) = \sum_i \left| \tilde{y}_i - \sum_j W_{ij} \tilde{y}_j \right|^2
   \]
   must solve for whole dataset simultaneously
   We add constraints to prevent multiple/degenerate solutions:
   \[
   \sum_i \tilde{y}_i = 0
   \]
   \[
   \frac{1}{N} \sum_i \tilde{y}_i \otimes \tilde{y}_i = I
   \]
   cost function becomes:
   \[
   M_{ij} = \delta_{ij} - W_{ij} - W_{ji} + \sum_k W_{ki} W_{kj}
   \]
   the optimal embedded coordinates are given by bottom $m+1$
eigenvectors of the marix $M$