EnsIR: An Ensemble Algorithm for Image Restoration via Gaussian Mixture Models

Shangquan Sun[1,2] **Wenqi Ren**[3,4*] **Zikun Liu**[5]
Hyunhee Park[6] **Rui Wang**[1,2] **Xiaochun Cao**[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]School of Cyber Science and Technology, Shenzhen Campus of Sun Yat-sen University
[4]Guangdong Provincial Key Laboratory of Information Security Technology
[5]Samsung Research China - Beijing (SRC-B)
[6]Camera Innovation Group, Samsung Electronics
`{sunshangquan,wangrui}@iie.ac.cn`
`{zikun.liu,inextg.park}@samsung.com`
`{renwq3,caoxiaochun}@mail.sysu.edu.cn`

Abstract

Image restoration has experienced significant advancements due to the development of deep learning. Nevertheless, it encounters challenges related to ill-posed problems, resulting in deviations between single model predictions and ground-truths. Ensemble learning, as a powerful machine learning technique, aims to address these deviations by combining the predictions of multiple base models. Most existing works adopt ensemble learning during the design of restoration models, while only limited research focuses on the inference-stage ensemble of pre-trained restoration models. Regression-based methods fail to enable efficient inference, leading researchers in academia and industry to prefer averaging as their choice for post-training ensemble. To address this, we reformulate the ensemble problem of image restoration into Gaussian mixture models (GMMs) and employ an expectation maximization (EM)-based algorithm to estimate ensemble weights for aggregating prediction candidates. We estimate the range-wise ensemble weights on a reference set and store them in a lookup table (LUT) for efficient ensemble inference on the test set. Our algorithm is model-agnostic and training-free, allowing seamless integration and enhancement of various pre-trained image restoration models. It consistently outperforms regression-based methods and averaging ensemble approaches on 14 benchmarks across 3 image restoration tasks, including super-resolution, deblurring and deraining. The codes and all estimated weights have been released in Github.

1 Introduction

Image restoration has witnessed significant progress over the decades, especially with the advent of deep learning. Numerous architectures have been developed to solve the problem of restoration, including convolutional neural networks (CNNs) [10, 82], vision Transformers (ViTs) [39, 81, 93] and recently, vision Mambas [27]. However, single models with different architectures or random initialization states exhibit prediction deviations from ground-truths, resulting in sub-optimal restoration results.

[*]Corresponding Author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

To alleviate this problem, ensemble learning, a traditional but influential machine learning technique, has been applied to image restoration. It involves combining several base models to obtain a better result in terms of generalization and robustness [18, 20, 26, 49, 57]. However, most ensemble methods in image restoration focus on training-stage ensemble requiring the ensemble strategy to be determined while training multiple models, thus sacrificing flexibility of changing models and convenience for plug-and-play usage [31, 35, 40, 44, 46, 54, 59, 70, 76]. In contrast, there is a demand for advanced post-training ensemble methods in the image restoration industry, where researchers still prefer averaging as their primary choice [1, 15, 41, 52, 66, 68, 92].

Despite the industrial demand, post-training ensemble in image restoration is challenging for tradition ensemble algorithms originally designed for classification or regression. Unlike classification and regression, image restoration predictions are matrices with each pixel is correlated with others and range from 0 to 255. As a result, traditional methods like bagging [6] and boosting [28] either require enormous computational resources for the restoration task or fail to generalize well due to the imbalance between candidate number and feature dimension. As an alternative, Jiang *et al.* propose a post-training ensemble algorithm for super-resolution by optimizing a maximum a posteriori problem with a reconstruction constraint [31]. However, this constraint requires an explicit expression of the degradation process, which is extremely difficult to define for other restoration tasks beyond super-resolution. It also necessitates prior knowledge of the base models' performance, further limiting its practical application. These issues with both traditional and recent ensemble methods lead researchers in image restoration to prefer weighted averaging as their primary ensemble approach [1, 15, 52, 68, 92].

To this end, we formulate the ensemble of restoration models using Gaussian mixture models (GMMs), where ensemble weights can be efficiently learned via the expectation maximization (EM) algorithm and stored in a lookup table (LUT) for subsequent inference. Specifically, we first base on the Gaussian prior that assumes the prediction error of each model follows a multivariate Gaussian distribution. Based on the prior, the predictions of multiple samples can be appended into a single variable following Gaussian. By partitioning pixels into various histogram-like bins based on their values, we can convert the problem of ensemble weight estimation into various solvable univariate GMMs. As a result, the univariate GMMs can be solved to obtain range-wise ensemble weights by the EM algorithm, with the means and variances of Gaussian components estimated based on the observation priors. We estimate these weights on a reference set and store them in a LUT for efficient inference on the test set. Our method does not require training or prior knowledge of the base models and degradation processes, making it applicable to various image restoration tasks.

Our contributions mainly lie in three areas:

- Based on the Gaussian prior, we partition the pixels of model predictions into range-wise bin sets of mutually exclusive ranges and derive the ensemble of multi-model predictions into weight estimation of various solvable univariate GMMs.
- To solve the univariate GMMs and estimate ensemble weights, we leverage the EM algorithm with means and variances initialized by observed prior knowledge. We construct a LUT to store the range-wise ensemble weights for efficient ensemble inference.
- Our ensemble algorithm does not require extra training or knowledge of the base models. It outperforms existing post-training ensemble methods on 14 benchmarks across 3 image restoration tasks, including super-resolution, deblurring and deraining.

2 Related Works

Ensemble Methods. Ensemble methods refer to approaches that fuse the predictions of multiple base models to achieve better results than any of individual model [20]. Traditional ensemble methods include bagging [6], boosting [28], random forests [7], gradient boosting [24], histogram gradient boosting [14, 34], etc. These methods have been applied to various fields in classification and regression, such as biomedical technology [77, 78], intelligent transportation [53, 84], and pattern recognition [61, 91].

Image restoration. Image restoration, as a thriving area of computer vision, has been making significant progress since the advent of deep learning [10, 65]. Various model architectures have been proposed to address image restoration tasks, such as convolutional neural networks (CNNs) [19, 56, 73, 90], multilayer perceptron (MLPs) [67], vision Transformers (ViTs) [39, 62, 81], etc. Additionally, various structural designs like multi-scale [56], multi-patch [64, 89], and progressive learning [82] have been adopted to improve the representative capacity. It is known that CNNs excel at encoding local features, while ViTs are adept at capturing long range dependencies. Despite this significant progress, single models still generate predictions that deviate from ground-truths, leading researchers in industry to adopt multi-model ensembles to achieve better performance [15, 52, 68, 92].

Ensemble Learning in Image Restoration. Some works incorporate ensemble learning into image restoration by training multiple networks simultaneously and involving ensemble strategy during the training process [2, 3, 9, 11–13, 16, 17, 21, 29, 31, 33, 35–38, 40, 42–46, 54, 55, 59, 63, 69, 70, 72, 75, 76, 79, 80, 85, 87]. However, the majority of them require additional training or even training from scratch alongside ensemble. Only a few focus on ensemble at the post-training stage [31, 66]. Among them, self-ensemble [66] geometrically augments an input image, obtains super-resolution predictions of augmented candidates and applies averaging ensemble to the candidates, which is orthogonal to the scope of our work. RefESR [31] requires a reconstruction objective which must get access to the degradation function, prohibiting its application to tasks other than super-resolution. In general, limited works focus on the training-free ensemble of image restoration. There lacks a general ensemble algorithm for restoration despite industry demand [1, 15, 52, 68, 92].

3 Proposed Ensemble Method for Image Restoration

In Sec. 3.1, we first present the formulation of the ensemble problem in image restoration. We then formulate our ensemble method in the format of Gaussian mixture models (GMMs) over partitioned range-wise bin sets in Sec. 3.2. We derive the expectation maximization (EM) algorithm with known mean and variance as prior knowledge to solve the GMMs problems in Sec. 3.3

3.1 Ensemble Formulation of Image Restoration

Given a test set $\mathbb{T} = \{\hat{\mathbf{X}}_n, \hat{\mathbf{Y}}_n\}$ with numerous pairs of input images and ground-truths, suppose we have M pre-trained base models for image restoration, $f_1, ..., f_M$. For a model f_m where $m \in \{1, ..., M\}$, its prediction is denoted by $\tilde{\mathbf{X}}_{m,n} = f_m(\hat{\mathbf{X}}_n)$ for abbreviation. We consider the ensemble of the predictions as a weighted averaging, i.e.,

$$\tilde{\mathbf{Y}}_n = \boldsymbol{\beta}_n^\top \begin{bmatrix} \tilde{\mathbf{X}}_{1,n} & \cdots & \tilde{\mathbf{X}}_{M,n} \end{bmatrix}, \forall n \tag{1}$$

where $\tilde{\mathbf{Y}}_n$ is the ensemble result, and $\boldsymbol{\beta}_n \in \mathbb{R}^M$ is the vector of weight parameters for the ensemble. The widely-used averaging ensemble strategy in image restoration assigns equal weights for all samples and pixels, i.e., $\boldsymbol{\beta}_n = \begin{bmatrix} \frac{1}{M} & \cdots & \frac{1}{M} \end{bmatrix}$. A recent method in the NTIRE 2023 competition assigns weights inversely proportional to the mean squared error between the predictions and their average [92]. However, they adopt globally constant weights for all pixels and samples, neglecting that the performances of base models may fluctuate for different patterns and samples.

Alternatively, we start from the prospective of GMMs and assign range-specific weights based on the EM algorithm.

3.2 Restoration Ensemble as Gaussian Mixture Models

Similar to RefESR [31], suppose we have a reference set $\mathbb{D} = \{\mathbf{X}_n, \mathbf{Y}_n\}_{n=1}^N$ with N pairs of input images and ground-truths. We assume the reference set and test set are sampled from the same data distribution, i.e., $\mathbb{D}, \mathbb{T} \sim \mathcal{D}$.

For each model f_m, its prediction is denoted by $\mathbf{X}_{m,n} = f_m(\mathbf{X}_n) \in \mathbb{R}^{3 \times H \times W}$. We use $\mathbf{x}_{m,n}, \mathbf{y}_n \in \mathbb{R}^L$ to represent the flattened vector of the matrices $\mathbf{X}_{m,n}, \mathbf{Y}_n$, where $L = 3HW$. Based on Gaussian prior, it can be assumed that the estimation error of a model on an image follows a zero-mean Gaussian distribution, namely $\boldsymbol{\epsilon}_{m,n} = \mathbf{y}_n - \mathbf{x}_{m,n} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{m,n})$, where $\boldsymbol{\Sigma}_{m,n} \in \mathbb{R}^{L \times L}$ is the covariance matrix of the Gaussian. Then the observed ground-truth can be considered following a multivariate Gaussian with the mean equal to the prediction, i.e., $\mathbf{y}_n | f_m, \mathbf{x}_n \sim \mathcal{N}(\mathbf{x}_{m,n}, \boldsymbol{\Sigma}_{m,n})$.

We can consider the ensemble problem as the weighted averaging of Gaussian variables and estimate the weights by solving its maximum likelihood estimation. However, solving the sample-wise mixture

Algorithm 1: EnsIR: an ensemble algorithm for image restoration

Input: A small reference dataset $\{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^{N}$ for ensemble weight estimation, test set $\{\hat{\mathbf{X}}_n\}$, M pre-trained models $f_1, ..., f_M$, bin width b, Empty lookup table LUT

Output: Ensemble result $\{\tilde{\mathbf{Y}}_n\}$

Estimation Stage:

Obtain restoration predictions by $\mathbf{x}_{m,n} = \text{flatten}\left(f_m(\mathbf{X}_n)\right), \ \forall m \in \{1, ..., M\}$;

Append restoration predictions and ground-truths into $\mathbf{y}_{1:N}$ and $\mathbf{x}_{m,1:N}$ based on Eq. 2 ;

Define bin set space $\mathbb{B} = \{[0, b), [b, 2b), ..., [(T-1)b, 255]\}$;

for *each bin set* $(B_1, ...B_M) \in \mathbb{B}^M$ **do**

  Compute the partition map $\mathbf{R}_r = \prod_{m=1}^{M} \mathbf{I}_{B_m}(f_m(\mathbf{x}_n))$;

  Partition images and obtain range-wise patches $(\mathbf{y}_{r,1:N}, \mathbf{x}_{r,1,1:N}, ..., \mathbf{x}_{r,M,1:N})$ by Eq. 3 ;

  $(\alpha_{r,1}, ..., \alpha_{r,M}) \leftarrow \mathbf{MPEM}(\mathbf{y}_{r,1:N}, \mathbf{x}_{r,1,1:N}, ..., \mathbf{x}_{r,M,1:N})$;

  Store LUT$[(B_1, ...B_M)] \leftarrow (\alpha_{r,1}, ..., \alpha_{r,M})$;

end

Inference Stage:

for *each test data* $\hat{\mathbf{X}}_n$ **do**

  for *each bin set* $(B_1, ...B_M) \in \mathbb{B}^M$ **do**

    Retrieve $(\alpha_{r,1}, ..., \alpha_{r,M}) \leftarrow$ LUT$[(B_1, ...B_M)]$;

    Partition input as $\tilde{\mathbf{X}}_{r,m,n} \leftarrow \mathbf{R}_r \cdot f_m(\hat{\mathbf{X}}_n)$, where $\mathbf{R}_r = \prod_{m=1}^{M} \mathbf{I}_{B_m}(f_m(\hat{\mathbf{X}}_n))$;

    $\tilde{\mathbf{Y}}_{r,n} \leftarrow \sum_{m=1}^{M} \alpha_{r,m} \tilde{\mathbf{X}}_{r,m,n}$; `/* Inner summation of Eq. 15 */`

  end

  $\tilde{\mathbf{Y}}_n \leftarrow \sum_{r=1}^{T^M} \tilde{\mathbf{Y}}_{r,n}$; `/* Outer summation of Eq. 15 */`

end

of Gaussian is not feasible because the covariance matrices are sample-wise different and thus hard to estimate. Besides, the number of prediction samples is much fewer than feature dimension, resulting in the singularity of the covariance matrices. Please refer to Sec. A.1 in Appendix for details.

In contrast, we alternatively append the reference set into a single sample following Gaussian as

$$\mathbf{y}_{1:N} | f_m, \mathbf{x}_{1:N} \sim \mathcal{N}\left(\mathbf{x}_{m,1:N}, \text{diag}(\mathbf{\Sigma}_{m,1}, ..., \mathbf{\Sigma}_{m,N})\right), \tag{2}$$

where $\mathbf{y}_{1:N} = [\mathbf{y}_1 \ \cdots \ \mathbf{y}_N] \in \mathbb{R}^{NL}$ and $\mathbf{x}_{m,1:N} = [\mathbf{x}_{m,1} \ \cdots \ \mathbf{x}_{m,N}] \in \mathbb{R}^{NL}$ are the concatenation of observed ground-truths and restored samples respectively. Since data samples can be considered following *i.i.d* data distribution $\mathcal{D}$, the variance of the concatenated samples is diagonal.

However, the covariance matrix is still singular due to the imbalance between prediction sample number and feature dimension. Thus, directly mixing the multivariate Gaussian is still infeasible to solve. We thus alternatively categorize pixels into various small bins of mutually exclusive ranges such that the pixels within each range can be considered following a univariate Gaussian distribution according to the central limit theorem. Concretely, we separate the prediction range of models into T bins with each of width b, i.e., $\mathbb{B} = \{[0, b), [b, 2b), ..., [(T-1)b, 255]\}$. The upper bound would be 1 instead of 255 if the value range is within $[0, 1]$. Given a bin $B_m = [(t-1)b, tb) \in \mathbb{B}$ and a pixel of prediction at location i, we define an indicator function $\mathbf{I}_{B_m}(\mathbf{x}_{m,1:N}^{(i)})$ such that it returns 1 if $\mathbf{x}_{m,1:N}^{(i)} \in B_m$ and 0 otherwise. For multiple models, we have M bins to form a bin set $(B_1, ..., B_M) \in \mathbb{B}^M$, and define the mask map as $\mathbf{R}_r = \prod_{m=1}^{M} \mathbf{I}_{B_m}(\mathbf{x}_{m,1:N}) \in \{0, 1\}^{NL}$ where $r = 1, ..., T^M$. It holds $\sum_{r=1}^{T^M} \mathbf{R}_r = \mathbf{1}$ and $\prod_{r=1}^{T^M} \mathbf{R}_r = \mathbf{0}$. For each bin set $(B_1, ..., B_M) \in \mathbb{B}^M$, we can select pixels within the bin set from the original image vectors by

$$\mathbf{y}_{r,1:N} = \mathbf{R}_r \cdot \mathbf{y}_{1:N}, \qquad \mathbf{x}_{r,m,1:N} = \mathbf{R}_r \cdot \mathbf{x}_{m,1:N}, \tag{3}$$

where the operation $\{\cdot\}$ denotes the element-wise product. By the central limit theorem, we assume the nonzero pixels within each bin follow a Gaussian distribution with the mean $\mu_{r,m,1:N}$ and variance $\sigma_{r,m,1:N}$, i.e.,

$$\mathbf{y}_{r,1:N}^{(i)} | f_m, \mathbf{x}_{r,m,1:N} \overset{i.i.d}{\sim} \mathcal{N}(\mu_{r,m,1:N}, \sigma_{r,m,1:N}), \ \forall i \in [1, ..., N_r], \tag{4}$$

where N_r is the number of nonzero pixels in $\mathbf{R}_r$ such that $\sum_{r=1}^{T^M} N_r = NL$, and the values of $\mu_{r,m,1:N}$ and $\sigma_{r,m,1:N}$ can be estimated by the mean and variance of N_r prediction pixels within the current bin set.

The reference set is therefore separated into T^M number of bin sets, and the ground-truth pixels inside each of them form a solvable univariate GMM. We then introduce the latent variable z such that $z = m$ if the pixel $\mathbf{y}_{r,1:N}^{(i)}$ follows the m-th Gaussian by the model f_m. It represents the probability of the pixel belonging to the m-th Gaussian component, which is equivalent to the role of the ensemble weight for the m-th base model. By writing $\alpha_{r,m} = P(z = m)$, we have

$$\mathbf{y}_{r,1:N}^{(i)} = \mathbb{E}_z \left[\mathbf{x}_{r,m,1:N}^{(i)}\right] = \sum_{m=1}^{M} \alpha_{r,m} \cdot \mathbf{x}_{r,m,1:N}^{(i)};\ P(\mathbf{y}_{r,1:N}^{(i)}) = \sum_{m=1}^{M} \alpha_{r,m} P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z = m\right), \tag{5}$$

where $P(\mathbf{y}_{r,1:N}^{(i)} | z = m) = \phi(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N})$ is the density function of Gaussian $\mathcal{N}(\mu_{r,m,1:N}, \sigma_{r,m,1:N})$.

The value of ensemble weights can be estimated by the maximum likelihood estimates of the observed ground-truths, i.e.,

$$\{\alpha_{r,m}\}_{r,m} \in \arg\max P(\mathbf{y}_{1:N}). \tag{6}$$

Because arbitrary two bin sets are mutually exclusive, we can safely split the optimization of maximum likelihood over $\mathbf{y}_{1:N}$ into T^M optimization problems of maximum likelihood over $\mathbf{y}_{r,1:N}$. Each of them is formulated as

$$\alpha_{r,m} \in \arg\max_{\alpha_{r,m}} P(\mathbf{y}_{r,1:N}) = \arg\max_{\alpha_{r,m}} \prod_{i=1}^{N_r} P(\mathbf{y}_{r,1:N}^{(i)}). \tag{7}$$

We have formulated the expression of GMMs for estimating the range-specific ensemble weights.

3.3 Restoration Ensemble via Expectation Maximization and Lookup Table

3.3.1 Weight Estimation via EM Algorithm

For each bin set $(B_1, ..., B_M)$, we estimate ensemble weights by maximizing the log likelihood as

$$\begin{aligned}
\log P(\mathbf{y}_{r,1:N}) &= \log \prod_{i=1}^{N_r} \sum_{m=1}^{M} \alpha_{r,m} \phi\left(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N}\right) \\
&= \sum_{i=1}^{N_r} \log \sum_{m=1}^{M} \alpha_{r,m} \phi\left(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N}\right) \\
&\geq \sum_{m=1}^{M} P\left(z = m \middle| \mathbf{y}_{r,1:N}^{(i)}\right) \log \frac{\alpha_{r,m} \phi(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N})}{P\left(z = m \middle| \mathbf{y}_{r,1:N}^{(i)}\right)},
\end{aligned} \tag{8}$$

We have an E-step to estimate the posterior distribution by

$$\gamma_{r,m,1:N} \leftarrow P\left(z = m \middle| \mathbf{y}_{r,1:N}^{(i)}\right) = \frac{\alpha_{r,m} \phi\left(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N}\right)}{\sum_{m=1}^{M} \alpha_{r,m} \phi\left(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N}\right)}. \tag{9}$$

After that, we have an M-step to obtain the maximum likelihood estimates by

$$\alpha_{r,m} \leftarrow \frac{1}{N_r} \sum_{i=1}^{N_r} \gamma_{r,m,1:N} \tag{10}$$

$$\sigma_{r,m,1:N} \leftarrow \frac{\sum_{i=1}^{N_r} \gamma_{r,m,1:N} \left(\mathbf{y}_{r,m,1:N}^{(i)} - \mu_{r,m,1:N}\right)^2}{\sum_{n=1}^{N_r} \gamma_{r,m,1:N}}. \tag{11}$$

Thanks to the separation of bin sets, we have prior knowledge of the mean and variance of each model, which can be estimated and initialized by

$$\mu_{r,m,1:N} \leftarrow \frac{1}{N_r} \sum_{i=1}^{N_r} \mathbf{x}_{r,m,1:N}^{(i)} \tag{12}$$

$$\sigma_{r,m,1:N} \leftarrow \frac{1}{N_r} \left\| \mathbf{x}_{r,m,1:N} - \mu_{r,m,1:N} \right\|_2 . \tag{13}$$

The complete and detailed derivation of the EM algorithm can be found in Sec. A.3 of Appendix.

3.3.2 Lookup Table and Inference

We store the range-specific weights estimated on the reference set into a LUT with each key of $(B_1, ..., B_M)$. During the inference stage for a test sample $\hat{\mathbf{X}}_n$, we have the prediction of the m-th base model as $\tilde{\mathbf{X}}_{m,n} = f_m(\hat{\mathbf{X}}_n)$. For a bin set $(B_1, ..., B_M)$, we partition input pixels of multiple models into each bin set as

$$\tilde{\mathbf{X}}_{r,m,n} = \mathbf{R}_r \cdot \tilde{\mathbf{X}}_{m,n}, \text{ where } \mathbf{R}_r = \prod_{m=1}^{M} \mathbf{I}_{B_m}(\tilde{\mathbf{X}}_{m,n}). \tag{14}$$

Then we retrieve the estimated range-wise weights $\alpha_{r,m}$ from the LUT based on each key of the bin set and obtain the aggregated ensemble by

$$\tilde{\mathbf{Y}}_n = \sum_{r=1}^{T^M} \tilde{\mathbf{Y}}_{r,n} = \sum_{r=1}^{T^M} \sum_{m=1}^{M} \alpha_{r,m} \tilde{\mathbf{X}}_{r,m,n}. \tag{15}$$

The main algorithm can be found in Algo. 1 and the EM algorithm with known mean and variance priors is shown in Algo. 2.

4 Experiment

4.1 Experimental Settings

Algorithm 2: MPEM: EM algorithm with known Mean Prior

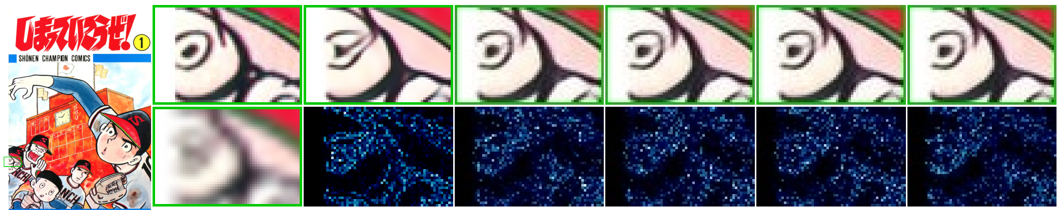
Input: $\mathbf{y}_{r,1:N}, \mathbf{x}_{r,1,1:N}, ..., \mathbf{x}_{r,M,1:N}$
Output: $(\alpha_{r,1}, ..., \alpha_{r,M})$

$N_r \leftarrow$ number of nonzero pixels in $\mathbf{y}_{r,1:N}$;
Initialize $\mu_{r,m,1:N}$ by Eq. 12;
Initialize $\sigma_{r,m,1:N}$ by Eq. 13;
while *not converge* **do**
  for $i \in [1, N_r]$ **do**
    for $m \in [1, M]$ **do**
      Update $\gamma_{r,m,1:N}$ by Eq. 9;
    end
  end
  for $m \in [1, M]$ **do**
    Update $\alpha_{r,m}$ by Eq. 10;
    Update $\sigma_{r,m,1:N}$ by Eq. 11;
  end
end

Benchmarks. We evaluate our ensemble method on 3 image restoration tasks including super-resolution, deblurring, and deraining. For super-resolution, we use Set5 [5], Set14 [83], BSDS100 [47], Urban100 [30] and Manga109 [48] as benchmarks. For deblurring, we use GoPro [51], HIDE [60], RealBlur-J and -R [58]. For deraining, we adopt Rain100H [74], Rain100L [74], Test100[88], Test1200 [86], and Test2800 [25].

Metrics. We use peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [71] to quantitatively evaluate the image restoration quality. Additionally, we compare the average runtime per image in seconds to evaluate the ensemble efficiency. Following prior works [27, 39, 67, 81, 82, 93], PSNR and SSIM are computed on Y channel of the YCbCr color space for image super-resolution and deraining.

Base Models. To evaluate the generalization of ensemble methods against model choices, we employ a wide variety of base models, including CNNs, ViTs, MLPs and Mambas. For image super-resolution, we use SwinIR [39], SRFormer [93], and MambaIR [27]. We choose MPRNet [82], DGUNet [50], and Restormer [81] for deblurring, as well as MPRNet [82], MAXIM [67], and Restormer [81] for deraining.



(a) Image	(b) HR & LR PSNR/SSIM	(c) SwinIR 33.174/0.9674	(d) HGBT [34] 33.919/0.9683	(e) Average 33.873/0.9695	(f) ZZPM [92] 33.874/0.9694	(g) Ours 33.935/0.9696

Figure 1: A visual comparison of ensemble on an image from Manga109 [48] for the task of super-resolution. "HR & LR" means high-resolution and bicubic-upscaled low-resolution images. The second line of (c)-(g) are error maps. Please zoom in for better visual quality.

Baselines. We utilize regression algorithms including bagging [6], AdaBoost [22], random forests (RForest) [7], gradient boosting decision tree (GBDT) [23], histogram gradient boosting decision tree (HGBT) [14, 34] as baselines. Averaging is also a commonly used ensemble baseline. A recent method proposed by team ZZPM in the NTIRE 2023 competition [92] is also included for comparison. Additionally, we adopt RefESR [31] for image super-resolution ensemble.

Implementation Details. We choose the bin width as 32 by default for the balance of efficiency and performance. The EM solver of GMM stops after 1000 iterations or when the change of log likelihood is less than $1e^{-5}$. For cases where N_r is fewer than 100 or the EM solution is undetermined, we use averaging weights by default. The values are reported by taking the average of 4 trials. For the construction of the reference set, we randomly select one image from the training set of DIV2K [4] for super-resolution, while for deblurring and deraining, we sample 10 images from the training sets of GoPro [51] and Rain13K [32], respectively. All the experiments are run in Python on a device with an 8-cores 2.10GHz Intel Xeon Processor and 32G Nvidia Tesla V100. The regression-based ensemble algorithms are implemented based on scikit-learn [8].

Table 1: Ablation study of bin width b on Rain100H [74] with maximum step number 1000. "Runtime" is the average runtime [s].

b	16	32	64	96	128
Runtime	1.2460	0.1709	0.0265	0.0132	0.0059
PSNR	31.745	31.739	31.720	31.713	31.725
SSIM	0.9093	0.9095	0.9094	0.9093	0.9093

Table 2: Ablation study of maximum step number in the EM algorithm on Rain100H [74] with $b = 32$. "Time" is the time of EM algorithm [s].

#step	10	100	500	1000	10000
Time	12.108	28.516	30.409	30.518	30.524
PSNR	31.734	31.738	31.738	31.739	31.739
SSIM	0.9093	0.9094	0.9095	0.9095	0.9095

Table 3: The ensemble results on the task of *image super-resolution*. The categories of "Base", "Regr." and "IR." in the first column mean base models, regression-based ensemble methods, and those ensemble methods designed for image restoration. The best and second best ensemble results are emphasized in **bold** and underlined respectively.

	Datasets	Set5 [5]		Set14 [83]		BSDS100 [47]		Urban100 [30]		Manga109 [48]	
	Metrics	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Base	SwinIR [39]	32.916	0.9044	29.087	0.7950	27.919	0.7487	27.453	0.8254	32.024	0.9260
	SRFormer [93]	32.922	0.9043	29.090	0.7942	27.914	0.7489	27.535	0.8261	32.203	0.9271
	MambaIR [27]	33.045	0.9051	29.159	0.7958	27.967	0.7510	27.775	0.8321	32.308	0.9283
Regr.	Bagging [6]	33.006	0.9050	29.119	0.7950	27.946	0.7498	27.546	0.8273	32.154	0.9270
	AdaBoost [22]	33.072	0.9049	29.175	0.7959	27.975	0.7503	27.786	0.8302	32.457	0.9286
	RForest [7]	33.032	0.9052	29.158	0.7954	27.964	0.7500	27.640	0.8287	32.287	0.9279
	GBDT [23]	33.085	0.9050	29.196	0.7956	27.980	0.7500	27.792	0.8311	32.467	0.9285
	HGBT [34]	33.078	0.9051	29.201	0.7959	27.984	0.7502	27.783	0.8310	32.444	0.9282
IR.	Average	33.097	0.9057	29.202	**0.7964**	27.983	0.7506	27.785	0.8313	32.466	0.9290
	RefESR [31]	33.091	0.9052	29.172	0.7960	27.972	0.7504	27.785	0.8312	32.447	0.9288
	ZZPM [92]	33.094	0.9057	29.203	0.7963	27.981	0.7506	27.786	0.8313	32.467	0.9290
	EnsIR (Ours)	**33.103**	**0.9058**	**29.205**	**0.7964**	**27.984**	**0.7507**	**27.795**	**0.8315**	**32.468**	**0.9291**

Table 4: The ensemble results on the task of *image deblurring*. The categories of "Base", "Regr." and "IR." in the first column mean base models, regression-based ensemble methods, and those ensemble methods designed for image restoration. The best and second best ensemble results are emphasized in **bold** and <u>underlined</u> respectively.

	Datasets	GoPro [51]		HIDE [60]		RealBlur-R [58]		RealBlur-J [58]	
	Metrics	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Base	MPRNet [82]	32.658	0.9362	30.962	0.9188	33.914	0.9425	26.515	0.8240
	Restormer [81]	32.918	0.9398	31.221	0.9226	33.984	0.9463	26.626	0.8274
	DGUNet [50]	33.173	0.9423	31.404	0.9257	33.990	0.9430	26.583	0.8261
Regr.	Bagging [6]	33.194	0.9418	31.437	0.9250	34.033	0.9456	26.641	0.8277
	AdaBoost [22]	33.205	0.9412	31.449	0.9251	34.035	0.9455	26.652	0.8280
	RForest [7]	33.173	0.9416	31.439	0.9247	34.039	0.9457	26.647	0.8280
	GBDT [23]	33.311	0.9418	31.568	0.9256	34.052	0.9465	26.684	0.8285
	HGBT [34]	33.323	0.9427	<u>31.583</u>	0.9267	33.986	0.9436	26.684	0.8296
IR.	Average	33.330	<u>0.9436</u>	31.579	<u>0.9277</u>	<u>34.090</u>	<u>0.9471</u>	<u>26.689</u>	**0.8309**
	ZZPM [92]	<u>33.332</u>	<u>0.9436</u>	31.580	<u>0.9277</u>	34.057	0.9468	26.688	<u>0.8308</u>
	EnsIR (Ours)	**33.345**	**0.9438**	**31.590**	**0.9278**	**34.089**	**0.9472**	**26.690**	**0.8309**

Table 5: The ensemble results on the task of *image deraining*. The categories of "Base", "Regr." and "IR." in the first column mean base models, regression-based ensemble methods, and those ensemble methods designed for image restoration. The best and second best ensemble results are emphasized in **bold** and <u>underlined</u> respectively.

	Datasets	Rain100H [74]		Rain100L [74]		Test100 [88]		Test1200 [86]		Test2800 [25]	
	Metrics	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Base	MPRNet [82]	30.428	0.8905	36.463	0.9657	30.292	0.8983	32.944	0.9175	33.667	0.9389
	MAXIM [67]	30.838	0.9043	38.152	0.9782	31.194	0.9239	32.401	0.9240	33.837	0.9438
	Restormer [81]	31.477	0.9054	39.080	0.9785	32.025	0.9237	33.219	0.9270	34.211	0.9449
Regr.	Bagging [6]	31.461	0.9001	39.060	0.9782	31.865	0.9107	33.115	0.9152	34.216	0.9446
	AdaBoost [22]	31.472	0.9006	39.067	0.9782	31.866	0.9112	33.117	0.9153	34.221	0.9443
	RForest [7]	31.492	0.9012	39.089	<u>0.9784</u>	31.900	0.9127	33.147	0.9169	34.224	0.9447
	GBDT [23]	31.581	0.9058	39.044	0.9778	<u>32.001</u>	0.9236	33.276	0.9274	34.211	0.9446
	HGBT [34]	<u>31.698</u>	0.9075	<u>39.115</u>	<u>0.9784</u>	31.988	<u>0.9241</u>	33.305	0.9282	34.229	<u>0.9450</u>
IR.	Average	31.681	<u>0.9091</u>	38.675	0.9770	31.626	0.9225	33.427	<u>0.9286</u>	34.214	0.9449
	ZZPM [92]	31.689	<u>0.9091</u>	38.725	0.9771	31.642	0.9227	<u>33.434</u>	<u>0.9286</u>	<u>34.231</u>	<u>0.9450</u>
	EnsIR (Ours)	**31.739**	**0.9095**	**39.216**	**0.9792**	**32.064**	**0.9258**	**33.445**	**0.9289**	**34.245**	**0.9451**

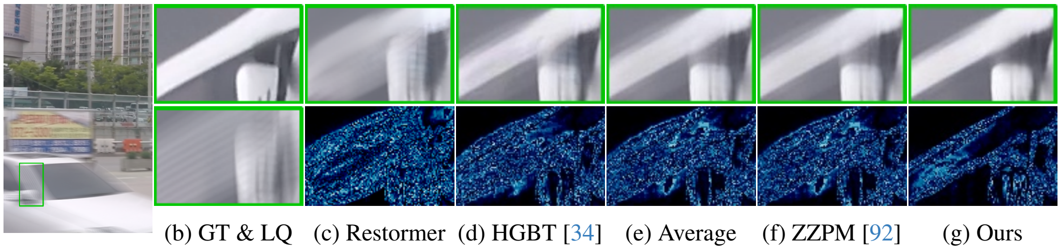
4.2 Experimental Results

4.2.1 Ablation Study

We conduct ablation studies on the bin width b and the maximum step number of the EM algorithm on the benchmark of Rain100H [74]. The results are shown in Tab. 1 and Tab. 2 respectively. We can observe that the bin width involves a trade-off between ensemble accuracy and the efficiency . A small b generates better ensemble results but makes ensemble inference slower. In the following experiments, we choose the bin width $b = 32$ by default for the balance of efficiency and performance. On the other hand, a large maximum step number does not necessarily yield better ensemble result. Therefore, we choose 1000 as the maximum step number.

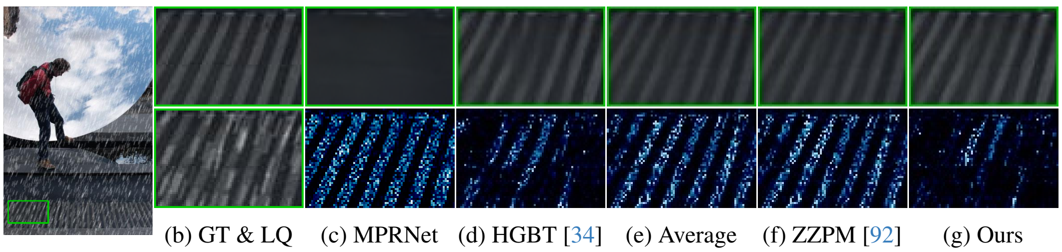
4.2.2 Quantitative Results

We present quantitative comparisons with existing ensemble methods in Tab. 3 for super-resolution, Tab. 4 for deblurring, and Tab. 5 for deraining. Our method generally outperforms all existing methods across the 3 image restoration tasks and 14 benchmarks. Note that Average and ZZPM [92] generally perform better than regression-based ensemble methods. However, in cases where one of the base



(a) Image | (b) GT & LQ PSNR/SSIM | (c) Restormer 29.980/0.9377 | (d) HGBT [34] 32.110/0.9481 | (e) Average 32.177/0.9479 | (f) ZZPM [92] 32.170/0.9475 | (g) Ours 32.584/0.9486

Figure 2: A visual comparison of ensemble on an image from GoPro [51] for the task of image deblurring. "GT & LQ" means ground-truth and low quality blurry images. The second line of (c)-(g) are error maps. Please zoom in for better visual quality.



(a) Image | (b) GT & LQ PSNR/SSIM | (c) MPRNet 34.008/0.9359 | (d) HGBT [34] 37.052/0.9615 | (e) Average 36.775/0.9603 | (f) ZZPM [92] 36.777/0.9601 | (g) Ours 37.372/0.9639

Figure 3: A visual comparison of ensemble on an image from Test100 [88] for the task of image deraining. "GT & LQ" means ground-truth and low quality rainy images. The second line of (c)-(g) are error maps. Please zoom in for better visual quality.

models significantly underperforms compared to the others, such as MPRNet [82] on Rain100L [74] and Test100 [88], these regression methods outperform Average and ZZPM [92]. In contrast, our method, which learns per-value weights, can recognize performance biases and alleviate such issue. ZZPM [92] performs comparably to Average in our experiments rather than outperforming it, because the base models are not always equally good and one model may be consistently better than the others. Thus, weights negatively proportional to the mean squared error may exaggerate deviation from optimal prediction. In contrast, our method consistently performs well for all cases.

4.2.3 Qualitative Results

We also provide qualitative visual comparisons in Fig. 1, 2 and 3. In Fig. 1, the base model SwinIR [39] mistakenly upscales the character's eye. While existing ensemble algorithms partially alleviate this mistake, they cannot fully discard the hallucinated line inside the eye. In contrast, our method with the bin width $b = 32$ that learns fine-grained range-wise weights successfully recovers the pattern. In Fig. 2, only our method can effectively obtain a better ensemble with sharp edges and accurate colors. In Fig. 3, we can observe that MPRNet [82] removes all stripe patterns on the ground together with rain streaks. The conventional weighted summation yields a dimmer ensemble result, and the HGBT method fails to learn accurate weight distributions, resulting in an unsmooth result. In contrast, ours alleviates the issue. More visual comparisons are provided in Fig. 7-18 in Appendix.

4.2.4 Extensions

Efficiency. The efficiency of ensemble methods is compared by measuring the average runtime on Rain100H [74], as shown in Tab. 6. Although our method is slower than Average and ZZPM [92],

Table 6: The average runtime per image in seconds of the ensemble methods on Rain100H [74].

Method	Bagging [6]	AdaBoost [22]	RForest [7]	GBDT [23]	HGBT [34]	Average	ZZPM [92]	Ours
Runtime	1.0070	1.1827	9.8598	1.2781	0.1773	0.0003	0.0021	0.1709

it is much faster than all the regression-based methods. By slightly sacrificing performance with $b = 64$, it can achieve real-time inference, as indicated in Tab. 1.

Visualization. We also present the visualization examples of ensemble weights, image features and pixel distributions in Fig. 4-6 in Appendix. Due to page limit, please refer to Sec. B of Appendix.

Limitation and Future Work. The trade-off between the runtime and performance has not been solved yet. Achieving real-time ensemble would lead to performance degradation. The issue could be resolved by GPU vectorization acceleration and distributed computing. Additionally, if all base models fail, ensemble methods cannot generate better result. We leave them in the future work.

5 Conclusion

In this paper, we propose an ensemble algorithm for image restoration based on GMMs. We partition the pixels of predictions and ground-truths into separate bins of exclusive ranges and formulate the ensemble problem using GMMs over each bin. The GMMs are solved on a reference set, and the estimated ensemble weights are stored in a lookup table for the ensemble inference on the test set. Our algorithm outperforms regression-based ensemble methods as well as commonly used averaging strategies on 14 benchmarks across 3 image restoration tasks, including super-resolution, deblurring and deraining. It is training-free, model-agnostic, and thus suitable for plug-and-play usage.

Acknowledgement

This work has been supported in part by National Natural Science Foundation of China (No. 62322216, 62025604, 62172409), in part by Shenzhen Science and Technology Program (Grant No. JCYJ20220818102012025, KQTD20221101093559018, RCYX20221008092849068).

References

[1] Abdelrahman Abdelhamed, Mahmoud Afifi, Radu Timofte, and Michael S Brown. Ntire 2020 challenge on real image denoising: Dataset, methods and results. In *CVPRW*, 2020. 2, 3

[2] Hazique Aetesam and Suman Kumar Maji. Noise dependent training for deep parallel ensemble denoising in magnetic resonance images. *Biomedical Signal Processing and Control*, 2021. 3

[3] Subhash Chand Agrawal and Anand Singh Jalal. Distortion-free image dehazing by superpixels and ensemble neural network. *The Visual Computer*, 2022. 3

[4] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *CVPRW*, 2017. 7

[5] Marco Bevilacqua, Aline Roumy, Christine M. Guillemot, and Marie-Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *BMVC*, 2012. 6, 7

[6] Leo Breiman. Bagging predictors. *Machine learning*, 1996. 2, 7, 8, 9

[7] Leo Breiman. Random forests. *Machine learning*, 2001. 2, 7, 8, 9

[8] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013. 7

[9] Chung Chan, Jian Zhou, Li Yang, Wenyuan Qi, and Evren Asma. Noise to noise ensemble learning for pet image denoising. In *IEEE NSS/MIC*, 2019. 3

[10] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *ECCV*, 2022. 1, 2

[11] Mingqin Chen, Yuhui Quan, Yong Xu, and Hui Ji. Self-supervised blind image deconvolution via deep generative ensemble learning. *IEEE TCSVT*, 2022. 3

[12] Rui Chen, Huizhu Jia, Xiaodong Xie, and Gao Wen. A structure-preserving image restoration method with high-level ensemble constraints. In *VCIP*, 2016.

[13] Ruoyu Chen, Hua Zhang, Siyuan Liang, Jingzhi Li, and Xiaochun Cao. Less is more: Fewer interpretable region via submodular subset selection. *ICLR*, 2024. 3

[14] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *KDD*, 2016. 2, 7

[15] Zheng Chen, Zongwei Wu, Eduard Zamfir, Kai Zhang, Yulun Zhang, Radu Timofte, Xiaokang Yang, Hongyuan Yu, Cheng Wan, Yuxin Hong, et al. Ntire 2024 challenge on image super-resolution ($\times 4$): Methods and results. *arXiv preprint arXiv:2404.09790*, 2024. 2, 3

[16] Zhibo Chen, Jianxin Lin, Tiankuang Zhou, and Feng Wu. Sequential gating ensemble network for noise robust multiscale face restoration. *IEEE TCYB*, 2019. 3

[17] Subhrajit Dey, Rajdeep Bhattacharya, Friedhelm Schwenker, and Ram Sarkar. Median filter aided cnn based image denoising: an ensemble approach. *Algorithms*, 2021. 3

[18] Thomas G Dietterich et al. Ensemble learning. *The handbook of brain theory and neural networks*, 2002. 2

[19] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, 2014. 3

[20] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 2020. 2

[21] Masud An Nur Islam Fahim, Nazmus Saqib, Shafkat Khan Siam, and Ho Yub Jung. Denoising single images by feature ensemble revisited. *Sensors*, 2022. 3

[22] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 1997. 7, 8, 9

[23] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 2001. 7, 8, 9

[24] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 2002. 2

[25] Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. Removing rain from single images via a deep detail network. In *CVPR*, 2017. 6, 8

[26] Mudasir A Ganaie, Minghui Hu, Ashwani Kumar Malik, Muhammad Tanveer, and Ponnuthurai Suganthan. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 2022. 2

[27] Hang Guo, Jinmin Li, Tao Dai, Zhihao Ouyang, Xudong Ren, and Shu-Tao Xia. Mambair: A simple baseline for image restoration with state-space model. *arXiv preprint arXiv:2402.15648*, 2024. 1, 6, 7, 22

[28] Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. Multi-class adaboost. *Statistics and its Interface*, 2009. 2

[29] Trung Hoang, Haichuan Zhang, Amirsaeed Yazdani, and Vishal Monga. Transer: Hybrid model and ensemble-based sequential learning for non-homogenous dehazing. In *CVPR*, 2023. 3

[30] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPR*, 2015. 6, 7, 22

[31] Junjun Jiang, Yi Yu, Zheng Wang, Suhua Tang, Ruimin Hu, and Jiayi Ma. Ensemble super-resolution with a reference dataset. *IEEE TCYB*, 2019. 2, 3, 7

[32] Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. In *CVPR*, 2020. 7

[33] Zhiying Jiang, Shuzhou Yang, Jinyuan Liu, Xin Fan, and Risheng Liu. Multi-scale synergism ensemble progressive and contrastive investigation for image restoration. *IEEE TIM*, 2023. 3

[34] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *NeurIPS*, 2017. 2, 7, 8, 9, 18, 22, 23, 24

[35] Chao Li, Dongliang He, Xiao Liu, Yukang Ding, and Shilei Wen. Adapting image super-resolution state-of-the-arts and learning multi-model ensemble for video super-resolution. In *CVPRW*, 2019. 2, 3

[36] Jinyang Li and Zhijing Liu. Ensemble dictionary learning for single image deblurring via low-rank regularization. *Sensors*, 2019.

[37] Yuenan Li, Yuhang Liu, Qixin Yan, and Kuangshi Zhang. Deep dehazing network with latent ensembling architecture and adversarial learning. *IEEE TIP*, 2020.

[38] Yufeng Li, Jiyang Lu, Zhentao Fan, and Xiang Chen. Learning an ensemble dehazing network for visible remote sensing images. *Journal of Applied Remote Sensing*, 2023. 3

[39] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCV*, 2021. 1, 3, 6, 7, 9, 19, 22

[40] Renjie Liao, Xin Tao, Ruiyu Li, Ziyang Ma, and Jiaya Jia. Video super-resolution via deep draft-ensemble learning. In *ICCV*, 2015. 2, 3

[41] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPRW*, 2017. 2

[42] Jianxin Lin, Tiankuang Zhou, and Zhibo Chen. Multi-scale face restoration with sequential gating ensemble network. In *AAAI*, 2018. 3

[43] Pengju Liu, Hongzhi Zhang, Jinghui Wang, Yuzhi Wang, Dongwei Ren, and Wangmeng Zuo. Robust deep ensemble method for real-world image denoising. *Neurocomputing*, 2022.

[44] Yingnan Liu and Randy Clinton Paffenroth. Ensemble cnn in transform domains for image super-resolution from small data sets. In *ICMLA*, 2020. 2

[45] Yu Luo, Menghua Wu, Qingdong Huang, Jian Zhu, Jie Ling, and Bin Sheng. Joint feedback and recurrent deraining network with ensemble learning. *The Visual Computer*, 2022.

[46] Qing Lyu, Hongming Shan, and Ge Wang. Mri super-resolution with ensemble learning and complementary priors. *IEEE TCI*, 2020. 2, 3

[47] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001. 6, 7

[48] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia tools and applications*, 2017. 6, 7

[49] Joao Mendes-Moreira, Carlos Soares, Alípio Mário Jorge, and Jorge Freire De Sousa. Ensemble approaches for regression: A survey. *ACM Computing Surveys*, 2012. 2

[50] Chong Mou, Qian Wang, and Jian Zhang. Deep generalized unfolding networks for image restoration. In *CVPR*, 2022. 6, 8, 18, 19, 20, 21

[51] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2017. 6, 7, 8, 9

[52] Seungjun Nah, Sanghyun Son, Suyoung Lee, Radu Timofte, Kyoung Mu Lee, Liangyu Chen, Jie Zhang, Xin Lu, Xiaojie Chu, Chengpeng Chen, et al. Ntire 2021 challenge on image deblurring. In *CVPR*, 2021. 2, 3

[53] Luciano Oliveira, Urbano Nunes, and Paulo Peixoto. On exploration of classifier ensemble synergism in pedestrian detection. *IEEE TITS*, 2009. 2

[54] Zhihong Pan, Baopu Li, Teng Xi, Yanwen Fan, Gang Zhang, Jingtuo Liu, Junyu Han, and Errui Ding. Real image super resolution via heterogeneous model ensemble using gp-nas. In *ECCVW*, 2020. 2, 3

[55] Long Peng, Aiwen Jiang, Haoran Wei, Bo Liu, and Mingwen Wang. Ensemble single image deraining network via progressive structural boosting constraints. *Signal Process Image Commun.*, 2021. 3

[56] Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang. Single image dehazing via multi-scale convolutional neural networks. In *ECCV*, 2016. 3

[57] Ye Ren, Le Zhang, and Ponnuthurai N Suganthan. Ensemble classification and regression-recent developments, applications and future directions. *IEEE CIM*, 2016. 2

[58] Jaesung Rim, Haeyun Lee, Jucheol Won, and Sunghyun Cho. Real-world blur dataset for learning and benchmarking deblurring algorithms. In *ECCV*, 2020. 6, 8, 23

[59] Ali Shahsavari, Sima Ranjbari, and Toktam Khatibi. Proposing a novel cascade ensemble super resolution generative adversarial network (cesr-gan) method for the reconstruction of super-resolution skin lesion images. *Informatics in Medicine Unlocked*, 2021. 2, 3

[60] Ziyi Shen, Wenguan Wang, Xiankai Lu, Jianbing Shen, Haibin Ling, Tingfa Xu, and Ling Shao. Human-aware motion deblurring. In *ICCV*, 2019. 6, 8, 18, 19, 20, 21, 23

[61] Yu Su, Shiguang Shan, Xilin Chen, and Wen Gao. Hierarchical ensemble of global and local classifiers for face recognition. *IEEE TIP*, 2009. 2

[62] Shangquan Sun, Wenqi Ren, Xinwei Gao, Rui Wang, and Xiaochun Cao. Restoring images in adverse weather conditions via histogram transformer. In *ECCV*, 2024. 3

[63] Shangquan Sun, Wenqi Ren, Jingzhi Li, Rui Wang, and Xiaochun Cao. Logit standardization in knowledge distillation. In *CVPR*, 2024. 3

[64] Shangquan Sun, Wenqi Ren, Jingzhi Li, Kaihao Zhang, Meiyu Liang, and Xiaochun Cao. Event-aware video deraining via multi-patch progressive learning. *IEEE TIP*, 2023. 3

[65] Shangquan Sun, Wenqi Ren, Tao Wang, and Xiaochun Cao. Rethinking image restoration for object detection. *NeurIPS*, 2022. 2

[66] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *CVPR*, 2016. 2, 3

[67] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxim: Multi-axis mlp for image processing. In *CVPR*, 2022. 3, 6, 8, 19, 23, 24

[68] Florin-Alexandru Vasluianu, Tim Seizinger, Radu Timofte, Shuhao Cui, Junshi Huang, Shuman Tian, Mingyuan Fan, Jiaqi Zhang, Li Zhu, Xiaoming Wei, et al. Ntire 2023 image shadow removal challenge report. In *CVPR*, 2023. 2, 3

[69] Jixiao Wang, Chaofeng Li, and Shoukun Xu. An ensemble multi-scale residual attention network (emra-net) for image dehazing. *Multimedia Tools and Applications*, 2021. 3

[70] Lingfeng Wang, Zehao Huang, Yongchao Gong, and Chunhong Pan. Ensemble based deep networks for image super-resolution. *Pattern recognition*, 2017. 2, 3

[71] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 2004. 6

[72] WenBo Wu, Yun Pan, Na Su, JingTao Wang, ShaoChuan Wu, ZeKun Xu, YouJian Yu, and YaPeng Liu. Multi-scale network for single image deblurring based on ensemble learning module. *Multimedia Tools and Applications*, 2024. 3

[73] Li Xu, Jimmy S Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. *NeurIPS*, 2014. 3

[74] Wenhan Yang, Robby T Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *CVPR*, 2017. 6, 7, 8, 9, 23, 24

[75] Xuhui Yang, Yong Xu, Yuhui Quan, and Hui Ji. Image denoising via sequential ensemble learning. *IEEE TIP*, 2020. 3

[76] Chenyu You, Guang Li, Yi Zhang, Xiaoliu Zhang, Hongming Shan, Mengzhou Li, Shenghong Ju, Zhen Zhao, Zhuiyang Zhang, Wenxiang Cong, et al. Ct super-resolution gan constrained by the identical, residual, and cycle learning ensemble (gan-circle). *IEEE TMI*, 2019. 2, 3

[77] Dong-Jun Yu, Jun Hu, Jing Yang, Hong-Bin Shen, Jinhui Tang, and Jing-Yu Yang. Designing template-free predictor for targeting protein-ligand binding sites with classifier ensemble and spatial clustering. *IEEE/ACM TCBB*, 2013. 2

[78] Guoxian Yu, Huzefa Rangwala, Carlotta Domeniconi, Guoji Zhang, and Zhiwen Yu. Protein function prediction using multilabel ensemble classification. *IEEE/ACM TCBB*, 2013. 2

[79] Mingzhao Yu, Venkateswararao Cherukuri, Tiantong Guo, and Vishal Monga. Ensemble dehazing networks for non-homogeneous haze. In *CVPRW*, 2020. 3

[80] Yankun Yu, Huan Liu, Minghan Fu, Jun Chen, Xiyao Wang, and Keyan Wang. A two-branch neural network for non-homogeneous dehazing via ensemble learning. In *CVPR*, 2021. 3

[81] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022. 1, 3, 6, 8, 19, 20, 21, 23, 24

[82] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-stage progressive image restoration. In *CVPR*, 2021. 1, 3, 6, 8, 9, 18, 19, 20, 21, 23, 24

[83] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*, 2012. 6, 7, 22

[84] Bailing Zhang. Reliable classification of vehicle types based on cascade classifier ensembles. *IEEE TITS*, 2012. 2

[85] Dan Zhang, Yingbing Xu, Liyan Ma, Xiaowei Li, Xiangyu Zhang, Yan Peng, and Yaoran Chen. Srenet: Structure recovery ensemble network for single image deraining. *Applied Intelligence*, 2024. 3

[86] He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *CVPR*, 2018. 6, 8, 24

[87] He Zhang, Vishwanath Sindagi, and Vishal M Patel. Multi-scale single image dehazing using perceptual pyramid deep network. In *CVPRW*, 2018. 3

[88] He Zhang, Vishwanath Sindagi, and Vishal M Patel. Image de-raining using a conditional generative adversarial network. *IEEE TCSVT*, 2019. 6, 8, 9

[89] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *CVPR*, 2019. 3

[90] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *CVPR*, 2017. 3

[91] Ping Zhang, Tien D Bui, and Ching Y Suen. A novel cascade ensemble classifier system with a high recognition performance on handwritten digits. *Pattern Recognition*, 2007. 2

[92] Yulun Zhang, Kai Zhang, Zheng Chen, Yawei Li, Radu Timofte, Junpei Zhang, Kexin Zhang, Rui Peng, Yanbiao Ma, Licheng Jia, et al. Ntire 2023 challenge on image super-resolution (x4): Methods and results. In *CVPR*, 2023. 2, 3, 7, 8, 9, 18, 19, 22, 23, 24

[93] Yupeng Zhou, Zhen Li, Chun-Le Guo, Song Bai, Ming-Ming Cheng, and Qibin Hou. Srformer: Permuted self-attention for single image super-resolution. In *ICCV*, 2023. 1, 6, 7, 22

Appendix

A Derivation

We first list the theorems necessary for the subsequent derivations

Theorem A.1. *Suppose* $X \sim \mathcal{N}(\mu_X, \sigma_X)$ *and* $Y \sim \mathcal{N}(\mu_Y, \sigma_Y)$ *are two independent random variables following univariate Gaussian distribution. If we have another random variable defined as* $Z = X + Y$*, then the variable follows* $Z \sim \mathcal{N}(\mu_X + \mu_Y, \sigma_X + \sigma_Y)$*.*

Theorem A.2. *Suppose* $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X)$ *and* $\mathbf{Y} \sim \mathcal{N}(\boldsymbol{\mu}_Y, \boldsymbol{\Sigma}_Y)$ *are two independent random variables following multivariate Gaussian distribution. If we have another random variable defined as* $\mathbf{Z} = \mathbf{X} + \mathbf{Y}$*, then the variable follows* $\mathbf{Z} \sim \mathcal{N}(\boldsymbol{\mu}_X + \boldsymbol{\mu}_Y, \boldsymbol{\Sigma}_X + \boldsymbol{\Sigma}_Y)$*.*

Theorem A.3. *Suppose* $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *is a random variable following multivariate Gaussian distribution, where* $\boldsymbol{\mu} \in \mathbb{R}^L$*. Given a constant vector* $\mathbf{a} \in \mathbb{R}^L$*, we have the variable* $\mathbf{Y} = \mathbf{a} \cdot \mathbf{X} \sim \mathcal{N}(\mathbf{a} \cdot \boldsymbol{\mu}, \mathbf{a}^\top \mathbf{a} \cdot \boldsymbol{\Sigma})$*.*

In Sec. A.1 and A.2, it is shown the mixture of multivariate Gaussian is difficult to solve with limited samples while large feature dimension. In Sec. A.3, we derive the updating of GMMs for the pixels within a bin set. Its convergence is proved in Sec. A.4.

A.1 Sample-wise Maximum Likelihood Estimation of multivariate Gaussian Ensemble

If we directly consider the ensemble problem by the sample-wise weighted summation of multivariate Gaussian with the ensemble weights $\beta_{n,m}$ such that $\sum_{m=1}^{M} \beta_{n,m} = 1$, i.e.,

$$\mathbf{y}_n | f_1, ..., f_M, \mathbf{x}_n \sim \mathcal{N}\left(\sum_{m=1}^{M} \beta_{n,m}\mathbf{x}_{m,n}, \sum_{m=1}^{M} \beta_{n,m}^2 \boldsymbol{\Sigma}_{m,n}\right). \tag{16}$$

The log likelihood is therefore

$$\begin{aligned} &\log P\left(\mathbf{y}_1, ..., \mathbf{y}_N | f_1, ..., f_M, \mathbf{x}_1, ..., \mathbf{x}_N\right) \\ =&\log \prod_{n=1}^{N} P\left(\mathbf{y}_n | f_1, ..., f_M, \mathbf{x}_n\right) \\ =&\sum_{n=1}^{N} \log \phi\left(\mathbf{y}_n; \sum_{m=1}^{M} \beta_{n,m}\mathbf{x}_{m,n}, \sum_{m=1}^{M} \beta_{n,m}^2 \boldsymbol{\Sigma}_{m,n}\right) \\ =&\sum_{n=1}^{N} -\frac{1}{2}\log\left(2\pi \sum_{m=1}^{M} \beta_{n,m}^2 \boldsymbol{\Sigma}_{m,n}\right) - \frac{\left(\mathbf{y}_n - \sum_{m=1}^{M} \beta_{n,m}\mathbf{x}_{m,n}\right)^2}{2\sum_{m=1}^{M} \beta_{n,m}^2 \boldsymbol{\Sigma}_{m,n}}. \end{aligned} \tag{17}$$

The weights can be found via computing the maximum likelihood estimates by $\frac{\partial L}{\partial \beta_{n,m}} = 0$. However, the derivative is complicated, sample-wisely different and related to both unknown and sample-wise covariance matrices. Directly making sample-wise ensemble is thereby difficult to solve.

Besides, if we use multivariate GMMs with the EM algorithm, the estimation of covariance requires the computation of its inverse. However, we only have M observed samples while L features with $L \gg M$ for each prediction sample. The covariance matrix is singular and thus multivariate GMMs cannot be solved.

A.2 Set-level Maximum Likelihood Estimation of multivariate Gaussian Ensemble

If we directly consider the ensemble problem by the weighted summation of multivariate Gaussian over the reference set with the ensemble weights $\beta_{n,m}$ such that $\sum_{m=1}^{M} \beta_m = 1$, i.e.,

$$\mathbf{y}_{1:N} | f_1, ..., f_M, \mathbf{x}_{1:N} \sim \mathcal{N}\left(\sum_{m=1}^{M} \beta_m \mathbf{x}_{m,1:N}, \sum_{m=1}^{M} \beta_m^2 \operatorname{diag}(\boldsymbol{\Sigma}_{m,1}, ..., \boldsymbol{\Sigma}_{m,N})\right) \tag{18}$$

The log likelihood is therefore

$$
\begin{aligned}
&\log P(\mathbf{y}_{1:N}|f_1,...,f_M,\mathbf{x}_{1:N}) \\
=& -\frac{1}{2}\log\left(2\pi\sum_{m=1}^{M}\beta_m^2\,\mathrm{diag}(\mathbf{\Sigma}_{m,1},...,\mathbf{\Sigma}_{m,N})\right)-\frac{\left(\mathbf{y}_{1:N}-\sum_{m=1}^{M}\beta_m\mathbf{x}_{m,1:N}\right)^2}{2\sum_{m=1}^{M}\beta_m^2\,\mathrm{diag}(\mathbf{\Sigma}_{m,1},...,\mathbf{\Sigma}_{m,N})}
\end{aligned}
\tag{19}
$$

We can compute the optimal maximum likelihood estimate by making $\frac{\partial L}{\partial \beta_m}=0$. Because we only have M observed samples while NL features with $NL \gg M$ for each prediction sample. The unknown covariance matrix still cannot be estimated by multivariate GMMs with the EM algorithm,. The estimation of covariance requires the computation of its inverse. The covariance matrix is singular and thus the multivariate GMMs cannot be solved.

A.3 Derivation of GMMs in a bin set

For each bin set, we have converted the problem into the format of GMMs, i.e.,

$$
\begin{aligned}
P(\mathbf{y}_{r,1:N}^{(i)}) &= \sum_{m=1}^{M}P(z=m)P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\, z=m\right) \\
&= \sum_{m=1}^{M}\alpha_{r,m}\phi\left(\mathbf{y}_{r,1:N}^{(i)};\mu_{r,m,1:N},\sigma_{r,m,1:N}\right),
\end{aligned}
\tag{20}
$$

where $z\in\{1,...,M\}$ is the latent variable that represents the probability of the m-th mixture, and $P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\, z=m\right)$ is the m-th mixture probability component. We use $\alpha_{r,m}=P(z=m)$ to denote the mixture proportion or the probability that $\mathbf{y}_{r,1:N}^{(i)}$ belongs to the m-th mixture component. We assume the pixels within the bin follow the Gaussian distribution based on the central limit theorem, i.e., $\mathbf{y}_{r,1:N}^{(i)} \overset{i.i.d}{\sim} \mathcal{N}(\mu_{r,m,1:N},\sigma_{r,m,1:N})$ and $P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\, z=m\right)=\phi\left(\mathbf{y}_{r,1:N}^{(i)};\mu_{r,m,1:N},\sigma_{r,m,1:N}\right)$. The mean and variance of the bin are denoted as $\mu_{r,m,1:N}$ and $\sigma_{r,m,1:N}$.

Suppose we have N_r data samples in the bin, then for the N_r observations, we have its joint probability

$$
\begin{aligned}
P(\mathbf{y}_{r,1:N}) = P\left(\mathbf{y}_{r,1:N}^{(1)},...,\mathbf{y}_{r,1:N}^{(N_r)}\right) &= \prod_{i=1}^{N_r}P(\mathbf{y}_{r,1:N}^{(i)}) \\
&= \prod_{i=1}^{N_r}\sum_{m=1}^{M}\alpha_{r,m}\phi\left(\mathbf{y}_{r,1:N}^{(i)};\mu_{r,m,1:N},\sigma_{r,m,1:N}\right)
\end{aligned}
\tag{21}
$$

Different from conventional GMMs, we have the observation prior that

$$
\mu_{r,m,1:N}=\frac{1}{N_r}\sum_{i=1}^{N_r}\mathbf{x}_{r,m,1:N}^{(i)}
\tag{22}
$$

and thus we want to find the maximum likelihood estimates of $\alpha_{r,m}$. The initial variance, $\sigma_{r,m,1:N}$ can also be estimated by

$$
\sigma_{r,m,1:N}=\frac{1}{N_r}\sum_{i=1}^{N_r}\left\|\mathbf{x}_{r,m,1:N}^{(i)}-\mu_{r,m,1:N}\right\|_2
\tag{23}
$$

Theorem A.4. *(Jensen's Inequality) Given a convex funnction f, we have $f(\mathbb{E}[X])\geq\mathbb{E}[f(X)]$*

The log likelihood of the GMMs is

$$
\begin{aligned}
\log P(\mathbf{y}_{r,1:N}) &= \log \prod_{i=1}^{N_r} \sum_{m=1}^{M} P(z=m) P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right) \\
&= \sum_{i=1}^{N_r} \log \sum_{m=1}^{M} P(z=m) P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right) \\
&= \sum_{i=1}^{N_r} \log \sum_{m=1}^{M} P\left(z=m \middle| \mathbf{y}_{r,1:N}^{(i)}\right) \frac{P(z=m) P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right)}{P\left(z=m \middle| \mathbf{y}_{r,1:N}^{(i)}\right)} \\
&= \sum_{i=1}^{N_r} \log \mathbb{E}_{z|\mathbf{y}_{r,1:N}^{(i)}} \left[\frac{P(z=m) P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right)}{P\left(z=m \middle| \mathbf{y}_{r,1:N}^{(i)}\right)} \right] \\
&\geq \sum_{i=1}^{N_r} \sum_{m=1}^{M} P\left(z=m \middle| \mathbf{y}_{r,1:N}^{(i)}\right) \log \frac{P(z=m) P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right)}{P\left(z=m | \mathbf{y}_{r,1:N}^{(i)}\right)},
\end{aligned}
\tag{24}
$$

Plug the expression of $P(z=m)$ and $P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| z=m\right)$ in and we get

$$
\log P(\mathbf{y}_{r,1:N}) \geq \sum_{i=1}^{N_r} \sum_{m=1}^{M} P(z=m|\mathbf{y}_{r,1:N}^{(i)}) \log \frac{\alpha_{r,m} \phi(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N})}{P(z=m|\mathbf{y}_{r,1:N}^{(i)})}, \tag{25}
$$

where the inequality is based on the Jensen's Inequality.

The problem can be effectively solved by the EM algorithms. We have the an E-step to estimate posterior distribution of z by

$$
\begin{aligned}
\gamma_{r,m,1:N} &\leftarrow P\left(z=m|\mathbf{y}_{r,1:N}^{(i)}\right) \\
&= \frac{P\left(\mathbf{y}_{r,1:N}^{(i)}|z=m\right) P(z=m)}{P\left(\mathbf{y}_{r,1:N}^{(i)}\right)} \\
&= \frac{P\left(\mathbf{y}_{r,1:N}^{(i)}|z=m\right) P(z=m)}{\sum_{m=1}^{M} P\left(\mathbf{y}_{r,1:N}^{(i)}|z=m\right) P(z=m)} \\
&= \frac{\alpha_{r,m} \phi(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N})}{\sum_{m=1}^{M} \alpha_{r,m} \phi(\mathbf{y}_{r,1:N}^{(i)}; \mu_{r,m,1:N}, \sigma_{r,m,1:N})}.
\end{aligned}
\tag{26}
$$

After that, we have an M-step to obtain maximum likelihood estimates by making the derivative of the log likelihood equal zero,

$$
\alpha_{r,m} \leftarrow \frac{1}{N_r} \sum_{i}^{N_r} \gamma_{r,m,1:N} \tag{27}
$$

$$
\sigma_{r,m,1:N} \leftarrow \frac{\sum_{i=1}^{N_r} \gamma_{r,m,1:N} (\mathbf{y}_{r,m,1:N}^{(i)} - \mu_{r,m,1:N})^2}{\sum_{n=1}^{N_r} \gamma_{r,m,1:N}} \tag{28}
$$

We do not update the mean because we have its prior knowledge of value as the mean of base model prediction pixels within the bin set.

A.4 Convergence of GMMs with mean prior

Let $\theta = \{\alpha_{r,1}, ..., \alpha_{r,M}, \sigma_{r,1,1:N}, ..., \sigma_{r,M,1:N}\}$ be the estimate variable. To validate the convergence of GMMs, we want to prove

$$
P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| \theta^{t+1}\right) \geq P\left(\mathbf{y}_{r,1:N}^{(i)} \middle| \theta^{t}\right), \tag{29}
$$

where θ^t means the estimate variables at the t-th step.

We start from

$$\log P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\theta\right) = \log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta\right) - \log P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta\right) \tag{30}$$

Based on the objective of the M-step, i.e.,

$$\theta \in \arg\max_{\theta} \log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta\right). \tag{31}$$

we can naturally guarantee

$$\log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta^{t+1}\right) \geq \log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta^t\right). \tag{32}$$

We also have

$$\mathbb{E}_{z|\mathbf{y}_{r,1:N}^{(i)},\theta}\left[\log \frac{P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^{t+1}\right)}{P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^t\right)}\right] = -D_{\mathrm{KL}}\left(P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^t\right)\middle\|P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^{t+1}\right)\right) \leq 0, \tag{33}$$

where D_{KL} denotes the Kullback–Leibler (KL) divergence function.

We can thus obtain

$$\begin{aligned} &\log P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\theta^{t+1}\right) = \log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta^{t+1}\right) - \log P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^{t+1}\right) \\ \geq &\log P\left(\mathbf{y}_{r,1:N}^{(i)}\middle|\theta^t\right) = \log P\left(\mathbf{y}_{r,1:N}^{(i)}, z\middle|\theta^t\right) - \log P\left(z\middle|\mathbf{y}_{r,1:N}^{(i)}, \theta^t\right). \end{aligned} \tag{34}$$

The convergence is therefore guaranteed. The prior of mean does not affect the convergence.

B More Visualizations

B.1 Weight Visualization

We plot the heatmap of ensemble weights on an example from HIDE [60]. The weight heatmaps for averaging, ZZPM [92] and ours are shown in Fig. 4. We can see that ours assigns more detailed weight on its prediction to preserve textures.

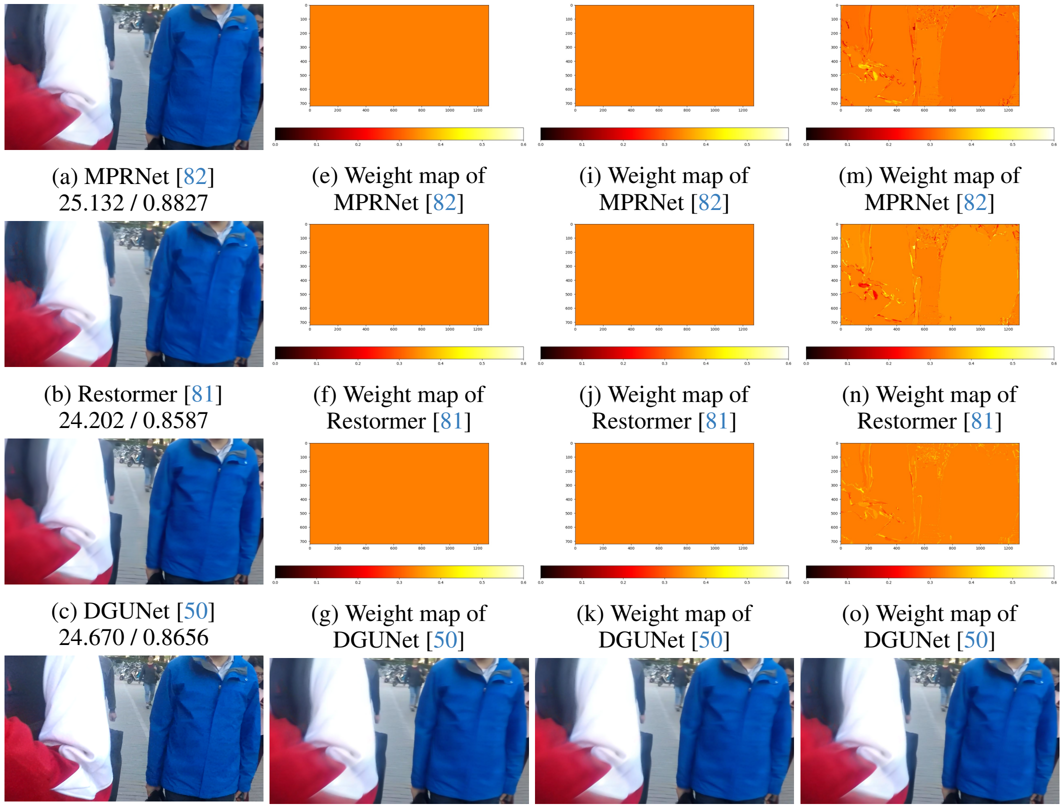
B.2 Feature Visualization

We plot the image feature visualization via the dimension reduction of principal components analysis (PCA). We randomly sample 8 images from HIDE [60]. For each image, we obtain its restoration predictions of base models, the ensemble results of averaging, ZZPM [92], HGBT [34] and ours. A pre-trained ResNet110 is used to encode images into features. The visualizations are shown in Fig. 5 It can be found that ours can consistently yield the results closer to the gorund-truths (GT).

B.3 Distribution Visualization

Note that we hypotheses of pixels within each bin set following Gaussian and their weights of GMMs learnt on the reference set can be used to fuse pixels on the test set. We validate them by plotting the histograms within bin sets in Fig. 6. As seen, the distribution of the pixels within a bin follows either typical Gaussian, plateau that can be considered as a flat Gaussian, or discrete signals that can be considered as steep Gaussian. The relative location of the distributions among base model predictions and ground-truths is also well preserved from the reference set to the test set.

We can notice that the simple averaging strategy can suffice for the case of the last row, while a biased averaging towards DGUNet [50] and MPRNet [82] will be better for the case of the first two rows. For the case of the third row, directly applying the prediction of DGUNet [50] will be the best. Our method estimating the range-wise weights as the latent probability of GMMs can handle all the cases, while the averaing strategies will fail.



(a) MPRNet [82] 25.132 / 0.8827
(e) Weight map of MPRNet [82]
(i) Weight map of MPRNet [82]
(m) Weight map of MPRNet [82]

(b) Restormer [81] 24.202 / 0.8587
(f) Weight map of Restormer [81]
(j) Weight map of Restormer [81]
(n) Weight map of Restormer [81]

(c) DGUNet [50] 24.670 / 0.8656
(g) Weight map of DGUNet [50]
(k) Weight map of DGUNet [50]
(o) Weight map of DGUNet [50]

(d) Ground-truth PSNR / SSIM
(h) Averaging 25.232 / 0.8817
(l) ZZPM [92] 25.233 / 0.8837
(p) Ours w/ $b = 32$ 25.247 / 0.8840

Figure 4: A sample of weight visualizations on HIDE [60]. Base models are DGUNet [50], MPRNet [82], and Restormer [81]. The first column shows the base model predictions and ground-truth. The second column shows the ensemble weights and result of the averaging strategy. The third column shows the ensemble weights and result of ZZPM [92]. The last column shows the ensemble weights and result of our method.

C More Visual Comparisons

We show more visual comparisons in Fig. 7-10 for image super-resolution, in Fig. 11-13 for image deblurring, and in Fig. 14-18 for image deraining.

In Fig. 7, only ours is able to preserve gray reflection of the boat from SwinIR [39]'s prediction. In Fig. 8-10, our method obtains the most accurate ensemble results in the case that one of base models generates mistaken textures.

In Fig. 11, our method yield the sharpest and straight line like the ground-truth. In Fig. 12 and 13, our ensemble method obtains the closest textures to the ground-truths.

In Fig. 14, our method gets the best ensemble despite the mistake from MAXIM [67]. In Fig. 15, ours preserves the cleanest background to the ground-truth. In Fig. 16, only ours recovers the grid near the eyeball. In Fig. 17 and 18, ours preserves the closest background textures to the ground-truths.

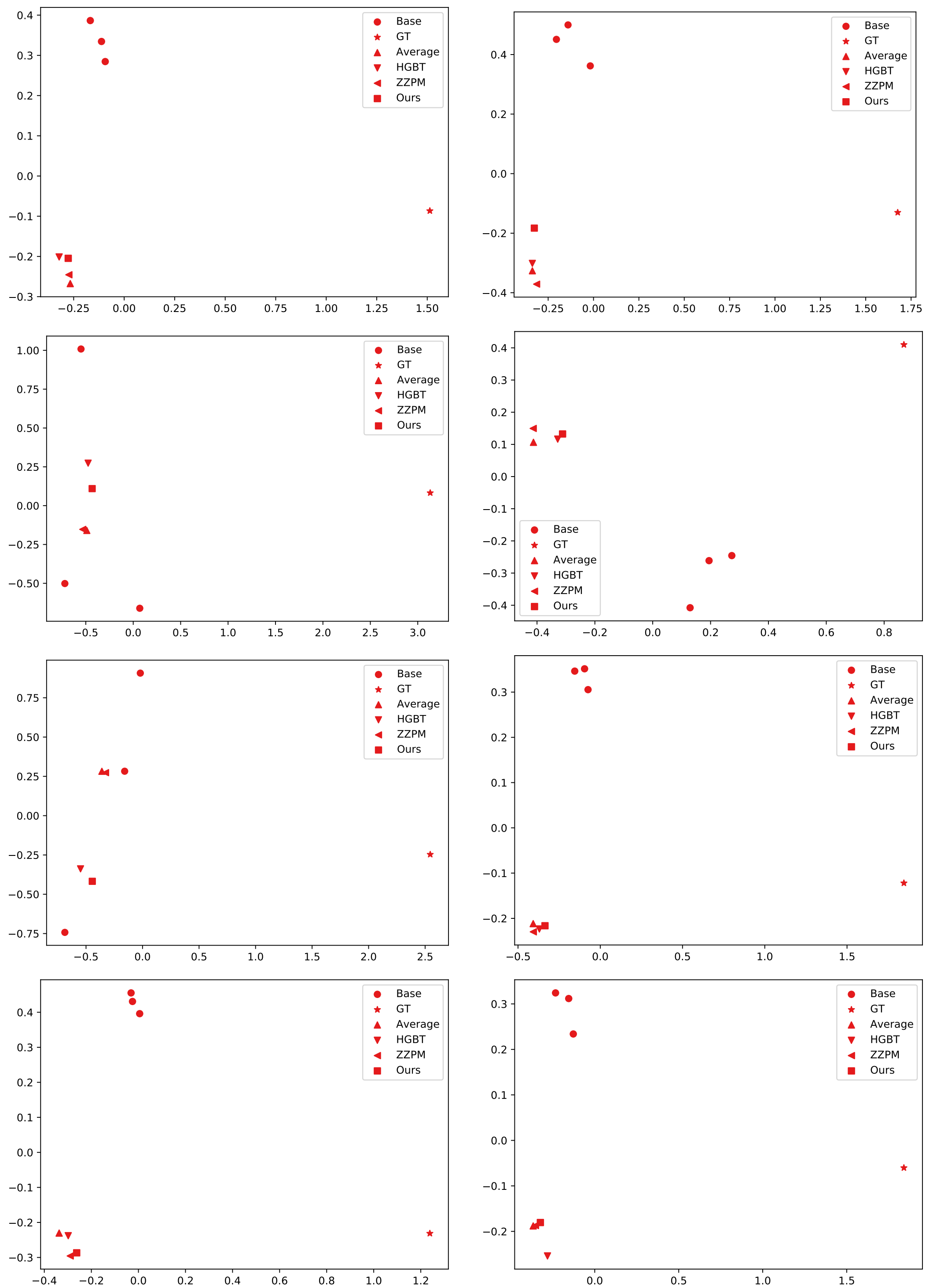
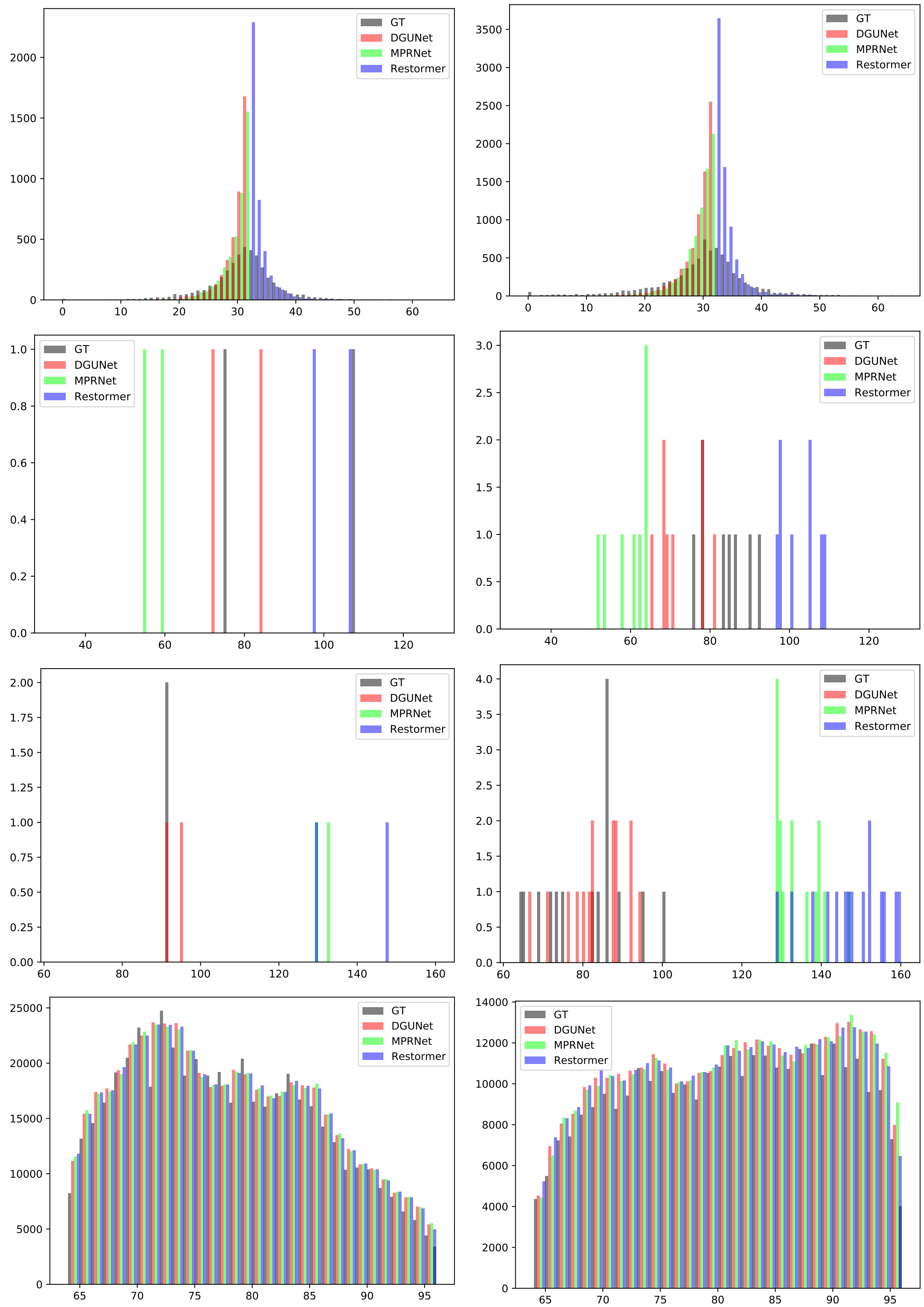


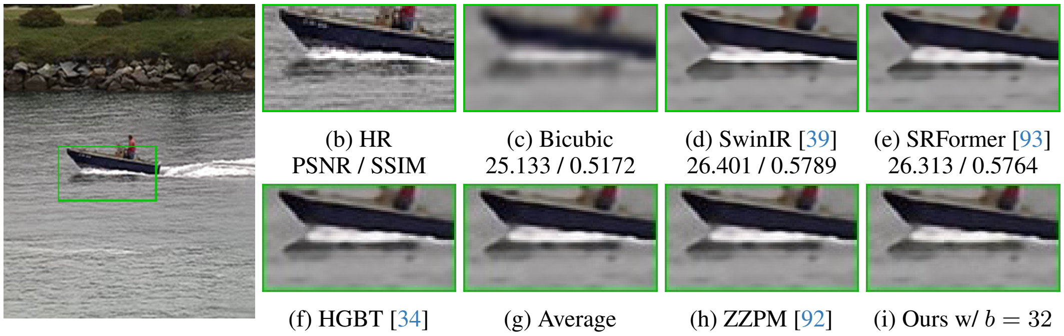
Figure 5: A sampled group of feature visualizations on HIDE [60]. “Base” denotes the features of three base models, i.e., DGUNet [50], MPRNet [82], and Restormer [81].



(a) Reference set

(b) Test set

Figure 6: A group of distribution visualizations on HIDE [60]. The bin sets of the first row is $(B_1 = [0, 32), B_2 = [0, 32), B_3 = [32, 64))$. The bin sets of the second row is $(B_1 = [64, 96), B_2 = [32, 64), B_3 = [96, 128))$. The bin sets of the third row is $(B_1 = [64, 96), B_2 = [128, 160), B_3 = [128, 160))$. The bin sets of the last row is $(B_1 = [64, 96), B_2 = [64, 96), B_3 = [64, 96))$. Base models are DGUNet [50], MPRNet [82], and Restormer [81].

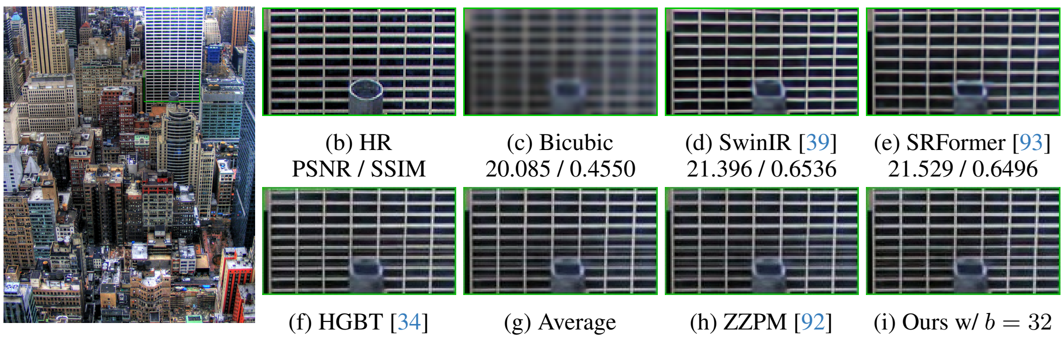


(a) Image

(b) HR PSNR / SSIM | (c) Bicubic 25.133 / 0.5172 | (d) SwinIR [39] 26.401 / 0.5789 | (e) SRFormer [93] 26.313 / 0.5764

(f) HGBT [34] 26.409 / 0.5790 | (g) Average 26.414 / 0.5793 | (h) ZZPM [92] 26.413 / 0.5792 | (i) Ours w/ $b = 32$ 26.430 / 0.5794

Figure 7: A visual comparison of ensemble on an image from Set14 [83] for the task of super-resolution. Please zoom in for better visual quality.

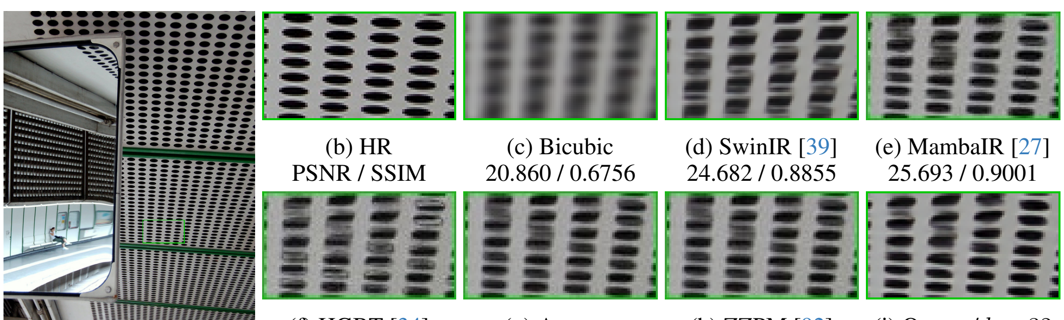


(a) Image

(b) HR PSNR / SSIM | (c) Bicubic 20.085 / 0.4550 | (d) SwinIR [39] 21.396 / 0.6536 | (e) SRFormer [93] 21.529 / 0.6496

(f) HGBT [34] 21.883 / 0.6604 | (g) Average 21.878 / 0.6616 | (h) ZZPM [92] 21.879 / 0.6615 | (i) Ours w/ $b = 32$ 21.893 / 0.6619

Figure 8: A visual comparison of ensemble on an image from Urban100 [30] for the task of super-resolution. Please zoom in for better visual quality.

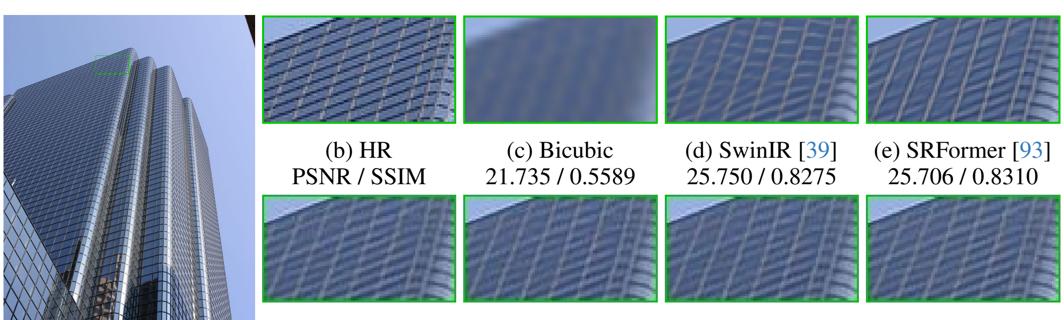


(a) Image

(b) HR PSNR / SSIM | (c) Bicubic 20.860 / 0.6756 | (d) SwinIR [39] 24.682 / 0.8855 | (e) MambaIR [27] 25.693 / 0.9001

(f) HGBT [34] 25.728 / 0.8987 | (g) Average 26.133 / 0.9021 | (h) ZZPM [92] 26.131 0.9020 | (i) Ours w/ $b = 32$ 26.197 / 0.9023

Figure 9: A visual comparison of ensemble on an image from Urban100 [30] for the task of super-resolution. Please zoom in for better visual quality.



(a) Image

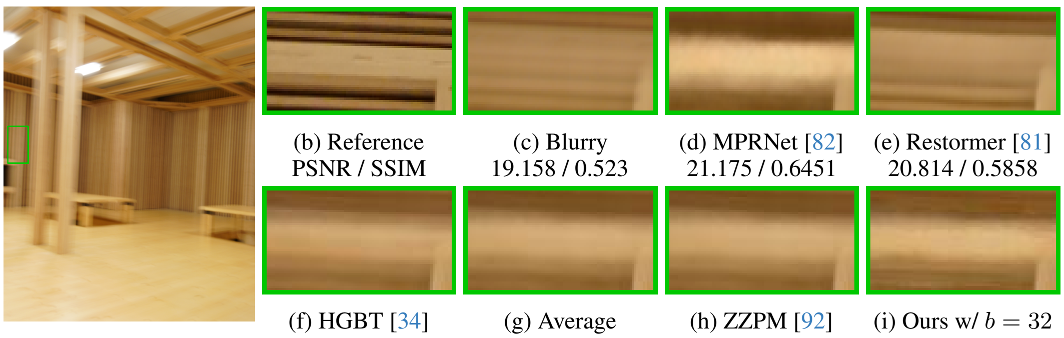
(b) HR PSNR / SSIM | (c) Bicubic 21.735 / 0.5589 | (d) SwinIR [39] 25.750 / 0.8275 | (e) SRFormer [93] 25.706 / 0.8310

(f) HGBT [34] 26.180 / 0.8406 | (g) Average 26.150 / 0.8402 | (h) ZZPM [92] 26.149 / 0.8401 | (i) Ours w/ $b = 32$ 26.189 / 0.8409

Figure 10: A visual comparison of ensemble on an image from Urban100 [30] for the task of super-resolution. Please zoom in for better visual quality.

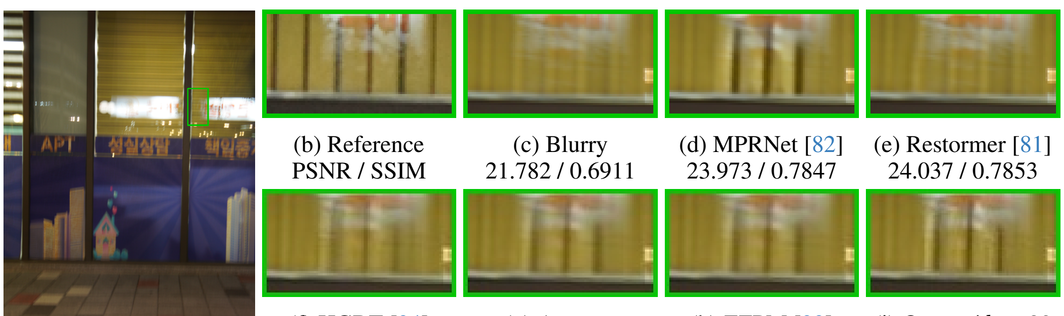
(a) Image | (b) Reference PSNR / SSIM | (c) Blurry 24.874 / 0.8636 | (d) MPRNet [82] 28.811 / 0.8800 | (e) HGBT [34] 29.802 / 0.8936 | (f) Average 29.783 / 0.8941 | (g) ZZPM [92] 29.781 / 0.8935 | (h) Ours w/ $b = 64$ 29.840 / 0.8943 | (i) Ours w/ $b = 32$ 29.864 / 0.8949

Figure 11: A visual comparison of ensemble on an image from HIDE [60] for the task of image deblurring. Please zoom in for better visual quality.



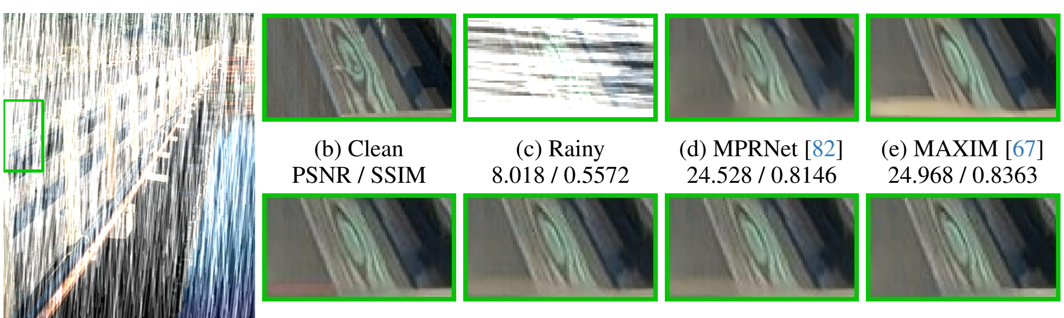
(a) Image | (b) Reference PSNR / SSIM | (c) Blurry 19.158 / 0.523 | (d) MPRNet [82] 21.175 / 0.6451 | (e) Restormer [81] 20.814 / 0.5858 | (f) HGBT [34] 21.498 / 0.6340 | (g) Average 21.471 / 0.6373 | (h) ZZPM [92] 21.470 / 0.6366 | (i) Ours w/ $b = 32$ 21.504 / 0.6374

Figure 12: A visual comparison of ensemble on an image from RealBlur-J [58] for the task of image deblurring. Please zoom in for better visual quality.



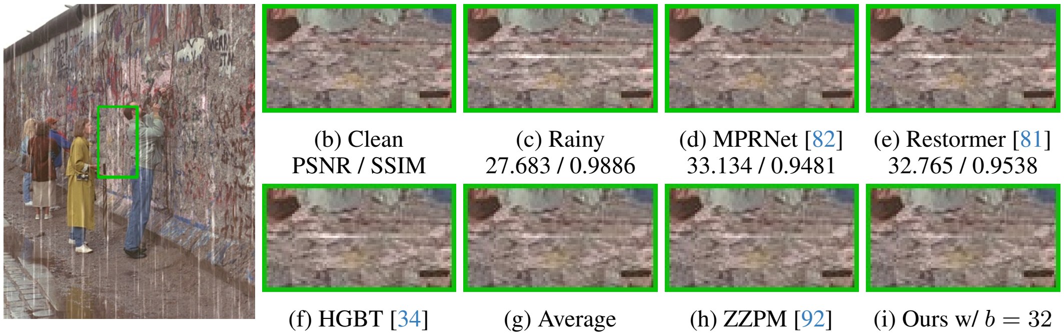
(a) Image | (b) Reference PSNR / SSIM | (c) Blurry 21.782 / 0.6911 | (d) MPRNet [82] 23.973 / 0.7847 | (e) Restormer [81] 24.037 / 0.7853 | (f) HGBT [34] 24.025 / 0.7848 | (g) Average 24.044 / 0.7873 | (h) ZZPM [92] 24.040 / 0.7865 | (i) Ours w/ $b = 32$ 24.079 / 0.7885

Figure 13: A visual comparison of ensemble on an image from RealBlur-J [58] for the task of image deblurring. Please zoom in for better visual quality.



(a) Image | (b) Clean PSNR / SSIM | (c) Rainy 8.018 / 0.5572 | (d) MPRNet [82] 24.528 / 0.8146 | (e) MAXIM [67] 24.968 / 0.8363 | (f) HGBT [34] 26.041 / 0.8541 | (g) Average 26.033 / 0.8531 | (h) ZZPM [92] 26.034 / 0.8530 | (i) Ours w/ $b = 32$ 26.146 / 0.855

Figure 14: A visual comparison of ensemble on an image from Rain100H [74] for the task of image deraining. Please zoom in for better visual quality.

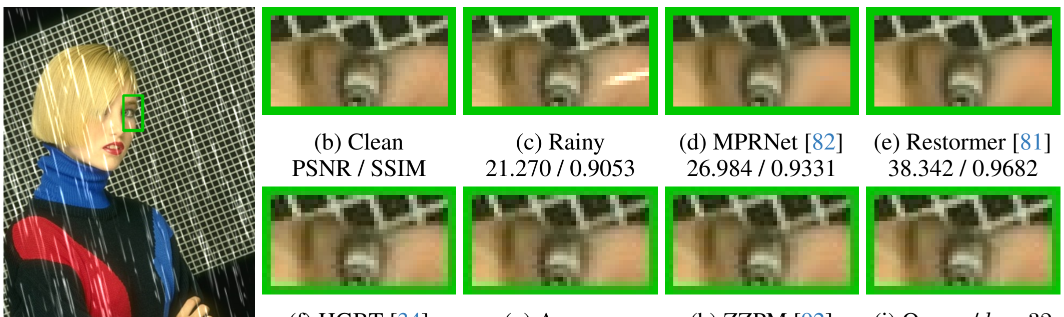


(a) Image

(b) Clean PSNR / SSIM | (c) Rainy 27.683 / 0.9886 | (d) MPRNet [82] 33.134 / 0.9481 | (e) Restormer [81] 32.765 / 0.9538

(f) HGBT [34] 33.235 / 0.9562 | (g) Average 34.659 / 0.9632 | (h) ZZPM [92] 34.656 / 0.9631 | (i) Ours w/ $b = 32$ 34.823 / 0.9658

Figure 15: A visual comparison of ensemble on an image from Rain100L [74] for the task of image deraining. Please zoom in for better visual quality.

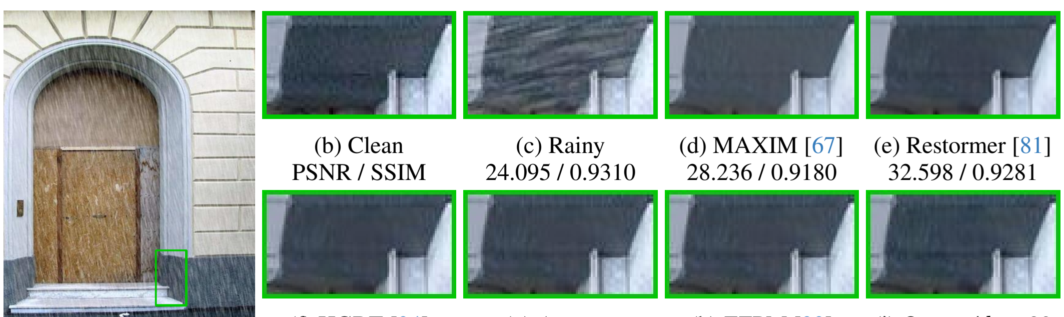


(a) Image

(b) Clean PSNR / SSIM | (c) Rainy 21.270 / 0.9053 | (d) MPRNet [82] 26.984 / 0.9331 | (e) Restormer [81] 38.342 / 0.9682

(f) HGBT [34] 38.011 / 0.9725 | (g) Average 34.532 / 0.9720 | (h) ZZPM [92] 34.533 / 0.9704 | (i) Ours w/ $b = 32$ 39.525 / 0.9791

Figure 16: A visual comparison of ensemble on an image from Rain100L [74] for the task of image deraining. Please zoom in for better visual quality.

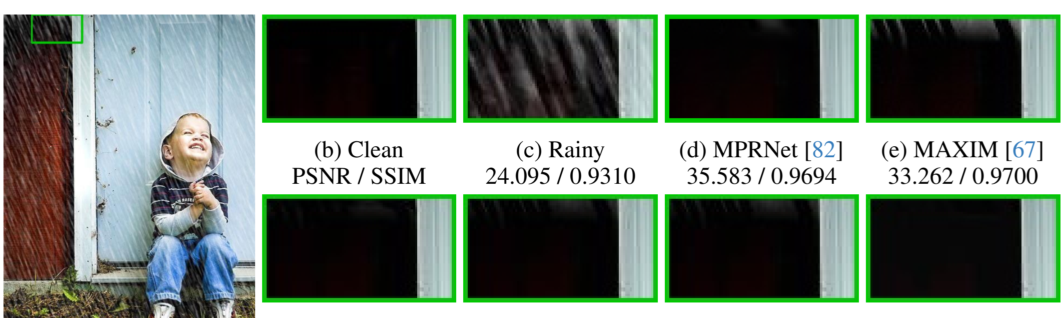


(a) Image

(b) Clean PSNR / SSIM | (c) Rainy 24.095 / 0.9310 | (d) MAXIM [67] 28.236 / 0.9180 | (e) Restormer [81] 32.598 / 0.9281

(f) HGBT [34] 32.516 / 0.9286 | (g) Average 32.371 / 0.9309 | (h) ZZPM [92] 32.380 / 0.9306 | (i) Ours w/ $b = 32$ 32.893 / 0.9310

Figure 17: A visual comparison of ensemble on an image from Test1200 [86] for the task of image deraining. Please zoom in for better visual quality.



(a) Image

(b) Clean PSNR / SSIM | (c) Rainy 24.095 / 0.9310 | (d) MPRNet [82] 35.583 / 0.9694 | (e) MAXIM [67] 33.262 / 0.9700

(f) HGBT [34] 35.950 / 0.9724 | (g) Average 35.926 / 0.9740 | (h) ZZPM [92] 35.931 / 0.9739 | (i) Ours w/ $b = 32$ 36.318 / 0.9746

Figure 18: A visual comparison of ensemble on an image from Test2800 [86] for the task of image deraining. Please zoom in for better visual quality.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and precede the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction has accurately reflected the paper's contribution and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We have discussed the limitation of the last page of the main manuscript.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We have provided the complete proof in Appendix and the assumptions.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have included all implementation and experimental details in Sec. 4

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have released our codes and all ensemble weights.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the implementation details in Sec. 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The scales of the tables are huge and the inclusion of error bars is expensive and violate the page limit.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have shown the computer resources in Sec. 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no social impact of the work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited all the original paper referenced to.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets in this work.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.