PROTOFLOW: UNIVERSAL ONE-STEP FLOW RESTORER VIA PROTOTYPE GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Universal image restoration seeks to recover high-quality images from diverse degradations within a unified framework. Recent diffusion- and flow-based methods achieve strong restoration performance by leveraging powerful generative priors, yet remain costly due to multi-step computation and offer only limited control through text prompts. In addition, restoration performance deteriorates when explicit degradation information is unavailable. To address these issues, we rethink image restoration as a short-horizon transport from degraded latent toward the clean-image manifold, where the timestep naturally serves as a restoration-strength controller. We further introduce ProtoFlow, a prototype-guided one-step flow model that leverages Gaussian prototypes to enhance the representation of both degradation and semantics, thereby bridging the gap between settings with and without explicit degradation. On CoTIR-Bench, ProtoFlow significantly outperforms the CoTIR-4B on full-reference metrics, remains competitive on no-reference metrics, and runs about $7.7\times$ faster. These results demonstrate that one-step flow restoration can achieve strong performance, flexible intensity control, and high efficiency when paired with structured prototype priors.

1 INTRODUCTION

Image restoration aims to recover high-quality images from degraded observations and serves as a fundamental task in photography, computational imaging, autonomous driving, surveillance, and embodied perception. In real-world settings, however, degradations rarely occur in isolation. Noise, blur, haze, low-light conditions, compression artifacts, and their mixtures often co-exist, making it difficult to design a separate model for each corruption type. This challenge has motivated the development of *universal image restoration*, which seeks to address diverse degradations within a single model. Early all-in-one restoration methods typically rely on single backbone to handle multiple corruption types (Li et al., 2022; Potlapalli et al., 2024; Guo et al., 2024; Conde et al., 2024). Although effective, these methods often struggle when degradations are severely entangled or recovering semantic content beyond local regression.

Recent diffusion and flow-based restoration methods have significantly improved restoration quality by exploiting powerful generative priors (Ai et al., 2024; Fei et al., 2026; Guo et al., 2026). By modeling restoration as a multi-step computation in latent space, these methods can better recover plausible textures and structures under challenging real-world degradations. In particular, large flow-based restoration models such as CoTIR inherit the editing ability of pretrained latent flow models and achieve strong universal restoration performance (Guo et al., 2026). However, these methods still rely on multi-step computation, making inference costly and offering no explicit restoration-strength control. Moreover, although CoTIR improves restoration performance without degradation types through an internalized chain-of-thought, a significant gap (see Table 1) remains compared with the setting where degradation types are given.

To address these issues, we revisit flow-based generative restoration from the perspective of restoration dynamics and make a simple but important observation: unlike image generation from pure gaussian noise, image restoration starts from a degraded observation that already contains the major scene layout and coarse content of the target image. Therefore, the degraded latent is closer to the clean-image manifold than a random initial state. This suggests that restoration does not necessarily require constructing a long reverse transport trajectory. Instead, it can be approximated as a short-horizon

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

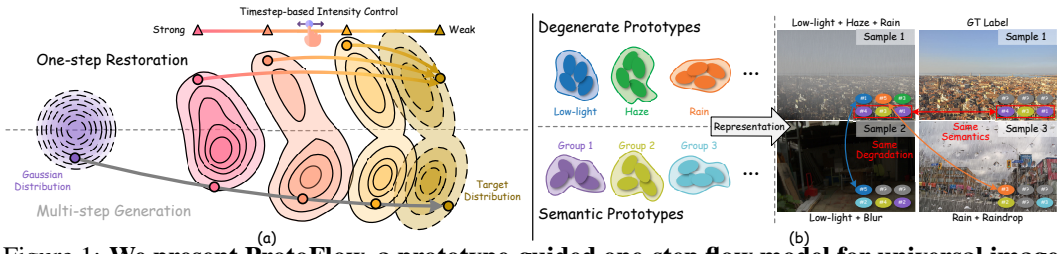


Figure 1: **We present ProtoFlow, a prototype-guided one-step flow model for universal image restoration.** (a) Comparison between the proposed one-step restoration and multi-step generation. (b) Illustration of the image representations via degradation and semantic prototypes.

correction from the degraded latent toward the clean endpoint. Motivated by recent progress in one-step generative modeling (Frans et al., 2025; Geng et al., 2025; Lin et al., 2025), we reformulate universal image restoration as a *one-step flow transport* problem, where the model directly predicts the restoration direction at the degraded latent and recovers the clean estimate with only one velocity evaluation. As shown in Fig. 1(a), this reformulation replaces conventional multi-step generation with one-step restoration. Under this formulation, the timestep naturally becomes an explicit restoration-strength controller, enabling the same model to flexibly trade off conservative enhancement and stronger correction.

While a one-step transport formulation improves efficiency and enables controllable strength, accurate restoration without degradation prompt still requires structured priors that disentangle degradation patterns from clean-image semantics. To this end, we present **ProtoFlow**, a **Proto**type-guided one-step **Flow** model for universal image restoration. Our key idea is to introduce two prototype banks to improve restoration performance and narrow the gap between settings with and without degradation. One bank captures degradation patterns in the input, while the other captures semantic features that should be preserved during restoration. Instead of representing each prototype with a fixed embedding, we model every prototype as a Gaussian distribution that captures both its representative mean and intra-prototype variance. This is particularly important for universal restoration, where the same degradation and semantic category may exhibit large appearance variation. The retrieved prototypes are injected into the pretrained restorer through a lightweight conditioner that provides explicit guidance. As shown in Fig. 1(b), three samples illustrate this behavior: the low-quality image and ground-truth label of the same sample are assigned to the same semantic prototypes, while different samples with a common degradation tend to select degradation prototypes from the same group. In summary, our contributions are three-fold:

- We reformulate universal restoration under a pretrained generative flow as a one-step transport problem, showing that degraded latent can be projected toward the clean manifold with one velocity evaluation, while the timestep serves as an explicit restoration-strength controller.
- We propose ProtoFlow, a prototype-guided one-step flow framework that introduces Gaussian degradation and semantic prototypes to accurately characterize degradation patterns and semantic content, thereby improving restoration performance.
- Extensive experiments on CoTIR-Bench demonstrate that ProtoFlow achieves state-of-the-art performance with substantially improved efficiency and flexible intensity control, validating the effectiveness of combining one-step flow restoration with structured prototype priors.

2 Related Work

Universal Image Restoration. Universal image restoration aims to address diverse degradations with a single model, avoiding the need to train a separate restorer for each degradation type. Early efforts rely on shared discriminative backbones with task-specific modulation, prompt learning, or weather-aware feature decomposition to improve robustness across multiple degradations (Li et al., 2022; Kulkarni et al., 2022; Zhu et al., 2023; Potlapalli et al., 2024; Guo et al., 2024; Conde et al., 2024). These methods have improved the practicality of all-in-one restoration, but they remain predominantly regression-based and often struggle to recover realistic structures when degradations are severely entangled. More recent works incorporate stronger generative or instruction-driven priors for universal restoration (Ai et al., 2024; Fei et al., 2026; Guo et al., 2026). For example,

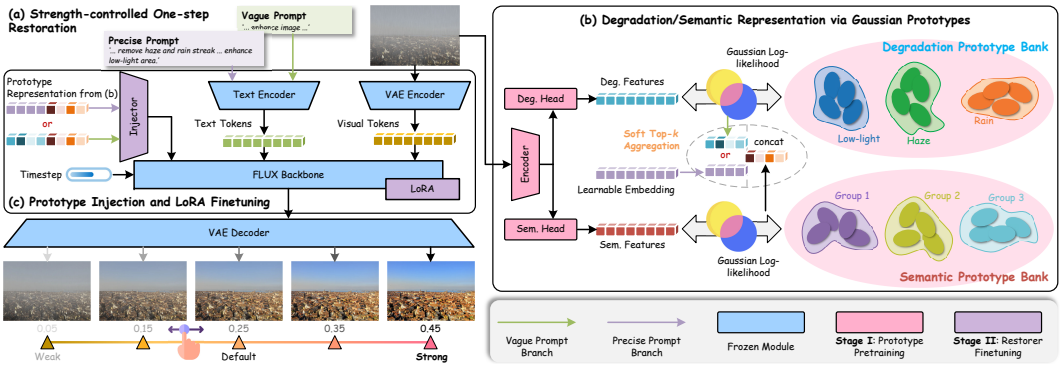


Figure 2: **Overall pipeline of ProtoFlow.** Stage I learns degradation and semantic prototype banks from image-level descriptors as restoration priors. Stage II freezes the prototype conditioner, retrieves prototype tokens from the low-quality input, and injects them into the pretrained backbone, where a lightweight injector and LoRA adapters are finetuned for one-step restoration.

CoTIR leverages a large pretrained latent flow model for restoration under both known and unknown degradations (Guo et al., 2026). In contrast to prior universal restoration methods that either depend on discriminative regression or inherit expensive multi-step processing, our work reformulates restoration as a one-step process and introduces prototype priors to represent degradation and semantic structure.

Diffusion and Flow-based Image Generation. Diffusion and flow-based models are two dominant paradigms for high-fidelity image generation. Diffusion models, including DDPM and DDIM, show that iterative denoising can model complex image distributions with strong fidelity (Ho et al., 2020; Song et al., 2020). Latent diffusion and diffusion transformers further improve scalability and representation capacity for high-resolution generation (Rombach et al., 2022; Peebles & Xie, 2023). Meanwhile, flow matching and rectified flow provide an alternative continuous transport formulation, enabling efficient trajectory learning and inspiring few-step and one-step generation (Lipman et al., 2023; Liu et al., 2023; Frans et al., 2025; Geng et al., 2025). These advances establish strong pretrained generative priors for downstream vision tasks. Unlike prior image-generation studies that synthesize images from noise, our work adapts a pretrained flow prior to universal restoration and reformulates restoration as a one-step transport with controllable strength.

Prototype Learning. Prototype learning aims to represent data with a set of representative prototypes, enabling compact structure modeling, interpretable assignment, and concept-aware reasoning. Early studies typically use prototype learning for case-based reasoning, metric matching, and interpretable visual recognition (Snell et al., 2017; Chen et al., 2019). Subsequent studies improve prototype learning with more flexible geometric modeling and stronger backbones, such as deformable prototypes and prototype-based vision transformers (Donnelly et al., 2022; Ma et al., 2024). More recent approaches moves beyond deterministic point prototypes by introducing richer structural guidance and distributional formulations. For example, MGProto models prototypes as Gaussian mixtures to improve representation capacity, prototype projection, and out-of-distribution trustworthiness (Wang et al., 2025), while ProtoMask uses segmentation guidance to align prototypes with semantically meaningful regions (Meinert et al., 2025). However, existing prototype methods have rarely been studied as explicit priors for universal generative restoration. We use distributional prototypes to represent diverse degradation patterns and degradation-invariant content, thereby guiding a one-step restorer when precise degradation is unavailable.

3 Problem Formulation

3.1 Multi-step Flow Matching

Recent flow matching-based generative models, such as FLUX (Batifol et al., 2025; Labs, 2025), learn a velocity field that transports a latent along a continuous trajectory. Let x_t denote the latent state at timestep t and let c denote the conditioning information. The reverse transport is given by

$$\frac{dx_t}{dt} = v_\theta(x_t, t, c), \tag{1}$$

where v_θ is the model-estimated velocity vector field and θ denotes the model parameters. Its solution is typically approximated by multi-step processing from $t = 1$ to $t = 0$. This formulation is well suited to image generation and editing, where the model starts from noise and progressively moves toward the target distribution under the guidance of c. Recently, CoTIR (Guo et al., 2026) extends FLUX to universal image restoration by using the degraded image and restoration instruction jointly as the conditioning variable c. However, it still inherits the high cost of multi-step computation, typically requiring 5–20 velocity evaluations, and mainly relies on text instructions to modulate restoration behavior.

3.2 Strength-controlled One-step Restoration

Based on these observations and inspired by recent generative modeling works (Frans et al., 2025; Geng et al., 2025; Lin et al., 2025), we reformulate universal image restoration as a short-horizon one-step transport problem. The key intuition is that restoration does not start from noise, but from a degraded latent that already preserves coarse scene structure and is therefore closer to an intermediate transport state than to a random sample. This makes it natural to use the low-quality image as the current flow state and to predict a direct transport toward the clean endpoint. Under this view, the timestep not only indexes the transport state but also determines the correction magnitude, thereby providing explicit restoration-strength control. Let I_{lq} and I_{gt} denote a degraded image and its clean target, respectively. We encode them into the latent space of the pretrained FLUX autoencoder as

$$z_{\text{lq}} = \mathcal{E}(I_{\text{lq}}), \qquad z_{\text{gt}} = \mathcal{E}(I_{\text{gt}}), \tag{2}$$

where $\mathcal{E}(\cdot)$ is the VAE encoder. The task can be viewed as transporting a corrupted latent back to the clean manifold.

More specifically, degraded latent and noisy intermediate states are similar in that both deviate from the clean-image manifold while preserving part of the scene structure. This similarity also supports adapting pretrained generative models to restoration through finetuning. Accordingly, we approximately treat the degraded latent as the current flow state at timestep t, i.e., $x_t \approx z_{\text{lq}}$, and recover the clean latent x_0 from this state. Under the reverse transport dynamics,

$$x_0 = x_t - \int_0^t v_\theta(x_\tau, \tau, c)\, d\tau. \tag{3}$$

Inspired by the average-velocity view in MeanFlow (Geng et al., 2025), Eq. (3) can be rewritten as

$$x_0 = x_t - t\,\bar{v}(x_t, t, c), \quad \bar{v}(x_t, t, c) = \frac{1}{t}\int_0^t v_\theta(x_\tau, \tau, c)\, d\tau,$$

where $\bar{v}(x_t, t, c)$ denotes the average reverse transport velocity over the interval $[0, t]$. In our task, restoration starts from a degraded state close to the clean manifold, and the remaining reverse horizon is short. We therefore adopt a first-order approximation $\bar{v}(x_t, t, c) \approx v_\theta(x_t, t, c)$, which gives

$$\hat{x}_0 = x_t - t\, v_\theta(x_t, t, c). \tag{4}$$

We therefore learn a velocity field such that the one-step estimate $\hat{x}_0$ is close to the clean latent z_{gt}:

$$\min_\theta \ \mathbb{E}_{(I_{\text{lq}}, I_{\text{gt}}, c)} \left[\mathcal{L}_{\text{img}}(z_{\text{gt}}, \hat{x}_0)\right]. \tag{5}$$

The pretrained multi-step flow model is used as a restoration prior, while the actual task is transferred to estimating a restoration-oriented velocity field at the degraded latent. The restored image is then recovered by decoding $\hat{x}_0$ with the pretrained decoder $\mathcal{D}$, i.e., $\hat{I} = \mathcal{D}(\hat{x}_0)$.

The one-step restoration update in Eq. (4) also explains why t naturally controls restoration strength. By taking the partial derivative of Eq. (4) with respect to t, we obtain

$$\frac{\partial \hat{x}_0}{\partial t} = -\, v_\theta(x_t, t, c) - \, t\frac{\partial v_\theta(x_t, t, c)}{\partial t}, \tag{6}$$

which shows that t affects restoration through both the explicit step size and the time-dependent velocity field. To further quantify restoration strength, define

$$\Delta_t = \hat{x}_0 - x_t = -t\, v_\theta(x_t, t, c). \tag{7}$$

The restoration magnitude is therefore characterized by

$$\|\Delta_t\|_2 = t\, \|v_\theta(x_t, t, c)\|_2, \tag{8}$$

which shows that, under a locally stable velocity field, a larger timestep generally induces a stronger correction. More discussion of this controllable restoration behavior is provided in the experimental section.

4 METHOD

The overall framework is shown in Fig. 2. Following the one-step restoration formulation, ProtoFlow augments a pretrained FLUX-based restorer with Gaussian prototypes that represent degradation patterns and semantic features. The prototype tokens are then injected into the pretrained backbone to guide one-step image restoration.

4.1 DEGRADATION AND SEMANTIC REPRESENTATION VIA GAUSSIAN PROTOTYPES

Visual Feature Extraction. Given a low-quality image I_{lq} and its clean target I_{gt}, we first extract visual latent with a shared encoder:

$$u_{\text{lq}} = \mathcal{E}_{\text{proto}}(I_{\text{lq}}), \qquad u_{\text{gt}} = \mathcal{E}_{\text{proto}}(I_{\text{gt}}), \tag{9}$$

where $\mathcal{E}_{\text{proto}}(\cdot)$ denotes the ResNet18 backbone followed by global pooling. We then extract a degradation query from the low-quality latent and semantic queries from both the low-quality and clean latent:

$$q^{\text{d}}_{\text{lq}} = h_{\text{d}}(u_{\text{lq}}), \qquad q^{\text{s}}_{\text{lq}} = h_{\text{s}}(u_{\text{lq}}), \qquad q^{\text{s}}_{\text{gt}} = h_{\text{s}}(u_{\text{gt}}), \tag{10}$$

where $h_{\text{d}}(\cdot)$ and $h_{\text{s}}(\cdot)$ denote the degradation and semantic heads, respectively.

Prototype Bank Construction. Based on these queries, we construct two Gaussian prototype banks in the shared feature space. The degradation bank allocates multiple prototypes to each degradation type, while the semantic bank is organized into semantic groups. Given a query q, its matching score to prototype (μ, σ^2) is computed by the Gaussian log-likelihood:

$$\log p(q \mid \mu, \sigma^2) = -\frac{1}{2} \sum_k \left[\log(2\pi\sigma_k^2) + \frac{(q_k - \mu_k)^2}{\sigma_k^2} \right]. \tag{11}$$

Compared with deterministic prototypes, this parameterization models both representative centers and intra-prototype dispersion, making the learned prior more suitable for mixed and diverse real-world degradations.

1) Degradation Prior Learning. Using Eq. equation 11, we compute the matching score between the degradation query and each degradation prototype:

$$s_{c,n} = \log p\Big(q^{\text{d}}_{\text{lq}} \mid \mu_{c,n}, \sigma^2_{c,n}\Big), \tag{12}$$

where c indexes the degradation type and $n = 1, \ldots, N_{\text{d}}$ indexes the prototypes within that type. We then aggregate prototype scores within each degradation type:

$$s^{\text{d}}_c = \log \left(\frac{1}{N_{\text{d}}} \sum_{n=1}^{N_{\text{d}}} \exp\left(s_{c,n}\right) \right), \tag{13}$$

Here, s^{d}_c is the score of degradation class c, $s^{\text{d}} = [s^{\text{d}}_1, \ldots, s^{\text{d}}_C]$, and $y \in \{0,1\}^C$ is the multi-hot degradation label vector. Since one image may contain multiple degradations, we supervise these class scores with a multi-label binary cross-entropy with logits loss:

$$\mathcal{L}_{\text{d}} = \text{BCEWithLogits}(s^{\text{d}}, y). \tag{14}$$

2) Semantic Prior Learning. Similar to the degradation branch, we use Eq. equation 11 to compute the matching scores between the semantic queries q^{s}_{lq} and q^{s}_{gt} and all semantic prototypes parameterized by $\mu_{g,n}$ and $\sigma^2_{g,n}$, where g indexes the semantic group and n indexes the prototype within each group. We then flatten the resulting responses into the prototype-response vectors s^{s}_{lq} and s^{s}_{gt}. Unlike the degradation branch, the semantic bank is learned without semantic labels and is optimized with a symmetric contrastive loss:

$$\mathcal{L}_{\text{s}} = \text{InfoNCE}\big(s^{\text{s}}_{\text{lq}}, s^{\text{s}}_{\text{gt}}\big), \tag{15}$$

where $\text{InfoNCE}(\cdot,\cdot)$ denotes the symmetric batch-wise InfoNCE loss, in which paired LQ/GT samples are positives and the remaining in-batch pairs are negatives.

Stage I: Prototype Pretraining. Before end-to-end training, we pretrain the prototype representations to obtain degradation and semantic prototype priors from the training data. The prototype pretraining is constrained by the following loss:

$$\mathcal{L}_{\text{proto}} = \lambda_{\text{d}}\mathcal{L}_{\text{d}} + \lambda_{\text{s}}\mathcal{L}_{\text{s}} + \mathcal{L}_{\text{reg}}, \tag{16}$$

where λ_{d} and λ_{s} are the weights of the degradation and semantic losses, respectively, and $\mathcal{L}_{\text{reg}}$ denotes regularization terms. Inspired by MGProto (Wang et al., 2025), we further adopt an Expectation-Maximization (EM) algorithm to stabilize prototype training. Details of the regularization terms and the EM-based updates are provided in Appendix A.

Table 1: **Quantitative comparisons with fine-tuned methods on CoTIR-Bench.** The reported time is measured on images of size 1024×1024. † means the input of degradation prompts. Superscripts on ProtoFlow and ProtoFlow† indicate the relative percentage change with respect to CoTIR-4B and CoTIR-4B†, respectively.

Methods	Venue	Full Reference			No Reference			Time (s)↓
		PSNR↑	SSIM↑	LPIPS↓	CLIP-IQA+↑	Q-Align↑	LIQE↑	
CNN/Transformer-based Methods								
PromptIR	NeurIPS23	22.78	0.8028	0.2325	0.5343	3.2822	2.7011	0.229
OneRestore†	ECCV24	24.76	0.8584	0.1676	0.5443	3.4235	2.9409	0.097
InstructIR	ECCV24	22.94	0.8070	0.2161	0.5404	3.3228	2.7986	0.132
InstructIR†	ECCV24	25.13	0.8603	0.1763	0.5353	3.3667	2.7914	0.132
Diffusion/Flow-based Methods								
DA-CLIP	ICLR24	24.00	0.8171	0.1593	0.5765	3.5149	3.1581	76.312
HYPIR†	TOG25	21.55	0.6930	0.1737	0.6318	3.8116	3.5813	0.285
LucidFlux	ICLR26	17.13	0.5364	0.3798	0.5632	3.5314	3.3671	18.760
CoTIR-4B	Arxiv26	24.52	0.8360	0.1206	0.6272	3.8849	3.7061	4.602
CoTIR-4B†	Arxiv26	25.19	0.8454	0.1128	0.6262	3.8676	3.7016	4.602
ProtoFlow		26.34$^{\uparrow 7.4\%}$	0.8741$^{\uparrow 4.6\%}$	0.0836$^{\uparrow 30.7\%}$	0.6282$^{\uparrow 0.2\%}$	3.8303$^{\downarrow 1.4\%}$	3.6889$^{\downarrow 0.5\%}$	0.594$^{\uparrow 7.7\times}$
ProtoFlow†		26.62$^{\uparrow 5.7\%}$	0.8764$^{\uparrow 3.7\%}$	0.0818$^{\uparrow 27.5\%}$	0.6278$^{\uparrow 0.3\%}$	3.8245$^{\downarrow 1.1\%}$	3.6802$^{\downarrow 0.6\%}$	0.591$^{\uparrow 7.8\times}$

4.2 Prototype Injection and LoRA Finetuning

Prototype Injection. After training the prototype-prior module, we retrieve one degradation token and one semantic token from the learned prototype banks via soft top-k aggregation,

$$u^b = \sum_{i \in \mathcal{K}^b} \omega_i^b \mu_i^b, \quad \omega_i^b = \frac{\exp(s_i^b)}{\sum_{j \in \mathcal{K}^b} \exp(s_j^b)}, \quad b \in \{\mathrm{d}, \mathrm{s}\}, \tag{17}$$

where $\mathcal{K}^b$ denotes the selected prototype set and s_i^b denotes the corresponding routing score for branch b. For vague prompts, we use the retrieved degradation token u^{d}; for precise prompts, we replace it with a learnable degradation embedding $u_\emptyset^{\mathrm{d}}$:

$$\tilde{u}^{\mathrm{d}} = \begin{cases} u^{\mathrm{d}}, & \text{vague prompt,} \\ u_\emptyset^{\mathrm{d}}, & \text{precise prompt,} \end{cases} \qquad r = \mathrm{Injector}\Big([\tilde{u}^{\mathrm{d}}; u^{\mathrm{s}}]\Big), \tag{18}$$

where r is the residual conditioning signal injected into the backbone. The final projection is zero-initialized to ensure stable training and gradually learn useful restoration biases.

Stage II: Model Finetuning. Stage II finetunes the restoration model with the pretrained prototype branch as frozen prior guidance. In this stage, only the LoRA layers and the prototype injector are optimized, while the prototype conditioner is kept fixed except for optional extra parameters. Training objective consists of MSE $\mathcal{L}_{\ell_2}$, perceptual loss $\mathcal{L}_{\mathrm{lpips}}$, and structural similarity loss $\mathcal{L}_{\mathrm{SSIM}}$:

$$\mathcal{L}_{\mathrm{img}} = \lambda_{\ell_2} \mathcal{L}_{\ell_2} + \lambda_{\mathrm{lpips}} \mathcal{L}_{\mathrm{lpips}} + \lambda_{\mathrm{ssim}} \mathcal{L}_{\mathrm{SSIM}}. \tag{19}$$

Here, λ_{ℓ_2}, λ_{lpips}, and λ_{ssim} are the weights of the MSE, perceptual, and structural similarity losses, respectively. In particular, $\mathcal{L}_{\mathrm{lpips}}$ and $\mathcal{L}_{\mathrm{SSIM}}$ are computed in the pixel space after the generated output is decoded by the VAE decoder, while $\mathcal{L}_{\ell_2}$ is computed in the latent space.

5 Experiments

5.1 Implementation Details

Our framework is implemented in PyTorch 2.8 with Python 3.10 and executed on a PC with two Intel Xeon Silver 4410Y CPUs and two NVIDIA RTX 5880 Ada GPUs. ProtoFlow is built on the pretrained FLUX.2-4B generative model. We train the model on the CoTIR-Bench training set Guo et al. (2026), which contains 19 degradation types and 5.2M 512×512 samples. For prototype-related hyperparameter settings, the degradation bank contains 19 degradation types, and the semantic bank contains 128 semantic groups. Each degradation type or semantic group includes 6 Gaussian prototypes. For both branches, we select the top-10 prototypes for soft aggregation. We set patch size 512×512, LoRA rank 256, and restoration strength $t = 0.25$. Both stages are optimized with AdamW using a base learning rate of 5×10^{-5} and cosine decay to 5×10^{-6}. Stage I uses 50k



Figure 3: **Restoration comparisons on the CoTIR-Bench.** † means the input of degradation prompts.

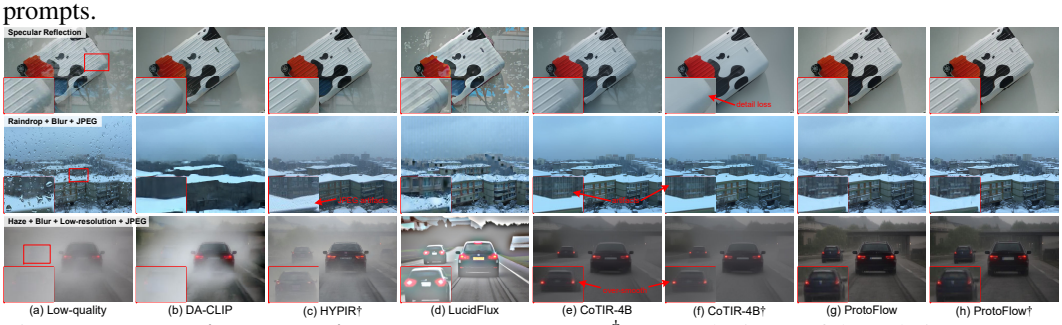


Figure 4: **Restoration comparisons on the real scenes.** † means the input of degradation prompts.

iterations with batch size 128, where the loss weights are empirically set to $\lambda_{\mathrm{d}} = 0.7$, $\lambda_{\mathrm{s}} = 0.35$, and 0.01 for each geometry regularizer. Stage II uses 50k iterations with batch size 4, and the restoration loss weights are set to $\lambda_{\mathrm{LPIPS}} = \lambda_{\ell_2} = \lambda_{\mathrm{SSIM}} = 1$.

5.2 Main Results on Image Restoration

Results on CoTIR-Bench. We evaluate ProtoFlow on the 2,000 test samples of CoTIR-Bench, which cover nearly 70 datasets and 19 degradation types. Following the benchmark protocol, we compare with the methods listed in Table 1. We report three full-reference metrics, including PSNR, SSIM, and LPIPS, together with three no-reference metrics, including CLIP-IQA+, Q-Align, and LIQE. Among them, lower LPIPS indicates better quality, while all the other metrics are higher-is-better. The quantitative results are summarized in Table 1. Compared with the previous state-of-the-art CoTIR, ProtoFlow shows clear advantages on full-reference metrics in both settings, with consistent improvements of roughly 5%–30% across PSNR, SSIM, and LPIPS. On no-reference evaluation, ProtoFlow remains highly competitive: its CLIP-IQA+ is only lower than HYPIR, while still exceeding CoTIR, and its Q-Align and LIQE are also comparable to CoTIR. Fig. 3 further provides three representative restoration examples. Most previous methods fail to recover faithful colors, illumination, or fine details. Although CoTIR is competitive when degradation types are given, it sometimes degrades noticeably without explicit degradation information. For example, in the first two cases without given degradation types, CoTIR shows unsatisfactory color recovery and illumination enhancement, while in the third case it produces blurred details and erroneous restoration under both prompt settings. In contrast, ProtoFlow achieves similarly satisfying results in both settings, benefiting from prototype-prior injection to guide the restoration. Moreover, ProtoFlow reduces the restoration process from five steps in CoTIR to a single step.

Results on More Real-World Scenes. Fig. 4 compares ProtoFlow with four generative baselines on three real-world cases. DA-CLIP and LucidFlux often fail to fully remove degradations, leaving residual artifacts with oversmoothing or abnormal colors. HYPIR performs better, but still fails on the third hazy case and cannot completely remove the degradations in the first two cases, where specular reflection and JPEG artifacts remain visible. CoTIR-4B gives stronger overall results, but under vague prompts it may fail to remove reflections sufficiently in the first case, and in both prompt

Table 2: **Ablation study on prototype priors.** "Deg." and "Sem." denote degradation and semantic prototypes. We report PSNR, SSIM, and LPIPS under vague and precise prompts.

Deg.	Sem.	Vague Prompt			Precise Prompt		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
		25.02	0.8518	0.0975	25.35	0.8562	0.0943
	✓	25.65	0.8638	0.0935	25.88	0.8651	0.0932
✓		25.80	0.8656	0.0929	26.11	0.8700	0.0897
✓	✓	26.34	0.8741	0.0836	26.62	0.8764	0.0818

Table 3: **Selection of default restoration strength.** We report PSNR, SSIM, and LPIPS under vague and precise prompts.

t	Vague Prompt			Precise Prompt		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
0.10	26.09	0.8683	0.0879	26.40	0.8714	0.0857
0.25	26.34	0.8741	0.0836	26.62	0.8764	0.0818
0.50	26.08	0.8709	0.0878	26.40	0.8736	0.0855
0.75	25.86	0.8686	0.0888	26.19	0.8719	0.0860
0.90	25.81	0.8662	0.0893	26.05	0.8695	0.0863

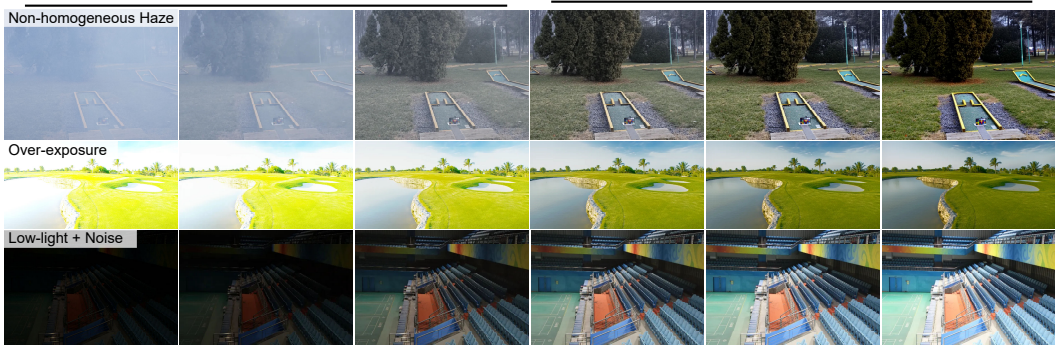


Figure 6: **Strength control via timestep adjustment.**

settings it still suffers from hallucinated artifacts or over-smoothing in the later two cases. In contrast, ProtoFlow delivers more stable restoration in both vague and precise prompt settings. It removes degradations more thoroughly while preserving scene structures and avoids the artifacts observed in previous generative methods. These comparisons suggest that the proposed prototype priors provide more reliable restoration guidance on complex real-world degradations.

5.3 Ablation Study

Selection of Standard Restoration Strength. Table 3 studies the default timestep t used to train the restoration model. We observe that $t = 0.25$ gives the best overall performance in both prompt settings. When $t = 0.10$, the training state is too close to the clean target, leaving limited room for effective correction and making it harder to control weaker restoration strength. In contrast, larger timesteps such as $t = 0.75$ and $t = 0.90$ place the starting point closer to random noise, lengthen the transport path, and require substantially stronger one-step adaptation, which increases finetuning difficulty and degrades fidelity. Overall, $t = 0.25$ provides the most suitable trade-off between learnability and restoration strength, and is used as the default setting.



Figure 5: **Runtime breakdown of ProtoFlow and CoTIR.**

Effectiveness of Prototype Priors. Table 2 validates the contribution of the two prototype banks. Starting from the baseline without prototype priors, adding only semantic prototypes already improves restoration, but adding only degradation prototypes yields larger gains in both vague and precise prompt settings. For example, under vague prompts, PSNR increases from 25.02 dB to 25.65 dB with only semantic prototypes, and further to 25.80 dB with only degradation prototypes. This shows that degradation prototypes play the more important role, while semantic prototypes provide complementary content cues. Combining both prototype priors gives the best results in all cases, improving PSNR over the baseline by 1.32 dB under vague prompts and 1.27 dB under precise prompts. These results confirm that the degradation and semantic prototype banks are complementary.

5.4 Discussion

Efficiency of One-step Restoration. Table 1 shows the efficiency advantage of ProtoFlow over multi-step generative baselines. Compared with the 5-step CoTIR-4B baseline, ProtoFlow reduces

the runtime from 4.602s to 0.594s on 1024×1024 images, yielding a $7.7\times$ speedup, while achieving better full-reference results and comparable no-reference results. Fig. 5 further shows the runtime breakdown. The prototype representation stage takes only 0.005s, since the prototype embeddings are stored as pretrained priors and inference only requires lightweight feature extraction and prototype selection. The transformer stage is reduced from 4.356s to 0.343s by replacing 5-step sampling with one-step restoration and using explicit prototype priors instead of the implicit signal from CoT adapter.

Prototype Representation. Table 4 reports the quantitative analysis of degradation and semantic prototype representations on the CoTIR-Bench test set. Specifically, DegConf and DegHit evaluate degradation confidence and hit accuracy, SemCover and SemDiff evaluate semantic consistency and inter-sample distinctiveness, and DegAct and SemAct evaluate the utilization of the degradation and semantic prototype banks.[1] For the degradation branch, the four composite-degradation cases in Fig. 7 show that the highest predicted probabilities consistently match the labeled degradations and remain clearly higher than other degradations, which is consistent with the strong DegConf and DegHit results. For the semantic branch, each low-quality image is much closer to its paired high-quality target than to other low-quality images, and the selected semantic prototypes also exhibit substantial overlap, consistent with SemCover and SemDiff. Moreover, the high DegAct and SemAct values indicate broad prototype activation rather than collapse to a few repeatedly selected prototypes, showing that the learned representation is both accurate and well utilized.

Table 4: **Quantitative analysis of degradation and semantic prototype representations.**

Metric	Value
DegConf↑	0.5844
DegHit↑	0.9140
DegAct↑	0.7105
SemCover↑	0.6518
SemDiff↑	0.9797
SemAct↑	0.9635

Restoration-strength Control. ProtoFlow also offers user-friendly control over restoration strength through the timestep t. As illustrated in Fig. 6, we visualize three representative cases, including non-homogeneous haze, over-exposure, and low-light with noise, under five strength levels $t \in \{0.05, 0.15, 0.25, 0.35, 0.45\}$. The results show that varying t yields a continuous adjustment of restoration behavior, providing a simple control variable over the restoration trajectory. In particular, the default setting $t = 0.25$ already achieves strong visual quality, while larger values lead to stronger correction and smaller values produce milder restoration, making the model practical for user-preferred enhancement.

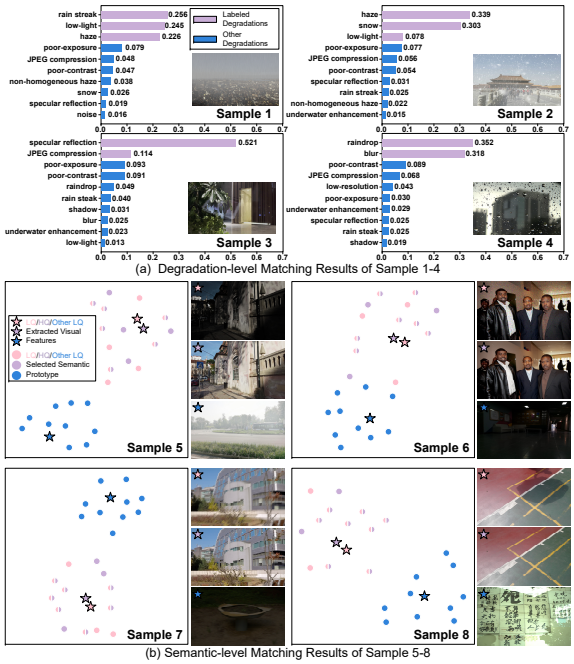


Figure 7: **Qualitative examples of degradation and semantic prototype matching.**

6 Conclusion

In this paper, we have presented ProtoFlow, a prototype-guided one-step flow model for universal image restoration. ProtoFlow reformulates restoration as a one-step transport from degraded latent to the clean manifold, and uses Gaussian degradation and semantic prototypes to provide structured prior guidance. Experiments on CoTIR-Bench show that ProtoFlow achieves superior full-reference quality, competitive no-reference performance, and much higher efficiency than strong flow-based baselines. Despite this improvement, ProtoFlow still faces two limitations: the 4B backbone remains costly for lightweight devices, and under extremely severe degradations the generative prior may produce plausible but inaccurate details. Future work will explore lighter backbones and stronger fidelity constraints.

[1]Their detailed definitions are provided in Appendix B.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing, drafting and revising parts of the manuscript, literature retrieval, and discussions of research methodology, experimental design, and result interpretation. We have not used generative AI tools for generating core technical claims, or producing the final figures and quantitative results presented in this paper. All AI-assisted content was independently reviewed, verified, and revised by the authors. The authors take full responsibility for the final content of this work, including all text, technical claims, experimental results, and conclusions.

ETHICS STATEMENT

This work does not involve human subjects, private user data, or sensitive personal information. The datasets and experimental settings used in this paper are intended for academic research purposes. We have not identified potentially harmful insights, discriminatory applications, or security risks directly arising from the proposed method. The authors take responsibility for ensuring that the methods and results presented in this work are used in a responsible and ethically appropriate manner.

REPRODUCIBILITY STATEMENT

To facilitate reproduction, we provide detailed descriptions of the proposed model architecture, training objectives, and optimization procedure in the main text. Dataset preprocessing, implementation details, hyperparameter settings, and additional experimental results are provided in the appendix and supplementary materials.

REFERENCES

Yuang Ai, Xiaoqiang Zhou, Huaibo Huang, Xiaotian Han, Zhengyu Chen, Quanzeng You, and Hongxia Yang. Dreamclear: High-capacity real-world image restoration with privacy-safe dataset curation. *NeurIPS*, 37:55443–55469, 2024.

Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv e-prints*, pp. arXiv–2506, 2025.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *NeurIPS*, 32, 2019.

Marcos V Conde, Gregor Geigle, and Radu Timofte. Instructir: High-quality image restoration following human instructions. In *ECCV*, pp. 1–21. Springer, 2024.

Jon Donnelly, Alina Jade Barnett, and Chaofan Chen. Deformable protopnet: An interpretable image classifier using deformable prototypes. In *CVPR*, pp. 10265–10275, 2022.

Song Fei, Tian Ye, Lujia Wang, and Lei Zhu. Lucidflux: Caption-free universal image restoration via a large-scale diffusion transformer. In *ICLR*, 2026.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *ICLR*, 2025.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *NeurIPS*, 2025.

Yu Guo, Yuan Gao, Yuxu Lu, Huilin Zhu, Ryan Wen Liu, and Shengfeng He. Onerestore: A universal restoration framework for composite degradation. In *ECCV*, pp. 255–272. Springer, 2024.

Yu Guo, Zhengru Fang, Shengfeng He, Senkang Hu, Yihang Tao, Phone Lin, and Yuguang Fang. Universal image restoration via internalized chain-of-thought reasoning. *arXiv preprint arXiv*, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33: 6840–6851, 2020.

Ashutosh Kulkarni, Shruti S Phutke, and Subrahmanyam Murala. Unified transformer network for multi-weather image restoration. In *ECCV*, pp. 344–360, 2022.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. `https://bfl.ai/blog/flux-2`, 2025.

Boyun Li, Xiao Liu, Peng Hu, Zhongqin Wu, Jiancheng Lv, and Xi Peng. All-in-one image restoration for unknown corruption. In *CVPR*, pp. 17452–17462, 2022.

Xinqi Lin, Fanghua Yu, Jinfan Hu, Zhiyuan You, Wu Shi, Jimmy S Ren, Jinjin Gu, and Chao Dong. Harnessing diffusion-yielded score priors for image restoration. *IEEE TOG*, 44(6):1–21, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

Chiyu Ma, Jon Donnelly, Wenjun Liu, Soroush Vosoughi, Cynthia Rudin, and Chaofan Chen. Interpretable image classification with adaptive prototype-based vision transformers. *NeurIPS*, 37: 41447–41493, 2024.

Steffen Meinert, Philipp Schlinge, Nils Strodthoff, and Martin Atzmueller. Protomask: Segmentation-guided prototype learning. In *ICTAI*, pp. 881–888. IEEE, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *CVPR*, pp. 4195–4205, 2023.

Vaishnav Potlapalli, Syed Waqas Zamir, Salman H Khan, and Fahad Shahbaz Khan. PromptIR: Prompting for all-in-one image restoration. *NeurIPS*, 36, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10684–10695, 2022.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *NeurIPS*, 30, 2017.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Chong Wang, Yuanhong Chen, Fengbei Liu, Yuyuan Liu, Davis James McCarthy, Helen Frazer, and Gustavo Carneiro. Mixture of gaussian-distributed prototypes with generative modelling for interpretable and trustworthy image recognition. *IEEE TPAMI*, 2025.

Yurui Zhu, Tianyu Wang, Xueyang Fu, Xuanyu Yang, Xin Guo, Jifeng Dai, Yu Qiao, and Xiaowei Hu. Learning weather-general and weather-specific features for image restoration under multiple adverse weather conditions. In *CVPR*, pp. 21747–21758, 2023.

APPENDIX

A PROTOTYPE PRETRAINING DETAILS

This section mainly introduces the active regularization losses and the EM update details used in Stage-I prototype pretraining.

A.1 REGULARIZATION LOSS

The regularization loss is defined as

$$\mathcal{L}_{\text{reg}} = \sum_{b\in\{\text{d,s}\}} \left(\lambda_b^{\text{cmp}}\mathcal{L}_b^{\text{cmp}} + \lambda_b^{\text{div}}\mathcal{L}_b^{\text{div}} + \lambda_b^{\text{ctr}}\mathcal{L}_b^{\text{ctr}}\right). \tag{20}$$

where $\mathcal{L}_b^{\text{cmp}}$, $\mathcal{L}_b^{\text{div}}$, and $\mathcal{L}_b^{\text{ctr}}$ denote the compactness, intra-group diversity, and centroid-separation losses for branch b, respectively, and λ_b^{cmp}, λ_b^{div}, and λ_b^{ctr} are their corresponding weights. Here, $b \in \{\text{d}, \text{s}\}$ denotes the degradation or semantic branch. For each bank, let $\mu_{g,n}^b$ be the n-th prototype mean in group g, and define the group centroid as

$$\bar{\mu}_g^b = \mathcal{N}\left(\frac{1}{N_b}\sum_{n=1}^{N_b}\mathcal{N}(\mu_{g,n}^b)\right). \tag{21}$$

where $\mathcal{N}$ denotes ℓ_2 normalization.

Compactness Loss. The compactness term is defined as

$$\mathcal{L}_b^{\text{cmp}} = \frac{1}{G_b N_b}\sum_{g=1}^{G_b}\sum_{n=1}^{N_b}\left(1 - \left\langle \mathcal{N}(\mu_{g,n}^b), \bar{\mu}_g^b \right\rangle\right), \tag{22}$$

This term pulls prototypes toward the centroid of their own group and encourages within-group compactness.

Intra-group Diversity Loss. The intra-group diversity term is defined as

$$\mathcal{L}_b^{\text{div}} = \frac{1}{G_b N_b(N_b-1)}\sum_{g=1}^{G_b}\sum_{n\neq m}\left(\left\langle \mathcal{N}(\mu_{g,n}^b), \mathcal{N}(\mu_{g,m}^b)\right\rangle - \delta\right)^2, \tag{23}$$

where δ is the target intra-group cosine similarity. This term prevents prototypes within the same group from collapsing to nearly identical directions and preserves diverse local modes inside each group.

Centroid-separation Loss. The centroid-separation term is defined as

$$\mathcal{L}_b^{\text{ctr}} = \frac{1}{G_b(G_b-1)}\sum_{g\neq h}\max\left(0, \left\langle \bar{\mu}_g^b, \bar{\mu}_h^b\right\rangle - m\right)^2, \tag{24}$$

where m is the centroid margin. This term suppresses excessive similarity between different group centroids and thus encourages better global separation of prototype groups.

In our Stage-I setting, all six active regularization weights are set to 0.01, i.e., $\lambda_{\text{d}}^{\text{cmp}} = \lambda_{\text{s}}^{\text{cmp}} = \lambda_{\text{d}}^{\text{div}} = \lambda_{\text{s}}^{\text{div}} = \lambda_{\text{d}}^{\text{ctr}} = \lambda_{\text{s}}^{\text{ctr}} = 0.01$.

A.2 EM-BASED PROTOTYPE UPDATE

To stabilize prototype estimation, we maintain a class-wise degradation memory bank and a global semantic memory bank, and periodically update the Gaussian prototype banks with an Expectation-Maximization (EM) procedure inspired by MGProto Wang et al. (2025).

Notation. In both algorithms, π denotes the prototype prior (mixture weight), μ denotes the prototype mean, and σ^2 denotes the diagonal variance of the Gaussian prototype. We use C for the number of degradation classes, G for the number of semantic groups, N_{d}^c for the number of prototypes in degradation class c, and N_{s} for the number of prototypes in each semantic group. The symbols $q_{\text{lq}}^{\text{d},(i)}$

Algorithm 1 EM update for the degradation prototype bank

Require: Degradation memory banks $\{\mathcal{M}_c^{\mathrm{d}}\}_{c=1}^{C}$, current degradation-bank parameters (π, μ, σ), smoothing factor α, momentum β
Ensure: Updated degradation-bank means and priors
E-step:
for each degradation class c **do**
  for each cached query $q_{\mathrm{lq}}^{\mathrm{d},(i)} \in \mathcal{M}_c^{\mathrm{d}}$ **do**
    Compute class-wise logits and responsibilities:
    $z_{c,n}^{(i)} = \log \pi_{c,n} + \log p\Big(q_{\mathrm{lq}}^{\mathrm{d},(i)} \mid \mu_{c,n}, \sigma_{c,n}^2\Big)$
    $\gamma_{c,n}^{(i)} = \mathrm{softmax}_n(z_{c,n}^{(i)})$
    $\tilde{\gamma}_{c,n}^{(i)} = (1-\alpha)\gamma_{c,n}^{(i)} + \alpha/N_{\mathrm{d}}^{c}$
  end for
  M-step:
  Run the weighted M-step with $w_i = 1$:
    $\eta_n = \sum_i w_i \tilde{\gamma}_{c,n}^{(i)}, \quad \hat{\mu}_n = \frac{\sum_i w_i \tilde{\gamma}_{c,n}^{(i)} q_{\mathrm{lq}}^{\mathrm{d},(i)}}{\eta_n}$
    $\hat{\sigma}_n = \sqrt{\frac{\sum_i w_i \tilde{\gamma}_{c,n}^{(i)} (q_{\mathrm{lq}}^{\mathrm{d},(i)} \odot q_{\mathrm{lq}}^{\mathrm{d},(i)})}{\eta_n} - \hat{\mu}_n \odot \hat{\mu}_n + \epsilon}$
    $\hat{\pi}_n = \frac{\eta_n}{\sum_{n'} \eta_{n'}},$
end for
Reweight degradation priors by class memory size and update the degradation bank with momentum β
Refine the updated means using the compactness, intra-group diversity, and centroid-separation losses
Write back $\mu \leftarrow \mathcal{N}(\beta\mu + (1-\beta)\hat{\mu})$ and keep bank scales fixed

and $q_{\mathrm{lq}}^{\mathrm{s},(i)}$ denote the i-th cached degradation and semantic queries, respectively, while w_i denotes the corresponding sample weight in the M-step. Moreover, z denotes the pre-softmax assignment logit, γ and $\tilde{\gamma}$ denote the original and smoothed prototype responsibilities, $a_{g,n}^{(i)}$ denotes the flattened semantic assignment probability, $r_g^{(i)}$ denotes the group-level routing probability, $\rho_g^{(i)}$ and $\tilde{\rho}_g^{(i)}$ denote the normalized and smoothed semantic group weights, and ϵ is a small constant for numerical stability.

Algorithms 1 and 2 summarize the degradation and semantic EM procedures, respectively. For the degradation bank, we perform a class-wise E-step on cached degradation queries, followed by a weighted M-step with uniform sample weights inside each class. For the semantic bank, the E-step contains two parts. We first compute flattened prototype assignments over all semantic prototypes to obtain the group-level routing probability $r_g^{(i)}$, and then flatten and smooth these probabilities to produce the sample weights $\tilde{\rho}_g^{(i)}$. We further compute the group-local responsibilities $\tilde{\gamma}_{g,n}^{(i)}$ under the current parameters of each semantic group. The subsequent M-step uses $\tilde{\rho}_g^{(i)}$ as sample weights and $\tilde{\gamma}_{g,n}^{(i)}$ as group-local responsibilities to update the prototypes in group g, while keeping the semantic priors uniform instead of re-estimating them from η_n.

Both branches share the same M-step update on the cached queries: weighted responsibilities are accumulated into component masses, which are then used to update the prototype means, diagonal variances, and priors. Afterward, the prototype means are further refined using the compactness, intra-group diversity, and centroid-separation losses defined above, and are written back with momentum β. The two banks differ only in how the priors are written back: degradation priors are reweighted by the relative memory size of each degradation class before global normalization, whereas semantic priors are reset to a uniform distribution after EM. Although the intermediate EM estimates include $\hat{\sigma}_n$, the current implementation keeps the bank scales fixed when writing parameters back and only updates the prototype means and priors.

This EM update stabilizes prototype estimation by aggregating cached features into weighted sufficient statistics, which reduces noisy per-mini-batch fluctuations and yields more reliable prototype means. We perform EM separately for the degradation and semantic banks because they follow different structures and supervision signals: degradation prototypes are organized by labeled degradation classes and therefore benefit from class-wise EM with class-frequency-aware priors, whereas semantic prototypes are organized by latent semantic groups induced by router assignments and therefore

Algorithm 2 EM update for the semantic prototype bank

Require: Semantic memory bank $\mathcal{M}^{\mathrm{s}}$, current semantic-bank parameters (π, μ, σ), smoothing factor α, momentum β
Ensure: Updated semantic-bank means and priors

E-step:
for each cached query $q_{\mathrm{lq}}^{\mathrm{s},(i)} \in \mathcal{M}^{\mathrm{s}}$ **do**
  Compute flattened prototype assignments over (g, n):
    $z_{g,n}^{(i)} = \log \pi_{g,n} + \log p\Big(q_{\mathrm{lq}}^{\mathrm{s},(i)} \mid \mu_{g,n}, \sigma_{g,n}^2\Big)$
    $a_{g,n}^{(i)} = \mathrm{softmax}_{(g,n)}(z_{g,n}^{(i)})$
    $r_g^{(i)} = \sum_{n=1}^{N_{\mathrm{s}}} a_{g,n}^{(i)}$
    $\rho_g^{(i)} = \frac{\sqrt{r_g^{(i)}}}{\sum_{g'=1}^{G} \sqrt{r_{g'}^{(i)}}}$, $\tilde{\rho}_g^{(i)} = (1-\alpha)\rho_g^{(i)} + \alpha/G$
end for
for each semantic group g **do**
  for each cached query $q_{\mathrm{lq}}^{\mathrm{s},(i)} \in \mathcal{M}^{\mathrm{s}}$ **do**
    Compute group-wise logits and responsibilities:
      $z_{g,n}^{(i)} = \log \pi_{g,n} + \log p\Big(q_{\mathrm{lq}}^{\mathrm{s},(i)} \mid \mu_{g,n}, \sigma_{g,n}^2\Big)$
      $\gamma_{g,n}^{(i)} = \mathrm{softmax}_n(z_{g,n}^{(i)})$
      $\tilde{\gamma}_{g,n}^{(i)} = (1-\alpha)\gamma_{g,n}^{(i)} + \alpha/N_{\mathrm{s}}$
  end for
end for
M-step:
for each semantic group g **do**
  Run the weighted M-step with $w_i = \tilde{\rho}_g^{(i)}$:
    $\eta_n = \sum_i w_i \tilde{\gamma}_{g,n}^{(i)}, \quad \hat{\mu}_n = \frac{\sum_i w_i \tilde{\gamma}_{g,n}^{(i)} q_{\mathrm{lq}}^{\mathrm{s},(i)}}{\eta_n}$
    $\hat{\sigma}_n = \sqrt{\frac{\sum_i w_i \tilde{\gamma}_{g,n}^{(i)} (q_{\mathrm{lq}}^{\mathrm{s},(i)} \odot q_{\mathrm{lq}}^{\mathrm{s},(i)})}{\eta_n} - \hat{\mu}_n \odot \hat{\mu}_n + \epsilon}$
    Keep the priors uniform within group g, i.e., $\hat{\pi}_{g,n} = 1/N_{\mathrm{s}}$
end for
Reset the semantic-bank priors to a global uniform distribution before write-back
Refine the updated means using the compactness, intra-group diversity, and centroid-separation losses
Write back $\mu \leftarrow \mathcal{N}(\beta\mu + (1-\beta)\hat{\mu})$ and keep bank scales fixed

require group-wise EM with routing-based sample weights and uniform priors to avoid early collapse toward a few dominant groups.

In our implementation, EM is performed every 20 iterations for 3 iterations with $\alpha = 0.10$ and $\beta = 0.9$. These EM updates are only used in Stage I and are disabled in Stage II.

B Prototype Representation Metrics

This section defines the six prototype-representation metrics used in our analysis. Following the notation in main text and the Gaussian matching score in Eq. equation 11, we consider an evaluation set of N samples. For sample i, we denote the ground-truth degradation label set by $\mathcal{G}^{(i)}$, the set of degradation classes appearing in the retrieved top-k results by $\mathcal{K}^{\mathrm{d},(i)}$, the full degradation-class scores by $s_c^{\mathrm{d},(i)}$, the semantic top-k index set of the low-quality image by $\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(i)}$, and that of the clean target by $\mathcal{K}_{\mathrm{gt}}^{\mathrm{s},(i)}$. Here, c indexes the degradation class, and the sample index is consistently written as a superscript. From the degradation scores, we define the full-class softmax probability

$$\phi_c^{\mathrm{d},(i)} = \frac{\exp(s_c^{\mathrm{d},(i)})}{\sum_{c'=1}^{C} \exp(s_{c'}^{\mathrm{d},(i)})}, \tag{25}$$

where C is the total number of degradation classes. The top-k normalized retrieval weight follows the soft top-k aggregation rule in Eq. (17), i.e., $\omega_c^{\mathrm{d},(i)}$ is obtained by normalizing $s_c^{\mathrm{d},(i)}$ over the retrieved class set $\mathcal{K}^{\mathrm{d},(i)}$.

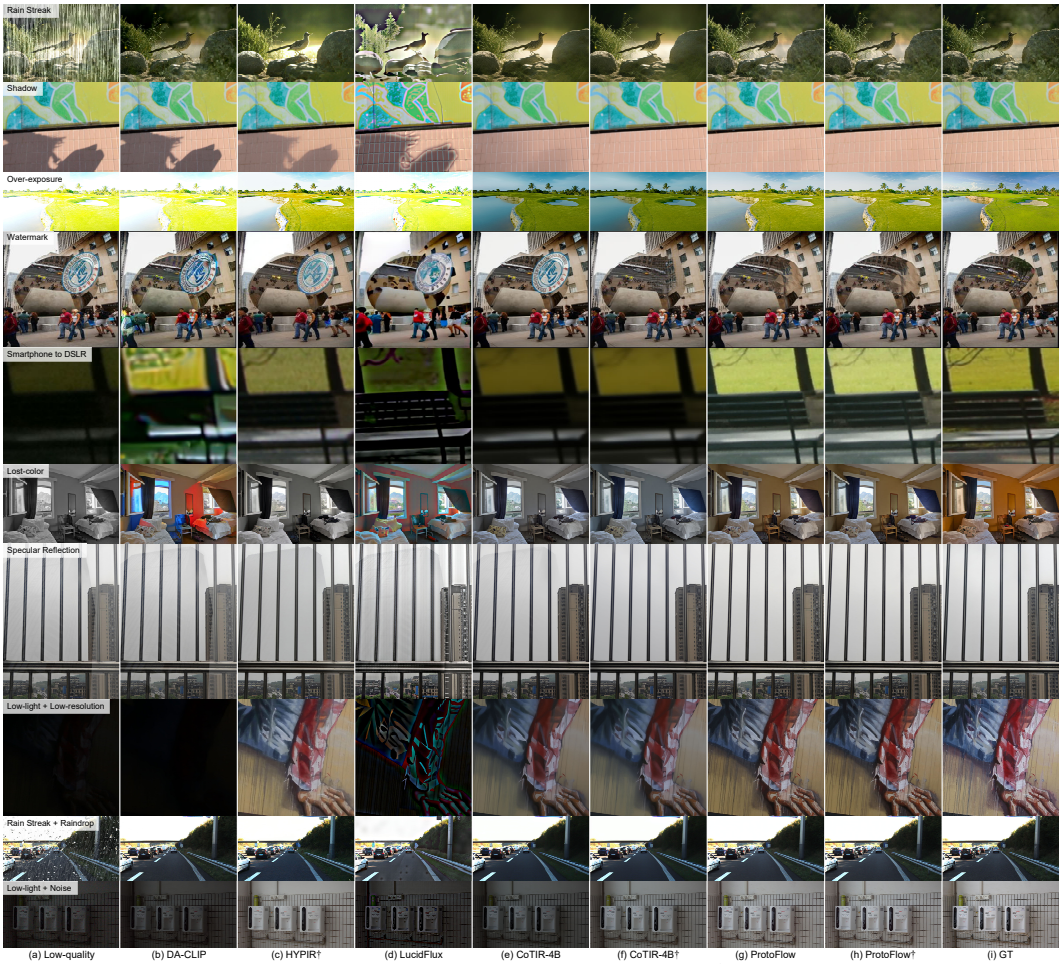


Figure 8: **Additional qualitative comparisons on CoTIR-Bench.** † means the input of degradation prompts.

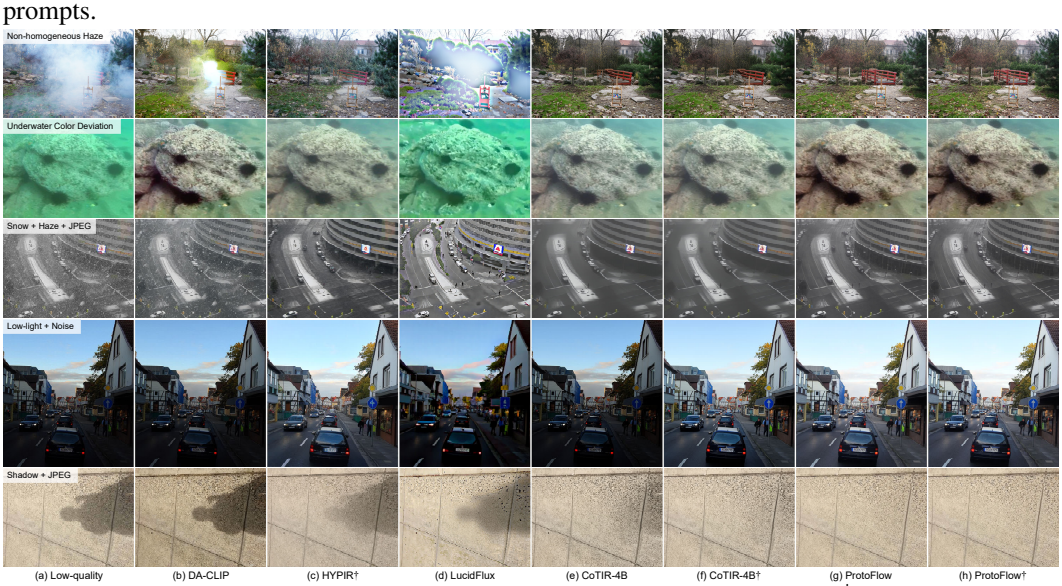


Figure 9: **Additional qualitative comparisons on real-world degradations.** † means the input of degradation prompts.

Degradation Identification Confidence (DegConf). The dataset-level degradation identification confidence is computed by averaging the full-class softmax probabilities assigned to the ground-truth

degradation labels:

$$\mathrm{DegConf} = \frac{1}{N}\sum_{i=1}^{N}\sum_{c\in\mathcal{G}^{(i)}}\phi_c^{\mathrm{d},(i)}. \tag{26}$$

This metric measures the estimated confidence assigned to the ground-truth degradations over the full 19-class degradation space.

Degradation Top-k Hit Rate (DegHit). We further evaluate whether every ground-truth degradation class is covered by the retrieved top-k results with a sufficiently large normalized weight. For sample i, we define

$$H^{\mathrm{d},(i)} = \begin{cases} 1, & \forall c \in \mathcal{G}^{(i)},\ c \in \mathcal{K}_{\mathrm{d}}^{(i)} \text{ and } \omega_c^{\mathrm{d},(i)} > 0.05, \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

The dataset-level hit rate is

$$\mathrm{DegHit} = \frac{1}{N}\sum_{i=1}^{N} H^{\mathrm{d},(i)}. \tag{28}$$

This metric measures whether the estimated retrieval successfully covers all ground-truth degradation classes with valid top-k weights.

Degradation Prototype Activation Rate (DegAct). Let $\mathcal{A}^{\mathrm{d}}$ denote the union of all degradation prototype indices activated by the N evaluation samples, and let P^{d} be the total number of degradation prototypes in the bank. We define

$$\mathrm{DegAct} = \frac{|\mathcal{A}^{\mathrm{d}}|}{P^{\mathrm{d}}}. \tag{29}$$

This metric measures how broadly the degradation prototype bank is utilized over the evaluation set.

Semantic LQ/GT Coverage (SemCover). The dataset-level semantic coverage is defined as

$$\mathrm{SemCover} = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(i)} \cap \mathcal{K}_{\mathrm{gt}}^{\mathrm{s},(i)}\right|}{\max\left(1, \min\left(|\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(i)}|, |\mathcal{K}_{\mathrm{gt}}^{\mathrm{s},(i)}|\right)\right)}. \tag{30}$$

This metric measures how consistently the semantic router selects similar prototype sets for the degraded image and its clean counterpart.

Semantic LQ-LQ Difference (SemDiff). To assess inter-sample distinctiveness, for each sample i, we randomly sample another low-quality image $j \neq i$ using a fixed random seed. The dataset-level semantic difference is then defined as

$$\mathrm{SemDiff} = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(i)} \cap \mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(j)}\right|}{\max\left(1, \min\left(|\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(i)}|, |\mathcal{K}_{\mathrm{lq}}^{\mathrm{s},(j)}|\right)\right)}. \tag{31}$$

This metric measures how well the semantic router distinguishes unrelated low-quality images.

Semantic Prototype Activation Rate (SemAct). Let $\mathcal{A}^{\mathrm{s}}$ denote the union of all semantic prototype indices activated on the evaluation set, and let P^{s} be the total number of semantic prototypes. We define

$$\mathrm{SemAct} = \frac{|\mathcal{A}^{\mathrm{s}}|}{P^{\mathrm{s}}}. \tag{32}$$

This metric measures how broadly the semantic prototype bank is utilized during evaluation.

Summary. The above six metrics jointly evaluate three aspects of prototype representation behavior: degradation confidence allocation (*DegConf*, *DegHit*), semantic consistency and distinctiveness (*SemCover*, *SemDiff*), and prototype-bank utilization (*DegAct*, *SemAct*). For all six metrics, higher values indicate better prototype representations.

Table 5: **Comparison between conventional and Gaussian prototypes on restoration performance under vague prompts.**

Prototype Type	PSNR↑	SSIM↑	LPIPS↓
Conventional Prototype	25.77	0.8540	0.0865
Gaussian Prototype	26.34	0.8741	0.0836

Table 6: **Selection of LoRA rank.** We report PSNR, SSIM, and LPIPS under vague and precise prompts.

Rank	Vague Prompt			Precise Prompt		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
64	25.87	0.8679	0.0905	26.00	0.8690	0.0891
128	26.07	0.8695	0.0882	26.30	0.8714	0.0865
256	26.34	0.8741	0.0836	26.62	0.8764	0.0818
512	26.50	0.8705	0.0802	26.66	0.8720	0.0788

C More Experimental Results

C.1 Additional Visual Comparisons

This subsection provides additional visual comparisons. We compare DA-CLIP, HYPIR†, LucidFlux, CoTIR, CoTIR†, ProtoFlow, and ProtoFlow† to further examine restoration quality and prompt robustness beyond the main text.

Fig. 8 shows additional visual comparisons on CoTIR-Bench, while Fig. 9 presents additional results on real-world degradations. Across both figures, ProtoFlow and ProtoFlow† achieve the best overall visual quality and remain highly consistent across prompt settings, indicating that the proposed prototype priors provide robust restoration guidance even without explicit degradation prompts. In contrast, CoTIR exhibits a clearer gap between the vague and precise prompt settings, since degradation misclassification under vague prompts may lead to incomplete restoration and thus inferior results to CoTIR†. The other baselines, including DA-CLIP, HYPIR†, and LucidFlux, are generally less stable and more prone to residual degradations, over-smoothing, or color artifacts on both benchmark and real-world cases.

C.2 Effectiveness of Gaussian Prototype Learning

This part analyzes the effect of Gaussian prototype learning on restoration performance under vague prompts. Table 5 directly compares conventional prototypes, which use only fixed prototype vectors, with Gaussian prototypes in the setting where restoration relies more directly on learned prototype representations. Replacing conventional prototypes with Gaussian prototypes improves all three metrics, increasing PSNR from 25.77 dB to 26.34 dB and SSIM from 0.8540 to 0.8741, while reducing LPIPS from 0.0865 to 0.0836. These results indicate that Gaussian prototypes provide more informative and robust degradation priors, which help the model better capture latent degradation characteristics and thus produce higher-quality restoration when explicit degradation prompts are unavailable.

C.3 Selection of LoRA Rank.

Table 6 analyzes the effect of LoRA rank on restoration performance. Increasing the rank from 64 to 128 and 256 consistently improves all metrics, while further increasing it to 512 yields slightly better PSNR and LPIPS but degrades SSIM in both prompt settings. Considering this trade-off, we choose rank 256 as the default setting, since it provides the best overall balance across the three metrics while remaining more effective than rank 512.