A Appendix

A.1 Social Impact

Diffusion models have experienced rapid advancements and have shown the merits of generating high-quality data. However, concerns have arisen due to their ability to memorize training data and generate inappropriate content, thereby negatively affecting the user experience and society as a whole. Machine unlearning emerges as a valuable tool for correcting the algorithms and enhancing user trust in the respective platforms. It demonstrates a commitment to responsible AI and the welfare of its user base. However, while unlearning protects privacy, it may also hinder the ability of relevant systems and potentially lead to biased outcomes.

A.2 Implementation details

Four and five feature map resolutions are adopted for CIFAR10 where image resolution is 32×32, UTKFace and CelebA where image resolution is scaled to 64×64, respectively. Our 32×32 model, and 64×64 model have around 36 million, and 79 million parameters, respectively. The well-trained unconditional DDPM models on CIFAR10[3] and CelebA-HQ[4] are downloaded from Hugging Face. We used A40 and A100 for all experiments. All models apply the linear schedule for the diffusion process. We set the batch size $B = 512$, $B = 128$, $B = 64$, $B = 16$ for CIFAR10, UTKFace and CelebA, CelebA-HQ respectively. The linear schedule is set from $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$, the inference time step for DDIM is set to be 100, the guidance scale $w = 0.1$, and the probability $p_{uncond} = 0.1$ for all models. For the Unscrubbed and Retrain models, the learning rate is 3×10^{-4} for CIFAR10 and 2×10^{-4} for other datasets. We train the CIFAR10 model for 2000 epochs, the UTKFace and CelebA models for 500 epochs. For Finetune models, the learning rate is 3×10^{-4} for CIFAR10 and 2×10^{-4} for other datasets, all the models are finetuned on the remaining data $\mathcal{D}_r$ for 100 epochs. For NegGrad models, the learning rate is 1×10^{-6} and all the models are trained on the forgetting data $\mathcal{D}_f$ for 5 epochs. For BlindSpot models, the learning rate is 2×10^{-4}. The partially-trained model is trained for 100 epochs on the remaining data $\mathcal{D}_r$ and then the scrubbed model is trained for 100 epochs on the data $\mathcal{D}$. For our scrubbed models, $N_{rs} = |\mathcal{D}_{rs}| \approx 8K$, the learning rate is 1×10^{-6} CelebA-HQ and 2×10^{-4} for other datasets. Note that the components (Dhariwal & Nichol, 2021; Nichol & Dhariwal, 2021) for improving the model performance are not applied in this work.

Algorithm 2 *EraseDiff*.

Input: Well-trained model $\epsilon_{\boldsymbol{\theta}_0}$, forgetting data $\mathcal{D}_f$ and subset of remaining data $\mathcal{D}_{rs} \subset \mathcal{D}_r$, outer iteration number S and inner iteration number K, learning rate ζ and hyparameter λ.
Output: Parameters $\boldsymbol{\theta}^*$ for the scrubbed model.

```
for iteration s in S do
    $\phi_s^0 = \boldsymbol{\theta}_s$.
    Get $\phi_s^K$ by $K$ steps of gradient descent on $f(\phi_s, \mathcal{D}_f)$ start from $\phi_s^0$ using Eq. (8):
        Sample $\{\mathbf{x}_0, c\} \subset \mathcal{D}_f, t \sim \text{Uniform}(1, \cdots, T), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$,
        Compute $\nabla_{\phi_s^k} \|\hat{\boldsymbol{\epsilon}} - \epsilon_{\phi_s^k}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t, c)\|$.
        Get the constant loss $\mathcal{L}_{cs} = \|\hat{\boldsymbol{\epsilon}} - \epsilon_{\phi_s^K}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t, c)\|$ if $k = K$.
    Set the approximation:
        Sample $\{\mathbf{x}_0, c\} \subset \mathcal{D}_f, t \sim \text{Uniform}(1, \cdots, T), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$,
        Compute the loss $\mathcal{L}_f = \|\hat{\boldsymbol{\epsilon}} - \epsilon_{\boldsymbol{\theta}_s}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t, c)\| - \mathcal{L}_{cs}$.
    Update the model:
        Sample $\{\mathbf{x}_0, c\} \subset \mathcal{D}_{rs}, t \sim \text{Uniform}(1, \cdots, T), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$,
        Compute the loss $\mathcal{L}_s = \|\boldsymbol{\epsilon} - \epsilon_{\boldsymbol{\theta}_s}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t, c)\| + \lambda\mathcal{L}_f$,
        Update $\boldsymbol{\theta}_{s+1} = \boldsymbol{\theta}_s - \zeta\nabla_{\boldsymbol{\theta}_s}\mathcal{L}_s$.
end for
```

[3] https://huggingface.co/google/ddpm-cifar10-32

[4] https://huggingface.co/google/ddpm-ema-celebahq-256

Algorithm 3 BlindSpot Unlearning (Tarun et al., 2023b).

```
Input: A well-trained model $\epsilon$ with parameters $\boldsymbol{\theta}_0$, a randomly initialized blind model $\epsilon_{\boldsymbol{\psi}}(\cdot)$, forgetting data $\mathcal{D}_f$, remaining data $\mathcal{D}_r$ and all training data $\mathcal{D} = \mathcal{D}_f \cup \mathcal{D}_r$. The learning rate $\zeta$, the number of epochs $E_r$ and $E_u$, hyper-parameter $\lambda$.
Output: Parameters $\boldsymbol{\theta}^*$ for the scrubbed model.
Initialization $\boldsymbol{\theta} = \boldsymbol{\theta}_0$.
for $1, 2, \ldots, E_r$ do
    train the blind model $\epsilon_{\boldsymbol{\psi}}(\cdot)$ with the remaining data $\mathcal{D}_r$.
end for
for $1, 2, \ldots, E_u$ do
    for $(\mathbf{x}_i, c_i) \in \mathcal{D}$ do
        $l_f^i = 1$ if $(\mathbf{x}_i, c_i) \in \mathcal{D}_f$ else $l_f^i = 0$.
        $\boldsymbol{\epsilon}_t = \epsilon_{\boldsymbol{\theta}}(\mathbf{x}_i, t, c_i)$, where $t$ is the timestep and $t \in [1, T]$.
        $\mathcal{L}_r = \mathcal{L}(\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon})$ and $\mathcal{L}_f = \mathcal{L}(\boldsymbol{\epsilon}_t, \epsilon_{\boldsymbol{\psi}}(\mathbf{x}_i, t, c_i))$.
        $\mathcal{L}_a = \lambda \sum_{j=1}^{k} \|\text{act}_j^{\boldsymbol{\theta}} - \text{act}_j^{\boldsymbol{\psi}}\|$, where $\text{act}_j$ is the output of each block in the UNet.
        $\mathcal{L} = (1 - l_f^i)\mathcal{L}_r + l_f^i(\mathcal{L}_f + \mathcal{L}_a)$.
        $\boldsymbol{\theta} = \boldsymbol{\theta} - \zeta \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$.
    end for
end for
```

A.3 More Results

In the following, we present the results of Ablation studies, results when replacing $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ with $\hat{\boldsymbol{\epsilon}}_t \sim \mathcal{N}(\mathbf{0.5}, \mathbf{I}_d)$ for Eq. (4), results when sampling from the uniform distribution $\hat{\boldsymbol{\epsilon}}_t \sim \mathcal{U}(\mathbf{0}, \mathbf{1})$, and results when trying to erase different classes/races/attributes under the conditional and unconditional scenarios. We include new comparisons (e.g., without access to $\mathcal{D}_r$, subjected to adversarial attacks, two alternative formulations to perform unlearning) in Tabs. 4 to 8 and Figs. 17 to 24. In general, with more remaining data during the unlearning process, the generated image quality over the remaining classes $\mathcal{C}_r$ would be better while those over the forgetting classes $\mathcal{C}_f$ would be worse. With generated images for unlearning, the image quality after scrubbing the model would be worse, but still surpasses other methods. When subjected to adversarial attacks, the quality of generated images of all models would decrease along with the step size of the attack increases, but the scrubbed model still would not contain information about $\mathcal{C}_f$. Simultaneously updating the model parameters can destroy the information about $\mathcal{C}_f$, but would also result in a significant drop in image quality $\mathcal{C}_r$. Disjoint optimization does not work as the second phase could bring back information about $\mathcal{C}_f$.

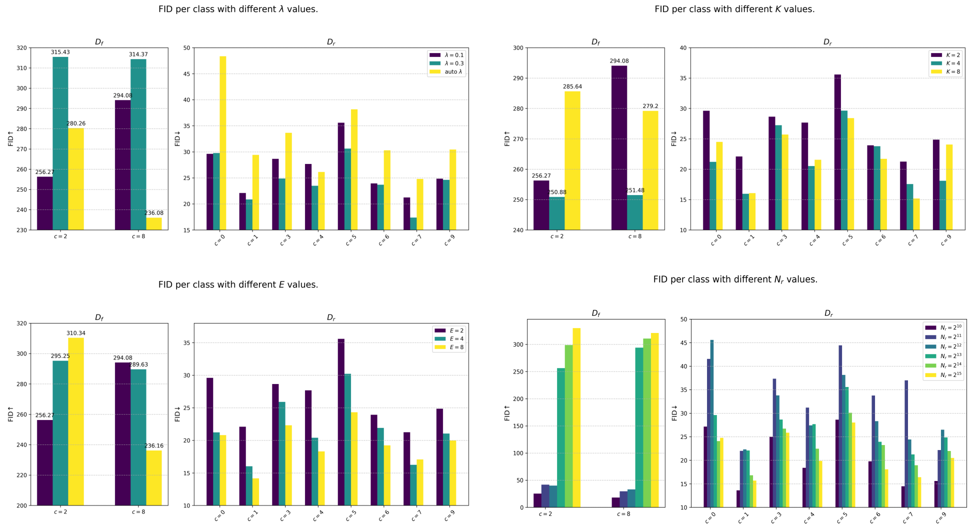

Figure 6: Ablation results with conditional DDIM on CIFAR10.

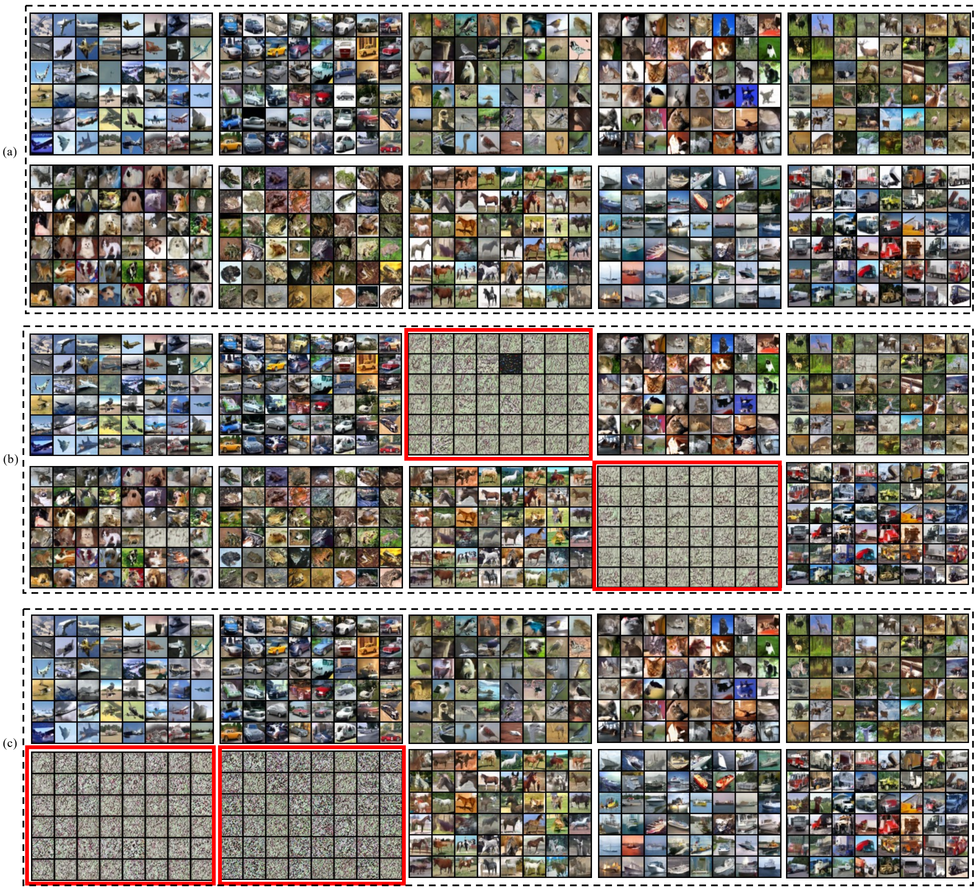

Figure 7: Conditional DDIM on CIFAR-10. (a) Generated images by the unscrubbed model. (b) and (c) are generated images by our scrubbed model when forgetting classes are $\mathcal{C}_f = \{c_2, c_8\}$, and $\mathcal{C}_f = \{c_5, c_6\}$, respectively. Images in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$, others are generated by conditioning on the remaining classes $\mathcal{C}_r$.

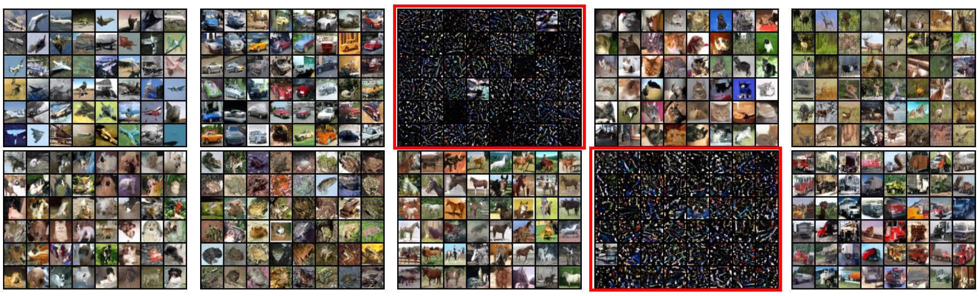

Figure 8: Images generated by our scrubbed conditional DDIM on CIFAR10 when we choose normal distribution $\hat{\epsilon}_t \sim \mathcal{N}(\mathbf{0.5}, \mathbf{I}_d)$. Images in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$, others are generated by conditioning on the remaining classes $\mathcal{C}_r$.

Figure 9: Images generated by our scrubbed conditional DDIM on CIFAR10. Sampling with noise from the uniform distribution $\hat{\mathbf{x}}_T \sim \mathcal{U}(\mathbf{0}, \mathbf{1})$.

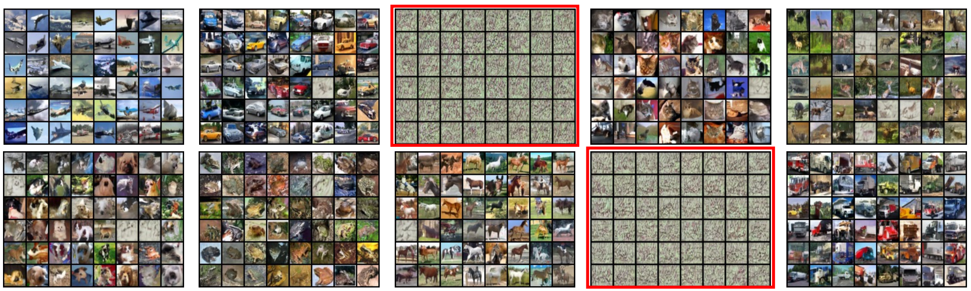

Figure 10: Images generated by our scrubbed conditional DDIM on CIFAR10 when $p_{uncond} \approx 0$.

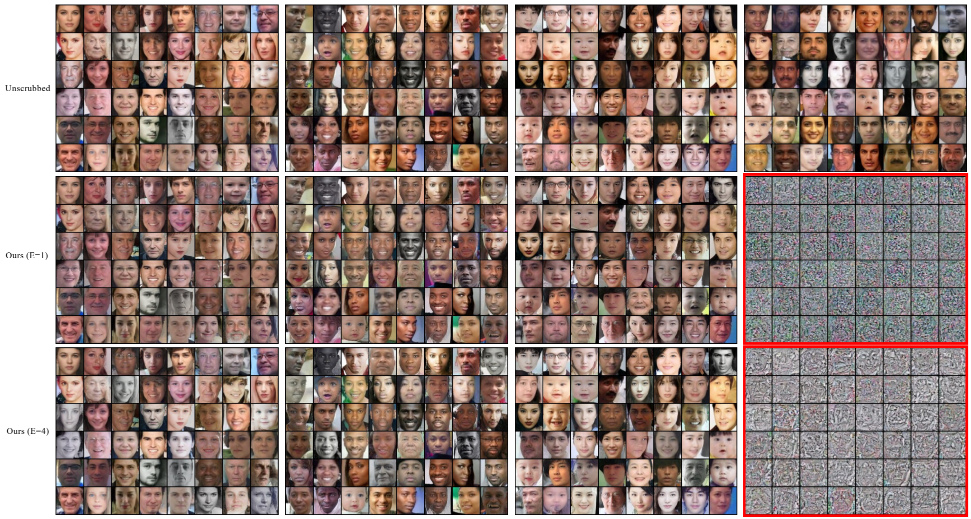

Figure 11: Images generated by conditional DDIM on UTKFace with different hyper-parameter E. Images in the red solid box are generated by conditioning on $\mathcal{C}_f$, others are generated by conditioning on $\mathcal{C}_r$. The larger the number E, the better the quality of generated images over $\mathcal{C}_r$.

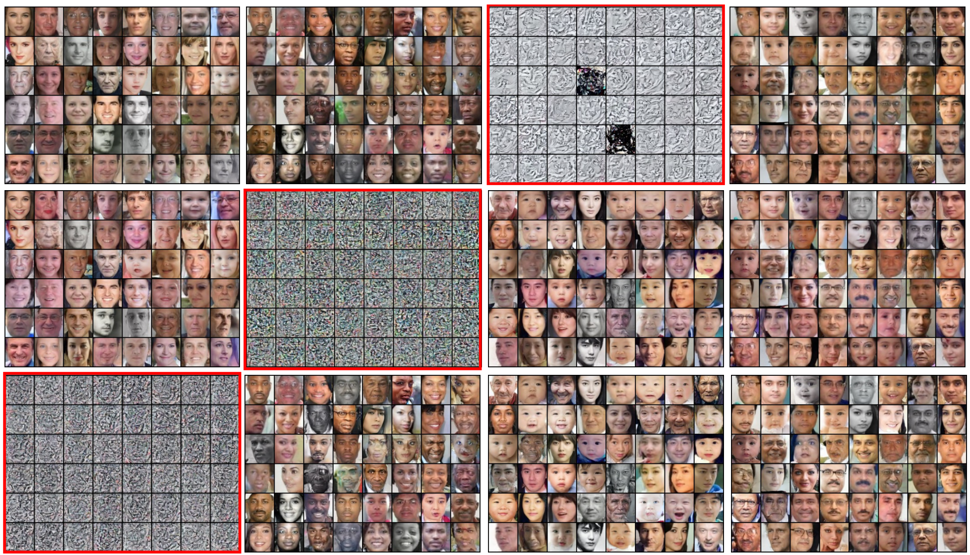

Figure 12: Images generated by our scrubbed conditional DDIM when unlearning different races (Top to Bottom: unlearning Asian, Black, and White, respectively). Images in the red solid box are generated by conditioning on $\mathcal{C}_f$, others are generated by conditioning on $\mathcal{C}_r$.

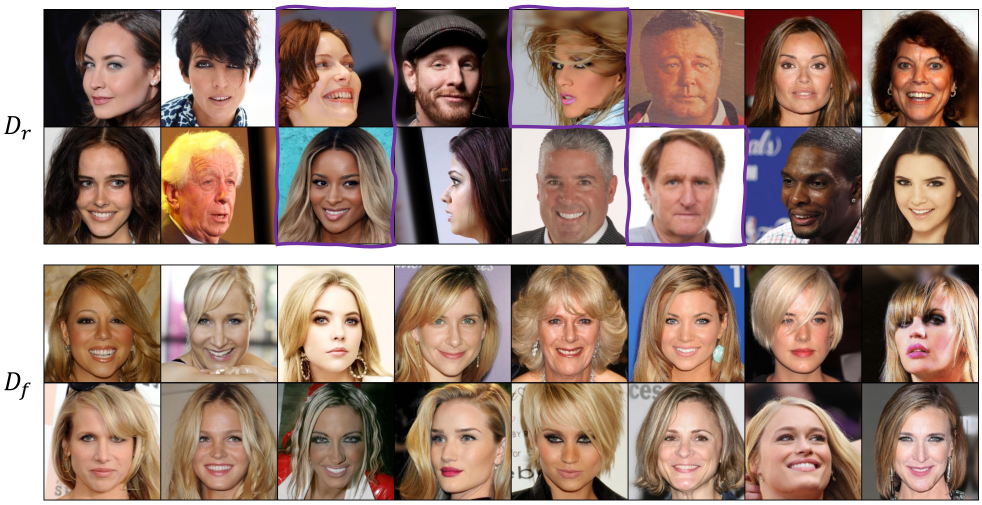

Figure 13: Examples from the remaining data $\mathcal{D}_r$ and forgetting data (Blond hair attribute) $\mathcal{D}_f$ on CelebA-HQ. Note that some examples in $\mathcal{D}_r$ (e.g., images in the purple solid box) have hair with a color that looks similar to the Blond hair attribute.

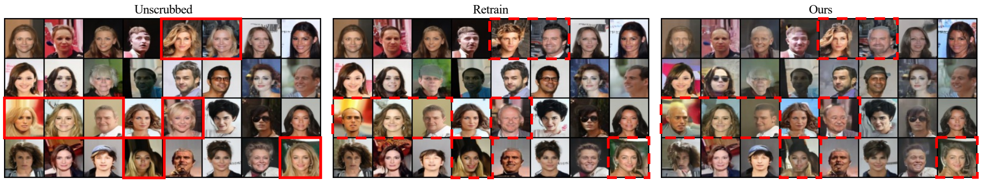

Figure 14: Images generated by unconditional DDIM on CelebA. We aim to unlearn the attribute of blond hair. Our unlearning algorithm obtains the results quite similar to those from the retrained model which is trained with the remaining data. Note that some images from the remaining data have hair attribute that looks like blond hair attribute as shown in Fig. 13

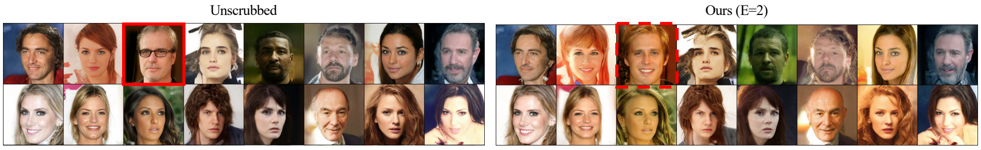

Figure 15: Images generated by the well-trained unconditional DDPM models from Hugging Face and our corresponding scrubbed models on CelebA-HQ. We aim to unlearn the Eyeglasses attribute.

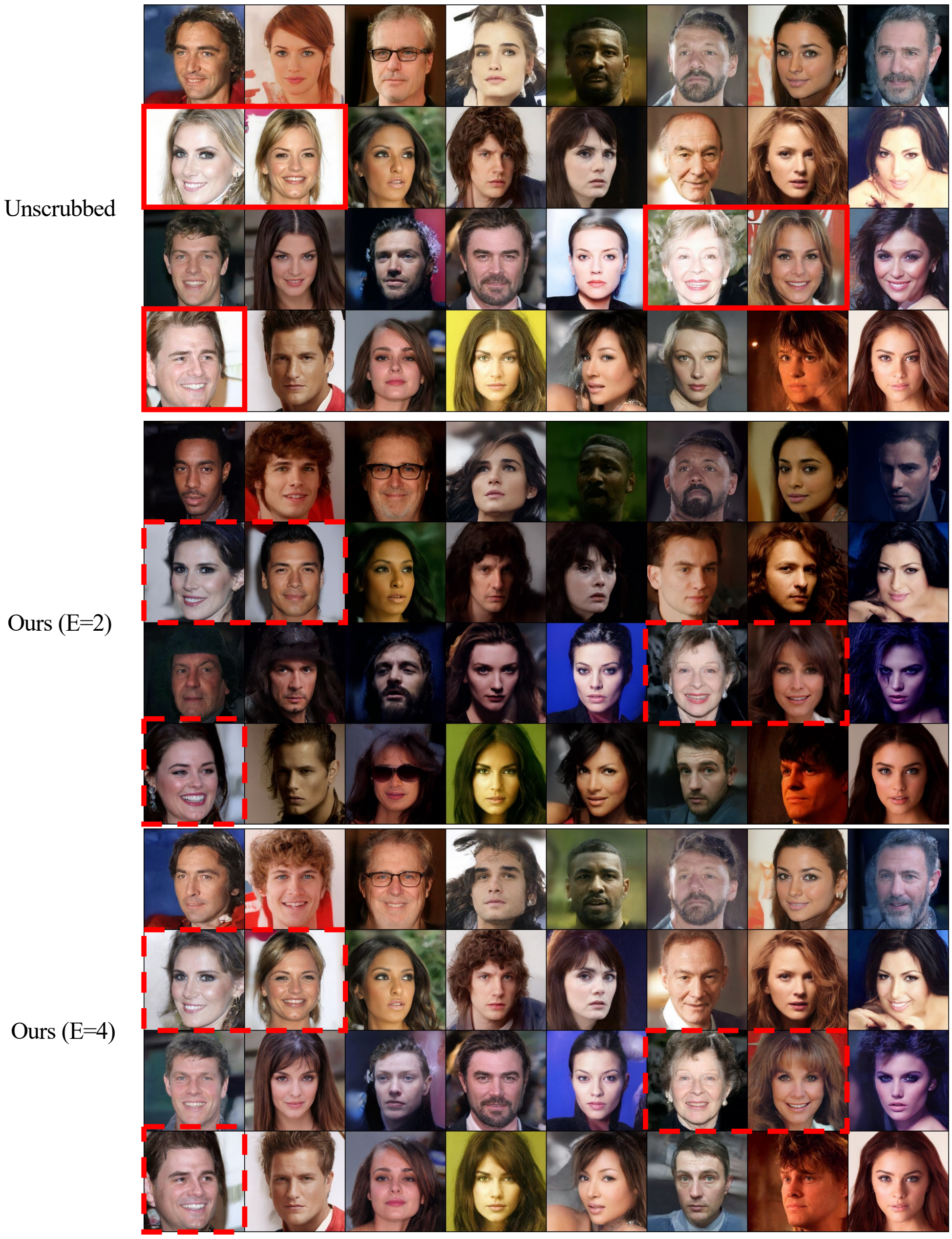

Figure 16: Images generated by the well-trained unconditional DDPM models from Hugging Face and our corresponding scrubbed models on CelebA-HQ. We aim to unlearn the Blond hair attribute.

Table 4: Results on CIFAR-10 with conditional DDIM, compared with simultaneously optimizing $\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_r) - \alpha\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_f)$ (denoted as SO). Generated examples are shown in Fig. 17. SO cannot achieve a good trade-off between erasing the influence of $\mathcal{D}_f$ and preserving model utility over $\mathcal{D}_r$.

Method	FID over forgetting classes		FID over remaining classes							
	$c = 2 \uparrow$	$c = 8 \uparrow$	$c = 0 \downarrow$	$c = 1 \downarrow$	$c = 3 \downarrow$	$c = 4 \downarrow$	$c = 5 \downarrow$	$c = 6 \downarrow$	$c = 7 \downarrow$	$c = 9 \downarrow$
Unscrubbed	19.62	12.05	17.04	9.67	19.88	14.78	20.56	17.16	11.53	11.44
Retrain	152.39	139.62	17.39	9.57	20.05	14.65	20.19	17.85	11.63	10.85
SO (α=0.1)	20.85	11.72	18.74	12.14	22.53	16.44	24.17	17.56	13.59	15.55
SO (α=0.3)	33.33	22.87	20.22	12.05	24.12	21.00	26.18	21.57	14.24	15.00
SO (α=0.5)	175.17	77.46	90.30	25.43	64.28	57.89	55.07	51.68	40.77	37.94
EraseDiff (Ours)	256.27	294.08	29.61	22.10	28.65	27.68	35.59	23.93	21.24	24.85

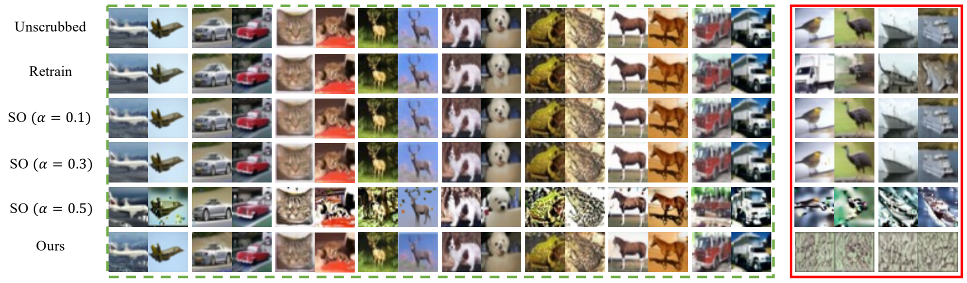

Figure 17: Images generated by conditional DDIM from Tab. 4. Images in the green dashed box are generated by conditioning on the remaining labels $\mathcal{C}_r$ and those in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$.

Table 5: Results on CIFAR-10 trained with conditional DDIM, compared with separate optimization (Two-steps, denoted as TS). TS will perform E_1 epochs for the first step (ie., NegGrad), then perform E_3 epochs for the second step (ie., relearn using $\mathcal{D}_r$). Generated examples are shown in Fig. 18. TS cannot completely erase the influence of $\mathcal{D}_f$ on the model.

Method	FID over forgetting classes		FID over remaining classes							
	$c = 2 \uparrow$	$c = 8 \uparrow$	$c = 0 \downarrow$	$c = 1 \downarrow$	$c = 3 \downarrow$	$c = 4 \downarrow$	$c = 5 \downarrow$	$c = 6 \downarrow$	$c = 7 \downarrow$	$c = 9 \downarrow$
Unscrubbed	19.62	12.05	17.04	9.67	19.88	14.78	20.56	17.16	11.53	11.44
Retrain	152.39	139.62	17.39	9.57	20.05	14.65	20.19	17.85	11.63	10.85
TS (step 1, $E_1 = 10$)	292.35	297.94	276.75	296.48	313.51	317.70	310.61	326.49	311.01	296.05
TS (step 2, $E_2 = 50$)	73.29	100.72	73.78	78.23	67.21	70.79	72.85	56.41	74.13	82.86
TS (step 2, $E_2 = 100$)	30.88	26.56	21.64	13.96	24.19	19.14	26.32	19.44	15.49	17.38
EraseDiff (Ours)	256.27	294.08	29.61	22.10	28.65	27.68	35.59	23.93	21.24	24.85

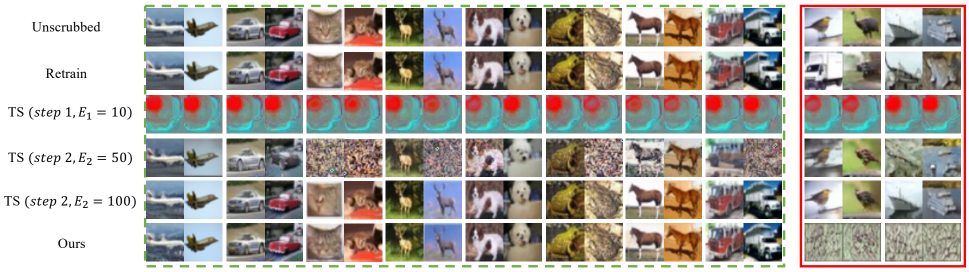
Unscrubbed
Retrain
TS ($step$ 1, E_1 = 10)
TS ($step$ 2, E_2 = 50)
TS ($step$ 2, E_2 = 100)
Ours

Figure 18: Images generated by conditional DDIM from Tab. 5. Images in the green dashed box are generated by conditioning on the remaining labels $\mathcal{C}_r$ and those in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$.

Table 6: Results on CIFAR-10 trained with conditional DDIM. $\mathcal{D}_r^{'}$: *EraseDiff* apply generated images to be the remaining data for the unlearning process.

Method	FID over forgetting classes		FID over remaining classes							
	$c = 2 \uparrow$	$c = 8 \uparrow$	$c = 0 \downarrow$	$c = 1 \downarrow$	$c = 3 \downarrow$	$c = 4 \downarrow$	$c = 5 \downarrow$	$c = 6 \downarrow$	$c = 7 \downarrow$	$c = 9 \downarrow$
Unscrubbed	19.62	12.05	17.04	9.67	19.88	14.78	20.56	17.16	11.53	11.44
Retrain	152.39	139.62	17.39	9.57	20.05	14.65	20.19	17.85	11.63	10.85
Finetune	31.64	21.22	20.49	12.38	23.47	17.80	25.51	18.23	14.43	16.09
NegGrad	322.67	229.08	285.25	290.57	338.49	290.23	312.44	339.43	320.63	278.03
BlindSpot	349.60	335.69	228.92	181.88	288.88	252.42	242.16	278.62	192.67	195.27
EraseDiff ($\mathcal{D}_r^{'}$)	298.60	311.59	33.01	24.09	34.23	34.79	45.51	38.05	24.59	28.10
EraseDiff (Ours)	256.27	294.08	29.61	22.10	28.65	27.68	35.59	23.93	21.24	24.85

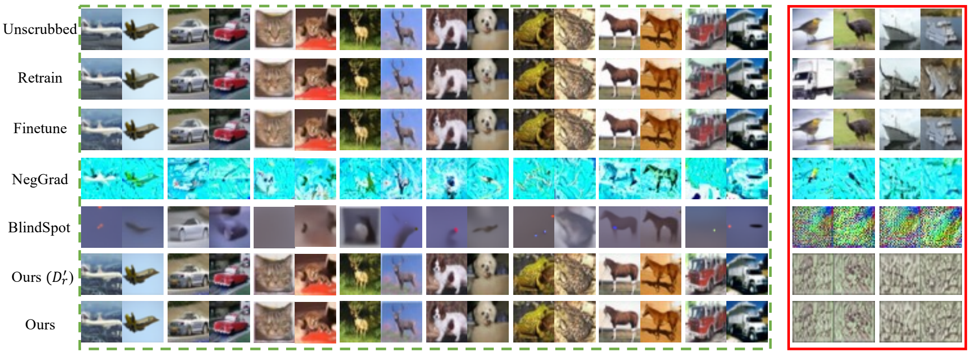

Figure 19: Images generated by conditional DDIM from Tab. 6. Images in the green dashed box are generated by conditioning on the remaining labels $\mathcal{C}_r$ and those in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$.

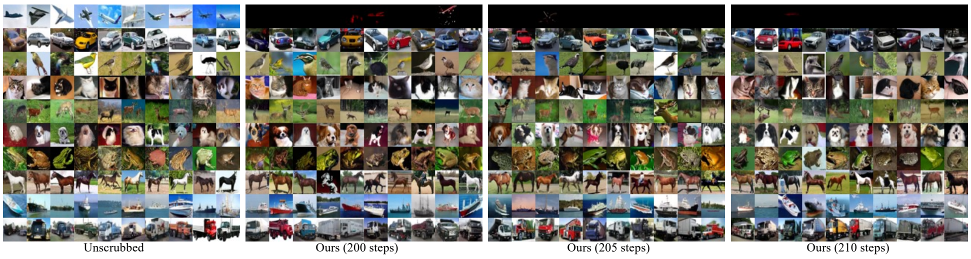

Figure 20: Conditional DDPM on CIFAR-10 when forgetting samples belonging to label '0'. Following Heng & Soh (2023) and using the well-trained model from Heng & Soh (2023), our method achieves a FID score of 8.93 at 210 steps, 8.83 at 205 steps, and 8.90 at 200 steps.

Table 7: Results of *EraseDiff* on CIFAR-10 with conditional DDIM. For each class, the FID score is computed over 5K generated images. Each row's forgetting classes are highlighted in orange.

$\mathcal{C}_f$	$c = 0$	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$	$c = 6$	$c = 7$	$c = 8$	$c = 9$
$c = 2$	26.60	17.04	295.48	27.07	32.32	30.45	28.58	19.77	17.60	20.67
$c = 2, 8$	29.61	22.10	256.27	28.65	27.68	35.59	23.93	21.24	294.08	24.85
$c = 5, 6$	30.03	16.51	29.37	33.50	22.12	321.09	302.01	20.06	21.94	21.10
$c = 2, 5, 8$	24.02	15.59	288.01	26.06	19.31	296.79	21.25	15.87	206.61	21.56

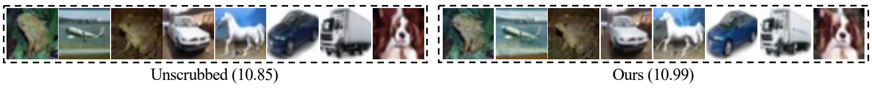

Figure 21: Unconditional DDPM on CIFAR-10 when forgetting randomly selected samples. 50K generated images by our scrubbed model have an FID score of 10.99, and the unscrubbed model has an FID score of 10.85.

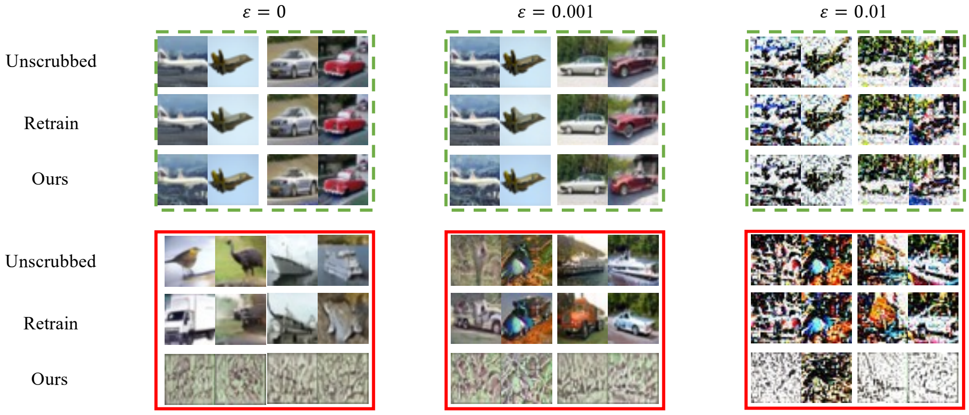

Figure 22: Generated examples when objected to FGSM Goodfellow et al. (2014) attack. Images in the green dashed box are generated by conditioning on the remaining labels $\mathcal{C}_r$ and those in the red solid box are generated by conditioning on the forgetting classes $\mathcal{C}_f$. With the step size ϵ increases, the quality of the generated images would decrease for all models. Note that our scrubbed model still doesn't contain information about the forgetting classes $\mathcal{C}_f$ in this setting.

Table 8: Results on UTKFace with conditional DDIM. SO: simultaneously optimizing $\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_r) - \alpha \mathcal{L}(\boldsymbol{\theta}; \mathcal{D}_f)$. Generated examples are shown in Fig. 23. *EraseDiff* achieves a better trade-off between erasing the influence of $\mathcal{D}_f$ and preserving model utility over $\mathcal{D}_r$ than SO.

Method	FID over forgetting classes	FID over remaining classes		
	$c = 3 \uparrow$	$c = 0 \downarrow$	$c = 1 \downarrow$	$c = 2 \downarrow$
Unscrubbed	8.87	7.37	11.28	9.72
SO (α=0.05)	216.35	14.09	15.73	15.62
SO (α=0.10)	417.90	22.00	24.34	22.60
EraseDiff (Ours)	330.33	8.08	13.52	12.37

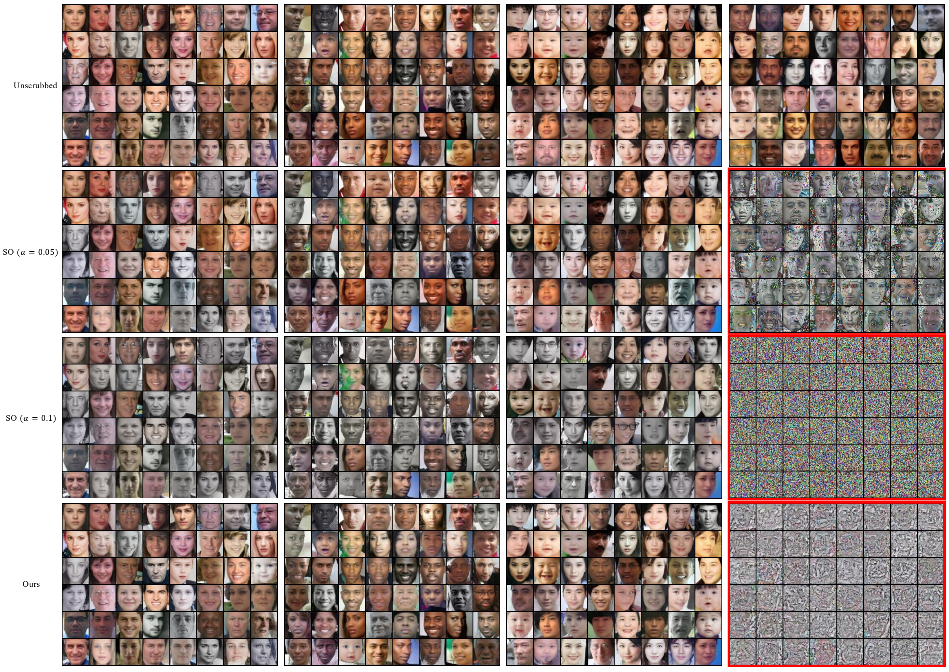

Figure 23: Images generated with conditional DDIM when unlearning the Indian celebrities from Tab. 8 (Top to Bottom: generated examples of the unscrubbed model, those of the model scrubbed by SO ($\alpha = 0.05$), those of the model scrubbed by SO ($\alpha = 0.10$), and those by our scrubbed model, respectively). Images in the red solid box are generated by conditioning on $\mathcal{C}_f$, others are generated by conditioning on $\mathcal{C}_r$. SO ($\alpha = 0.05$) cannot completely erase information about $\mathcal{C}_f$ and SO ($\alpha = 0.10$) has a significant drop in the quality of generated images.

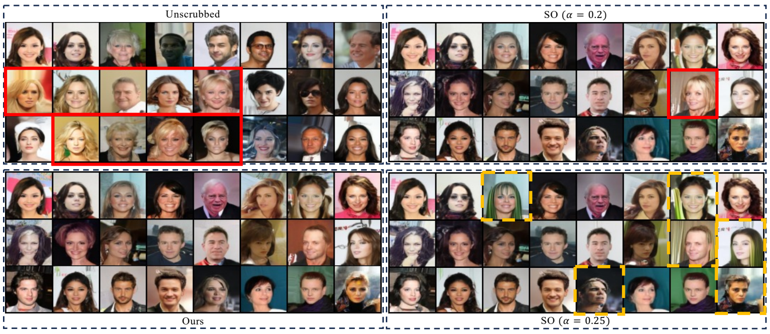

Figure 24: Images generated by unconditional DDIM on CelebA, focusing on the removal of the blond hair attribute. Images in the red solid box present the attribute of blond hair and those in the yellow dashed box display distortions. The FID score of the unscrubbed model, that of ours, that of SO ($\alpha = 0.2$), and that of SO ($\alpha = 0.25$) are 8.95, 10.70, 12.35, and 17.21 respectively.

B Detailed formulation

Our formulation is

$$\boldsymbol{\theta}^* := \arg\min_{\boldsymbol{\theta}} \mathcal{F}(\boldsymbol{\theta}), \quad \text{where } \mathcal{F}(\boldsymbol{\theta}) = \mathcal{L}(\mathrm{Alg}(\boldsymbol{\theta}, \mathcal{D}_f), \mathcal{D}_r) := \mathcal{F}(\boldsymbol{\theta}, \mathcal{D}_r) + \lambda \hat{f}(\boldsymbol{\theta}, \mathcal{D}_f), \tag{10}$$

We consider

$$h(\boldsymbol{\theta}, \phi) := \mathcal{F}\left(\boldsymbol{\theta}, \mathcal{D}_r\right) + \lambda \hat{f}(\boldsymbol{\theta}, \phi), \tag{11}$$

where $\hat{f}(\boldsymbol{\theta}, \phi) = f\left(\phi, \mathcal{D}_f\right) - f\left(\boldsymbol{\theta}, \mathcal{D}_f\right)$, then

$$\mathrm{Alg}\left(\boldsymbol{\theta}, D_f\right) = \phi^*(\boldsymbol{\theta}) = \mathrm{argmin}_\phi\, h(\boldsymbol{\theta}, \phi) = \arg\min_{\phi} \hat{f}(\boldsymbol{\theta}, \phi) = \arg\min_{\phi \mid \boldsymbol{\theta}} f\left(\phi, \mathcal{D}_f\right), \tag{12}$$

where $\phi \mid \boldsymbol{\theta}$ means ϕ is started from $\boldsymbol{\theta}$ for its updates. Finally, we reach

$$\min_{\boldsymbol{\theta}} \min_{\phi \in \phi^*(\boldsymbol{\theta})} h(\boldsymbol{\theta}, \phi) = \min_{\boldsymbol{\theta}} \min_{\phi \in \mathrm{Alg}(\boldsymbol{\theta}, D_f)} h(\boldsymbol{\theta}, \phi). \tag{13}$$

We can characterize the solution of our algorithm as follows:

Theorem 1 (Pareto optimality)**.** *The stationary point obtained by our algorithm is Pareto optimal.*

Proof. Let θ^* be the solution to our problem. Because given the current θ_s, in the inner loop, we find ϕ_s^K to minimize $\hat{f}(\phi, \mathcal{D}_f) = f(\theta_s, \mathcal{D}_f) - f(\phi, \mathcal{D}_f)$. Assume that we can update in sufficient number of steps K so that $\phi_s^K = \phi^*(\theta_s) = \arg\min_{\phi \mid \theta_s} \hat{f}(\phi, \mathcal{D}_f) = \arg\min_{\phi \mid \theta_s} f(\phi, \mathcal{D}_f)$. Here $\phi \mid \theta_s$ means ϕ is started from θ_s for its updates.

The outer loop aims to minimize $\mathcal{F}(\theta, \mathcal{D}_f) + \lambda \hat{f}(\phi^*(\theta), \mathcal{D}_r)$ whose optimal solution is θ^*. Note that $\hat{f}(\phi^*(\theta), \mathcal{D}_r) \geq 0$ and it decreases to 0 for minimizing the above sum. Therefore, $\hat{f}(\phi^*(\theta^*), \mathcal{D}_r) = 0$. This further means that $\hat{f}(\theta^*, \mathcal{D}_f) = \hat{f}(\phi(\theta^*), \mathcal{D}_f)$, meaning that θ^* is the current optimal solution of $\hat{f}(\phi, \mathcal{D}_f)$ because we cannot update further the optimal solution. Moreover, we have θ^* as the local minima of $\mathcal{F}\left(\theta, \mathcal{D}_f\right)$ because $\hat{f}\left(\phi^*\left(\theta^*\right), \mathcal{D}_f\right) = 0$ and we consider a sufficiently small vicinity around θ^*. □