Appendix

A. BLO framework of WATER4MU for image generation

In the context of prompt-wise forgetting in image generation, the BLO problem for WATER4MU can be also cast as (5):

$$\begin{aligned} \underset{\psi,\phi}{\text{minimize}} \quad & \hat{\mathcal{L}}(\psi,\phi,\boldsymbol{\theta}_{\mathrm{u}}(\psi)) \\ \text{subject to} \quad & \boldsymbol{\theta}_{\mathrm{u}}(\psi) = \arg\min_{\boldsymbol{\theta}} \ \mathcal{L}_{\mathrm{mu}}(\boldsymbol{\theta};\hat{\mathcal{D}}_{\mathrm{f}},\hat{\mathcal{D}}_{\mathrm{r}}), \end{aligned} \tag{A1}$$

In the lower-level optimization, we first obtain the watermarked dataset for unlearning. Then, we extend the use of Random Label (RL) to the image generation context for unlearning concept c. Also, to maintain the generation capability of the model, we introduce the regularization loss on the watermarked retain set. Finally, we can obtained an unlearned model $\boldsymbol{\theta}_{\mathrm{u}}$:

$$\begin{aligned} \boldsymbol{\theta}_{\mathrm{u}}(\psi) = \arg\min_{\boldsymbol{\theta}} \ & \mathbb{E}_{\mathbf{x}\in\hat{\mathcal{D}}_{\mathrm{f}},t,\boldsymbol{\epsilon}}[||\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(x_t|c') - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(x_t|c)||_2^2] \\ & + \mathbb{E}_{\mathbf{x}\in\hat{\mathcal{D}}_{\mathrm{r}},t,\boldsymbol{\epsilon}}[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(x_t|c_r)||_2^2], \end{aligned} \tag{A2}$$

where $c' \neq c$, c_r is the prompt of $x \in \hat{\mathcal{D}}_{\mathrm{r}}$, $\hat{\mathcal{D}}_{\mathrm{f}}$ (or $\hat{\mathcal{D}}_{\mathrm{r}}$) denotes the watermarked forget (or retain) dataset, $\boldsymbol{\theta}$ is the pretrained generative model and $\boldsymbol{\beta}$ is a regularization parameter.

The design of the upper-level optimization follows (4):

$$\begin{aligned} \hat{\mathcal{L}}(\psi,\phi,\boldsymbol{\theta}_{\mathrm{u}}(\psi)) := & \underbrace{\mathcal{L}_{\mathrm{mu}}(\boldsymbol{\theta}_{\mathrm{u}}(\psi);\mathcal{D}_{\mathrm{f}},\mathcal{D}_{\mathrm{r}})}_{\text{(a) Unlearning validation}} \\ & + \underbrace{\mathcal{L}_{\mathrm{wm}}(\psi,\phi;\mathbf{m},\mathcal{D}_{\mathrm{f}}\cup\mathcal{D}_{\mathrm{r}})}_{\text{(b) Watermarking validation}}, \end{aligned} \tag{A3}$$

where ℓ_{mu} is defined in (A2) and $\mathcal{L}_{\mathrm{mu}}(\boldsymbol{\theta}_{\mathrm{u}}(\psi);\mathcal{D}_{\mathrm{f}},\mathcal{D}_{\mathrm{r}})$ is to validate the lower-level unlearned model $\boldsymbol{\theta}_{\mathrm{u}}(\psi)$ on the unwatermarked dataset, and ℓ_{wm} is the training loss of the watermarking network.

B. Additional Experiment Setup

B.1. WATER4MU for image classification

For the exact unlearning method Retrain, the training process comprises 182 epochs, utilizing the SGD optimizer with a cosine-scheduled learning rate initially set to 0.1. For FT, the unlearning process takes 10 epochs, during which the optimal learning rate is searched within the range of $[10^{-3}, 10^{-1}]$. For GA, the unlearning process spans 5 epochs with the interval $[10^{-5}, 10^{-3}]$. Regarding the method Sparse, the unlearning-enabled model updating process also takes 10 epochs, searching the optimal sparse ratio in the range $[10^{-6}, 10^{-4}]$ and exploring learning rate within $[10^{-3}, 10^{-1}]$. Finally, for IU, the parameter α (associated with the WoodFisher Hessian Inverse approximation) is searched within the range $[1, 20]$.

B.2. WATER4MU for image generation

We follow the settings in the UNLEARNCANVAS benchmark, selecting 20 objects and 50 styles for unlearning. The $\boldsymbol{\beta}$ is set at 0.5, with a batch size of 1. The sampling settings involve the use of DDPM, 100 time steps, and a conditional scale of 7.5.

C. Additional Experiment Results

C.1. Ablation study

The computational costs of WATER4MU. We measure the run-time efficiency (RTE) of applying an MU method, *i.e.*, its computation time. In Tab. A1, we compared the RTE of different methods with and without using WATER4MU on (CIFAR-10, RESNET-18). As we can see, the introduction of WATER4MU does not hamper the computation efficiency, highlighting its practicality.

Table A1. Performance of RTE (min) of different unlearning methods under unwatermarked forget/retain sets (Original) and WATER4MU-induced watermarked forget/retain sets on (CIFAR-10, RESNET-18) for class-wise forgetting.

MU	Retrain	GA	FT	Sparse	IU
Original	42.35	0.25	2.50	2.52	3.25
WATER4MU	49.93	0.30	3.07	3.09	3.98

Choice of λ in diagonalization approximation of the Hessian matrix. We next examine the hyperparameter λ in Hessian's diagonalization approximation in (8) for WATER4MU. In our experiments, the default setting is $\lambda = 10^{-2}$. We conduct a more detailed examination of λ in Fig. A1. We can observe that $\lambda = 10^{-2}$ is a reasonable option, and higher or lower value of λ would reduce the effectiveness of WATER4MU.

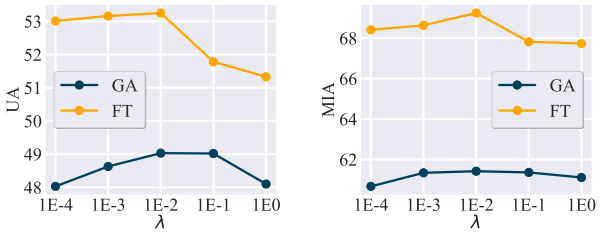

Figure A1. Unlearning effectiveness (in terms of UA and MIA) of GA and FT against the choice of λ in WATER4MU for class-wise forgetting under (CIFAR-10, RESNET-18).

Decoding performance of WATER4MU. While WATER4MU enhances the unlearning effectiveness, the decoding performance of WATER4MU as a watermarking method should also be maintained. We then use BER (Bit Error Rate) to measure the performance of WATER4MU to decode

Table A2. Performance of different unlearning methods under unwatermarked forget/retain sets (Orignal) and WATER4MU-induced watermarked forget/retrain sets on (CIFAR-100, RESNET-18) and (SVHN, RESNET-18) for class-wise forgetting.

Metric	Retrain			GA			FT		
	Original	WATER4MU	Diff	Original	WATER4MU	Diff	Original	WATER4MU	Diff
Class-wise forgetting, RESNET-18, CIFAR-100									
UA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$83.67_{\pm 0.78}$	$90.46_{\pm 0.43}$	6.33▲	$24.73_{\pm 2.63}$	$29.56_{\pm 2.78}$	4.83▲
MIA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$91.51_{\pm 2.66}$	$98.22_{\pm 0.12}$	6.71▲	$45.61_{\pm 0.3.31}$	$50.36_{\pm 2.09}$	4.75▲
RA	$99.98_{\pm 0.01}$	$98.87_{\pm 0.05}$	1.11▼	$91.56_{\pm 0.54}$	$88.32_{\pm 1.18}$	3.24▼	$99.18_{\pm 0.26}$	$99.03_{\pm 0.72}$	0.15▼
TA	$74.43_{\pm 0.23}$	$72.89_{\pm 0.13}$	0.16▼	$65.79_{\pm 0.69}$	$64.61_{\pm 0.19}$	1.18▼	$74.33_{\pm 0.14}$	$72.50_{\pm 0.27}$	1.83▼
Class-wise forgetting, RESNET-18, SVHN									
UA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$83.29_{\pm 0.42}$	$89.07_{\pm 0.13}$	5.78▲	$16.98_{\pm 4.60}$	$29.84_{\pm 3.75}$	12.86▲
MIA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$98.23_{\pm 0.38}$	$99.61_{\pm 0.09}$	1.38▲	$85.57_{\pm 2.32}$	$90.10_{\pm 0.69}$	4.53▲
RA	$100.00_{\pm 0.00}$	$99.96_{\pm 0.02}$	0.04▼	$99.51_{\pm 0.21}$	$98.16_{\pm 0.79}$	1.35▼	$100.00_{\pm 0.00}$	$98.64_{\pm 0.56}$	1.36▼
TA	$95.82_{\pm 0.04}$	$94.86_{\pm 0.07}$	0.96▼	$95.33_{\pm 0.19}$	$94.16_{\pm 0.08}$	1.17▼	$95.95_{\pm 0.18}$	$95.76_{\pm 0.22}$	0.19▼

Table A3. Performance of different unlearning methods under unwatermarked forget/retain sets (Orignal) and WATER4MU-induced watermarked forget/retrain sets on (CIFAR-10, SWIN TRANSFORMER) and (CIFAR-10, RESNET-50) for class-wise forgetting.

Metric	Retrain			GA			FT		
	Original	WATER4MU	Diff	Original	WATER4MU	Diff	Original	WATER4MU	Diff
Class-wise forgetting, SWIN TRANSFORMER, CIFAR-10									
UA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$45.96_{\pm 0.67}$	$55.43_{\pm 0.50}$	9.47▲	$94.89_{\pm 1.56}$	$99.80_{\pm 0.17}$	4.91▲
MIA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$58.40_{\pm 1.03}$	$64.82_{\pm 1.89}$	6.42▲	$97.86_{\pm 1.21}$	$99.68_{\pm 0.10}$	1.82▲
RA	$100.00_{\pm 0.01}$	$99.71_{\pm 0.07}$	0.29▼	$99.85_{\pm 0.26}$	$99.72_{\pm 0.17}$	0.13▼	$95.01_{\pm 1.26}$	$93.45_{\pm 0.92}$	1.56▼
TA	$85.63_{\pm 2.10}$	$85.34_{\pm 1.27}$	0.29▼	$85.67_{\pm 1.04}$	$85.32_{\pm 1.41}$	0.35▼	$80.58_{\pm 2.16}$	$78.32_{\pm 1.67}$	2.26▼
Class-wise forgetting, RESNET-50, CIFAR-10									
UA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$36.76_{\pm 1.28}$	$50.00_{\pm 0.79}$	13.24▲	$42.16_{\pm 2.63}$	$59.83_{\pm 2.11}$	17.67▲
MIA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$59.60_{\pm 0.65}$	$77.66_{\pm 1.03}$	18.06▲	$58.35_{\pm 1.53}$	$67.75_{\pm 1.93}$	9.40▲
RA	$100.00_{\pm 0.00}$	$100.00_{\pm 0.00}$	0.00-	$99.84_{\pm 0.08}$	$99.53_{\pm 0.19}$	0.31▼	$98.76_{\pm 0.14}$	$99.15_{\pm 0.23}$	0.39▲
TA	$94.13_{\pm 1.20}$	$94.02_{\pm 0.98}$	0.11▼	$93.67_{\pm 0.72}$	$93.12_{\pm 1.57}$	0.55▼	$90.36_{\pm 1.63}$	$90.96_{\pm 1.49}$	0.60▲

Table A4. Number of nude body parts detected by Nudenet on I2P dataset with threshold 0.6.

Method	Breast	Genitalia	Buttocks	Feet	Belly	Armpits	Total
SD v1.4	229	31	44	42	171	129	646
ESD	**22**	**6**	5	24	31	33	121
FMN	172	17	12	56	116	42	415
UCE	50	14	11	20	55	36	186
WATER4MU	29	15	**5**	**10**	**29**	**21**	**109**

the watermark message $\mathbf{m}$. In our experiments, we set the message length L to 10 by default. We evaluate the decoding performance of HIDDEN used by WATER4MU. We find that its average BER is merely 2.78×10^{-8} compared to 1×10^{-8} using the standard HIDDEN w/o taking into account MU. This indicates that WATER4MU preserves the watermarking network's encoding-decoding capabilities.

C.2. Additional class-wise forgetting results

As an expansion of Tab. 1, Tab. A2 presents the performance of class-wise forgetting with and without the integration of WATER4MU on the additional setups (CIFAR-100,RESNET-18) and (SVHN,RESNET-18). Also, Tab. A3 shows the performance of class-wise forgetting on the setups (CIFAR-10,SWIN TRANSFORMER) and (CIFAR-10,RESNET-50) to further validate the scalability of WATER4MU. Notably, the WATER4MU-integrated unlearning methods demonstrate enhanced unlearning performance across both datasets and models, as evidenced by improvements in UA and MIA metrics. This improvement effectively balances model utility preservation, as reflected in RA and TA scores. Importantly, the gains in unlearning performance outweigh the losses in utility. These observations are consistent with the findings reported in Tab. 1.

C.3. Effect of Watermark Message Selection in WATER4MU.

In **Fig. A2**, we present the performance of the optimized watermark message, *i.e.*, selecting the watermark message most favorable for unlearning, as described in Sec. 5. As shown, using the optimized watermark message in WATER4MU further enhances unlearning effectiveness (UA and MIA) without causing any degradation in model utility (TA and RA). This improvement is particularly notable when compared to the use of random watermark messages in WATER4MU.

We also provide additional results on the impact of watermark message selection on Retrain. Extended from Fig. A2, Fig. A3 presents the performance of the optimized watermark message on the exact unlearning method, Retrain. As we can see, using the optimized watermark message in WATER4MU also enhances unlearning effectiveness of Retrain without causing any degradation in model utility, when compared to the use of random watermark message in WATER4MU.

C.4. Evaluation on image generation safety tasks.

We conduct experiments on the Inappropriate Image Prompts (I2P) dataset. Our evaluation focuses on the erasure of nudity. A total of 4703 images are generated with I2P prompts for

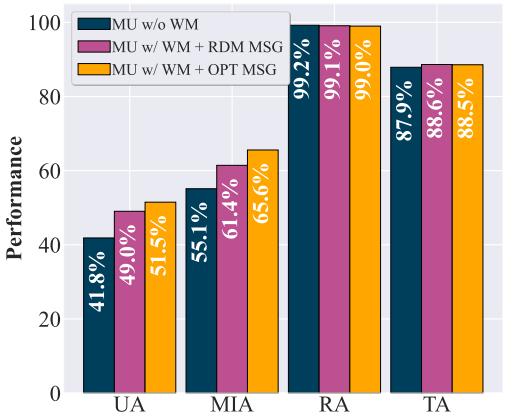

Figure A2. Performance of the optimized watermark message in WATER4MU. We choose GA as the unlearning method and compare the unlearning performance among MU without WATER4MU, MU with WATER4MU and MU with WATER4MU and optimized watermark message.

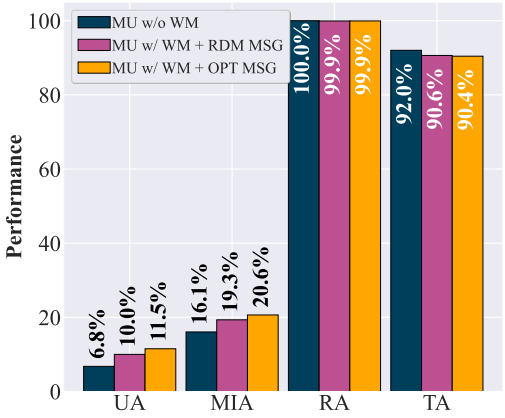

Figure A3. Performance of Retrain using the optimized watermark message in WATER4MU vs. baselines including MU without WATER4MU and MU with WATER4MU (but implemented using random watermark message).

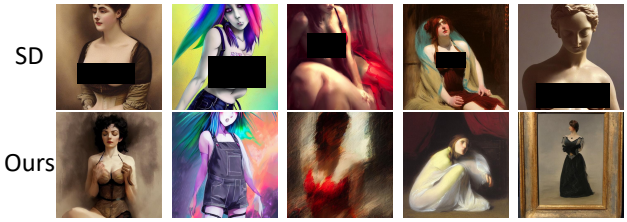

Figure A4. Qualitative results of nudity removing. All prompts originate from I2P dataset. The images in the top row are generated by SD, while the images in the bottom row are generated by MU with WATER4MU.

each model. Then, Nudenet is introduced to detect nude body parts in these images. As shown in Tab. A4, we present the number of detected nude body parts across six categories. WATER4MU achieves the best performance in preventing the generation of nude content, reducing the total instances from 646 to 109. We provide more examples on the erasure of the nudity concept in Fig. A4.

D. Limitations

While WATER4MU demonstrates promising unlearning effectiveness, the introduction of the watermarking network and the computation of higher-order derivatives in the BLO process will both incur additional training overhead. Further work is needed to assess the scalability of WATER4MU to larger datasets and models, particularly in enhancing the efficiency of the bi-level optimization process used for watermarking design.